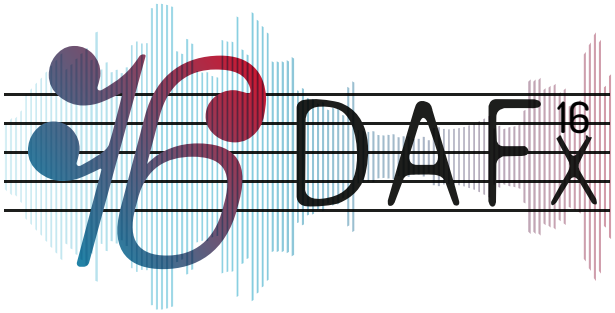
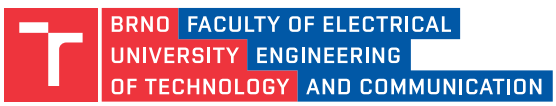


19th International Conference on Digital Audio Effects

Brno, Czech Republic
September 5–9, 2016

http://dafx16.vutbr.cz



Proceedings published by:
Brno University of Technology
The Faculty of Electrical Engineering and Communication

Credits:
Proceedings edited, produced and cover designed by
Pavel Rajmic, František Rund, and Jiří Schimmel
DAFx-16 logotype by Adéla Geislerová

ISSN 2413-6700 (Print)
ISSN 2413-6689 (Online)

www.dafx16.vutbr.cz
www.dafx.de

Welcome to DAFx-16 !

It is our great pleasure to welcome you to Brno to the 19th International Conference on Digital Audio Effects held from Monday September 5 to Thursday September 8, 2016.

The local DAFx community in Brno, Czech Republic was established in 1997 when the Brno University of Technology joined the European research project for co-operation and scientific transfer EU-COST-G6 "Digital Audio Effects". Since that time, we had great opportunities to meet people from the DAFx community around the world at DAFx conferences. In 2011, together with other colleagues from the university, we founded the Signal Processing Laboratory that includes an acoustic signal processing group. One of our challenges is to build and equip the laboratories for R&D in the area of audio signal processing and to attract more students to the fields of digital audio effects, sound analysis and synthesis, and acoustics. Our great success was the launch of a new, interdisciplinary bachelor study programme in Audio Engineering in 2013 and of a master study programme in 2016, together with the Janáček Academy of Music and Performing Arts in Brno. We are deeply honored of having the opportunity to host DAFx-16 at our institution.

DAFx-16 offers a scientific program intermingled with a concert and social events. The first day of the conference consists of three tutorials. The first tutorial and musical demonstration given by Václav Peloušek and Lennart Schierling is on the instrument design at the Bastl Instruments company. The second tutorial given by Jaromír Mačák offers an overview of basic techniques used in real-time virtual analog modeling. The third tutorial given by Jukka Pätynen and Sakari Tervo from the Aalto University School of Science is on the latest advances in investigating the spatial, spectral, and temporal structure of the acoustics of enclosures with visual techniques.

The DAFx program will also feature three distinguished speakers who will give their keynote speeches on Tuesday, Wednesday, and Thursday. The first keynote speech given by Doz. Dr. Peter Balazs from the Acoustics Research Institute in Vienna is focused on aspects of short-time audio signal analysis and synthesis, and a number of interesting questions will be answered using the general theory of frames, with applications to audio signal processing. The second keynote speech given by Michael Hlatky will provide an insight into how the people working in Native Instruments design the hardware, develop the software, decide upon features, and what lessons they learned in more than 20 years after NI introduced the first modular synthesizer for a desktop computer environment. The third keynote speech given by D. Sc. Sakari Tervo from the Aalto University School of Science is on an approach to the analysis and synthesis of the sound field measured with a microphone array via parametric models.

The proceedings contain complete manuscripts of all peer-reviewed papers presented at the conference. Each of the 56 submitted papers was evaluated by 3 to 6 reviewers, and finally, 43 contributions were selected for presentation from Tuesday to Thursday: 29 oral presentations and 14 poster presentations. The oral presentations are organized in 8 sessions, covering topics such as time-frequency representation of audio signals, virtual analog, sound synthesis, physical modeling, audio and music analysis, wave digital filters, and spatial audio. The posters are not organized into topics to favor exchanges between scientists of the same field of research. There are seven poster presentations both on Tuesday and Wednesday, including two short announcement sessions in the auditorium.

Besides the scientific program, social events will be held during the evenings. The welcome reception on Monday evening will take place in the Mechanic Music Parlor of the Technical Museum in Brno. The reception will include a guided tour of the exposition and the presentation of musical instruments. The DAFx concert will be held on Tuesday evening at the Orlí Street Theatre, a new theatre venue of the Janáček Academy of Music and Performing Arts in Brno. Before the concert, a short guided tour to the Brno city centre will be organized. The social program will be concluded with a conference dinner on Wednesday at the winery "U Kapličky" surrounded with wine yards, south from Brno.

We would like to thank all the people that helped us make this year's conference happen, including the local organization team, all the reviewers from the DAFx-16 program committee and the DAFx board members. Warm thanks go to the organizers of DAFx-14 and DAFx-15, who were very helpful by sharing their experience. Special thanks go to the DAFx annual sponsors – Native Instruments GmbH, Soundtoys Inc., Ableton AG, and iZotope, Inc., and to this year's new sponsors – Steinberg Media Technologies GmbH and AVT Group a.s. We would also like to thank the supporting institutions – Brno University of Technology, Applied Sciences Open Access Journal, Technical Museum in Brno, Orlí Street Theatre, and Moravia Convention Bureau. But most of all we would like to thank the DAFx community who makes these conferences possible by contributing great scientific work and enlightened discussions whenever we meet. The biggest acknowledgment therefore goes to you, the authors, researchers and participants of this conference.

The DAFx-16 conference committee

DAFx Board

DAFx-16 Local Organizing Committee

Jiří Schimmel (Brno University of Technology, General Chair)
Pavel Rajmic (Brno University of Technology, Paper Chair)
František Rund (Czech Technical University in Prague, Paper Chair)
Jan Karásek (BioVendor Instruments, Social Events Coordinator)
Václav Mach (Brno University of Technology, Technical Coordinator)

DAFx-16 Program Committee

Stefan Bilbao
Philippe Depalle
Giovanni De Poli
Myriam Desainte-Catherine
Sascha Disch
Gianpaolo Evangelista
Robert Höldrich
Pierre Hanna
Jean-Marc Jot
Victor Lazzarini
Sylvain Marchand
Damian Murphy
Søren Nielsen
Markus Noisternig
Luis Ortiz Berenguer
Geoffroy Peeters
Rudolf Rabenstein
Pavel Rajmic
Josh Reiss
Jøran Rudi
František Rund
Jiri Schimmel
Xavier Serra
Julius O. Smith
Alois Sontacchi
Todor Todoroff
Jan Tro
Vesa Välimäki
Udo Zölzer

DAFx-16 (Sub)reviewers

Alexander Adami
Areti Andreopoulou
Paolo Annibale
Federico Avanzini
Miroslav Balik
Balázs Bank
Yann Bayle
Ilker Bayram
Edgar Berdahl
Dmitry Bogdanov
Sebastian Böck
Elliot Kermit-Canfield
Thibaut Carpentier
Rafael C. D. de Paiva
Stefania Cecchi
Jean-Michaël Celerier
Brecht De Man
Olivier Derrien
Charlotte Desvages
Christian Dittmar
Simon Dixon
Michele Ducceschi
Ross Dunkel
Monika Dörfler
Felix Eichas
Fabian Esqueda
John Ffitch
Marco Fink
Federico Fontana
Matthias Frank
Simone Füg
Volker Gnann
Brian Hamilton
Reginald Harrison
Nicki Holighaus
Martin Holters
Petr Honzík
Libor Husnik
Yukio Iwaya
Gavin Kearney
Minje Kim
Johannes Klein
Zbynek Koldovsky
Jaroslav Koton
Matthieu Kowalski
Sebastian Kraft
Thomas Lidy
Alexey Lukin
Roland Maas
Jaromir Macak
Piotr Majdak
Jiri Malek
Petr Marsalek
Jiri Mekyska
Rémi Mignot
Dave Moffat
Thibaud Necciari
Maciej Niedźwiecki
Antonin Novak
Michael Olsen
Darian Onchis
Julian Parker
Alfonso Perez
Hannes Pomberger
Anna Pribilova
Zdenek Prusa

Maximilian Rest
Benjamin Ricaud
Kamil Riha
Antonio Roda
Samuel Rodriguez
Axel Roebel
David M. Ronan
Jean-Luc Rouas
Mark Sandler
Augusto Sarti
Sigurd Saue
Diemo Schwarz
Tim Schwerdtfeger
Maximilian Schäfer
Rod Selfridge
Esben Skovenborg
Zdenek Smekal
Julius Stroffek
Peter Svensson
Petr Sysel
Sakari Tervo
Petr Tichavsky
Joseph Timoney
Maarten Van Walstijn
Radoslav Vargic
Jozef Vavrek
Gino Angelo Velasco
Vaclav Vencovsky
Olivier Warusfel
Jeremy Wells
Kurt Werner
Hagen Wierstorf
William Wilkinson
Aaron Wishnick
Markus Zaunschirm

Contents

Poster Session 2

Oral Session 6: Audio and Music Analysis

Oral Session 7: Wave Digital Filters

Oral Session 8: Spatial Audio

Keynotes

Peter Balazs: Frames in audio processing: What you use, but might not know
Given a certain number of sampling points, can it be useful to represent them with a larger collection of points/values? If not, why are spectrograms usually using overlapping windows? Given a particular analysis filter bank, when and how can a synthesis procedure be found that enables perfect reconstruction? What are the conditions for that? Is a quadrature-mirror condition the only way? How can a time-varying filter be implemented by directly manipulating the time-frequency coefficients? What properties do such time-frequency filters have? All those (and similar) questions will be answered by using the theory of frames and its application to audio signal processing.

Michael Hlatky: Design at Native Instruments
Since Native Instruments introduced Generator, the first modular synthesizer for a desktop computer environment more 20 years ago, the company has released a large variety of hardware and software to perform and produce music. This talk will give an insight into how we—the people working at NI—design the hardware, develop the software, decide upon features and what lessons we learned.

Sakari Tervo: Parametric Spatial Room Impulse Response Analysis and Synthesis: A High-Resolution Approach
The analysis of room acoustics is of great interest in subjective and objective studies of acoustic spaces. Often, the goal in room acoustic studies is to explain the subjective experience of sound, for example, speech clarity or bassiness, with the objective measurements of the sound field. In order to describe a sound field spatially, a microphone array impulse response measurement is required. This keynote lecture presents an approach to the analysis and synthesis of the sound field measured with a microphone array via parametric models. Estimation methods of the parameters in the model, and the detection of which model fits the data the best are described. Pros and cons of the parametric approach are discussed and examples with some commercially available microphone arrays are described.

Tutorials

Jaromír Mačák: Analog effects modeling—new ways to get old sounds

The term virtual analog effects has been known in digital audio effects community for several years. But until recently, this type of audio effects have been accepted by a wider range of musicians and end users as adequate equivalent to classic analog audio effects. This is due to a significant improvement of the methods and algorithms for real-time digital simulation of analog circuits. This tutorial will give an overview of basic techniques used in virtual analog modeling based either on deep analysis of the circuit schematic or measurement of analog audio effect. Pros and cons of both approaches will be mentioned as well as real examples using these techniques with attention for real-time implementation of these algorithms.

Jukka Pätynen & Sakari Tervo: Detailed Analysis of Room Acoustics by Spatio-temporal Methods

The analysis of sound fields in rooms has been under interest for a long time. Many traditional methods aim to describe the room-acoustic effect in terms of single number values. However, numeric parameters often fail in communicating the multi-dimensional effects. The introduction of compact microphone arrays have enabled an increasingly detailed analysis of the sound field. This tutorial will present the latest advances in investigating the spatial, spectral, and temporal structure of the acoustics of enclosures with visual techniques. These methods are demonstrated with a recently published, freely available analysis toolbox.

Václav Peloušek & Lennart Schierling: Instrument Design Upside Down with Bastl Instruments

Emulating inherently digital artefacts with analog technology? Sounds created by running sound processing on processors virtually unable to render them? This Bastl [local term for hack] mindset as a design approach for musical instruments and tools has been the key for Bastl Instruments main developer Václav Peloušek to creating a range of desktop hardware instruments and eurorack modules. Analog and digital circuits running at the edge of collapse to work in harmony as part of digital-analog-mechanical hybrid systems. Lennart Schierling is the main developer of Thyme—robot operated digital tape machine—very universal sequencable hardware DSP processing unit to be released soon by Bastl.

REAL-TIME AUDIO VISUALIZATION WITH REASSIGNED NON-UNIFORM FILTER BANKS

*Zdenek Prusa, Nicki Holighaus**

Acoustics Research Institute,
Austrian Academy of Sciences
Vienna, Austria
{zdenek.prusa,nicki.holighaus}@oeaw.ac.at

ABSTRACT

Filter banks, both uniform and non-uniform, are widely used for signal analysis and processing. However, the application of a time-frequency localized filter inevitably causes some amount of spectral and temporal leakage that, simultaneously, cannot be arbitrarily reduced. Reassignment is a classical procedure to eliminate this leakage in short-time Fourier spectrograms, thereby providing a sharper, more exact time-frequency domain signal representation. The reassignment technique was recently generalized to general filter banks, opening new possibilities for its application in signal analysis and processing. We present here the very first implementation of filter bank reassignment in a real-time analysis setting, more specifically as visualization in a basic audio player application. The visualization provides a low delay moving spectrogram with respect to virtually any time-frequency filter bank by interfacing the C backend of the LTFAT open-source toolbox for time-frequency processing. Low delay is achieved by blockwise processing, implemented with the JUCE C++ Library.

1. INTRODUCTION

Time-frequency representations, be it in the form of the short-time Fourier transform (STFT) [1, 2], windowed MDCT [3] or (non-uniform) filter banks [4, 5], are crucial tools for many signal analysis and processing applications. In particular, their squared magnitude, the *spectrogram* is frequently used to determine the local frequency content of an analyzed signal. The filtering process, by convolution with a time- and frequency-localized filter, introduces spectral and temporal *leakage* in the representation. In other words, a perfectly frequency-(time-)localized signal will be represented with a certain spread over several frequency bands (time positions), depending on the filter shape. This smoothing effect is subject to Heisenberg's uncertainty inequality and thus cannot be arbitrarily reduced.

In an attempt to overcome this smoothing effect, various alternative time-frequency representations have been proposed. The quadratic time-frequency representations in Cohen's class [6, 7], for example, are given by the Wigner-Ville distribution (WVD) [8, 9] convolved with a smoothing kernel. While the spectrogram suffers from a large amount of smoothing, the WVD produces undesirable interference terms. Cohen's class representations allow the choice of a kernel that places them somewhere between these two extrema. Consequently, representations of this kind are often designed with a certain trade-off between smoothing and interference attenuation in mind, e.g. the smoothed pseudo WVD [10] and Born-Jordan distributions.

Non-uniform filter banks can to some degree compensate for the leakage problem by varying the filter shape along frequency, and thereby attempting to locally choose a time-frequency leakage trade-off that least obscures important signal features. This can be achieved either by following a fixed rule or in a signal-dependent fashion. In particular, wavelet filter banks [11, 12] can be constructed through a dilation rule.

Similary, variation along time leads to nonstationary Gabor transforms [13, 14]. Joint time-frequency adaptation provides additional flexibility and has been studied e.g. in [15, 16]. All these methods have in common with the filter bank setting, that they do not reduce the smoothing effect overall, but instead try to select locally a good trade-off between time and frequency smoothing, adapted to signal characteristics.

In contrast, the reassignment method, introduced by Kodera et al. [17] and extended by Auger and Flandrin [18, 19], attempts the deconvolution of the short-time Fourier spectrogram. This is achieved by obtaining *instantaneous frequency* and *group delay* estimates from the partial derivatives of the phase of the STFT, which are subsequently applied to *reassign* time-frequency energy to its supposed point of origin, resulting in a considerably sharpened time-frequency representation without unnecessary interference. In [18], the authors also proposed an efficient means of obtaining the partial phase derivatives, which was recently used to extend the reassignment method to general non-uniform filter banks [20]. The combination of non-uniform filter banks and the reassignment method facilitates the design of highly efficient time-frequency representations with excellent concentration. Therefore, they can be of great use in audio signal analysis, visualization and processing alike.

In this contribution, we recall the results from [20] and present the first *real-time capable implementation* of reassigned filter banks in the form of low delay audio visualization integrated into a basic audio player. The audio player and visualization software rely on the open-source Large Time-Frequency Analysis Toolbox (LTFAT)[1], specifically on its C backend, and the JUCE C++ Library[2].

Organization of the paper: The next section recalls the theoretical background behind the reassignment method for general filter banks. In particular, we show how to efficiently obtain the reassigned filter bank representation from the original filter bank coefficients by means of two additional filter bank analyses, specifically designed for the task. Section 2.2 recalls the derivation of the

* This work was supported by the Austrian Science Fund (FWF) START-project FLAME ("Frames and Linear Operators for Acoustical Modeling and Parameter Estimation"; Y 551-N13).

[1] http://ltfat.github.io
[2] http://www.juce.com

general reassignment operators from the short-time Fourier case. Section 3 describes the user experience and functionality of the player during runtime, while Section 4 provides some technical information about the implementation. Finally, the manuscript concludes with a short discussion of possible future work.

2. PRELIMINARIES

Although the previous publications on reassignment were developed for continuous time signals, see references in Section 1, we will present the theory for finite energy sequences $f \in \ell^2(\mathbb{Z})$ to reflect the actual implementation. We denote the DTFT of f as $\widehat{f}(\omega) = \sum_{l\in\mathbb{Z}} f(l)e^{-2\pi i\omega l}$ and the set of nonnegative integers strictly smaller than $K > 0$ by $\mathbb{Z}_K$.

For a signal $f \in \ell^2(\mathbb{Z})$, its STFT with respect to the window $g \in \ell^2(\mathbb{Z})$ is given by

$$\begin{aligned} V_g f(x,\omega) &= \langle f, g_{x,\omega}\rangle \\ &= \sqrt{\mathcal{S}_g f(x,\omega)}e^{2\pi i\phi(x,\omega)}, \end{aligned} \tag{1}$$

where $g_{x,\omega}[l] = e^{2\pi i\omega(l-x)}g[l-x]$, $\mathcal{S}_g f = |\mathcal{V}_g f|^2$ is the spectrogram and $\phi : \mathbb{Z}\times\mathbb{T} \to \mathbb{R}$ is the phase of the STFT.

Let $\boldsymbol{g} = \{g_k \in \ell^2(\mathbb{Z})\}_{k\in\mathbb{Z}_K}$, $\boldsymbol{a} = \{a_k \in \mathbb{N}\}_{k\in\mathbb{Z}_K}$ a sequence of functions and *decimation factors*, respectively. We call the system $\left\{\overline{g_k[na_k - \cdot]}\right\}_{n,k\in\mathbb{Z}}$ a *(analysis) filter bank (FB)*. The associated K-channel *filter bank analysis* is given by

$$c_{n,k} := c_f[n,k] := \langle f, \overline{g_k[na_k - \cdot]}\rangle. \tag{2}$$

A filter bank forms a *frame*, if there are constants $0 < A \le B < \infty$, such that $A\|f\|_2^2 \le \|c\|^2 \le B\|f\|_2^2$, for all $f \in \ell^2(\mathbb{Z})$. The frame property guarantees the stable invertibility of the coefficient mapping by means of a *dual frame* $\{\widetilde{g_{n,k}}\}_{n\in\mathbb{Z},k\in\mathbb{Z}_K}$, i.e.

$$f = \sum_{n,k} c_{n,k}\widetilde{g_{n,k}}, \text{ for all } f \in \ell^2(\mathbb{Z}). \tag{3}$$

2.1. Reassignment for filter banks

In [20] we show that reassignment can be applied to time-frequency FBs. For that purpose denote by ω_k the center frequency[3] of g_k. Define

$$g_k^T[l] := lg_k[l],\ g_k^F := e^{2\pi i\omega_k l}(e^{-2\pi i\omega_k l}g_k)', \tag{4}$$

for all $l \in \mathbb{Z}$, $k \in \mathbb{Z}_K$. Here f' denotes any suitable discrete derivative of f. We can write

$$c_f^T[n,k] = \langle f, \overline{g_k^T[na_k - \cdot]}\rangle,\ c_f^F[n,k] = \langle f, \overline{g_k^F[na_k - \cdot]}\rangle. \tag{5}$$

Moreover, we obtain the estimated true time position in samples

$$\boldsymbol{x}_0(na_k,\omega_k) = na_k + \mathbf{Re}\left(c_f^T[n,k]/c_f[n,k]\right), \tag{6}$$

whenever $c_f[n,k] \neq 0$ and similarly the estimated true normalized frequency position

$$\boldsymbol{\omega}_0(na_k,\omega_k) = \omega_k - \mathbf{Im}\left(c_f^F[n,k]/c_f[n,k]\right). \tag{7}$$

[3]For the reassignment procedure to make sense, we assume that g_k is well-localized around time 0 and $\widehat{g_k}$ is well-localized around frequency ω_k with $\omega_j < \omega_k$ if $j < k$ and that the points ω_k, $k \in \mathbb{Z}_K$ are well-distributed on the torus.

We might prefer the reassigned representation to be defined on the sampling points (na_k, ω_k). This leads to the *reassigned filter bank (RFB)* defined as

$$c_f^{\mathcal{R}}[n,k] := \sum_{(m,l)\in L_{n,k}} |c_f[m,l]|^2, \tag{8}$$

where

$$L_{n,k} := \{(m,l) \in \mathbb{Z}^2 \ :\ (\boldsymbol{n}_0[m,l], \boldsymbol{k}_0[m,l]) = (n,k)\}, \tag{9}$$

and

$$\boldsymbol{k}_0[m,l] := \operatorname*{arg\,min}_{k\in\mathbb{Z}_K} |\omega_k - \boldsymbol{\omega}_0(la_k,\omega_k)| \text{ and} \tag{10}$$

$$\boldsymbol{n}_0[m,l] := \left\lfloor \frac{\boldsymbol{x}_0(la_k,x_k)}{a_{k_0(m,l)}} \right\rceil. \tag{11}$$

Note that in this setup, frequency reassignment has priority over time reassignment.

2.2. Derivation of the reassignment operators

The reassignment operators for the filter bank case are in fact easily derived from known expressions for the reassignment operators for the short-time Fourier transform. To see that, consider a function $g \in \ell^2(\mathbb{Z})$, well-localized around $x \in \mathbb{Z}$ with its Fourier transform $\hat{g}$ well-localized around $\omega \in \mathbb{T}$. Then

$$\tilde{g}[l] := e^{-2\pi i\omega l}g[l+x] \tag{12}$$

is well-localized around 0 in time and frequency and we have

$$V_{\tilde{g}} f(x,\omega) = \langle f, \tilde{g}_{x,\omega}\rangle = \langle f, g\rangle. \tag{13}$$

For the short-time Fourier transform $V_{\tilde{g}}f$, the reassignment operators are derived from the instantaneous frequency and group delay and defined as $\boldsymbol{x}_0(x,\omega) = x - \frac{\partial}{\partial\omega}\phi(x,\omega)$ and $\boldsymbol{\omega_0}(x,\omega) = \frac{\partial}{\partial x}\phi(x,\omega)$, where ϕ is the phase of $V_{\tilde{g}}f$ as in Eq. (1). In contrast to the continuous case $f, g \in \mathbf{L}^2(\mathbb{R})$ usually present in the literature, the derivative $\frac{\partial}{\partial x}$ over $x \in \mathbb{Z}$ depends on a chosen convention for discrete derivatives. In practice however, different discrete derivatives provide similar reassignment results.

Auger and Flandrin [18] have shown that the reassignment operators can be expressed as the pointwise product of the STFTs with respect to 3 window functions, depending on the window $\tilde{g}$ and this is true for the discrete signals as well. We obtain:

$$\begin{aligned} \boldsymbol{x}_0(x,\omega) &= x + \mathbf{Re}\left(\mathcal{V}_{\tilde{g}_T} f(x,\omega)/\mathcal{V}_{\tilde{g}} f(x,\omega)\right) \\ &= x + \mathbf{Re}\left(\frac{\mathcal{V}_{\tilde{g}_T} f(x,\omega)\overline{\mathcal{V}_{\tilde{g}} f(x,\omega)}}{\mathcal{S}_{\tilde{g}} f(x,\omega)}\right) \end{aligned} \tag{14}$$

and

$$\begin{aligned} \boldsymbol{\omega_0}(x,\omega) &= \omega - \mathbf{Im}\left(\mathcal{V}_{\tilde{g}'} f(x,\omega)/\mathcal{V}_{\tilde{g}} f(x,\omega)\right) \\ &= \omega - \mathbf{Im}\left(\frac{\mathcal{V}_{\tilde{g}'} f(x,\omega)\overline{\mathcal{V}_{\tilde{g}} f(x,\omega)}}{\mathcal{S}_{\tilde{g}} f(x,\omega)}\right), \end{aligned} \tag{15}$$

whenever $\mathcal{S}_{\tilde{g}} f(x,\omega) \neq 0$ and 0 else. Here, $\tilde{g}'[l] = \frac{\partial}{\partial l}\tilde{g}[l]$ is the discrete derivative and $\tilde{g}_T[l] := l\tilde{g}[l]$ is a time weighted version of $\tilde{g}$.

Note that differentiation is a translation-invariant operator and time weighting is invariant under modulation, i.e. multiplication

with $e^{2\pi i\xi l}$ for $\xi \in \mathbb{T}$. Therefore, with $g^T[l] := [l-x]g[l]$ and $g^F(t) := e^{2\pi i\omega l}(e^{-2\pi i\omega l}g)'[l]$, we obtain

$$\mathcal{V}_{\tilde{g}_T} f(x,\omega) = \langle f, g^T \rangle, \text{ and } \mathcal{V}_{\tilde{g}'} f(x,\omega) = \langle f, g^F \rangle. \quad (16)$$

Given a filter bank or continuous filter bank, simply set $g = \overline{g_k[-\cdot]}$ to obtain Eqs. (6) and (7).

2.3. Filter bank choice

The reassignment method works optimally, only if the chosen filter bank is already able to provide some (phase space/time-frequency) separation of the individual signal components. Although RM is able to improve the separation, and localization, of time-freuqency components, if this is not the case, there are clear limits to this. If, for example, a coefficient $c_f[n,k]$ contains equal amounts of energy from two signal components centered at frequencies $\omega_1 \neq \omega_2$, then it cannot be expected that the estimated true frequency position $\boldsymbol{\omega}_0(na_k, \omega_k)$ provides a meaningful value. In practice, it will usually point at some frequency in $[\omega_1, \omega_2]$, at least not degrading localization.

Consequently, RM can only provide optimal performance when combined with a filter bank that is adapted to the signal class at hand. In the setting of complex audio signals, this usually requires good frequency localization in a band of low frequencies and, for increasing frequency, a gradual increase localization in time resolution. Therefore, filter banks adapted to perceptually-motivated frequency scales such as Bark, ERB or Mel [21, 22, 23, 24], or constant-Q filter banks [25, 26, 27, 28], can be expected to perform well.

3. DESCRIPTION OF THE AUDIO PLAYER AND VISUALIZATION

The main purpose of the proposed audio player application is to provide a body and interface for the visualization of audio content by means of a low delay spectro-temporal filter bank representation, with emphasis on reassigned filter banks. Nonetheless, it provides most functionality expected of a basic audio player such as playback control (*play, pause, stop*), loading several files into a playlist, switching between tracks in the list and automatic looping of the full playlist or single tracks. The basic interface, consisting of the toolbar below the moving spectrogram in the main player window, as well as the playlist window, should be familiar to anyone that has used one of the countless media players available. For an illustration and detailed explanation, see Figures 1 and 2.

In addition to the files in the playlist, the player provides a temporary slot for loading a single file without adding it to the list. This is done by selecting *Open File* → *Load Audio file* from the top menu bar.

Audio devices can be selected through *Options* → *Audio Settings*. If an input source is selected, the *source selector* allows on-the-fly switching between file playback and the input source.

The *main features*, however, are the ability to switch between the original and reassigned filter bank representations by the press of the *reassignment toggle* and loading of alternative analysis filter banks during runtime, through the *load filter bank* button, see Figure 1. The filterbanks can be generated using LTFAT in MATLAB/GNU Octave and exported in a binary format using script `write_filterbank_bin.m` included in the plugin source code archive, see Section 4. The binary file contains the generated filterbank g_k, a_k together with g_k^T and g_k^F for $k = 0, \ldots, K-1$

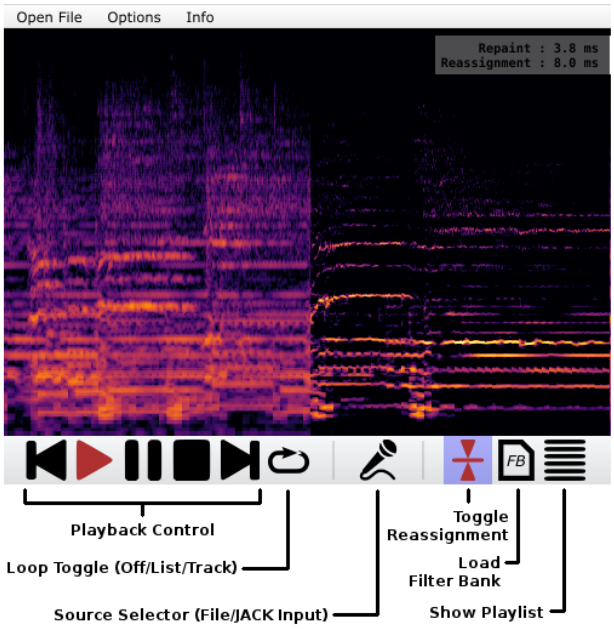


Figure 1: Screenshot of the audio player interface, showing (left) the non-reassigned spectrogram and the reassigned spectrogram (right). The main player window performs the usual playback control tasks through the 5 playback control buttons. The *loop toggle* can be used to choose between (not) looping the playlist or looping the current track. The *source selector* switches between file input (default) and input through the JACK input stream (only available if JACK is selected in the audio settings). On-the-fly toggling between the standard and reassigned spectrogram is enabled by the *reassignment toggle*, while the last two buttons allow loading an alternative filter bank exported from MATLAB through the provided `write_filterbank_bin.m` and showing the playlist.

for a fixed buffer length. Currently, LTFAT contains the following band-limited perfect-reconstruction filterbank generating routines:

- `erbfilters` – Equivalent rectangular bandwidth filterbank [23, 24]
- `cqtfilters` – Constant-Q filterbank [27]
- `warpedfilters` – General frequency warped filterbanks [29]
- `audfilters` – Band-limited filters adapted to auditory scales [24]

Custom filterbanks can be used provided they conform to LTFAT filterbank format.

Finally, the dynamic range of the spectrogram visualization can be varied by a slider, available on right-clicking the spectrogram element, see Figure 3.

4. TECHNICAL BACKGROUND OF THE IMPLEMENTATION

Although the reassignment procedure was defined for general filterbanks, in our contribution, we use *band-limited* filterbanks with possibly rational subsampling rates (that is non-integer a_k) which

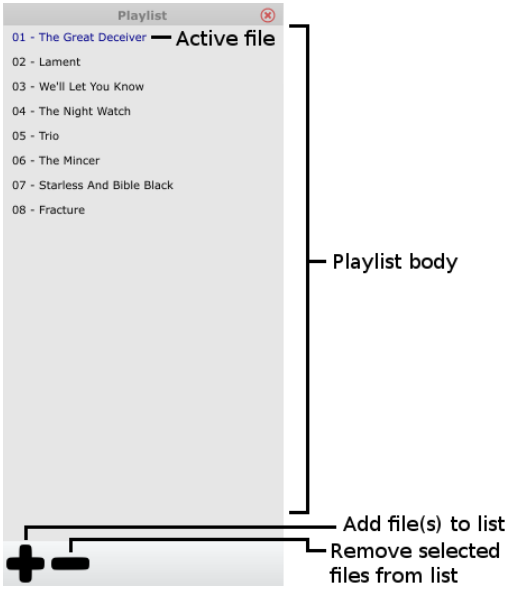


Figure 2: Screenshot of the playlist element. The playlist body shows all the loaded files, with the currently played file highlighted in blue. Individual items can be selected by a single click on the entry. Multiple entries are selected by additionally pressing the Ctrl or Shift buttons, similar to standard window manager behavior. Double-clicking starts playback of the clicked file, while the + and − buttons on the bottom of the window allow for adding files to the playlist or removing them. If the playlist is closed, pressing the *playlist button* in the main window makes it reappear.

enables an efficient implementation. More precisely, for the filtering, we follow the computational framework from [28] (using the phase convention introduced later in [30]) which was derived for constant-Q transform, but it applies to general band-limited filterbanks as well. By default, we use ERBlets [24] with 510 filters with rational subsampling rates and buffer length $L = 4096$ samples. The decimation factors are chosen to provide the least redundant, perfect reconstruction system with no aliasing in the subbands, given the fixed ERBlet filters. The frequency responses are shown in Fig. 4.

4.1. Dependencies

The player was implemented using the C++ library JUCE [31] (version 4.1.0), the implementation of the filterbank and the reassignment was linked from the C backend library[4] of LTFAT [32] which in turn uses FFTW library (version 3.3.4) [33] for the FFT calculations.

All involved libraries are licensed under GPL, therefore our program is licensed under terms of GPLv3 and its source codes are freely available[5].

The program was primarily developed on Linux, but it also compiles and runs on Windows. The interested reader can find the *Introjucer* project template included with the source code.

[4] `http://github.com/ltfat/libltfat`

[5] `http://ltfat.github.io/related/reassignment`

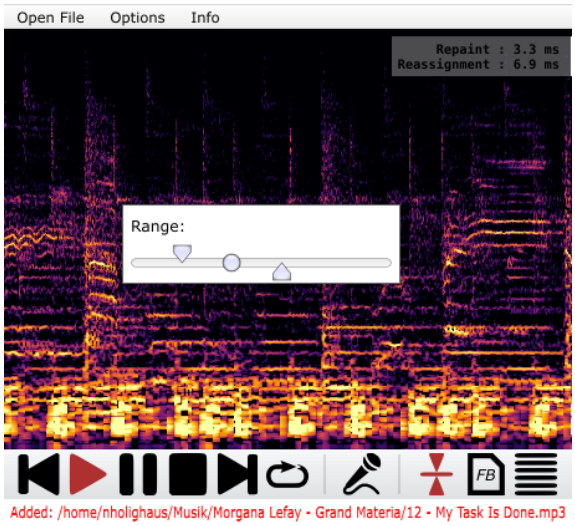


Figure 3: The dynamic range of the moving spectrogram can be conveniently modified through the use of the *dynamic range slider* appearing on right-click on the spectrogram.

4.2. Implementation

The input stream of audio data blocks is rebuffered according to the filterbank buffer length L with half buffer length overlap. For that purpose we employ a lock-free circular buffer. On the audio thread, the data is only written to it as they arrive. The circular buffer is repeatedly polled for new data in regular intervals ($1/60$ s in our implementation) by a separate thread. When available, L samples are read to a working buffer, but the circular buffer read index is advanced only by $L/2$ samples.

Next, the buffer is weighted with a Hann window such that sum of the shifted windows equals constant 1.

Three sets of filterbank coefficient subbands are computed using the chosen filterbank g_k, a_k $(k = 0, \dots, K-1)$ and its two derived versions[6] g_k^T and g_k^F. Each filter is defined by its frequency response being nonzero only in the range $[b_{\min,k}, b_{\max,k}]$ that is for $B_k = b_{\max,k} - b_{\min,k} + 1$ frequency bins. The bandwidth B_k is common for all three versions of k-th filter. Each filter produces a subband consisting of $N_k = L/a_k$ coefficients. In the following, we always assume that there is no aliasing in the resulting subbands; therefore $N_k \geq B$ and a_k is allowed to be a rational number.

Formally written, the three sets of coefficient subbands $c_{f_w,k}$, $c^T_{f_w,k}$ and $c^F_{f_w,k}$ for a single buffer are obtained as follows (in pseudo Matlab code):

1. Load buffer f of length L.
2. $f_w = w. * f$, where w is length L Hann window.
3. $F_w = \mathrm{FFT}_L(f_w)$

[6] Our implementation of g_k^F relies on the spectral derivative, i.e. the filter is obtained by multiplying the frequency response of g_k with $(\omega - \omega_k)$, where ω_k is the center frequency of g_k. As long as g_k and its Fourier transform $\widehat{g_k}$ are concentrated, we have not observed any meaningful difference in the reassignment result when substituting the spectral derivative with any other suitable discrete derivative. The reason for us choosing this particular convention is simply that g_k^F is obtained by the same procedure as g_k^T, applied in the Fourier domain.

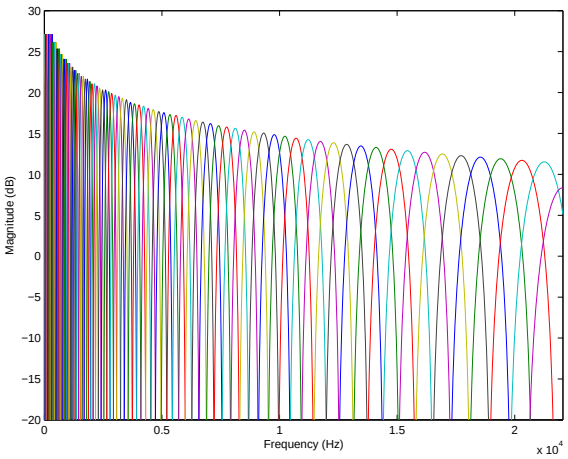


Figure 4: Frequency responses of the default filterbank. Every fifth filter is shown.

4. For $k = 0, \ldots, K-1$ do

   (a) $C_{f_w,k} = F_w . * g_k$ reduced to range $[b_{\min,k}, b_{\max,k}]$

   (b) Extend $C_{f_w,k}$ with zeros to length N_k

   (c) $c_{f_w,k} = \text{IFFT}_{N_k}(\text{circshift}_{N_k}(C_{f_w,k}, -b_{\min,k}))$

5. Repeat loop 4 for g_k^T and for g_k^F to obtain $c_{f_w,k}^T$ and $c_{f_w,k}^F$.

If subband aliasing is present, i.e. $N_k < B$ for some k, then step 4(b) needs to be modified to *fold* C_{f_w} to the appropriate length N_k, but the remainder of this section remains valid.

Before applying the reassignment procedure, we use $c_{f_w,k}$, $c_{f_w,k}^T$ and $c_{f_w,k}^F$ from the current and from the previous buffers to approximate the corresponding true coefficients c_f, c_f^T, c_f^F at the appropriate buffer location. To achieve that, the first half of each subband k from the current buffer is element-wise added to the second half of the corresponding subband from the previous buffer.

The last step is obtaining the reassigned coefficients from (8) using the just computed approximation of c_f, c_f^T, c_f^F.

The overall delay, that is the delay between the instant the circular buffer has L samples available and the instant the data actually appears on the screen, consists of several contributions. The constant part comes from the overlapping of subbands and it is equal to $L/2$ samples. Additional delay is introduced by the computation itself and by the time required to repaint the screen. Sum of these values must be less than $L/2$ times the sampling rate, otherwise the program starts skipping samples. Due to the multithreaded nature of the program, the periodic polling of the circular buffer introduces jitter equal to the request period.

5. CONCLUSION AND OUTLOOK

In this contribution, we discussed the application of the reassignment method to general filter banks. We provided a low delay implementation in a basic audio player that uses the reassigned coefficients for real-time visualization.

In [20], the authors have shown that the reassigned representation can be used as an interface for time-frequency processing, by relying on the *inverse reassignment map*

$$\mathbf{iR}[n,k] = \{(m,l) \in \mathbb{Z} \times \mathbb{Z}_K \ : \ (n_0[m,l], k_0[m,l]) = (n,k)\},$$

which serves as a lookup table during processing. In other words, the user selects a time-frequency region $\Omega \subset \mathbb{Z} \times \mathbb{Z}_K$ in the reassigned spectrogram to be processed and decides on a processing operation. That processing operation is then applied automatically to the filter bank coefficients $c_f[\mathbf{iR}[\Omega]]$. If the filter bank forms a frame, then the modified coefficients can be synthesized to obtain a processed signal.

Simple processing capabilities could be easily added to the current audio player by providing a *brush tool* similar to image processing applications. Modifications could be applied by *painting* over the moving spectrogram, or by pausing playback and painting in a stationary image of the currently displayed audio segment. By implementing a synthesis filter bank, operating in the background, the user could switch between playback the incoming audio stream on the *right end* of the visualization and the, possibly modified outgoing audio stream on the *left end*.

Instead of loading separate filter banks individually, a single file, containing a number of filter banks to switch between on-the-fly would make changing between representations more convenient. In fact, this type of functionality is already partially implemented

In applications where slightly longer delay is acceptable, block processing scheme could be modified to more closely resemble the framework proposed in [28]. This would reduce blocking artifacts, especially when short blocks and filters with very narrow frequency response are considered.

6. REFERENCES

[1] D. Gabor, "Theory of communication," *J. IEE*, vol. 93, no. 26, pp. 429–457, 1946.

[2] Karlheinz Gröchenig, *Foundations of Time-Frequency Analysis*, Appl. Numer. Harmon. Anal. Birkhäuser, Boston, MA, 2001.

[3] Henrique Malvar, *Signal Processing with Lapped Transforms*, Boston, MA: Artech House. xvi, 1992.

[4] Martin Vetterli, "Filter banks allowing perfect reconstruction," *Signal Process.*, vol. 10, pp. 219–244, 1986.

[5] Sony Akkarakaran and P. P. Vaidyanathan, "Nonuniform filter banks: new results and open problems," in *Beyond wavelets*, vol. 10 of *Studies in Computational Mathematics*, pp. 259–301. Elsevier, 2003.

[6] Leon Cohen, "Time-frequency distributions - a review," *Proc. IEEE*, vol. 77, no. 7, pp. 941–981, 1989.

[7] Leon Cohen, *Time-Frequency Analysis: Theory and Applications.*, Prentice Hall Signal Processing Series. Prentice Hall, 1995.

[8] Eugene P. Wigner, "On the quantum correction for thermodynamic equilibrium.," *Phys. Rev., II. Ser.*, vol. 40, pp. 749–759, 1932.

[9] J de Ville et al., "Théorie et applications de la notion de signal analytique," *Cables et transmission*, vol. 2, no. 1, pp. 61–74, 1948.

[10] Wolfgang Martin and Patrick Flandrin, "Wigner-ville spectral analysis of nonstationary processes," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 33, no. 6, pp. 1461–1470, 1985.

[11] Martin Vetterli and Jelena Kovacevic, *Wavelets and subband coding*, vol. 87, Prentice Hall PTR Englewood Cliffs, New Jersey, 1995.

[12] Stéphane Mallat, *A wavelet tour of signal processing: The sparse way*, Academic Press, Third edition, 2009.

[13] Peter Balazs, Monika Dörfler, Florent Jaillet, Nicki Holighaus, and Gino Angelo Velasco, "Theory, implementation and applications of nonstationary Gabor frames," *J. Comput. Appl. Math.*, vol. 236, no. 6, pp. 1481–1496, 2011.

[14] Marco Liuni, Peter Balazs, and Axel Röbel, "Sound Analysis and Synthesis Adaptive in Time and Two Frequency Bands," in *Proc. of the 14th Int. Conference on Digital Audio Effects (DAFx-11), Paris, France, September 19-23*, September 2011.

[15] Cormac Herley, Jelena Kovacevic, Kannan Ramchandran, and Martin Vetterli, "Tilings of the time-frequency plane: Construction of arbitrary orthogonal bases and fast tiling algorithms," *Signal Processing, IEEE Transactions on*, vol. 41, no. 12, pp. 3341–3359, 1993.

[16] Monika Dörfler, "Quilted Gabor frames - A new concept for adaptive time-frequency representation," *Adv. in Appl. Math.*, vol. 47, no. 4, pp. 668–687, Oct. 2011.

[17] Kunihiko Kodera, Claude De Villedary, and Roger Gendrin, "A new method for the numerical analysis of non-stationary signals," *Physics of the Earth and Planetary Interiors*, vol. 12, no. 2, pp. 142–150, 1976.

[18] F. Auger and P. Flandrin, "Improving the readability of time-frequency and time-scale representations by the reassignment method," *Signal Processing, IEEE Transactions on*, vol. 43, no. 5, pp. 1068 –1089, may 1995.

[19] Eric Chassande Mottin, Ingrid Daubechies, Francois Auger, and Patrick Flandrin, "Differential reassignment," *IEEE Signal Processing Letters*, vol. 4, no. 10, pp. 293 –294, oct 1997.

[20] Nicki Holighaus, Zdenek Prusa, and Peter L. Søndergaard, "Reassignment and synchrosqueezing for general time-frequency filter banks, subsampling and processing," *Signal Processing*, pp. 1–8, Aug. 2016.

[21] B. R. Glasberg and B. C. J. Moore, "Derivation of auditory filter shapes from notched-noise data," *Hear. Res.*, vol. 47, pp. 103–138, 1990.

[22] V. Hohmann, "Frequency analysis and synthesis using a gammatone filterbank," *Acta Acust. united Ac.*, vol. 88, no. 3, pp. 433–442, 2002.

[23] Thibaud Necciari, Peter Balazs, Nicki Holighaus, and Peter Søndergaard, "The ERBlet transform: An auditory-based time-frequency representation with perfect reconstruction," in *Proceedings of the 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2013)*, 2013, pp. 498–502.

[24] Thibaud Necciari, Nicki Holighaus, Peter Balazs, and Zdenek Prusa, "A perceptually motivated filter bank with perfect reconstruction for audio signal processing," *submitted, preprint available: http://arxiv.org/abs/1601.06652*, 2015.

[25] Judith Brown, "Calculation of a constant Q spectral transform," *J. Acoust. Soc. Amer.*, vol. 89, no. 1, pp. 425–434, 1991.

[26] Christian Schörkhuber and Anssi Klapuri, "Constant-Q toolbox for music processing," in *Proceedings of the 7th Sound and Music Computing Conference (SMC), 2010*, 2010.

[27] Gino Angelo Velasco, Nicki Holighaus, Monika Dörfler, and Thomas Grill, "Constructing an invertible constant-Q transform with non-stationary Gabor frames," *Proceedings of DAFX11*, 2011.

[28] Nicki Holighaus, Monika Dörfler, Gino Angelo Velasco, and Thomas Grill, "A framework for invertible, real-time constant-Q transforms," *IEEE Trans. Audio Speech Lang. Process.*, vol. 21, no. 4, pp. 775 –785, 2013.

[29] Nicki Holighaus, Zdenek Prusa, and Christoph Wiesmeyr, "Designing tight filter bank frames for nonlinear frequency scales," *Sampling Theory and Applications 2015*, 2015.

[30] Christian Schörkhuber, Anssi Klapuri, Nicki Holighaus, and Monika Dörfler, "A Matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution," in *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. AES, January 2014.

[31] Jules Storer, "JUCE: Jules' utility class extensions (version 4.1)," 2014, Available at: `http://www.juce.com`.

[32] Zdenek Prusa, Peter L. Søndergaard, Nicki Holighaus, Christoph Wiesmeyr, and Peter Balazs, "The Large Time-Frequency Analysis Toolbox 2.0," in *Sound, Music, and Motion*, Mitsuko Aramaki, Olivier Derrien, Richard Kronland-Martinet, and Sølvi Ystad, Eds., Lecture Notes in Computer Science, pp. 419–442. Springer International Publishing, 2014.

[33] Matteo Frigo and Steven G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005, Special issue on "Program Generation, Optimization, and Platform Adaptation".

ESTIMATES OF THE RECONSTRUCTION ERROR IN PARTIALLY REDRESSED WARPED FRAMES EXPANSIONS

Thomas Mejstrik *

Department of Mathematics
University of Vienna
Vienna, Austria
tommsch@gmx.at

Gianpaolo Evangelista

Institute for Composition and Electroacoustics,
MDW, University of Music and Performing Arts
Vienna, Austria
evangelista@mdw.ac.at

ABSTRACT

In recent work, redressed warped frames have been introduced for the analysis and synthesis of audio signals with nonuniform frequency and time resolutions. In these frames, the allocation of frequency bands or time intervals of the elements of the representation can be uniquely described by means of a warping map. Inverse warping applied after time-frequency sampling provides the key to reduce or eliminate dispersion of the warped frame elements in the conjugate variable, making it possible, e.g., to construct frequency warped frames with synchronous time alignment through frequency. The redressing procedure is however exact only when the analysis and synthesis windows have compact support in the domain where warping is applied. This implies that frequency warped frames cannot have compact support in the time domain. This property is undesirable when online computation is required. Approximations in which the time support is finite are however possible, which lead to small reconstruction errors. In this paper we study the approximation error for compactly supported frequency warped analysis-synthesis elements, providing a few examples and case studies.

1. INTRODUCTION

The availability of configurable time-frequency schemes is an asset for sound analysis and synthesis, where the elements of the representation can efficiently capture features of the signal. These include features of interpretation: e.g. glissandi and vibrati; human perception: e.g. non-uniform frequency sensitivity of the cochlea; music theory: e.g. scales (pentatonic, 12-tone, indian scales with unequally spaced tones, etc.); physical effects: e.g. nonharmonic overtones in the low register of the piano or of percussion instruments and so forth. When used in the context of Music Information Retrieval, the adaptation of the representation to music scales is bound to improve. e.g. for instrument-, note-, chord- and overtone detection and recognition.

Traditionally, two extreme cases have been considered: Gabor expansions featuring uniform time and frequency resolutions and orthogonal wavelet expansions and frames featuring octave band allocation and constant uncertainty of the representation elements. In previous work [1, 2, 3, 4, 5, 6, 7], generalised Gabor frames have been constructed which allow for non-uniform time-frequency schemes with perfect reconstruction. In [3] the allocation of generalised Gabor atoms is specified according to a frequency or time warping map. In [8] the STFT redressing method is introduced, which, with the use of additional warping in time-frequency, shows under which conditions one can have generalised Gabor frames. These conditions stem from the interaction of sampling in time-frequency and frequency or time warping operators, which allow to incorporate the results in [3] in a more general context. It is shown that arbitrary allocation of the atoms is exactly possible in the so called *painless case*, i.e. in the case of finite time support of the windows for arbitrary time interval allocation and of finite frequency support of the windows for arbitrary frequency band allocation. Non-uniform frequency analysis by means of warping was introduced in [7]. Non-uniform Gabor frames with constant-Q were previously introduced in [1] , based on the theory developed in [2], where an ad hoc procedure was employed for their construction. In [3] we provided an alternative general method for their construction, using warping. In [8] the redressing method was introduced and applied to the general construction of non-stationary Gabor frames. In [9] the general theory was revisited with the real-time computational aspects in mind. There, the first approximate schemes were introduced without extensive testing.

Since online computation of the generalised Gabor analysis-synthesis is only possible with finite duration windows, the arbitrary frequency band allocation does not lead to an exact solution in applications that require real-time, while the arbitrary time interval allocation presents little or no problem. In [9] approximations leading to nearly exact representations were introduced.

In this paper we expand on the results in [9] and provide a study of the approximation error on a wide class of signals when finite duration windows are required in arbitrary frequency band allocation.

The paper is organised as follows. In Section 2 we review the concept of applying time and frequency warping to time-frequency representations, together with the redressing method, which involves a further warping operation in the time-frequency domain to reduce or eliminate dispersion. In Section 3 we introduce approximations suitable for the online computation of redressed frame expansions. In Section 4 the results of numerical experiments are shown, which provide estimations of the approximation error. In Section 5 we draw our conclusions.

2. REDRESSED WARPED GABOR FRAMES

In this section we review the concepts leading to the definition of redressing in the context of frequency warped time-frequency representations. First we review the basic notions of STFT (Short-Time Fourier Transform) and Gabor frames. Then we move on to the definition of warped frames and then to the redressing procedure.

* Part of the research contained in this work was performed by this author while he was a Master student at Mdw, Wien, under the guidance of Prof. Gianpaolo Evangelista

2.1. STFT and Gabor Frames

Gabor expansions can be considered as a form of sampling and exact reconstruction of the STFT. As is well-known, given a window h and defining the time-shift operator $\mathbf{T}_\tau s(t) = s(t-\tau)$ and the modulation operator $\mathbf{M}_\nu s(t) = e^{j2\pi\nu t}s(t)$, the STFT is obtained by applying the operator $\mathcal{S}$ to the signal s:

$$[\mathcal{S}s](\tau,\nu) = \langle s, h_{\tau,\nu}\rangle = \langle s, \mathbf{T}_\tau\mathbf{M}_\nu h\rangle = \int_{-\infty}^{+\infty} s(t)\overline{h_{0,0}(t-\tau)}e^{-j2\pi\nu(t-\tau)}dt, \tag{1}$$

where the overbar denotes complex conjugation.

A *Gabor system* is generated by the kernel of $\mathcal{S}$ by sampling the time-frequency plane (τ,ν):

$$\mathcal{G}(h,a,b) = \{\mathbf{T}_{na}\mathbf{M}_{qb}h : n,q \in \mathbb{Z}\} \tag{2}$$

where $a, b > 0$ are sampling parameters.

The scalar products of the signal with the members of the Gabor system

$$\langle s, \mathbf{T}_{na}\mathbf{M}_{qb}h\rangle, \qquad n,q \in \mathbb{Z} \tag{3}$$

provide evaluations of the STFT (1) of a signal s with window h at the time-frequency grid of points (na, qb), with $n, q \in \mathbb{Z}$. The question whether the signal s can be reconstructed from these evaluations can be addressed by introducing the concept of frame.

A sequence of functions $\{\psi_l\}_{l\in I}$ in the Hilbert space $\mathcal{H}$ is called a frame if there exist both positive constant lower and upper bounds A and B, respectively, such that

$$A\|s\|^2 \le \sum_{l\in I}|\langle s,\psi_l\rangle|^2 \le B\|s\|^2 \quad \forall s \in \mathcal{H}, \tag{4}$$

where $\|s\|^2 = \langle s, s\rangle$ is the norm square or total energy of the signal. Frames generate signal expansions, i.e., the signal can be perfectly reconstructed from its projections over the frame.

A Gabor system that is a frame is called a *Gabor frame*. In this case, the signal can be reconstructed from the corresponding samples of the STFT (3). While not unique, reconstruction can be achieved with the help of a dual frame, which in turn is a Gabor frame generated by a dual window $\tilde{h}$. Perfect reconstruction essentially depends on the choice of the window and the sampling grid. One can show that there exist no Gabor frames when $ab > 1$. See [10] for more informations about Gabor frames.

2.2. Warped STFT and Gabor Frames

The warped STFT can be obtained by warping the signal prior to applying the STFT operator. In this paper we focus on pure frequency warping.

A frequency warping operator $\mathbf{W}_{\tilde{\theta}}$ is completely characterised by a function composition operator $\mathbf{W}_\theta$, such that $\mathbf{W}_\theta x = x \circ \theta$, in the frequency domain:

$$\mathbf{W}_{\tilde{\theta}} = \mathcal{F}^{-1}\mathbf{W}_\theta\mathcal{F}, \tag{5}$$

where $\mathcal{F}$ is the Fourier transform operator. The function θ is the frequency warping map, which transforms the Fourier transform $\hat{s} = \mathcal{F}s$ of a signal s into the Fourier transform $\hat{s}_{fw} = \mathcal{F}s_{fw}$ of another signal s_{fw}. We affix the ˜ symbol over the map θ as a reminder that the map operates in the frequency domain.

If the warping map is one-to-one and almost everywhere differentiable then a unitary form $\mathbf{U}_{\tilde{\theta}}$ of the warping operator can be defined by the following frequency domain action

$$\hat{s}_{fw}(\nu) = \left[\widehat{\mathbf{U}_{\tilde{\theta}}s}\right](\nu) = \sqrt{\left|\tfrac{d\theta}{d\nu}\right|}\hat{s}(\theta(\nu)), \tag{6}$$

where ν denotes frequency. We assume henceforth that all warping maps are almost everywhere increasing so that the magnitude sign can be dropped from the derivative under the square root.

Given a frequency warping operator $\mathbf{W}_{\tilde{\theta}}$, the warped STFT is defined through the operator $\mathcal{S}_{\tilde{\theta}}$ as follows

$$[\mathcal{S}_{\tilde{\theta}}s](\tau,\nu) = [\mathcal{S}\mathbf{W}_{\tilde{\theta}}s](\tau,\nu) = \langle \mathbf{W}_{\tilde{\theta}}s, h_{\tau,\nu}\rangle = \left\langle s, \mathbf{W}^\dagger_{\tilde{\theta}}h_{\tau,\nu}\right\rangle, \tag{7}$$

which is indeed a warped version of (1), where $\mathbf{W}^\dagger_{\tilde{\theta}}$ is the adjoint of the warping operator. If the warping operator is unitary then we have $\mathbf{W}^\dagger_{\tilde{\theta}} = \mathbf{W}^{-1}_{\tilde{\theta}} = \mathbf{W}_{\tilde{\theta}^{-1}}$. In that case, warping the signal prior to STFT is perfectly equivalent to perform STFT analysis with inversely frequency warped windows. The warped STFT is unitarily equivalent to the STFT so that a number of properties concerning conditioning and reconstruction hold [11].

The Fourier transforms of the frequency warped STFT analysis elements are

$$\hat{\tilde{h}}_{\tau,\nu}(f) = \left[\widehat{\mathbf{W}_{\tilde{\theta}^{-1}}h_{\tau,\nu}}\right](f) = \sqrt{\tfrac{d\theta^{-1}}{df}}\hat{h}(\theta^{-1}(f)-\nu)e^{-j2\pi\theta^{-1}(f)\tau}, \tag{8}$$

i.e., the warped STFT analysis elements are obtained from frequency warped modulated windows centred at frequencies $f = \theta(\nu)$. The windows are time-shifted with dispersive delay, where the group delay is $\tau\frac{d\theta^{-1}}{df}$.

Frequency warping generally disrupts the time organisation of signals due to the fact that the time-shift operator $\mathbf{T}_\tau$ does not commute with the frequency warping operator [9].

From (4) it is easy to see that any unitary operation, in particular unitary warping, on a frame results in a new frame with the same frame bounds A and B [11]. Since the atoms are not generated by shifting and modulating a single window function, the resulting frames are not necessarily of the Gabor type. However, warping prior to conventional Gabor analysis and unwarping after Gabor synthesis always leads to perfect reconstruction.

Starting from a Gabor frame (analysis) $\{\varphi_{n,q}\}_{q,n\in\mathbb{Z}}$ and dual frame (synthesis) $\{\gamma_{n,q}\}_{n,q\in\mathbb{Z}}$:

$$\begin{aligned}\varphi_{n,q} &= \mathbf{T}_{na}\mathbf{M}_{qb}h\\ \gamma_{n,q} &= \mathbf{T}_{na}\mathbf{M}_{qb}g,\end{aligned} \tag{9}$$

where h and g are dual windows, warped frames can be generated, following (8), by unwarping the analysis and synthesis frames. In the case of non-unitary warping, a frequency domain scaling operation is necessary in order to reconstruct the original signal. For the case of unitary warping we simply have:

$$\begin{aligned}\tilde{\varphi}_{n,q} &= \mathbf{U}^\dagger_{\tilde{\theta}}\varphi_{n,q} = \mathbf{U}_{\tilde{\theta}^{-1}}\mathbf{T}_{na}\mathbf{M}_{qb}h\\ \tilde{\gamma}_{n,q} &= \mathbf{U}^\dagger_{\tilde{\theta}}\gamma_{n,q} = \mathbf{U}_{\tilde{\theta}^{-1}}\mathbf{T}_{na}\mathbf{M}_{qb}g,\end{aligned} \tag{10}$$

where $\{\tilde{\varphi}_{n,q}\}_{q,n\in\mathbb{Z}}$ is the frequency warped analysis frame and $\{\tilde{\gamma}_{n,q}\}_{n,q\in\mathbb{Z}}$ is the dual warped frame for the synthesis. With these

definitions, one obtains the signal expansion

$$s = \sum_{n,q\in Z} \langle s, \tilde{\varphi}_{n,q}\rangle \tilde{\gamma}_{n,q}. \tag{11}$$

Warped Gabor frames suffer from the same problem as the warped STFT. Indeed the Fourier transforms of the warped Gabor frame elements bear frequency dispersive delays so that dispersive time samples are produced by the direct application of the frequency warped frame analysis.

2.3. Redressing Methods

As shown in [8, 9], the dispersive delays intrinsic to the warped STFT can be redressed, i.e. made into constant delays in each analysis band if frequency unwarping is performed in the transformed time domain, i.e. with respect to time shift. In other words, instead of (7) we consider the similarity transformation $\mathbf{W}_{\tilde{\theta}}^{\dagger}\mathcal{S}\mathbf{W}_{\tilde{\theta}}$ on the STFT operator, which is time-shift covariant. In fact, one has:

$$\left[\widehat{\mathbf{W}_{\tilde{\theta}^{-1}}\mathcal{S}\mathbf{W}_{\tilde{\theta}}}s\right](f,\nu) = \overline{\hat{h}_{0,0}(\theta^{-1}(f)-\nu)}\hat{s}(f), \tag{12}$$

which is in the form of a time-invariant filtering operation, corresponding to convolution in time domain. The filters are frequency warped versions of the modulated windows in the traditional STFT. The Fourier transform of the redressed analysis elements are

$$\hat{\tilde{h}}_{\tau,\nu}(f) = \left[\widehat{\mathbf{T}_{\tau}\mathbf{W}_{\tilde{\theta}}h_{0,\nu}}\right](f) = \hat{h}_{0,0}(\theta^{-1}(f)-\nu)e^{-j2\pi f\tau}, \tag{13}$$

which shows that the dispersive delays in the analysis elements (8) are brought back to non-dispersive delays.

In redressing warped Gabor frames one faces a further difficulty due to time-frequency sampling. In this case, inverse frequency warping can only be applied to sequences (with the respect to the time shift index) and may not perfectly reverse the dispersive effect of the original map on delays.

Unitary frequency warping in discrete time can be realised with the help of an orthonormal basis of $\ell^2(\mathbb{Z})$ constructed from an almost everywhere differentiable warping map ϑ that is one-to-one and onto $[-\frac{1}{2}, +\frac{1}{2}[$, as follows:

$$\mu_m(n) = \int_{-\frac{1}{2}}^{+\frac{1}{2}} \sqrt{\tfrac{d\vartheta}{d\nu}} e^{j2\pi(n\vartheta(\nu)-m\nu)}d\nu, \tag{14}$$

where $n, m \in \mathbb{Z}$ (see [12, 13, 14, 15, 16]). The map ϑ can be extended over the entire real axis as congruent modulo 1 to a 1-periodic function.

Given any sequence $\{x(n)\}$ in $\ell^2(\mathbb{Z})$, the action of the discrete-time unitary warping operator $\mathbf{D}_{\tilde{\vartheta}}$ is defined as follows:

$$\tilde{x}(m) = [\mathbf{D}_{\tilde{\vartheta}}x](m) = \langle x, \mu_m\rangle_{\ell^2(\mathbb{Z})}. \tag{15}$$

In fact, the sequence $\{\tilde{x}(m)\}$ in $\ell^2(\mathbb{Z})$ satisfies

$$\hat{\tilde{x}}(\nu) = \sqrt{\tfrac{d\vartheta}{d\nu}}\hat{x}(\vartheta(\nu)), \tag{16}$$

where the $\hat{}$ symbol, when applied to sequences, denotes discrete-time Fourier transform. The sequences $\overline{\eta_m(n)}$ define the nucleus of the inverse unitary frequency warping $\ell^2(\mathbb{Z})$ operator $\mathbf{D}_{\tilde{\vartheta}^{-1}} = \mathbf{D}_{\tilde{\vartheta}}^{\dagger}$. where $\eta_m(n) = \mu_n(m)$.

In order to limit or eliminate time dispersion in the frequency warped Gabor expansion, the discrete-time frequency warping operator $\mathbf{D}_{\tilde{\vartheta}^{-1}}$ is applied to the time sequence of expansion coefficients over the warped Gabor frames. Since the operator is applied only on the time index, for generality, one can include dependency of the map and of the sequences η_n on the frequency index q. The process can be equivalently described by defining the redressed frequency warped Gabor analysis and synthesis frames as follows:

$$\begin{aligned}\tilde{\tilde{\varphi}}_{n,q} &= \mathbf{D}_{\tilde{\vartheta}_q^{-1}}\tilde{\varphi}_{\bullet,q} = \sum_m \eta_{n,q}(m)\tilde{\varphi}_{m,q}\\ \tilde{\tilde{\gamma}}_{n,q} &= \mathbf{D}_{\tilde{\vartheta}_q^{-1}}\tilde{\gamma}_{\bullet,q} = \sum_m \eta_{n,q}(m)\tilde{\gamma}_{m,q},\end{aligned} \tag{17}$$

obtaining:

$$s = \sum_{n,q\in(Z)} \langle s, \tilde{\tilde{\varphi}}_{n,q}\rangle \tilde{\tilde{\gamma}}_{n,q}. \tag{18}$$

One can show [9] that the Fourier transforms of the redressed frame are

$$\hat{\tilde{\tilde{\varphi}}}_{n,q}(f) = A(f)\hat{h}(\theta^{-1}(f)-qb)e^{-j2\pi n\vartheta_q(a\theta^{-1}(f))}, \tag{19}$$

where

$$A(f) = \sqrt{\tfrac{d\theta^{-1}}{df}}\left.\sqrt{\tfrac{d\vartheta_q}{d\nu}}\right|_{\nu=a\theta^{-1}(f)}. \tag{20}$$

Hence, dispersion is completely eliminated if

$$\vartheta_q(a\theta^{-1}(f)) = d_q f \tag{21}$$

for any $f \in \mathbb{R}$, where d_q are positive constants controlling the time scale in each frequency band. In this case, the Fourier transforms of the redressed frame elements become:

$$\hat{\tilde{\tilde{\varphi}}}_{n,q}(f) = \sqrt{\tfrac{d_q}{a}}\hat{h}(\theta^{-1}(f)-qb)e^{-j2\pi n d_q f}. \tag{22}$$

When all d_q are identical all the time samples are aligned to a uniform time scale throughout frequencies. If the d_q are distinct, time realignment when displaying the non-uniform spectrogram is a simple matter of different time base or time scale for each frequency band.

Unfortunately, due to the discrete nature of the redressing warping operation, each map ϑ_q is constrained to be congruent modulo 1 to a 1-periodic function, while the global warping map θ can be arbitrarily selected. Moreover, the functions ϑ_q must be one-to-one in each unit interval, therefore they can have at most an increment of 1 there.

In Fig. 1 we illustrate the phase linearisation problem. There, the black curve is the amplitude scaled warping map $d_q\theta(\nu)$ and the grey curve represents the map $\vartheta_q(a\nu)$, which is $1/a$-periodic. Both maps are plotted in the abscissa $\nu = \theta^{-1}(f)$. By amplitude scaling the warping map θ one can allow the values of the map to lie in the range of the discrete-time warping map ϑ_q. The amplitude scaling factors happen to be the new time sampling intervals d_q of the redressed warped Gabor expansion.

In the "painless" case, which was hand picked in [3], the window h is chosen to have compact support in the frequency domain. Through equation (19) and condition (21), the redressing method shows that, given any continuous and almost anywhere differentiable and increasing warping map, only in the painless case one can exactly eliminate the dispersive delays with the help of (17). In fact, in this case linearisation of the phase is only required within a

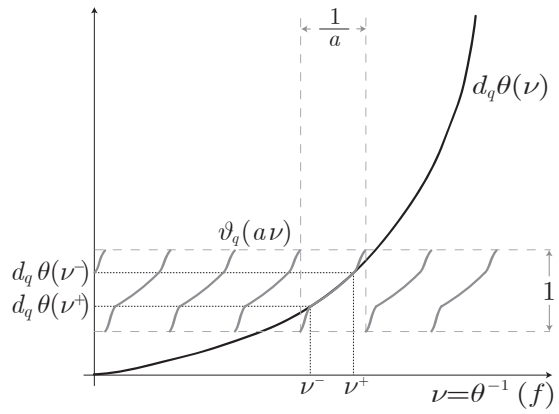


Figure 1: Locally eliminating dispersion by means of discrete-time frequency warping. Black line: curve derived from the original map θ by amplitude scaling. Gray line: discrete-time frequency warping characteristics for local phase linearisation.

finite frequency range given by the frequency support of the frame elements [8, 9], which is compatible with the periodicity constraint of the redressing maps ϑ_q.

In the general case, a perfect time realignment of the components is not guaranteed. Notice, however, that, by construction, the redressed warped Gabor systems are guaranteed to be frames for any choice of the maps ϑ_q satisfying the stated periodicity conditions, even when the phase is not completely linearised. Locally, within a certain band of the warped modulated windows it is possible to linearise the phase of the complex exponentials in (19). In the sequel we will refer to this band as the *essential bandwidth* since, hopefully, the magnitude of the Fourier transform of the window is negligible outside it, at least as a design goal.

Unfortunately, in general both the painless case and the partially redressed cases lead to infinite duration windows, which are undesirable when online computation is required. In the next we are going to propose some approximations and study the reconstruction error also through numerical experiments.

3. REAL-TIME COMPUTATION OF THE WARPED GABOR EXPANSION

For real-time computation one needs to make assumptions on the signal, as well on the window functions h in order to keep the computational load as low as possible.

By requiring the window h to be real-valued and θ to be odd, we obtain $\hat{\tilde{\varphi}}_{n,-q}(-f) = \overline{\hat{\tilde{\varphi}}_{n,q}(f)}$. If, additionally the input signal is real-valued, then the coefficients $c_{n,q} = \langle s, \tilde{\tilde{\varphi}}_{n,q}\rangle$ fulfil $c_{n,-q} = \overline{c_{n,q}}$ and thus we only need to compute about half the coefficients and frame elements. By enforcing shift-invariance of the warped frame elements (i.e. $\tilde{\tilde{\varphi}}_{n,q}(t) = \tilde{\tilde{\varphi}}_{0,q}(t - na)$), which we must do since otherwise the pre-computation of the frame elements and hence the real-time computation of the expansion would be impossible, it is sufficient to only store $\tilde{\tilde{\varphi}}_{0,q}$ for non-negative values of q. It is advisable that the warping map is selected as a continuously differentiable function, since the error in the resynthesised signal in the frequency domain at the points of discontinuity of θ' is very high. A strictly positive derivative helps the atoms not get too stretched in the time domain, which is undesirable in real time applications. To avoid aliasing we further require that $\theta(qb) = \mathrm{SR}/2$ and $\theta''(qb) = 0$ for some $q \in \mathbb{N}$, where SR denotes the sampling rate. This ensures that the high frequencies are smoothly mapped back to the negative frequencies and avoids extra aliasing introduced by the approximation. By requiring the Fourier transform of the window h to have an analytic expression, we can avoid numerical errors in the computation of the warped windows. We further only worked with windows which are dual to itself, i.e. $\varphi_{n,q} = \gamma_{n,q}$ which is actually not a necessary restriction.

3.1. Implementation Details

We implemented the approximate warped redressed Gabor analysis and synthesis as C externals interfaced to Pure-Data (32 bit and 64 bit). It runs both under Windows and Linux and can use multiple cores. The warped windows are computed using equation (22) and applying an IDFT of that data. The inner products $c_{n,q}$ are directly computed with a loop. Also for the synthesizing part, the sum $\sum_{n,q} c_{n,q}\tilde{\tilde{\varphi}}_{n,q}$ is directly computed. It turned out that this is sufficiently fast for real-time computation and whence we made no use of fast convolution algorithm. Since the data rate of the analysis part is not uniform in time, PD's signal connections are not suitable to transfer the coefficients. Therefore we use the signal connections to transfer pointers rather than signals.

3.2. Interface Details

For simplicity we require that the essential bandwidth B is a multiple $K \in \mathbb{N}$ of the frequency shift parameter b, i.e. $\mathrm{B} = Kb$. Then, in order to obtain a frame the following conditions must be fulfilled as can be seen easily in Figure 1 [9, eq. (34) to (36)]:

$$\begin{aligned} abK &\leq 1 \\ d_q \mathrm{B}_q &\leq 1 \end{aligned} \tag{23}$$

where B_q is the ess. bandwidth of the warped modulated window.

$$\mathrm{B}_q = \theta\left(qb + \frac{\mathrm{B}}{2}\right) - \theta\left(qb - \frac{\mathrm{B}}{2}\right) \tag{24}$$

In the case of an exponential increasing warping map, it makes sense to set the frequency shifting parameter in a way that adjacent notes fall away from the windows main lobe in the frequency domain (If there is such a main lobe, as in the case of the raised cosine window). This on the other hand determines the window length T_h, a minimal value for R and the time shift parameter

$$a = \frac{T_h}{R} \tag{25}$$

with $R \geq K$ due to equation (23). We set $R = K$ for simplicity. Equations (23) are fulfilled by setting

$$\begin{aligned} b &= \frac{1}{aRC_b} \\ d_q &\simeq \frac{1}{\mathrm{B}_q C_d} \end{aligned} \tag{26}$$

where C_b, C_d can be chosen inside the PD-patch. C_b together with R controls the oversampling, where C_d controls the bandwidth in

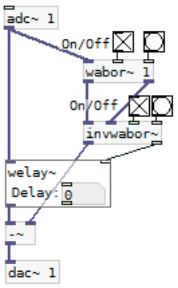


Figure 2: This is a sample patch for the use of our Pure-Data Implementation. *wabor~* and *invwabor~* are for analysing and synthesising. They are connected with two signal-paths. *invwabor~* holds at the right outlet the delay in samples due to the analysis-synthesis procedure. *welay~* is a simple delay line. If perfect reconstruction is achieved, the output at *dac~* is zero. With the toggles the externals can be switched on and off. The bangs are for outputting the parameters used.

which the phase is linearised and also influences the oversampling. The d_q are different for each band which results in a non-uniform data rate for each band. We remark that the numbers d_q have to be chosen, such that $d_q/\mathrm{SR} \in \mathbb{N}$, where SR is the sampling rate.

The number of bands q_{sup} we need for a given SR is

$$q_{\mathrm{sup}} = \frac{\theta^{-1}(SR/2)}{b}. \tag{27}$$

The assumptions on the warping map ensure that this is a natural number.

Due to the warping, the supports of the windows are in general unbounded. Thus we compute the windows with a zero padded array of length of T_c (c for compute), which is defined as

$$\mathrm{T}_c = \frac{\mathrm{T}_h}{\theta'_{\mathrm{inf}}} C_{\mathrm{T}_c} \tag{28}$$

where $C_{\mathrm{T}_c} \geq 1$ can be chosen inside the PD-patch and $\theta'_{\mathrm{inf}} = \inf_{f\in\mathbb{R}} \theta'(f)$. Due to the same reason it is indispensable to cut the windows after warping. To define a sensible atom length T_q after warping, we set the atoms to zero after the point from which they were smaller than $\left|\tilde{\tilde{\varphi}}_q(t)\right| < 1/C_{\mathrm{cut}} \max_t \left|\tilde{\tilde{\varphi}}_q(t)\right|$, with C_{cut} a constant which can be chosen inside the PD-patch. Afterwards the windows are truncated accordingly and with respect to the parameter $\mathrm{T}_{\max}$, which defines the maximal desired window length. It is important that the windows are cut after aligning them to the time origin. The second approach suggested in [9] to obtain windows with finite length by computing only a linearised version of the warped windows, leads to very bad reconstruction.

3.3. Computational Costs

3.3.1. Analysis

A rough estimation yields the following. Let T_q denote the length of the q^{th}-window . It will be clear from the context if T_q denotes seconds or samples. To compute the inner product of that window with the signal one needs $2\mathrm{T}_q$ many real multiplications and summations. This has to be done every d_q samples. Hence, per sample we have $4\mathrm{T}_q/d_q$ many floating point operations (flops) in average. If the essential bandwidth is not too large, then T_q is proportional

Abbrev.	Explanation
SR	Sampling Rate
B_q	Essential bandwidth of the q^{th} window
T_c, C_{T_c}	Length of the window used in precomputation
T_h	Length of the original window
T_q	Length of the qth window
$\mathrm{T}_{\max}$	Maximal window length used for real time computation
C_{cut}	Parameter controlling the truncation of the windows in the time domain after precomputing
b, C_b	Frequency shift parameter
d, C_d	Time shift parameter of the warped windows
q_{sup}	Number of bands

Figure 3: Summary of the variables used in the PD implementation. The parameters C_X control their variable X.

to $\mathrm{T}_0/\theta'(qb)$ and d_q is proportional to $1/(Kb\theta'(qb))$ since, linearising θ around qb yields $\theta(qb + \frac{Kb}{2}) \simeq \theta(qb) + \theta'(qb)\frac{Kb}{2}$ and hence

$$\mathrm{B}_q = \theta\left(qb + \frac{Kb}{2}\right) - \theta\left(qb - \frac{Kb}{2}\right) \simeq \theta'(qb)Kb$$
$$\Rightarrow d_q = \frac{1}{\mathrm{B}_q} \simeq \frac{1}{\theta'(qb)Kb} \tag{29}$$

Summing up over all windows we get the average number of operations per sample N_{avg}:

$$N_{\mathrm{avg}} = \sum_q \frac{4\mathrm{T}_q}{d_q} \sim \sum_q \frac{4\mathrm{T}_0}{\theta'(qb)} \frac{Kb\theta'(qb)}{1} = 4Kbq_{\mathrm{sup}}\mathrm{T}_0 \tag{30}$$

Where q_{sup}, defined above, denotes the number of bands and depends on θ^{-1}. That means the complexity of the analysis part is proportional to T_0, K and b and $q_{\mathrm{sup}} \sim \theta^{-1}(SR)$.

If there is a lower bound for the d_q's, one can choose identical numbers for all bands. However, this can increase the computational load tremendously, making real time computation impossible.

The above is an estimation of the average computational cost. In the worst case all inner products of the windows with the signal have to be computed starting at one frame. The number of operations for that frame is

$$\sum_q 4\mathrm{T}_q \simeq 4\sum_q \frac{\mathrm{T}_0}{\theta'(qb)} \tag{31}$$

However, if one processes the audio stream block wise, the worst case cannot arise for all samples in the block at once, because two worst case scenarios have a distance of at least $M = \max_q d_q$ (actually $M = \Pi_q d_q$). The next M samples have for sure lower computational cost. Furthermore, the next $m = \min_q d_q$ samples have zero computational cost. Therefore the average costs are a suitable measure for the analysis process.

3.3.2. Synthesis Part

The complexity of the synthesis part is the same as that of the analysis part. Furthermore, in the synthesis part the worst case scenario can be avoided, because only the parts of the next frame buffer have to be computed in real-time, the rest can be computed later. Nevertheless, our given implementation is not optimized in this direction.

3.3.3. Memory Costs

Our algorithm for precomputing the windows needs $2q\mathrm{T}_0$ cells. With slight modifications it only needs $\mathrm{T}_0 + \sum_q \mathrm{T}_q$. The smallest possible number of cells being needed is $\sum_q \mathrm{T}_q$. The analysis and synthesis algorithm both need at least a buffer of size *Audiobuffer* $+ \mathrm{T}_{\max}$, where $\mathrm{T}_{\max}$ denotes the window length of the longest windows used in the analysis and synthesis, and *Audiobuffer* the buffer length in which the audio is processed.

The frame elements for our tests below needed about 65 MB.

4. COMPUTATIONAL ERROR

The measured err-numbers are the difference between the averaged RMS-amplitude in dB of the input signal and the analysed-synthesised output signal (i.e. negative signal to noise ratio). For comparison: 16 bit quantization (which is CD-quality) has $err \simeq -96$ dB, 8 bit quantization has $err \simeq -50$ dB. The tests were conducted with the PD-patch available at tommsch.com/science.php in real-time over a time of about 20 s with double precision floating point numbers and a sample rate of 44.1 kHz. We used the following stationary and non-stationary test signals

- **white:** White noise
- **sine X:** A pure sine tone with X Hz
- **const:** A constant signal
- **clicks:** Clicks with a spacing of 1 s
- **beet:** Beethoven - Piano Sonata op 31.2, length 25 s.
- **speech:** A man counting from one to twenty, length 20 s.
- **fire:** A firework, length 23 s.
- **atom:** A sample of sparse synthesised warped Gabor atoms which were also used for that specific test run.

Since our method would lead to perfect reconstruction if the windows were time-shift invariant, the behaviour of the algorithm for stationary signals over a long time is of greatest interest. The constant signal is of interest since it is the one with the lowest possible frequency and our algorithm may bear problems with low frequencies due to the necessary cutting. The clicks represent the other extreme point of signals. The atoms are of interest since they shows whether our algorithm has the ability to reproduce its atoms with high quality.

For the warping map, we used a function which is exponentially increasing between two frequencies f_{in} and f_{out}, namely $\theta(f) = f_0 2^{-f/k}$ where $f_0, k \in \mathbb{R}$ are constant parameters which can be set inside the PD-patch. For all the tests we set $f_0 = 12$ and $k = 36$. Outside of $\pm f_{out}$ and between $\pm f_{in}$ the map is linear. The function attains exactly Nyquist frequency, i.e. at $\theta((q_{\sup} - 1)b) = \mathrm{SR}/2$. The frequencies f_{in}, f_{out} are chosen such, that the resulting map is C^1. See Figure 4 for a plot of θ^{-1}.

4.1. Raised Cosine Window

As proposed in [9] we first performed tests with a raised cosine window.

$$h(t) = \begin{cases} \sqrt{\frac{2b}{R}} \cos \frac{\pi t}{\mathrm{T}} & -\frac{\mathrm{T}}{2} \le t \le +\frac{\mathrm{T}}{2} \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

with Fourier-Transform

$$\hat{h}(\nu) = \sqrt{\frac{b}{2R}} \Big(\mathrm{sinc}(\nu \mathrm{T} - \tfrac{1}{2}) + \mathrm{sinc}(\nu \mathrm{T} + \tfrac{1}{2}) \Big) \tag{33}$$

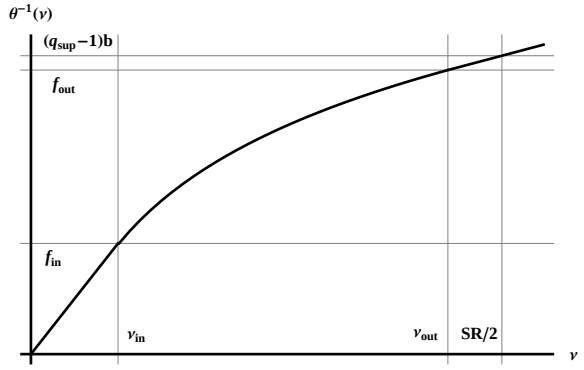


Figure 4: The used warping map θ. It is C^1, linear between $\pm f_{in}$ and outside of $\pm f_{out}$ and passes through $SR/2$.

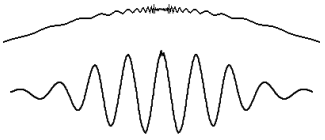


Figure 5: The picture shows a warped raised cosine window in the time domain. The upper part is a magnification around zero. Ripples, due to the slow decay in the frequency domain, are clearly visible.

where T is the total duration of the window, $R \ge 3$ is an integer, $f_0 = \frac{3}{2\mathrm{T}}$, $a = \frac{3}{2Rf_0}$, $b = \frac{2f_0}{R}$. See [9, Section 4] for an ansatz about how to compute these parameters. This window has a very slow decay in the frequency domain after warping:

$$\hat{\tilde{\varphi}}_{0,q}(f) = \sqrt{\frac{d_q}{a}} \hat{h}(\theta_e^{-1}(f) - qb) \simeq \frac{1}{\log_2 f} \tag{34}$$

This means that either the warped windows are not bounded anymore (i.e.: $\tilde{\tilde{\varphi}}_q \notin L^\infty(\mathbb{R})$) or that they are discontinuous, which is clearly visible in the warped windows, a plot of one window can be seen in Figure 5. Hence the computation of the warped windows with the IDFT bears numerical errors. Also the test results with this window were suboptimal, yielding an error between -50 dB and -60 dB, depending on the chosen parameters and input signal.

Changing the parameter R or the parameter C_d had a significant effect on the error. In Figure 6 on can see the influence of R. This can be expected, since the essential bandwidth, in which the phase is linearized, depends on that parameter. Since oversampling is directly proportional to the computational load (as computed in (30)), for real-time applications there is a natural upper bound. $C_d \approx 2,5$ provided good results. Too small and too big numbers for C_d both gave worst results.

On the contrary, changing the parameter b while preserving the oversampling factor R and the parameter a had no effect, as long $C_b \le 2$.

The relation between the window length after cutting and the error was nearly proportional to the value of $\sum_q \mathrm{T}_q$ in a certain range, see Figure 8. From this observation one can determine a suitable cutting parameter C_{cut}. Changing the parameter $\mathrm{T}_{\max}$ has clearly only an influence on the err for low frequencies.

$4\sum \mathrm{T}_q/d_q$	R	q_{sup}	b	err
13.1k	2,0	67	6 Hz	-74.4
22.1k	2,5	83	4,8 Hz	-86.4
25.9k	3,0	99	4 Hz	-94.4

Figure 6: Influence of the parameter R on the error. Gaussian Window, B=24 Hz, R, q_{sup}, b=variable, T=0.167 s, $C_d = 2$, $\mathrm{T}_{\mathrm{max}}$=0.629 s, C_{T_c}=2, a=0.04167 s, $C_{\mathrm{cut}} = 20$, test signal: white noise. Values in dB.

$\mathrm{T}_c/\mathrm{T}_h$	Raised Cosine	Gaussian
1,0	$-63,1$	$-62,8$
1,2	$-66,9$	$-66,1$
1,4	$-70,2$	$-71,3$
1,6	$-71,0$	$-73,3$
1,8	$-71,1$	$-76,0$
2,0	$-71,1$	$-77,7$
2,5	$-71,1$	$-78,3$
5,0	$-71,0$	$-78,3$
16,0	$-71,0$	$-71,5$

Figure 7: Influence of the parameter T_c on the error for the Gaussian and the raised cosine window. Parameters as in Figure 10 and 9. Only the Parameter C_{T_c} was changed. Signal: White noise. Values in dB

The parameter C_{T_c} had no big influence as long it was about T_c was about twice the window length T_h, see Figure 7

The warping map has a very big influence on the error. At points where the map is not smooth the error for these frequencies is order of magnitudes higher. This can be seen in Figures 9 and 10 for the sine with 30 Hz signal. At this point, our used warping map is only C^1. For a C^0 warping map the error was again 10 dB higher.

The table in Figure 9 shows selected numerical results with well chosen parameters for the raised cosine window. All values in dB, values above -60 dB and below -70 dB are colored. The number $4\sum \mathrm{T}_q/d_q$ denotes the average computational average complexity (see above).

4.2. Gaussian Window

In order to obtain a window with proper decay in the frequency domain after warping, we used a Gaussian window. This window does not overlap-add to one. Hence higher overlap is necessary to minimize the deviation from one. The warped windows were still fast decaying in the time domain, resulting in the possibility to cut them much shorter then the warped raised-cosine windows which compensated the high computational load due to the high overlap.

Our used Gaussian window and its Fourier Transform is

$$h(t) = C\sqrt{\frac{b}{2R}}e^{-\frac{1}{4}\frac{t^2}{\mathrm{T}^2}}, \quad \hat{h}(t) = C\,\mathrm{T}\sqrt{\frac{b}{R}}e^{-t^2\mathrm{T}^2} \tag{35}$$

where T is the total duration of the window, $R \geq 2$ the overlap factor is an integer, $f_0 = \frac{R}{2\mathrm{T}}$, $a = \frac{\mathrm{T}}{R}$, $b = \frac{1}{aR}$ and $C \simeq 0.893249\cdots$ is a constant factor used to approximate the overlap-add to one condition.

This window has a fast enough decay to ensure that the warped windows in the frequency domain still are in $L^1(\mathbb{R})$ and hence their Inverse Fourier transformed (i.e. the warped windows in the time domain) are bounded and continuous. This window leads to significantly better results down to -96 dB which is the limit when using PD's single precision numbers. The influence of the parameters on the error was the same as for the raised-cosine window.

The tables in Figure 10 show selected numerical results.

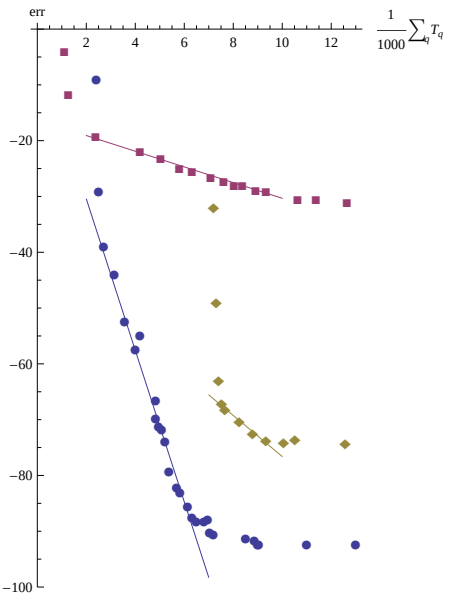


Figure 8: Influence of the truncation length of the computed window. Test signal: white noise. Red squares and yellow diamonds: Raised Cosine Window with different parameters, blue dots: Gaussian window. C_{cut}=variable, all other parameters are fixed. The value of $\sum \mathrm{T}_q$ is directly proportional to the error in a certain range.

signal	err	signal	err
white	$-70,6$	clicks	$-57,0$
sine 30	$-53,3$	beet	$-66,8$
sine 440	$-63,3$	speech	$-61,9$
sine 20k	$-76,9$	fire	$-62,8$
const	$-84,7$	atom	$-62,7$

Figure 9: Test results for the raised cosine window. B=24 Hz, R=7, q_{sup}=229, T=0.30 s, T_c=1.4 s, $\mathrm{T}_{\mathrm{max}}$=0.5 s, a=0.04167 s, b=1.714 Hz, C_b=2, C_d=4, C_{cut}=55, $4\sum \mathrm{T}_q/d_q = 33.6$k. Values in dB.

signal	err	signal	err
white	$-71,3$	clicks	$-57,3$
sine 30	$-61,5$	beet	$-70,2$
sine 440	$-70,0$	speech	$-69,8$
sine 20k	$-81,2$	fire	$-68,4$
const	$-63,7$	atom	$-69,1$

Figure 10: Test results for the Gaussian window. B=24 Hz, R=4, q_{sup}=132, T=0.167 s, T_c=0.8 s, $\mathrm{T}_{\mathrm{max}}$=0.4 s, a=0.04167 s, b=3 Hz, C_b=2, C_d=2, C_{cut}=1000, $4\sum \mathrm{T}_q/d_q = 7.3$k. Values in dB.

4.3. Comparison of these two windows

For the raised cosine a high number of bands must be used to achieve a small error. In Figure 8 the tests of the red dots are conducted using 92 bands, the tests with yellow dots were with 229 bands. Since the windows have a bad decay in the frequency domain, a high essential bandwidth has to be used too, which entails big overlap in time. and hence a very high average computational load, in our example 33.6k floating point operations per sample. If one uses similar parameters to the ones used for the Gaussian window in Figure 10, the error is in the range of -36 dB.

Gaussian windows on the other hand do not overlap add to one and hence are not dual to themselves. Hence we at first used a very high overlap to minimize the deviation from 1, which turned out to be not necessary later. The fast decay in the time domain allows to cut the windows much shorter than the raised cosine window. This decreases the computational load, in our example only 7.3k flops per sample in average. In Figure 8 one can see that for the Gaussian window, with proper parameters, the error is in the range of -90 dB.

5. CONCLUSIONS

We have introduced a novel, flexible, easy way to construct frames starting from a classical Gabor frame. Tests show, that even in the *painful case*, where perfect time realignment of the components is not guaranteed and hence our method does not lead to perfect reconstruction, the error can be made as small as the accuracy of single precision floating point numbers for a wide range of signals. This does not prove that the error is small for all signals, but gives a good estimation of the error to be expected. The transform is working in real time.

We are going to implement maps that can be arbitrarily defined, e.g., by means of interpolation of a selected number of points. We will also implement Gabor Multipliers [17] in the redressed warped Gabor expansion (partially done already with the external *winary˜*).

Since we already implemented this method as a Pure Data external, it is ready to use for audio-applications. The whole external explained in detail as well as the source code can be found online at tommsch.com/science.php and in [18].

6. ACKNOWLEDGMENTS

Many thanks to the great number of suggestions by the great number of anonymous reviewers on how to improve the paper.

7. REFERENCES

[1] G. A. Velasco, N. Holighaus, M. Dörfler, and T. Grill, "Constructing an invertible constant-Q transform with nonstationary Gabor frames," in *Proceedings of the Digital Audio Effects Conference (DAFx-11)*, Paris, France, 2011, pp. 93–99.

[2] P. Balazs, M. Dörfler, F. Jaillet, N. Holighaus, and G. A. Velasco, "Theory, implementation and applications of nonstationary Gabor Frames," *Journal of Computational and Applied Mathematics*, vol. 236, no. 6, pp. 1481–1496, 2011.

[3] G. Evangelista, M. Dörfler, and E. Matusiak, "Phase vocoders with arbitrary frequency band selection," in *Proceedings of the 9th Sound and Music Computing Conference*, Copenhagen, Denmark, 2012, pp. 442–449.

[4] N. Holighaus and C. Wiesmeyr, "Construction of warped time-frequency representations on nonuniform frequency scales, Part I: Frames," *ArXiv e-prints*, Sep. 2014.

[5] T. Twaroch and F. Hlawatsch, "Modulation and warping operators in joint signal analysis," in *Time-Frequency and Time-Scale Analysis, 1998. Proceedings of the IEEE-SP International Symposium on*, Oct. 1998, pp. 9–12.

[6] R. G. Baraniuk and D. L. Jones, "Warped wavelet bases: unitary equivalence and signal processing," in *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, Apr. 1993, vol. 3, pp. 320–323 vol.3.

[7] C. Braccini and A. Oppenheim, "Unequal bandwidth spectral analysis using digital frequency warping," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 22, no. 4, pp. 236–244, Aug. 1974.

[8] G. Evangelista, "Warped Frames: dispersive vs. non-dispersive sampling," in *Proceedings of the Sound and Music Computing Conference (SMC-SMAC-2013)*, Stockholm, Sweden, 2013, pp. 553–560.

[9] G. Evangelista, "Approximations for Online Computation of Redressed Frequency Warped Vocoders," in *Proceedings of the Digital Audio Effects Conference (DAFx-14)*, Erlangen, Germany, 2014, pp. 1–7.

[10] K. Gröchenig, *Foundations of Time-Frequency Analysis*, Applied and Numerical Harmonic Analysis. Birkhäuser Boston, 2001.

[11] R. G. Baraniuk and D. L. Jones, "Unitary equivalence : A new twist on signal processing," *IEEE Transactions on Signal Processing*, vol. 43, no. 10, pp. 2269–2282, Oct. 1995.

[12] P. W. Broome, "Discrete orthonormal sequences," *Journal of the ACM*, vol. 12, no. 2, pp. 151–168, Apr. 1965.

[13] L. Knockaert, "On Orthonormal Muntz-Laguerre Filters," *IEEE Transactions on Signal Processing*, vol. 49, no. 4, pp. 790–793, Apr. 2001.

[14] G. Evangelista, "Dyadic Warped Wavelets," *Advances in Imaging and Electron Physics*, vol. 117, pp. 73–171, Apr. 2001.

[15] G. Evangelista and S. Cavaliere, "Frequency Warped Filter Banks and Wavelet Transform: A Discrete-Time Approach Via Laguerre Expansions," *IEEE Transactions on Signal Processing*, vol. 46, no. 10, pp. 2638–2650, Oct. 1998.

[16] G. Evangelista and S. Cavaliere, "Discrete Frequency Warped Wavelets: Theory and Applications," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 874–885, Apr. 1998, special issue on Theory and Applications of Filter Banks and Wavelets.

[17] Hans G. Feichtinger and Krzysztof Nowak, *A First Survey of Gabor Multipliers*, pp. 99–128, Birkhäuser Boston, Boston, MA, 2003.

[18] Thomas Mejstrik, "Real time computation of redressed frequency warped gabor expansion," Master thesis, University of Music and Performing Arts Vienna, tommsch.com/science.php, 2015.

REAL-TIME SPECTROGRAM INVERSION USING PHASE GRADIENT HEAP INTEGRATION

Zdeněk Průša *

Acoustics Research Institute,
Austrian Academy of Sciences
Vienna, Austria
zdenek.prusa@oeaw.ac.at

Peter L. Søndergaard

Oticon A/S
Smørum, Denmark
peter@sonderport.dk

ABSTRACT

The knowledge of the phase of STFT is a prerequisite for a successful signal reconstruction. However, the phase might be lost or no longer applicable depending on the kind of processing involved.

We propose a real-time spectrogram inversion algorithm based on the relationship of the gradients of the phase and the logarithm of the magnitude and on the gradient integration theorem. We present a detailed comparison with the state-of-the-art phase reconstruction algorithms.

1. INTRODUCTION

The first algorithm for the signal reconstruction from the (modified) STFT magnitude was introduced by Griffin and Lim in [1] (Griffin-Lim Algorithm – GLA) more than 30 years ago. Since then, several other algorithms have been developed, but the fact the algorithms typically require many iterations acting on the whole signal prohibited their widespread use e.g. as an alternative reconstruction procedure for the phase vocoder [2]. Other possible applications include e.g. compression, source separation, channel mixing, adaptive filtering and denoising. See e.g. [3] for a detailed overview of the reconstruction algorithms and applications.

A real-time version of GLA was introduced in [4] (Real Time Spectrogram Inversion Algorithm with Look Ahead – RTISI-LA). The algorithm is still iterative, but the signal is reconstructed frame-by-frame using a clever phase initialization such that only few iterations are necessary in order to get a good result. The downside of this algorithm is that it requires several "look-ahead" frames which increases the processing delay. Modifications of RTISI-LA were proposed in [5, 6, 7].

Recently, another real-time capable algorithm was introduced in [8] (Single Pass Spectrogram Inversion – SPSI). The algorithm is based on the notion of *phase consistency* introduced in connection with the phase-locked vocoder. Assuming the signal consists of sum of sinusoidal components, their phase grows in time at the rate of their instantaneous frequencies. In the algorithm, the instantaneous frequency is estimated in each frame by peak picking and quadratic interpolation. Thanks to this, the algorithm does not introduce any additional delay, but, on the other hand, it cannot properly handle any deviation from the model assumptions e.g. the quality of reconstructed transients and impulse-like components is poor.

The proposed algorithm (Real-Time Phase Gradient Heap Integration – RTPGHI) is based on the STFT phase-magnitude relationship first introduced in [9]. The offline version of the present algorithm (PGHI) along with the theoretical background has already been presented in [10]. This paper focuses on adapting the algorithm to the real-time setting. In the basic form it requires one look-ahead frame, but even zero delay can be achieved at a cost of a performance degradation.

The paper is organized as follows: section 2 contains a short theoretical introduction while section 3 presents the phase reconstruction algorithm itself. The paper is concluded with section 4 where the performance evaluation can be found.

* This work was supported by the Austrian Science Fund (FWF) START-project FLAME ("Frames and Linear Operators for Acoustical Modeling and Parameter Estimation"; Y 551-N13).

2. THEORY SUMMARY

The STFT of f with respect to a real valued window g is usually defined as

$$(\mathcal{V}_g f)(\omega,t) = \int_{\mathbb{R}} f(\tau+t)g(\tau)\mathrm{e}^{-\mathrm{i}2\pi\omega\tau}\,\mathrm{d}\tau,\ \ \omega,t\in\mathbb{R} \tag{1}$$

$$= M_g^f(\omega,t)\mathrm{e}^{\mathrm{i}\Phi_g^f(\omega,t)}, \tag{2}$$

and the *spectrogram* as the modulus squared. The (complex) logarithm separates the modulus and the phase such as

$$\log(\mathcal{V}_g f)(\omega,t) = \log M_g^f(\omega,t) + \mathrm{i}\Phi_g^f(\omega,t). \tag{3}$$

The Gaussian window with time-frequency support ratio γ

$$\varphi_\gamma(t) = \left(\frac{\gamma}{2}\right)^{-\frac{1}{4}} \mathrm{e}^{-\pi\frac{t^2}{\gamma}} \tag{4}$$

is known to posses optimal properties and, moreover, to allow algebraic treatment of the formulas. In particular, it has been shown that the phase gradient

$$\nabla\Phi_{\varphi_\gamma}^f(\omega,t) = \left(\tfrac{\partial\Phi_{\varphi_\gamma}^f}{\partial\omega}(\omega,t), \tfrac{\partial\Phi_{\varphi_\gamma}^f}{\partial t}(\omega,t)\right) \tag{5}$$

and the gradient of the log-magnitude relate to each other in the following way [9, 11, 10]

$$\frac{\partial\Phi_{\varphi_\gamma}^f}{\partial\omega}(\omega,t) = -\gamma\frac{\partial}{\partial t}\log(M_{\varphi_\gamma}^f(\omega,t)) \tag{6}$$

$$\frac{\partial\Phi_{\varphi_\gamma}^f}{\partial t}(\omega,t) = \frac{1}{\gamma}\frac{\partial}{\partial\omega}\log(M_{\varphi_\gamma}^f(\omega,t)) + 2\pi\omega. \tag{7}$$

In theory, the knowledge of the original phase at a single point $\Phi_{\varphi_\gamma}^f(\omega_0,t_0)$ and the *gradient theorem* would be sufficient to recover the original phase using

$$\Phi_{\varphi_\gamma}^f(\omega,t) = \int_0^1 \nabla\Phi_{\varphi_\gamma}^f\left(L(\tau)\right)\cdot\frac{\mathrm{d}L}{\mathrm{d}\tau}(\tau)\,\mathrm{d}\tau + \Phi_{\varphi_\gamma}^f(\omega_0,t_0), \tag{8}$$

where $L(\tau) = [L_\omega(\tau), L_t(\tau)]$ is any line $(\omega_0, t_0) \to (\omega, t)$. When the phase is unknown completely, one obtains a global constant phase shift. In case of real signals, the global phase shift turns into the reconstructed signal sign ambiguity.

In practice however, and in the real-time setting in particular, several factors make the direct application of such result difficult. First and foremost, the discretization of STFT inevitably introduces aliasing in some form. This causes the relations (6) and (7) not to hold exactly everywhere. Second, one can only work with truncated Gaussian window or with other finitely supported windows, for which the relations hold only approximately. Third, the numerical line integration is prone to error propagation. As a result, the reconstruction algorithm typically produces a phase error with "patches" of constant phase shifts, which is common for all algorithms available. Fig. 1 shows an example of such an error using an excerpt from the *glockenspiel* test signal. It depicts the absolute difference between the original and the reconstructed phase (modulo 2π) taking the circular nature of the phase into account i.e. taking the shorter distance on the circle. The phase error is in rad$/\pi$ and it was set to zeros for very small coefficients.

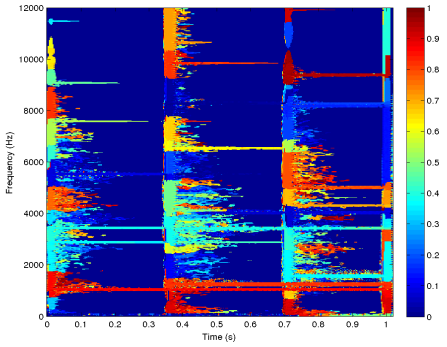


Figure 1: Typical phase difference pattern for the proposed algorithm.

Luckily, it seems that such relative phase shifts of time-frequency components do not degrade the perceived quality substantially.

Even more ambiguity enters in case of modified or completely synthetic spectrograms, for which it is not clear whether (6) and (7) hold at all.

3. THE ALGORITHM

The goal of the present algorithm is to exploit (6),(7) and (8) to recover phase from the magnitude and, ultimately, to reconstruct the original signal.

The first step is obtaining the phase gradient via numerical differentiation and the second one is the numerical line integration on the rectangular grid.

3.1. Phase Gradient Approximation

The discrete version of STFT is defined as

$$(V_g f)(m,n) = \sum_{l\in\mathcal{I}} f(l+na)g(l)\mathrm{e}^{-\mathrm{i}2\pi ml/M} \tag{9}$$

$$= s(m,n)\mathrm{e}^{\mathrm{i}\phi(m,n)}, \tag{10}$$

where $f \in \ell^2(\mathbb{Z})$ and $g \in \ell^2(\mathbb{Z})$ are both real sequences, $m = 0, \ldots, \lfloor M/2 \rfloor$ where M is the number of frequency channels, $n \in \mathbb{Z}$ and the parameter a is the time step in samples. The finite support of g will be bounded such that it is zero outside of the sum index set $\mathcal{I} = \{-\lceil M/2 \rceil + 1, \ldots, \lfloor M/2 \rfloor\}$. The windows will be further restricted to be whole point symmetric such that their discrete-time Fourier transform is real.

The time-frequency ratio γ of the Gaussian window which is closest to the given window g can be obtained as

$$\gamma = C_g \cdot \mathrm{len}(g)^2, \tag{11}$$

where $\mathrm{len}(g)$ determines the length of the window support in samples and the constant C_g is window specific. The values for windows used in this paper can be found in Table 1. They were obtained by a simple heuristic search.

Table 1: C_g constants for (12) for common windows

Hann	0.25645
Hamming	0.29794
Blackman	0.17954

It is also possible to use a truncated discrete Gaussian window $\varphi_\gamma^{\mathrm{T}}$. Assume the window is truncated at the relative height h and it is required to be w samples long; then the γ parameter is obtained as

$$\gamma = -\frac{\pi}{4}\frac{w^2}{\log(h)}. \tag{12}$$

The (scaled) discrete STFT phase gradient $\nabla\phi = (\phi_\omega, \phi_t)$ can be approximated using centered difference scheme

$$\phi_\omega(m,n) = -\frac{\gamma}{2aM}\left(s_{\log}(m,n+1) - s_{\log}(m,n-1)\right), \tag{13}$$

$$\phi_t(m,n) = \frac{aM}{2\gamma}\left(s_{\log}(m+1,n) - s_{\log}(m-1,n)\right) + 2\pi am/M, \tag{14}$$

assuming $s_{\log}(m,n) = \log\left(s(m,n)\right)$ and setting $\phi_t(0,n)$ and $\phi_t(\lfloor M/2 \rfloor, n)$ to zeros. The use of the centered difference scheme ensures the correct alignment of the sampling grids of both the gradient components with the sampling grid of the coefficients. The gradient is scaled such that the lengths of steps in the following numerical integration scheme are equal to 1 in both directions. Alternatively, a causal finite difference scheme can be used for computing ϕ_ω

$$\phi_\omega^{\mathrm{causal}}(m,n) = -\frac{\gamma}{2aM}\left(3s_{\log}(m,n) - 4s_{\log}(m,n-1) + s_{\log}(m,n-2)\right). \tag{15}$$

In general, the misalignment of the sampling grids however introduces a performance degradation.

3.2. Real-Time Heap Integration

The gradient integration is done using the cumulative sum and the trapezoidal rule. The integration paths are chosen adaptively according to the coefficient magnitude following the direction of the ridges fist. The algorithm actually does not use integration paths

directly because only a single coefficient and its immediate neighbors are needed at a time.

Another simplification comes from the fact that the integration is only done in the horizontal or in the vertical directions which makes the (vector) derivative of the line segment ($\frac{\mathrm{d}L}{\mathrm{d}\tau}$ from (8)) to have only one nonzero element. Therefore, only one element of the gradient is active at a time during the integration.

For the purpose of keeping track of the coefficients with already computed phase, the algorithm employs a *heap* data structure which always keeps the coefficient with the largest magnitude at the top. The heap data structure is equipped with efficient insertion and deletion operations. The algorithm further accepts a relative magnitude threshold tol which controls which coefficients are included in the integration. The coefficients below tol are assigned a random phase value. The reasoning for this choice is that when tol is high, the random phase causes less disturbing artifacts when compared to zero phase. The algorithm is summarized as Alg. 1.

In words (assuming $tol = 0$ for simplicity), the algorithm starts of by marking coefficients from the n frame as unknown (line 2) and continues by inserting coefficients from the $n-1$ frame into the *heap* (line 5) making them all potential initial points. The integration itself starts by removing the biggest coefficient from the heap (line 8) and it is used to spread the phase to the only neighbor in the n frame (line 11). The just computed coefficient is marked as known (line 12) and it is then inserted into the heap (line 13) to serve as a potential phase source. The algorithm then continues by selecting and removing the biggest coefficient from the heap and this time, the coefficient from frame n might have been selected (line 16) and in that case, the phase is spread to the neighbors above (line 18) and below (line 23) and both are marked as known and they are inserted into the heap. The algorithm continues until no coefficients with unknown phase are left.

The reconstructed phase $\widehat{\phi}(m,n)$ is combined with the magnitude such that $\widehat{c}(m,n) = s(m,n)\mathrm{e}^{\mathrm{i}\widehat{\phi}(m,n)}$ and the signal $\widehat{f}$ is reconstructed using a dual window and the overlap-add procedure.

In some cases, the phase is partially known. Such information can be easily exploited by altering the algorithm such that the line 5 in Alg. 1 is repeated with indices (m,n) of the coefficients with the known phase of the current frame n.

Algorithm 1: Phase Gradient Heap Integration for n-th frame

Input: Phase time derivative $\phi_t(m,n)$ and magnitude $s(m,n)$ of frames n and $n-1$, phase frequency derivative $\phi_\omega(m,n)$ for frame n, estimated phase $\widehat{\phi}(m,n)$ for frame $n-1$ and relative tolerance tol.

Output: Phase estimate $\widehat{\phi}(m,n)$ for frame n.

1 $abstol \leftarrow tol \cdot \max\big(s(m,n) \cup s(m,n-1)\big)$;
2 Create set $\mathcal{I} = \big\{(m,n) : s(m,n) > abstol\big\}$;
3 Assign random values to $\widehat{\phi}(m,n)_{(m,n)\notin\mathcal{I}}$;
4 Construct a self-sorting *heap* for (m,n) tuples;
5 Insert $(m,n-1)$ for $m = \big(m : s(m,n-1) > abstol\big)$ into the *heap*;
6 **while** $\mathcal{I}$ *is not* $\emptyset$ **do**
7 **while** heap *is not empty* **do**
8 $(m_{heap}, n_{heap}) \leftarrow$ remove the top of the *heap*;
9 **if** $n_{heap} == n-1$ **then**
10 **if** $(m_{heap}, n) \in \mathcal{I}$ **then**
11 $\widehat{\phi}(m_{heap},n) \leftarrow \widehat{\phi}(m_{heap},n-1) + \frac{1}{2}\big(\phi_t(m_{heap},n-1) + \phi_t(m_{heap},n)\big)$;
12 Remove (m_{heap},n) from $\mathcal{I}$;
13 Insert (m_{heap},n) into the *heap*;
14 **end**
15 **end**
16 **if** $n_{heap} == n$ **then**
17 **if** $(m_{heap}+1,n) \in \mathcal{I}$ **then**
18 $\widehat{\phi}(m_{heap}+1,n) \leftarrow \widehat{\phi}(m_{heap},n) + \frac{1}{2}\big(\phi_\omega(m_{heap},n) + \phi_\omega(m_{heap}+1,n)\big)$;
19 Remove $(m_{heap}+1,n)$ from $\mathcal{I}$;
20 Insert $(m_{heap}+1,n)$ into the *heap*;
21 **end**
22 **if** $(m_{heap}-1,n) \in \mathcal{I}$ **then**
23 $\widehat{\phi}(m_{heap}-1,n) \leftarrow \widehat{\phi}(m_{heap},n) - \frac{1}{2}\big(\phi_\omega(m_{heap},n) + \phi_\omega(m_{heap}-1,n)\big)$;
24 Remove $(m_{heap}-1,n)$ from $\mathcal{I}$;
25 Insert $(m_{heap}-1,n)$ into the *heap*;
26 **end**
27 **end**
28 **end**
29 **end**

4. EVALUATION

In the experiments we used the following error measure introduced in [1]

$$E = \frac{\left\| s - \left| V_g \widehat{f} \right| \right\|_{\text{fro}}}{\|s\|_{\text{fro}}}, \tag{16}$$

where s is the target magnitude spectrogram, $\widehat{f}$ is the reconstructed signal and $\|\cdot\|_{\text{fro}}$ denotes the Frobenius norm. The transform V_g uses the same parameters (g, a and M) as the one used to obtain s. Values in decibels are obtained by $20\log_{10}(E)$. It has often been pointed out that such error measure does not reflect error of the reconstructed signal (see e.g. [3]) nor the actual perceived quality degradation. Therefore it should only be considered as a rough indicator. Listening tests would have been conducted in order to get a fair comparison.

For the tests, we used the SQAM database [12] which consists of 70 recordings sampled at 44.1 kHz. Only the first 10 seconds from the first channel of each sound sample was used in the evaluation.

The code (runnable in Matlab or GNU Octave[13]) for reproducing tables presented in this manuscript are freely available[1]. Note that LTFAT – Large Time-Frequency Analysis Toolbox[2] [14] (version 2.1.2 and above) and PHASERET – Phase Retrieval Toolbox[3] (version 0.1.0 and above) are required in order to run the scripts. Both toolboxes are freely available.

In the tests, 4 different compactly supported windows of full FFT length $M = 2048$ are used. The Gaussian window was truncated at relative height $h = 0.01$. The hop factor a and therefore the redundancy M/a was varied.

[1] `http://ltfat.github.io/notes/043`
[2] `http://ltfat.github.io`
[3] `http://ltfat.github.io/phaseret`

4.1. Performance Comparison

We evaluate the performance of variants of the algorithms for zero (0) or one (1) lookahead frames. The SPSI algorithm naturally falls just into the former category, while RTISI-LA can benefit from more than one lookahead frame. The relative tolerance of the proposed algorithm was set to 10^{-6}. For the RTISI-LA algorithm an asymmetric window and 16 per-frame iterations were used. All tested algorithms are able to run in real-time in the tested setting.

Tables 2, 3 and 4 show average E in dB for the test database for $a = 512$, $a = 256$ and $a = 128$ respectively. Scores for individual files and Matlab/GNU Octave scripts reproducing the results can be found at the accompanying web page.

Table 2: Algorithm comparison, $a = 512$.

	Gauss	Hann	Hamming	Blackman
SPSI	-17.82	-16.72	-16.53	-17.62
RTPGHI (0)	-17.76	-17.50	-17.58	-17.82
RTISI (0)	**-22.80**	**-21.75**	**-21.08**	**-22.82**
RTPGHI (1)	-20.91	-20.25	-20.07	-20.76
RTISI-LA (1)	**-26.08**	**-25.14**	**-24.01**	**-26.63**

Table 3: Algorithm comparison, $a = 256$.

	Gauss	Hann	Hamming	Blackman
SPSI	-17.88	-17.09	-16.79	-17.79
RTPGHI (0)	**-21.79**	**-21.10**	**-21.01**	**-21.78**
RTISI (0)	-21.15	-20.20	-19.53	-21.07
RTPGHI (1)	**-24.87**	**-22.74**	**-21.98**	**-24.59**
RTISI-LA (1)	-22.11	-19.90	-19.14	-21.90

Table 4: Algorithm comparison, $a = 128$.

	Gauss	Hann	Hamming	Blackman
SPSI	-17.99	-17.16	-16.82	-17.92
RTPGHI (0)	**-26.13**	**-23.48**	**-22.33**	**-25.93**
RTISI (0)	-20.18	-19.70	-19.01	-20.35
RTPGHI (1)	**-26.83**	**-23.50**	**-22.47**	**-26.21**
RTISI-LA (1)	-17.85	-16.66	-16.12	-17.71

Table 2 clearly shows that the RTISI-LA algorithm is superior when a bigger hop factor ($a = 512$) is used.

Table 3 already shows improvement for the proposed algorithm and in table 4, the proposed algorithm performs the best by a large margin. Another observation is that the performance gap between both variants of the proposed algorithm is virtually nonexistent when high enough overlap is used.

While the proposed algorithm clearly benefits from a higher window overlap due to the reduction of the aliasing, it is not true for the other two algorithms. The error achieved by SPSI seems to be unaffected and for RTISI it actually grows. This is a known property of RTISI and it is explained in [4]. Moreover, since higher overlap STFTs are more robust to (unwanted) coefficient perturbations the reconstructed signals tend to sound better than lower window overlap STFT reconstruction with the same error E.

The performance of the RTISI-LA algorithm can be obviously further improved by employing more lookahead frames and more iterations. Each lookahead frame however introduces additional a samples to the overall processing delay and only a limited number of per-frame iterations can be done within the real-time deadline restriction.

A real-time demo script comparing all three algorithms can be found in the PHASERET toolbox [4].

4.2. Modified Spectrogram

In order to compare performance of the algorithms on modified spectrograms, we implemented comb-filter free audio mixing of two signals from [5]. The phaseless reconstruction is done with the sum of STFT modulus of a signal with its slightly delayed version. Since usage of any objective measure does not seem to be relevant for a systematic comparison, we only provide the real-time implementation and leave the evaluation to interested readers. The demo is available in the PHASERET toolbox [5].

4.3. Timing

All three algorithms were implemented in C and per-frame execution times were compared in Tab. 5 for the same setting as in Tab. 3. Because the execution time of the present algorithm is highly signal dependent, we are mainly interested in the worst-case performance, which is also the crucial indicator for the real-time setting. The implementations were done in such a way that no memory allocation occurs during the execution.

Table 5: Worst per-frame exetution time in ms, from $\sim 10^5$ frames

SPSI	RTPGHI	RTISI-LA			
		1 it.	4 it.	8 it.	16 .it
0.23	0.56	0.31	0.58	1.16	2.18

The benchmark was run with the highest priority on an idle PC equipped with Intel Core i5-4570@3.20GHz and 8 GB RAM running Xubuntu 14.04.3 LTS with generic kernel 3.13.0-65. GCC 5.3.0 with the `-O3` optimization switch was used to produce the binary. The FFTW3 library [15] was used for computing (shortened) FFT for real signals with the `FFTW_MEASURE` plan option. Though we cannot rule out the possibility that the implementations are suboptimal (no explitic attempt was made to exploit SSE/AVX instruction sets or paralelization of the code), we tried to use the best practices for all of them. Moreover, the source code of the benchmark is available at the accompanying webpage.

5. CONCLUSION

A novel real-time algorithm for signal reconstruction from the STFT magnitude was presented. The tests showed that for fixed delay it is able to outperform the state-of-the-art algorithms when high enough window overlap is used. Moreover, the benchmark shows that the worst per-frame execution time of the proposed algorithm is about twice as much as of the SPSI algorithm and rouhly equal to execution time of the RTISI-LA algorithm with 4 iterations.

[4] `demo_blockproc_phaseret.m`

[5] `demo_blockproc_phaseretmix.m`

6. REFERENCES

[1] D. Griffin and J.S. Lim, "Signal estimation from modified short-time Fourier transform," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 32, no. 2, pp. 236–243, Apr 1984.

[2] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of audio," *Speech and Audio Processing, IEEE Transactions on*, vol. 7, no. 3, pp. 323–332, May 1999.

[3] Nicolas Sturmel and Laurent Daudet, "Signal reconstruction from STFT magnitude: A state of the art," *Proc. Int. Conf. Digital Audio Effects DAFx*, vol. 2012, pp. 375–386, 2011.

[4] Xinglei Zhu, Gerald T. Beauregard, and Lonce Wyse, "Real-time signal estimation from modified short-time Fourier transform magnitude spectra," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 5, pp. 1645–1653, July 2007.

[5] Volker Gnann and Martin Spiertz, "Comb-filter free audio mixing using STFT magnitude spectra and phase estimation," in *Proc. 11th International Conference on Digital Audio Effects (DAFx-08)*, Sept. 2008.

[6] Volker Gnann and Martin Spiertz, "Inversion of STFT magnitude spectrograms with adaptive window lengths," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP '09*, Apr. 2009, pp. 325–328.

[7] Volker Gnann and Martin Spiertz, "Improving RTISI phase estimation with energy order and phase unwrapping," in *Proc. 13th International Conference on Digital Audio Effects (DAFx-10)*, Sept. 2010.

[8] Gerald T. Beauregard, Mithila Harish, and Lonce Wyse, "Single pass spectrogram inversion," in *Digital Signal Processing (DSP), 2015 IEEE International Conference on*, July 2015, pp. 427–431.

[9] Michael R. Portnoff, "Magnitude-phase relationships for short-time Fourier transforms based on Gaussian analysis windows," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '79.*, Apr 1979, vol. 4, pp. 186–189.

[10] Zdeněk Průša, Peter Balazs, and Peter L. Søndergaard, "A Non-iterative Method for (Re)Construction of Phase from STFT magnitude," 2016, Preprint available from http://ltfat.github.io/notes/ltfatnote040.pdf.

[11] F. Auger, E. Chassande-Mottin, and P. Flandrin, "On phase-magnitude relationships in the short-time Fourier transform," *Signal Processing Letters, IEEE*, vol. 19, no. 5, pp. 267–270, May 2012.

[12] "Tech 3253: Sound Quality Assessment Material recordings for subjective tests," Tech. Rep., The European Broadcasting Union, Geneva, Sept. 2008.

[13] John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring, *GNU Octave version 4.0.0 manual: a high-level interactive language for numerical computations*, 2015.

[14] Zdeněk Průša, Peter L. Søndergaard, Nicki Holighaus, Christoph Wiesmeyr, and Peter Balazs, "The Large Time-Frequency Analysis Toolbox 2.0," in *Sound, Music, and Motion*, Lecture Notes in Computer Science, pp. 419–442. Springer International Publishing, 2014.

[15] Matteo Frigo and Steven G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005, Special issue on "Program Generation, Optimization, and Platform Adaptation".

MODIFYING SIGNALS IN TRANSFORM DOMAIN: A FRAME-BASED INVERSE PROBLEM.

Roswitha Bammer *

NuHAG, Faculty of Mathematics,
University of Vienna
Oskar-Morgenstern-Platz 1
1090 Vienna, Austria
roswitha.bammer@univie.ac.at

Monika Dörfler *

NuHAG, Faculty of Mathematics,
University of Vienna
Oskar-Morgenstern-Platz 1
1090 Vienna, Austria
monika.doerfler@univie.ac.at

ABSTRACT

Within this paper a method for morphing audio signals is presented. The theory is based on general frames and the modification of the signals is done via frame multiplier. Searching this frame multiplier with given input and output signal, an inverse problem occurs and a priori information is added with regularization terms. A closed-form solution is obtained by a diagonal approximation, i.e. using only the diagonal entries in the signal transformations. The proposed solutions for different regularization terms are applied to Gabor frames and to the constant-Q transform, based on non-stationary Gabor frames.

1. INTRODUCTION AND MOTIVATION

What does it mean to convert one signal into another? In this paper a sound-signal modification is performed by morphing one sound into another, i.e. it is assumed that there exist sounds "in between" two given, distinct sounds. This morphing enables to interpolate between two sounds with sufficient similarity, i.e. in the case of instrument morphing, the same fundamental frequency.
Existing methods are based on parametric models based on parameter interpolation [1, 2].Our method allows to observe the modification necessary for morphing directly in the time-frequency domain. In our task the input and output signals are given and the transfer function which is modeled as a frame multiplier has to be estimated. Hence, the preferred output is given and we would like to compute the cause for this output. Reformulating the problem into a minimization of a functional, the estimation is transformed into a linear inverse problem. In order to add some a priori information to the minimization problem, we add regularization terms. Such an inverse problem normally can be solved by iterative shrinkage methods [3, 4] among others, because otherwise a huge matrix system must be inverted. One possible simplification, to gain a better understanding and to obtain a closed-form solution of satisfactory quality, is to perform a diagonal approximation, i.e. considering only the diagonal entries of the matrix from the signal transformations. Stating the exact solution for several regularization terms is the main result of this paper and can be found in Theorem 3.1 in Section 3.1.
Moreover we perform some numerical experiments using the obtained solutions in MATLAB. In these experiments Gabor frames and non-stationary Gabor frames leading to a constant-Q transform, as described in Section 4, are considered. Additional experiments as well as the MATLAB code and corresponding sounds can be found on the website [5]. This paper is a generalization of the first author's master thesis [6]. The basic principles have been developed, in the context of Gabor multipliers, in [7, 8, 9, 10].

* This work was supported by the Vienna Science and Technology Fund (WWTF) project SALSA (MA14-018).

2. BASICS

Frames generalize the concept of a basis, in the sense that the frame functions need not be linearly independent. The resulting *redundancy* leads to increased stability against noise or data loss. In the following we consider a general Hilbert space $\mathcal{H}$, for example $L^2(\mathbb{R}^d)$ or $\mathbb{C}^L$.

Definition 2.1 ([11]). **(Frame, Frame Bounds, Tight Frame)** *A sequence $\{e_j : j \in J\} \subseteq \mathcal{H}$ is called a frame if there exist $A, B > 0$ such that $\forall f \in \mathcal{H}$*

$$A\|f\|^2 \leq \sum_{j\in J} |\langle f, e_j\rangle|^2 \leq B\|f\|^2. \tag{1}$$

Any two constants A, B satisfying equation (1) are called frame bounds. If $A = B$, then we call $\{e_j : j \in J\}$ a tight frame.

If $\mathcal{H} = \mathbb{C}^L$, the coefficient space is also finite dimensional, i.e. $|J| = K < \infty$.
The following important operators included in a signal processing procedure will help to develop the theory of our problem.

Definition 2.2 ([11]). **(Analysis-, Synthesis- and Frame operator)** *Let $\{e_j : j \in J\}$ be a sequence in a Hilbert space $\mathcal{H}$ and $f \in \mathcal{H}$, then the coefficient operator or analysis operator $\mathcal{T} : \mathcal{H} \to \ell^2(J)$ is defined as*

$$(\mathcal{T}f)_j = \langle f, e_j\rangle = c_j, \qquad j \in J \tag{2}$$

The adjoint of the analysis operator $\mathcal{T}^* : \ell^2(J) \to \mathcal{H}$ is the synthesis operator or reconstruction operator and is defined for a finite sequence $\tilde{c} = (\tilde{c}_j)_{j\in J} \in \ell^2(J)$ by

$$\mathcal{T}^*\tilde{c} = \sum_{j\in J} \tilde{c}_j e_j \in \mathcal{H}. \tag{3}$$

Combining these two operators leads to the definition of the frame operator $S : \mathcal{H} \to \mathcal{H}$

$$Sf = \mathcal{T}^*\mathcal{T}f = \sum_{j\in J} \langle f, e_j\rangle e_j. \tag{4}$$

In the following proposition we introduce dual frames which yield a reconstruction formula.

Proposition 2.3 ([11]). **(Dual frame)**
If $\{e_j : j \in J\}$ is a frame with frame bounds $A, B > 0$, then $\{S^{-1}e_j : j \in J\}$ is a frame with frame bounds $B^{-1}, A^{-1} > 0$, the so-called dual frame. Every $f \in \mathcal{H}$ has non-orthogonal expansions

$$f = \sum_{j\in J} \langle f, S^{-1}e_j\rangle e_j = \sum_{j\in J} \langle f, e_j\rangle S^{-1}e_j,$$

where both sums converge unconditionally in $\mathcal{H}$.

A signal processing step between the analysis and the synthesis operator in Definition 2.2, where the coefficients are multiplied by weights $w_j, j \in J$, can be performed. Thus

$$\tilde{c_j} = w_j \cdot c_j.$$

This leads to the the following definition:

Definition 2.4 ([12]). **(Frame multiplier)**
Let $\mathcal{H}_1, \mathcal{H}_2$ be Hilbert spaces, let $(g_j)_{j\in J} \subseteq \mathcal{H}_1$ and $(\gamma_j)_{j\in J} \subseteq \mathcal{H}_2$ be frames. Fix a sequence $m = (m_j)_{j\in J} \in l^\infty$, then we define the frame multiplier

$$\mathbb{M}_{m;g,\gamma} : \mathcal{H}_1 \to \mathcal{H}_2$$

for the frames (g_j) and (γ_j), as

$$\mathbb{M}_{m;g,\gamma}(f) = \sum_j m_j \langle f, g_j\rangle \gamma_j.$$

The sequence $(m_j)_{j\in J}$ mentioned in this definition is called the *symbol mask* of $\mathbb{M}$ and can be interpreted as a *time-frequency transfer function.*
In the following section, we are going to introduce the inverse problem which leads to the estimation of a set of masks, in dependence on a regularization parameter λ, for the controlled modification of one given sound towards a given target sound.

3. ESTIMATION OF THE FRAME MULTIPLIER

In this section, we consider a normed tight frame (i.e. $A = B = 1$) $(g_j)_{j\in J} \subseteq \mathcal{H}$. Assume the input and output signal $s, z \in \mathcal{H}$ to be given, hence the relation

$$z = \mathbb{M}_{m;g}s$$

to be valid. Now we want to identify the linear system, where the system is treated as a frame multiplier. Let $\mathcal{T}$ and $\mathcal{T}^*$ be fixed, we can reformulate optimality as the minimization of a functional and its estimation can therefore be transformed into a linear inverse problem:

$$\tilde{m} = \arg\min_{\mathrm{m}} \|z - \mathcal{T}_g^* m \mathcal{T}_g s\|_2^2.$$

To gain more stability in order to solve the inverse problem, we add a regularization term. We therefore have to minimize the expression

$$\Phi(m) = \|z - \mathcal{T}_g^* m \mathcal{T}_g s\|_2^2 + \lambda r(m), \tag{5}$$

where $r(m) : \ell^\infty \to \mathbb{R}^+$ is a regularization term and $\lambda \in \mathbb{R}^+$ is a regularization parameter. The choice of regularization term is discussed in the next section,Theorem 3.1.

3.1. Diagonal Approximation

Since the frames used in analysis and synthesis usually lack orthogonality iterative methods need to be employed to obtain an exact solution of (5), cp. [3]. In the special case of Gabor frames it has been shown [7, 8] that an approximate solution can be achieved by reducing the term $\mathcal{T}_g^* m \mathcal{T}_g s$ to its diagonal entries. We will address a different example, namely non-stationary Gabor frames leading to a constant-Q transform. The diagonal case brings us to closed-form solutions. These solutions lead to satisfactory quality for example in experiments on audio signals as has been observed in the experimental Section 4.
Using these solutions, iterative algorithms can be applied to achieve exact solutions of equation 5. However the difference in perception is marginal, but the computational effort increases.
For some further information we refer to [13], [7], [8] and [14]. In order to achieve a diagonal approximation, we reformulate (5) in the transform domain

$$\Phi(m) = \|\mathcal{T}_g^*(\mathcal{T}_g z - m\mathcal{T}_g s)\|_2^2 + \lambda r(m).$$

Reducing to the diagonal and writing $S = \mathcal{T}_g s$ and $Z = \mathcal{T}_g z$, leads to

$$\Phi(m) = \|Z - m \cdot S\|_2^2 + \lambda r(m). \tag{6}$$

If the source equals the target, the mask m should be equal to 1 and the regularization term should vanish. This motivates the choice of regularization terms with entries $m - \vec{1}$. If Z and S are different, we can use different terms of regularization. The regularization term helps us to indicate some a priori information in the shape of the solution (the transformed signal). The choice of $r(m)$ is discussed in Remark 3.2. The parameter λ helps balancing between these a priori information of the form and the properties of reconstructing the mask [7]. We are now going to present different choices of regularization terms by stating the following theorem.

Theorem 3.1. Let $\Phi : \mathbb{C}^L \to \mathbb{R}$ be a functional of the form

$$\Phi(m) = \|Z - m \cdot S\|_2^2 + \lambda r(m), \tag{7}$$

where $\lambda \in \mathbb{R}^+$ and $r : \mathbb{C}^L \to \mathbb{R}$ is a regularization term. Minimizing this functional for different solutions with respect to different regularization terms as follows:

a) $r(m) = \|m - 1\|_2^2$ leads to the solution

$$\tilde{m_\ell} = \frac{\overline{S_\ell} Z_\ell + \lambda}{|S_\ell|^2 + \lambda} \qquad \forall \ell \in \{0, ..., L\}.$$

b) $r(m) = \| \, |m| - 1\|_2^2$ leads to the solution

$$\tilde{m_\ell} = \frac{|Z_\ell S_\ell| + \lambda}{|S_\ell|^2 + \lambda} \cdot e^{i \arg(S_\ell \overline{Z_\ell})} \qquad \forall \ell \in \{0, ..., L\}.$$

c) $r(m) = \|m - 1\|_1$ leads to the solution

$$\tilde{m_\ell} = \begin{cases} \frac{|S_\ell||Z_\ell - S_\ell| - \frac{\lambda}{2}}{|S_\ell|^2} \cdot e^{i\varphi} + 1 & \text{if } |S_\ell|\,|T_\ell - S_\ell| > \frac{\lambda}{2} \\ 1 & \text{else} \end{cases},$$

where $\varphi = arg(\overline{S_\ell}(Z_\ell - S_\ell)) \quad \forall \ell \in \{0, ..., L\}$.

d) $r(m) = \| \, |m| - 1\|_1$ leads to the solution

$$\tilde{m_\ell} = \begin{cases} \frac{|Z_\ell S_\ell| - \frac{\lambda}{2}}{|S_\ell|^2} e^{i \arg(S_\ell \overline{Z_\ell})} & \text{if } \frac{|Z_\ell S_\ell|}{|S_\ell|^2} > 1 + \frac{\lambda}{2|S_\ell|^2} \\ \frac{|Z_\ell S_\ell| + \frac{\lambda}{2}}{|S_\ell|^2} e^{i \arg(S_\ell \overline{Z_\ell})} & \text{if } \frac{|Z_\ell S_\ell|}{|S_\ell|^2} < 1 - \frac{\lambda}{2|S_\ell|^2} \\ 1 & \text{else} \end{cases}$$

$\forall \ell \in \{0, ..., L\}$.

Note that some of these solution formulas can be found for the case of Gabor multipliers in [7, 8]. Since we found them to be useful in the general case of frame multipliers [12], we include their proof in the appendix, Section 7.

Remark 3.2. *Let $\Phi : \mathbb{C}^L \to \mathbb{R}$ be as in (7). Then the different regularization terms have the following properties:*

a) $r(m) = \|m-1\|_2^2$ helps to control the total energy. Moreover, if we use normed tight frame bounds, i.e. $A = B = 1$, we favor a multiplier close to the identity operator. This regularization term produces spurious oscillations in the mask $\tilde{m}$, caused by a bad estimation of the phase. A simple calculation shows the reason of the oscillations. Let (j,k) be a point of the time-frequency plane and let the input and the output signal have a phase difference of π, i.e. $Z = Se^{i\pi}$. Then $\tilde{m}$ at the point (j,k) is given by

$$\tilde{m} = \frac{\overline{S}Z + \lambda}{|S|^2 + \lambda} = \frac{|S|^2 e^{i\pi} + \lambda}{|S|^2 + \lambda}.$$

This short calculation shows the presence of amplitude modulations of the mask due to the diagonal approximation, cp. [7, p. 43 et seq.].

b) $r(m) = \| |m| - 1\|_2^2$ gives us the possibility of avoiding spurious oscillations of the amplitude of $\tilde{m}$, apart from that fact it has the same properties as the previous regularization term in a).

c) $r(m) = \|m-1\|_1$ yields sparsity, where the mask is forced to stay close to 1 which corresponds to "no transformation". This regularization term also produces spurious oscillations.

d) $r(m) = \| |m| - 1\|_1$ forces $\tilde{m}$ to sparsity of the deviation from the absolute value 1 and also avoids the oscillations of the previous regularization term in c). For some more information on this regularization term consider [14].

In the next section we will visualize these properties by analyzing examples for diagonal approximation with different regularization terms.

4. NUMERICAL EXPERIMENTS

4.1. Two examples of frames used in audio processing

The results in Section 3.1 hold for general frames and in particular also in higher dimensions, that is, for frames for $L^2(\mathbb{R}^d)$ with $d > 1$. This can be interesting for image or video processing. In the current work, however, we focus on audio signals and present some numerical simulations using classical Gabor frames [11] on the one hand and a constant-Q transform based on non-stationary time-frequency Gabor frames, [15, 16] on the other hand. We briefly introduce the necessary notions next.

The frame elements of Gabor frame are given by time-frequency shifted versions of a non-zero window function $g \in \mathcal{H}$, i.e. $\mathcal{G}(g,a,b) = \{g_{n,k} = T_{ak}M_{bn}g : k,n, \in \mathbb{Z}\}$. Here, $T_{ak}g(t) = g(t-ak)$ and $M_{bn}g(t) = g(t)\cdot e^{2\pi ibn}$ denote time- and frequency shift, respectively.

For the non-stationary Gabor frame-based constant-Q transform, the construction of Gabor frames is generalized as to allow for windows with adaptive bandwidth. To this end, the frame elements are given by $\{g_{n,k} = T_{na_k}g_k : k,n, \in \mathbb{Z}\}$. Thus, while the time-shifts are carried out along a regular lattice as in the Gabor case, the frequency shifts are replaced by choosing a separate window for each desired frequency band. Accordingly, the time-shift parameter a_k can be chosen separately for each band. For all details, in particular regarding the precise choice of parameters for the constant-Q transform, we refer to [15, 16]. We note that for both Gabor frames and and non-stationary Gabor frames, careful choice of windows and sampling parameters a, b leads to the situation of *painless non-orthogonal expansions*, [17], for which straight-forward inversion is possible.

Implementations along with excellent documentation for both Gabor frames and the constant-Q transform can be found in the LTFAT-toolbox, [18, 19] and [16].

4.2. Experimental setup

In this section we describe the setup for the subsequent numerical experiments. All mentioned MATLAB routines are found on the website [5]. We want to find the best fitting multiplier of the inverse problem (5) using different regularization terms introduced in Theorem 3.1. To do so, we use an input and an output signal with sufficient similarity. In the following two experiments we use the sound of a flute and a violin of the VSL [20] playing the same fundamental frequency and vowels sang by a man [21] and a woman [22], also using the same fundamental frequency for sufficient similarity. The sound files are sampled with a rate of 44100 Hertz. To make the sound files the same length, we use the MATLAB code `samesize_power2.m` which fades out the signals with exponential decrease. Moreover, to display the spectrogram of our sounds we use a logarithmic scale, cp. [6, p.24].

In order to show that different classes of frames can be used, we will consider Gabor frames and non-stationary Gabor frames as introduced in Section 4.1, within the numerical examples. The MATLAB code `diagaprox.m` is used to do the numerical calculations with Gabor frames by using the closed-form solutions stated in Theorem 3.1 and the code `diagaprox_cq.m` does this by using a constant-Q transform, based on non-stationary Gabor frames. Note that in the finite discrete case underlying the numerical implementations $\mathcal{H} = \mathbb{C}^L$.

The algorithm basically uses the following steps:

- Input: s (source signal), z (target signal) of the same length (here 1 second) due to `samesize_power2.m`, a preferred norm and λ.
- Tranformation done with:
  - Gabor transform [`dgt.m`] of $s \to S$ and $z \to Z$ with a Hann-window and the parameters $a = 256, M = \frac{L}{b} = 1024$.
  - Constant-Q transform [`cqt.m`] of $s \to S$ and $z \to Z$ with frequency region $[100\text{Hz}; 22050\text{Hz}]$, 64 bin per octave and 1000 time channels per second.
- Obtain the mask m as in Theorem 3.1 corresponding to the respective norm used for regularization.
- Inverse transform [`idgt.m`] or [`icqt.m`], respectively, of $m * S$ to obtain a $\tilde{z}$, the target signal.
- Output: m and $\tilde{z}$.

4.3. Sound morphing

4.3.1. Using musical instruments

For the first experiment we use the sound of a flute and a violin from the VSL [20]. Since sufficient similarity is required, we con-

sider the same fundamental fequency of the instruments for each morphing procedure. Morphed sounds can be obtained by varying λ in Formula (5). A high value of λ puts heavy weight on the regularization term $r(m)$, hence forces the mask to be close to one, i.e. "no transformation" and the signal reconstructed from $\tilde{m} * S$ is similar to the source/input signal. A small λ does not take the regularization term that much into account. This leads to a mask which yield a reconstructed signal close to the target signal.
The choice of these two instruments is due to their different timbres and harmonics. The violins sound is rich in overtones, whereas the flute has less overtones.
The following experiment is done with constant-Q transform. Similar results can be achieved using the Gabor transform, cp. [5]. In Figure 1 we show stepwise morphing, i.e. using different λ $(= 10^{-2}, 10^{-6})$ starting from the original flute sound as source going to the violin as target. This case is very interesting, because it is more difficult for the mask to 'generate' overtones, since the violin has more overtones than the flute, than suppressing them, as it would be the other way round. As common fundamental frequency we use B5; the original sounds and sounds resulting from morphing steps can be found online [5].
In Figure 1 one can see how the noise, coming from the violin increases from step to step. For $\lambda = 10^{-2}$ the sound is a mixture of flute and violin, but for $\lambda = 10^{-6}$ we can verify the sound as a violin.

4.3.2. Using spoken vowels

The second experiment considers German vowels, sung by a professional singer. We use a female [22] as well as a male [21] voice. Both have the fundamental frequency E4. This morphing task is interesting because vowels build different formants, i.e. acoustic resonance of the human vocal tract, where certain harmonics are stronger than others. Within this experiment it is visible that similar vowels, like the german spoken "e" and "i" which sound very similar, also morph with comparably bigger λ into each other. Several tables summarizing which λ has to be used to get reconstruction of the target signal can be found in [5]. In Figure 2, we perform again a morphing procedure using constant-Q transform. The morphing is performed stepwise starting from the vowel "a", with steps in between with $\lambda = 10^{-4}$ and $\lambda = 10^{-8}$, reaching target vowel "i". The noise level goes down in the range between $[600\text{Hz}, 2000\text{Hz}]$. Hence the harmonics are better visible and focus on 300Hz which is the characteristic formant for the vowel "i" [23].
Another thing that can be observed, concerning the mask of the morphing steps are spurious oscillations which are mentioned in Remark 3.2 a). Since the observation is almost only visible using the Gabor transform, we use this transformation to generate Figure 3. Nevertheless there was no audible difference within the morphing experiments. The morphing is, as mentioned above, performed going from vowel "a" to "i" and to show the oscillations, we take the mask corresponding to $\lambda = 10^{-2}$. In Figure 3 the upper image shows the mask obtained with the regularization term $r(m) = \|m - 1\|_2^2$. Here oscillations are visible between 3000Hz and 4000Hz. The lower image neglects the phase by using the modulus of the mask, i.e. $r(m) = \| \, |m| - 1\|_2^2$, hence no oscillations are visible.

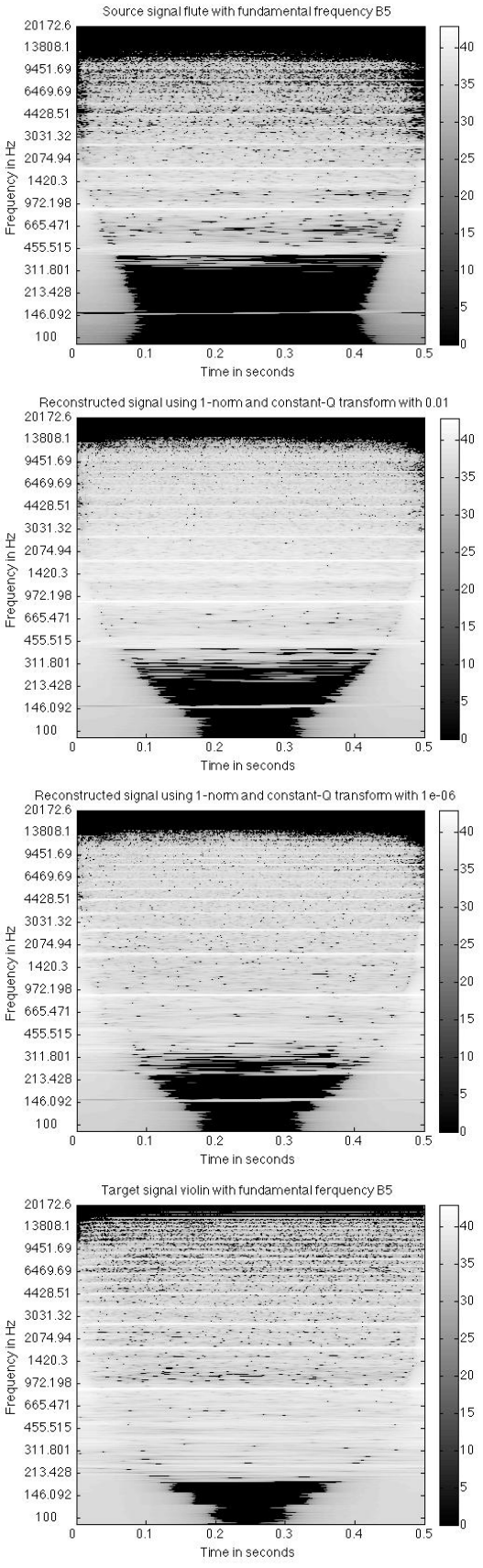


Figure 1: Stepwise morphing from flute to violin, with steps in between at $\lambda = 10^{-2}$ and $\lambda = 10^{-6}$.

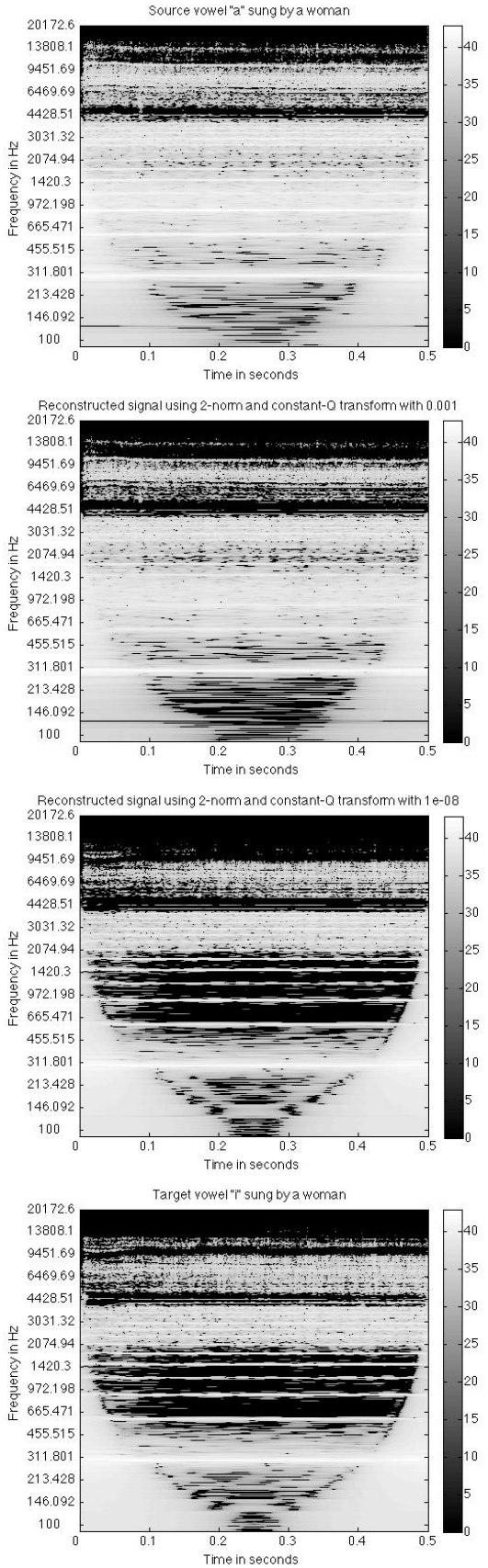


Figure 2: Stepwise morphing from vowel "a" to "i", with steps in between at $\lambda = 10^{-4}$ and $\lambda = 10^{-8}$.

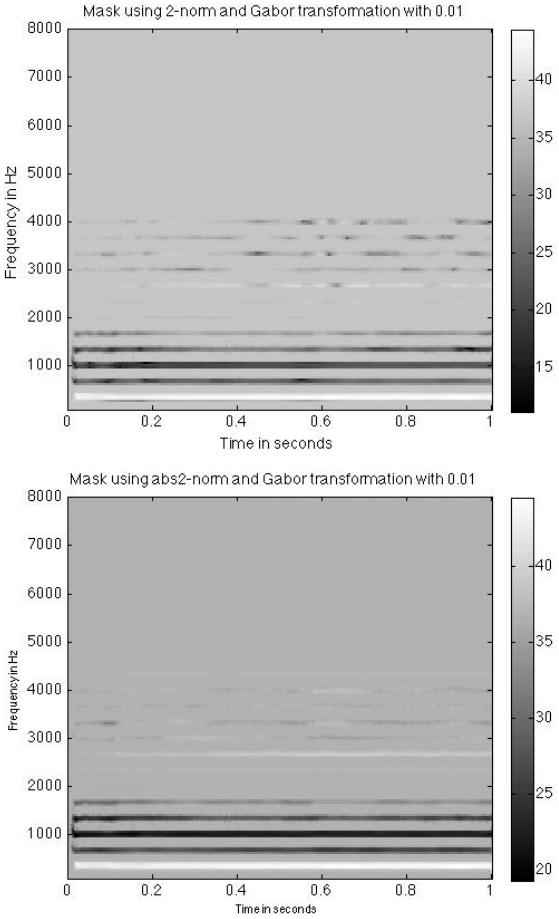


Figure 3: Spurious oscillations for $\lambda = 10^{-2}$, obtained by morphing vowel "a" into "i".

5. CONCLUSION AND PERSPECTIVES

In this paper a morphing procedure was proposed, using frame multipliers in order to morph one audio sound into another. A diagonal approximation was performed in order to get a closed form solution of a regularized inverse problem.
Comparing the solution of the diagonal approximation with the solution computed by iterative shrinkage threshold algorithms (ISTA) yield only a marginal difference. For the monotone fast ISTA the solution was better audible. Here it would be interesting to figure out, why this is the case. Nevertheless the computational effort increases strongly. The obtained solutions were used in the experimental chapter, to show that this approximation also leads to satisfactory perceptive quality. The general concept of frames allowed to state different examples. We focused on the usage of Gabor frames and non-stationary Gabor frame based constant-Q transform.
Extensions of the presented work will include the usage of other frames, for example wavelet frames in the context of image morphing and other non-stationary time-frequency frames. Furthermore, the class of coefficient priors will be extended to mixed-norm and neighborhood-based priors, [24, 25], which will lead to a structure-based signal modification.
Another strand of research will investigate, why spurious oscillations, mentioned in Remark 3.2 and rather prominent if the morphing is based on Gabor frames, are barely visible for constant-Q transform, while the perceptual difference seems marginal. To this end, we will set up an evaluation framework based on perceptual criteria, cp. [26], since comprehensive listening experiments are costly. In parallel, we will isolate the oscillations and use subsequent synthesis to understand their behaviour and consequence on perceptual outcome. Finally, the variety of sounds obtained by con- trolled morphing can be used for data augmentation, [27], within machine learning tasks for audio signals.

6. REFERENCES

[1] A. Chadha, B. Savardekar, and J. Padhya, "Analysis of a modern voice morphing approach using gaussian mixture models for laryngectomees," *CoRR*, vol. abs/1208.1418, 2012.

[2] M.F. Caetano and X. Rodet, "Musical instrument sound morphing guided by perceptually motivated features," *IEEE Trans. Audio, Speech & Language Processing*, vol. 21, no. 8, pp. 1666–1675, 2013.

[3] I. Daubechies, M. Defrise, and C. De Mol, "An Iterative Thresholding Algorithm for Linear Inverse Problems with a Sparsity Constraint," *Comm. Pure Apl. Math.* , vol. 57, no. 11, pp. 1413–1457, 2004.

[4] A. Beck and M. Teboulle, "A Fast Iterative Shrinkage-Threshold Algorithm for Linear Inverse Problems," *SIAM J.Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[5] R. Bammer and M. Dörfler, "Modifying Signals in Transform Domain," `http://projektservice-mathematik.univie.ac.at/morph`.

[6] R. Bammer, "Signaltransfomation via Gabor Multiplier," M.S. thesis, University of Vienna, 2015, `http://projektservice-mathematik.univie.ac.at/morph`.

[7] A. Olivero, *Les Multiplicateurs Temps-Fréquence. Application á l'Analyse et la Synthése de Signaux Sonores et Musicaux*, Ph.D. thesis, University of AIX-Marseille, 2012.

[8] A. Olivero and B. Torrésani and R. Kronland-Martinet, "A Class of Algorithms for Time-Frequency Multiplier Estimation.," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 21, no. 8, pp. 1550–1559, 2013.

[9] A. Olivero and B. Torrésani and P. Depalle and R. Kronland-Martinet, "Sound morphing strategies based on alterations of time-frequency representations by Gabor multipliers," in *AES 45th International Conference on Applications of Time-Frequency Processing in Audio*, Helsinki, Finland, 2012, p. 17.

[10] A. Olivero and B. Torrésani and R. Kronland-Martinet, "A new method for Gabor multipliers estimation : application to sound morphing," in *European Signal Processing Conference (EUSIPCO 2010)*, Aalborg, Denmark, 2010, pp. 507–511.

[11] K. Gröchenig, *Foundations of Time-Frequency Analysis*, Appl. Numer. Harmon. Anal. Birkhäuser, 2001.

[12] P. Balazs, "Basic definition and properties of Bessel multipliers ," *Journal of Mathematical Analysis and Applications* , vol. 325, no. 1, pp. 571 – 585, 2007.

[13] P. Depalle and R. Kronland-Martinet and B. Torrésani, "Time-Frequency multipliers for sound synthesis," in *SPIE annual Symposium Wavelet XII*, San Diego, United States, 2007, vol. 6701, pp. 670118–1 – 670118–15.

[14] M. Dörfler and E. Matusiak, "Sparse Gabor multiplier estimation for identification of sound objects in texture sound.," in *Sound, Music, and Motion*, M. Aramaki et al., Ed., vol. LNCS 8905, pp. 443–462. Springer International Publishing Switzerland, 2014.

[15] G. A. Velasco, N. Holighaus, M. Dörfler, and T. Grill, "Constructing an invertible constant-Q transform with non-stationary Gabor frames," *Proceedings of DAFX11*, 2011.

[16] N. Holighaus, M. Dörfler, G. A. Velasco, and T. Grill, "A framework for invertible, real-time constant-Q transforms," *IEEE Trans. Audio Speech Lang. Process.*, vol. 21, no. 4, pp. 775 –785, 2013, `http://www.univie.ac.at/nonstatgab`.

[17] I. Daubechies, A. Grossmann, and Y. Meyer, "Painless nonorthogonal expansions," *J. Math. Phys.*, vol. 27, no. 5, pp. 1271–1283, May 1986.

[18] P. L. Søndergaard, B. Torrésani, and P. Balazs, "The Linear Time Frequency Analysis Toolbox," *International Journal of Wavelets, Multiresolution Analysis and Information Processing*, vol. 10, no. 4, 2012, `http://ltfat.sourceforge.net/`.

[19] Z. Průša, P. L. Søndergaard, N. Holighaus, C. Wiesmeyr, and P. Balazs, "The Large Time-Frequency Analysis Toolbox 2.0," in *Sound, Music, and Motion*, M. Aramaki et al., Ed., vol. LNCS 8905, pp. 419–442. Springer International Publishing Switzerland, 2014.

[20] "Vienna Symphonic Library: Vienna Super Package," 2014.

[21] E. Tarilonte, "Altus: The Voice of Renaissance," Software: `http://www.bestservice.de/altus.html`.

[22] E. Tarilonte, "Shevannai: The Voice of Elves," Software: `http://www.bestservice.de/shevannai.html`.

[23] K. Johnson, *Acoustic and auditory phonetics*, Blackwell Cambridge, Mass. a.o., 2003.

[24] B. Torrésani and M. Kowalski, "Sparsity and Persistence: mixed norms provide simple signal models with dependent coefficients," *Signal, Image and Video Processing*, to appear, 2008.

[25] M. Kowalski, K. Siedenburg, and M. Dörfler, "Social Sparsity! Neighborhood Systems Enrich Structured Shrinkage Operators," *IEEE Trans. Signal Process.*, vol. 61, no. 10, pp. 2498 – 2511, 2013.

[26] P. Kabal, *An examination and interpretation of itu- r bs. 1387: Perceptual evaluation of audio quality*, Ph.D. thesis, tech. rep., Dep. of Electrical Engineering and Com- puter Engineering, McGill University, 2003.

[27] J. Schlüter and T. Grill, "Exploring Data Augmentation for Improved Singing Voice Detection with Neural Networks," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR 2015)*, Malaga, Spain, 2015.

[28] L.V. Ahlfors, *Complex analysis : an introduction to the theory of analytic functions of one complex variable*, International series in pure and applied mathematics. McGraw-Hill, New York, 1979.

7. APPENDIX

Proof of Theorem 3.1. To find a minimum of (7), we have to find zeros of its first derivative. We first rewrite $\|Z - m \cdot S\|_2^2$ as the sum of its entries squared

$$\begin{aligned}\Phi(m) &= \sum_{\ell\in\Lambda} \Phi^\ell(m_\ell) \\ &= \sum_{\ell\in\Lambda} \big(|Z_\ell - m_\ell S_\ell|^2 + \lambda r(m_\ell)\big) \\ &= \sum_{\ell\in\Lambda} \big((Z_\ell - m_\ell S_\ell)\overline{(Z_\ell - m_\ell S_\ell)} + \lambda r(m_\ell)\big).\end{aligned}$$

Writing the complex vector m as $m = m^r + i\,m^i$, where $m^r, m^i \in \mathbb{R}^L$ we obtain

$$= \sum_{\ell\in\Lambda} \big((Z_\ell - S_\ell(m^r + i\,m^i))\overline{(Z_\ell - S_\ell(m^r + i\,m^i))} + \lambda r(m_\ell)\big).$$

The derivative of Φ can now be understood as a derivative in two variables. Using the formula

$$\frac{\partial\Phi(m)}{\partial m} = \frac{1}{2}\left(\frac{\partial\Phi(m^r, m^i)}{\partial m^r} - i\frac{\partial\Phi(m^r, m^i)}{\partial m^i}\right) \tag{8}$$

we obtain the derivative for holomorphic functions [28]. Next we fix one ℓ since, if we take the derivative component-wise, the other components will vanish. The derivative with respect to the first variable m_ℓ^r is

$$\begin{aligned}\frac{\partial\Phi^\ell(m_\ell^r, m_\ell^i)}{\partial m_\ell^r} =& - S_\ell\overline{(Z_\ell - S_\ell(m^r + i\,m^i))} \\ &+ (Z_\ell - S_\ell(m^r + i\,m^i))\overline{(-S_\ell)} + \lambda\frac{\partial r(m_\ell)}{\partial m_\ell^r}.\end{aligned}$$

Using $\frac{z+\overline{z}}{2} = \mathfrak{Re}(z)$ we get

$$= -2\mathfrak{Re}\big(S_\ell\overline{Z_\ell}\big) + 2\,|S_\ell|^2\, m_\ell^r + \lambda\frac{\partial r(m_\ell)}{\partial m_\ell^r}.$$

Similarly we compute the derivative with respect to m_ℓ^i

$$\frac{\partial\Phi^\ell(m_\ell^r, m_\ell^i)}{\partial m_\ell^i} = 2\cdot\mathfrak{Im}\big(S_\ell\overline{Z_\ell}\big) + 2\,|S_\ell|^2\, m_\ell^i + \lambda\frac{\partial r(m_\ell)}{\partial m_\ell^i}.$$

Using Equation (8) we obtain

$$\begin{aligned}\frac{\partial\Phi^\ell(m_\ell)}{\partial m_\ell} =& \frac{1}{2}\Big(- 2\mathfrak{Re}\big(S_\ell\overline{Z_\ell}\big) + 2\,|S_\ell|^2\, m_\ell^r + \lambda\frac{\partial r(m_\ell)}{\partial m_\ell^r} \\ &- 2i\mathfrak{Im}\big(S_\ell\overline{Z_\ell}\big) - 2i\,|S_\ell|^2\, m_\ell^i - i\lambda\frac{\partial r(m_\ell)}{\partial m_\ell^i}\Big).\end{aligned}$$

To obtain a minimum, we have to set the following equation to zero:

$$\frac{\partial\Phi^\ell(m_\ell)}{\partial m_\ell} = -S_\ell\overline{Z_\ell} + |S_\ell|^2\,\overline{m_\ell} + \underbrace{\frac{\lambda}{2}\Big(\frac{\partial r(m_\ell)}{\partial m_\ell^r} - i\frac{\partial r(m_\ell)}{\partial m_\ell^i}\Big)}_{\rho(m_\ell)} = 0. \tag{9}$$

Now we have to distinguish according to the different regularization terms.

a) Considering $r(m) = \|m-1\|_2^2$ as the first regularization term, we have

$$\rho(m_\ell) = \frac{\lambda}{2}(2m_\ell^r - 2 - 2im_\ell^i) = \lambda\overline{m_\ell} - \lambda.$$

Thus, solving Equation (9) with respect to $\overline{m_\ell}$ and taking the conjugate, we obtain for every ℓ

$$\tilde{m_\ell} = \frac{\overline{S_\ell}Z_\ell + \lambda}{|S_\ell|^2 + \lambda}.$$

b) For the regularization term $r(m) = \|\,|m| - 1\|_2^2$ we have

$$\rho(m_\ell) = \frac{\lambda}{2}\left(2\lambda m_\ell^r - 2\frac{\lambda m_\ell^r}{|m_\ell|} - 2\lambda m_\ell^i + 2\frac{\lambda m_\ell^i}{|m_\ell|}\right) = \lambda\overline{m_\ell} - \frac{\lambda\overline{m_\ell}}{|m_\ell|}$$

If we plug this term into Formula (9), we obtain

$$\overline{m_\ell}\big(\,|S_\ell|^2 + \lambda - \frac{\lambda}{|m_\ell|}\big) = S_\ell\overline{Z_\ell}. \tag{10}$$

Since there is a term containing $|m_\ell|$ in the brackets of this solution, we multiply $\overline{m_\ell}$ with its conjugate and obtain

$$\overline{m_\ell}m_\ell = |m_\ell|^2 = \frac{|Z_\ell S_\ell|^2}{\big(\,|S_\ell|^2 + \lambda - \frac{\lambda}{|m_\ell|}\big)^2}.$$

Solving this equation with respect to the modulus of m_ℓ and using the formula $z = |z|\,e^{i\arg(z)}$, the phase is only given by $S_\ell\overline{Z_\ell}$ in Equation (10), because the term in the brackets is real. The solution of our functional for every ℓ is

$$\tilde{m_\ell} = \frac{|Z_\ell S_\ell| + \lambda}{|S_\ell|^2 + \lambda}\cdot e^{i\arg(S_\ell\overline{Z_\ell})}.$$

c) For the regularization term $r(m) = \|m - 1\|_1$ we apply a substitution $m - 1 = \mu$, hence

$$\rho(\mu_\ell) = \frac{\lambda}{2}\left(\frac{\mu_\ell^r}{|\mu_\ell|} - i\frac{\mu_\ell^i}{|\mu_\ell|}\right) = \frac{\lambda}{2}\frac{\overline{\mu_\ell}}{|\mu_\ell|}.$$

Again we have to multiply with the conjugate in a similar manner as in case b) and we again use the formula $\mu = |\mu|\, e^{i\arg(\mu)}$. Undoing the substitution and applying a threshold argument obtained due to $|\mu_\ell| > 0$, we get

$$\tilde{m_\ell} = \frac{\left|\overline{S_\ell}\right| |Z_\ell - S_\ell| - \frac{\lambda}{2}}{|S_\ell|^2} \cdot e^{i\arg(\overline{S_\ell}(Z_\ell - S_\ell))} + 1 \qquad (11)$$

as long as

$$|S_\ell|\,|T_\ell - S_\ell| > \frac{\lambda}{2}.$$

d) For $r(m) = \|\,|m| - 1\|_1$ we have to make a distinction in two cases $|m_\ell| > 1$ and $|m_\ell| < 1$, from which

$$\rho(m_\ell) = \frac{\lambda}{2}\left(\pm\frac{m_\ell^r}{|m_\ell|} \mp \frac{m_\ell^i}{|m_\ell|}\right) = \pm\frac{\lambda}{2}\frac{\overline{m_\ell}}{|m_\ell|}$$

is obtained.
Using $|m_\ell|^2 = \overline{m_\ell} \cdot m_\ell$ and $z = |z|\, e^{i\arg(z)}$ we obtain

$$\tilde{m_\ell} = \begin{cases} \frac{|Z_\ell S_\ell| - \frac{\lambda}{2}}{|S_\ell|^2} e^{i\arg(S_\ell \overline{Z_\ell})} & \text{if } \frac{|Z_\ell S_\ell|}{|S_\ell|^2} > 1 + \frac{\lambda}{2|S_\ell|^2} \\ \frac{|Z_\ell S_\ell| + \frac{\lambda}{2}}{|S_\ell|^2} e^{i\arg(S_\ell \overline{Z_\ell})} & \text{if } \frac{|Z_\ell S_\ell|}{|S_\ell|^2} < 1 - \frac{\lambda}{2|S_\ell|^2} \\ 1 & \text{else.} \end{cases}$$

□

TIME-VARIANT GRAY-BOX MODELING OF A PHASER PEDAL

Roope Kiiski, Fabián Esqueda and Vesa Välimäki*

Aalto University
Department of Signal Processing and Acoustics
Espoo, Finland
roope.kiiski@aalto.fi

ABSTRACT

A method to measure the response of a linear time-variant (LTV) audio system is presented. The proposed method uses a series of short chirps generated as the impulse response of several cascaded allpass filters. This test signal can measure the characteristics of an LTV system as a function of time. Results obtained from testing of this method on a guitar phaser pedal are presented. A proof of concept gray-box model of the measured system is produced based on partial knowledge about the internal structure of the pedal and on the spectral analysis of the measured responses. The temporal behavior of the digital model is shown to be very similar to that of the measured device. This demonstrates that it is possible to measure LTV analog audio systems and produce approximate virtual analog models based on these results.

1. INTRODUCTION

Historically, guitar effect pedals have been designed and implemented using analog circuitry. Nowadays, digital effects have become more common and widely accepted, as they are more versatile and their costs can be reduced. Yet many of the old, analog devices have cult-like following, due to their alleged distinctive sound or their association with famous musicians [1]. Vintage analog pedals are rare and can reach high prices in secondary markets [2]. Therefore, to cater to consumers who are after particular effects, virtual analog modeling of these classic devices is needed. Some research on modeling different distortion, fuzz, reverb, and delay units can be found in literature [1, 3]. One particular effect, the *phaser*, is an interesting topic of study due to its time-varying nature. In this paper, an overview on the internal workings of a phaser is given. Then, a method to measure time-varying linear effects is introduced, along with results obtained from measuring a phaser pedal. These results are then used to develop a digital model of the phaser effect.

Two opposite system modeling approaches are 'black-box' and 'white-box' models [1]. Black-box models are such that the modeling is entirely based on input-output measurements of the system and there is no knowledge of the inner workings of the system. An example of black-box modeling of effect units can be found in [4]. On the other hand, white-box models are based on circuit analysis or other form of knowledge of how the system works. A white-box model of a phaser pedal was presented by Eichas et al. [2]. The technique followed in this study falls under the category of 'gray-box' modeling, which is defined as an approach where some knowledge of the internal characteristics of the system is used when modeling it based on measurements [1].

* The work of F. Esqueda is funded by the Aalto ELEC Doctoral School.

This paper is organized as follows. Section 2 discusses phasing using allpass filters. In Section 3, the steps necessary to measure the time-varying response of a phaser are presented. Section 4 presents the measured responses of an analog phaser pedal. Section 5 shows how the measured response is modeled in the digital domain. Finally, concluding remarks are given in Section 6.

2. BASICS OF PHASING

The phasing effect introduces time-varying notches in the spectrum of the input signal, creating a characteristic *swooshing* sound. Historically-famous phaser pedals, such as the MXR series and the Uni-Vibe, work by feeding a copy of the input signal into a chain of identical first-order allpass filters and mixing the output of this chain with the original signal [2, 3]. In this design, the location of the notches is determined by the break frequencies of the filters, which are modulated by a low frequency oscillator (LFO).

The block diagram for a basic digital phaser is shown in Fig. 1. By adjusting the *wetness* of the phaser, i.e. the ratio between original and filtered signals, using gain parameters G and W, the depth of the notches and the overall intensity of the effect can be controlled. To keep the output signal bounded, G and W are usually coupled so that $G = 1 - W$. The deepest possible notches are obtained when $W = G$. In this architecture, the number of notches introduced is determined by the number of the first-order allpass filters used, so that for N allpass filters (assuming N is even), $N/2$ notches are produced on positive frequencies. Usually the number of allpass filters in phasers in phasers is even.

The transfer function of a first-order digital allpass filter is

$$A(z) = \frac{a_1 + z^{-1}}{1 + a_1 z^{-1}}, \tag{1}$$

where coefficient a_1, which also determines the pole, is defined in

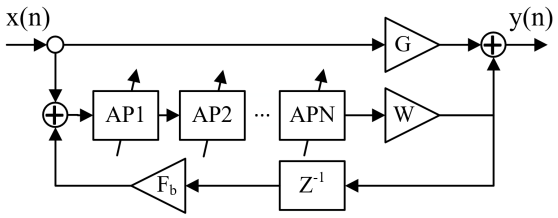


Figure 1: *Block diagram of a digital phaser with a feedback loop via a unit delay. In most common phasers, there is no feedback loop, meaning that their* $F_b = 0$.

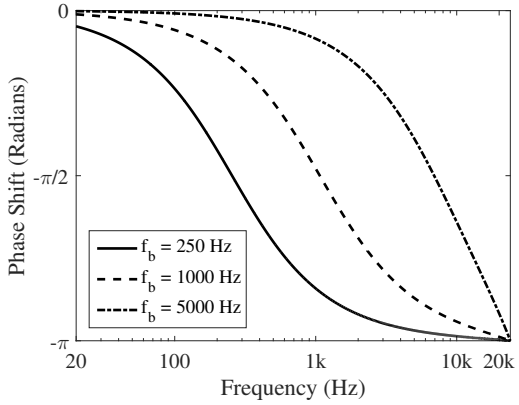


Figure 2: *Phase response of a first-order allpass filter with break frequencies 250 Hz ($a_1 = -0.967$), 1000 Hz ($a_1 = -0.869$), and 5000 Hz ($a_1 = -0.346$).*

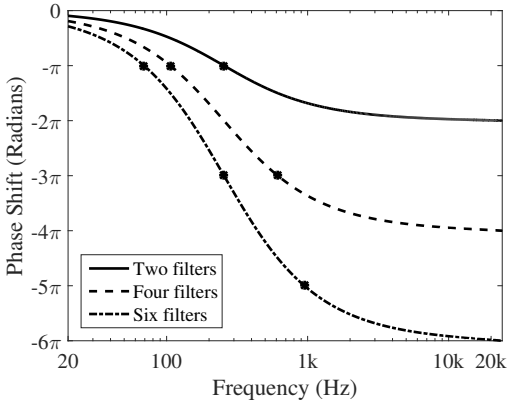


Figure 3: *Phase response of two, four, and six cascaded first-order allpass filters with break frequency at 250 Hz ($a_1 = -0.967$). The squares indicate the resulting notch frequencies in each case, when used in a phaser.*

terms of ω_b, the break frequency of the filter in rad/s, as

$$a_1 = -\frac{1-\tan(\omega_\mathrm{b}T/2)}{1+\tan(\omega_\mathrm{b}T/2)} \approx -\frac{1-\omega_\mathrm{b}T/2}{1+\omega_\mathrm{b}T/2} \approx -1+\omega_\mathrm{b}T, \quad (2)$$

where T is the sampling period of the system. For stability, the condition $|a_1| < 1$ is required [5]. The phase response of the allpass filter (1) as a function of angular frequency ω is given by

$$\Theta(\omega) = -\omega + 2\arctan\left(\frac{a_1 \sin\omega}{1+a_1\cos\omega}\right). \quad (3)$$

During phasing, the allpass filter chain itself does not change the magnitude spectrum of the signal; it causes *phase shifting* [6]. From (3), we observe that each first-order allpass filter causes a frequency shift of $-\pi$ radians at half the sampling frequency (i.e. at $\omega = \pi$). At the break frequency the filter will introduce a phase shift of exactly $-\pi/2$ rad/s [7, 8]. Fig. 2 shows example phase responses of the first-order allpass filter with three different break frequencies. The sample rate $f_\mathrm{s} = 1/T$ in these and all other examples in this paper is 44.1 kHz.

When several allpass filters are cascaded, their combined phase response is the sum of the individual phase responses. So, when N first-order allpass filters are cascaded, their combined phase response will introduce a maximum phase shift of $-N\pi$ radians. In this case, we assume N to be a positive even integer. Adding the filtered and original signals will cause notches at the frequencies at which the phase shift is an exact odd integer multiple of $-\pi$ rad/s, or $-k\pi$ with $k = 1, 3, 5, ..., N-1$ [3, 6, 7, 9]. Fig. 3 shows the phase response of two, four, and six cascaded allpass filters, highlighting the frequencies where notches will appear in a phasing scenario.

The position of the spectral notches in a phaser is changed over time by modulating the break frequencies of the filters using an LFO. The LFO usually has a frequency in the range from 0.1 to 10 Hz, and its waveform is typically triangular or sinusoidal. In analog phasers, the LFO changes the physical values of components in the allpass filters. In digital implementations, the LFO controls the coefficients of the allpass filters [3, 6, 8].

A limitation of using first-order allpass filters to implement phasers is that the width of the notches cannot be modified independently. Both the notch width and depth are controlled using the wetness parameter W, which changes both of them simultaneously. Additionally, controlling the location of the notches is not trivial. Both of these issues can be addressed by using second-order allpass filters instead, as they allow a more independent control of the notch characteristics [9].

Instead of using allpass filters, phasers can also be implemented by using cascaded notch filters [3, 10]. Notch filters allow better control of notch locations and depth than allpass filters. However, phasers implemented in this manner are more complex and expensive than their allpass counterparts [3, 10]. In analog phasers, more complexity means more physical components, higher costs, and bigger pedals. On the other hand, in a digital implementation the complexity may not represent such a big problem, but should still be considered [3].

Some phasers incorporate a feedback loop in their structure, feeding the output of the allpass chain back into its input. This architecture is illustrated in Fig. 1. A unit delay was inserted to this feedback path to avoid having a delay-free loop [8]. The summing of the output of the allpass chain with the input signal causes a similar effect as the summing of the output with the original signal, as again the two added signals have the phase difference of $-\pi$, -3π, -5π, etc. at certain frequencies. In practice, the feedback loop causes drastic changes in the magnitude response of the phaser, since resonances are introduced and the frequencies between the notches are boosted. This changes the overall shape of the magnitude response, as shown in Fig. 4 [7, 8].

3. MEASURING A TIME-VARIANT SYSTEM

Linear time-invariant (LTI) systems are typically analyzed by measuring their impulse response using a sine sweep, white noise, or another test signal of long duration. The time-varying nature of phasers means that these methods are unreliable, as the response of the system changes over time. To counter this, the measurement signal used must be short and should be played repeatedly through the system in order to observe the response of the system at different moments in time.

In this work, we assume the system under study to be linear and time-variant (LTV), i.e. it varies over time but does not intro-

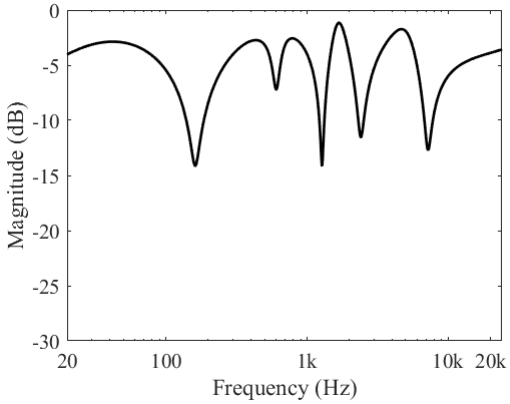


Figure 4: *Magnitude response of a phaser having ten first-order allpass filters and a feedback loop ($F_b = 0.4$, $G = 0.5$, $W = 0.5$). Four of the allpass filters use the coefficient value -0.9490 and six use -0.8991.*

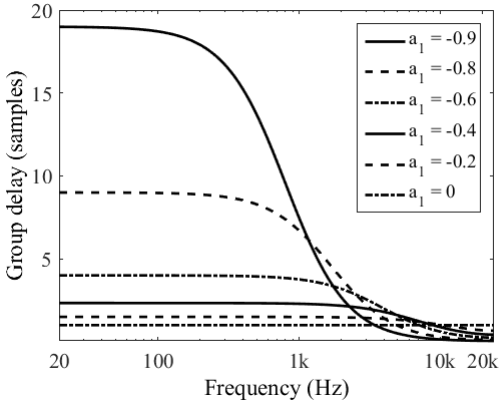


Figure 5: *Group delay of a first-order digital allpass filter with different coefficient values.*

duce harmonic distortion. This condition is mostly true for filter-based non-saturating effects, such as the phaser, when the input signal level is not very high. The following subsections describe the measurement process used to evaluate the frequency response of an LTV system.

3.1. Measurement signal

To measure the response of an LTV system at an arbitrary moment in time, we must first design a short test signal. This signal should be short enough to guarantee the system being measured will remain fairly static during the measurement process. Arnardottir et al. used an impulse train to measure the behavior of a time-varying tape delay effect [11]. Alternatively, a very short chirp can be used for this purpose, since it yields an improved signal-to-noise ratio (SNR). Such a signal is described in [5] and [12]. Repeatedly evaluating the instantaneous response of the system over a period of time will allow us to observe its time-varying response.

Following the work of [5] and [12], we can synthesize a chirp using the impulse response of a cascade of first-order allpass filters. The idea is that even though allpass filters do not alter the frequency content of a signal, when dozens of them are cascaded, the result is a system that introduces significantly different delays for different frequencies. These differences cause frequencies to be played at different times, similar to a sine sweep. The advantage of synthesizing a chirp in this manner is that extremely short signals, e.g. 10 or 50 ms long, can be produced with relative ease [5, 12]. In sine sweep measurements, the minimum length of the signal is limited by the low frequencies [12]. The lower the frequencies that need to be measured, the longer the sine sweep must be.

As indicated by (3), the pole a_1 of the first-order allpass filter determines the overall phase response of the filter. Since the phase-shift and group delay are closely related, this pole also changes the group delay of the filter. The group delay of a single first-order allpass filter can be calculated with the following formula:

$$\tau_g(\omega) = \frac{1 - a_1^2}{1 + 2a_1 \cos(\omega) + a_1^2}, \tag{4}$$

where ω is angular frequency. Fig. 5 illustrates the group delay of a first-order allpass filter for different poles values. As can be noted, the group delays are very small even for large pole values, less than 20 samples. However, when several allpass filters are cascaded, their group delays are added, so with ten filters in cascade the total group delay at low frequencies is almost 200 samples [5].

The group delay curves show that the resulting chirp-like impulse response first plays the high frequencies and then sweeps downwards to the lowest frequency. A chirp constructed using 64 allpass filters by setting $a_1 = -0.9$ is shown in Fig. 6. We can see that the chirp is only about 1500 samples long (approx. 30 ms), yet it still covers all the frequencies of interest [5].

In addition to meeting the requirement of being short, this chirp can be easily repeated periodically, which makes it a suitable test signal for LTV devices. Additionally, the way in which this signal is generated has one significant advantage; since the chirp is generated by feeding an impulse into an allpass filter chain, this impulse can be reconstructed by processing the chirp backwards through the same filter chain. In fact, this operation implements the deconvolution, which is needed to retrieve the impulse response of a system. This makes measuring LTV systems with a signal containing multiple chirps simple, as the measured signal can then be processed backwards through the allpass filter chain. The resulting multiple responses can be analyzed, generating a picture of how the magnitude response of the system changes over time [12].

The fact that the unprocessed chirp can be reverted back into an impulse means that several chirps can be superimposed (added) within a short time period. In fact, they may overlap in time. The only restriction for the spacing of the chirps is that when the signal is reverted back through the allpass chain, the resulting impulse responses should not overlap. If they do, the analysis cannot be performed so easily. A measurement signal with a duration of approx. 1 s, synthesized using chirps generated by 64 allpass filters with coefficient $a_1 = -0.9$ (see Fig. 6), spaced at 30 ms intervals is shown in Fig. 7. This figure illustrates the test signal in the time domain, containing a total of 33 chirps in 1 s.

3.2. Measurement analysis

The aim of this work is to measure a real phaser pedal and observe all the interesting aspects about its behavior. For a basic phaser

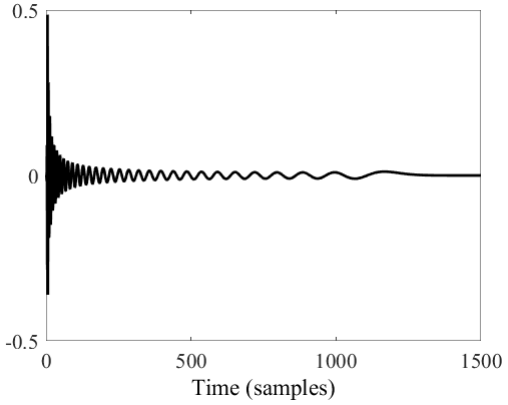


Figure 6: *Time-domain waveform of a single chirp synthesized using 64 first-order allpass filters with pole parameter* $a_1 = -0.9$.

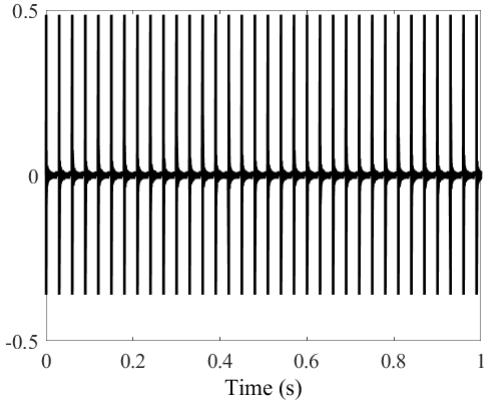


Figure 7: *Time-domain waveform of an example measurement signal synthesized using 33 overlapping chirps (see Fig. 6).*

pedal with only a speed knob, this can be achieved by measuring the pedal at different speed settings. From the obtained measurements we wish to observe the behavior and shape of the LFO, along with the number and position of the notches over time.

The notches caused by a phaser can be identified by finding the minima of the magnitude responses. To do so, the following procedure was followed. First, the global minimum of the magnitude spectrum was found. Once the location of the global minimum was known, the adjacent local maxima were found, and the area between these two local maxima was zeroed out. After the original global minimum of the spectrum was removed, the next one was found by repeating these steps. This process was repeated until all the minima were found.

In this study, when converting the magnitude values to dB-scale, a reference value of 1.7325 was used. This is the maximum value of the spectrum of the first chirp used on the first measurement. This value was chosen so that for all measurements, spectra, and other values can easily be compared with each other.

An interesting aspect about the proposed measurement method is how it reacts to changes in the length of the chirps forming the measurement signal. If the chirps are too long, the minima will appear to be 'out of sync'. Due to the nature of the chirp, high

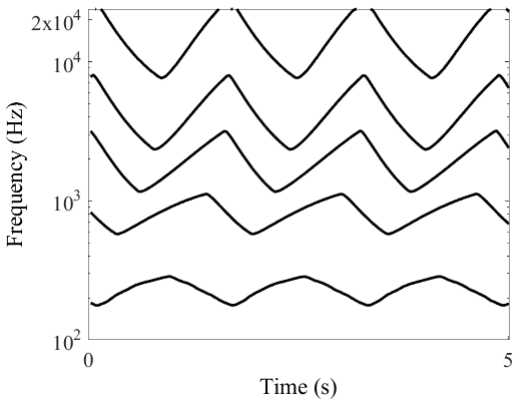


Figure 8: *Distorted minima locations measured with a chirp generated with 2,048 allpass filters, which is too long.*

frequencies are played before low frequencies. When high frequencies are processed, the minima will be at a given location; later, when the low frequencies pass through the system, the minima locations will have already moved. This was tested by playing signals with different chirp lengths through a phaser pedal and observing how the resulting data changed. The chirp length was adjusted by increasing the amount of cascaded allpass filters used to generate it. The chirp was generated with 2,048 allpass filters in cascade, with coefficient $a_1 = -0.9$, which resulted in a chirp that is roughly 40,000 samples long. The total test signal lasted for 240,000 samples, with the chirps overlapping. The measured results are shown in Fig. 8, where it can be clearly observed that the minima do not occur at the same time. This can be corrected by using shorter chirps. With these tests it was also noticed that increasing the chirp length increases the SNR of the measured signal.

As expected, when the chirp length is very short, the SNR of the measurement becomes low and the measured magnitude spectrum has clear artefacts. Therefore, it is desirable to find an optimal chirp length with which the spectral minima appear to be synchronized at the same time with a good SNR. In this work, the measurements were conducted with a chirp generated using a cascade of 64 first-order allpass filters with $a_1 = -0.9$, which was repeated every 0.03 s.

4. MEASUREMENT RESULTS

The phaser pedal chosen for this study is the Fame Sweet Tone Phaser PH-10. The similarities between the general appearance of this pedal and the MXR Phase 100 suggest that the measured pedal is a clone of the Phase 100.

The measured pedal has a single 'speed' potentiometer and two switches. These switches are grouped and labeled simply as 'Intensity'. Based on the observed behavior, we named the switch on the right-hand side the 'LFO switch' and the one on the left the 'feedback switch'. Both switches have two modes ('on' and 'off') so overall there are four working modes.

The pedal was measured at different LFO speed settings for all switch combinations. Seven steps were used: 0%, 17%, 34%, 50%, 67%, 84%, and 100% of the speed range. These steps were chosen because the knob of the speed potentiometer was seven-

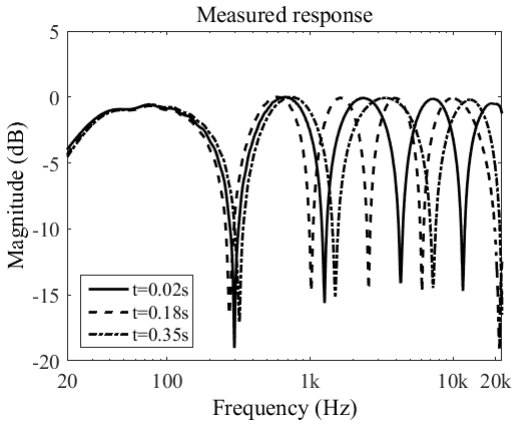


Figure 9: *Magnitude response of the measured pedal at different times.*

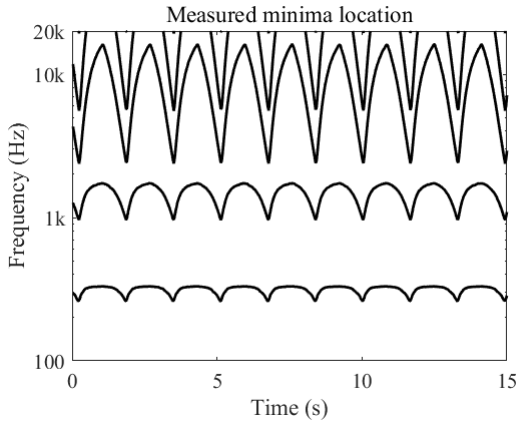


Figure 10: *Estimated notches of the phaser pedal over time (LFO speed at approx. 50%, LFO and feedback switches both 'off').*

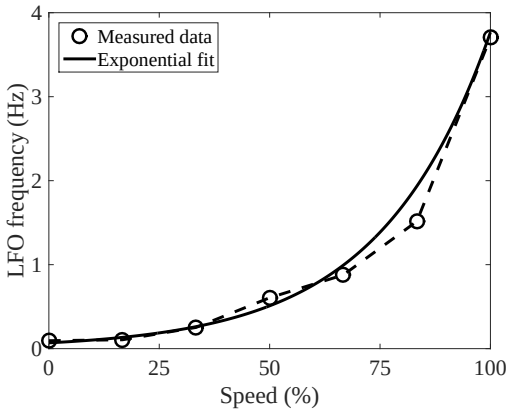


Figure 11: *Measured LFO frequency with seven different speed settings and an exponential fit.*

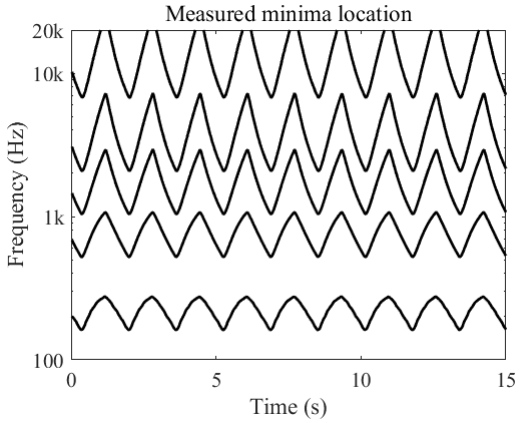


Figure 12: *Estimated notches of the measured phaser over time with the LFO switch 'on' and speed at approx. 50%.*

sided and the pedal did not have any marks printed on it. Then the angles of the knob made it easy to see when the speed had increased by $1/7^{\text{th}}$ of its range. All measurements were conducted at a 48-kHz sampling rate.

4.1. Measurements with both switches OFF

The Fame pedal was measured with both switches in the 'off' position using a 25-s test signal synthesized using non-overlapping chirps placed at 30 ms intervals. Fig. 9 shows the measured magnitude response at three different times. In this figure, we can observe that the pedal introduces five notches, with the fifth one lying well above 20 kHz. This means the phaser most likely operates around a 10-stage filter network, another indication that the studied pedal is a clone of the MXR Phase 100.

Fig. 10 shows the estimated notch trajectories during a 15-second period. These data were generated using the method described in the previous section. This figure reveals the existence of five notches, with the fifth one lying mostly outside the audible range. It can also be noticed that the first notch has a range of about a quarter of an octave, the second one has roughly one octave, and the third one has about three octaves. Additionally, the notch trajectories seem to resemble a full-wave rectified sinewave. This suggests that the LFO has this waveform in the 'off' mode, rather than the typical triangular or sinusoidal shapes.

Moving on with the measurements, Fig. 11 shows the effect of the speed potentiometer, which is approximately exponential. The LFO range can be observed to be from about 0.1 Hz to 3.7 Hz. These frequencies were estimated from the time differences between the lowest notch locations. For instance, in Fig. 10 the time difference between the lowest notch locations is approx. 1.6 s, which corresponds to the LFO frequency of about 0.6 Hz.

4.2. Effect of the LFO switch

The pedal was measured again with the LFO switch in the 'on' position. The overall appearance of the resulting magnitude responses were observed to be similar to the case of both switches turned 'off'. The LFO speed measurements also remained unchanged. The main observed difference was the behavior of the notches over time. This can be seen in Fig. 12, where it can be noticed that the notch trajectories resemble triangular waves now. Secondly, all five notches are now visible in the spectrum. In the previous measurements only three notches were within the audible band during an entire cycle; the fourth notch was only partially

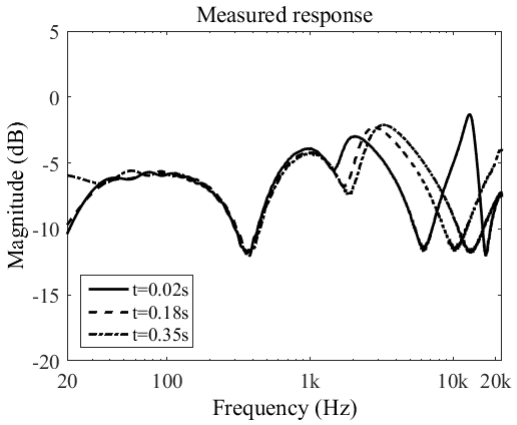


Figure 13: *Magnitude responses of the measured phaser at different times with the feedback switch in the 'on' position (LFO speed 50%).*

visible and the fifth notch was above 20 kHz throughout the measurements. Therefore, it can be deduced that the right switch of the pedal modifies the waveshape and range of the LFO, causing the effect to be perceived as more 'intense', hence the label on the pedal's enclosure.

4.3. Effect of the feedback switch

From the measurements of the pedal with the feedback switch 'on' it was observed that the minima and speed behavior remained identical to the 'off' case. However, differences were observed in the magnitude response of the pedal. These changes are quite drastic, as the gain between the notches is boosted, the notch depths vary considerably with time, and the overall shape of the magnitude responses at any given time is considerably different. This behavior is shown in Fig. 13. It can be assumed that the left switch activates a feedback path to the allpass filter chain, as the changes in the response are similar to those seen in Fig. 4.

As a final note on the measured data, when both switches were in the 'on' position, the resulting responses were a combination of both effects: The notch behavior was similar to that of Fig. 12, and the magnitude response behaved as in Fig. 13.

5. MODELING OF THE MEASURED PHASER

A gray-box digital model of the measured phaser was implemented based on the collected data. Several design choices were made based on basic knowledge of the internal workings of phaser pedals, hence the term gray-box. First, it was decided to use first-order allpass filters only, as these are the building blocks of standard analog phaser pedals [2, 9]. This design makes it impossible to control the width of the notches independently, but is faithful to the original analog design. Since measurements showed that the phaser introduces five notches, the designed called for a network of ten first-order allpass filters paired in groups of two. Then, with the help of a simplified study of the circuit [13], it was observed that the first and last pairs of filters were not being modulated. This can be written as

$$A_1 = A_2 = A_9 = A_{10} \tag{5}$$

$$A_3 = A_4 = A_5 = A_6 = A_7 = A_8 \tag{6}$$

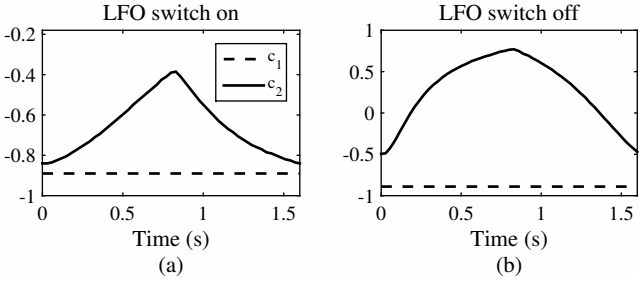


Figure 14: *Estimated values of allpass filter coefficients c_1 and c_2 for a single LFO cycle, when the LFO switch is (a) 'on' and (b) 'off'.*

where A_m is the transfer function of the m^{th} allpass filter. This means that only two coefficients are needed to model the phaser.

The location of the notches over a single LFO cycle for both positions of the LFO switch were readily available from the measurements shown in Figs. 10 and 12. An algorithm was devised to find the pair of coefficients that best fit the measured data. This was done by iterating over a large set of coefficient combinations and evaluating the phase response of the allpass network at the target frequencies analytically and computing the mean absolute error. This optimization was performed using a standard sampling rate of 44.1 kHz.

After some initial tests, it was observed that the pair of coefficients that gave the smallest error performed rather poorly at low frequencies. This was attributed to the inaccurate nature of analog systems. Even when the filters used in the original analog pedal are supposed to be identical, the high tolerance levels of the components make this condition impossible. To minimize this effect, we decided to neglect the fourth and fifth notches for the case of the LFO switch 'off' and the fifth notch for the case of the LFO switch 'on' during the optimization process.

Several coefficient evaluations demonstrated the initial assumption that the first and last pair of allpass filters are static. Based on testing, coefficient c_1, i.e. the coefficient for filters A_1, A_2, A_9, and A_{10} was assigned a fixed value of -0.89. Fig. 14(a) shows the estimated value for coefficient c_2, i.e. the coefficient for filters $A_3 ... A_8$, for a single cycle when the LFO switch is 'on'. As inferred from the measurements, this coefficient can be modulated using a triangular LFO. The range of this LFO must be between $[-0.84, -0.39]$. Fig. 14(b) shows the estimated values for c_2 when the LFO switch is in the 'off' position. This LFO can be modeled using a rectified sinewave ranging between $[-0.49, 0.77]$.

The LFO speed control was modeled with a weighted least squares fit of an exponential function to the data shown in Fig. 11:

$$f_{\text{lfo}} = 0.069e^{0.040s}, \tag{7}$$

where $0 \geq s \geq 100$ is the LFO speed parameter and f_{lfo} is the fundamental frequency of the LFO.

To validate the model, a test signal was phasered with the model using coefficients estimated previously. A second-order IIR filter was used to simulate the DC blocker observed in the measurements. A DC blocking filter is common in analog audio electronics and is used to remove hum [14] or to assure that the waveform is symmetric before it enters a distortion unit. The transfer function

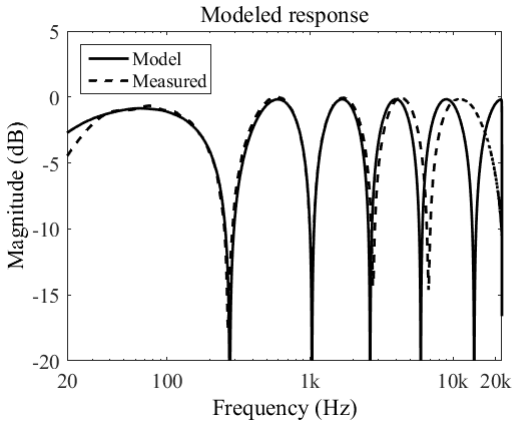


Figure 15: *Magnitude response of the modeled phaser compared to the measured response with the LFO switch 'off'. Parameters* $G = 0.5$ *and* $W = 0.5$ *were used.*

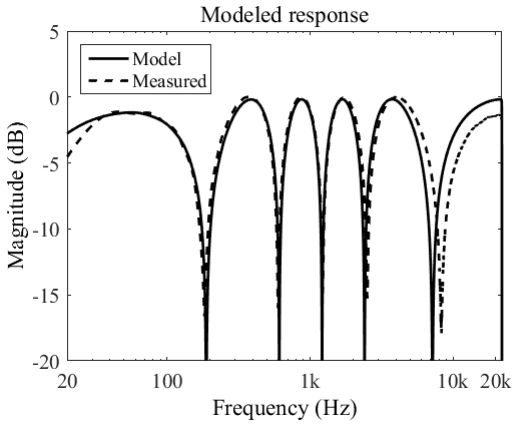


Figure 16: *Magnitude response of the modeled phaser compared to the measured response with the LFO switch 'on'* ($G = W = 0.5$)*.*

of the second-order DC blocker is

$$H_{\mathrm{DC}} = \frac{1+p}{2}\frac{1-z^{-2}}{1-pz^{-2}}, \tag{8}$$

where $p = 0.992$ determines the pole location, which is an octave lower than in the corresponding first-order design [14].

The phasered test signal was then analyzed using the same method as the pedal measurements, and the magnitude responses and notch frequencies at each time were found. A comparison of the model's magnitude response to that of the measured pedal in Fig. 15 and Fig. 16 shows a close resemblance, but it can also be seen that the notch frequencies are matched much better at low than at high frequencies. Perceptually, the model accuracy in the highest octave is not highly important, however.

When comparing the behavior of notches in the model, shown in Fig. 17 and Fig. 18, and in the measured pedal (see Fig. 10 and Fig. 12), it can be noticed that they are very similar. The measured and the modeled notches have nearly the same the period, waveform, and frequency range. This can easily be seen in Fig. 19 in which the notches of both the measured and the modeled phaser are synchronized in time and are plotted for roughly two periods

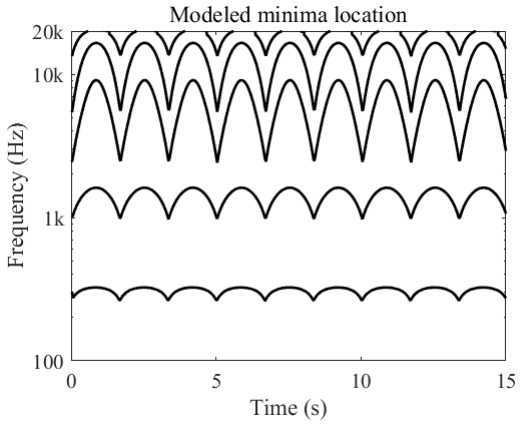


Figure 17: *Estimated notch frequencies of the digital phaser model emulating the case when the LFO switch is 'off' (full-wave rectified sine LFO), the feedback switch is 'off', and the speed is set at 54%. Cf. Fig. 10.*

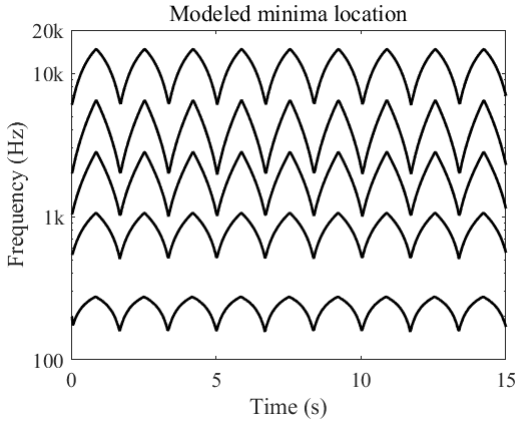


Figure 18: *Estimated notch frequencies of the digital phaser model emulating the case when the LFO switch is 'on' (triangular LFO), the feedback switch is 'off', and the speed is set at 54%. Cf. Fig. 12.*

with zoomed in frequency axis to better see the differences. One reason for the minor deviation is the LFO waveshape, which was modeled as strictly triangular in this case. However, an analog LFO of a phaser does not produce a perfect triangular waveform, but there is some degree of curvature in the waveform.

Furthermore, modeling of the phaser with the feedback was tested, but it was noticed that the model used could not produce a response identical to the measured one with the feedback switch 'on'. An example of the model's response with feedback is shown in Fig. 4. It can be compared to the response of the measured system in Fig. 13. In the model, the notch locations are again correct, but the resonances between the notches cannot be matched with those of the measured system.

Sound examples and Matlab code related to this work are available online [15].

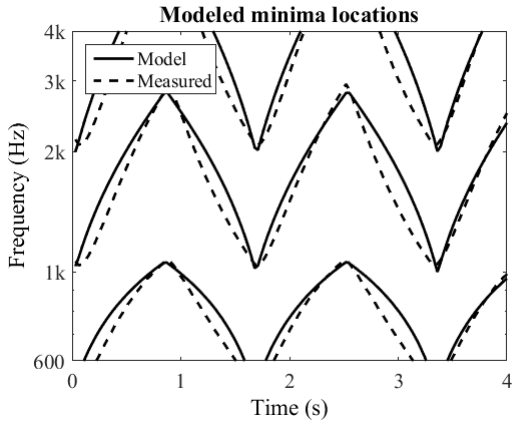


Figure 19: *Comparison of three notches of the measured and modeled phaser as a function of time (LFO switch 'on'; triangular LFO).*

6. CONCLUSIONS

A method to measure time-varying audio effects was proposed. It uses a sequence of very short chirps to measure the impulse response of the system under measurement multiple times, and from them a characterization of the system can be produced. In the case of phasers, the spectral notches at different times are extracted from the measurements. The behavior of the magnitude response and notch locations in time reveals the operating principle of the measured phaser.

The method was tested by measuring a phaser pedal, and the results were analyzed. For example, the LFO frequency for different speed settings was measured and an exponential mapping was observed. The effect of the two switches of the pedal were analyzed based on the measurements, with one changing the LFO signal's waveshape and range while the other adding a feedback around the allpass filter loop. A gray-box virtual analog model of the phaser was calibrated based on the measurements.

In the gray-box model the idea was to emulate the measurements, making it easier through some knowledge of the pedal. Overall the resulting model behaved similarly as the data after which it was modeled. However, the modeling of the feedback feature was not accurate in details. It could be improved in the future by using a modeling technique which does not need a fictitious unit delay in the feedback loop [16, 17].

7. ACKNOWLEDGMENT

The helpful comments of anonymous reviewers are gratefully acknowledged.

8. REFERENCES

[1] J. Pakarinen and D. T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music J.*, vol. 33, no. 2, pp. 85–100, 2009.

[2] F. Eichas, M. Fink, M. Holters, and U. Zölzer, "Physical modeling of the MXR Phase 90 guitar effect pedal," in *Proc. 17th Int. Conf. Digital Audio Effects (DAFx-14)*, Erlangen, Germany, 2014, pp. 153–158.

[3] U. Zölzer, *DAFX: Digital Audio Effects, 2nd Edition*, 2011.

[4] F. Eichas, M. Fink, and U. Zölzer, "Feature design for the classification of audio effect units by input/output measurements," in *Proc. 18th Int. Conf. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov.-Dec. 2015, pp. 27–33.

[5] V. Välimäki, J. S. Abel, and J. O. Smith, "Spectral delay filters," *J. Audio Eng. Soc.*, vol. 57, no. 7/8, pp. 521–531, Jul.–Aug. 2009.

[6] W. M. Hartmann, "Flanging and phasers," *J. Audio Eng. Soc*, vol. 26, no. 6, pp. 439–443, June 1978.

[7] M. L. Beigel, "A digital "phase shifter" for musical applications, using the Bell Labs (Alles-Fischer) digital filter module," *J. Audio Eng. Soc.*, vol. 27, no. 9, pp. 673–676, Sept. 1979.

[8] A. Huovilainen, "Enhanced digital models for analog modulation effects," in *Proc. 8th Int. Conf. Digital Audio Effects (DAFx-05)*, Madrid, Spain, Sept. 2005, pp. 155–160.

[9] J. O. Smith, "An allpass approach to digital phasing and flanging," in *Proc. Int. Comp. Music Conf.*, Paris, France, Oct. 1984, pp. 103–109.

[10] F. Esqueda, V. Välimäki, and J. Parker, "Barberpole phasing and flanging illusions," in *Proc. 18th Int. Conf. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov.–Dec. 2015, pp. 87–94.

[11] S. Arnardottir, J. S. Abel, and J. O. Smith III, "A digital model of the Echoplex tape delay," in *Proc. Audio Eng. Soc. 125th Conv.*, San Francisco, CA, Oct. 2008.

[12] D. Griesinger, "Impulse response measurements using allpass deconvolution," in *Proc. Audio Eng. Soc. 11th Int. Conf. Test & Measurement*, Portland, OR, May 1992.

[13] J. D. Sleep, "MXR Phase 100 Phase Shifter [Online]," `http://www.generalguitargadgets.com/pdf/ggg_p100_sc.pdf`, 2009, Accessed 9 June 2016.

[14] J. Pekonen and V. Välimäki, "Filter-based alias reduction for digital classical waveform synthesis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Las Vegas, NV, Mar.–Apr. 2008, pp. 133–136.

[15] R. Kiiski, F. Esqueda, and V. Välimäki, "Supplementary materials for time-variant gray-box modeling of a phaser pedal [Online]," `http://research.spa.aalto.fi/publications/papers/dafx16-phaser/`, 2016, Accessed 10 June 2016.

[16] F. Fontana, F. Avanzini, and D. Rocchesso, "Computation of nonlinear filter networks containing delay-free paths," in *in Proc. Int. Conf. Digital Audio Effects (DAFx-04)*, Naples, Italy, Oct. 2004, pp. 113–118.

[17] D. Medine, "Dynamical systems for audio synthesis: Embracing nonlinearities and delay-free loops," *Applied Sciences*, vol. 6, no. 5/134, 10 May 2016, Available online at `http://www.mdpi.com/2076-3417/6/5/134`.

BLACK-BOX MODELING OF DISTORTION CIRCUITS WITH BLOCK-ORIENTED MODELS

Felix Eichas, Udo Zölzer

Department of Signal Processing and Communications,
Helmut Schmidt University
Hamburg, Germany
felix.eichas@hsu-hh.de

ABSTRACT

This paper describes black-box modeling of distortion circuits. The analyzed distortion circuits all originate from guitar effect pedals, which are widely used to enrich the sound of an electric guitar with harmonics. The proposed method employs a block-oriented model which consists of a linear block (filter) and a nonlinear block. In this study the nonlinear block is represented by an extended parametric input/output mapping function. Three distortion circuits with different nonlinear elements are analyzed and modeled. The linear and nonlinear parts of the circuit are analyzed and modeled separately. The Levenberg–Marquardt algorithm is used for iterative optimization of the nonlinear parts of the circuits. Some circuits could not be modeled with high accuracy, but the proposed model has shown to be a versatile and flexible tool when modeling distortion circuits.

1. INTRODUCTION

Virtual analog modeling has widely been done before. The focus in virtual analog modeling of electric guitar equipment lies on recreating an analog reference device as close as possible. In [1–7] this has been done with great success by analyzing the analog reference circuit and transferring the circuit into a mathematical model, which is able to recreate the original's characteristics. This circuit-based approach achieves very convincing results and the digital model is mostly indistinguishable from the analog reference device for the human ear. But this precise approach also has drawbacks. To create the digital model the circuit diagram has to be known, as well as the characteristic curves of every nonlinear circuit element, e.g. diodes, transistors, transformers or vacuum tubes. If no circuit diagram is obtainable, time-consuming reverse engineering of the circuit has to be performed, as described in e.g. [8].

Another drawback of this method is the computational effort which arises due to nonlinear circuit elements. For every nonlinear circuit element at least one nonlinear equation has to be solved per time step. Depending on the nonlinear solver and the initial parameter set, this can drastically influence the computational load of the digital model. Although Holmes et al. described a method for improving the nonlinear solver in [9], the computational effort is still high, especially for complex circuits with multiple nonlinearities.

Alternative approaches for modeling of distortion circuits are found in [10] and [11]. Block-oriented models are used to represent the distortion circuit. A block-oriented model generally consists of linear blocks and static nonlinear blocks. Some common topologies have conventional names, like Hammerstein model (static nonlinearity followed by a filter in series), Wiener model (linear filter followed by a static nonlinearity in series) or Wiener–Hammerstein model (filter followed by static nonlinearity followed by filter). In [10] a parallel, generalized polynomial Hammerstein model is used. Each parallel branch represents the different harmonic components of the reference system. In [11] principal component analysis is used to reduce the complexity of the model presented in [10].

In [12] a completely parametric Wiener–Hammerstein model is used to automatically identify distortion guitar effect pedals. The structure of the used Wiener–Hammerstein model is the series connection of a parametric filter followed by a nonlinear block, which is also used in this work, followed by another parametric filter. The filters are identified by iterative parameter optimization, using low-level noise as input signal. To find the initial parameter set for the nonlinear part of the model, a time-consuming grid search is performed.

As in [12] the basic idea behind this work is to analyze and model nonlinear distortion circuits, without knowledge of the circuit itself. Only input/output measurements are performed to adjust the parameters of a block-oriented, nonlinear model to recreate the characteristics of the analog reference device. In this work, only parts of distortion circuits should be modeled, which use different electronic components to create distortion. In this work the Wiener–Hammerstein model from [12] could be reduced to an extended Wiener model (filter followed by nonlinear block), because all chosen circuit parts did not have any filter at the output. The aim of this work is to analyze how well such a simple model can recreate the behavior of the circuits with an automated optimization procedure.

Section 2 describes the analyzed distortion circuits and Sec. 3 explains the topology of the extended Wiener model. Sections 4 and 5 describe the modeling process. Section 6 compares the identified models to the reference circuits and Sec. 7 concludes the paper.

2. HARDWARE

Three different distortion circuits were analyzed in this work. The first one was a simple diode clipper with pre-amplification of the input signal, the second one a BJT distortion stage of an Electro Harmonix - Big Muff Pi™ and the operational amplifier based distortion stage of an Ibanez - Tube Screamer™. All circuits were simulated with a spice circuit simulator, `LTSpice` [13].

2.1. Diode Clipper

The circuit of the diode clipper can be seen in Fig. 1. It is an extension of the diode clipper circuit from [14] with an additional non-inverting amplifier.

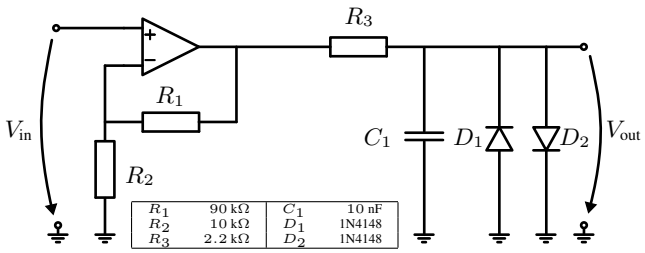


Figure 1: *Circuit of the diode clipper with pre-amplification.*

The input signal is amplified by the non inverting amplifier with a factor of $1 + R_1/R_2 = 10$ and the resulting amplified signal passes through a low-pass RC circuit with cut-off frequency $f_c = 1/(2\pi R_3 C_1) \approx 7.23\,\text{kHz}$ and finally through the anti-parallel 1N4148 diodes. The supply voltage of the operational amplifier was set to a relatively high value of $\pm 30\,\text{V}$ to avoid additional clipping. The simulated operational amplifier was a TL072.

2.2. Big Muff Distortion Stage

The distortion stage of the Big Muff can be seen as an extended BJT transimpedance gain stage as described in [15]. It was also modeled in [7], using wave digital filters.

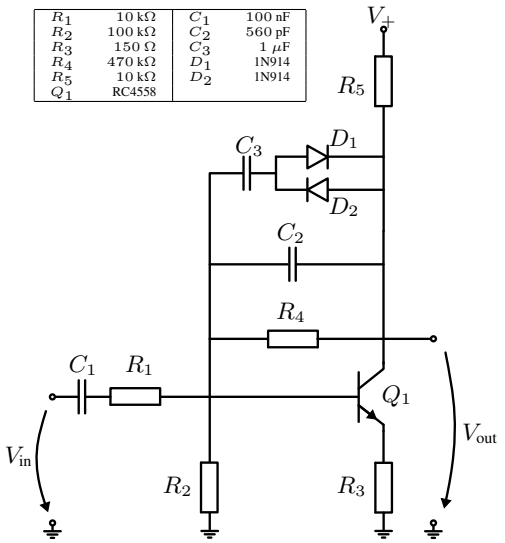


Figure 2: *Circuit of a Big Muff transimpedance gain stage.*

Fig. 2 shows the circuit around the NPN transistor. In addition to the feedback from collector to base via R_4 and C_2, there are two anti-parallel diodes D_1, D_2 as well as the capacitor C_3. These diodes introduce further clipping in addition to the clipping of the BJT circuit itself. The supply voltage was set to $V_+ = 9\,\text{V}$.

2.3. Tube Screamer Distortion Stage

The Tube Screamer is based on an operational amplifier gain stage, also described in [15–18]. The input signal is amplified and additionally distorted by two anti-parallel diodes in the feedback path from output to negative input of the op-amp.

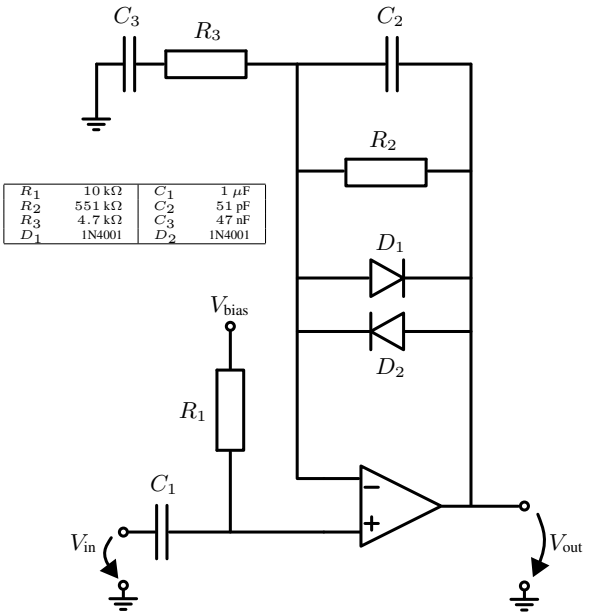


Figure 3: *Circuit of a Tube Screamer operational amplifier gain stage.*

Fig. 3 shows the circuit with resistor R_2 which is a potentiometer in the Tube Screamer circuit. It takes values from $R_{2,\text{min}} \approx 51\,\text{k}\Omega$ to $R_{2,\text{max}} = 551\,\text{k}\Omega$. In this work the potentiometer was always set to $R_{2,\text{max}}$ to maximize the amplification of the operational amplifier and thus the distortion of the output signal. The bias voltage was set to $V_{\text{bias}} = 4.5\,\text{V}$, which is half the supply voltage of the operational amplifier. The used op-amp was a general-purpose amplifier RC4558 by Texas Instruments.

3. MODEL TOPOLOGY

The digital model which was chosen to represent these distortion circuits is an extended Wiener model. It consists of a linear time invariant block followed by a nonlinear block. Fig. 4 shows the block diagram of the extended Wiener model. In this work the LTI

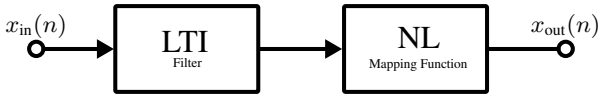


Figure 4: *Block diagram of a Wiener model.*

block is represented by a FIR filter, but it could be easily modified to include any other LTI system e.g. state-space systems or IIR filters. The nonlinear block consists mainly of a mapping function, mapping input amplitude to output amplitude. Figure 5 illustrates the function principle of a static, memory-less mapping function. Each input sample is processed by the nonlinear equation, creating

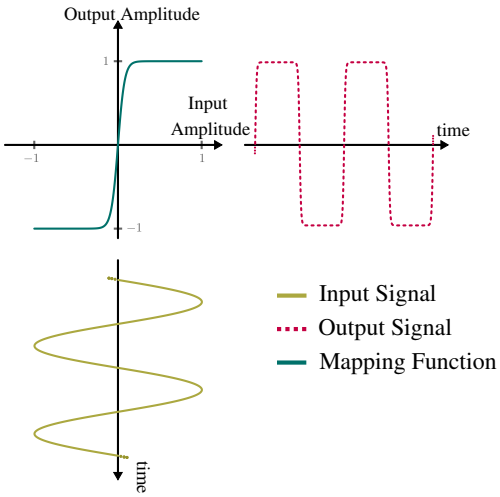


Figure 5: *Function principle of a static mapping curve.*

a distorted output signal. Some extensions have been made to the mapping function (see Sec. 5.2). Especially the side-chain with low-pass filter leads to a nonlinear block which is not memoryless anymore. So the model is not a Wiener model in the classical sense, it is an extended Wiener model with distortion-circuit specific refinements.

4. MODELING: LINEAR PART

The linear part of all circuits was measured with exponential sine sweeps, as described in [19]. The sweep is described in terms of start frequency $\omega_1 = 2\pi \mathrm{f}_{\mathrm{start}}/\mathrm{f}_s$, stop frequency $\omega_2 = 2\pi \mathrm{f}_{\mathrm{stop}}/\mathrm{f}_s$ and amplitude A,

$$x_{\mathrm{sw}}(n) = A \cdot \sin\left(\frac{\omega_1 \cdot (L-1)}{\log\left(\omega_2/\omega_1\right)} \cdot \left(e^{\left(\frac{n}{L-1}\log(\omega_2/\omega_1)\right)} - 1\right)\right), \tag{1}$$

where L is the total length of the sweep in samples. An inverse-filter signal to the sweep can be created which fulfills the condition

$$x_{\mathrm{sw}}(n) * x_{\mathrm{inv}}(n) \approx c \cdot \delta(n - n_0). \tag{2}$$

This means that the sweep convolved with the inverse filter yields a Dirac delta function which is only shifted in time and scaled by some factor c. Due to the assumption that very low signal levels will pass through the linear region of the distortion circuits, the maximum amplitude of the sweep was set to $A = 0.01\,\mathrm{V}$, to ensure that no nonlinear distortion occurs while measuring the output signal. The output sweep $y_{\mathrm{sw}}(n)$ is recorded and convolved with the inverse filter

$$x_{\mathrm{inv}}(n) = x_{\mathrm{sw}}(L-1-n) \cdot (\omega_2/\omega_1)^{\frac{-n}{L-1}} \tag{3}$$

to get the impulse response of the system

$$h(n) = \frac{1}{c} \cdot x_{\mathrm{inv}}(n) * y_{\mathrm{sw}}(n). \tag{4}$$

When using this technique $h(n)$ does not only contain the linear response of the system. It also contains the impulse responses for higher order harmonics, as described in [10]. Therefore the impulse response has to be segmented in time-domain and is normalized to a maximum magnitude of 0 dB in frequency-domain. Afterwards it is saved and directly used as FIR filter coefficients in the digital model.

This methods performs better than the iterative parameter optimization approach based on white noise, described in [12]. The small-signal impulse response is directly measured and used in the model, instead of iteratively adapting several filters to approximate the frequency response of the circuit.

5. MODELING: NONLINEAR PART

Modeling of the nonlinear part is done by creating a reference signal, in this case the output voltage of each circuit to a specific (known) input signal. Afterwards it is compared to the output of the digital model to compute the error between both signals according to a cost function. This cost function has to be minimized to find the optimal set of parameters for the given reference signal.

5.1. Input Signal

When designing the input signal it is important to consider the influence of the parameters on the output. The nonlinear block in the extended Wiener model is frequency independent, which means that it is not necessary for the input signal to excite more than one frequency. But it is most important to excite all possible amplitudes of the input signal, so their modification by the reference system can be observed. A single frequency sine wave with logarithmically rising amplitude was used as the input signal,

$$x_{\mathrm{nl}}(n) = a(n) \cdot \sin\left(\frac{2\pi f_0 n}{f_s}\right). \tag{5}$$

The fundamental frequency was set to $f_0 = 1000\,\mathrm{Hz}$. The amplitude scaling function $a(n)$ is logarithmically increasing from a start value of $a(1) = 1 \cdot 10^{-5}$ to the largest value $a(N) = 1$, with N as the total amount of samples.

5.2. Parametric Nonlinear Block

The parametric nonlinear block is based on a mapping function, described in Sec. 3. It was already used in [12] to model distortion pedals. Figure 6 shows the nonlinear block. Its main component is

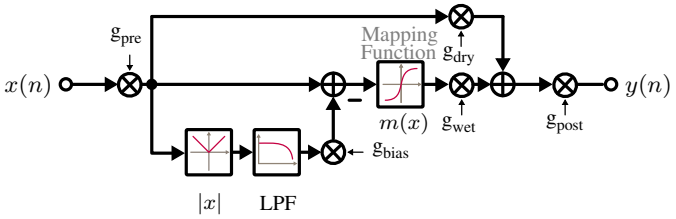


Figure 6: *Nonlinear mapping function with extensions.*

the mapping function, which is a combination of three hyperbolic tangent functions. The combination of the three functions, with the amplitude of the signal x as input, is shown by (6).

$$m(x) = \begin{cases} \tanh(k_p) - \left[\frac{\tanh(k_p)^2-1}{g_p}\tanh(g_p x - k_p)\right] & \text{if } x > k_p \\ \tanh(x) & \text{if } -k_n \leq x \leq k_p \\ -\tanh(k_n) - \left[\frac{\tanh(k_n)^2-1}{g_n}\tanh(g_n x + k_n)\right] & \text{if } x < -k_n \end{cases} \tag{6}$$

The additional terms for $x > k_p$ and $x < -k_n$ are needed to ensure that $m(x)$ has a continuous derivative at the connection points k_n and k_p. Besides the mapping function, the nonlinear block has a pre-gain $\mathrm{g_{pre}}$ to scale the input signal and a post-gain $\mathrm{g_{post}}$ to scale the output. The side-chain envelope detector, consisting of absolute value calculation and low-pass filtering with a cut-off frequency of $f_{\mathrm{c,LP}} = 5\,\mathrm{Hz}$, and a dry/wet mixing stage. Note that the dry gain parameter is calculated automatically by $\mathrm{g_{dry}} = 1 - \mathrm{g_{wet}}$. The envelope subtraction from the direct signal is used to emulate the signal-dependent bias-point shift which occurs for vacuum tubes or transistors. It is an adapted version of the so called 'tube stage' from [20], but to avoid feedback, it is constructed in a feed-forward loop. Please note that this extension prevents the nonlinear block from being memoryless, because the output is dependent on the previous values of the input signal. This leads to a total amount of eight parameters, four gains and four parameters for the mapping function, which are combined in the parameter vector

$$\mathbf{p} = \begin{pmatrix} \mathrm{g_{pre}} & \mathrm{g_{bias}} & k_p & k_n & \mathrm{g}_p & \mathrm{g}_n & \mathrm{g_{wet}} & \mathrm{g_{post}} \end{pmatrix}^T. \tag{7}$$

With $g_{\mathrm{wet}} = 1 - g_{\mathrm{dry}}$.

5.3. Parameter Optimization

The nonlinear block is initialized with a parameter set that only introduces a slight distortion for high signal levels. To improve the robustness of convergence during optimization, the parameters are optimized in three different steps. The used algorithm is the gradient-based Levenberg–Marquardt optimization procedure [21, 22].

5.3.1. Levenberg–Marquardt

Consider a reference system which produces output $y_{\mathrm{sys}}(n)$ (after A/D conversion) and the corresponding digital model which produces output $y_{\mathrm{mod}}(n, \mathbf{p})$, depending not only on the input signal $x(n)$, but also on the parameter vector $\mathbf{p}$. The residual

$$\mathrm{res}(n, \mathbf{p}) = y_{\mathrm{sys}}(n) - y_{\mathrm{mod}}(n, \mathbf{p}) \tag{8}$$

describes the difference of reference system and digital model.

The Levenberg–Marquardt algorithm is a combination of the *gradient-descent* and the *Gauss–Newton* methods. The parameter vector $\mathbf{p}$ is updated by,

$$\mathbf{\Delta p} = \left(\mathbf{J}^T\mathbf{J} + \lambda \cdot \mathrm{diag}(\mathbf{J}^T\mathbf{J})\right)^{-1} \cdot \mathrm{grad}(\mathbf{p}) \ . \tag{9}$$

$\mathbf{J}$ is the Jacobi matrix, where each column represents the derivative of the residual with respect to the parameter vector $\mathbf{p}$,

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathrm{res}(1,\mathbf{p})}{\partial p_1} & \cdots & \frac{\partial \mathrm{res}(1,\mathbf{p})}{\partial p_M} \\ \frac{\partial \mathrm{res}(2,\mathbf{p})}{\partial p_1} & \cdots & \frac{\partial \mathrm{res}(2,\mathbf{p})}{\partial p_M} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathrm{res}(N,\mathbf{p})}{\partial p_1} & \cdots & \frac{\partial \mathrm{res}(N,\mathbf{p})}{\partial p_M} \end{bmatrix} \tag{10}$$

with M as the total number of parameters and N as the total length of the residual. The gradient

$$\mathrm{grad}(\mathbf{p}) = \mathbf{J}^T \cdot \begin{bmatrix} \mathrm{res}(1,\mathbf{p}) & \cdots & \mathrm{res}(N,\mathbf{p}) \end{bmatrix}^T \tag{11}$$

describes in which direction of each entry in the parameter vector we have to descend to minimize the error between reference system and digital model.

For large values of λ the algorithm behaves more like *gradient-descent*, while for small values of λ the algorithm behaves more like *Gauss–Newton* [23]. Although *Gauss-Newton* is an efficient method there exist cases where the algorithm needs a long time to converge into the minimum, or does not converge at all. If λ is initially set to a relatively small value and the cost function does not decrease, e.g.

$$C(\mathbf{p}) < C(\mathbf{p} + \mathbf{\Delta p}),$$

λ is increased to get quicker convergence with the *gradient-descent* method.

If the current step was successful, e.g.

$$C(\mathbf{p}) > C(\mathbf{p} + \mathbf{\Delta p}),$$

the parameter vector for each iteration k is updated,

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{\Delta p}_k,$$

and λ is decreased to make use of the advantageous properties of the *Gauss–Newton* algorithm near the solution.

5.3.2. Cost Function

The choice of the cost function is crucial for the robustness of the optimization process. The straight-forward approach would be to simply calculate the difference of digital model output and analog reference output in time-domain, as shown by (8). But if the phase characteristic of reference and model is not matched perfectly, the time-domain error is quite high, which does not necessarily represent the human perception of the difference between the two signals. To neglect any phase shift between reference system and digital model the cost function is designed to match the envelopes of both systems. The envelopes are calculated for positive and negative signal amplitudes separately, because the nonlinear mapping function, described in Sec. 5.2 is able to shape positive and negative amplitudes independently. To calculate the envelope the signals are half-wave rectified and low-pass filtered by a second order IIR low-pass with a cut-off frequency of $f_c = 5\,\mathrm{Hz}$. To calculate the envelope for negative amplitudes the signals are multiplied with -1 before half-wave rectification.

Since the Levenberg Marquardt algorithm is gradient-based, convergence into a local minimum is possible when the initial set of parameters is too far from the global minimum of the cost function. This is why the main optimization procedure is divided into three parts.

1. Optimize parameters for positive amplitudes (while ignoring parameters which change negative amplitude)
2. Optimize ignored parameters from step 1. for negative amplitudes
3. Refine all parameters for positive and negative amplitudes

Another benefit of matching the signals' envelopes is a robust identification. The output of the digital model depends on fewer parameters in step one and two than in step three. In step one, for example, the parameters k_n and g_n can be ignored, because they only influence negative amplitudes. If the initial parameter set leads to a slightly nonlinear mapping function, the Levenberg–Marquardt algorithm always matches the envelopes satisfactorily. If this is done for positive and negative amplitudes, the parameter vector will be close to the optimal solution when starting the refinement in step three.

6. RESULTS

In this section the results of the modeling process are presented and each digital model is compared to the corresponding reference system.

6.1. Metrics Definition

To rate the result of the optimization, a recorded guitar track was played back through both systems and the percentage of error energy or 'error to signal ratio' (ESR) was calculated. It is defined as the ratio of error energy to the energy of the reference output,

$$\mathrm{ESR} = \frac{E_{\mathrm{res}}}{E_{\mathrm{sys}}} = \frac{\sum_{n=-\infty}^{\infty} |y_{\mathrm{sys}}(n) - y_{\mathrm{mod}}(n, \mathbf{p})|^2}{\sum_{n=-\infty}^{\infty} |y_{\mathrm{sys}}(n)|^2}. \tag{12}$$

Another way to calculate the difference is via the correlation coefficient, which describes the linear dependence of two random variables. The computation of the correlation coefficient is shown by

$$\rho(\mathrm{A,B}) = \frac{\mathrm{cov(A,B)}}{\sigma_{\mathrm{A}}\sigma_{\mathrm{B}}}, \tag{13}$$

where $\mathrm{A} = y_{\mathrm{sys}}(n)$ and $\mathrm{B} = y_{\mathrm{mod}}(n, \mathbf{p})$ are the random variables (in our case reference and model output), cov(A,B) is the covariance of A and B and $\sigma_{\mathrm{A,B}}$ is the standard deviation of the random variables.

6.2. Modeling Results

The results of the modeling process are shown in Table 1. The diode clipper obtained the best results with an error to signal ratio of only ESR $= 5.78\%$ and a correlation coefficient of $\rho = 0.9983$. The informal listening test proved that there is no noticeable difference between the output of the circuit and the digital model. The

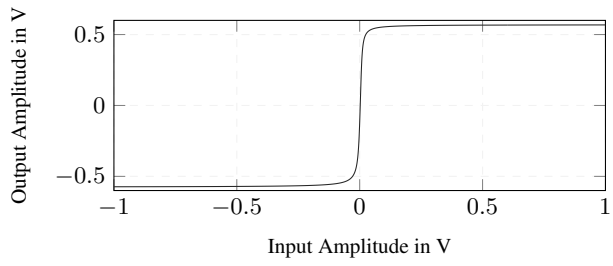
Circuit	**ESR**	ρ(**A,B**)
Diode Clipper	0.0578	0.9983
Big Muff	0.0901	0.9578
Tube Screamer	0.1832	0.9062

Table 1: *Results of the modeling process.*

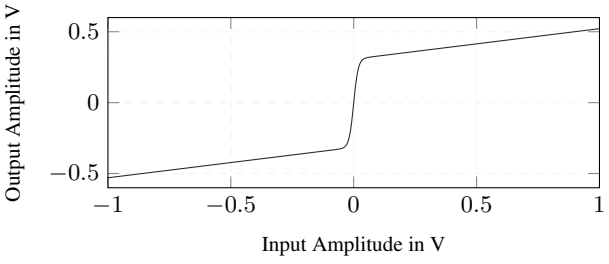
results for the other circuits are not as good. The Big Muff BJT gain stage has an ESR $= 9.01\%$ and a correlation coefficient of $\rho = 0.9578$, which already leads to a slightly perceivable difference between the signals. This can be explained by the feedback path from the collector of the NPN transistor to its base (see Fig. 2). This feedback path is not modeled in the extended Wiener model, so the result of the modeling process is only an approximation of the real circuit.

The Tube Screamer has an ESR$= 18.32\%$ and a correlation coefficient of $\rho = 0.9062$, which also leads to a small noticeable difference between digital model and circuit output. This difference can also be explained by the simplicity of the extended Wiener model. In the original circuit, there is a feedback path from operational amplifier output to its negative input. This could be modeled by a parallel path to the mapping function with a filter, whose frequency response is unknown, because only the global frequency response can be measured, without detailed measurements of the analog circuit.

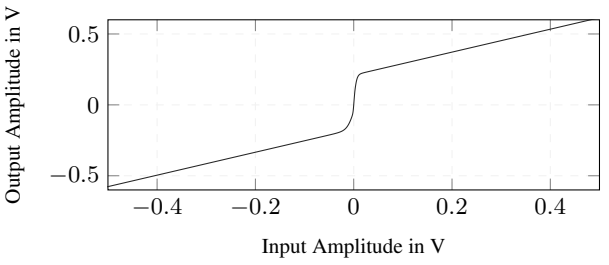
Figure 7 shows the mapping function of the *digital model* after the optimization procedure is finished. It can be seen that the diode clipper circuit has no dry signal at the output of the nonlinear block, because the slope of the mapping function at $x(n) = -1$ and $x(n) = 1$ is zero.



(a) Diode Clipper



(b) Big Muff



(c) Tube Screamer

Figure 7: *Nonlinear mapping functions $m(x)$ after optimization.*

The Big Muff circuit has a little dry signal mixed together with the wet signal, which can be explained as a 'compensation' of the missing feedback path in the model. The Tube Screamer has a lot

of dry signal mixed together with the distorted wet signal, which confirms the assumption that the circuit can be modeled with a parallel dry signal, which can be mixed with the distorted signal.

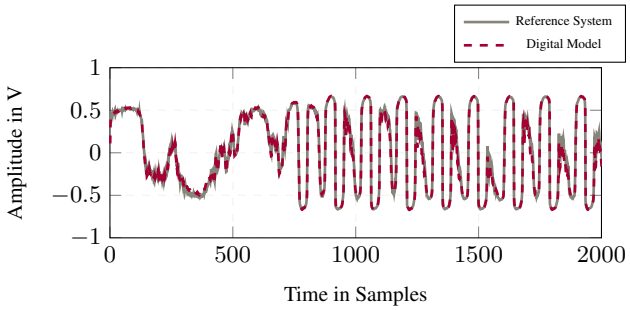
All mapping functions look symmetrical, which is due to using the same model for the two diodes in each `Spice` simulation, leading to the same shape for positive and negative amplitudes. Only in the Tube Screamer circuit there is a slight difference between the shape of positive amplitudes and negative amplitudes, which is also visible in the mapping function, Fig. 7 (c), because the transition from steep middle part of the mapping function to higher amplitudes is a little softer for negative amplitudes.

The time-domain signals for each reference circuit and their comparison to the corresponding model are shown in Fig. 8. The input signal was a self-recorded riff played on a stratocaster-type electric guitar using the humbucker bridge-pickup. The guitar was directly connected to an RME - Fireface 800 audio interface. For the diode clipper, the digital model waveform is very close to the waveform of the reference signal, which leads to no perceivable difference between the two signals. With a rising ESR value for Big Muff and Tube Screamer the waveform of the digital model differs more and more from the reference output. Generally, it can be observed that the difference in the waveforms is proportional to the input amplitude, because for the first 500 samples of the test signal, all models are close to the reference signal, while for higher input amplitudes (sample 600 to 2000) the model is not accurate enough to recreate the more complex reference circuits.

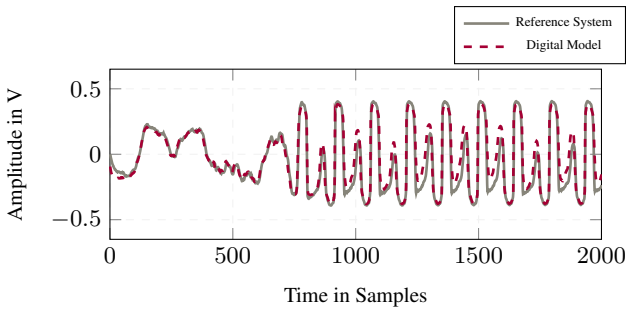
In addition to these scores, an informal listening test was conducted. The participants of the test were five experienced researchers in virtual analog modeling, who should test if they are able to hear a difference between digital model and reference signal. In case of the diode clipper none of the participants was able to hear a difference between simulation and reference. For Big Muff and Tube Screamer, the results were not as convincing, since every test subject was able to hear a difference. Nevertheless all of the participants confirmed that the overall characteristic of the reference device could be captured by the corresponding digital version.

6.3. Listening Examples

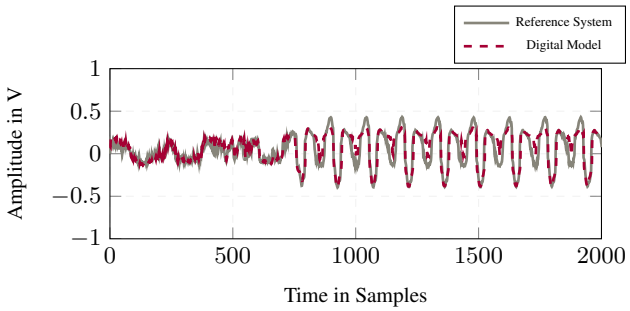
To give the reader an impression of the model and compare it to the reference system some listening examples were created. The input signal consists of single notes, played by an electric guitar and decaying guitar chords as well as decaying single notes, played by an electric bass-guitar. No effect has been used to alter the signals before or after processing them with the digital model or the reference circuit simulation. The listening examples can be found on-line at [24].



(a) Diode Clipper



(b) Big Muff



(c) Tube Screamer

Figure 8: *Time-domain response to a recorded guitar input for all three circuits.*

7. CONCLUSION

Three distortion circuits have been modeled by an extended Wiener model, consisting only of one linear time-invariant block (filter) and an extended nonlinear mapping block, which maps input amplitudes to output amplitudes. The modeling was very successful for the diode clipper circuit, because it matches the model topology. Although the model is not able to emulate more complex circuits perfectly, it is able to recreate the reference circuit to a satisfying degree, given its simplicity. To really capture all the characteristics of a distortion circuit, especially for bigger, more complex circuits, the model needs to be expanded. A serial approach, concatenating several simple models, to refine the results would be conceivable. Another expansion could be a feedback path with a unit delay, which allows more possibilities of shaping the waveform.

8. REFERENCES

[1] D.T. Yeh and J.O. Smith, "Simulating guitar distortion circuits using wave digital and nonlinear state-space formulations," in *Proc. Digital Audio Effects (DAFx-08)*, Espoo, Finland, Sept. 1–4, 2008, pp. 19–26.

[2] D.T. Yeh, J.S. Abel, and J.O. Smith, "Automated physical modeling of nonlinear audio circuits for real-time audio effects: part 1—theoretical development," in *IEEE Trans. Audio, Speech, and Language Process.*, May 2010, vol. 18, pp. 203–206.

[3] J. Mačák, *Real-time Digital Simulation of Guitar Amplifiers*

as Audio Effects, Ph.D. thesis, Brno University of Technology, 2011.

[4] K. Dempwolf, *Modellierung analoger Gitarrenverstärker mit digitaler Signalverarbeitung*, Ph.D. thesis, Helmut-Schmidt-Universität, 2012.

[5] M. Holters and U. Zölzer, "Physical modelling of a wah-wah effect pedal as a case study for application of the nodal dk method to circuits with variable parts," in *Proc. Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 19–23, 2011, pp. 31–35.

[6] K.J. Werner, J.O. Smith, and J.S. Abel, "Wave digital filter adaptors for arbitrary topologies and multiport linear elements," in *Proc. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[7] K.J. Werner, V. Nangia, J.O. Smith, and J.S. Abel, "Resolving wave digital filters with multiple/multiport nonlinearities," in *Proc. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[8] C. Gnegy and K.J. Werner, "Digitizing the Ibanez Weeping Demon wah pedal," in *Proc. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[9] B. Holmes and M. van Walstijn, "Improving the robustness of the iterative solver in state-space modelling of guitar distortion circuitry," in *Proc. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[10] A. Novak, L. Simon, P. Lotton, and J. Gilbert, "Chebyshev model and synchronized swept sine method in nonlinear audio effect modeling," in *Proc. Digital Audio Effects (DAFx-10)*, Graz, Austria, Sept. 6–10, 2010.

[11] R.C.D. de Paiva, J. Pakarinen, and V. Välimäki, "Reduced-complexity modeling of high-order nonlinear audio systems using swept-sine and principal component analysis," in *Audio Engineering Society Conference: 45th International Conference: Applications of Time-Frequency Processing in Audio*, Mar. 2012.

[12] F. Eichas and U. Zölzer, "Block-oriented modeling of distortion audio effects using iterative minimization," in *Proc. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov 30 – Dec 3 2015.

[13] "Linear Technologies SPICE Simulator," Website, Available at `http://http://www.linear.com/designtools/software/` — accessed March 17th 2016.

[14] D.T. Yeh, J.S. Abel, and J.O. Smith, "Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinary differential equations," in *Proc. Digital Audio Effects (DAFx-07)*, Bordeaux, France, Sep. 10 – 15 2007.

[15] D.T. Yeh, *Digital implementation of musical distortion circuits by analysis and simulation*, Ph.D. thesis, Stanford University, 2009.

[16] D.T. Yeh, J.S. Abel, and J.O. Smith, "Simplified, physically-informed models of distortion and overdrive guitar effects pedals," in *Proc. Digital Audio Effects (DAFx-07)*, Bordeaux, France, Sep. 10 – 15 2007.

[17] R.C.D. Paiva, S. D'Angelo, J. Pakarinen, and V. Välimäki, "Emulation of operational amplifiers and diodes in audio distortion circuits," in *IEEE Trans. on Circuits and Systems*, Bordeaux, France, Oct. 2012, vol. 59.

[18] K.J. Werner, V. Nangia, A. Bernardini, J.O. Smith, and A. Sarti, "An improved and generalized diode clipper model for wave digital filters," in *Audio Engineering Society Convention 139*. Audio Engineering Society, 2015.

[19] A. Farina, "Simultaneous measurement of impulse response and distortion with a swept-sine technique," in *Audio Engineering Society Convention 108*, Paris, France, 2000.

[20] J. Pakarinen and D.T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music Journal*, vol. 33, no. 2, pp. 85–100, 2009.

[21] K. Levenberg, "A method for the solution of certain problems in least squares," *Quarterly of applied mathematics*, vol. 2, pp. 164–168, 1944.

[22] D.W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the Society for Industrial & Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[23] T. Strutz, *Data Fitting and Uncertainty: A practical introduction to weighted least squares and beyond*, Vieweg and Teubner, 2010.

[24] "Listening Examples," Website, Available at `http://www.hermelinband.de/felix/dafx16/audio_examples_local_version.html` — accessed March 16th 2016.

PHYSICAL MODEL PARAMETER OPTIMISATION FOR CALIBRATED EMULATION OF THE DALLAS RANGEMASTER TREBLE BOOSTER GUITAR PEDAL

Ben Holmes and Maarten van Walstijn

Sonic Arts Research Centre,
School of Electronics, Electrical Engineering and Computer Science,
Queen's University Belfast
Belfast, U.K.
{bholmes02, mvanwalstijn}@qub.ac.uk

ABSTRACT

In this work we explore optimising parameters of a physical circuit model relative to input/output measurements, using the Dallas Rangemaster Treble Booster as a case study. A hybrid metaheuristic/gradient descent algorithm is implemented, where the initial parameter sets for the optimisation are informed by nominal values from schematics and datasheets. Sensitivity analysis is used to screen parameters, which informs a study of the optimisation algorithm against model complexity by fixing parameters. The results of the optimisation show a significant increase in the accuracy of model behaviour, but also highlight several key issues regarding the recovery of parameters.

1. INTRODUCTION

Accurate simulation of vintage audio circuits by physical modelling invariably requires the determination of component parameters. In principle, parameters can be obtained through measurement of individual components. This can present a practical dilemma though as it requires isolation and therefore deconstruction of the circuit, which particularly for vintage circuits carries hazards such as component damage. A possible way around this problem is to estimate the parameters solely from input/output (I/O) measurements, which can be taken without disassembly. Only requiring the most basic interface with a system, I/O measurements can provide significant information regarding its behaviour with relatively little effort.

The literature provides several techniques that use I/O measurements to calibrate black-box models, which have the advantage that knowledge of the actual physical component parameters is not required, and are generally designed to allow a relatively straight-forward model parameter estimation. For nonlinear systems a method of using a swept-sine to excite a system and applying an inverse filter to the output, described as 'nonlinear convolution', was proposed by Farina [1, 2]. This method was initially used for acoustic systems but was further applied to nonlinear audio circuitry in the form of Chebyshev [3] and generalised polynomial Hammerstein models [4]. Accurate modelling of the phase response is not guaranteed using this method, requiring synchronisation between input and output measurements [4, 5]. Additionally, a single set of kernels only accurately models the system at a single input level. As nonlinear systems can vary based upon amplitude it is necessary to model a continuous range of input amplitudes, which has been achieved using interpolation between sets of kernels [6].

Alleviation of these issues can be achieved using *a priori* knowledge of a system. For example, a block-oriented parametric Wiener-Hammerstein model designed specifically for distortion circuitry [7] can be said to use partial system knowledge. The partial knowledge, in this case the form of the nonlinear behaviour, constrains the identification procedure to aid in the capture of the effect's behaviour.

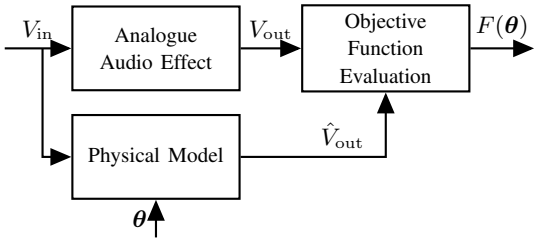


Figure 1: *Diagram illustrating the evaluation of a parameter set $\boldsymbol{\theta}$ by comparison of a physical model with said parameters and the desired analogue audio effect.*

Going a step further, physical models are based on the presumption of total knowledge of the system's behaviour. As well as accurately modelling a system's phase response and response to input amplitude (dependent upon accurate modelling of the governing physical laws), physical models can additionally capture parametric behaviour exhibited in audio effects by potentiometers, which change the behaviour of the audio effects. The two most popular physical modelling techniques, Wave Digital Filters [8] and state-space models [9], both have methods of working with parametric behaviour, with particular focus on efficiency given for state-space models [10]. However, this gain in functionality comes with a large increase in model and computational complexity. Furthermore, unlike models derived from system measurements, parameters for these models are typically extracted from nominal information available in datasheets and schematics which may not be representative of a real circuit. An exception to this strategy is when complex component models are tabulated using measurements [11]), but this requires isolation of the component.

In this paper we explore the utilisation of I/O measurements to improve the accuracy of physical models relative to specific real-world circuits. Figure 1 illustrates the evaluation of an objective function $F(\boldsymbol{\theta})$ as a measure of the fit of a physical model to a specific audio effect based upon a comparison between the circuit output voltage V_{out} and the model output voltage $\hat{V}_{\text{out}}$. The central challenge then becomes finding a parameter set that minimises the objective function for a sufficiently general excitation signal, which can be done using iterative optimisation methods. Such an approach evidently relies on the ability of the chosen physi-

cal model to capture the circuit's behaviour. An immediate further question that arises is whether a parameter set that has a low objective function value but nevertheless deviates in one or more parameters (the search space proves to typically contain many of such local minima) should be considered successful. While ideally one aims to recover the parameter set that lies closest to a physical target set, in practice (i.e. when optimising on measured results) such a target reference is not available, and the only workable criterion is the objective function. Given this lack of a direct measure of the parameter accuracy, it is proposed here that the optimisation is deemed successful if (a) the parameters lie within a physically feasible range and (b) the optimised set results in accurate model output over the relevant range of input signals and circuit potentiometer settings. The first criterion implies that the optimisation should be constrained, and the second criterion motivates performing a post-optimisation validation of the optimised parameter set using a map of driving signals of different amplitudes and frequencies within the expected input ranges.

The chosen case study for this paper is the Dallas Rangemaster Treble Booster pedal, the schematic of which is shown in Figure 2. The circuit creates a high-pass filter effect and distortion caused by the nonlinear behaviour of the germanium transistor, a Mullard OC44. The Rangemaster is a suitable initial test case for exploration of parameter optimisation of nonlinear audio effects units because despite the relative simplicity of the circuit its study fully exposes the same key challenges that can be expected in more complex systems. Three individual I/O data sets are taken of the circuit, one simulated from a stochastic parameter set, and two measured from the circuit using different transistors.

The rest of this paper is structured as follows: Section 2 discusses the selected model, details the approach used to optimise the model based on measurements, and also describes the measurement procedure. Section 3 details the application of sensitivity analysis to the model to screen the parameter set, informing a study of the performance of the optimisation algorithm with respect to increasing complexity in the model. Section 4 presents the results of the optimisation upon the I/O data sets and analyses the results using a validation data set. Sound examples are available on the first author's website[1].

2. OPTIMISATION METHOD

2.1. Circuit Description

2.1.1. Time Domain Model

A circuit can be characterised by its topology and component's behaviour. While the behaviour of two-pin linear components such as resistors and capacitors can typically be characterised with sufficient accuracy using simple laws involving a single parameter (e.g. Ohm's Law), nonlinear components such as transistors or vacuum tubes are usually modelled with several parameters. The Nodal DK-method provides a structure for automated derivation of models from this information in the form of a netlist [12], and for this reason was chosen to create the models used for the optimisation procedure. This enables direct optimisation of the parameters of the circuit as opposed to e.g. state-space matrices.

The Nodal DK-method creates a discrete-time state-space model of the form

$$\boldsymbol{x}[n] = \boldsymbol{A}\boldsymbol{x}[n-1] + \boldsymbol{B}\boldsymbol{u}[n] + \boldsymbol{C}\boldsymbol{f}(\boldsymbol{v}_{\mathrm{n}}[n]) \tag{1}$$

[1] `http://bholmesqub.github.io/DAFx16/`

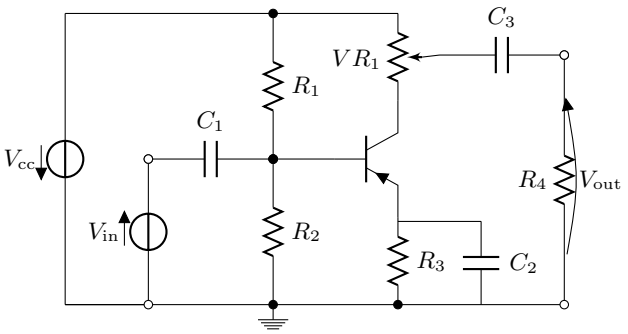


Figure 2: *Schematic of the modelled Dallas Rangemaster Treble Booster. The potentiometer VR_1, named 'Set' on the original pedal, controls the output gain or volume of the circuit.*

$$\boldsymbol{y}[n] = \boldsymbol{D}\boldsymbol{x}[n-1] + \boldsymbol{E}\boldsymbol{u}[n] + \boldsymbol{F}\boldsymbol{f}(\boldsymbol{v}_{\mathrm{n}}[n]) \tag{2}$$

$$\boldsymbol{v}_{\mathrm{n}}[n] = \boldsymbol{G}\boldsymbol{x}[n-1] + \boldsymbol{H}\boldsymbol{u}[n] + \boldsymbol{K}\boldsymbol{f}(\boldsymbol{v}_{\mathrm{n}}[n]) \tag{3}$$

where $\boldsymbol{x}$, $\boldsymbol{u}$ and $\boldsymbol{y}$ represent the state, input and output respectively, and the matrices $\boldsymbol{A} - \boldsymbol{H}$ and $\boldsymbol{K}$ specify the linear combinations that are used to update the model. The behaviour of nonlinear components is modelled by the function $\boldsymbol{f}(\boldsymbol{v}_{\mathrm{n}})$, where $\boldsymbol{v}_{\mathrm{n}}$ represents the voltages across the nonlinear components. This function is specific to the model, e.g. a model for diodes might use the Shockley diode equation. A comprehensive description of the modelling technique used can be found in [10], with the adaptation to potentiometers described in [13]. Relating this model to the Dallas Rangemaster circuit, $\boldsymbol{u}$ is the vector of the voltages $[V_{\mathrm{in}} \quad V_{\mathrm{cc}}]^{\mathrm{T}}$ applied to the circuit, and $\boldsymbol{y}$ is the modelled output voltage $\hat{V}_{\mathrm{out}}$.

It is important to note that the Dallas Rangemaster is parametric, but that the potentiometer is near equivalent to a linear scaling in output voltage. This can be inspected by testing two different models: the first with the potentiometer wiper set to 20% towards full volume, the second with the potentiometer wiper set to 100% and the output voltage scaled by 0.2. A chirp test signal between 20 Hz-3 kHz with a peak voltage of $0.2\,\mathrm{V}$ was processed by both models, the comparison revealing a mean squared error of $6\,\mu\mathrm{V}$. For this reason, one measurement of the circuit with the potentiometer set at 100% is sufficiently accurate, removing the need to model the parametric behaviour.

An additional simplification is the removal of the power supply bypass capacitor, originally placed across V_{cc}. The capacitor creates a low-pass filter with the internal resistance of the power supply, which serves to smooth changes in the supplied voltage. As the focus of this paper is the usage of I/O measurements of the signal path, the supply voltage is presumed to be constant, therefore making the capacitor obsolete.

2.1.2. Bipolar Junction Transistor

To model the nonlinear behaviour of the BJT, the Ebers-Moll equations are used. This model has been widely used within the field of VA modelling [14, 10] due to its computational efficiency in comparison with more complex BJT models. The Ebers-Moll model is represented by two currents with the third being calculated using Kirchhoff's current law. Here, the base and collector currents were selected:

$$I_{\mathrm{B}} = \frac{I_{\mathrm{S}}}{\beta_{\mathrm{F}}}\left(\mathrm{e}^{\frac{V_{\mathrm{EB}}}{NV_{\mathrm{T}}}} - 1\right) + \frac{I_{\mathrm{S}}}{\beta_{\mathrm{R}}}\left(\mathrm{e}^{\frac{V_{\mathrm{EB}}-V_{\mathrm{EC}}}{NV_{\mathrm{T}}}} - 1\right) \tag{4}$$

$$I_{\mathrm{C}} = I_{\mathrm{S}}\left(\mathrm{e}^{\frac{V_{\mathrm{EB}}}{NV_{\mathrm{T}}}} - 1\right) - I_{\mathrm{S}}\frac{\beta_{\mathrm{R}}+1}{\beta_{\mathrm{R}}}\left(\mathrm{e}^{\frac{V_{\mathrm{EB}}-V_{\mathrm{EC}}}{NV_{\mathrm{T}}}} - 1\right). \quad (5)$$

The currents are controlled by the voltages of the emitter-base junction V_{EB} and the emitter-collector junction V_{EC}. The parameters are: I_{S}, the saturation current, β_{F} and β_{R}, the forward and reverse current gains, and N, the ideality factor. The ideality factor is not used in the original Ebers-Moll model [15], but is commonly included in SPICE models, and has been included in the model so that the model can match a larger range of behaviour. Temperature was not measured for this study, so the parameter N also serves to correct the value of $V_{\mathrm{T}} = 25.85\,\mathrm{mV}$, which presumes a temperature of 300 K.

2.2. Excitation Signal

Here the term excitation signal refers to the input voltage applied to the system. To expose the model to a range of frequencies, a multi-sine signal was selected, consisting of a sum of sinusoids between two frequency boundaries. This can be represented by

$$V_{\mathrm{in}}[n] = \sum_{m=m_{\mathrm{l}}}^{m_{\mathrm{u}}} A_m \sin(2\pi m f_0 nT + \phi_m) \quad (6)$$

where $f_0 = f_{\mathrm{s}}/N_{\mathrm{s}}$ and T is the sampling period $1/f_{\mathrm{s}}$. The lower and upper boundaries m_{l} and m_{u} provide a method of bandlimiting the signal, by selecting values closest to the desired lower and upper frequency boundaries. Bandlimiting is a desirable property as it enables a convenient method of focusing the measurements, for example on the expected frequency range of a guitar.

The phase terms ϕ_m are generated using Schroeder phases [16]

$$\phi_m = -2\pi \sum_{l=1}^{m-1} (m-l)A_m, \quad m = m_{\mathrm{l}}, m_{\mathrm{l}}+1, ..., m_{\mathrm{u}}. \quad (7)$$

This selection of phases distributes the sinusoids such that the peak to peak voltage is minimised, creating a flat amplitude envelope.

To determine the amplitude terms A_m, it is helpful to consider the circuit with the BJT linearised using the Hybrid Pi model [17]. The amplitude response of the linearised circuit illustrated in Figure 3 shows a significant boost to high frequencies which could prevent low frequencies from being represented in the output signal. One method of alleviating this issue is to filter the input signal using the inverse transfer function, i.e.

$$A_m = |H(j\omega_m)|^{-1}, \quad \omega_m = 2\pi m f_0 \quad (8)$$

where $|H(j\omega_m)|$ is the magnitude of the transfer function of the linearised circuit. The value of A_m is infinite at DC, but as the multi-sine signal can be band-limited, this can be managed by excluding frequencies close to DC.

Finally, to ensure that the range of voltages is sufficient, a Hann window is applied, and the signal is scaled so that the peak voltage is at 2 V. The excitation signal applied to the system has a length of 200 ms, containing frequencies between $50\,\mathrm{Hz} - 2\,\mathrm{kHz}$, as illustrated in Figure 4.

2.3. Optimisation Algorithm

Initial experiments in optimising the parameter set using a gradient descent method revealed many local minima in the search space.

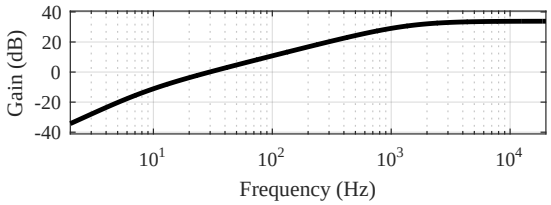


Figure 3: *Amplitude response of the linearised Dallas Rangemaster circuit.*

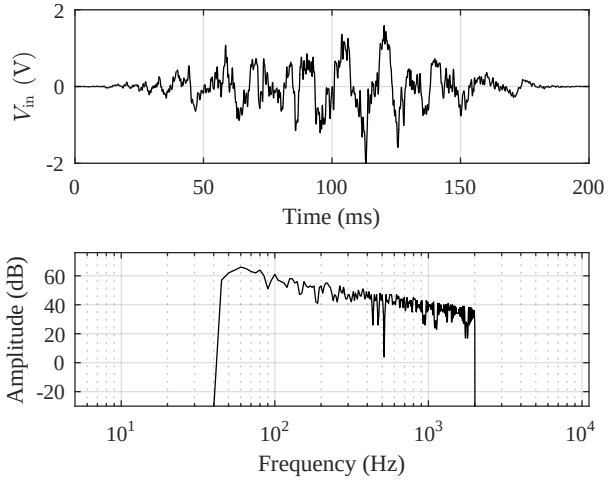


Figure 4: *Time and frequency domain representations of the excitation signal used for the I/O measurements. The ripple in the amplitude response is caused by the Hann window.*

To overcome local minima, a hybrid metaheuristic/descent method was implemented, using a Genetic Algorithm (GA) as a technique to provide a more exhaustive search. The MATLAB function `ga` provides a versatile implementation that allows the usage of floating point values as opposed to bit strings which were used in the original design of GA.

The basis of GA is to mimic principles observed in genetics and natural selection. The following description is of the MATLAB specific implementation; for a comprehensive introduction to GA see e.g. [18]. An *individual* refers to an instance of the set of parameters that characterise the model, $\boldsymbol{\theta} = [\theta_1, ..., \theta_k]^{\mathrm{T}}$ where k is the number of parameters in the set. The fitness of the individual is defined by an objective function chosen here to be of least-squares design:

$$F(\boldsymbol{\theta}) = \frac{1}{N_{\mathrm{s}}} \sum_{n=1}^{N_{\mathrm{s}}} (V_{\mathrm{out}}[n] - \hat{V}_{\mathrm{out}}[\boldsymbol{\theta}, n])^2, \quad \boldsymbol{\theta} \in \boldsymbol{\Omega} \subset \mathbb{R}^k \quad (9)$$

where V_{out} is the measured output signal, $\hat{V}_{\mathrm{out}}$ is the modelled output, and $\boldsymbol{\Omega}$ is the search space for the parameter set.

To initiate the algorithm, a *population* of individuals is randomly generated using a uniform distribution within a range of parameter values, further discussed in Section 3. The fitness value of each individual is then evaluated. A population size of 1000 individuals was chosen by increasing the size of the population until

the difference in successive generation's fitness values was negligible. Upon determining the fitness of the population, *parents* are selected to create the next generation. The most fit individuals are selected as *elites*, which are passed to the next generation without change. The remaining *children* are created from either crossover or mutation. Crossover children are created from two parents, with individual parameters selected from both parents, combined to create children. Mutation children are created from a single parent by stochastically changing parameter values. Parents are selected using a stochastic selection which helps to maintain a diverse population (i.e. a high variance of parameter values).

The implemented algorithm creates the next generation using 5% of the past population as elites, and of the remaining population 70% are generated from crossover, and 30% are generated from mutation. This process is then repeated, with the best performing parameter sets being retained while new parameter sets are generated using crossover and mutation. The main termination criterion of the algorithm is a limit of 100 generations.

A critical issue encountered when stochastically selecting parameter sets for the Dallas Rangemaster is that the simulation can fail. This happens when the nonlinear solver does not converge to the root of the equation [19]. To counteract this, failing parameter sets are regenerated using the stochastic technique used to generate the initial population. This is repeated until the simulation is successful. In addition to this, the least fit 10% of individuals are re-generated each generation to improve population diversity.

The GA algorithm is combined with the interior point method [20] which uses the top 1% of the population as starting points. This ensures that local minima are found which is not guaranteed using GA on its own. Individuals optimised with the interior point method will have a much lower fitness value than non-optimised individuals, which will cause them to be repeatedly in the top 1% of the population. As further use of the interior point method on these individuals will cause no change, they are excluded from the set which are optimised using the interior point method, and instead replaced with the next most fit individuals.

2.4. Measurement of the I/O Data Sets

Three I/O measurement data sets are used in this paper: one from a simulation using random parameters, and two measured from the Dallas Rangemaster circuit using different BJTs. The two BJTs selected for the optimisation procedure are a silicon BC557 and a germanium OC44. The OC44 is the transistor used in the original Dallas Rangemaster circuit. The BC557 is a general purpose transistor with no history of use within guitar pedals, and was selected solely for the purpose of comparison with the OC44, providing a frame of reference for the modelling of the BJT.

Measured I/O data sets used for the optimisation and sensitivity analysis were experimentally obtained from the Dallas Rangemaster circuit assembled on a breadboard. This was interfaced with MATLAB via a National Instruments ELVIS II DAQ. For the data used in the optimisation, measurements were taken at a sample rate of 100 kHz, with 100 measurements averaged to reduce noise.

A simulated I/O data set was generated to use as a comparison against the measured data. In this case there is no possibility of unmodelled behaviour, thus ensuring that the optimisation can in principle recover the parameters. This then provides a tool to assess the optimisation problem separate from problems that may be encountered with the measurements.

Table 1: *Parameters used in the modelling of the Dallas Rangemaster.*

Parameter	Value		
	Nominal	Measured	Stochastic Sample
R_1	470 kΩ	473.250 kΩ	508.209 kΩ
R_2	68 kΩ	68.596 kΩ	70.262 kΩ
R_3	3.9 kΩ	3.8965 kΩ	3.5416 kΩ
R_4	1 MΩ	0.997 MΩ	0.901 MΩ
VR_1	10 kΩ	9.999 kΩ	10.56 kΩ
C_1	4.7 nF	4.92 nF	4.44 nF
C_2	47 µF	46.95 µF	47.54 µF
C_3	10 nF	11.57 nF	9.31 nF
β_F (BC557)	340	-	-
β_F (OC44)	90	$46 - 175$	123.95
β_R (BC557)	15	-	-
β_R (OC44)	7	$2 - 12$	5.16
I_S	0.1 pA	-	0.06 pA
N	1.6	-	1.22

3. PARAMETER ANALYSIS

3.1. Determination of Parameter Values and Ranges

The first column of Table 1 shows the nominal parameters of the Dallas Rangemaster. Values for the linear parameters were extracted from the schematic. The selected optimisation algorithm allows for constraints to be placed on the range of parameters, which enables the exploitation of the tolerances specified by component manufacturers. Each value of the resistors belongs to the E12 standard which is specified at $\pm 10\%$, suggesting a sensible range with which to constrain the value of each linear component parameter. Due to the idealised component laws used in the design of the physical model, there is a possibility that the model will not capture the full behaviour of the circuit. Because of this, the linear component parameters were constrained to a range of $\pm 20\%$ to allow for compensation of the possible unmodelled behaviour. A more effective method would be to increase the complexity of the component models to capture this behaviour, but the increase in computational complexity is difficult to justify prior to observation of unmodelled behaviour. A uniform distribution was chosen for the linear component parameter set as real distributions depend on manufacturing techniques.

The BC557 values for β_F and β_R are based upon values taken from Linear Technology's LTspiceIV[2]. Datasheets and SPICE models could not be found for the OC44, so the gain parameters were measured from a small set of the BJT. The measurements were performed by applying a base current while measuring the collector current when biased in the forward-active region, and the emitter current when biased in the reverse-active region. This gives values for β_F and β_R using the approximate relations

$$\beta_\mathrm{F} \approx I_\mathrm{C}/I_\mathrm{B}, \quad \beta_\mathrm{R} \approx I_\mathrm{E}/I_\mathrm{B}.$$

Although this technique provides only a coarse approximation, it was only required to inform the range of values to be expected. The nominal values for I_S and N were set to the same value for both BJTs. The saturation current of different BJTs is often in

[2] `www.linear.com/designtools/software/#LTspice`

the same range, and is difficult to measure accurately. The ideality factor can vary widely between BJTs, particularly for vintage transistors as early manufacturing techniques provided less consistency.

Measurements of the OC44 BJT showed the range of β_{F} to be between 46 and 175, and the range of β_{R} to be between 2 and 14. Because of the wide range of these parameters, and the uncertainty of values of the parameters I_{S} and N, the BJT parameters were constrained to $\pm 100\%$ of their nominal value. A uniform distribution was again selected.

The third column of Table 1 shows the values of parameters used in the simulated data set, using BJT parameters based upon the nominal OC44 values. These were stochastically generated using the discussed uniform distributions across the parameter ranges.

3.2. Sensitivity Analysis

Global sensitivity analysis refers to the study of attributing uncertainty in a model's output to uncertainty in a model's parameters and input. The prefix 'global' specifies that the analysis is upon the whole search space as opposed to local operating points. To provide useful analysis for the optimisation, the fitness function is analysed to rank the effects of each parameter, and compare this between each I/O data set. The implemented method, the Morris method [21] generates *trajectories* through the search space using a one-at-a-time strategy i.e. there is a change in only one parameter between neighbouring sample points. An *elementary effect* of a parameter can then be defined as

$$EE_i = \frac{F(\theta_1, ..., \theta_i + \Delta, ..., \theta_k) - F(\theta_1, ..., \theta_k)}{\Delta} \tag{10}$$

where θ_i is the parameter changed by the value Δ for the elementary effect. The number of calculated elementary effects for each parameter is given by the number of trajectories, r. To prevent incorrect analysis of the sensitivity of each parameter, it is essential to select a large enough value of r. The elementary effects are processed to create two sensitivity measures, μ^* and σ expressed by

$$\mu_i^* = \frac{1}{r}\sum_{j=1}^{r}|EE_i^j|, \quad \sigma_i = \left(\frac{1}{r-1}\sum_{j=1}^{r}(EE_i^j - \mu_i)^2\right)^{\frac{1}{2}}. \tag{11}$$

The estimated absolute mean μ^* reflects the overall influence of the parameter on the fitness, and differs from the mean μ by using absolute values of the elementary effects, preventing type II errors which are caused by negative values [22]. The estimated standard deviation σ groups both the nonlinearity of the parameter and the dependence on other parameters relative to the change in the fitness function. Intuitively, this can be understood by considering a change in the value of the elementary effects: the change must either be caused by a nonlinear parameter i.e. the effect changes across the range of parameter values, or by a change in another parameter due to sampling at other locations in the space.

Analysis was performed using SAFE, a MATLAB toolbox for Global Sensitivity Analysis [23]. To help prevent any non-convergent simulations which would create unusable results, the linear component and BJT parameters were restricted to $\pm 10\%$ and $\pm 40\%$ of their nominal value respectively. A value of 300 was selected for r to ensure the search space was sufficiently analysed, although lower values or r also correctly identified the parameters with the largest and least effect on the fitness function. Figure 5 shows the results of the analysis upon the fit of the model to output measurements for the simulated, BC557, and OC44 I/O data sets. It is worth noting the near-linear relationship between μ^* and σ, indicating the parameters that have the largest effect on the fitness function are more nonlinear or heavily influenced by other parameters (i.e. parameters with high μ^* values of also have high σ values).

Parameter	Rank		
	Simulated	BC557	OC44
R_1	4	3	4
R_2	3	2	5
R_3	8	5	8
R_4	10	11	10
VR_1	5	4	6
C_1	6	8	3
C_2	9	9	9
C_3	11	12	11
I_{S}	7	6	7
N	1	1	2
β_F	2	7	1
β_R	12	10	12

Table 2: *Ranking of the circuit parameters by sensitivity value S.*

A metric for ranking parameters by their influence on the optimisation procedure was designed as $S = \sqrt{\mu^{*2} + \sigma^2}$. The rank of each parameter for each data set is shown in Table 2. Several similarities can be seen between the data sets: each fitness function is very sensitive to the parameter N and quite sensitive to both R_1 and R_2. The fitness function's low sensitivity to many of the parameters in combination with the large range of measured sensitivities indicates that the search space will contain many points with a similar fitness value; it is likely that this partly reflects a level of parameter redundancy. As a result, the optimisation may struggle to find physically meaningful parameter values.

3.3. Limitations of the Optimisation

Using the ranking of parameters of the simulated I/O data set it is possible to test the optimisation algorithm's ability to recover the values of the parameters. The performance of the algorithm can be tested against model complexity by optimising on only the N_{p} most sensitive parameters. Figure 6 displays the results of 10 optimisations performed per case for increasing values of N_{p}. The parameter error ϵ_{p} is calculated as

$$\epsilon_{\mathrm{p}} = \frac{1}{N_{\mathrm{p}}}\sum_{i=0}^{N_{\mathrm{p}}}\left(\frac{\hat{\theta}_i - \theta_i^*}{\theta_i^*}\right)^2 \tag{12}$$

where θ^* is the accurate parameter value (shown in the third column of Table 1) and $\hat{\theta}$ is the parameter value after optimisation. As the number of optimised parameters increases, the ability of the algorithm to accurately retrieve the values decreases. For the full model, the error is over 15 orders of magnitude higher than when optimising for just 2 parameters. The transition from accurately recovering parameters to failing to recover them occurs at the inclusion of the 6th parameter, C_1. After this transition the effect of the increasing quantity of parameters is negligible due to the constraints of the parameters. Given the nominal values of the

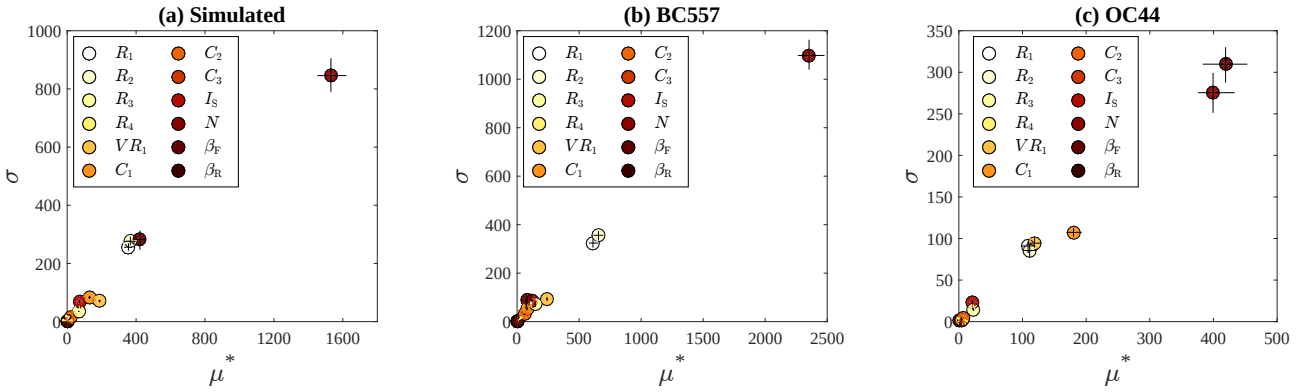


Figure 5: *Results of sensitivity analysis showing Morris measures μ^* and σ for the (a) simulated, (b) BC557, and (c) OC44 data. Confidence bounds are indicated by the black lines placed on each marker.*

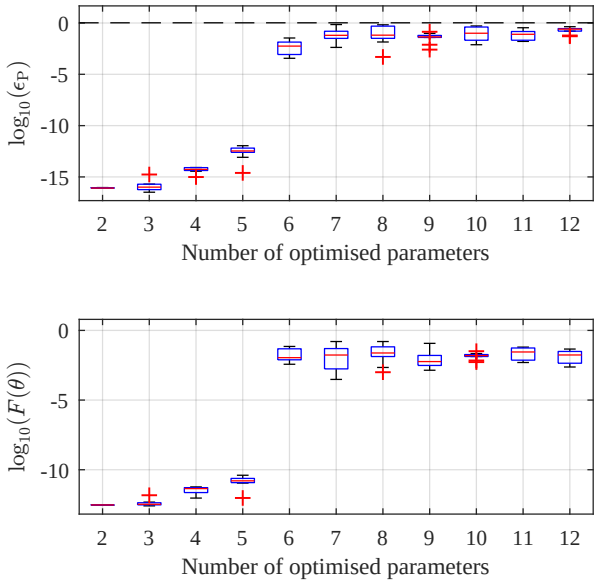


Figure 6: *Box plots displaying the logarithmic scaled error of the parameter set after optimisation on increasing numbers of optimised parameters. Red crosses indicate outliers, dashed line indicates maximum error possible with parameter constraints.*

simulated data set, the maximum parameter error is 1.0433, indicated by the dashed line on Figure 6. Results with parameter error of this order indicate that the parameters have not been recovered. The bottom plot of Figure 6 shows the log scale fitness of the resultant parameter sets from the optimisations. Although there is strong correlation with the parameter error, there is no constraint upon fitness values so this correlation cannot imply the same conclusion as with the parameter error. Fitness values of this order still provide a good fit between model and system over a wide range of operation, which can be demonstrated with the use of validation data (see Section 4).

4. RESULTS AND VALIDATION

4.1. Optimisation on the Full Parameter Set

A map of sinusoidal signals was created as a validation data set. The map covers a range of 30 peak voltages between $[0.1\,\text{V}, ..., 3\,\text{V}]$, and 30 frequencies between $[20\,\text{Hz}, ..., 3000\,\text{Hz}]$, selected logarithmically. This map was processed by both the simulation and circuit with a sample rate of 400 kHz to ensure that the nonlinear solver would converge. Ten measurements from the circuits were averaged to reduce noise.

Figure 7 shows the error of the model against the measurements of the validation data set. Validation error ϵ_V values were calculated using

$$\epsilon_V = 10\log_{10}\left(\sum_{n=1}^{N_s}(V_{out}[n] - \hat{V}_{out}[\hat{\boldsymbol{\theta}}, n])^2\right) \qquad (13)$$

where $\hat{\boldsymbol{\theta}}$ is the optimised parameter set. As the optimisation algorithm is stochastic, it was repeated 15 times for each I/O data set. Representative plots were selected by producing an average data map using each result, and selecting the map that closest matched this map.

The top row of the contour plots in Figure 7 shows how accurately the I/O data sets are modelled using the nominal values from Table 1. The bottom row of plots shows how accurately the I/O data sets are modelled after optimising the model parameters. The results of the simulated I/O data set illustrated in Figure 7 (a) and (d) show the results of an ideal optimisation, where there is no noise from the measurements, and the model is guaranteed to be able to represent the I/O data set. As the contour plot (d) shows, the model using the optimised parameters does accurately model the behaviour of the simulated I/O data set, but not equally well across the presented range of amplitudes and frequencies, with the most significant increase in error at higher amplitudes.

A similar increase in error is observed in Figure 7 (e) and (f), illustrating the model fit to the measured data sets of the BC557 and OC44 BJTs. Generally, the error is larger than that of the simulated data set, but significantly lower than the error of nominal parameter plots (b) and (c). This shows that despite not modelling the circuit behaviour to high accuracy, the fit is significantly better than with nominal parameters.

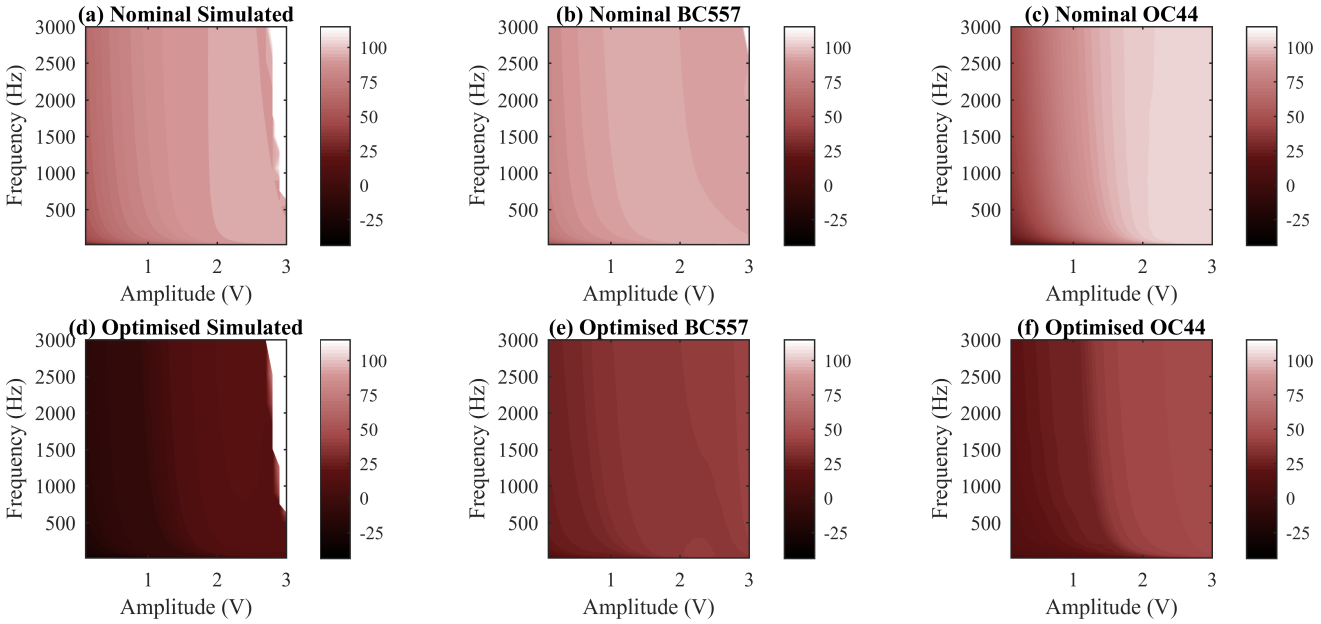


Figure 7: *Contour plots of validation error ϵ_V against amplitude (peak voltage) and frequency. White space in (a) and (d) indicates unsuccessful simulations due to nonconvergence.*

4.2. Optimisation Using Fixed Linear Component Values

To further investigate the error of the model, a second optimisation process was performed with fixed values for the linear component parameters, using the measured values shown in Table 1. This places the focus of the optimisation on the four BJT parameters used in the Ebers-Moll model. As two different BJTs were used in the measurement of the I/O data, the second optimisation aimed to investigate the performance of the Ebers-Moll model's ability to capture the behaviour of both the BC557 and OC44 BJTs. Figure 8 shows an excerpt from the output of the optimised models and the measured data they are attempting to fit. Figure 8(a) shows significantly more error than Figure 8(b), including an obvious phase difference between the model and the measurements. This points to the Ebers-Moll model not accurately capturing the behaviour of the OC44 transistor, indicating unmodelled behaviour that could be caused by e.g. junction capacitances, terminal resistances, parasitic effects etc.

5. CONCLUSION

A preliminary study on fitting a physical model of a guitar effects pedal to measured I/O data using a brute-force parameter optimisation approach has been presented. In Section 3.3 true recovery of the parameters from simulated I/O data was shown to work only for reduced sets of parameters. The results of the sensitivity analysis in Section 3.2 indicate that this could be (at least partly) due to the objective function's insensitivity to some of the parameters. It is therefore of particular interest in future research to consider alternative objective functions, alongside exploring different, even more exhaustive methods for searching the parameter space. Another possible route towards improved results regarding recovery of parameters is to consider other ways of driving the circuit and collecting the output data; potentially this includes attempts to link the physical model parameters directly to those typically used in black-box representations, (e.g. Volterra series kernel coefficients).

Although recovering the actual physical parameters proved very challenging, the results presented in Section 4 show that the accuracy of the physical model can be significantly improved using optimised instead of nominal parameters. While changes in the measurement method, objective function and optimisation (as mentioned above) can potentially further reduce the remaining errors, the results plotted in Figure 8 imply that attention must also be given to the physical model formulation. More specifically, the significantly better fit of the model to the BC557 transistor I/O data is a strong indication that the Ebers-Moll model is too simplistic to capture the behaviour of the OC44 transistor. Hence additional transistor modelling elements are required to accurately simulate the Rangemaster and other audio effects pedals and amplifiers featuring germanium BJTs. To address this, a more rigorous study of such vintage components is required, including high-accuracy measurement and subsequent comparison with more sophisticated formulations such as the Gummel-Poon model [24].

A key question to address in the longer term is to what extent models of more complex, larger circuits could be calibrated through optimisation on I/O measurement data; for systems with many more parameters than the Rangemaster it is likely that order reduction will play an even greater role, which is potentially aided by parameter screening techniques such as that employed in the present study.

6. REFERENCES

[1] A. Farina, "Simultaneous measurement of impulse response and distortion with a swept-sine technique," in *Audio En-*

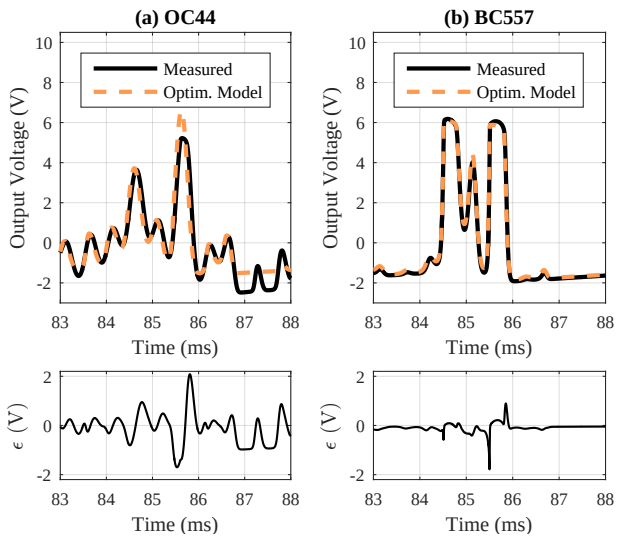


Figure 8: *Comparison of simulation output to measurement output of the Dallas Rangemaster for both the (a) OC44 and (b) BC557 circuits. The error is shown beneath each plot.*

gineering Society Convention 108. 2000, Audio Engineering Society.

[2] A. Farina, A. Bellini, and E. Armelloni, "Non-linear convolution: A new approach for the auralization of distorting systems," in *Audio Engineering Society Convention 110*. 2001, Audio Engineering Society.

[3] A. Novak, L. Simon, P. Lotton, and J. Gilbert, "Chebyshev model and synchronized swept sine method in nonlinear audio effect modeling," in *Proc. 13th Int. Conference on Digital Audio Effects*, 2010.

[4] A. Novak, L. Simon, F. Kadlec, and P. Lotton, "Nonlinear System Identification Using Exponential Swept-Sine Signal," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 8, pp. 2220–2229, Aug. 2010.

[5] A. Novak, P. Lotton, and L. Simon, "Synchronized Swept-Sine: Theory, Application, and Implementation," *Journal of the Audio Engineering Society*, vol. 63, no. 10, pp. 786–798, Nov. 2015.

[6] L. Tronchin and V. L. Coli, "Further Investigations in the Emulation of Nonlinear Systems with Volterra Series," *Journal of the Audio Engineering Society*, vol. 63, no. 9, pp. 671–683, Oct. 2015.

[7] F. Eichas, S. Möller, and U. Zölzer, "Block-oriented modeling of distortion audio effects using iterative minimization," in *Proceedings of the 18th International Conference on Digital Audio Effects*, Trondheim, Norway, Dec. 2015.

[8] K. J. Werner, J. O. Smith III, and J. S. Abel, "Wave digital filter adaptors for arbitrary topologies and multiport linear elements," in *Proceedings of the 18th International Conference on Digital Audio Effects*, Trondheim, Norway, Dec. 2015.

[9] M. Holters and U. Zölzer, "A generalized method for the derivation of non-linear state-space models from circuit schematics," in *23rd European Signal Processing Conference (EUSIPCO), 2015*, 2015.

[10] M. Holters and U. Zölzer, "Physical Modelling of a Wah-Wah Pedal as a Case Study for Application of the Nodal DK Method to Circuits with Variable Parts," in *Proc. of the 14th Internation Conference on Digital Audio Effects*, Paris, France, Sept. 2011.

[11] F. Eichas, M. Fink, M. Holters, and U. Zölzer, "Physical Modeling of the MXR Phase 90 Guitar Effect Pedal," in *Proc. of the 17 th Int. Conference on Digital Audio Effects*, Erlangen, Germany, Sept. 2014.

[12] D. T. Yeh, J. S. Abel, and J. O. Smith, "Automated Physical Modeling of Nonlinear Audio Circuits For Real-Time Audio Effects; Part I: Theoretical Development," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 728–737, May 2010.

[13] J. Macak, J. Schimmel, and M. Holters, "Simulation of fender type guitar preamp using approximation and state-space model," in *Proceedings of the 12th International Conference on Digital Audio Effects, York, UK*, 2012.

[14] K. Dempwolf and U. Zölzer, "Discrete State-Space Model of the Fuzz-Face," in *Proceedings of Forum Acusticum*, Aalborg, Denmark, June 2011, European Acoustics Association.

[15] J. J. Ebers and J. L. Moll, "Large-signal behavior of junction transistors," *Proceedings of the IRE*, vol. 42, no. 12, pp. 1761–1772, 1954.

[16] M. Schroeder, "Synthesis of low-peak-factor signals and binary sequences with low autocorrelation (Corresp.)," *Information Theory, IEEE transactions on*, vol. 16, no. 1, pp. 85–89, 1970.

[17] R. C. Jaeger, "Small-signal models for Bipolar Junction Transistors," in *Microelectronic circuit design*. McGraw-Hill, a business unit of The McGraw-Hill Companies, Inc, New York, NY, fifth edition edition, 2015.

[18] R. L. Haupt and S. E. Haupt, *Practical genetic algorithms*, John Wiley, Hoboken, N.J, 2nd ed edition, 2004.

[19] B. Holmes and M. van Walstijn, "Improving the robustness of the iterative solver in state-space modelling of guitar distortion circuitry," in *Proceedings of the 18th International Conference on Digital Audio Effects*, Trondheim, Norway, Dec. 2015.

[20] S. P. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University Press, Cambridge, UK ; New York, 2004.

[21] M. D. Morris, "Factorial Sampling Plans for Preliminary Computational Experiments," *Technometrics*, vol. 33, no. 2, pp. 161–174, May 1991.

[22] F. Campolongo, J. Cariboni, and A. Saltelli, "An effective screening design for sensitivity analysis of large models," *Environmental Modelling & Software*, vol. 22, no. 10, pp. 1509–1518, Oct. 2007.

[23] F. Pianosi, F. Sarrazin, and T. Wagener, "A Matlab toolbox for Global Sensitivity Analysis," *Environmental Modelling & Software*, vol. 70, pp. 80–85, Aug. 2015.

[24] H. K. Gummel and H. C. Poon, "An Integral Charge Control Model of Bipolar Transistors," *Bell System Technical Journal*, vol. 49, no. 5, pp. 827–852, 1970.

CIRCUIT SIMULATION WITH INDUCTORS AND TRANSFORMERS BASED ON THE JILES-ATHERTON MODEL OF MAGNETIZATION

Martin Holters, Udo Zölzer

Department for Signal Processing and Communications,
Helmut Schmidt University
Hamburg, Germany
martin.holters|udo.zoelzer@hsu-hh.de

ABSTRACT

The sound of a vacuum tube guitar amplifier may be significantly influenced by the non-linear behavior of its output transformer, which therefore should also be considered in digital simulations. In this work, we develop a model for inductors and transformers with the magnetization following the model of Jiles and Atherton. For this purpose, the original magnetization model is rewritten to a differential equation with respect to time which can then easily be integrated into a previously developed circuit simulation framework. The model thus derived is then exercised in the simulation of three simple circuits where it shows the expected behavior.

1. INTRODUCTION

Non-linear behavior of its output transformer may have a significant influence on the sound produced by a vacuum tube guitar amplifier. It is therefore desirable to include this non-linearity in digital simulations. A commonly used method to do so is the application of the gyrator-capacitor model, which maps magnetic quantities to electric ones [1, and references therein]. This allows complex magnetic topologies to be modeled by mapping them to corresponding electric circuits. Non-linear effects are then represented by non-linear resistors and capacitors. While the Jiles-Atherton model [2, 3] provides a good model of the non-linear magnetization effects, it is often deemed too complex and replaced by simpler heuristics.

In this work, we propose an inductor/transformer model that does apply the Jiles-Atherton magnetization model, but forgoes the gyrator-capacitor approach. It will hence be restricted to simple topologies where the magnetic field of all windings is completely contained by one and the same core and is furthermore uniform within the core. This holds for toroidal inductors/transformers where the core is thin compared to its diameter. We hope, however, that it sufficiently approximates other topologies found in guitar equipment. In [4] and [5], specific circuits (a series LC oscillator and a tube guitar amplifier) were simulated based on the Jiles-Atherton model in a similar fashion and very good agreement to measurements could be observed.

While in [4] and [5] the circuits were approached holistically using ad-hoc methods, this paper instead aims to develop an inductor/transformer model that can be used in various circuits using a systematic analysis technique. In particular, the model will be developed such that it is usable with the methodology developed in [6].

2. PREVIOUS WORK

In this section, the Jiles-Atherton model of magnetization to be used in the following will be briefly introduced and the most important aspects of the employed circuit modeling framework will be repeated, while a detailed discussion is left to [2] and [6], respectively.

2.1. Jiles-Atherton Model

The Jiles-Atherton model relates magnetic field H and magnetization M via a differential equation. It was originally developed in [2], but the slightly modified form of the results given in [3] will be used as a basis here. The magnetization is decomposed as

$$M = M_{\text{rev}} + M_{\text{irr}}, \tag{1}$$

where M_{rev} and M_{irr} denote the reversible and irreversible magnetization component, respectively, both of which depend on the anhysteretic magnetization M_{an}.

In particular, the reversible magnetization is given by

$$M_{\text{rev}} = c \cdot (M_{\text{an}} - M_{\text{irr}}), \tag{2}$$

where c is the ratio of normal and anhysteretic initial susceptibilities, while M_{irr} is implicitly defined with the differential equation

$$\frac{dM}{dH} = (1-c) \cdot \frac{M_{\text{an}} - M_{\text{irr}}}{\delta k - \alpha(M_{\text{an}} - M_{\text{irr}})} + c \cdot \frac{dM_{\text{an}}}{dH}, \tag{3}$$

where

$$\delta = \begin{cases} 1 & \text{if } H \text{ is increasing} \\ -1 & \text{if } H \text{ is decreasing} \end{cases} \tag{4}$$

denotes the direction of change of H. Furthermore, k, and α are parameters, where k is a measure of the width of the hysteresis loop, and α is a mean field parameter, representing inter-domain coupling [4]. The first term in equation (3) describing the irreversible magnetization process has to be set to zero if $M - M_{\text{an}}$ and δ have opposite signs.

The anhysteretic magnetization M_{an} is a function of the effective field $H_{\text{e}} = H + \alpha M$ and is responsible for the saturating behavior. While various mathematical functions could serve as a model, in [2] the Langevin function

$$L(x) = \coth(x) - \frac{1}{x} \tag{5}$$

with the continuous extension $L(0) = 0$ (see Figure 1) has been employed as

$$M_{\text{an}} = M_{\text{s}} \cdot L\left(\frac{H + \alpha M}{a}\right) \tag{6}$$

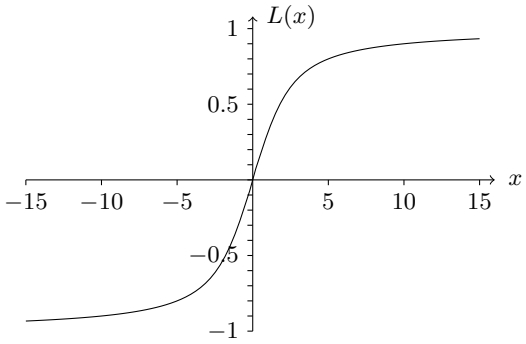


Figure 1: *The Langevin function* $L(x)$.

with good success, where M_s is the saturation magnetization and a characterizes the shape of the anhysteretic magnetization.

In [4], further details were modified. First, from equations (1) and (2), one can find that

$$M_\mathrm{an} - M_\mathrm{irr} = \frac{1}{1-c}(M_\mathrm{an} - M), \tag{7}$$

which can be substituted in equation (3). Second, the first term of equation (3) is explicitly multiplied with

$$\delta_\mathrm{M} = \begin{cases} 1 & \text{if } \delta \text{ and } M_\mathrm{an} - M \text{ have the same sign} \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

so that

$$\frac{dM}{dH} = \frac{(1-c)\cdot\delta_M\cdot(M_\mathrm{an}-M)}{(1-c)\cdot\delta\cdot k - \alpha(M_\mathrm{an}-M)} + c\cdot\frac{dM_\mathrm{an}}{dH} \tag{9}$$

will be the magnetization model used in this work.

2.2. Circuit Modeling Framework

The circuit modeling framework of [6] allows for a very general specification of circuit element behavior. While the reader is referred to [6] for details on how a non-linear state-space model is derived from these specifications together with a circuit topology, the form an element model needs to be in shall be briefly repeated here.

Circuit elements need to enforce a relationship between voltages $\boldsymbol{v}$ and currents $\boldsymbol{i}$, measured between pairs of their terminals, and (if needed) states $\boldsymbol{x}$ and their derivatives $\dot{\boldsymbol{x}}$ with respect to time. There is some freedom in the choice of terminal pairs to define $\boldsymbol{v}$ and $\boldsymbol{i}$. For the transformer model, the obvious choice is to pair those terminals connected to the same winding. Likewise, the states $\boldsymbol{x}$ can be defined as is most suitable for the model. For e.g. a capacitor, either having voltage or charge as state would both work equally well. As will be detailed in Sec. 3, for the non-linear inductor/transformer model, the state vector will comprise magnetic field and magnetization.

To facilitate faster simulations, linear and non-linear equations are strictly separated in [6]. For the element description, all non-linear equations have to be formulated in terms of an auxiliary vector $\boldsymbol{q}$ which in turn is coupled with $\boldsymbol{v}$, $\boldsymbol{i}$, $\boldsymbol{x}$, and $\dot{\boldsymbol{x}}$ by linear equations. Once more, the element model has all freedom in choosing a suitable $\boldsymbol{q}$. In the model to be developed, it will hold magnetic field and magnetization along with their derivatives.

All linear equations are collected in one system

$$\boldsymbol{M}_\mathrm{v}\boldsymbol{v} + \boldsymbol{M}_\mathrm{i}\boldsymbol{i} + \boldsymbol{M}_\mathrm{x}\boldsymbol{x} + \boldsymbol{M}_\dot{\mathrm{x}}\dot{\boldsymbol{x}} + \boldsymbol{M}_\mathrm{q}\boldsymbol{q} = \boldsymbol{u}, \tag{10}$$

where the matrices $\boldsymbol{M}_\mathrm{v}$, $\boldsymbol{M}_\mathrm{i}$, $\boldsymbol{M}_\mathrm{x}$, $\boldsymbol{M}_\dot{\mathrm{x}}$, and $\boldsymbol{M}_\mathrm{q}$, and the vector $\boldsymbol{u}$ have to be provided by the element model to obtain the desired behavior. These, together with an implicit non-linear equation

$$\boldsymbol{f}(\boldsymbol{q}) = \boldsymbol{0}, \tag{11}$$

constitute the element model. The matrices and the function $\boldsymbol{f}(\boldsymbol{q})$ for the non-linear inductor/transformer will be derived in Sec. 3.

3. MODEL DEVELOPMENT

In this section, the magnetization model of equation (9) will be rewritten and the magnetic quantities will be related to electric quantities to yield an element model suitable for the circuit modeling technique of [6]. In particular, the differential equation of equation (9) has to be rewritten such that only derivatives with respect to time occur, as the method of [6] cannot handle other derivatives.

In the first step, $\frac{dM_\mathrm{an}}{dH}$ shall be replaced by applying the chain rule of differentiation, giving

$$\frac{dM_\mathrm{an}}{dH} = \frac{M_\mathrm{s}}{a}\left(1+\alpha\frac{dM}{dH}\right)L'\left(\frac{H+\alpha M}{a}\right), \tag{12}$$

where

$$L'(x) = \frac{d}{dx}L(x) = \frac{1}{x^2} - \coth^2(x) + 1 \tag{13}$$

with the continuous extension $L'(0) = \frac{1}{3}$ is the first derivative of the Langevin function. Substituting equation (12) in equation (9) yields

$$\frac{dM}{dH} = \frac{(1-c)\cdot\delta_\mathrm{M}\cdot(M_\mathrm{an}-M)}{(1-c)\cdot\delta\cdot k - \alpha\,(M_\mathrm{an}-M)} + c\cdot\frac{M_\mathrm{s}}{a}\left(1+\alpha\frac{dM}{dH}\right)L'\left(\frac{H+\alpha M}{a}\right). \tag{14}$$

Observing that

$$\frac{dM}{dt} = \frac{dM}{dH}\frac{dH}{dt}, \tag{15}$$

the differential equation can thus be transformed into one where all derivatives are with respect to time by multiplying with $\frac{dH}{dt}$, giving

$$\frac{dM}{dt} = \frac{(1-c)\cdot\delta_\mathrm{M}\cdot(M_\mathrm{an}-M)}{(1-c)\cdot\delta\cdot k - \alpha\,(M_\mathrm{an}-M)}\frac{dH}{dt} + c\cdot\frac{M_\mathrm{s}}{a}\left(\frac{dH}{dt}+\alpha\frac{dM}{dt}\right)L'\left(\frac{H+\alpha M}{a}\right). \tag{16}$$

Note also that $\delta = \operatorname{sign}\left(\frac{dH}{dt}\right)$.

In equation (16), the derivatives of H and M are required, which therefore constitute the state vector

$$\boldsymbol{x} = \begin{pmatrix} H \\ M \end{pmatrix} \tag{17}$$

of the model being developed. And while M is needed as such, H only appears as $\frac{H+\alpha M}{a}$, so the auxiliary vector linking linear and

non-linear equations can be chosen as

$$\boldsymbol{q} = \begin{pmatrix} \frac{H+\alpha M}{a} \\ M \\ \frac{dH}{dt} \\ \frac{dM}{dt} \end{pmatrix} = \begin{pmatrix} \frac{x_1+\alpha x_2}{a} \\ x_2 \\ \dot{x}_1 \\ \dot{x}_2 \end{pmatrix}. \tag{18}$$

By subtracting $\frac{dM}{dt} = q_4$, equation (16) can then be rewritten in the required form as

$$f(\boldsymbol{q}) = \frac{(1-c)\cdot\delta_{\mathrm{M}}\cdot(M_{\mathrm{an}}-q_2)}{(1-c)\cdot\delta\cdot k-\alpha\,(M_{\mathrm{an}}-q_2)}q_3 + c\cdot\frac{M_{\mathrm{s}}}{a}\,(q_3+\alpha q_4)\,L'\,(q_1) - q_4 = 0 \tag{19}$$

with

$$M_{\mathrm{an}} = M_{\mathrm{s}}\cdot L(q_1) \tag{20}$$

$$\delta = \operatorname{sign}(q_3) \tag{21}$$

$$\delta_{\mathrm{M}} = \begin{cases} 1 & \text{if } \delta \text{ and } M_{\mathrm{an}}-q_2 \text{ have the same sign} \\ 0 & \text{otherwise,} \end{cases} \tag{22}$$

which now only depends on constant parameters and entries of the auxiliary vector $\boldsymbol{q}$.

Next, the magnetic quantities have to be related to the electric quantities. For simplicity, we only consider a toroidal inductor which is thin compared to its diameter so that the magnetic field is uniform inside the core. Assuming K individual wires having $n_1,\ldots,n_K$ turns around the core and carrying currents $i_1,\ldots,i_K$, the magnetic field is given as

$$H = \frac{1}{\pi D}\sum_{k=1}^{K} n_k i_k \tag{23}$$

by Ampère's law, where D is the torus' diameter. From the flux $\Phi = \mu_0 A\cdot(H+M)$, where A is the cross-sectional area of the core, the voltages $v_1,\ldots,v_K$ are given by Faraday's law as

$$v_k = n_k\cdot\frac{d}{dt}\Phi = \mu_0 A n_k\cdot\left(\frac{dH}{dt}+\frac{dM}{dt}\right). \tag{24}$$

Letting $\boldsymbol{v} = \begin{pmatrix} v_1 & \ldots & v_K \end{pmatrix}^T$ and $\boldsymbol{i} = \begin{pmatrix} i_1 & \ldots & i_K \end{pmatrix}^T$ and collecting equations (24), (23) and (18) in a single equation system of the form of equation (10), the model matrices can thus be determined as

$$\boldsymbol{M}_{\mathrm{v}} = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{pmatrix}, \quad \boldsymbol{M}_{\mathrm{i}} = \begin{pmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \\ n_1 & \cdots & n_K \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \end{pmatrix}, \tag{25}$$

$$\boldsymbol{M}_{\mathrm{x}} = \begin{pmatrix} 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ -\pi D & 0 \\ -\frac{1}{a} & -\frac{\alpha}{a} \\ 0 & -1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad \boldsymbol{M}_{\dot{\mathrm{x}}} = \begin{pmatrix} -\mu_1 A n_1 & -\mu_1 A n_1 \\ \vdots & \vdots \\ -\mu_0 A n_K & -\mu_0 A n_K \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{26}$$

$$\boldsymbol{M}_q = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \boldsymbol{u} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \tag{27}$$

(28)

where $\boldsymbol{M}_{\mathrm{v}}$ and $\boldsymbol{M}_{\mathrm{i}}$ have K columns and all matrices have $K+5$ rows. The upper K rows correspond to equation (24) for $k = 1,\ldots,K$, the row below to equation (23) multiplied with πD, and the lower four rows to equation (18).

4. NUMERICAL CONSIDERATIONS

For a successful implementation of the developed model, some numerical considerations are in order that will be detailed in the following.

First, the definition of the Langevin function and its derivatives as stated above are very susceptible to rounding errors for values close to zero. For numerical evaluation, they are better replaced with a truncated Taylor series around zero as

$$L(x) = \begin{cases} \coth(x)-\frac{1}{x} & \text{for } |x| > 10^{-4} \\ \frac{x}{3} & \text{otherwise,} \end{cases} \tag{29}$$

$$L'(x) = \begin{cases} \frac{1}{x^2}-\coth^2(x)+1 & \text{for } |x| > 10^{-4} \\ \frac{1}{3} & \text{otherwise,} \end{cases} \tag{30}$$

$$L''(x) = \begin{cases} 2\coth(x)\cdot\left(\coth^2(x)-1\right)-\frac{2}{x^3} & \text{for } |x| > 10^{-3} \\ -\frac{2}{15}x & \text{otherwise,} \end{cases} \tag{31}$$

where the second derivative $L''(x) = \frac{d^2}{dx^2}L(x)$ will be used momentarily. Furthermore, to avoid a division-by-zero problem when $\boldsymbol{q} = \boldsymbol{0}$, we let $\delta = 1$ if $q_3 = \frac{dH}{dt} = 0$, noting that the only term it occurs in is multiplied by q_3 anyway.

For numerical solution of systems containing equation (19), the Jacobian

$$\boldsymbol{J_f}(\boldsymbol{q}) = \begin{pmatrix} \frac{df(\boldsymbol{q})}{dq_1} & \frac{df(\boldsymbol{q})}{dq_2} & \frac{df(\boldsymbol{q})}{dq_3} & \frac{df(\boldsymbol{q})}{dq_4} \end{pmatrix} \tag{32}$$

of $f(\boldsymbol{q})$ will typically be needed, which can be determined as

$$\frac{df(\boldsymbol{q})}{dq_1} = \frac{(1-c)^2\delta_M\delta k M_{\mathrm{s}} L'(q_1)}{\left((1-c)\delta k-\alpha(M_{\mathrm{an}}-q_2)\right)^2}q_3 + \frac{cM_{\mathrm{s}}}{a}(q_3+\alpha q_4)L''(q_1) \tag{33}$$

```
1  function transformer(::Type{Val{:JA}}; D=2.4e-2, A=4.54e-5, ns=[],
2                       a=14.1, α=5e-5, c=0.55, k=17.8, Ms=2.75e5)
3      const μ0 = 1.2566370614e-6
4      nonlinear_eq = quote
5          coth_q1 = coth(q[1])
6          a_q1 = abs(q[1])
7          L_q1 = a_q1 < 1e-4 ? q[1]/3 : coth_q1 - 1/q[1]
8          Ld_q1 = a_q1 < 1e-4 ? 1/3 : 1/q[1]^2 - coth_q1^2 + 1
9          Ld2_q1 = a_q1 < 1e-3 ? -2/15*q[1] : 2*coth_q1*(coth_q1^2 - 1) - 2/q[1]^3
10         δ = q[3] > 0 ? 1.0 : -1.0
11
12         Man = $(Ms)*L_q1
13         δM = sign(q[3]) == sign(Man - q[2]) ? 1.0 : 0.0
14
15         den = δ*$(k*(1-c))-$(α)*(Man-q[2])
16         res[1] = $(1e-4/Ms) * ($(1-c) * δM*(Man-q[2])/den * q[3]
17                               + $(c*Ms/a)*(q[3]+$(α)*q[4])*Ld_q1 - q[4])
18         J[1,1] = $(1e-4/Ms) * ($((1-c)^2*k*Ms) * δM*Ld_q1*δ/den^2 * q[3]
19                               + $(c*Ms/a)*(q[3]+$(α)*q[4])*Ld2_q1)
20         J[1,2] = $(1e-4/Ms) * -$((1-c)^2*k) * δM*δ/den^2 * q[3]
21         J[1,3] = $(1e-4/Ms) * ($(1-c) * δM*(Man-q[2])/den + $(c*Ms/a)*Ld_q1)
22         J[1,4] = $(1e-4/Ms) * ($(c * Ms/a * α)*Ld_q1 - 1)
23     end
24     Element(mv=[speye(length(ns)); spzeros(5, length(ns))],
25             mi=[spzeros(length(ns), length(ns)); ns.'; spzeros(4, length(ns))],
26             mx=[spzeros(length(ns), 2); -π*D 0; -1/a -α/a; 0 -1; 0 0; 0 0],
27             mxd=[-μ0*A*ns -μ0*ns*A; 0 0; 0 0; 0 0; -1 0; 0 -1],
28             mq=[zeros(length(ns)+1,4); eye(4)], nonlinear_eq = nonlinear_eq)
29 end
```

Figure 2: *Julia/ACME source code of the developed inductor/transformer model.*

$$\frac{df(\boldsymbol{q})}{dq_2} = -\frac{(1-c)^2\delta_M\delta k}{\big((1-c)\delta k - \alpha(M_{\text{an}} - q_2)\big)^2} q_3 \tag{34}$$

$$\frac{df(\boldsymbol{q})}{dq_3} = \frac{(1-c)\delta_M \cdot (M_{\text{an}} - q_2)}{(1-c)\delta k - \alpha\,(M_{\text{an}} - q_2)} + \frac{cM_{\text{s}}}{a} L'(q_1) \tag{35}$$

$$\frac{df(\boldsymbol{q})}{dq_4} = \frac{cM_{\text{s}}\alpha}{a} L'(q_1) - 1, \tag{36}$$

where the discontinuities in δ and δ_M have been ignored.

Finally, note that q_4 can assume rather large values, therefore, small relative errors may still yield values for the residual $f(\boldsymbol{q})$ several orders of magnitude larger than, say, small relative errors in a diode current if that is used to define a diode's non-linear equation. If several different non-linear components are used in the same circuit, it would therefore be necessary to have different convergence criteria for the individual residuals of the combined non-linear equation. Alternatively, one can scale $f(\boldsymbol{q})$ (and hence $\boldsymbol{J_f}(\boldsymbol{q})$), which was done in the implementation used here to evaluate the derived model.

5. IMPLEMENTATION AND RESULTS

The proposed model has been implemented as part of the ACME[1] project, a circuit simulation package for the Julia programming language[2]. Julia is a relatively young programming language, still evolving rapidly. It is intended to be used in technical computing, providing a high level of abstraction with convenient syntax combined with the ability to generate highly efficient code [7, 8]. ACME is an implementation of the method of [6], providing a circuit simulation framework and a testbed for further developments, especially additional and improved element models and automated optimization techniques for faster simulations. ACME is implemented in Julia mainly due to two language features, namely good support for matrix operations and linear algebra (similar to e.g. MATLAB) and the ability to work with Julia expressions in the language itself, combined with the efficiency of the code being generated. Good matrix support is important as [6] makes heavy use of matrices. Being able to manipulate Julia expressions from within the language allows to generate a single function representing the non-linear equation of a whole circuit with multiple nonlinear elements on the fly.

The code corresponding to the proposed model given by equations (19) to (22) and (25) to (36) is shown in Figure 2. While a general introduction into Julia and ACME are beyond the scope of this paper, the main features of Figure 2 shall be explained in the following. The first two lines start the definition of the function `transformer` (ended in line 29), where the parameters of the Jiles-Atherton model and the core geometry occur as function arguments with default values taken from [4]. The number of turns of the individual windings are passed as a vector to the argument `ns`, defaulting to the empty vector, i.e. a transformer without any windings. In lines 4 to 23, the non-linear equation is defined as a Julia expression: The code inside the `quote ... end` block is parsed but not executed, rather, its abstract syntax tree is stored in `nonlinear_eq`. While parsing, values can be spliced in using the `$(...)` syntax, inserting the model parameters as constants in the expression. To evaluate the non-linear equation, first the Langevin function and its derivatives are computed in lines 5 to 9, then δ, M_{an} and δ_{M} are determined in lines 10 to 13. Before computing the residual and the Jacobian in lines 16 to 22, the common denominator `den` occurring in them is pre-computed in line 15. Here, a scaling as mentioned above with $10^{-4}/M_{\text{s}}$ is ap-

[1] `https://github.com/HSU-ANT/ACME.jl`
[2] `http://julialang.org/`

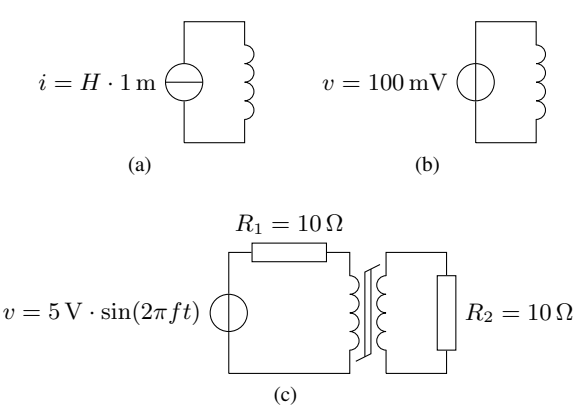


Figure 3: *Simulated circuits: (a) inductor driven by a current source ramped up and down to obtain hysteresis loops; (b) inductor driven by constant voltage; (c) simplified output transformer stage driven by sinusoidal voltage.*

Table 1: *Model parameters used in the simulations, taken from [2] and [4]. Note that the definition of c is different in [2]; there* $c = 0.2$*, which is converted to* $0.2/(1 + 0.2) \approx 0.17$ *here.*

parameter	value in [2]	value in [4]
M_s	1.6×10^6 A/m	2.75×10^5 A/m
a	1.1×10^3 A/m	1.41×10^1 A/m
α	1.6×10^{-3}	5.00×10^{-5}
k	4.0×10^2	1.78×10^1
c	1.7×10^{-1}	5.50×10^{-1}
D	—	2.40×10^{-1} m
A	—	4.54×10^{-5} m^2
n	—	230

plied. Note that the residual `res` and the Jacobian `J` are assumed to be a vector and a matrix, respectively, so the assignment of the results is to specific entries of them. The ACME framework will take care that the resulting expression is used in a context where `res` and `J` of appropriate size as well as the auxiliary vector `q` are defined. Finally, in lines 24 to 28, the non-linear equation is combined with the matrices of equations (25) to (27) to yield the desired model description. The syntax for matrix definition is very similar to MATLAB; `speye` and `spzeros` create sparse identity and all-zero matrices, respectively.

To verify model and implementation, the three simple circuits shown in Figure 3 are simulated using ACME version 0.1.1. The circuit of Figure 3a is used to obtain magnetic hysteresis loops, while the circuit of Figure 3b shall exemplify the behavioral difference between a linear and a non-linear inductor. Finally, the circuit of Figure 3c is a simplified model output transformer stage, demonstrating the model in a likely application.

For the circuit in Figure 3a, the core parameters from [2] (see Table 1) are used and the geometry chosen as $D = \frac{1}{\pi}\,\mathrm{m}, A = 1\,\mathrm{m}^2, n = 1$, so that current and magnetic field have the same numeric value. The sourced current is ramped up and down to increasing values and the magnetization is recorded. As can be seen in Figure 4, the resulting magnetic hysteresis loops are in good agreement with the results of [2].

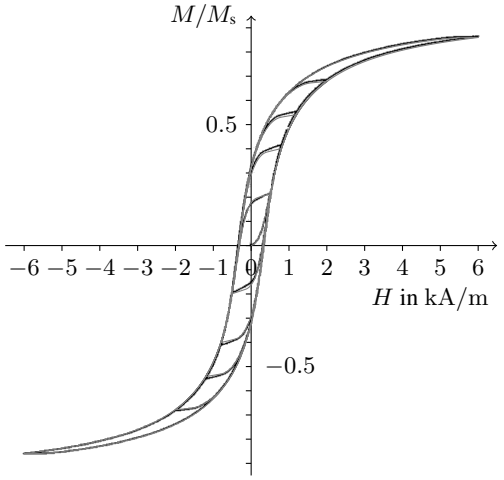


Figure 4: *Comparison of magnetic hysteresis loops from [2, Fig. 12] (black) with the ones obtained with the implemented model of Figure 2 (overlaid in gray) using the same parameters (given in Table 1).*

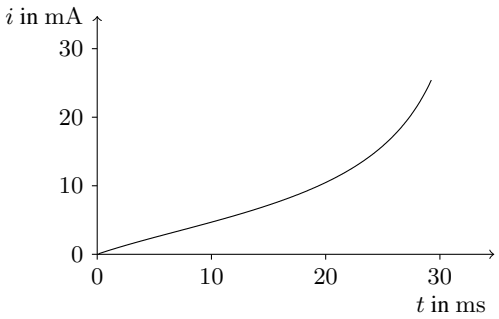


Figure 5: *Inductor current* i *over time* t *for constant-voltage excitation with* $100\,\mathrm{mV}$ *using the inductor parameters from [4] given in Table 1.*

In the second experiment, corresponding to the circuit of Figure 3b, the parameters determined for a real inductor in [4] are employed. A constant voltage of $100\,\mathrm{mV}$ is applied and the current is recorded. For a linear inductor, the current would increase linearly, being proportional to the integrated voltage. For a non-linear inductor where the core saturates, the effective inductance is decreased, leading to a faster current increase. This is exactly the behavior that can be observed in Figure 5: At the beginning, the current grows linearly, matching the linear behavior. For larger currents, however, the core reaches saturation, and accordingly the current grows ever faster.

Finally, a transformer circuit vaguely reminiscent of a tube amplifier's output transformer stage as shown in Figure 3c is examined. Here $R_1 = R_2 = 10\,\Omega$ model the tube stage output impedance and load impedance, respectively. Core parameters and geometry are as in [4] again, with $n_1 = 230$ turns on the primary side and $n_2 = 23$ turns on the secondary side. Note that the inductor would be grossly undersized for a real amplifier. The code setting up the corresponding model in ACME is shown in Figure 6. In lines 1 to 4, the transformer model is instantiated by calling the

```
1  l = transformer(Val{:JA}, Ms=2.75e5, a=14.1,
2                  α=5e-5, k=17.8, c=0.55,
3                  D=2.4e-2, A=4.54e-5,
4                  ns=[230, 23])
5  vsrc = voltagesource()
6  r1 = resistor(10)
7  r2 = resistor(10)
8  vprobe = voltageprobe()
9  circ = Circuit()
10 connect!(circ, vsrc[:+], r1[1])
11 connect!(circ, r1[2], l[1])
12 connect!(circ, l[2], vsrc[:-])
13 connect!(circ, r2[1], l[3], vprobe[:+])
14 connect!(circ, r2[2], l[4], vprobe[:-])
15 fs=44100
16 model = DiscreteModel(circ, 1/fs)
17 u=5*sin(2pi*1000/fs*(0:fs-1)).'
18 y=run!(model, u)
```

Figure 6: *Julia/ACME code for the circuit of Figure 3c.*

function of Figure 2. Similarly, in lines 5 to 8, the driving voltage source, the two resistors, and a voltage probe to obtain the output voltage are created. The `Circuit` object created in line 9 holds the circuit description and is populated in lines 10 to 14 by specifying how the circuit elements are connected. For example, the positive terminal of the voltage source is connected to terminal 1 of resistor R_1. The circuit description is converted to a runnable model for a sampling rate of $44.1\,\mathrm{kHz}$ in lines 15 and 16 which is then executed in line 18. The circuit is driven with a sinusoidal voltage with an amplitude of $5\,\mathrm{V}$, set up in line 17. For high frequencies f, as shown in Figure 7a for $f = 1\,\mathrm{kHz}$, the circuit almost perfectly achieves the voltage scaling by $n_2/n_1 = 1/10$. On the contrary, for lower frequencies like $f = 100\,\mathrm{Hz}$ shown in Figure 7b, the saturation of the core yields a very non-linear behavior, similar in shape to e.g. the results in [1].

6. CONCLUSIONS

We have presented an inductor/transformer model based on the Jiles-Atherton model of magnetization. It foregoes the commonly found gyrator-capacitor translation step to map magnetic to electric quantities. Instead, it exploits the possibilities of the employed framework of [6] to have the magnetic field and the magnetization as state variables. An implementation as part of the ACME package allows flexible use of the developed model for various circuits. Simulations conducted with simple circuits prove that the model exhibits the expected behavior. In the future, it will be interesting to model a real output transformer and combine it with a tube model to simulate a real guitar amplifier's power stage.

7. REFERENCES

[1] R.C. de Paiva, J. Pakarinen, V. Välimäki, and M. Tikander, "Real-time audio transformer emulation for virtual tube amplifiers," *EURASIP J. Advances in Signal Process.*, vol. 2011, pp. 1–15, 2011.

[2] D.C. Jiles and D.L. Atherton, "Theory of ferromagnetic hysteresis," *J. of Magnetism and Magnetic Materials*, vol. 61, no. 1–2, pp. 48–60, Sept. 1986.

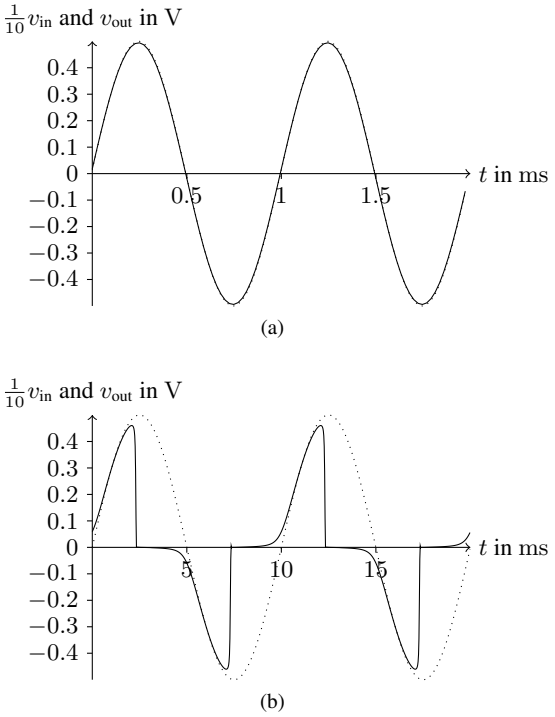


Figure 7: *Input voltage v_{in} (dotted) and output voltage v_{out} (solid) of the transformer circuit for a sinusoid at (a)* $1\,\mathrm{kHz}$ *and (b)* $100\,\mathrm{Hz}$.

[3] D.C. Jiles, J.B. Thoelke, and M.K. Devine, "Numerical determination of hysteresis parameters for the modeling of magnetic properties using the theory of ferromagnetic hysteresis," *IEEE Trans. on Magn.*, vol. 28, no. 1, pp. 27–35, Jan. 1992.

[4] J.H.B. Deane, "Modeling the dynamics of nonlinear inductor circuits," *IEEE Trans. Magn.*, vol. 30, no. 5, pp. 2795–2801, Sept. 1994.

[5] J. Macak and J. Schimmel, "Simulation of a vacuum-tube push-pull guitar power amplifier," in *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011, pp. 59–62.

[6] M. Holters and U. Zölzer, "A generalized method for the derivation of non-linear state-space models from circuit schematics," in *23rd European Signal Process. Conf. (EUSIPCO)*, Nice, France, 2015, pp. 1078–1082.

[7] J. Bezanson, S. Karpinski, V.B. Shah, and A. Edelman, "Julia: A fast dynamic language for technical computing," arXiv:1209.5145v1, 2012.

[8] J. Bezanson, A. Edelman, S. Karpinski, and V.B. Shah, "Julia: A fresh approach to numerical computing," arXiv:1411.1607v4, 2015.

A COSINE-DISTANCE BASED NEURAL NETWORK FOR MUSIC ARTIST RECOGNITION USING RAW I-VECTOR FEATURES

Hamid Eghbal-Zadeh
Department of Computational Perception,
Johannes Kepler University
Linz, Austria
hamid.eghbal-zadeh@jku.at

Matthias Dorfer
Department of Computational Perception,
Johannes Kepler University
Linz, Austria
matthias.dorfer@jku.at

Gerhard Widmer
Department of Computational Perception,
Johannes Kepler University
Linz, Austria
gerhard.widmer@jku.at

ABSTRACT

Recently, i-vector features have entered the field of Music Information Retrieval (MIR), exhibiting highly promising performance in important tasks such as music artist recognition or music similarity estimation. The i-vector modelling approach relies on a complex processing chain that limits by the use of engineered features such as MFCCs.

The goal of the present paper is to make an important step towards a truly *end-to-end* modelling system inspired by the i-vector pipeline, to exploit the power of Deep Neural Networks[1] (DNNs) to learn optimized feature spaces and transformations. Several authors have already tried to combine the power of DNNs with i-vector features, where DNNs were used for feature extraction, scoring or classification. In this paper, we try to use neural networks for the important step of i-vector post-processing and classification for the task of music artist recognition.

Specifically, we propose a novel neural network for i-vector features with a cosine-distance loss function, optimized with stochastic gradient decent (SGD). We first show that current networks do not perform well with unprocessed i-vector features, and that post-processing methods such as Within-Class Covariance Normalization (WCCN) and Linear Discriminant Analysis (LDA) are crucially important to improve the i-vector representation. We further demonstrate that these linear projections (WCCN and LDA) can not be learned using general objective functions usually used in neural networks.

We examine our network on a 50-class music artist recognition dataset using i-vectors extracted from frame-level timbre features. Our experiments suggest that using our network with fully unprocessed i-vectors, we can achieve the performance of the i-vector pipeline which uses i-vector post processing methods such as LDA and WCCN.

1. INTRODUCTION

In the area of MIR, music artist modelling can have different applications including in recommender systems, playlist generation and music similarity estimation. Each artist can be recognized by a combination of multiple factors such as musical instruments, genre and voice of the singer(s).

I-vector features first were proposed in the field of speaker verification [1] and after their revolutionary success, they were used in other areas such as emotion recognition [2], language recognition [3] and audio scene classification [4]. Recently, they were imported into the MIR domain, for singing language identification [5], music artist recognition [6] and music similarity [7].

I-vector features have shown to be a promising song-level representation for artists. These features project songs into a fixed-length and low-dimensional space, which is built from frame-level features such as Mel-Frequency Cepstrum Coefficients (MFCCs) using Factor Analysis (FA).

First by using a Universal Background Model (UBM) trained on a sufficient number of songs, similarities among all the songs of different artists are captured, then via FA these similarities are discarded and songs are projected into a new space called **Total Variability Space (TVS)** which contains the remaining factors that are in a stronger correlation with artist variability. Further, an estimation of these factors in each song is calculated which contains rich information about the artist. These estimated factors are called **identity vectors** or in short, **i-vectors**. The use of Neural Networks (NNs) in different areas is increasing every day and recent advances in this area, enabled researchers to tackle problems which were previously solved by a variety of different approaches in machine learning, now by only using NNs. The outcome is the appearance of different NN layers and architectures specialized for different tasks.

I-vector based systems usually follow a specific pipeline which contains a chain of different processing steps with specific goals. Multiple efforts are done by different researchers to come up with a solution that replaces each of those blocks with NNs. The reason is that once all of these blocks are replaced with a NN, they all can be connected through a deep network and optimized together using the back-propagation algorithm.

The frame-level feature extraction – a part of the i-vector extraction procedure – and scoring and classification of i-vectors are examples of the steps that have been replaced with NNs. Yet, a solution for post-processing the raw i-vectors[2] using neural networks is not provided in classification tasks. We seek for a NN-based solution to post-process and classify i-vectors without any help from the i-vector pipeline. We hope that our efforts makes us one step closer to an *end-to-end* music artist recognition system inspired by the i-vector pipeline, using neural networks.

In this paper, we extract i-vectors from frame-level timbre features and use them as input to NNs. By defining a cosine-distance loss function, we lead the network to learn a cosine metric which works the best with i-vector features. Our results suggest that us-

[1] The term **Deep Neural Networks** in this paper refers to **Multi-Layer Artificial Neural Networks** which use recent techniques from **Deep Learning** such as **batch-normalization**, **drop-out** and **stochastic gradient descent**.

[2] In this paper, i-vectors that are not encountered with linear projections such as LDA and WCCN are called *raw i-vectors*. In contrast, i-vectors that are projected with linear projections such as LDA and WCCN are called *processed i-vectors*.

ing our network, we can achieve the performance of the i-vector pipeline which is the state-of-the-art in music artist recognition.

The remainder of this paper is organized as follows. In Section 2, the related work is provided. In Section 3, the i-vector features are described. In Section 4, we explain our proposed neural network. In Section 5 the details of the experiments are explained and the results are reported. And finally, Section 6 concludes the paper.

2. RELATED WORK

For the task of Music Artist Recognition (MAR) in MIR, multiple approaches have been followed. Frame-level features [8], ensemble [9] and coding approaches [10] are some of the most used methods.

NNs are also known to perform well in MAR. In [11, 12, 13], Deep Belief Networks (DBNs) are used MAR using spectrograms and timbral features.

I-vector features have proven to be a promising song-level feature for MAR. In [6], i-vector features extracted from MFCCs and spectral features are used for MAR. Also, in [14] i-vectors have shown a significant performance for MAR in noisy environments.

Even though i-vector features are not used with DNNs for artist recognition, they are combined with deep learning techniques in many different ways. In [15, 16] speech recognition DNNs are used to produce statistical vectors needed for i-vector extraction. In [17], a DNN is used to extract a low-dimensional representation similar to i-vectors. Also, to extract bottleneck features used for i-vector extraction, DNNs are utilized in [18]. And in [19], DNNs are employed to manipulate post-processed i-vectors for speaker recognition.

In [20] DBNs are used with raw[3] i-vector features to model discriminatively the target and impostor i-vectors in a speaker verification scenario and in [21], DBNs and DNNs are combined together for single and multi-session speaker recognition.

The methods mentioned above that use raw i-vectors with DNNs, pursue adaptation purposes or have used DBNs which are trained in an unsupervised manner. Others, use the processed i-vectors as an input.

3. I-VECTOR FEATURES

3.1. Theoretical background

An i-vector refers to vectors in a low-dimensional space called Total Variability Space (TVS). The TVS models variabilities encountered with both the artist and song [7] where, the song variability defines as the variability exhibited by a given artist from one song to another.

TVS is created using a matrix $\mathbf{T}$ known as *TVS matrix*. This matrix is obtained by applying Factor Analysis (FA) on the adapted means of a Gaussian Mixture Model (GMM) known as Universal Background Model (UBM). This UBM is trained on the acoustic features of a sufficient amount of data. The means of UBM are then adapted to each song and then are used for the FA procedure explained in [1].

In the TVS, a given song is represented by an **i-vector** which indicates the directions that best separate different artists.

[3] I-vectors that are not encountered with post-processing methods such as LDA or WCCN.

A GMM mean supervector $\mathbf{M}$ adapted to a song from artist α can be decomposed as follows:

$$\mathbf{M} = \mathbf{m} + \mathbf{T}\mathbf{y} \tag{1}$$

where $\mathbf{m}$ is the GMM mean supervector and $\mathbf{T}\mathbf{y}$ is an offset. $\mathbf{y}$ is a latent variable with the standard normal prior. The i-vector is defined as the MAP estimate of $\mathbf{y}\mathbf{M}$ is assumed to be normally distributed with mean vector $\mathbf{m}$. The obtained i-vector is an artist and song dependent vector. The matrix $\mathbf{T}$ is used to extract i-vectors from statistical supervectors (known as $\mathbf{N_s}$ and $\mathbf{F_s}$) of songs which are computed using UBM.

We calculate statistical supervectors for a specific song s using UBM. These supervectors are $\mathbf{N_s}$ and $\mathbf{F_s}$ of song s:

$$\mathbf{N_s}^c = \sum_{t=1}^{L} \gamma_t(c), \quad \mathbf{F_s}^c = \sum_{t=1}^{L} \gamma_t(c) Y_t \tag{2}$$

where $\gamma_t(c)$ is the posterior probability of Gaussian component c of UBM for frame t and Y_t is the MFCC feature vector at frame t.

After calculating $\mathbf{N_s}$ and $\mathbf{F_s}$, using the statistical supervectors of songs in training set we learn the $\mathbf{T}$ matrix via an Expectation Maximization (EM) algorithm as follows: E-step, computes the probability of $P(w|X)$ where X is the given song and w is its i-vector. M-step, optimizes $\mathbf{T}$ by updating the following equation:

$$\mathbf{w} = (\mathbf{I} + \mathbf{T}^t \mathbf{\Sigma}^{-1} \mathbf{N(s)} \mathbf{T})^{-1} \mathbf{T}^t \mathbf{\Sigma}^{-1} \mathbf{F(s)} \tag{3}$$

where $\mathbf{N(s)}$ and $\mathbf{F(s)}$ are diagonal matrices with $\mathbf{N_s}^c\mathbf{I}$ and $\mathbf{F_s}^c\mathbf{I}$ on diameter. $\mathbf{I}$ is the identity matrix and $\mathbf{\Sigma}$ is the diagonal covariance matrix. The actual computation of an i-vector $\mathbf{w}$ for a given song s can be done using (3) after training $\mathbf{T}$. The curious reader is referred to [1, 22] for more information about the training procedure of $\mathbf{T}$.

3.2. I-vector post-processing and scoring

As explained before, i-vectors contain both artist and song variability. The song variability can be reduced by applying post-processing techniques such as LDA [23] and WCCN [24]. These techniques project i-vectors into a space which minimizes the song variability and maximizes the artist variability. In this section, we describe three techniques (LDA, WCCN and Length Normalization) that are frequently used in i-vector pipeline for post-processing. Also we describe a scoring method for classifying the i-vectors.

WCCN: provides a linear projection with an effective compensation. WCCN scales the i-vector space in the opposite direction of its inter-class covariance matrix, so that directions of intra-artist variability are improved for i-vector scoring.

LDA: yields a linear projection that tries to find a orthogonal basis with a better discrimination between different classes. LDA projection maximizes the between-class and minimize the within-class covariance of the data.

Length Normalization: Length (amplitude) of i-vectors are in correlation with negative effects such as song variability. For this reason, an iterative length-normalization for i-vectors is proposed in [25] where suggest to divide each i-vector by its length (norm). It is suggested to apply length-normalization before each post processing, also before feeding to the classifier/scoring step.

Cosine Scoring (CS): In the TVS, a simple cosine scoring has been successfully used to compare two i-vectors, as described in [26]. To predict the artist label for an i-vector, the artist with

the highest cosine score is chosen as the label where the score is defined as the cosine score of the given i-vector and class-averaged i-vectors[4].

3.3. I-vector pipeline

In Figure 1 (top), a diagram of an i-vector based system is shown. As you can see, first the frame-level features are extracted and then the i-vector models (such as UBM and **T** matrix) are trained and then i-vectors are computed. Further, these raw i-vectors are encountered with post-processing methods. First LDA is applied and the resulting i-vectors are projected using WCCN. Finally, LDA-WCCN projected i-vectors are used for scoring via Cosine Scoring.

4. THE PROPOSED NETWORK

In this section, we introduce our proposed NN for music artist recognition using i-vector features. As we show in Figure 1 (bottom), instead of using post-processing methods such as LDA and WCCN, and scoring methods such as cosine-scoring, our network is able to use raw i-vectors directly as an input.

Our experiments show that linear projections such as LDA and WCCN play a significant role in the performance of i-vector pipeline. Hence, we would like to replace such linear projections with a NN. We use linear activation function (LIN) in our NN. The reason to choose LIN is that other layer activation functions such as rectify units discard all the negative values by replacing them with zero Because i-vectors have a mean value close to zero, by using a rectified activation function [27], the layer's output activations that still have negative values, might be forced to throw away the information related to negative values of i-vector features.

Instead of the common loss functions used with NNs such as Mean Squared Error (MSE) and Categorical Cross-Entropy (CCE), we introduce a novel Cosine-distance based loss function for multi-class classification tasks using i-vector features. Our Cosine-Distance Based Neural Network (CDB-Net) is described in details in the following.

Architecture: Our CDB-Net consists of 4 layers with 1 hidden layer. The first hidden layer is a dense (fully connected) layer with linear activation function. It is followed by a batch-normalization layer [28] then a drop-out layer [29] and finally a dense output layer with linear activation function. Using the drop-out, prevents the network from over-fitting and let each feature to be learned, without relying on other dimensions. During training a NN the parameters of a layer change, and consequently the distribution of each layer's outputs changes as well. Batch-normalization layer normalizes each layer output for each training mini-batch. This allows us to use higher learning rates and be less careful about initialization. The aim of the first hidden layer is to learn a linear projection that improves the i-vector representation. We expect that since the i-vector pipeline benefits from LDA and WCCN linear projections, our CDB-Net also should be able to learn a projection with similar characteristics. Our first hidden layer has 400 hidden neurons (the same as the i-vectors dimensionality).

The output activations of this layer and one-hot encoding of the correct class are used to calculate a cosine loss. In the output layer, we use the same number of hidden neurons as our classes to produce a score for each class given an i-vector.

Cosine loss function: For the loss optimization, we use a novel Cosine-distance based loss. This loss is the cosine distance of the activations of the output layer with one-hot encoding of the correct class. The calculation of our proposed loss is as follows.

For a C classes problem, a one-hot encoding of class i ($i = 1, \ldots, C$) is one at the index i and zero otherwise. The cosine loss of the network for a batch size of b is defined as follows:

$$loss_{cos} = 1 - \frac{1}{b}\sum_{n=1}^{b} cos(out_n, l_n) \tag{4}$$

where out_n is the output activation of the output layer and l_n is the one-hot encoding for the correct class. Also, cos is a standard cosine similarity [26]. Then maximum of the cosine loss is equal to 1. Also, each one-hot encoding is orthogonal to the others. So the labels have the maximum distance from each other. Hence, minimizing the cosine loss of the output activations to the correct one-hot encoding, will increase the discrimination power of the network. The network tries to minimize this loss by using stochastic gradient descent. Then it back-propagates the error through the previous layers to update the weight of the layers.

This is not the first time that cosine-distance is used as loss in a NN. In [30], a NN optimized with a similar cosine distance-based loss function is used for the task of signature verification. Although the cosine loss function in [30] is used to process both imposter and target signature features. Some of the differences between our CDB-Net loss and loss used in [30] can be explained as: 1) the loss definitions are different. The loss in [30] is defined for a special NN called "Siamese" containing two identical sub-networks with a special training procedure. This network is designed to compare two signatures for signature verification. The loss defined in our CDB-Net can be used in any multi-class classification task. 2) The Siamese network training tries to minimize the cosine distance of two output activations from two different signatures. CDB-Net minimizes the cosine distance of the output activations of a given i-vector with its correct one-hot encoding label in a discriminative manner.

Network distance metric: The proposed network is forced to use the cosine distance metric in its optimization because of two reasons: 1) the input raw i-vectors are length-normalized. So the network can not distinguish between different classes using a distance metric such as Euclidean distance that relies on the amplitude of the output activations. 2) The loss function is a cosine distance between output layer activation and their respective one-hot encoding class label. So the network's objective function is defined by a cosine metric.

5. EXPERIMENTS

5.1. Data

Similar to the data used in [12] for music artist recognition, we used a subset of Million Song Dataset [31] (MSD) in the artist recognition experiments. We follow a similar procedure as used in [12]. We first removed the duplications from MSD by using the official duplication list provided in MSD website which reduced the number of songs from one million to around 900,000. Then we selected all the artists with more than 100 songs. From these artists, we selected top 50 artists with more songs and further selected 100 random songs from each artist (in sum, 5,000 songs)

[4] class-averaged i-vectors are defined as the average of i-vectors in each class, in training set.

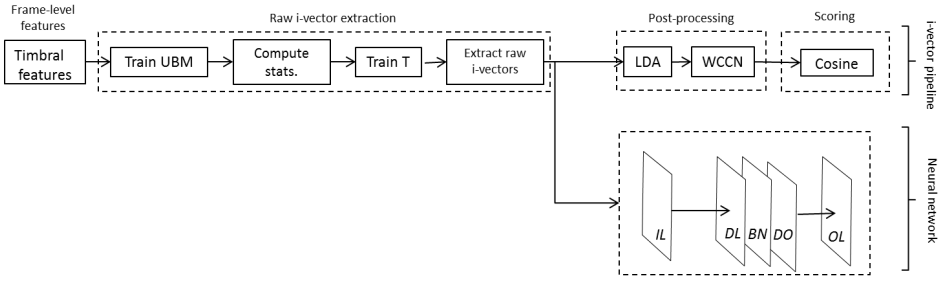


Figure 1: *The block-diagram of i-vector based artist recognition system. On the top, classic state-of-the-art i-vector pipeline. On the bottom, the proposed artist recognition method using a deep network.* ***IL****: input layer.* ***DL****: dense layer.* ***BN****: batch-normalization layer.* ***DO****: dropout layer.* ***OL****: output layer.*

as our dataset for the experiments in this paper. 80% of the songs are used for training and the rest are used for validation (10%) and testing (10%). We trained our models on training set, optimized using validation set and reported the results on the testing set.

The results reported in [12] (35.74% accuracy) can not be compared with the performance of our network because of two reasons:

1) in [12] both timbre and chroma features are used together. and 2) the performance reported in [12] is in a bar level (which is smaller than a song), but i-vectors are song-level features and our performance is reported in the song level.

5.2. Features

Using the features available through The Echo Nest API[5], we extracted i-vectors from Echo Nest Analyzer's timbre feature. Each Echo Nest Analyzer's timbre feature consists of a vector that includes 12 unbounded values roughly centered around 0. Those values are high level abstractions of the spectral surface, ordered by degree of importance such as the average loudness of the segment, brightness, the flatness of a sound and attack [32]. As meta-data, we used the artist ids provided in MSD in our artist recognition experiments. To extract i-vectors, all the Echo Nest timbre features for each song are used.

5.3. Setup

I-vector extraction: For i-vector extraction, a 1024 components GMM is trained as UBM on Echo Nest timbre features. Then, **T** matrix with 400 dimensions is learned using the statistics computed from Echo Nest timbre features and UBM. The training set is used to train UBM and **T**. Further i-vectors are extracted using UBM and **T** for both training and testing sets. All the i-vectors are length normalized.

The **T** matrix is trained using matlab MSR identity toolbox [33].

CDB-Net: We used 400 neurons in our hidden layer and 50 neurons in the output layer with 50. The hidden layer is followed by batch normalization and 50% droup-out.

We apply learning-rate schedule during training and decrease the learning rate by its half after each 10 epochs. The initial learning rate is 1.0 the learning-rate starts to decrease at the 100^{th} epoch out of 200 epochs used for training.

[5]http://the.echonest.com/

A stochastic gradient descent (SGD) with back-propagation algorithm and a momentom of 0.9 is used with the batch size of 500 samples. The one-hot coding of the labels are computed to be used in the cosine-loss calculation.

For all of our experiments with NNs, the open-source python library *Lasagne* [34] is used. Our network is implemented in Python using *Theano* [35].

We used a PC running on Linux with a NVIDIA Titan X GPU card, an Intel Core i7 CPU and 16 GB of RAM for our experiments. All the NN experiments are optimized on GPU.

We use the averaged F-measure to compare the performance of different methods. This measurement is calculated by averaging the F-measures of all the classes in each experiment.

Baselines: Four baseline methods are used in this work. We use the i-vector pipeline method and three NNs similar to our CDB-Net as baselines. The difference between our CDB-Net and other NN baselines is the loss function and the activation functions of the hidden layer. Our first baseline is the Cosine Scoring (CS) used in i-vector pipeline. CS first projects i-vectors using LDA and then by WCCN. Further, it uses the cosine distance to calculate a score for each given testing i-vector and class-averaged i-vectors from training set. Finally, it classifies testing i-vectors by minimizing the cosine score. Similar to the architecture of CDB-Net, our second baseline (CCE-REC-Net) is a 4 layers feed-forward network with 1 hidden layer of 400 neurons which uses rectified activation function. The hidden layer is followed by batch-normalization layer and a drop-out layer with 50% drop outs. At the output layer, CCE-REC-Net uses a soft-max activation function. CCE-REC-Net is optimized using a CCE loss function.

Our third baseline (CCE-TAN-Net) is a a 4 layers feed-forward network with 1 hidden layer of 400 neurons. CCE-TAN-Net has exactly the same architecture as CCE-REC-Net, only uses $\tanh$ activation function instead of rectified activation function. Our fourth baseline (CCE-LIN-Net) is also a a 4 layers feed-forward network with 1 hidden layer of 400 neurons. CCE-LIN-Net has exactly the same architecture as CCE-REC-Net, only uses linear activation function instead of rectified activation function.

Experiment design: We examine the performance of our CDB-Net in three different experiments: 1) dealing with LDA-WCCN projected i-vectors, 2) dealing with raw i-vectors and 3) The effect of weight initialization in NNs.

In our first experiment, we compare the performance of CDB-Net and our baselines on processed i-vectors. Since the i-vector pipeline uses LDA and then WCCN projections for post-

processing, we use the LDA, then WCCN projected i-vectors in our first experiment. Our first experiment reveals how different networks–as well as the i-vector pipeline–deal with processed i-vectors. The results of this experiment are provided in Table 1

Our second experiment compares the CDB-Net with the baselines encountering raw i-vectors. This experiment is the core of this paper and shows how our CDB-Net performs compared to all the other baselines.

Finally, in our third experiment, we study the effect of weight initialization in the hidden layer of the baseline NNs as well as our CDB-Net. In [36] the importance of layer's weight initialization in feed-forward NNs is discussed in details. In the experiments 1 and 2, we initialize the weight of our hidden layer from a uniform distribution as explained in [36]. In experiment 3, we would like to compare this initialization with an initialization using the LDA-WCCN projection matrix which is used in the i-vector pipeline for i-vector post-processing.

Even though we are aware that initializing the hidden layer with LDA-WCCN projection matrix is similar to use processed i-vectors, we would like to provide proof that our CDB-Net is able to find an optimum point that other NNs are unable to find using SGD without a proper initialization. If we initialize the hidden layer's weight, or use processed i-vectors, other netowrks are able to reach that optimum point.

Using a LDA-WCCN projection matrix, we initialize the hidden layer's weight matrix in all of our baseline NNs as well as our CDB-Net. Then we feed all the networks with raw i-vectors. The LDA-WCCN projection matrix is computed by multiplying LDA and WCCN projection matrices. In [26] a procedure is described about how to combine LDA and WCCN projection matrix to be used with i-vector features and cosine scoring in an efficient way. We follow the same procedure to compute our LDA-WCCN projection matrix.

5.4. Results

The performance of NNs with LDA-WCCN projected i-vectors can be found in Table 1. Also, the performance of the i-vector pipeline can be found under (CS) name.

It can be seen that using LDA-WCCN projected i-vectors, all the networks achieved similar performances to the i-vector pipeline. Also, it can be observed that CCE-LIN-Net and CDB-Net achieved better performances than CCE-REC-Net and CCE-TAN-Net. It shows that rectified and $\tanh$ activation functions were not useful, as expected.

As the main experiment of this paper, in Table 2 we compare the CDB-Net with other baselines using raw i-vectors. As can be seen, the performance of the i-vector pipeline is very poor without LDA-WCCN projection step. This shows the importance of the post-processing for i-vector features. Also, looking at the other baseline NNs, it can be seen that all the other baseline networks also could not achieve a F-measure of more than 41.46%. This show that similar to i-vector pipeline, post-processing is very effective to process i-vector features using NNs optimized with CCE loss function. The proposed CDB-Net could achieve the performance of **57.86%** and outperformed all the baselines. The good performance of CDB-Net reveals that even though no weight initialization using LDA-WCCN projections were used, also i-vectors were not processed with such linear projections, using cosine loss function was very effective to optimize the network. From the second experiment, we can observe that changing the loss function from CCE to cosine, the NN's behavior changes and it can find much better optimum points which leads to achieving much higher performances.

In our final complimentary experiment (exp. 3) we studied the effect of weight initialization with LDA-WCCN projection matrix in the hidden layer. In Table 3 it can be seen that as expected, by initializing the weight of the hidden layer with the projection matrices of LDA-WCCN, all the baseline methods and the proposed method achieved similar performances to exp. 1 that processed i-vectors were used.

By looking at 3 baselines used in experiment 3, it can be seen that the networks which previously did not perform well with raw i-vectors, now can perform much better if the hidden layer's weight matrix initializes with the LDA-WCCN projection matrix. Even though the results are not surprising as we observed from the performance of our baseline NNs in experiment 1, we learned that it is not necessary to only use processed i-vectors to achieve good performances with NNs. By a right initialization, the performance of a CCE-optimized NN can be significantly improved.

Table 1: ***Experiment 1**- Artist recognition F-measure using **processed (LDA-WCCN projected) i-vector features** and different methods. The method marked with an asterisk (∗) is the i-vector pipeline.*

method	in. dim.	hid. neu.	out. layer	F1 (%)
*CS	49	–	–	56.17
CCE-REC-Net	49	49	SM	54.12
CCE-TAN-Net	49	49	SM	57.45
CCE-LIN-Net	49	49	SM	**59.77**
CDB-Net	49	49	LIN	58.24

Table 2: ***Experiment 2**- Artist recognition F-measure using **raw i-vector features** and different methods.*

method	in. dim.	hid. neu.	out. layer	F1 (%)
CS	400	–	–	26.99
CCE-REC-Net	400	400	SM	37.10
CCE-TAN-Net	400	400	SM	40.26
CCE-LIN-Net	400	400	SM	41.46
CDB-Net	400	400	LIN	**57.86**

Table 3: ***Experiment 3**- Artist recognition F-measure using **raw i-vector features** with **LDA-WCCN weight initialization** for different methods.*

method	in. dim.	hid. neu.	out. layer	F1 (%)
CCE-REC-Net	400	49	SM	53.33
CCE-TAN-Net	400	49	SM	57.14
CCE-LIN-Net	400	49	SM	**58.41**
CDB-Net	400	49	LIN	56.89

6. CONCLUSION

Our experiment results (exp. 1) suggest that feed-forward NNs can be used as a classifier with processed i-vectors and achieve the performance of i-vector pipeline. Also in exp. 2 we showed that the

same NNs that performed well with processed i-vectors, are unable to achieve good performances with raw i-vectors and the performance of these NNs drops significantly when the post-processing step is removed.

To tackle this problem, we introduced a NN with a cosine-distance loss function and linear dense layers. We showed that this network can achieve the performance of the NNs that used processed i-vectors. It demonstrate that our network has the ability to learn similar projections to LDA-WCCN which significantly improve the i-vector representation for music artist recognition.

Our complimentary experiment results (exp. 3) suggest we can improve the performance of the feed-forward NNs by initializing their hidden layer's weights using LDA-WCCN projection matrix.

7. ACKNOWLEDGMENTS

This work was supported by the Austrian Science Fund (FWF) under grant no. Z159 (Wittgenstein Award) and by the Austrian Ministry for Transport, Innovation and Technology, the Ministry of Science, Research and Economy, and the Province of Upper Austria in the frame of the COMET center SCCH. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of a Titan X GPU used for this research.

8. REFERENCES

[1] Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, 2011.

[2] Rui Xia and Yang Liu, "Using i-vector space model for emotion recognition.," in *INTERSPEECH*, 2012.

[3] Najim Dehak, Pedro A Torres-Carrasquillo, Douglas A Reynolds, and Reda Dehak, "Language recognition via i-vectors and dimensionality reduction.," in *INTERSPEECH*. Citeseer, 2011.

[4] Benjamin Elizalde, Howard Lei, and Gerald Friedland, "An i-vector representation of acoustic environments for audio-based video event detection on user generated content," in *ISM*. IEEE, 2013.

[5] Anna M Kruspe, "Improving singing language identification through i-vector extraction," in *DAFx*, 2011.

[6] Hamid Eghbal-Zadeh, Markus Schedl, and Gerhard Widmer, "Timbral modeling for music artist recognition using i-vectors," in *EUSIPCO*, 2015.

[7] Hamid Eghbal-zadeh, Bernhard Lehner, Markus Schedl, and Gerhard Widmer, "I-vectors for timbre-based music similarity and music artist classification," in *ISMIR*, 2015.

[8] Daniel PW Ellis, "Classifying music audio with timbral and chroma features," in *ISMIR*, 2007.

[9] James Bergstra, Norman Casagrande, Dumitru Erhan, Douglas Eck, and Balázs Kégl, "Aggregate features and adaboost for music classification," *Machine learning*, 2006.

[10] Pavel P. Kuksa, "Efficient multivariate kernels for sequence classification," *CoRR*, 2014.

[11] Honglak Lee, Peter Pham, Yan Largman, and Andrew Y Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in neural information processing systems*, 2009, pp. 1096–1104.

[12] Sander Dieleman, Philémon Brakel, and Benjamin Schrauwen, "Audio-based music classification with a pretrained convolutional network," in *ISMIR*, 2011.

[13] Philippe Hamel and Douglas Eck, "Learning features from music audio with deep belief networks.," in *ISMIR*. Utrecht, The Netherlands, 2010.

[14] Hamid Eghbal-Zadeh and Gerhard Widmer, "Noise robust music artist recognition using i-vector features," in *ISMIR*, 2016.

[15] Yun Lei, Luciana Ferrer, Moray McLaren, et al., "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014.

[16] Patrick Kenny, Vishwa Gupta, Themos Stafylakis, P Ouellet, and J Alam, "Deep neural networks for extracting baum-welch statistics for speaker recognition," in *Proc. Odyssey*, 2014.

[17] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Jorge Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014.

[18] Pavel Matejka, Le Zhang, Tim Ng, HS Mallidi, Ondrej Glembek, Jeff Ma, and Bing Zhang, "Neural network bottleneck features for language identification," *Proc. of IEEE Odyssey*, 2014.

[19] Albert Jiménez Sanfiz, "Deep neural networks for channel compensated i-vectors in speaker recognition," *BA Thesis, Universitat Politécnica De Catalunya*, 2014.

[20] Omid Ghahabi and Juan Hernando, "Deep belief networks for i-vector based speaker recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014.

[21] Omid Ghahabi and Javier Hernando, "Deep learning for single and multi-session i-vector speaker recognition," *arXiv preprint arXiv:1512.02560*, 2015.

[22] Patrick Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," *CRIM, Montreal,(Report) CRIM-06/08-13*, 2005.

[23] Bernhard Scholkopft and Klaus-Robert Mullert, "Fisher discriminant analysis with kernels," *Neural networks for signal processing IX*, 1999.

[24] Andrew O Hatch and Andreas Stolcke, "Generalized linear kernels for one-versus-all classification: application to speaker recognition," *Proc. Int. Conf. Acoust. Speech and Signal Process.*, 2006.

[25] Daniel Garcia-Romero and Carol Y Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems.," in *INTERSPEECH*, 2011.

[26] Najim Dehak, Reda Dehak, James R Glass, Douglas A Reynolds, and Patrick Kenny, "Cosine similarity scoring without score normalization techniques.," in *Odyssey*, 2010.

[27] Xavier Glorot, Antoine Bordes, and Yoshua Bengio, "Deep sparse rectifier neural networks," in *International Conference on Artificial Intelligence and Statistics*, 2011.

[28] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, 2014.

[30] Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah, "Signature verification using a siamese time delay neural network," *International Journal of Pattern Recognition and Artificial Intelligence*, 1993.

[31] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere, "The million song dataset," in *ISMIR 2011: Proceedings of the 12th International Society for Music Information Retrieval Conference, October 24-28, 2011, Miami, Florida*. University of Miami, 2011.

[32] Tristan Jehan and Davis DesRoches, "Analyzer documentation," *The Echo Nest*, 2011.

[33] Seyed Omid Sadjadi, Malcolm Slaney, and Larry Heck, "Msr identity toolbox-a matlab toolbox for speaker recognition research," *Microsoft CSRC*, 2013.

[34] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, SK Sønderby, D Nouri, D Maturana, M Thoma, E Battenberg, J Kelly, et al., "Lasagne: First release," *Zenodo: Geneva, Switzerland*, 2015.

[35] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, 2016.

[36] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International conference on artificial intelligence and statistics*, 2010.

[37] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

[38] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[39] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th international conference on machine learning (ICML-13)*, 2013.

[40] Mohamad Hasan Bahari, Rahim Saeidi, David Van Leeuwen, et al., "Accent recognition using i-vector, gaussian mean supervector and gaussian posterior probability supervector for spontaneous telephone speech," in *ICASSP*. IEEE, 2013.

[41] Andrew O Hatch, Sachin S Kajarekar, and Andreas Stolcke, "Within-class covariance normalization for svm-based speaker recognition.," in *INTERSPEECH*, 2006.

HUBNESS-AWARE OUTLIER DETECTION FOR MUSIC GENRE RECOGNITION

Arthur Flexer *

Austrian Research Institute for Artificial Intelligence
Freyung 6/6/7, Vienna, Austria
arthur.flexer@ofai.at

ABSTRACT

Outlier detection is the task of automatic identification of unknown data not covered by training data (e.g. a new genre in genre recognition). We explore outlier detection in the presence of hubs and anti-hubs, i.e. data objects which appear to be either very close or very far from most other data due to a problem of measuring distances in high dimensions. We compare a classic distance based method to two new approaches, which have been designed to counter the negative effects of hubness, on two standard music genre data sets. We demonstrate that anti-hubs are responsible for many detection errors and that this can be improved by using a hubness-aware approach.

1. INTRODUCTION

Outlier detection[1] is the identification of new or unknown data that a machine learning system is not aware of during training (see [18] for a recent review and [29] for a survey on high-dimensional outlier detection). It is a fundamental requirement for every machine learning system to automatically identify data from regions not covered by the training data since in this case no reasonable decision can be made. In the case of music information retrieval (MIR), an application scenario is the rejection of songs from a previously unseen genre in genre recognition. The same holds for other classifications tasks (e.g. tag or mood), but also for retrieval of similar songs in case a query song is too different from all other songs in a data base. Another example is the automatic rejection of songs from play-lists because they do not fit the overall flavor of the majority of the list. Only little research on outlier detection in MIR so far exists [6, 12, 26, 8].

Hubness is a general problem of learning in high-dimensional spaces and has been recognized as a new aspect of the curse of dimensionality in machine learning literature [20, 23]. Hub objects appear very close to many other data objects and anti-hubs very far from most other data objects. It has been argued and demonstrated that anti-hubs might act as 'artificial' outliers since they are far away from many other data points [20]. A recent review on outlier detection in high dimensional data concluded that the "relation of hubness and outlier degree appears to be remaining an open issue" [29]. It has been demonstrated [10], that many MIR models are inherently high-dimensional and highly prone to hubness. In a first MIR study on outlier detection in high dimensions [8], we were able to show that it is possible to improve outlier detection by using a hubness reduction method as a preprocessing step.

* This work was supported by the Austrian Science Fund (FWF, grant P27082).

[1] Please note that the terms outlier and novelty detection are closely related although not fully synonymous. We will use the term outlier throughout the paper without further distinction for reasons of convenience.

In this paper we explore whether such improvements in high-dimensional outlier detection are on account of the changed role of hubs and anti-hubs due to the hubness reduction method. This is done by analyzing the performance of hubs and anti-hubs in a classic distance based method and in two hubness-aware approaches, all applied in a music genre recognition setting.

2. RELATED WORK

Outlier detection, also known as novelty detection, is the task of automatically recognizing data that differ in some respect from the data seen during training by a machine learning system. In case new data differs substantially from training data, no sensible decision can be made by a machine learning system. This should of course be an integral part of any data analysis system and therefore a vast literature concerning the topic exists. For this paper, we follow the systematic of a very recent and comprehensive review [18], which also contains a representative list of references concerning the topic. According to this review, outlier detection can be distinguished into probabilistic, distance-based, reconstruction-based, domain-based and information theoretic approaches. The methods we will present in Section 4 are all distance-based, more specifically based on nearest neighbor information. They are also all unsupervised, i.e. class labels are not needed for detection of outliers. Of greater importance is a recent review that deals specifically with outlier detection in high-dimensional data [29]. Although the authors show how some classic outlier detection methods are affected by the concentration of distances (see also next paragraph), hubness is only reviewed as a remaining open issue.

The concept and term of hubness has been discovered and first described in MIR [1], but then gained attention in a machine learning context where it has been discussed as a new aspect of the curse of dimensionality and a general problem of learning in high-dimensional spaces [20, 23]. Hubness is related to the phenomenon of concentration of distances, which is the fact that all points are at almost the same distance to each other for dimensionality approaching infinity [11]. Radovanović et al. [20] presented the argument that for any finite dimensionality, some points are expected to be closer to the center of all data than other points and are at the same time closer, on average, to all other points. Such points closer to the center have a high probability of being hubs, i.e. of appearing in nearest neighbor lists of many other points. Points which are further away from the center have a high probability of being anti-hubs, i.e. points that never appear in any nearest neighbor list. Hubness has been shown to have a negative impact on many tasks including classification [20], nearest neighbor based recommendation [10] and retrieval [24], clustering [27, 22] and visualization [4]. Many of these reports are from the MIR community (e.g. [14, 2, 10, 4, 5]). It also affects data from diverse domains including multimedia (text, music, images, speech), biol-

ogy and general machine learning (see [20, 23, 3] for large scale empirical studies).

In order to reduce hubness and its negative effects, we have proposed two unsupervised methods to re-scale high-dimensional distance spaces [23]: Local Scaling (LS) and Mutual Proximity (MP). Both methods aim at repairing asymmetric nearest neighbor relations. The asymmetric relations are a direct consequence of the presence of hubs. A hub y is the nearest neighbor of x, but the nearest neighbor of the hub y is another point a ($a \neq x$). This is because hubs are by definition nearest neighbors to very many data points but only one data point can be the nearest neighbor to a hub. The principle of the scaling algorithms is to re-scale distances to enhance symmetry of nearest neighbors. A small distance between two objects should be returned only if their nearest neighbors concur. Application of LS and MP resulted in a decrease of hubness and an accuracy increase in k-nearest neighbor classification on thirty real world datasets including text, image and music data.

Just recently we re-visited our own results [6] on outlier detection in MIR and tried to use mutual proximity (MP) for the task [8]. After all, MP rescales distances to probabilities of mutual proximity which allows for convenient thresholding to detect outlier data due to its probabilistic interpretation. We were able to show that outlier detection based on MP improves the ability to reject outlier data when compared to a classic distance based method, again in a genre classification context. A necessary next step is to investigate whether the improvement achieved is on account of the changed role of hubs and anti-hubs due to the application of mutual proximity. It seems clear that anti-hubs, being far away from most points, will probably always be rejected as outliers and that hub objects, being close to many points, should be harder to reject. It is therefore our hypothesis that in high-dimensional data, hub and anti-hub points are responsible for many errors being made when rejecting data.

Such a first analysis of the role of anti-hubs in outlier detection has recently been presented [19]. More specifically, the authors have analysed two variants of the ODIN method [13], *k*-NN outlier scoring [21] and three other methods concerning their relation to anti-hubs. The two variants of the ODIN method use reverse nearest neighbor counts, i.e. counts of how often every data object appears among the *k* nearest neighbors of every other data object. Per definition anti-hubs have very small or even zero reverse nearest neighbor counts. The authors show that outlier scores based on these counts are correlated to scores from other detection methods but do provide some extra information. Outlier detection results of the ODIN-based methods are however rather mixed when compared to other methods applied to twelve real world data sets. It also has to be said that these data sets are not really very high dimensional (from only 5 to at most 100) and therefore most of them are probably not affected by hubness at all.

Concerning the dimensionality of data sets, it is important to note that the degree of concentration and hubness is linked to the intrinsic rather than extrinsic dimension of the data space. Whereas the extrinsic dimension is the actual number of dimensions of a data space, the intrinsic dimension is the, often much smaller, number of degrees of freedom of the submanifold in which the data space can be represented [11]. Our previous research [23] has shown that real world data with extrinsic dimensionality as small as 34 can already exhibit the negative effects of hubness, while other data with extrinisc dimensionality of more than 10000 is still not affected. It is also true that simple dimensionality reduction does not reduce hubness. On the contrary it has been shown that only projections to very few dimensions, well below the intrinsic dimensionality of a data set, are able to reduce hubness, but at the cost of a loss of distance information [20].

3. DATA

For our analysis, we chose to use a genre classification framework, following the hypothesis that songs within a certain genre are more similar to each other than songs from different genres. During evaluation in Section 5, we will always reserve all songs belonging to one of the genres as outlier songs to be detected against the rest of the songs from all other genres. Songs from an unknown genre therefore act as outliers.

For our experiments we used two standard music databases: the "GTZAN" collection consisting of $N = 1000$ audio tracks (each 30 s length) evenly spread over $G = 10$ music genres [28]; the "ISMIR2004" [2] collection containing $N = 1458$ tracks of $G = 6$ genres, with full-length audio being available and exhibiting a highly imbalanced genre distribution with classical music comprising almost half of the tracks.

We decided to compute timbre information from the audio, since this is an integral part of many MIR systems and at the same time has already been shown to be susceptible to hubness [10]. Every track is divided into overlapping frames for which 20 MFCCs are being computed which are modeled via a single Gaussian with full covariance matrix. To compute a distance value between two Gaussians the symmetrized Kullback-Leibler (SKL) divergence is used (see [17] for details on both MFCCs and SKL). This results in $N \times N$ distance matrices D^G and D^I for the GTZAN and ISMIR data sets. Therefore the data sets are represented as distance spaces only, not vector spaces, and it is not possible to report their extrinsic dimensionality. Their intrinsic dimensionality measured via a maximum likelihood estimator [16] is 10.94 for GTZAN and 9.00 for ISMIR.

4. METHODS

We now describe all three methods we will use for outlier detection. All of them compute an outlier score (S^{kNN}, S^{AH} and S^{MP}), which is bounded between 0 and 1 and is compared to a threshold p to decide whether a data object is an outlier or not. A data object is rejected if:

$$S > p. \tag{1}$$

4.1. kNN-reject

The first method is a standard distance-based approach known as *k*-NN outlier scoring [21]. The outlier score is the average distance to the k nearest neighbors:

$$S^{kNN}(x) = \frac{1}{k} \sum_{i=1}^{k} D_{x,\mathrm{NN}_i(x)}, \tag{2}$$

with $\mathrm{NN}_i(x)$ being the ith nearest neighbor of x. The distance matrices D^G and D^I (see Section 3) are normalized to the interval 0 to 1 by subtracting the minimum distance and dividing through the maximum distance. This also bounds the outlier score S^{kNN} between 0 and 1.

[2] `http://ismir2004.ismir.net/genre_contest/index.html`

4.2. AH-reject

The next method is based on previous work of using reverse nearest neighbor counts for outlier detection [19]. In hubness research, the reverse nearest neighbor count of a point x is usually called n-occurrence $O^n(x)$ [20]. It is the number of times x occurs in the first n nearest neighbors of all other objects in the collection. The proposed method simply uses $O^n(x)$ as the outlier score. It has been termed "Antihub" because anti-hubs have very small or even zero n-occurrence and are therefore very likely to be rejected as outliers. The same authors proposed a variant called "Antihub2" which also includes information from the k nearest neighbors of x:

$$S^{AH}(x) = (1-\alpha)\frac{1}{O^n(x)+1} + \alpha\sum_{i=1}^{k}\frac{1}{O^n(\mathrm{NN}_i(x))+1}. \quad (3)$$

We set $\alpha = k/(k+1)$, which basically gives the average across a function of the n-occurrences of x itself and its k nearest neighbors $\mathrm{NN}_i(x)$. The outlier score S^{AH} is also bounded between 0 and 1, with $S^{AH} = 1$ in case all involved n-occurrences O^n are equal zero, and $S^{AH} = 1/N$ in case all involved $O^n = N-1$. An n-occurrence is equal $N-1$ in case a data point appears in all neighborhood lists of all other data points.

4.3. MP-reject

Mutual Proximity (MP) [23] rescales the original distance space so that two objects sharing similar nearest neighbors are more closely tied to each other, while two objects with dissimilar neighborhoods are repelled from each other. MP has been devised to counter the negative effects of hubness in high dimensional data spaces. For MP-reject we exploit the fact that MP rescales distances to probabilities which enables comparability and simple thresholding. MP reinterprets the distance of two objects as a mutual proximity in terms of their distribution of distances. To compute MP, we assume that the distances $D_{x,i=1..N}$ from an object x to all other objects in our data set follow a certain probability distribution $P(X)$, thus any distance $D_{x,y}$ can be reinterpreted as the probability of y being the nearest neighbor of x, given their distance $D_{x,y}$ and the probability distribution $P(X)$:

$$P(X > D_{x,y}) = 1 - P(X \leq D_{x,y}) = 1 - \mathcal{F}_x(D_{x,y}), \quad (4)$$

with $\mathcal{F}$ denoting the cumulative distribution function (cdf). MP is then defined as the probability that y is the nearest neighbor of x given $P(X)$ and x is the nearest neighbor of y given $P(Y)$:

$$MP(D_{x,y}) = P(X > D_{x,y} \cap Y > D_{y,x}). \quad (5)$$

To compute MP in our experiments we assume that the distances $D_{x,i=1..N}$ follow a Gaussian distribution. We define the outlier score as the average of the MP-distances to the k nearest neighbors of x:

$$S^{MP}(x) = \frac{1}{k}\sum_{i=1}^{k}(1 - MP(x, \mathrm{NN}_i(x))), \quad (6)$$

with $\mathrm{NN}_i(x)$ being the ith nearest neighbor of x. Please note that we use the term $(1-MP)$ because mutual proximity computes similarities and we need distances for the rejection rule. Outlier score S^{MP} is bounded between 0 and 1 since it is based on MP which computes a probability.

5. RESULTS

Before evaluating the outlier detection methods, we present an analysis of the hubness of the data sets GTZAN and ISMIR in Table 1. The table gives the number of data objects N, number of genres G, the number of hubs $\#hub$, anti-hubs $\#anti$ and normal $\#normal$ data objects. Anti-hubs are defined as data objects with an n-occurrence O^n (see Sec. 4.2) equal 0, hubs with $O^n > 5n$, all based on numbers of nearest neighbors of $n = 5$. Normal data objects are all non-hub and non-anti-hub objects, i.e. $0 < O^n \leq 5n$. Please note that the mean n-occurrence across all objects in a data base is equal to n. Any n-occurrence significantly bigger than n therefore indicates existence of a hub. As in previous work [23], we chose objects appearing more than five times the expected value ($5n = 25$) as hub objects. As can be seen, consistent with theory, in both data sets there are small numbers of hubs (GTZAN 21, ISMIR 24) and large numbers of anti-hubs (GTZAN 186, ISMIR 276). The last column of Table 1 gives the hubness H^n. It is the skewness of the distribution of n-occurrences, i.e. the third moment of the distribution. A data set having high hubness produces few hub objects with very high n-occurrence and many anti-hubs with n-occurrence of zero. This makes the distribution of n-occurrences skewed with positive skewness indicating high hubness. The hubness values H^n of 3.29 for GTZAN and 3.94 for ISMIR show that there is a clear hubness effect in these data sets. Previous work [23] has shown that values above 1.4 are already problematic.

Table 1: *Hubness analysis of data sets GTZAN and ISMIR, see Sec. 5.*

data set	N	G	$\#hub$	$\#anti$	$\#normal$	H^n
GTZAN	1000	10	21	186	793	3.29
ISMIR	1458	6	24	276	1158	3.94

To evaluate the three outlier detection methods described in Sec. 4 we use the following approach shown as MATLAB style pseudo-code in Table 2. First we set aside all songs belonging to a genre `g` as new songs (`[new,data]=separate(alldata, g)`) which yields data sets `new` and `data` (all songs not belonging to genre g). Then we do a $C = 10$-fold crossvalidation using `data` and `new`: we randomly split `data` into `train` and `test` fold (`[train,test] = split(data,c)`) with `train` always consisting of 90% and `test` of 10% of `data`. We compute the percentage of `new` songs which are rejected as being outliers (`outlier_reject(g,c) = outlier(new)`) and do the same for the `test` songs (`test_reject(g,c) = outlier(test)`). Last we compute the classification accuracy on `test` data that has not been rejected as being outliers (`accuracy(g,c) = classify(test(not test_reject))`). As a classifier we use simple one-nearest neighbor classification. The evaluation procedure gives $G \times C$ (GTZAN 10×10, ISMIR 6×10) matrices of `outlier_reject`, `test_reject` and `accuracy` for each parameterization of the outlier detection approaches, i.e. for different values of k (see Sec. 4). In what follows we always report average numbers across these $G\times C$ sized matrices of results, i.e. averages across crossvalidation folds and genres.

The results for outlier detection are given in Figs. 5 and 6 as Receiver Operating Characteristic (ROC) curves. To obtain an ROC curve, the fraction of false positives (object is not an outlier

Table 2: *Outline of evaluation procedure, see Sec. 5.*

```
for g = 1 : G
  [new,data] = separate(alldata,g)
  for c = 1 : C
    [train,test] = split(data,c)
    outlier_reject(g,c) = outlier(new)
    test_reject(g,c) = outlier(test)
    accuracy(g,c) =
      classify(test(not test_reject))
  end
end
```

but it is rejected, in our case `test_reject`) is plotted versus the fraction of true positives (object is an outlier and correctly rejected, in our case `outlier_reject`) for varying threshold values p. We vary the threshold values from $p = 0$ to $p = 1$ in steps of .02. An ROC curve shows the trade off between how sensitive and how specific a method is. Any increase in sensitivity will be accompanied by a decrease in specificity. If a method becomes more sensitive towards outlier objects it will reject more of them but at the same it will also become less specific and also falsely reject more non-outlier objects. Consequently, the closer a curve follows the left-hand border and then the top border of the ROC space, the better the performance of the method is.

To summarize the information contained in ROC curves, we also compute the Area Under the Curve (AUC), which gives the percentage of the whole ROC space that lies underneath an ROC curve. An AUC of 1 indicates perfect performance, while an AUC of .5 indicates performance at chance level.

As a first analysis step, we tried to find optimal parameters k (neighborhood size for algorithms kNN, AH, MP) by comparing AUC results based on values of $k = 1, 2, 3, 5, 10, 20, 30, 40, 50$. Results for GTZAN can be found in Figure 1, for ISMIR in Figure 2. Looking at the GTZAN results, the AUC values (y-axis) for all three methods monotonically decrease with neighborhood size increasing beyond $k = 1$. The only exception is a very small gain in AUC for MP going from $k = 1$ to $k = 2$. Looking at the results for ISMIR in Figure 2, the AUC values again monotonically decrease beyond $k = 1$ for methods kNN and MP. Only for method AH, there is a small and slow rebound starting at about $k = 20$. We therefore conclude that there is no gain in increasing the neighborhood size k beyond 1 for any of the methods. All following results are therefore based on the choice of $k = 1$. We can also see from Figures 1 and 2, that MP improves AUC results over kNN across the whole range of k and for both data sets. It is also already evident that AH is never able to improve performance of kNN.

We now make a detailed AUC analysis separate for all data as well as for hubs, anti-hubs and normal data as defined at the beginning of this section (neighborhood size $k = 1$ for all outlier detection methods). Looking at the results for GTZAN in Figure 3, we can see that for all data together (left most group of bars in figure), MP increases performance to .81 compared to kNN which achieves .70. Method AH actually decreases performance to .67. Looking at hubs and anti-hubs separately for method kNN, it is clear that anti-hub objects perform worst with an AUC of .59 versus .71 for hubs and .73 for the remaining normal data. We see

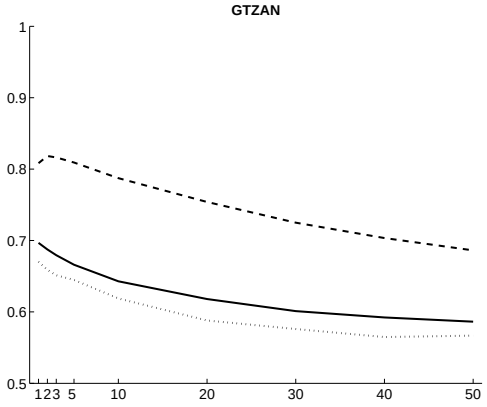


Figure 1: *AUC (y-axis) analysis for data set GTZAN and parameter k ranging from 1,2,3,5,10,20,30,40 to 50 (x-axis), solid line for kNN, dotted for AH, dashed for MP.*

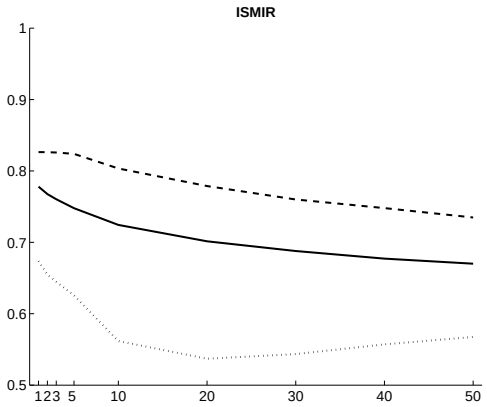


Figure 2: *AUC (y-axis) analysis for data set ISMIR and parameter k ranging from 1,2,3,5,10,20,30,40 to 50 (x-axis), solid line for kNN, dotted for AH, dashed for MP.*

the same pattern of lowest AUC for anti-hubs for method AH also, albeit at an even lower level. Method MP is able to increase AUC compared to kNN for both anti-hubs (from .59 to .71) and normal data (from .73 to .86). There is a small decrease in AUC for hubs (from .71 to .67). Looking at results for ISMIR in Figure 4, we basically see the same pattern. Method kNN performs worst for anti-hubs, normal and hub objects perform at about the same level. Method AH repeats this pattern at a lower level. Method MP is able to improve the performance for anti-hubs and normal objects but not for hubs. Gains in performance compared to kNN are smaller than for GTZAN (e.g. from .78 to .83 for all data objects).

We now present the ROC plots that the above AUC results are based on, again for all data as well as for hubs, anti-hubs and normal data separately (neighborhood size $k = 1$ for all outlier detection methods). Looking at the results for GTZAN in Figure 5, we can see that the ROC curve for method MP (dashed line) is above those for kNN (solid line) and AH (dotted line) for almost the whole ROC space for all data together as well as for anti-hubs and normal objects (sub-plots titled 'ALL', 'ANTI-HUB' and 'NORMAL'). The ROC curves for hub objects are quite comparable for all three methods. It is also interesting to see, that the ROC curve for method AH and anti-hub objects (sub-plot titled 'ANTI-

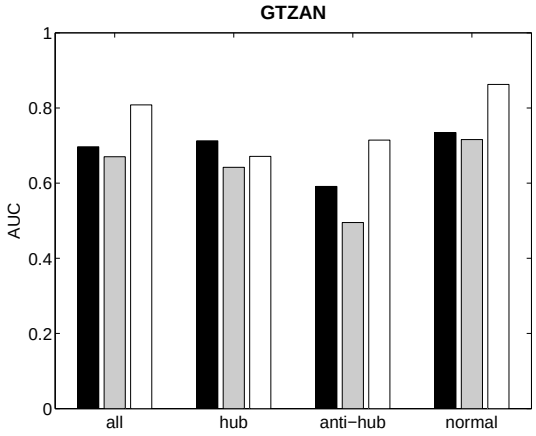


Figure 3: *AUC plot for data set GTZAN, black bars for kNN, grey for AH, white for MP (neighborhood size $k = 1$).*

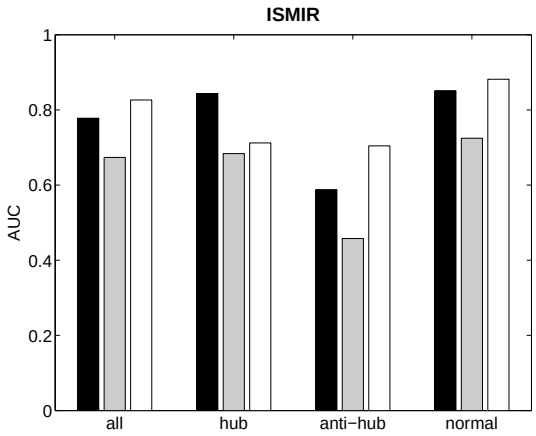


Figure 4: *AUC plot for data set ISMIR, black bars for kNN, grey for AH, white for MP (neighborhood size $k = 1$).*

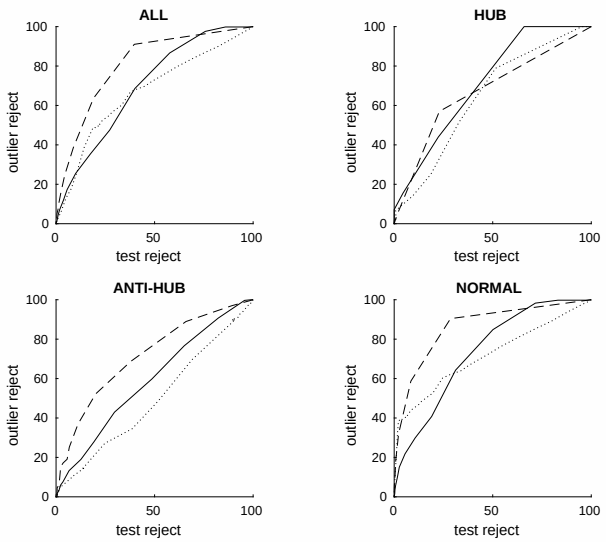


Figure 5: *ROC plots for data set GTZAN, solid line for kNN, dotted for AH, dashed for MP (neighborhood size $k = 1$).*

Table 3: *Accuracy analysis of data set GTZAN, see Sec. 5.*

method	acc	acc_rej	o_rej	t_rej
kNN	77.54	98.90	99.82	85.94
AH	77.54	82.03	76.94	56.77
MP	82.06	94.69	91.02	39.69

Table 4: *Accuracy analysis of data set ISMIR, see Sec. 5.*

method	acc	acc_rej	o_rej	t_rej
kNN	87.11	100.00	100.00	62.76
AH	87.11	93.08	73.72	46.60
MP	91.11	98.04	87.77	28.10

HUB', dotted line) is very close to the main diagonal indicating performance at chance level. This explains the very low AUC of .59 for this curve, with .5 indicating chance level. This is due to the fact, that method AH is based on n-occurrence counts (see Section 4.2), basically detecting everything with a low n-occurrence as outliers. It therefore rejects all anti-hubs as outliers, no matter whether they are true outliers or test data that should not be rejected. Figure 6 gives the ROC plots for ISMIR, repeating the same patterns of behavior we just described, albeit less clearly. ROC curves for method MP more or less dominate those for kNN and AH for all data together as well as anti-hubs and normal data. Method AH performs even below chance level for anti-hubs with an AUC of only .46.

Finally, we present results concerning one-nearest neighbor classification accuracy gains due to the outlier rejection. While steadily lowering threshold value p and rejecting more and more test and outlier data, less and less data is being classified but usually with increased accuracy. The following results are averages across test data not used for training of the classifier and not rejected by the respective outlier detection methods. In Tables 3 and 4 we present for data sets GTZAN and ISMIR the classification accuracy without any outlier rejection (acc), the maximum achieved accuracy after rejection (acc_rej), the percentage of rejected outlier data (o_rej) and the percentage of rejected test data (t_rej) at the respective threshold level. The baseline accuracy acc of using method kNN and not rejecting at all is 77.54% for GTZAN. Please note that this baseline accuracy is already at 82.06% when using distances rescaled via MP, again while not rejecting at all. With outlier rejection this can be improved up to 98.90% with kNN and 94.69% with MP, but only 82.03% with AH. But to reach this maximum accuracy, kNN does not only correctly reject 99.82% of outliers, but also falsely rejects 85.94% of test data. Method MP on the other hand reaches its maximum accuracy while correctly rejecting 91.02% of outliers but only 39.69% of test data. Data set ISMIR shows the same pattern with accuracies improving up to 100.00% and 98.04% for kNN and MP, with AH lagging behind at 93.08%. Again kNN has to reject much more test data than MP to reach these results (62.76% vs. only 28.10%).

6. DISCUSSION

In discussing our results obtained in Section 5, we like to recapitulate our main findings.

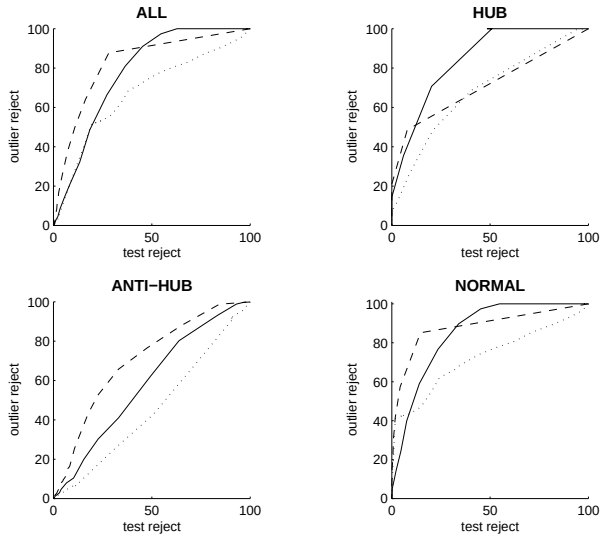


Figure 6: *ROC plots for data set ISMIR, solid line for kNN, dotted for AH, dashed for MP (neighborhood size $k = 1$).*

Our first result is that classic distance-based outlier rejection methods are negatively affected by hubness. As can be seen by looking at the results for distance-based method kNN separately for hubs, anti-hubs and normal data, especially anti-hubs present problems for outlier detection as evident from their lower AUC values. Since anti-hubs per definition are far away from most other data points in a data base, it seems logical that they are being detected as outliers even when they not really are. As for hub objects, we would have expected that they are also responsible for more detection errors than normal data, which is not really the case. In our analysis we only looked at hub objects as being candidates for outlier detection and there are maybe too few hub objects (21 in GTZAN and 24 in ISMIR) in the data sets to gain meaningful statistics. But hub objects are per definition in nearest neighbor lists of many other data objects and through these lists very likely enter computation of outlier scores of many data objects in a data set. It would definitely be very interesting to do another analysis of outlier detection considering this fact.

Our second result is that reverse nearest neighbor information cannot improve distance-based outlier detection that is affected by hubness. On the contrary, our results for method AH show that AUC values compared to kNN even deteriorate. Especially AUC results for anti-hubs are sometimes even below chance level. Given the fact that AH basically detects everything with a low reverse nearest neighbor count as an outlier, this is not surprising. After all this means that anti-hubs, which already are a problem for distance-based methods, are detected as outliers no matter whether they really are or not.

Our third result is that our hubness-aware algorithm MP is able to improve outlier detection and that this improvement is also due to the changed role of anti-hubs. Looking at our results for using MP, we can see that we gain overall improvements in AUC which are especially pronounced for anti-hub and normal data. Mutual proximity has been shown to decisively reduce the negative impact of hubs and anti-hubs and produce distance spaces with much more normal behavior [23]. Rescaling via MP is able to prevent anti-hubs from being far away from most other data points, therefore not acting as 'artificial' outliers anymore.

Given these three main results, it would now of course be very interesting to analyse the behavior of other outlier detection methods when confronted with data affected by hubness. Of special interest are methods that have already been designed for high-dimensional outlier detection like e.g. "angle-based outlier detection" (ABOD) [15].

Our data analysis is based on a model of timbre similarity only, since this is an important part of most MIR systems modeling music similarity. Previous results have shown that many different parametrizations of audio are susceptible to hubness [10], therefore our results are important for MIR models beyond timbre also. But it also clear that there exist models of music similarity that are not prone to hubness, e.g. certain combinations of timbre and rhythm aspects [9]. And it is also clear that there exist other methods to reduce hubness in MIR models, e.g. the usage of Universal Background Models [2], which is a method from speech analysis.

Concerning usage of data sets in our study, repetition with larger data sets and other MIR problems (e.g. tag classification) would of course be interesting. We are also aware of the criticism concerning the GTZAN data set and its faults [25] like e.g. mislabeling. But these problems mainly concern classification results and not so much outlier detection which after all is the main focus of our work here.

Of greater importance is maybe the fact that we do not use artist filters in our analysis. An artist filter [7] prevents songs from the same artist to be both in the training and test set. This is important since songs from the same artist are often very similar and not using an artist filter can lead to over optimistic results, e.g. in terms of genre classification. But as we just said above, this is not our main focus here and very likely this problem affects all three of our methods (kNN, AH and MP) in the same way. But it is also a wellknown effect that songs from the same artist dominate the nearest neighbor lists in audio-based music similarity. We can only speculate whether songs from the same artist are able to even prevent hub songs from entering nearest neighbor lists. If this were the case, usage of an artist filter would definitely change our outlier detection results. Maybe the impact of hubness on outlier detection is even greater if artist filters are being used. Such an analysis is beyond the scope of this paper but definitely interesting future work.

7. CONCLUSIONS

We have presented a first detailed study of the role of hubness in outlier detection for music genre recognition. We found that classic distance-based methods for outlier rejection are negatively impacted by hubness, where especially anti-hubs pose a problem. We also showed that a recently proposed outlier method based on reverse nearest neighbor counts is not able to help in this respect. But a hubness-aware method based on hubness reduction via computation of mutual proximity is able to improve outlier detection results. Improvements concerning the problematic role of anti-hubs are part of the success. Since hubness is due to a general problem of measuring distances in high-dimensional spaces and since many models in music information retrieval have been shown to be affected, our results are of interest beyond the focus on genre recognition in this paper.

8. REFERENCES

[1] Aucouturier, J.-J., Pachet F.: Improving Timbre Similarity: How high is the sky?, Journal of Negative Results in Speech and Audio Sciences, 1(1), 2004.

[2] Charbuillet C., Tardieu D., Peeters G.: GMM supervector for Content Based Music Similarity, Proceedings of the 14th International Conference on Digital Audio Effects (DAFX'11), Paris, France, 2011.

[3] Feldbauer R., Flexer A.: Centering versus Scaling for Hubness Reduction, Proceedings of the 25th International Conference on Artificial Neural Networks (ICANN'16), Barcelona, Spain, 2016.

[4] Flexer A.: Improving visualization of high-dimensional music similarity spaces, 16th International Society for Music Information Retrieval Conference, Malaga, Spain, 2015.

[5] Flexer A.: The impact of hubness on music recommendation, Machine Learning for Music Discovery Workshop at the 32nd International Conference on Machine Learning, Lille, France, 2015.

[6] Flexer A., Pampalk E., Widmer G.: Novelty detection based on spectral similarity of songs, in Proc. of the 6th Int. Conf. on Music Information Retrieval, 2005.

[7] Flexer A., Schnitzer D.: Effects of Album and Artist Filters in Audio Similarity Computed for Very Large Music Databases, Computer Music Journal, Volume 34, Number 3, pp. 20-28, 2010.

[8] Flexer A., Schnitzer D.: Using mutual proximity for novelty detection in audio music similarity, in Proceedings of the 6th International Workshop on Machine Learning and Music, Prague, Czech Republic, 2013.

[9] Flexer A., Schnitzer D., Gasser M., Pohle T.: Combining features reduces hubness in audio similarity, Proceedings of the Eleventh International Society for Music Information Retrieval Conference, 2010.

[10] Flexer A., Schnitzer D., Schlüter J.: A MIREX meta-analysis of hubness in audio music similarity, Proceedings of the 13th International Society for Music Information Retrieval Conference, 2012.

[11] Francois D., Wertz V., Verleysen M.: The concentration of fractional distances, IEEE Transactions on Knowledge and Data Engineering, 19:873-886, 2007.

[12] Hansen L.K., Lehn-Schiøler T., Petersen K.B., Arenas-Garcia J., Larsen J., Jensen S.H.: Learning and clean-up in a large scale music database, in Proc. of the European Signal Processing Conference (EUSIPCO), pp. 946-950, IEEE, 2007.

[13] Hautamäki V., Kärkkäinen I., Fränti P.: Outlier Detection Using k-Nearest Neighbour Graph, 17th International Conference on Pattern Recognition, pp. 430-433, 2004.

[14] Karydis I., Radovanović M., Nanopoulos A., Ivanović M.: Looking Through the "Glass Ceiling": A Conceptual Framework for the Problems of Spectral Similarity, in Proc. of the 11th Int. Conf. on Music Information Retrieval, pp. 267-272, 2010.

[15] Kriegel H. P., Schubert M., Zimek A.: Angle-based outlier detection in high-dimensional data, Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, pp. 444-452, 2008.

[16] Levina E., Bickel P.J.: Maximum likelihood estimation of intrinsic dimension, in Advances in Neural Information Processing Systems 17, pp. 777-784, 2005.

[17] Mandel M., Ellis D.: Song-level features and support vector machines for music classification, in Proc. of the 6th Int. Conf. on Music Information Retrieval, 2005.

[18] Pimentel M.A.F., Clifton D.A., Clifton L., Tarassenko L.: A review of novelty detection, Signal Processing, Vol. 99, pp. 215-249, 2014.

[19] Radovanović M., Nanopoulos A., Ivanović M.: Reverse nearest neighbors in unsupervised distance-based outlier detection, IEEE Transactions on Knowledge and Data Engineering, 27(5), 1369-1382, 2015.

[20] Radovanović M., Nanopoulos A., Ivanović M.: Hubs in space: Popular nearest neighbors in high-dimensional data, Journal of Machine Learning Research, 11:2487-2531, 2010.

[21] Ramaswamy S., Rastogi R., Shim K.: Efficient algorithms for mining outliers from large data sets, Proceedings of the 2000 ACM SIGMOD international conference on Management of data (SIGMOD '00), pp. 427-438, 2000.

[22] Schnitzer D., Flexer A.: The Unbalancing Effect of Hubs on K-medoids Clustering in High-Dimensional Spaces, Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2015.

[23] Schnitzer D., Flexer A., Schedl M., Widmer G.: Local and Global Scaling Reduce Hubs in Space, Journal of Machine Learning Research, 13(Oct):2871-2902, 2012.

[24] Schnitzer D., Flexer A., Tomašev N.: A Case for Hubness Removal in High-Dimensional Multimedia Retrieval, Proceedings of the 36th European Conference on Information Retrieval (ECIR), 2014.

[25] Sturm, B: An analysis of the GTZAN music genre dataset, Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies, ACM, 2012.

[26] Sturm B.L.: Music genre recognition with risk and rejection, in Proceedings of International Conference on Multimedia and Expo, 2013.

[27] Tomašev N., Radovanović M., Mladenić D., Ivanović M.: The Role of Hubness in Clustering High-dimensional Data, IEEE Transactions on Knowledge and Data Engineering, Volume 26, Issue 3, 2013.

[28] Tzanetakis G., Cook P.: Musical genre classification of audio signals, IEEE Transactions on Speech and Audio Processing, Vol. 10, Issue 5, 293-302, 2002.

[29] Zimek A., Schubert E., Kriegel H.-P.: A survey on unsupervised outlier detection in high-dimensional numerical data, Statistical Analysis and Data Mining, 5: 363-387, 2012.

SEPARATING PIANO RECORDINGS INTO NOTE EVENTS USING A PARAMETRIC IMITATION APPROACH

Wen Xue
Samsung Electronics

xue.wen@samsung.com

ABSTRACT

In this paper we present a working system for separating a piano recording into events representing individual piano notes. Each note is parameterized with a transient-plus-harmonics model that, should all the parameters be reliably estimated, would produce near perfect reconstruction for each note as well as for the whole recording. However, interference between overlapping notes makes it hard to estimate parameters from their combination. In this work we propose to assess the estimability of sinusoidal parameters via their apparent degree of interference, estimate the estimable ones using algorithms suitable for different interference situations, and infer the hard-to-estimate parameters from the estimated ones. The outcome is a sequence of separate, parameterized piano notes that perceptually highly resemble, if are not identical to, the notes in the original recording. This allows for later analysis and processing stages using algorithms designed for separate notes.

1. INTRODUCTION

Musical note separation is the task of breaking audio-based musical content into separate pieces of audio, each corresponding to a musical note in the original content, as if recorded separately. By allowing access to individual notes, note separation has immense potential in the production, maintenance and consumption of recorded music. Unfortunately, high-quality fully-automatic audio-based note separation remains hard. More feasible alternatives have been proposed to address the task. For example, automatic score-informed separation, e.g. [1][2], uses the musical score to guide the separation process. Supervised separation, e.g. [3][4], engages human power to provide more detailed and reliable information that the separation module may benefit from.

In this paper we propose another supervised note separator configured for real-world piano recording. Among musical instruments the piano is known to produce note sounds that are more predictable and less volatile therefore easier to describe and measure. Despite this we still have two points to consider before we can separate the complete set of notes from a recording. First, note separation implicitly includes music transcription, which is still an open problem itself; second, piano music is largely polyphonic, and it is common to have severe interference between concurrent notes, which makes clean separation difficult.

Our answer (or concession) to the first point is an interactive, supervised process that relies on a human participant to make decisions and correct errors. For the second point we propose an automated method that makes measurements where interference is low, and guesses where interference is high. Since some parts of the notes are inferred rather than measured, the method does not qualify for signal separation in the strict sense. We call it a parametric imitation approach, as it imitates an original note with incomplete measurements. By not measuring the hard-to-measure parts of the signal, this scheme minimizes the risk of unstable estimates from high-interference zones. This allows resynthesis of notes that 1) resemble the original ones in loudness, pitch, timbre and dynamics, and 2) sound convincing by themselves.

Our note separator works in four stages:

1. transcription, for finding out what notes are in the music and decide their timing and component frequencies;
2. interference classification, for finding high, mid and low interfering zones in the time-frequency (T-F) plane;
3. stationary component estimation, for estimating note parameters in low and mid interference zones and guessing them in high interference zones;
4. transient extraction and note reconstruction.

Stage 1 is semi-automatic with limited human participation; the rest are fully automatic.

The rest of this paper is arranged as follows. Section 2 describes the signal model we use to represent piano notes. Section 3 describes the user interface in the transcription stage. Sections 4, 5, 6 and 7 present the algorithms in the four stages above, respectively. Section 8 includes experimental result that highlights our performance for interfering notes. Future improvements are discussed in section 9.

2. MODEL AND ASSUMPTIONS

For underlying signal model we use a transient-plus-harmonics model similar to [5]. The transient part models the attack of a note; the harmonic part models the pitched stationary resonance. We assume that piano notes have constant pitch after the initial attack, so the harmonics part $x(t)$ of a note can be written as

$$x(t) = \sum_{m=1}^{M} a^m(t)\cos(2\pi f^m t + \varphi^m) \tag{1}$$

where M is the number of component sinusoids (partials). $a^1(t)$, …, $a^M(t)$ are the amplitudes of the sinusoids and f^1, …, f^M are their frequencies. Functions $a^1(t)$, …, $a^M(t)$ are constrained to vary slowly with t. In this paper we use superscripts for partial indices. To distinguish them from exponents, we write the latter with brackets on the base, like $(m)^2$ or $|X|^2$.

Many pitched instruments have all partial frequencies determined by the fundamental frequency via a harmonicity (or equivalently, "inharmonicity") model:

$$f^m = f^m(m, \mathrm{F0}; \bullet) \tag{2}$$

where m is the partial index, F0 is the fundamental frequency and $\bullet$ represents optional parameters. For the piano we choose the stiff string model in [6], plus to a small additional shift:

$$f^m(m, \mathrm{F0}; B) = m \cdot \mathrm{F0}\sqrt{1 + B\left((m)^2 - 1\right)} + \varepsilon^m \quad (3)$$

where B is a small positive number representing string stiffness, and ε^m covers inharmonicity due to other factors. In section 4 we estimate F0 and B by minimizing these ε^m's.

While the vibrating modes of an ideal wave-radiating string are characterized by exponentially decaying amplitudes, such is not suitable for modelling partial amplitudes in (1). This is because modern pianos use 2-string and 3-string notes, producing amplitude modulation typical of beating sinusoids. This prevents us from using parametric method like ESPRIT [7] for estimation. Fortunately, in the low to mid frequency range where most energy lies, this amplitude modulation is usually slow enough to be effectively captured with a uniformly-sampled sinusoidal representation like [5]. Given the complexity of real-world recordings, we make no special assumption for measuring amplitude parameters except that they vary slowly with time, and that they decay in the long term.

In this paper we evaluate all parameters from the short-time Fourier transform (STFT) of the recording, computed using a Hann window 2048 points long applied to waveform data sampled at 44.1kHz. Adjacent windows overlap by 50%, which makes the hop size 23.2 milliseconds. The parameter set for each note includes the position of starting frame (1), length in frames L (1), number of partials M (1), frequency of each partial (M), initial phase angle of each partial (M), amplitude of each partial at each frame (LM) and transient spectrum (2048).

3. USER INTERACTION IN SUPERVISED TRANSCRIPTION AND FREQUENCY ESTIMATION

The goal of the transcription stage (stage 1) is to find out what notes are in the recording, where they begin and end, and what their partial frequencies are. While note separation necessarily includes music transcription as subtask, this paper is not about automatic transcription. Instead, we follow the supervised path and involve a human user, the *supervisor*, who provides information hard to retrieve reliably using present automatic techniques. More specifically, in this paper the user's job is to help the computer locate harmonic partials of each note in the T-F plane using the spectrogram. To reduce his workload it is recommended that an automatic transcription system like [8]-[11] be used as front end to provide initial guesses of existing notes and their whereabouts. The proposed workflow does not tell if the initial guess comes from a human user or an automatic transcriber. Subsequent steps will require user participation to clean up any mistakes previously made.

The workflow of our note separator is shown in Figure 1. The shaded blocks are automated modules and the clear ones need user attention. The block labelled "supervisor input" may also include an automatic transcription front end, if there is one.

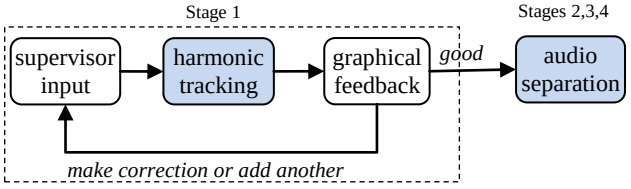


Figure 1 Note separation workflow

The core module of stage 1 is an automatic harmonics tracker (section 4) that locates all harmonics of a note in the T-F plane. For each note, the tracker takes one or more *seed* points as input. A seed point comprises a time-frequency-partial_index triple (t, f, m), meaning that the m^{th} partial of the note has frequency near f at time t. The initial seed point can be provided either by an automatic front end or by the human user. To supply the seed without automatic transcription, a graphical user interface with an image of the spectrogram is presented to the user. The user identifies a note picking a partial index m then pointer-clicking on an unambiguous point of the m^{th} partial in the image. The partial index and the coordinates of the pointer click make up the seed point, with which the automatic tracker is launched. The tracker tolerates no less than two bins of input frequency inaccuracy, so that the average user can supply seed points with comfort.

The tracking result is fed back to the user on the same user interface, with note duration and partial frequencies plotted as frequency trails on the spectrogram, like in Figure 1. The hollow star in the figure marks the initial guess where the user has seeded the harmonics tracker. The user can judge if the tracking has been successful by comparing the trails against the spectrogram.

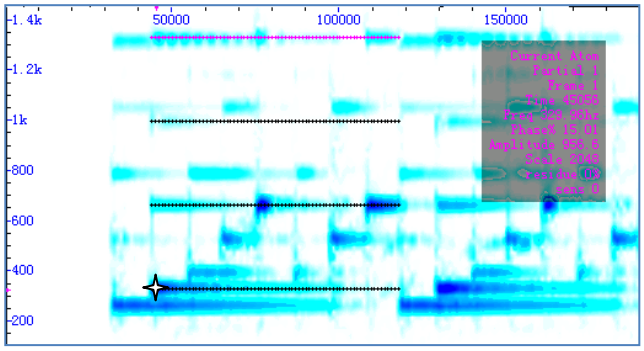


Figure 2 Selecting a note on the spectrogram

Three types of mistakes may occur during automatic tracking: wrong partial frequency, wrong start position and wrong end position. Start and end position errors can be corrected by the user fixing them directly on the spectrogram. Frequency mistakes typically involve frequency estimates being associated with harmonics from other notes. The user corrects them by adding one more seed point, which strengthens the harmonicity constraint and forces estimates off the wrong sinusoids. Our user interface allows the user to drag a frequency trail to another position using a pointer device, upon which the harmonics tracker is relaunched with an additional seed point at the new position. Each error correction requires one pointer clicking or dragging operation. In our experiment a prominent piano note takes no more than 2 operations to mark (including the initial guess), while a heavily masked note usually takes 3 or 4. This track-and-correct procedure is repeated for all notes we intend to separate, plus all their concurrent notes. The rest is left to the note separation module that does not need supervision.

4. HARMONIC FREQUENCY TRACKING FOR CONSTANT PITCHES

This section discusses the harmonic tracker in the supervised transcription stage (stage 1). Given one or more seed points that identify a note, its task is to automatically find out all partial frequencies of the note, along with its start and end positions.

In standard sinusoidal modelling [12]-[14], the frequency of a sinusoid is measured independently from spectral peaks at individual frames. Such measurement is easily corrupted by noise or interference from other sources. Luckily the constant-frequency and harmonicity constraints allow us to ignore data from areas in the T-F plane where such corruption is high, and use only the less corrupted parts for estimating frequencies.

Our constant-pitch harmonics tracker uses plain spectrogram for audio input. From the spectrogram a collection of spectral *atoms* are obtained using standard peak picking. Given the presence of noise and interference neither constant-frequency tracks nor harmonic atom groups are guaranteed to emerge from these raw atoms. However, it is plausible to assume that at least *some* of the atoms are relatively less corrupted. These atoms should assume spectral shape typical to constant-frequency sinusoids; their frequency estimates should be accurate and satisfy constant-frequency and harmonicity constraints. Our harmonics tracker uses peak shape and the frequency constraints to look for the "good" atoms and use them for frequency estimation. We call them the *core* atoms.

4.1. Peak shape

All spectral atoms identified by peak picking are located at spectral peaks. Given an atom with frequency estimate *f*, in bins, we evaluate the peak shape by its cross-correlation with that of a pure sinusoid at *f*, i.e.

$$\lambda_{\mathrm{p}} = \frac{\sum_k X_k W^*(k-f)}{\left(\sum_k |X_k|^2\right)^{1/2}\left(\sum_k |W(k-f)|^2\right)^{1/2}} \tag{4}$$

in which X_k is the STFT of the signal x at bin k, and $W(f)$ is the discrete-time Fourier transform of the window function used for spectrogram computation. The summations are done over a few bins near f. The value of $|\lambda_p|$ lies on [0, 1] (Cauchy-Schwarz). The higher is $|\lambda_{\mathrm{p}}|$, the closer the atom spectrum resembles that of a sinusoid. The harmonic tracker uses $|\lambda_{\mathrm{p}}|$ to screen spectral atoms as candidates for the core set: only those with $|\lambda_{\mathrm{p}}|$ value higher than a threshold (e.g. 0.9) may become a core atom.

4.2. Constant-frequency constraint

Given a set of L atoms detected from L different frames with frequency estimates $f_0, \ldots, f_{L-1}$, we measure how constant these estimates are by their average deviation from a presumed frequency $\hat{f}$:

$$\lambda_{\mathrm{f}} = \frac{1}{L}\sum_{l=0}^{L-1}\left(D(f_l, \hat{f})\right)^2 . \tag{5}$$

where D is a distance function. Eq. (5) cannot be evaluated without knowing $\hat{f}$. We select both $D(\cdot)$ and $\hat{f}$ to fit into a harmonicity model so that the constant-frequency and harmonicity constraints are combined in one, as described below.

4.3. Harmonicity constraint

For the piano we choose the model given in [6] for stiff strings:

$$\hat{f}^{\,m}(m, \mathrm{F0}, B) = m \cdot \mathrm{F0}\sqrt{1 + B\left((m)^2 - 1\right)} . \tag{6}$$

where B is a small positive number. Given a set of I atoms with frequency estimates $f^{m_0}, \ldots, f^{m_{I-1}}$, m_i being the assumed partial index of the i^{th} atom, we measure their harmonic consistency by their deviation from the harmonicity model:

$$\lambda_{\mathrm{f}} = \frac{1}{I}\sum_{i=0}^{I-1}\left(D(f^{m_i}, \hat{f}^{\,m_i}(\mathrm{F0}, B))\right)^2 \tag{7}$$

The smaller is λ_{f}, the more likely are the atoms to belong to the same note. Evaluating (7) requires knowing the values F0 and B, which we choose by minimizing λ_{f}, as follows.

Eq. (6) is nonlinear regarding F0 and B. We linearized it by taking $F=\mathrm{F0}^2$ and $G=FB$ (also see [14]) as

$$(\hat{f}^{\,m})^2 = (m)^2 F + (m)^2\left((m)^2 - 1\right)G \tag{8}$$

Eq. (8) is linear regarding F and G. We choose the distance function $D(\cdot)$ as:

$$D\left(f^{m_i}, \hat{f}^{\,m_i}\right) = \frac{1}{m_i}\left((f^{m_i})^2 - (\hat{f}^{\,m_i})^2\right) \tag{9}$$

where the multiplier $1/m$ removes extra emphasis put onto high frequencies by the squaring. Substitute (8) and (9) in (7) we get

$$\lambda_{\mathrm{f}} = \frac{1}{I}\sum_{i=0}^{I-1}\frac{1}{(m_i)^2}\left((f^{m_i})^2 - (m_i)^2 F - (m_i)^2((m_i)^2 - 1)G\right)^2 . \tag{10}$$

We find F and G that minimize λ_{f} with

$$F = \frac{\sum (f^{m_i})^2 \sum m_i^2(m_i^2-1)^2 - \sum (f^{m_i})^2 (m_i^2-1)\sum m_i^2(m_i^2-1)}{\sum m_i^2 \sum m_i^2(m_i^2-1)^2 - \sum m_i^2(m_i^2-1)\sum m_i^2(m_i^2-1)},$$

$$G = \frac{\sum m_i^2 \sum (f^{m_i})^2 (m_i^2-1) - \sum (f^{m_i})^2 \sum m_i^2(m_i^2-1)}{\sum m_i^2 \sum m_i^2(m_i^2-1)^2 - \sum m_i^2(m_i^2-1)\sum m_i^2(m_i^2-1)} . \tag{11}$$

Using these values we are able to evaluate λ_{f} by (10). Notice there is no constraint on time or partial index associated with each atom involved, except that they cannot all have the same partial index. If (10) is applied to multiple frames, λ_{f} measures frequency consistency both across time and across partials.

The harmonics tracker does not use λ_{f} directly, but uses F and G instead for screening spectral atoms as candidates for the core set, as detailed below.

4.4. Frequency range of eligible atoms given other atoms

The harmonics tracker needs a set of core atoms from multiple partials and frames to estimate the frequencies. We construct this core atom set by incrementally including new atoms as the tracking progresses. The frequency of a newly incorporated core atom should be consistent with existing core atoms. One way to evaluate this consistency is to estimate F and G from the current core set using (11), then predict the frequency $\hat{f}^{\,m}$ for any partial index m using (6). Only atoms within a δ-vicinity of $\hat{f}^{\,m}$, i.e. ($\hat{f}^{\,m}$ - δ , $\hat{f}^{\,m}$ +δ), as considered eligible candidates for core atoms of the m^{th} partial. Since core atoms are assumed to have accurate frequency estimates, we choose a relatively small vicinity, e.g. δ=0.1 bins.

To be able to estimate F and G above with (11), we must already have at least two core atoms with different partial indices. In the case only one, say f^m, is available, we can determine the eligible range for another partial index by fixing an upper bound for B, say B_{M}. The eligible range for F0 then becomes:

$$\frac{f^m - \delta}{m\sqrt{1 + B_{\mathrm{M}}\left((m)^2 - 1\right)}} < \mathrm{F0} < \frac{f^m + \delta}{m} . \tag{12}$$

This gives an eligible range for the n^{th} partial as

$$\frac{n}{m}\frac{f^m-\delta}{\sqrt{1+B_M\left((m)^2-1\right)}}-\delta<f^n<\frac{n}{m}(f^m+\delta)\sqrt{1+B_M\left((n)^2-1\right)}+\delta\,. \quad (13)$$

4.5. The tracking algorithm

The constant-pitch harmonics tracker proceeds frame-by-frame, from the seed point forwards and backwards until an endpoint is met. We require that the starting point be an actual atom detected with high λ_p, e.g. above 0.9. The tracking algorithm maintains a core atom set **C**, which is initiated as empty. The eligible frequency range for the first atom (when **C** is empty) is defined as a few bins around the seed point.

In each tracking step the tracker moves one frame forward or backward, finds new core atoms from the new frame, and removes existing core atoms that appear no longer good enough. Given the current core atom set **C**, a single-frame tracking step proceeds as follows.

Within the current frame, do
1 ºfind the eligible ranges for all partials consistent with **C**;
2 ºfind all atoms with high λ_p within these eligible ranges, let this set of new atoms be $\mathbf{C}_{\text{new}}$;
3 ºif there are multiple atoms found for any partial index, do
4 %~5 º
4 º use $\mathbf{C}\cup\mathbf{C}_{\text{new}}$ to predict the frequency for that partial index;
5 º keep the atom closest to the predicted frequency in $\mathbf{C}_{\text{new}}$ and remove competing atoms;

With the current set of core atoms, do
6 ºuse $\mathbf{C}\cup\mathbf{C}_{\text{new}}$ to predict the frequencies for all atoms in the core set, remove those that deviate more than δ from the prediction.

From the seeding frame l, forward tracking repeats this step for frames l, l+1, l+2,…, until the number of consistent atoms found falls below a threshold level for three consecutive frames. We currently set the threshold at a fifth of the total number of partials for the relevant pitch, which is computed with (8). Similarly, backward tracking repeats the step for frames l-1, l-2, …, until an end point is met. The tracking algorithm returns the event duration and all partial frequencies estimated from the final core atom set by partial-wise average, weighted by the atom energy. If not enough core atoms are available to compute the average for some partial, its frequency is computed with (11).

While the previous tracking algorithms [12]-[14] also estimate amplitudes and phase angles, doing so without considering interference between concurrent notes leads to faulty results. In this paper we address the interference using what we call the collision regions. We explain them in the next section.

5. COLLISION REGIONS

This section discusses the interference classification stage (stage 2), whose goal is to provide detailed description on the interference between sinusoidal partials, so that the subsequent stage can use this information for estimating amplitudes.

Spectral interference occurs if partials of concurrent notes have very close frequencies. To properly address interference we want to know exactly what partials have what level of interference during which time interval. Each such group of partials has its own characteristics concerning interference and are best treated with an estimator tuned to that special occasion. As interference is caused by concurrency in time and proximity in frequency, such interfering partial groups occupy localized regions in the time-frequency plane.

5.1. Colliding sinusoids

We say two frequencies *collide* if they are closer than a reference threshold δ_f from each other, so that the presence of one may compromise the estimation of the other. The value of δ_f is related to the frequency resolution of the estimator. As we measure sinusoids from STFT, a convenient choice is a fixed value of δ_f in DFT bins. We say two sinusoids collide if their frequencies collide.

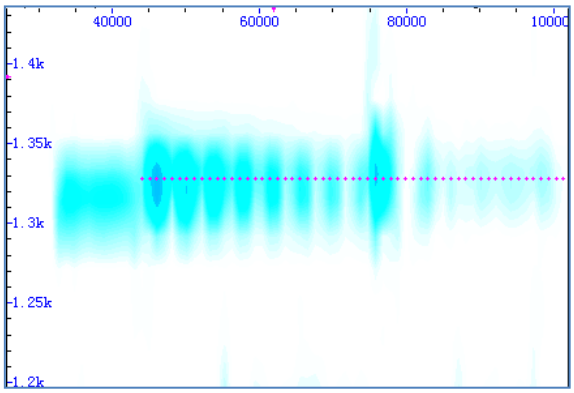


Figure 3 Detail of Figure 2: colliding partials

Figure 3 shows the spectral detail of the 1st half of the signal in Figure 2 near 1.3kHz. At the beginning we have the first note (C4=262Hz) starting around time 32800 (at 44.1kHz sample rate). Its 5th partial has frequency measured at 1316.6Hz. Then at time 44000 the second note (E4) enters. Its 4th partial frequency is 1328.1Hz, which is very close to the 5th partial of the first note. The 11.5Hz difference in their frequencies is too small to be told by the spectrogram, computed with window size 2048 (46.4ms). Consequently we observe a distinctive interference pattern in the central part of Figure 3. Such beating pattern is a typical sign of colliding partials. Since components from different notes cancel one another from time to time, nonnegativity-based methods (e.g. [1]-[3]) cannot handle the case without redesign. It is for the estimation of sinusoids from such spectrograms that we propose the idea of collision regions.

Given a set of constant-frequency sinusoids with known durations and frequencies, it is trivial to determine whether and when any two of them collide. Additional complication arises as more than two sinusoids lie close to each other, all starting and finishing at different times. To describe sinusoid collisions in an organized way suitable for algorithmic handling, we cut the T-F plane into rectangular tiles called collision regions, each containing a number of colliding sinusoids from start to end.

5.2. Definition

We define a *collision region* (CR) of a set **S** of sinusoids as a rectangular area in the time-frequency plane, described by a pair of coordinates (t_1, f_1) and (t_2, f_2), so that:

a) $\forall s \in \mathbf{S}$ have durations containing $[t_1, t_2]$ and frequencies within (f_1, f_2);

b) $f \in (f_1, f_2)$ if and only if there is $s \in \mathbf{S}$ so that f and s collide.

It follows that for every $s \in \mathbf{S}$ there is at least one other $s' \in \mathbf{S}$ that collides with s over $[t_1, t_2]$. We denote a collision region as CR:($\mathbf{S}$; t_1, f_1, t_2, f_2), or simplified as CR($\mathbf{S}$).

A sinusoid $s' \notin \mathbf{S}$ is said to collide with CR:($\mathbf{S}$; t_1, f_1, t_2, f_2) if it collides with any $s \in \mathbf{S}$ at any $t \in [t_1, t_2]$. A collision region of $\mathbf{S}$ is said to be *closed* if no sinusoid outside $\mathbf{S}$ collides with it. Sinusoids in a closed CR($\mathbf{S}$; t_1, f_1, t_2, f_2) are considered free from interference from sinusoids outside $\mathbf{S}$ during $[t_1, t_2]$. If the CR contains multiple sinusoids, their mutual interference should be considered for estimating their parameters during $[t_1, t_2]$.

For example, an isolated sinusoid s of duration $[t_1, t_2]$ and frequency f has its own trivial CR:({s}; t_1, f-δ_f, t_2, f+δ_f) which is also closed. Two colliding sinusoids s_1 and s_2 of duration [0,2] and [1,3] have 3 closed CRs, on intervals [0,1], [1,2] and [2,3], respectively.

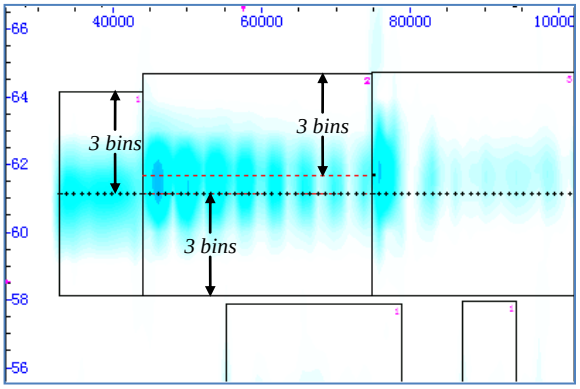


Figure 4 Collision regions of Figure 3

Figure 4 shows the collision regions marked out for the T-F range of Figure 3 with δ_f=3 bins. In Figure 4 we have marked the frequency axis in bins. Between time 32800 and 44000 the 5th partial of the first note is free from interference, therefore it occupies its own CR with total bandwidth of 6 bins, 3 (i.e. δ_f) on each side. Between 32800 and 74800 we have two partials 0.54bin apart. Since 0.54< 3, these two partials are put into the same CR, whose bandwidth becomes 6.54 bins. At 74800 the fifth note starts whose 2nd partial almost overlaps the 4th partial of the second note. This establishes a new CR that contains 3 partials.

5.3. Finding collision regions

Given a set of sinusoids $\mathbf{S}$, we look for a complete set of closed CRs to cover $\mathbf{S}$ using an iterative process. Starting from an arbitrary sinusoid $s_0 \in \mathbf{S}$ with its trivial CR, we add other sinusoids s_1, s_2, …, one at a time into the picture. Each time a new sinusoid s_i is added, we update the set of CRs so that 1) every CR is still closed and 2) the whole CR set covers the new sinusoid as well as the previous ones.

5.3.1. *Collision table*

The update process relies on a data structure we call a *collision table*. Given a closed CR set $\mathbf{C}$ covering sinusoid set $\mathbf{S}$ and a new sinusoid $s \notin \mathbf{S}$, the collision table tells which CRs in $\mathbf{C}$ collide with s at what time. To be more specific, the table $\mathbf{T}(s,\mathbf{C})$ contains a sequence of non-overlapping intervals that together cover the duration of s, so that on each of these intervals s collide with a different combination of 0, 1 or 2 collision regions in $\mathbf{C}$.[1] The collision table is generated by finding all CRs in $\mathbf{C}$ which collide with s, segmenting the duration of s at their boundaries, and enumerating the 0, 1 or 2 CRs that collide with s over each segment.

5.3.2. *Updating the complete CR set*

Let $\mathbf{C}$ be a closed CR set covering $\mathbf{S}_{i-1}$, and $\mathbf{T}(s_i,\mathbf{C})$ be the collision table computed for the next sinusoid s_i. The following routine updates $\mathbf{C}$ to a closed CR set covering $\mathbf{S}_i = \mathbf{S}_{i-1} \cup \{s_i\}$.

1º let $\tau_1 = (t_1, \cdot)$ be the first segment in $\mathbf{T}$, for all CRs c:($\mathbf{S}_c$; t_{c1}, f_{c1}, t_{c2}, f_{c2})$\in\mathbf{C}$ that collide with s_i on τ_1, do 2 º;
  2 ºif $t_{c1} < t_1$, add new CR:($\mathbf{S}_c$; t_{c1}, f_{c1}, t_1, f_{c2}) to $\mathbf{C}$;
3 ºlet $\tau_{-1} = (\cdot, t_2) \in \mathbf{T}$ be the last segment in $\mathbf{T}$, for all CRs c:($\mathbf{S}_c$; t_{c1}, f_{c1}, t_{c2}, f_{c2})$\in\mathbf{C}$ that collide with s_i on τ_2, do 4 º;
  4 ºif $t_{c2} > t_2$, then add new CR:($\mathbf{S}_c$; t_2, f_{c1}, t_{c2}, f_{c2}) to $\mathbf{C}$;
5 ºfor every $\tau = (t_{\tau 1}, t_{\tau 2})$ in $\mathbf{T}$, there are 0, 1, or 2 CRs in $\mathbf{C}$ colliding s_i on τ, do one of 6 º, 7 ºor 8 ºin each case;
  6 ºif no CR in $\mathbf{C}$ collides with s_i on τ, add a new CR:({s}; $t_{\tau 1}$, f_s-δ_f, $t_{\tau 2}$, f_s+δ_f) to $\mathbf{C}$;
  7 ºif one CR:($\mathbf{S}_c$; t_{c1}, f_{c1}, t_{c2}, f_{c2})$\in\mathbf{C}$ collides with s_i on τ, then replace it with CR:($\mathbf{S}_c$+{s_i}; $t_{\tau 1}$, min(f_{c1}, f_s-δ_f), $t_{\tau 2}$, max(f_{c2}, f_s+δ_f));
  8 ºif two CRs:($\mathbf{S}_{c1}$; t_{c11}, f_{c11}, t_{c12}, f_{c12}), ($\mathbf{S}_{21}$; t_{c21}, f_{c21}, t_{c22}, f_{c22})$\in\mathbf{C}$ collide with s_i on τ, then replace them with one CR:($\mathbf{S}_{c1}$+$\mathbf{S}_{c2}$+{s}; $t_{\tau 1}$, min(f_{c11}, f_{c21}), $t_{\tau 2}$, max(f_{c12}, f_{c22})).

Once this update has been performed for all partials of all notes, we have the complete set of collision regions ready. Figure 5 shows the CRs found for the signal in Figure 2. While CRs may overlap themselves, no CR overlaps any sinusoidal partial inside another CR.

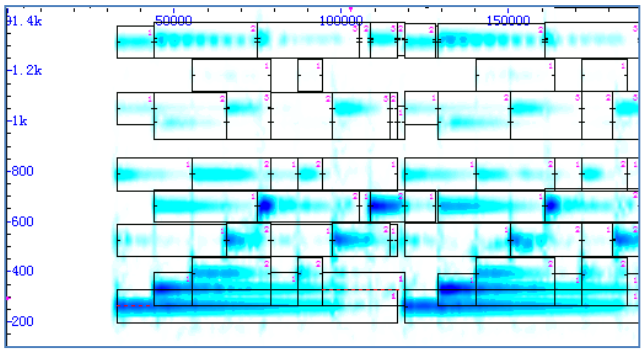


Figure 5 Collision regions identified for Figure 1

6. AMPLITUDE EVALUATION

This section discusses the stationary component estimation stage (stage 3) whose goal, in the current implementation, is to estimate sinusoidal amplitudes based on pre-detected interference details. Estimation of amplitude, like that of frequency, can be corrupted by noise and interference. For the amplitude we adopt

[1] s cannot collide with more than two CRs in $\mathbf{C}$ at any time, because three CRs colliding with s must have collisions among their member sinusoids, so they cannot be all closed.

a similar strategy as we did for frequency, i.e. we measure them directly from data only at "good" atoms. However, since amplitudes are not assumed to satisfy strict constraints as the frequencies do, it is crucial we obtain as many direct estimates as we can to describe amplitudes closely. Assuming that sinusoids within one closed CR do not suffer interference from outside the CR, we do amplitude estimation on a CR-by-CR basis.

6.1. Isolate partials

The simplest CRs are those containing only one sinusoid. Atoms in these CRs are considered free from interference, so can be estimated using any algorithm for estimating solo sinusoid. In this work we use standard orthogonal projection of the spectrum:

$$\lambda = ae^{j\varphi} = \frac{\mathbf{w}^{\mathrm{H}}\mathbf{x}}{\|\mathbf{w}\|^2} \tag{14}$$

where $\mathbf{w}$ is the spectrum of a pure windowed sinusoid at the estimated partial frequency with zero phase, and a and φ are the amplitude and phase angle estimates.

6.2. Least square estimator

Tolonen [15] proposed to use the least square method for estimating "colliding" sinusoids, of which orthogonal projection can be regarded as a special case. For a set of given frequencies, the least square method estimates the amplitudes and phase angles at each frame by solving a linear system involving all the sinusoids. To apply this method we need to know what frequencies are colliding at which frames, which is conveniently handled by collision regions.

For each frame of a closed CR spanning N bins and containing M sinusoids, $N>M$, a linear system of size M is constructed using the spectral data from these N bins. To be more specific, it takes the form of

$$\mathbf{W}^{\mathrm{H}}\mathbf{W}\boldsymbol{\lambda} = \mathbf{W}^{\mathrm{H}}\mathbf{x} \tag{15}$$

$\mathbf{W}$ is an $N\times M$ matrix whose M columns are spectra of pure windowed sinusoids at the M frequencies and zero phase, truncated to contain only the N bins of the CR. $\boldsymbol{\lambda}=(\lambda_1, \ldots, \lambda_M)^{\mathrm{T}}$, $\lambda_m=a_m e^{\mathrm{j}\varphi_m}$, encodes the amplitude and phase angle of the m^{th} sinusoid involved, and $\mathbf{x}$ is the spectrum of this frame, also truncated to those N bins. Noticing that $\mathbf{W}$ remains the same for all frames of a CR, we compute $\mathbf{U}_{M\times N}=[u_{ij}]=(\mathbf{W}^{\mathrm{H}}\mathbf{W})^{-1}\mathbf{W}^{\mathrm{H}}$, then use

$$\boldsymbol{\lambda} = \mathbf{U}\mathbf{x} \tag{16}$$

to compute $\boldsymbol{\lambda}$ for all these frames.

6.3. Exceptions

The least square method assumes that a *good* estimate must yield a *small* square error so as not to produce a large residue. Intuitively, this small square error must be close to the least square error, and ideally we may hope this good estimate also be close to the least square estimate. However, in the note separation task there are two exceptions to such reasoning: that the good estimate may not yield a small square error, and that the proximity of errors may not lead to the proximity of estimates.

6.3.1. Onset exception

When the harmonics of a note is corrupted by an additive noise, a good estimate shall produce a residue close to that noise. In this case it is not adequate to run a least square estimator. A main source of additive noise in a piano recording is note onset attacks. While lasting a very short duration, the onset displays a wide-band behaviour that contaminates the spectrum of harmonics of other on-going notes. Accordingly, at the onset of each note we mark all on-going partials of other notes as not estimable for one frame, so that the least square estimation is not attempted.

6.3.2. Heavy interference exception

The least square estimator usually works fine when $\mathbf{W}$ is well-conditioned. However, as frequencies become close so that the sinusoids get highly correlated, $\mathbf{W}$ gets ill-conditioned and (15) becomes unstable. Intuitively, as the sinusoids become similar to each other, the amplitudes and phase angles can be traded between themselves without incurring much change to the residue, so that a good estimate with nearly least square error may deviate far from the least square estimate. Consequently we may observe large biases on least square estimates of individual sinusoids, even if the total residue is kept minimal.

To evaluate the reliability of amplitude and phase estimated with (16), we consider the sensitivity of the least square estimate of the m^{th} partial with regard to $\mathbf{x}$:

$$J_m = \left(\sum\nolimits_j u_{mj}^2\right)^{1/2} . \tag{17}$$

A change of $\boldsymbol{\delta}$ in the residue can contribute a bias up to $J_m\cdot\|\boldsymbol{\delta}\|$ to λ_m. The smaller is J_m, the more likely has λ_m a reliable estimate. In a CR which contains only one sinusoid, $J_1=(1+\varepsilon_w)\ \|\mathbf{w}\|^{-1}$, where $\|\mathbf{w}\|$ is the L^2 norm of the window spectrum, and $|\varepsilon_w|<<1$ is attributed to the truncations used to construct $\mathbf{W}$. We use $J_m\|\mathbf{w}\|$ as an indicator of the reliability of least square estimates. Amplitudes are estimated with (16) for partial m only if $J_m\|\mathbf{w}\|$ is below a preset threshold, e.g. 2. Other amplitudes are marked not estimable and passed on to the next stage (6.4).

6.4. Interpolation and extrapolation

Amplitude parameters that have not been measured due to reliability concerns are inferred from the already estimated ones by means of interpolation and extrapolation. Using such "guessing" techniques means we no longer target accurate additive separation as an objective. However, by common-sense design we may still maintain the naturalness of separated notes and their perceptual resemblance to what is heard in the original recording.

6.4.1. Interpolation

Interpolation is applied on a partial-by-partial basis to atoms whose amplitudes have not been estimated but lie between other atoms whose amplitudes have. More specifically, if amplitudes have been measured for a partial m at frames l_1 and l_2, $l_1<l_2$-1, but not at frames between the two frames, then we interpolate between the two frames exponentially:

$$a_{m,l} = \left(a_{m,l_1}^{l_2-l} a_{m,l_2}^{l-l_1}\right)^{\frac{1}{l_2-l_1}}, \ l=l_1+1, \ldots, l_2\text{-}1. \tag{18}$$

where $a_{m,l}$ is the amplitude of partial m at frame l.

The interpolation stage fills the gaps between atoms with measured amplitudes, but does not estimate amplitudes of atoms at the start and end of partials. For these an extrapolation stage is involved.

6.4.2. Extrapolation using amplitude modulation profile

To infer amplitudes before the first or after the last direct estimates of a sinusoid, we need to make assumptions about amplitude laws in these places. While it is always possible to extrapolate directly from the measured amplitudes of each partial, doing so, according to our experiments, is not advisable near onsets or far beyond measured atoms. On the other hand, since various partials can have reliable estimates during different intervals, it is possible to make cross-partial reference. In this paper we compute an *amplitude modulation (AM) profile* from the already estimated amplitudes for this purpose.

An AM profile is a function of time that describes the overall amplitude variation rate of the partials of a note. Let l be the frame index, the AM profile, denoted by P_l, is given as

$$P_l = \frac{\sum_{m=1}^{M} a_m \log \frac{a_{m,l}}{a_{m,l-1}}}{\sum_{m=1}^{M} a_m}, \; l=1, \ldots, L\text{-}1. \tag{19}$$

where L is the length of the note in frames, $a_{m,l}$ is the amplitude of partial m at frame l, and $a_m = \sum_l a_{m,l}$. P_l is interpreted as the average of amplitude rates at frame l weighted by partial amplitudes.

If none of the partials of a note has reliable amplitude estimates for the first L_1 frames (onset frames), we cannot compute P_l with (19) at frames 1 to L_1. In this case we obtain rough amplitude estimates by orthogonal projection[2], assuming partials near the onset have dominating energy. These rough amplitudes are used to estimate P_l for l=1, …, L_1-1. P_{L_1} is linearly interpolated from P_{L_1-1} and P_{L_1+1}.

If none of the partials of a note have reliable estimates for the *last* L_2 frames, we cannot turn to orthogonal projection like for onset frames, as leftover partials near offsets are often masked by noise or interference. For these frames we simply linearly extrapolate the AM profile itself, while taking special care that the amplitude rate be non-positive.

Further smoothing can be applied to the AM profile for improved smoothness. Once the AM profile is ready the extrapolation is done by applying the profile directly:

$$a_{m,l} = e^{P_l} \cdot a_{m,l-1} \text{ (forward)} \tag{20a}$$

$$a_{m,l} = e^{-P_{l+1}} \cdot a_{m,l+1} \text{ (backward)} \tag{20b}$$

7. TRANSIENTS

This section discusses the transient extraction stage (stage 4), which separates the initial attack of each note from audio. Piano notes have transients at keystrokes. Although in theory all sounds can be represented as sinusoids, the sinusoidal representation of a transient would involve faster amplitude and frequency changes than the standard technique could handle, and the physical meanings of the parameters would be unclear.

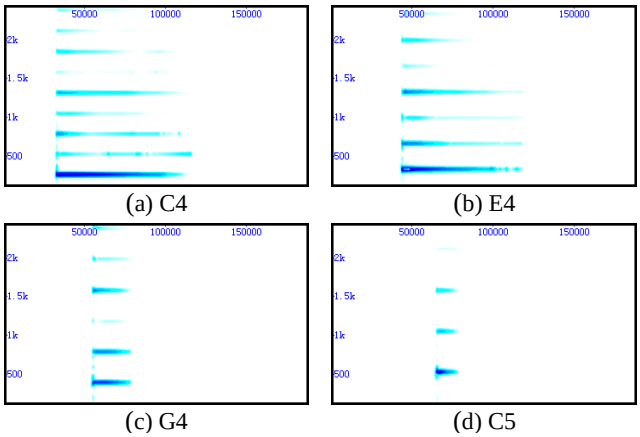
In this work we simply represent the transient with the complex spectrum. The transient is assumed to stretch the length of one long "transient" frame before the start of the harmonics. Let the DFT of this frame be $X(k)$, k=0, …, K/2-1, where K is the length of the transient frame; let f^1, …, f^M be the frequencies of the note(s) starting with this transient, and g^1, …, g^N be the frequencies of other on-going sinusoids during this frame. We derive the spectrum of the transient by notching out the on-going sinusoids, while preserving the starting ones:

1 °for each g^n, n=1, …, N, remove 4 bins from $X(k)$ centred at g^nK by setting at value at these bins to 0;
2 °for each f^m, m=1, …, M, recover 4 bins centred at f^mK by restoring these bins to their original value.

The separated transient is synthesized from the spectrum with inverse DFT. To reconstruct a complete note, we join the transient to the harmonics with standard overlap-add method. We initialize the phase angles of the harmonics to their spectral phases at the first frame. As the same phases are also preserved in the transient, the transition between transient and harmonics is kept smooth.

8. RESULT

We run our note separator on a commercial recording of Bach's Prelude in C, BWV 846a, using one channel sampled at 44.1kHz. The initial part of its spectrogram is given in Figure 2. Figure 6 shows the separation results for the first four notes. Graphically these single-note spectrograms look smoother than the originals in Figure 2, owing to the explicit modelling as constant-frequency harmonic sinusoids. We do observe irregularities in the harmonics where they used to overlap, but the result is well contained in plausible range. Since guessed parameters have guaranteed smoothness, we know that these less regular parts are the result of direct estimates from data, and it is these parts that pinned down the characteristics of the notes in the original recording. Perceptually we have found no audible artefact with the separated notes, and heard very little difference between the separated notes and their counterparts in the original recording.



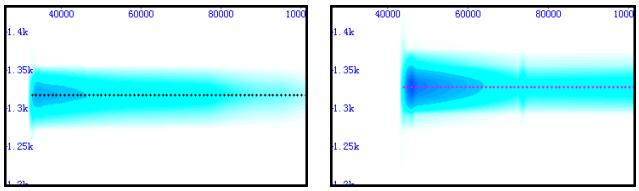
(a) C4 (b) E4

(c) G4 (d) C5

Figure 6 First four notes separated

Figure 7 shows details of Figure 6(a) and Figure 6(b) around 1.3kHz, i.e. the 5th partial of the first note (C4) and the 4th partial of the second note (E4). Although these two partials were shown to interfere heavily in the original spectrogram (Figure 3), our method is able to obtain reasonably clean separation from the same spectrogram. A small anomaly in the result is the inaudible amplitude spike estimated for the second note when the fifth note (E5) kicks in, showing that our handling of parameter estimation during other notes' onsets still has room for improvement.

[2] Or any other estimator for single, interference-free sinusoids.



(a) C4: 5th partial (b) E4: 4th partial

Figure 7 Details of Figure 6

Figure 8 shows the reconstruction we get by summing up all resynthesized notes in Figure 1. As with separated notes, the reconstructed spectrogram has a smoother and sharper look than the original. Because of parametric modelling, much of the non-music content in the original recording, such as background noise, the pianist's humming-along, and other unidentified extras, are removed from the reconstruction. Listening comparison between the reconstruction and the original shows the two highly similar to the ear. Minor difference can be heard at some note attacks, mainly towards the end of the recording where low-pitched notes become more frequent. Since low pitches introduce more interference to the harmonics and eats out more spectral bins from the transient, we can expect some performance loss on both the stationary and the transient parts.

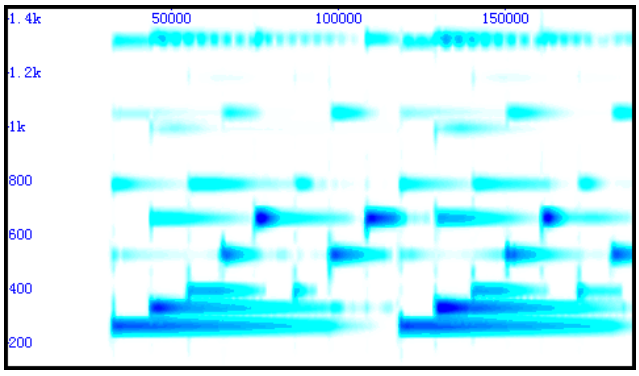


Figure 8 Reconstruction of recording

9. SUMMARY AND FUTURE WORK

In this paper we have presented a semi-automatic system for separating a piano recording into individual notes. The system employs two concepts: the imitation method that allows us to trade between modelling accuracy and result stability, and the collision region that helps organize systematic handling of interfering sinusoids. Our results show that the system is able to obtain separated notes with high quality and true to the original recording even though the signal processing techniques involved at fairly basic. The notes are represented with a transient-plus-harmonics model, therefore are ready to be used by all algorithms handling sinusoidal models, in addition to those handling monophonic or general audio waveforms. Direct reconstruction from the separated notes gives a de-noised version of the recording.

While reasonably self-contained for proof of concept, the proposed system is only a small step towards general note separation task. The system can be improved in many ways for sound quality, robustness and usability. For example, the phase angle at individual frames may be reincorporated during or after amplitude estimation to preserve minor pitch and wave shape variations. For amplitude estimation we may use multiple window sizes up to the duration of the collision region instead of a constant size of 2048. The transient energy where it collides with ongoing sinusoids may be partially restored, and the handling of transient may go beyond one frame to capture the full force and span of note attacks. We may also start bringing musical instruments with continuously variable pitches, as well as non-pitched instruments, into the picture.

10. ACKNOWLEDGEMENT

Part of this work was finished when the author was with Queen Mary, University of London and Professor Mark Sandler, with support from the Centre for Digital Music platform grant.

11. REFERENCES

[1] S. Ewert, B. Pardo, M. Müller and M.D. Plumbley, "Score-informed source separation for musical audio recordings: an overview," *IEEE Signal Processing Magazine*, vol.31 no.3, pp.116-124, May 2014.

[2] R. Hennequin, B. David and R. Badeau, "Score informed audio source separation using a parametric model of non-negative spectrogram," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Prague, 2011, pp.45-48.

[3] N. Bryan and G. Mysore, "Interactive refinement of supervised and semi-supervised sound source separation estimates," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vancouver, 2013.

[4] P.-K. Jao, Y.-H. Yang and B. Wohlberg, "Informed monaural source separation of music based on convolutional sparse coding," in Proc. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Brisbane, 2015.

[5] S. N. Levine and J. O. Smith III, "A sines+transients+noise audio representation for data compression and time/pitch scale modifications," in *Proc. AES 105th Convention*, San Francisco, 1998.

[6] N.H. Fletcher and T.D. Rossing, *The Physics of Musical Instruments* (2nd ed.), Springer-Verlag New York Inc., 1998.

[7] A. Gunnarsson & I. Gu, "Music signal synthesis using sinusoid models and sliding-window ESPRIT," in *Proc. IEEE International Conference on Multimedia and Expo*, Toronto, 2006.

[8] J. P. Bello, L. Daudet and M. B. Sandler, "Automatic piano transcription using frequency and time-domain information," *IEEE Transactions on Audio, Speech, and Language Processing*, vol.14, no.6, 2006, pp.2242-2251.

[9] S. Sigtia, E. Benetos and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol.24, no.5, 2016, pp.927-939.

[10] K. O'Hanlon and M. D. Plumbley, "Polyphonic piano transcription using non-negative matrix factorisation and group sparsity," in Proc. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Florence, 2014.

[11] M. Marolt, "A connectionist approach to automatic transcription of polyphonic piano music," *IEEE Transactions on Multimedia*, vol.6 no.3, 2004, pp.439-449.

[12] R.J. McAulay and T.F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol.34, no.4, 1986, pp.744-754.

[13] X. Serra, "Musical sound modeling with sinusoids plus noise," *Musical Signal Processing*, Swets & Zeitlinger Publishers, 1997.

[14] Wen X. and M. Sandler, "Sinusoid modeling in a harmonic context," in *Proc. 10th International Conference on Digital Audio Effects (DAFx)*, Bordeaux, 2007.

[15] T. Tolonen, "Methods for separation of harmonic sound sources using sinusoidal modeling," in *Proc. AES 106th Convention*, Munich, 1999.

ASSESSING THE SUITABILITY OF THE MAGNITUDE SLOPE DEVIATION DETECTION CRITERION FOR USE IN AUTOMATIC ACOUSTIC FEEDBACK CONTROL

Mr. Marc Ciufo Green
Audio Lab
Dept. of Electronics
University of York
York, UK
marc.c.green@york.ac.uk

Dr. John Szymanski
Audio Lab
Dept. of Electronics
University of York
York, UK
john.szymanski@york.ac.uk

Dr. Matt Speed *
Allen & Heath,
Penryn, UK

ABSTRACT

Acoustic feedback is a recurrent problem in live sound reinforcement scenarios. Many attempts have been made to produce an automated feedback cancellation system, but none have seen widespread use due to concerns over the accuracy and transparency of feedback howl cancellation. This paper investigates the use of the Magnitude Slope Deviation (MSD) algorithm to intelligently identify feedback howl in live sound scenarios. A new variation on this algorithm is developed, tested, and shown to be much more computationally efficient without compromising detection accuracy. The effect of varying the length of the frequency spectrum history buffer available for analysis is evaluated across various live sound scenarios. The MSD algorithm is shown to be very accurate in detecting howl frequencies amongst the speech and classical music stimuli tested here, but inaccurate in the rock music scenario even when a long history buffer is used. Finally, a new algorithm for setting the depth of howl-cancelling notch filters is proposed and investigated. The algorithm shows promise in keeping frequency attenuation to a minimum required level, but the approach has some problems in terms of time taken to cancel howl.

1. INTRODUCTION

In any system where sound captured by a microphone is amplified and reproduced by a nearby loudspeaker, a portion of the amplified sound emanating from the loudspeaker will be received by the microphone. This sound is subsequently re-amplified and fed back to the loudspeaker [1]. In this way, the sound system forms a closed loop (see figure 1). The most apparent effect of this acoustic feedback loop is the screeching sound that can develop (termed 'feedback howl'), which can cause severe limitations to the system's performance.

Formulated in a form relating specifically to acoustic feedback by van Waterschoot and Moonen [2], Nyquist's criterion states that if for a radial frequency ω:

$$\begin{cases} |G(\omega,t)F(\omega,t)| \geqslant 1 \\ \angle G(\omega,t)F(\omega,t) = n2\pi, \qquad n \in \mathbb{Z} \end{cases} \tag{1}$$

where $G(\omega,t)$ and $F(\omega,t)$ represent the short-term frequency responses of the forward and return parts of the loop respectively, then the system is unstable and has potential to feed back at that frequency.

Howl inevitably occurs in sound reinforcement systems as amplification levels are increased beyond a certain level - the system's Maximum Stable Gain (MSG) [2]. Given the system loop magnitude response, it is possible to predict the MSG as follows:

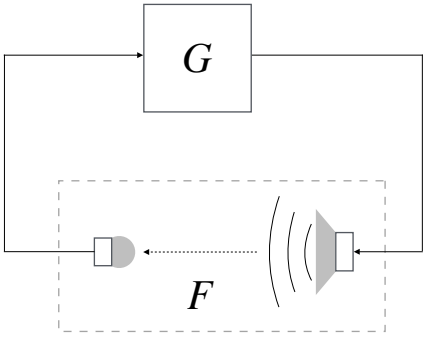


Figure 1: *Schematic for a PA system producing acoustic feedback.* ***G*** *represents the electroacoustic forward signal path, and* ***F*** *represents the acoustic return path.*

$$\text{MSG [dB]} = -10\log_{10}\frac{\max(|G(\omega)F(\omega)|^2)}{|\overline{G(\omega)F(\omega)}|^2}, \;\; \omega \in \mathcal{P} \tag{2}$$

where the denominator of the fraction is the Mean Loop Gain (MLG) and $\mathcal{P}$ represents the set of frequencies in the range of interest that fulfil the phase condition of equation (1). The MSG defines the upper limit for usable amplification levels from any given PA system. It is the aim of feedback control systems to increase the MSG of a system, giving more usable gain before feedback howl occurs.

Despite the availability of numerous automatic feedback management systems on the market, the majority of professional sound engineers prefer to manage feedback manually, typically reducing the magnitude of problematic frequency bands using a graphic equaliser [3, 4]. This is largely due to the common perception that automatic feedback control systems are unreliable - there is always the risk of an automatic system falsely identifying a desired sound component as feedback howl and attempting to suppress it, or an actual instance of howl going undetected. Both scenarios could ruin a carefully-constructed live mix.

Setting aside these flaws, a reliable automatic feedback management system would make a desirable addition to live sound technology. The acoustic return path response $F(\omega)$ can change dramatically over time depending on room temperature [5], the addition of an audience into a performance space and particularly

* M Speed is now affiliated with Focusrite Audio Engineering

microphone movement [6, 7]. This can affect the frequencies at which feedback howl is likely to occur.

This paper investigates the Magnitude Slope Deviation (MSD) method for automatic detection of feedback howl, and is organised as follows: Section 2 outlines previously-proposed feedback howl detection methods, details the MSD method and proposes new algorithms based upon this method. Section 3 introduces the software toolkit created for this research and outlines the tests that were undertaken. Results from these tests are presented in Section 4 and discussed in Section 5. Section 6 concludes the paper.

2. HOWL DETECTION

Most modern automatic feedback control systems focus on breaking the gain condition of equation (1) by applying filters to the audio signal in the forward path in order to attenuate problematic frequencies. The Notch filter-based Howling Suppression (NHS) technique is by quite some way the method that has seen the widest use [8, 6, 9]. NHS systems use a bank of narrowband notch filters, reducing the gain at very localised frequency bands to remove howl frequencies.

A key factor in the effectiveness of these systems is the means by which howl is identified from a background of 'desired' musical or speech sound. In order to minimise false positive identifications that could result in incorrect suppression of music or speech, it is important to accurately differentiate howl from desired signal components

The first step in howl identification is spectral analysis of the incoming signal, followed by the application of a standard peak picking algorithm to find local maxima in the spectrum and identify a number of candidate howl frequencies (generally around 10) [2, 10, 9]. Each candidate frequency peak is subsequently analysed to determine whether the peak is caused by a feedback howl or a desired source signal component. Various methods of doing this have been proposed, based upon observed spectral and temporal characteristics of feedback howl that are distinct from music or speech, including:

- higher magnitude than desired components (when allowed to develop) [10]
- sinusoidal in nature - highly localised in frequency [11]
- lack of harmonic components until signal clips [11]
- consistently present across time, very little frequency deviation [2]
- exponential increase in magnitude until signal clips [12]

These features are illustrated in Figure 2, which shows a spectrogram of a simulated microphone signal featuring speech components and a howl component that is clearly visible at just over 700 Hz.

The methods of howl identification broadly fall into two categories based on which howl characteristics they utilise in their analysis: spectral or temporal. Spectral methods compare the magnitude of a candidate howl peak to a reference magnitude that can be set manually, or obtained by a number of means. If the ratio between the candidate howl peak and the reference magnitude exceeds a certain threshold, the candidate peak is flagged by the system as howl. Reference magnitudes can be pre-set absolute values [3, 2, 13, 8, 10], or calculated from the average power across the spectrum [8, 14, 15, 9], neighbouring frequency magnitudes [16, 12] or harmonic frequency magnitudes [4, 8].

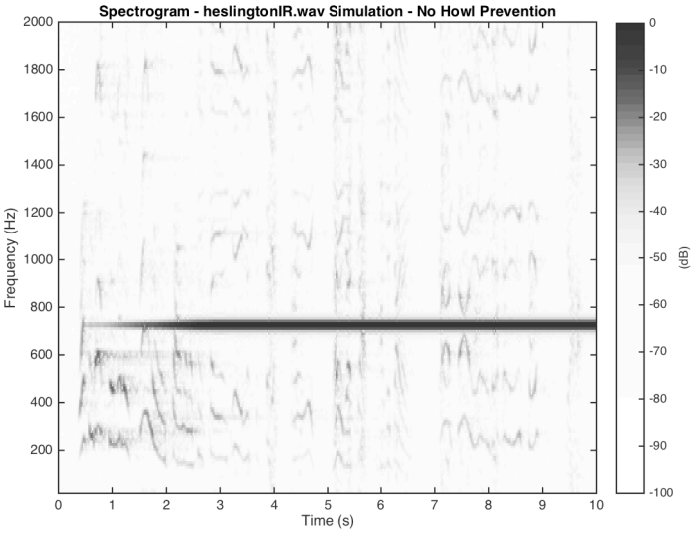


Figure 2: *Spectrogram of feedback simulation using a speech signal and a 'church' acoustic environment. No howl prevention measures have been applied.*

As opposed to the spectral methods, temporal methods of howl identification require several frames of frequency data in order to conduct their analysis. The Peak Magnitude Persistence technique [13, 15, 17] looks for candidate frequencies that are present across large amounts of time relative to the typical duration of speech or music components.

The most recently-proposed temporal method of howl identification is measurement of the so-called Magnitude Slope Deviation (MSD) of candidate frequency bins [10]. This technique, proposed by Osmanovic et al. [12], utilises the fact that howl component power increases linearly over time when plotted on a decibel scale. The system described (intended for use in aviation communication systems) looks at the changes in frequency magnitude of a candidate frequency band over time by storing an 8-frame 'history buffer' of magnitude values. Once peak picking has identified potential howl frequencies, the historic data for those frequencies is analysed and a 'global reference' line gradient between the first and last magnitude values in the memory buffer is calculated. Gradient values for adjacent magnitude values are subsequently calculated and compared against this reference gradient to find deviations in gradient value. If the average deviation between gradients is < 0.05 dB per frame then the frequency band in question is flagged as a probable howl candidate. The MSD technique was verified to work well in aircraft communication scenarios, but has not previously been tested in live music or speech sound reinforcement scenarios.

2.1. The 'Summing' MSD Method

The originally-proposed method to calculate MSD involves numerous gradient calculations for each new frame of frequency spectrum data. For a magnitude history buffer of length N and with K frequency bins, KN gradients must be calculated. For more detailed frequency analyses, the number of calculations required increases significantly, and this computational inefficiency is not ideal for time-critical feedback cancellation scenarios. To mitigate

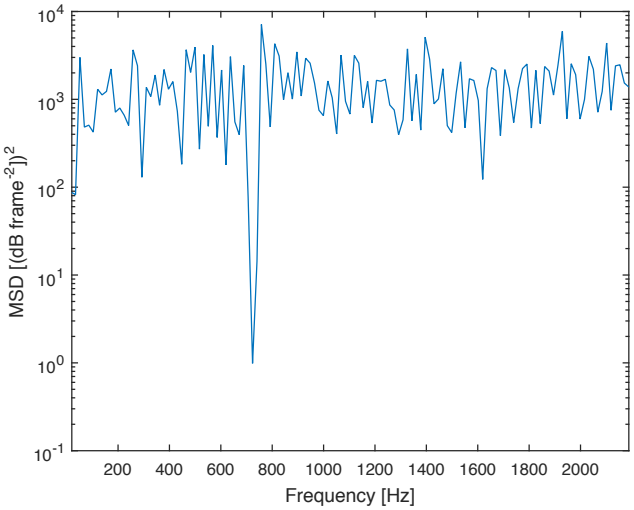


Figure 3: *MSD data calculated with the summing method using a 16-frame FFT history buffer for the howl simulation shown in Figure 2.*

this problem, a new method for assessing the MSD of frequency spectrum data was implemented and tested as part of this research.

In the new 'summing' method, a finite-difference approximation to the second-order derivative (with respect to time) of the decibel-scale magnitude history buffer data is calculated. Any frequency bin that is linearly growing in magnitude over time should have values for gradient change that are consistently close to zero. To calculate the Magnitude Slope Deviation, the absolute values of historical gradient change data for each frequency bin are squared (in order to accentuate the difference between minima and the rest of the data), then summed. The approach can be expressed as:

$$\mathrm{MSD}(k,m) = \sum_{n=(m-N)+1}^{m} |G''(k,n)|^2 \qquad (3)$$

where m is the present analysis frame and $G''(k,n)$ is the finite-difference approximation of the second derivative of the dB-scale magnitude history data by frequency bin k and analysis frame n, with respect to n. This produces pronounced minima where magnitude gradients are more consistent over time, and minima that are below a certain threshold are flagged as howl frequencies. Figure 3 shows data calculated using this technique for the howl simulation shown in Figure 2, using a history buffer of 16 frames (N = 16) of FFT analysis data, each calculated using 256 samples (k = 128). The pronounced valley at 725 Hz is a strong indication that howl is present at that frequency.

One problem with this summing approach is that $\mathrm{MSD}(k,m)$ values would approach zero if there were periods of time with no energy in the kth bin. This would give false positive howl identifications. A future version of this algorithm should address this issue with the inclusion of a condition to exclude the identification of frequency bins with zero or very low energy.

2.2. The MSD-Inspired Notch Depth Setting Algorithm

One of the problems faced by Notch filter-based Howling Suppression systems is how best to set the depths of notch filters introduced to cancel howl. It is preferable that the notch filter depths be as shallow as possible so as not to affect the 'desired' sound too much, whilst still being deep enough to ensure that the gain condition of equation (1) is broken. This paper proposes a new method to set notch filter depths known as MSD-Inspired Notch Depth Setting (MINDS) algorithm.

The idea behind MINDS, given that feedback howls are characterised by their growth over time, is that one way to find the minimum filter depth required to cancel the howl is to monitor the candidate frequency's magnitude over time and gradually increase the depth of a notch filter until the magnitude of the howl ceases to increase. At this point, if the howl magnitude remains relatively constant, the filter must be holding the howl in equilibrium (loop gain $\approx$ 0 dB) and only a small additional depth increase should be needed to cancel the howl.

MINDS is implemented by utilising historical frequency magnitude data to compare the latest two magnitude gradients. If both these gradients are above zero (or close, allowing very slowly decaying howls where loop gain is just below unity) then the notch depth is increased by 1 dB, up to a maximum depth of -25 dB. It is reasoned that if both these gradient values are negative, this indicates that any howl present must be in decline, whereas if one gradient is positive and the other negative, this is indicative of a rapidly-changing signal that would not be consistent with the presence of howl. Filter depth is not increased in either case. This behaviour is outlined in Algorithm 1, where k_c is a candidate frequency bin. In this way, even newly-received howl candidate frequencies may not trigger the increase of filter depth. The process repeats every time new frequency magnitude data (and with this, new howl candidate data) becomes available.

At present, this algorithm provides no provision for the removal of notch filters or any reduction in their depth after howling has been suppressed. This was not a problem for the short simulations described here.

if $G'(k_c,m) > -0.5$ **and** $G'(k_c,m-1) > -0.5$ **then**
  if *notch depth* > -25 **then**
    notch depth = notch depth - 1;
  end
end

Algorithm 1: MSD-Inspired Notch Depth Setting

3. TEST METHODOLOGY

3.1. The Feedback Analysis and Cancellation Toolkit

In order to test the effectiveness of the MSD and MINDS algorithms in different scenarios in a repeatable fashion, it was necessary to create a system capable of simulating feedback scenarios using any given 'desired' stimulus sound and acoustic environment. To this end, a system known as the Feedback Analysis and Cancellation Toolkit (FACT) was developed. FACT is split into three subsections, dealing with creating the virtual feedback loop (simulation), detecting any howls that arise (detection) and cancelling them out (notch filtering).

The most important part of the FACT system is the simulation of the acoustic feedback itself. This is initialised using a stimulus sound that represents the 'desired' microphone input (this can be any monaural audio file) and a monaural Room Impulse Response (RIR) representing the loop response of the sound system. All of

the simulations conducted for this study use stimulus sounds 20-30 seconds in length, and RIRs gathered using one loudspeaker and one microphone using the swept-sine technique. All simulations were run using a sampling frequency of 44.1 kHz.

Since the development of feedback howl is dependent on the Maximum Stable Gain of the sound system, the MSG of each RIR used was calculated using equation (2). In order to achieve a desired target Mean Loop Gain for the simulation, a factor by which to multiply the RIR values was specified as:

$$2^{(\text{Target Gain [dB]} - \text{MLG [dB]})/6} \tag{4}$$

In order to facilitate the introduction of filters mid-simulation when modelling the acoustic feedback loop, the stimulus sound is split into frames. Each frame is convolved with the RIR one at a time and the output from the convolution is recorded and simultaneously added back into the input, starting at the beginning point of the next frame. The convolution output is not confined to the next frame, allowing convolution outputs to accumulate over time. This effectively simulates the coupling of the loudspeaker output to the microphone input in addition to the desired sound. FACT makes use of Hann windowing, and each frame is overlapped by 50% with adjacent frames in order to smooth changes in notch filter processing that may occur between frames. Figure 4 shows a diagrammatic representation of this process. Hann windows were used as overlapping these by 50% gives unity gain in the overlapped region. A signal that is split into frames that overlap by 50%, windowed using the Hann function and then recombined can be reconstructed exactly.

To gain frequency spectrum data to use in howl analysis, an FFT process using analysis frames of 256 samples was run concurrently with the howl simulations. Before FFT analysis, the simulation output signal was downsampled by one order of magnitude to 4410 Hz using MATLAB's resample function, which automatically applies an anti-aliasing filter. This was in order to cut down on the amount of frequency data to be analysed by the system (full-spectrum analysis would produce ten times more frequency data). Analysis of RIRs using the criteria in equation (1) showed no feedback howls developing below the upper frequency analysis cutoff of 2205 Hz. This was corroborated in preliminary howl simulations without any notch filtering, so the reduced frequency range was deemed sufficient for these simulations.

3.2. Testing 'Summing' MSD

To test the effectiveness of the new 'summing' MSD algorithm against the original, simulations were run using a short sample of conversational speech as stimulus sound. This stimulus was chosen as the original MSD algorithm has previously been established as working well with speech signals [12]. Simulations were run using four different RIRs in order to assess the effectiveness of the algorithm across several different acoustic environments. These were an RIR of a living room recorded by the author - referred to as 'Small Room' - and three RIRs of larger spaces sourced from openairlib.net [18], which will be referred to as 'Church 1', 'Church 2' and 'Hall'.

The simulations were run using an MLG of 2 dB above the MSG level calculated for each IR. Firstly, each simulation was run with howl detection disabled. This was to allow howl to develop in each case in order to confirm the presence of howl frequencies predicted by analysing the RIRs using equation (1). Figure 2 shows the output of one such simulation. Next, the simulations were re-run twice, once using the 'original' MSD detection method and once using the 'summing' MSD detection method, both using magnitude history buffer lengths of 16 frames. The gradient tolerance for the 'original method' simulations were set to 0.5 dB change per frame and the 'summing method' simulations used a minima threshold of 1 $(\text{dB frame}^{-2})^2$ (see equation (3)). This makes the testing more stringent on the summing method, as calculating the MSD based on the values used here for the original method would equate to using a summing minima threshold of 4 $(\text{dB frame}^{-2})^2$. It should be noted that this also has the potential to make results from the original algorithm less accurate, though this was not observed in the tests presented here.

The effectiveness of each algorithm was assessed by analysing FACT output data. Detection speeds were found by examining timestamp data corresponding to the earliest introduction of filters at correct frequencies. The accuracy of each algorithm was determined by finding the number of unique filter frequency values and calculating the percentage of unique values that correspond to actual howl frequencies. Howl identifications in adjacent frequency bins were treated as identification of a single howl (allowing for spectral leakage). For these tests, in order to evaluate computational efficiency, MATLAB's 'Run and Time' feature was also used to generate a report indicating the processing time of each simulation. The simulations were run on an Apple MacBook Pro, powered by a 2.4 GHz Intel Core 2 Duo CPU.

3.3. The Impact of Varying the History Buffer Length

In the original proposal for the MSD method, the buffer length of magnitude history data used for analysis is 8 frames [12]. Since this system was initially proposed to cancel howl only in speech signals, it is possible that 8 frames might not be adequate data for musical scenarios. A series of tests were therefore run to assess how varying the length of magnitude history data used to calculate gradient deviations by the MSD algorithm affects its accuracy in detecting howl from different stimulus sounds.

Tests were run using the 'Small Room' RIR at a gain 3 dB above the MSG. Loop response analysis of the IR at this gain level showed two frequencies liable to howl. Simulations were then run iteratively, with magnitude history buffer lengths varied between three frames - the minimum required to detect a persisting gradient - and twenty-four frames. The simulations were run using three stimuli in order to assess how using different buffer lengths could yield different levels of detection accuracy in multiple scenarios. The stimuli were the conversational speech sample used previously, an excerpt from 'Jupiter' from Holst's *The Planets* [19] ('Classical Music') and an excerpt from 'The Raven That Refused To Sing' by Steven Wilson [20] ('Rock Music'). Howl detection times and accuracies were calculated as described in section 3.2.

3.4. Testing the MINDS Algorithm

The effectiveness of the MINDS algorithm was assessed using data gathered for the magnitude history buffer length tests described in section 3.3. In these simulations, the same instance of feedback howl was detected at slightly different times and therefore had been allowed to grow to different magnitudes before a filter was introduced. In this way, the final depths reached by the filters introduced to counter the howls could be compared against the magnitude of the howl when they were introduced. Notch filter depth

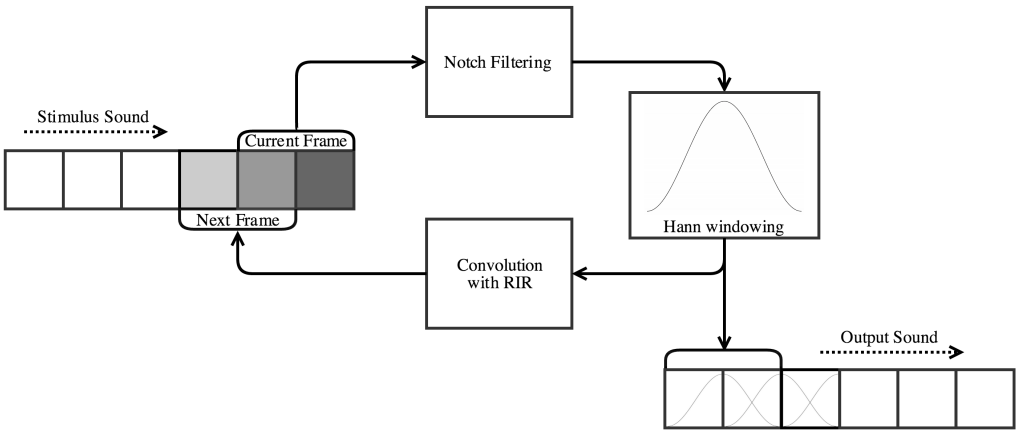


Figure 4: *Convolution process used to create the virtual acoustic feedback loop.*

Table 1: *MSD Algorithm Howl Detection Times [seconds]*

	Howl 1		**Howl 2**	
IR	Original	Summing	Original	Summing
Small Room	1.074	1.045	5.050	3.846
Hall	2.148	2.017	5.050	5.050
Church 1	7.619	7.619	8.664	8.606
Church 2	1.509	1.451		

Table 2: *MSD Algorithm Processing Times*

Time [s]	Original	Summing
Small Room	48.394	0.262
Hall	46.610	0.339
Church 1	43.841	0.330
Church 2	44.260	0.385

data was only used when the introduced filter remained assigned to the howl frequency by the end of the simulation, not reassigned due to any false positive howl IDs.

4. RESULTS

4.1. MSD Algorithm Types

Both original and summing forms of the algorithm were 100% accurate in howl identification. Table 1 shows the detection times for the first and second howl frequencies in each simulation as detected by both the original and summing algorithms. As can be seen, the summing algorithm detects howl occurrences as quickly or faster than the original algorithm in every case.

Table 2 shows the processing time taken in each simulation to run the MSD algorithm. It can be clearly seen that the summing algorithm is an order of magnitude more efficient to run than the original algorithm. The quoted times represent 1754 calls to the MSD evaluation function, meaning that, on average, the original algorithm takes 26ms to run, whereas the summing algorithm takes 188μs. The summing algorithm is therefore almost 140 times more efficient than the original algorithm on average. It is just as accurate as the original algorithm at detecting howls and always detects howls just as fast or faster than the original algorithm.

Although these tests represent only a small number of scenarios, the results demonstrate the advantages of the summing algorithm very clearly. For this reason, the other simulations in this study used the summing algorithm.

4.2. History Buffer Length

Figure 5 shows how detection accuracy varied with magnitude history buffer length using each of the three stimuli. Accuracy of detecting howl from speech increases rapidly as buffer length is increased, reaching 100% using a buffer of just seven frames. This perhaps illuminates why a buffer of eight frames was originally proposed by Osmanovic *et al.* [12], as the original system was designed for use with speech.

At that same seven-frame mark, accuracy has reached 66% (meaning only one incorrect howl ID) in the 'Classical Music' simulations. The accuracy hits 100% using an eleven-frame buffer, briefly dipping back down to 66% before staying at 100% from a thirteen-frame buffer onwards.

Howl detection using the 'Rock Music' stimulus is significantly less accurate. Accuracy does steadily increase as the buffer length is increased, eventually climbing to 22% as the buffer length reaches twenty-four frames. This provides some indication that a much longer buffer length could remedy the inaccuracy of the MSD algorithm in the rock music scenario. Unfortunately, twenty-four analysis frames represents over a third of a second of audio - already a long time to allow howl to develop. Going by the upward trend, over 100 frames of data would be required in order to approach 100% accuracy (although this has not been tested). This would correspond to almost 1.5 seconds of audio - clearly an unacceptable amount of time to allow potential howls to develop before detection.

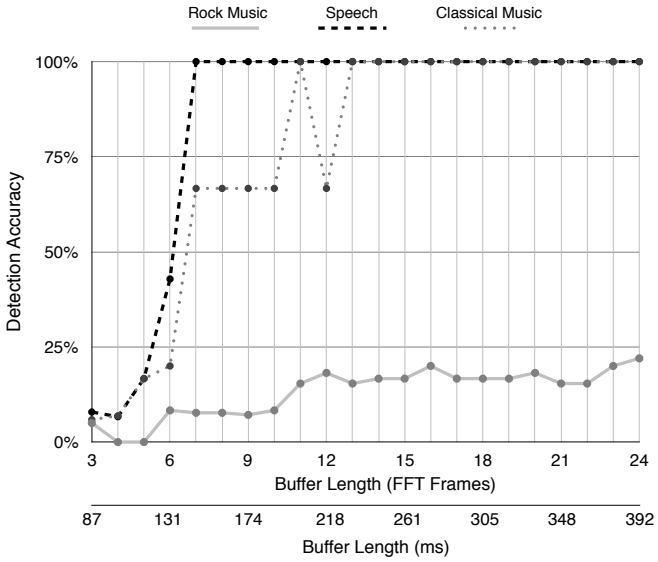


Figure 5: *Detection accuracies of howl from different stimuli when varying the length of magnitude history buffer available for MSD analysis.*

4.3. MINDS Algorithm Evaluation

Figure 6 shows final notch depths plotted against initial howl magnitudes for both first and second howl frequencies. As can be seen, there is a slight trend downwards in filter depths as howl magnitude increases in both cases, but this is not as pronounced as might be expected. In the case of the first howl, for instance, the magnitude of the howls before the filter was added varies from -9.2 dB to over 8 dB - a range of over 17 dB, whilst the final filter depths only vary 3 dB across that entire range. With the second howl, the magnitude varies over a range of over 20 dB whilst the filter depths only vary by 4 dB. These results give some evidence of the effectiveness of the MINDS algorithm at arriving at the minimum notch depth required to cancel a howl peak, regardless of the magnitude of the howl upon detection. As these results are limited to one set of simulations, however, more tests would be required to give conclusive evidence on this point.

Despite the promising nature of the aforementioned results, there is some evidence from other simulations that the algorithm's mandate to keep frequency attenuation to a minimum can sometimes have unintended consequences. Figure 7 shows one such case. The initial howl frequency at 724 Hz is quickly detected and effectively cancelled. The second howl, at 1171 Hz, is detected and a filter is introduced at 2.93 seconds. The depth of the filter reaches its final value of -10 dB by 3.3 seconds. It can be seen from the spectrogram that the howl persists at a significant magnitude for several seconds after the filter depth ceases to increase, remaining a component of the signal for the duration of the ten-second segment depicted in the figure. Since this howl is no longer growing in magnitude and is not significantly higher in magnitude than the desired audio, most detection algorithms are not able to identify it. Since it is not likely to be flagged as a howl candidate again, the howl is allowed to persist, which could affect listener perception of sound quality in a live scenario.

5. DISCUSSION

Results from the testing of the two forms of MSD algorithm reflect favourably on the summing method. The summing method never took longer to catch howl than the original method in any case, and the increase in computational efficiency is vast, which should make it more feasible to implement MSD analysis on embedded processing chips with limited computing power. It could be especially useful in keeping the necessary number of calculations low when applying the MSD algorithm to rock music scenarios, which tests indicated may require much longer buffer lengths (and hence more calculations if the original MSD method was used) than other scenarios to maintain accuracy.

More testing is required to confirm its effectiveness in a wider variety of scenarios, however it seems the summing MSD algorithm can be recommended over the original algorithm despite the limited amount of data comparing the two methods in this study.

The results presented show that in addition to the previously-confirmed effectiveness of the MSD algorithm at identifying howl from speech, the algorithm can also discern howl from classical music, albeit requiring a larger magnitude history buffer to achieve optimal accuracy than for speech in this case. The findings reported here contradict those of van Waterschoot and Moonen [10], who reported results from a similar test indicating that an MSD algorithm using a 16-frame history buffer is 55% likely to trigger in error as gradient deviation threshold is adjusted to give a 100% chance of howl detection. Even more interesting is the fact that their tests used a solo violin piece as stimulus sound. One would expect this kind of stimulus to be relatively spectrally sparse, which the results of this study indicate should make howl detection accurate using the MSD algorithm. One possible reason for this discrepancy is the fact that van Waterschoot and Moonen used a full-spectrum FFT analysis in their test.

The clear exception to the generally excellent accuracy of the MSD algorithm found here is when the rock music stimulus was used. Accuracy ratings in this case were much lower than for other stimuli. The fact that so many false howl identifications were made is probably due to the harmonic richness of the bass guitar sound, clearly visible on the simulation shown in Figure 8. This figure shows the close correspondence of many of the filter frequencies to these harmonics and how a great deal of the low-end of the signal has been attenuated by the five-second mark. Since the stimulus is a polished rock production, the bass guitar likely has dynamic range compression applied. This means that there is a period of time after the instrument's attack phase (when the strings are plucked) where the amplitude of the instrument will be more or less constant, rather than exhibiting a natural amplitude decay. It is this period of constancy that is likely triggering the false howl identifications. This could also explain why the MSD algorithm performed relatively well here when classical music was used as the stimulus, as dynamic range compression is not typically used in classical recordings. Since dynamic range compression is common in rock and pop live performances as well as recordings, this represents a shortcoming of the MSD detection algorithm that needs to be addressed before it can be incorporated into any commercial feedback control systems intended for use in those scenarios.

The problems encountered using MSD with a rock-music stimulus sound recall the early systems using Peak Magnitude Persistence howl detection, which examined the sound signal for correlation at time intervals that had to be "greater than the duration of... a single note in a musical performance" [13] for any level of

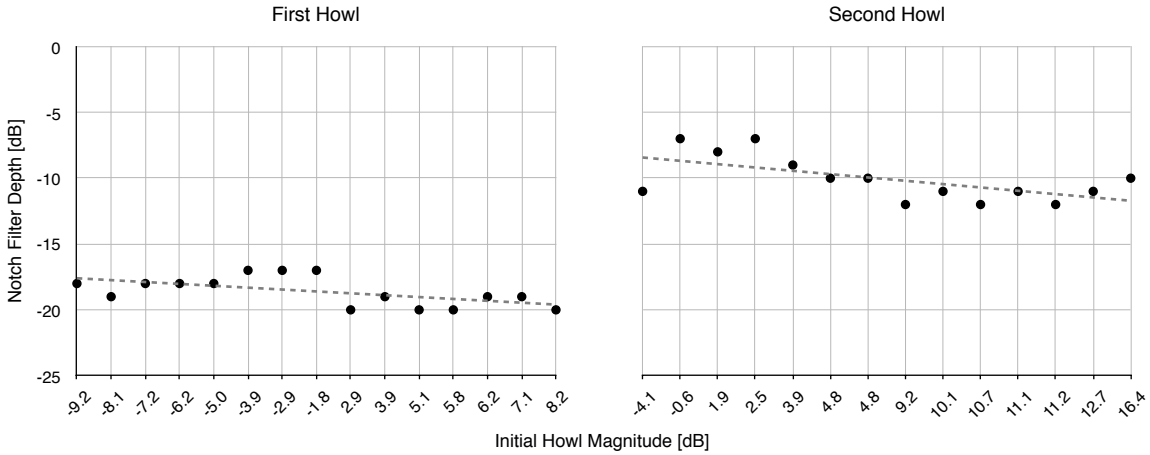


Figure 6: *Final notch filter depths against howl magnitude upon identification. Simulations used 'Classical Music' as stimulus and 'Small Room' as loop response.*

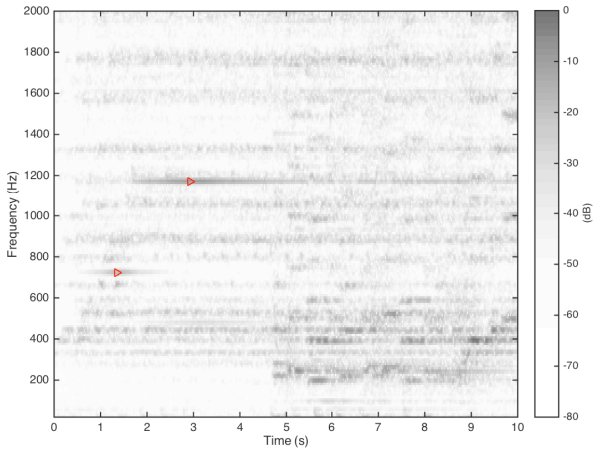


Figure 7: *Spectrogram of simulation output using MINDS to set notch filter depths. Triangles indicate addition of notch filters.*

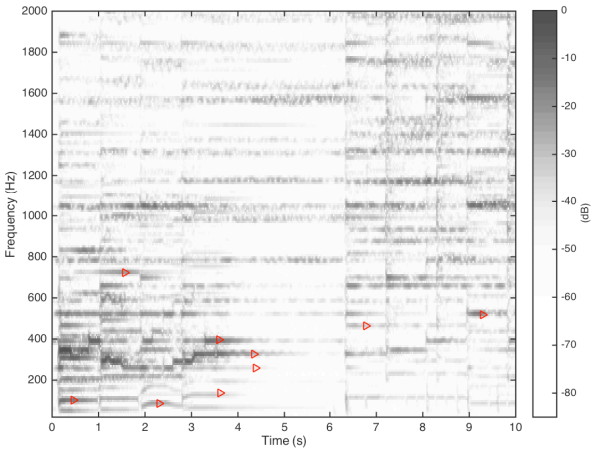


Figure 8: *Spectrogram of simulation output using MSD algorithm on rock music stimulus. Triangles indicate addition of notch filters.*

accuracy to achieved. It seems that the MSD algorithm may face a similar stipulation, at least in situations where dynamic range compression is in use. The typical length of a bass note in 'The Raven That Refused To Sing' is about 1 second. This corresponds closely to the trends shown in Figure 5, which indicate that around 1.5 seconds of audio information would need to be analysed by the MSD algorithm in order for detection accuracy approaching 100% to be achieved with this stimulus.

5.1. MINDS Algorithm

The results presented here show some evidence for the effectiveness of the MINDS algorithm at finding the optimum depth for a notch filter cut, regardless of the magnitude of the howl upon its detection. Since only gradient information is considered, the variance in final filter depths is very small compared with that of the howl magnitudes upon their introduction. Whilst these results are very promising, there is a tendency to make the notch filter depths slightly too shallow. This problem could perhaps be alleviated by introducing a 'final cut' stage to the algorithm. In such a feature, the depth of each notch filter would be increased by an additional fixed amount when the gradient of the howl frequency magnitude turns negative, thus ensuring a fast cancellation of the howl at the expense of perhaps attenuating the problematic frequency slightly more than the absolute minimum required. In its present incarnation the MINDS algorithm simply stops increasing the filter depth at this point, allowing the howl to decay at its own pace, which can sometimes take several seconds.

6. CONCLUSION

The aim of this paper has been to investigate the viability of the MSD and MINDS algorithms for use in automatic acoustic feedback cancellation systems in live-sound scenarios. The new 'sum-

ming' method for calculating MSD has been shown to be much more computationally efficient, yet no less accurate or timely, and can be recommended as an MSD implementation of choice moving forward. The algorithm has been shown to work well in the speech and classical music scenarios tested here, but potentially less so in rock music scenarios where extensive use of dynamic range compression can interfere with the functionality of the algorithm, causing it to be 8 less accurate when using comparable history buffer lengths to classical music or speech scenarios.

The MINDS algorithm has been shown to be very promising in terms of its ability to cancel howl instances whilst keeping notch filter depths to a minimum. There are some problems with this approach in terms of the speed at which howl instances can be cancelled, and it would be desirable to modify this algorithm in order to reduce the time taken to calculate optimum notch filter depths.

7. ACKNOWLEDGEMENTS

This research was conducted in conjunction with Allen & Heath Limited, who provided support and supervision throughout the project.

8. REFERENCES

[1] R. O'Donnell, "Prolog to 'fifty years of acoustic feedback control: State of the art and future challenges'," *Proceedings of the IEEE*, vol. 99, no. 2, pp. 285–287, Feb. 2011.

[2] Toon Van Waterschoot and Marc Moonen, "Fifty years of acoustic feedback control: State of the art and future challenges," *Proceedings of the IEEE*, vol. 99, no. 2, pp. 288–327, Feb. 2011.

[3] Joseph O Renauer, "The Feedback Finder: A Simple Solution to a Common Problem," in *Preprints 63rd AES Conv.*, Los Angeles, May 1979.

[4] Michael P Lewis, Timothy J Tucker, and Doran M Oster, "Method and apparatus for adaptive audio resonant frequency filtering," U.S. Patent 5,245,665, Sep. 14, 1993.

[5] K Heinz, "Feedback Elimination Simplified," in *Preprints 54th AES Conv.*, Los Angeles, May 1976.

[6] G Rombouts and Toon van Waterschoot, "Proactive notch filtering for acoustic feedback cancellation," in *Proc. 2nd Annual IEEE Benelux / DSP Valley Signal Process. Symp.*, Antwerp, Mar. 2006, vol. 22, pp. 169–172.

[7] W. K. Connor, "Experimental investigation of sound-system-room feedback," *Journal of the Audio Engineering Society*, vol. 21, no. 1, pp. 27–32, Feb. 1973.

[8] M. H. Er, T. H. Ooi, L. S. Li, and C. J. Liew, "A DSP-based acoustic feedback canceller for public address systems," *Microprocessors and Microsystems*, vol. 18, no. 1, pp. 39–47, Jan. 1994.

[9] Ariel F Rocha and Aníbal J S Ferreira, "An Accurate Method of Detection and Cancellation of Multiple Acoustic Feedbacks," in *Proc. 118th AES Convention*, May 2005.

[10] Toon van Waterschoot and Marc Moonen, "Comparative evaluation of howling detection criteria in notch-filter-based howling suppression," in *Proc. 126th AES Convention*, Munich, May 2009.

[11] C R Boner and C P Boner, "Minimizing Feedback in Sound Systems and Room-Ring Modes with Passive Networks," *Journal of the Acoustical Society of America*, vol. 37, no. 1, pp. 771–775, Jan. 1965.

[12] Nermin Osmanovic, Victor Clarke, and Erich Velandia, "An In-Flight Low Latency Acoustic Feedback Cancellation Algorithm," in *Proc. 123rd AES Convention*, New York, Oct. 2007.

[13] Eugene T. Patronis, "Electronic Detection of Acoustic Feedback and Automatic Sound System Gain Control," *Journal of the Audio Engineering Society*, vol. 26, no. 5, pp. 323 – 326, May 1978.

[14] Mitsuru Hanajima, Michiaki Yoneda, and Toshiyuki Okuma, "Howling Eliminating Apparatus," U.S. Patent 6,125,187, Sep. 26, 2000.

[15] Paul Robert Williams, "System for Elimination of Acoustic Feedback," U.S. Patent 8,634,575, Jan. 21, 2014.

[16] S. Ando, "Howling detection and prevention circuit and a loudspeaker system employing the same," U.S. Patent 6,252,969, Jun. 26, 2001.

[17] J. B. Foley, "Adaptive Periodic Noise Cancellation for the Control of Acoustic Howling," in *Proc. IEEE Colloq. Adaptive Filters*, London, Mar. 1989, pp. 7/1 – 7/4.

[18] "OpenAIR | The Open Acoustic Impulse Response Library." [Online]. Available: http://www.openairlib.net/ [Accessed: Jul. 13, 2015].

[19] G. Holst, Composer, C. Dutoit, Conductor, Holst: The Planets (Classic FM: The Full Works) [Sound recording]., Decca (UMO), 2012.

[20] S. Wilson, Artist, The Raven That Refused To Sing (And Other Stories) [Sound recording]., Kscope, 2013.

A ROBUST STOCHASTIC APPROXIMATION METHOD FOR CROSSTALK CANCELLATION

Huaxing Xu Risheng Xia Junfeng Li Yonghong Yan *

Key Laboratory of Speech Acoustics and Content Understanding
Institute of Acoustics
Beijing, China
xuhuaxing, lijunfeng@hccl.ioa.ac.cn

ABSTRACT

Crosstalk cancellation serves as an important role in binaural signals playback through loudspeakers, which reproduce a particular auditory scene to the listener's ears. In practice, due to either the listener's head movement or rotation, etc, the actual transfer function matrix will differ from the design matrix, which results in deterioration in the performance of crosstalk cancellation. Crosstalk cancellation system (CCS) is very non-robust to these perturbations. Generally, in order to improve the robustness of CCS, several pairs of loudspeakers using a multi-band approach processing band-passed content to appropriately spaced loudspeakers are needed. In this paper, by means of assumed stochastic analysis, a stochastic robust approximation method based on random perturbation matrix modeling the variations of the transfer function matrix is introduced and evaluated. Under free-field condition, simulation results demonstrate the effectiveness of the proposed method.

1. INTRODUCTION

Binaural technology is often used to reproduce a virtual auditory scene to the listener as if he/she is personally on the scene. The principle of binaural technology is to reconstruct the acoustic pressures at the listener's eardrums so that the reproduced sound field is identical with what would be produced and can deliver an extremely realistic three dimensional virtual acoustic environment to a listener, which could be of great benefit in virtual reality, augmented reality, computer multimedia, home theater, video games, digital television, and so forth [1, 2]. First, the binaural signals are synthesized by appropriately encoding spatial cues corresponding to the desired target scene, which is suitable for headphone production. In practice, headphone binaural audio production suffers from in-head localization and poor frontal imaging [3], while playback through loudspeakers is largely immune to these problems. In addition, compared with headphone reproduction, cues by the involvement of the listener's own head, torso and pinnae in sound diffraction and reflection during playback can enhance the perceived realism of sound reproduction [4]. When the binaural audio is reproduced through loudspeakers, it suffered from a problem of so-called "crosstalk" component of the signals, i.e. the component of the signal for right ear fed to the left ear and vice versa, which severely destroys the 3D spatial information for the listener.

Ideally, the expected signals obtained at the listener's ears are delayed copies of the input binaural signals. To suppress, if not totally eliminate, the unintended crosstalk, in mathematics, it boils down to designing a crosstalk cancellation matrix to approximate the inversion of the transfer function matrix. Since the concept of crosstalk cancellation was introduced in 1960s [5, 6], many studies with the aim of minimizing the crosstalk were extensively investigated [7, 8]. In terms of design of crosstalk cancellation filters, different algorithms have been proposed. In the time domain, the least mean square (LMS) algorithm [7] and its variations [9, 10] are the predominant ones. In contrast to the time-domain method that is time consuming for long filters, the fast frequency-domain deconvolution method offers more advantage in terms of computational speed and is also widely used [8]. Thus far, all of the above-mentioned crosstalk cancellation methods employ the LMS optimization technique. In [11] a method based on a minimax design criterion is proposed, and its solution is obtained by utilizing second-order cone programming (SOCP) techniques. Although it achieves excellent channel separation, especially at low frequencies, its huge computational cost limits its practical applications. In addition, to efficiently implement the crosstalk cancellation system, a number of filter topologies, such as recursive and shuffler form [3, 12, 13], are also presented. Generally, the crosstalk cancellation system is optimized to achieve optimum cancellation at a given transfer function matrix corresponding to a nominal listener's position. However, in practical applications, many factors that disturb the transfer function matrix are unavoidable, such as tiny movements or rotations of the listener's head, noise, etc. All these disturbances or errors have adverse effects on crosstalk cancellation system (CCS), especially when CCS is ill-conditioned. The inverse filter is very sensitive to small errors in the transfer function matrix and may reproduce large distortions in the filter's output. To improve the robustness of CCS, a circular or linear array is suggested using a multi-band approach processing band-passed content to appropriately spaced loudspeakers [14, 15].

However, even with such multiple loudspeakers reproduction, design of the crosstalk cancellation filters with some inherent improved robustness is still necessary. This raises the need for dealing with improving the robustness of crosstalk cancellation system against slight disturbances or errors. When the transfer function between the loudspeakers to the ears is characterized by the room impulse response, they are very are very sensitive to spatial mismatch. Under certain circumstances, the transfer function is a stochastic one. In [16], a spatial robust crosstalk cancellation method is proposed in the case of far-field in reverberant environments. Further, a method that jointly handles the three problems of crosstalk, reverberation reduction, and spatial robustness with respect to varying listening positions was proposed in [17].

* This author is also with Xinjiang Laboratory of Minority Speech and Language Information Processing, Chinese Academy of Sciences, Urumqi, Xinjiang

Still, due to the fact that crosstalk cancellation of room impulse response in a reverberant environment is extremely non-robust, practical crosstalk cancellation system are commonly designed to cancel only the direct-path transfer functions. In this paper, the aim of this study attempts to model of the disturbance of transfer function itself due to movement of the listener's head from a statistical view. A random variable matrix is introduced to characterize the variations of the transfer function matrix between the loudspeakers and the listener's ears on the basis of statistical modeling. Then the traditional crosstalk cancellation problem turns into stochastic robust approximation problem [18]. In this framework, joint least squares optimization crosstalk cancellation method [19, 20] that take multiple positions into account can be treated as special cases, when the transfer function matrix is subject to discrete distribution. Simulation results demonstrate that this method can improve the robustness of crosstalk cancellation, especially when the nominal transfer function matrix is ill-conditioned.

2. CROSSTALK CANCELLATION FORMULATION

This section presents an overview of well-established material about crosstalk cancellation. Fig. 1 shows a geometry diagram of the implementation of crosstalk cancellation system under investigation, in which p_L and p_R denote the left and right input audio signal, respectively, and h_n^L; n = 1, 2; represent the impulse response (IR) from the nth loudspeaker to the left ear (a similar pair of IRs for the right ear, for concision, are not shown).

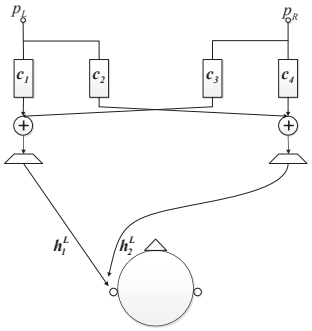


Figure 1: *Geometry diagram for a typical crosstalk cancellation system.*

Considering reproducing only the left audio signal, i.e., p_R = 0, in matrix form, it can be written as

$$\begin{bmatrix} \hat{b}_L \\ \hat{b}_R \end{bmatrix} = \begin{bmatrix} A_1^L & A_2^L \\ A_1^R & A_2^R \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \tag{1}$$

where $\hat{b}_L$ and $\hat{b}_R$ are the transfer functions between p_L and the listener's ears, respectively, and A_1^L is a convolution matrix, which is expressed as

$$A_1^L = \begin{bmatrix} h_1^L(0) & 0 & \cdots & 0 \\ h_1^L(1) & h_1^L(0) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & h_1^L(M-1) \end{bmatrix} \tag{2}$$

and similarly for A_2^L ; A_1^R and A_2^R, and c_1, c_2 are the corresponding crosstalk cancellation filter coefficients vectors. A more simplified form in matrix can be expressed as

$$Ac = b \tag{3}$$

where the transfer function matrix A is composed of A_1^L, A_2^L; A_1^R and A_2^R.

The widely used criterion for crosstalk cancellation is least mean squares (LMS), which minimizes the squared distance between the set of desired input signals and the actual obtained signals at the listener's ears. In our case, b_L is a pure delay, and b_R is a zero vector. For a given head position, CCS filter coefficients can be solved by

$$J_0(c) = \|b - Ac\|_2^2 \tag{4}$$

The optimum filter coefficients are then expressed as

$$c_{opt} = \arg\min_c J_0(c) = A^\dagger b \tag{5}$$

where $A^\dagger = (A^T A)^{-1} A^T$ is the Moore-Penrose generalized inverse of real-valued A. Obviously, such crosstalk cancellation system is only effective when the listener is in the prescribed position and the so-called "sweet spot" is small.

3. PROPOSED STOCHASTIC ROBUST CROSSTALK CANCELLATION METHOD

In practice, the transfer function matrix A is unavoidably influenced by some perturbations and errors due to misalignments, tiny head movement. etc. In this section, we consider the statistical model for the variations in A from statistics point of view.

3.1. Stochastic Robust Approximation

Assuming that A is a random variable matrix taking values in $R^{m\times n}$ with mean $\bar{A}$, A can be expressed as $A = \bar{A} + U$, where U is a random matrix with zero mean. Here, the constant matrix $\bar{A}$ represents the average value of A, and U characterises its statistical variation. Naturally, employing the expected value as the objective function, we can get

$$\arg\min_c E\{\|Ac - b\|\} \tag{6}$$

where E represents the mathematical expectation. This problem is referred to as the stochastic robust approximation problem [21]. When A is a discrete random variable with only a finite number of values, i.e

$$prob(A = A_i) = p_i, i = 1, ..., k \tag{7}$$

where $prob$ means the probability of different $A_i \in R^{m\times n}$, $\mathbf{1}^T p = 1$, $p \succeq 0$, the problem turns into

$$\arg\min_c (p_1\|A_1 c - b\| + ... + p_k\|A_k c - b\|) \tag{8}$$

Therefore, both the joint multi-position optimization [19] and multi-position weighted optimization [20] for crosstalk cancellation can be seen as special cases of the stochastic robust approximation, given by equation (8). Considering the LMS norm, the stochastic LMS method for crosstalk cancellation can be described as

$$\arg\min_c E\{\|Ac - b\|_2^2\} \tag{9}$$

Further, it can be expanded as

$$\begin{aligned} E\{\|Ac-b\|_2^2\} &= E\{(\bar{A}c-b+Ux)^T(\bar{A}c-b+Ux)\} \\ &= (\bar{A}c-b)^T(\bar{A}c-b)+E\{c^TU^TUc\} \\ &= \|\bar{A}c-b\|_2^2+c^TPc \end{aligned} \tag{10}$$

where $P = E\{U^TU\}$ corresponds to mathematical expectation of the autocorrelation matrix of the perturbation matrix U. Therefore the statistical robust approximation problem shows a similar form with the regularized least-squares method [22]

$$\arg\min_c \|\bar{A}c-b\|_2^2+\|P^{1/2}c\|_2^2 \tag{11}$$

with analytical solution

$$c_{opt} = (\bar{A}^T\bar{A}+P)^{-1}\bar{A}^Tb \tag{12}$$

3.2. Modeling Random Perturbation

In the following, the variations of the transfer function are modeled in a way to improve the spatial robustness of crosstalk cancellation from a statistical point of view. Without loss of generality, the perturbation $\xi_i^L(i=1,2)$ on the transfer function from the loudspeakers to listener's left ear is modeled as a statistical variable with zero mean and variance $\sigma_i^L(i=1,2)$ (modeling $\xi_i^R(i=1,2)$ similarly). Then, the perturbed transfer function is expressed as $u_i^L = \xi_i^L h_i^L(i=1,2)$. Further, the perturbation matrix U is denoted as

$$U = \begin{bmatrix} \xi_1^L\bar{A}_1^L & \xi_2^L\bar{A}_2^L \\ \xi_1^R\bar{A}_1^R & \xi_2^R\bar{A}_2^R \end{bmatrix} \tag{13}$$

The expectation matrix P of autocorrelation of the perturbation matrix U is expressed as

$$\begin{aligned} P &= E\{U^TU\} \\ &= E\{\begin{bmatrix} \xi_1^L\bar{A}_1^L & \xi_2^L\bar{A}_2^L \\ \xi_1^R\bar{A}_1^R & \xi_2^R\bar{A}_2^R \end{bmatrix}^T \begin{bmatrix} \xi_1^L\bar{A}_1^L & \xi_2^L\bar{A}_2^L \\ \xi_1^R\bar{A}_1^R & \xi_2^R\bar{A}_2^R \end{bmatrix}\} \\ &= \begin{bmatrix} P_1^L & P_2^L \\ P_1^R & P_2^R \end{bmatrix} \end{aligned} \tag{14}$$

where $P_1^L = E\{(\xi_1^L)^2(\bar{A}_1^L)^T(\bar{A}_1^L)+(\xi_1^R)^2(\bar{A}_1^R)^T(\bar{A}_1^R)\}$, P_2^L, P_1^R, P_2^R denoted similarly.

Due to its uncertainty in practice, it's reasonable to further assume that all the perturbation random variables independent and identically distributed (IID) with zero mean and variance σ, and then the antidiagonal block elements P_2^L, P_1^R of the P matrix are reduced to zeros. Finally, the P matrix is expressed as

$$P = \sigma^2\begin{bmatrix} (\bar{A}_1^L)^T\bar{A}_1^L+(\bar{A}_1^R)^T\bar{A}_1^R & \mathbf{0} \\ \mathbf{0} & (\bar{A}_2^L)^T\bar{A}_2^L+(\bar{A}_2^R)^T\bar{A}_2^R \end{bmatrix} \tag{15}$$

4. EXPERIMENTAL VERIFICATION AND ANALYSIS

In this section, the performance of the proposed stochastic LMS method is compared with the traditional LMS method by simulations under free-field condition.

4.1. Performance Metrics and Experimental Setup

To analyze the crosstalk cancellation performance, the channel separation (CHS) is adopted as the evaluation measure, which is defined as the ratio between the desired signal and the crosstalk signal. In our case, owing to set the input right signal zeros in prior, the signal received by the listener's left ear is the desired signal and the signal received by the listener's right ear is the crosstalk. There, the channel separation is expressed as

$$CHS(k) = 20\log|\frac{b_L(k)}{b_R(k)}| \tag{16}$$

where k denotes different discrete frequencies. The average channel separation is defined as

$$\overline{CHS} = \frac{1}{n_L-n_H+1}\sum_{k=n_L}^{n_H} CHS(k) \tag{17}$$

where n_L and n_H are the entire frequency ranges of interest.

In order to verify the robustness of the proposed method, the slight movement of the listener's head is selected as the perturbation factor among all the perturbation factors. The average channel separations of different listener's head positions are calculated and compared with traditional LMS method. According to human auditory characteristics, usually, the interaural level difference (ILD) servers as a predominant cue at frequencies below 5 kHz, while in higher frequencies, the listener's head, especially the pinna have a dominant effect in sound localization. Due to free-field condition without consideration of the listener's head effects, the frequency range of computing average channel separation is selected between 200-5000 Hz and the frequency sample is selected as 16 kHz. Fig. 2 illustrates the schematic diagram of the listener's head movement. The loudspeakers are separated by a distance of $d_s = 0.1$ m

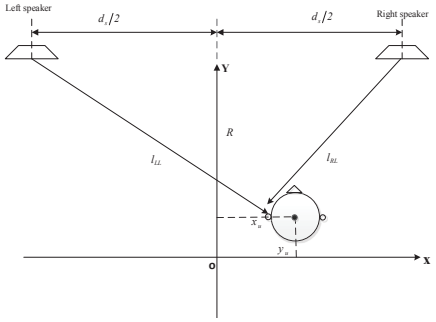


Figure 2: *The schematic of listener's head movement in experiment.*

and an $\theta = 10^\circ$ from the default listening position (with the head placed symmetrically between the loudspeakers) corresponding to the "stereo dipole" configuration [23]. The vertical distance R between the center of the nominal listener's head position to the two speakers is 0.5 m. Because of fundamental difficulties in achieving good crosstalk cancellation at low frequencies, the desired $b_L(n)$ was designed as an unit impulse response filtered by a high-pass filter with a cut-off frequency of 200 Hz. The optimum delay for crosstalk cancellation is calculated according to the rule suggested in [19]. The region x_u for listener's head slight movement is chosen between (1, 2, 3, 4) cm corresponding to the head movement

towards the right (the cancellation is more effective as the head moves forwards/backwards than if it moves sideways [24] and further, due to the symmetry only consider the right movement and set y_u = 0). In free field, the transfer function (for example, the left speaker to the left ear) in frequency domain is expressed as

$$H(w) = \frac{1}{4\pi l_{LL}} e^{-jkl_{LL}} \tag{18}$$

where l_{LL} is the distance from the left loudspeaker to the listener's left ear, $k = w/c$ is the wave number and c is the sound speed, set by 340 m/s.

4.2. Analysis of Experimental Result

For proposed stochastic LMS method, the optimal proper σ of the perturbation needs to be determined in advance. A series of tests were conducted by changing the variance σ range (0.01-1) with interval 0.005. For $\theta = 10^\circ$, the average channel separations designed according to the traditional LMS method (solid line) and stochastic LMS method (dashed line) with different σ are shown in Fig. 3. In Fig. 3, the lines from top to bottom describe different head movement positions from x_u = 1 cm to x_u = 4 cm with different line styles represent different methods.

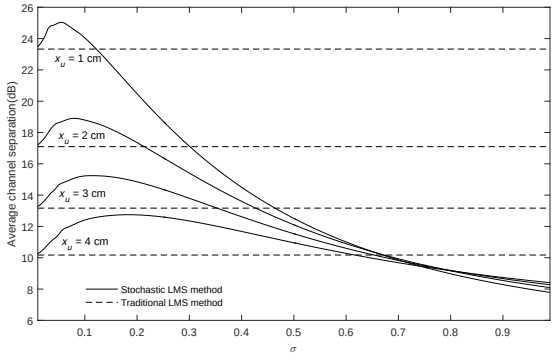


Figure 3: *Comparison of the average CHSs at different head positions: from top to bottom x_u = 1 cm, 2 cm, 3 cm, 4 cm for $\theta = 10^\circ$ with different σ (solid line for the stochastic LMS method and dashed line for the traditional LMS method).*

Strictly speaking, for the traditional LMS method, the average channel separation of different head movement is a specific value and does not vary as the σ varies, which is depicted as a straight line. It's clearly shown from the Fig. 3 that in the vicinity of σ = 0.1, the average channel separation of the proposed stochastic LMS method is higher than the corresponding traditional LMS method, demonstrating that the proposed method is robust against the listener's slight movement. For clearly showing the improvement, the listener's ear responses are drawn in Fig. 4 with different head movements. Ideally, the left ear response should be unity (above 200 Hz) and the right ear response should be zero. As shown from the Fig. 3 and Fig. 4, compared with the traditional LMS method, introducing the perturbation brings improved channel separation in the vicinity of 1000 Hz.

Under the free-field condition, the analysis of the transfer function matrix revealed that, for a given loudspeaker angle, its robust frequency range is determined by the "ring frequency"(RF), which is inversely proportional to the angle [23]. It indicates that the crosstalk cancellation is inherently non-robust in the frequency range above the RF. The RF of the "stereo dipole" is about 11 kHz, which is beyond the scope of the frequency (8 kHz) considered in our experiment. For further comparison, the loudspeaker angle is increased to 20°, where the RF is about 5.6 kHz and repeat the experiment with other parameters keeping the same. Similar to Fig. 3, for $\theta = 20^\circ$, the the average channel separation designed according to the traditional LMS method and stochastic LMS method with different variance σ and head positions are shown in Fig. 5.

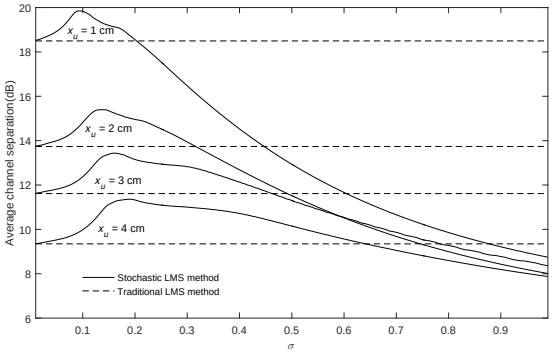
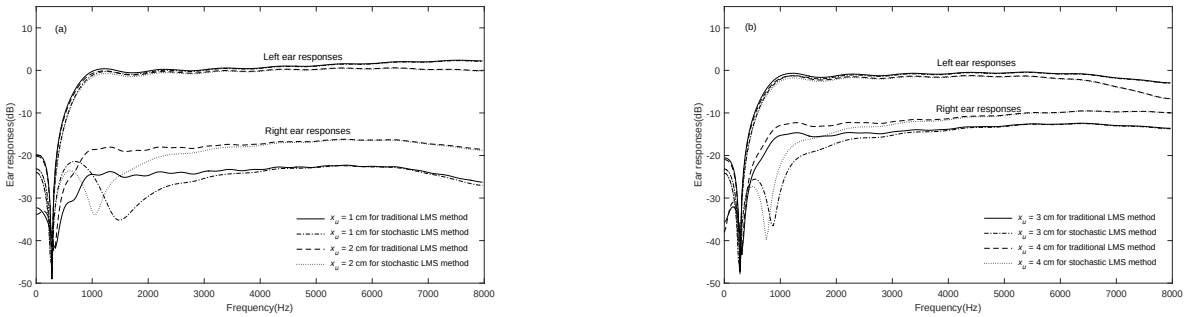


Figure 5: *Comparison of the average CHSs at different head positions: from top to bottom x_u = 1 cm, 2 cm, 3 cm, 4 cm for $\theta = 20^\circ$ with different σ (solid line for the stochastic LMS method and dashed line for the traditional LMS method).*

As can be seen from Fig. 5, in the vicinity of σ = 0.15, the average channel separation of the proposed stochastic LMS method is still higher than the traditional LMS method, which shows good agreement with the first experiment. When the variance σ = 0.15, the listener's ear responses are drawn in Fig. 6 with different head movement. From discussions described above, there exists non-robust frequency point corresponding the "ring frequency" around 5000 Hz. The perturbation introduced by the listener's slight head movement results in the rapid decrease of its performance and the spectral distortion. The proposed stochastic LMS method not only improves channel separation in the vicinity of 1000 Hz, but also greatly reduces the spectral distortion around the "ring frequency". This again confirms the expectation that proposed stochastic approximation crosstalk cancellation method provides an enhanced robustness.

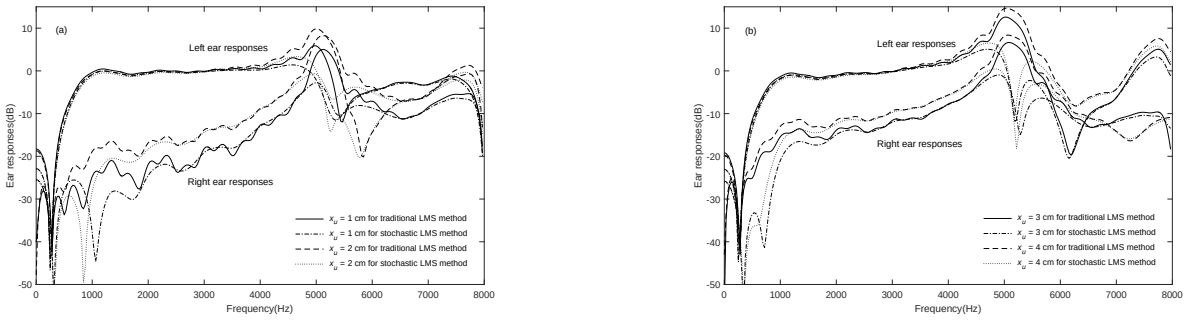
5. CONCLUSION

A novel stochastic LMS crosstalk cancellation method based on statistical modeling is proposed for the designing of crosstalk cancellation system. A random perturbation matrix modeling the variations of the transfer functions due to perturbations is introduced and lied in parallel to the actual nominal transfer matrix during driving the crosstalk cancellation filters. Under the free-field condition, simulation results proved that the proposed method is robust against listener's slight head movement.



(a) (b)

Figure 4: *Ear responses at head positions with different line styles representing different methods for $\theta = 10^\circ$: (a) x_u = 1 cm, 2 cm; (b) x_u = 3 cm, 4 cm.*



(a) (b)

Figure 6: *Ear responses at head positions with different line styles representing different methods for $\theta = 20^\circ$: (a) x_u = 1 cm, 2 cm; (b) x_u = 3 cm, 4 cm.*

6. ACKNOWLEDGMENTS

This work is partially supported by the National Natural Science Foundation of China (Nos. 11461141004, 61271426, 11504406, 11590770, 11590771, 11590772, 11590773, 11590774), the Strategic Priority Research Program of the Chinese Academy of Sciences (Nos. XDA06030100, XDA06030500, XDA06040603), National 863 Program (No. 2015AA016306), National 973 Program (No. 2013CB329302) and the Key Science and Technology Project of the Xinjiang Uygur Autonomous Region (No. 201230118- 3).

7. REFERENCES

[1] Yiteng Huang, Jingdong Chen, and Jacob Benesty, "Immersive audio schemes," *Signal Processing Magazine, IEEE*, vol. 28, no. 1, pp. 20–32, 2011.

[2] Bosun Xie, *Head-related transfer function and virtual auditory display*, J. Ross Publishing, 2013.

[3] William G Gardner, *3-D audio using loudspeakers*, vol. 444, Springer Science & Business Media, 1998.

[4] Edgar Y Choueiri, "Optimal crosstalk cancellation for binaural audio with two loudspeakers," *Princeton University*, p. 28, 2008.

[5] Benjamin B Bauer, "Stereophonic earphones and binaural loudspeakers," *Journal of the Audio Engineering Society*, vol. 9, no. 2, pp. 148–151, 1961.

[6] Bishnu S Atal and Manfred R Schroeder, "Apparent sound source translator," Feb. 22 1966, US Patent 3,236,949.

[7] Philip A Nelson, Hareo Hamada, and Stephen J Elliott, "Adaptive inverse filters for stereophonic sound reproduction," *Signal Processing, IEEE Transactions on*, vol. 40, no. 7, pp. 1621–1632, 1992.

[8] Ole Kirkeby, Philip A Nelson, Hareo Hamada, and Felipe Orduna-Bustamante, "Fast deconvolution of multichannel systems using regularization," *Speech and Audio Processing, IEEE Transactions on*, vol. 6, no. 2, pp. 189–194, 1998.

[9] Jong-Soong Lim and Chris Kyriakakis, "Multirate adaptive filtering for immersive audio," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*. IEEE, 2001, vol. 5, pp. 3357–3360.

[10] Athanasios Mouchtaris, Panagiotis Reveliotis, and Chris Kyriakakis, "Inverse filter design for immersive audio rendering over loudspeakers," *Multimedia, IEEE Transactions on*, vol. 2, no. 2, pp. 77–87, 2000.

[11] Harsha IK Rao, V John Mathews, and Young-Cheol Park, "A minimax approach for the joint design of acoustic crosstalk

cancellation filters," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 8, pp. 2287–2298, 2007.

[12] Makoto Iwahara and Toshinori Mori, "Stereophonic sound reproduction system," Oct. 3 1978, US Patent 4,118,599.

[13] Duane H Cooper and Jerald L Bauck, "Prospects for transaural recording," *Journal of the Audio Engineering Society*, vol. 37, no. 1/2, pp. 3–19, 1989.

[14] Takashi Takeuchi and Philip A Nelson, "Optimal source distribution for binaural synthesis over loudspeakers," *The Journal of the Acoustical Society of America*, vol. 112, no. 6, pp. 2786–2797, 2002.

[15] Jianwen Zheng, Tianyi Zhu, Jing Lu, and Xiaojun Qiu, "A linear robust binaural sound reproduction system with optimal source distribution strategy," *Journal of the Audio Engineering Society*, vol. 63, no. 9, pp. 725–735, 2015.

[16] Markus Kallinger and Alfred Mertins, "A spatially robust least squares crosstalk canceller," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*. IEEE, 2007, vol. 1, pp. I–177.

[17] Jan Ole Jungmann, Radoslaw Mazur, Markus Kallinger, Tiemin Mei, and Alfred Mertins, "Combined acoustic mimo channel crosstalk cancellation and room impulse response reshaping," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 6, pp. 1829–1842, 2012.

[18] Stephen Boyd and Lieven Vandenberghe, *Convex optimization*, Cambridge university press, 2004.

[19] Darren B Ward, "Joint least squares optimization for robust acoustic crosstalk cancellation," *Speech and Audio Processing, IEEE Transactions on*, vol. 8, no. 2, pp. 211–215, 2000.

[20] Wang Jie, Ye Qing-hua, Zheng Cheng-shi, and Li Xiao-Dong, "A robust algorithm for binaural audio reproduction using loudspeakers," in *Measuring Technology and Mechatronics Automation (ICMTMA), 2010 International Conference on*. IEEE, 2010, vol. 1, pp. 318–321.

[21] Laurent El Ghaoui and Hervé Lebret, "Robust solutions to least-squares problems with uncertain data," *SIAM Journal on Matrix Analysis and Applications*, vol. 18, no. 4, pp. 1035–1064, 1997.

[22] Ole Kirkeby and Philip A Nelson, "Digital filter design for virtual source imaging systems," in *Audio Engineering Society Convention 104*. Audio Engineering Society, 1998.

[23] Ole Kirkeby, Philip A Nelson, and Hareo Hamada, "The "stereo dipole" : A virtual source imaging system using two closely spaced loudspeakers," *Journal of the Audio Engineering Society*, vol. 46, no. 5, pp. 387–395, 1998.

[24] Darren B Ward and Gary W Elk, "Optimum loudspeaker spacing for robust crosstalk cancellation," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*. IEEE, 1998, vol. 6, pp. 3541–3544.

SIMULATION OF ANALOG FLANGER EFFECT USING BBD CIRCUIT

Jaromír Mačák
Independent researcher
Brno, Czech Republic
Jarda.macak@seznam.cz

ABSTRACT

This paper deals with simulation of BBD circuit based analog flanger effects. The famous Electro-Harmonix Deluxe Electric Mistress flanger effect was used as a case study in this paper. The main attention of this paper is paid to the analysis and simulation of the LFO circuit, the BBD clock generator circuit and BBD circuit simulation of this effect. However, in order to compare the simulation results with measured data, the signal path simulation using the DK-method has been introduced as well.

1. INTRODUCTION

A delay line is an essential part of many audio effects, foremost delay effects and some modulation effects like chorus and flanger. The construction of these audio effects has been known since late 1960s when the bucket-brigade device (BBD) was invented by F. Sangster in Philips Research Labs [1, 2]. The BBD is basically an analog shift register to shift the electrical charge, representing the input signal, along internal capacitors. The resulting delay time is given by the number of the internal capacitors and by the period of the clock signal which is used to control the shifting the charge inside the BBD. The clock signal could be modulated by a low frequency oscillator (LFO) in order to obtain a time-variable delay typical for chorus and flanger audio effects.
Although the BBD has been still used especially for analog guitar effect pedals, digital implementations of modulation effects became more popular. The delay line is realized by a circular buffer in the memory where the time delay is given by the length of the circular buffer. Because the simple circular buffer provides only a discrete time delays equal to an integer multiple of the sampling period, fractional delay lines have been introduced for audio effects which require any value of delay time to work properly – e.g. modulation effects where the delay time is modulated by the digital LFO [3]. The fractional delay line is implemented using the circular buffer and the neighboring samples in the circular buffer are interpolated in order to obtain the time delay of any value. Linear interpolation, spline interpolation or finite impulse response (FIR) filter are often used as interpolators with different sonic qualities and computational demands [3]. The latest generation of the digital audio effects however tents to emulate sonic qualities of their analog models by simulation of their electric circuits in real time. Many types of such audio effects were simulated during last years, foremost guitar amplifiers with tubes, guitar distortion pedals, guitar compressor pedals, guitar wah pedals, etc. The audio delay effects with the BBD are the main topic in paper [4]. The general structure of such effects is described and further, simulation of some parts of these effects is introduced there – namely antialiasing and reconstruction filters and compandor circuit used in analog delay guitar pedals. The BBD circuit is simulated using the black box approach – by measurement of frequency response and harmonic distortion and their simulation using a simple digital filter and waveshaping technique although an approach for the BBD circuit has been proposed. Because the authors dealt foremost with delay audio effect analysis and simulation where the delay time is fixed and the clock signal generation unit has not been described there, foremost the simulation of the clock generation circuit and a simple model of the BBD circuit are described in this paper.

2. CIRCUIT ANALYSIS AND SIMULATION

The Electro-Harmonix Deluxe Electric Mistress effect has been used as case study in this paper. Although it is possible to perform the simulation of the whole circuit, this approach is not suitable for real time simulation due to high computational demands and therefore a division into separate functional units is used.

2.1. Low frequency oscillator

The low frequency oscillator circuit very often comprises of an integrator and a comparator circuits connected in a feedback loop. The circuit schematic of the LFO is shown in Figure 1. The simulation of this circuit using the DK method is fully described in [6]. This circuit generates a triangle signal with the frequency given by R_1, R_2, R_3 and C_1 values. The R_1 resistor is often a potentiometer to control the LFO speed.

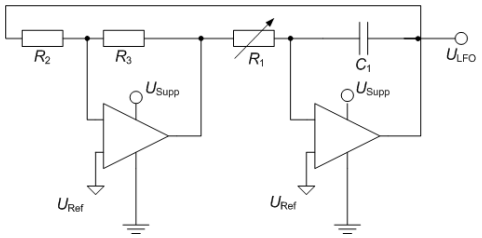


Figure 1: *Typical LFO circuit.*

The first operational amplifier (OPA) is the non-inverting comparator with hysteresis

$$U_{\mathrm{H}} = 2U_{\mathrm{Supp}}\frac{R_2}{R_3} \tag{1}$$

and the second OPA serves as the integrator with the output

$$U_{\mathrm{LFO}}(t) = U_{\mathrm{Ref}} - \frac{1}{R_1 C_1}\int \pm\left(U_{\mathrm{Supp}} - U_{\mathrm{Ref}}\right)dt, \tag{2}$$

giving, together with the comparator, the integration time

$$t_{\mathrm{Int}} = 2\frac{R_1 R_2 C_1}{R_3} \tag{3}$$

and LFO frequency $f = \frac{1}{2t_{\mathrm{Int}}}$. U_{Supp} is the OPA power supply and U_{Ref} is the virtual zero voltage offset.
The digital simulation of this circuit might be implemented according to

$$U_{\mathrm{LFO}}[n+1] = U_{\mathrm{LFO}}[n] + s\frac{U_{\mathrm{Supp}}}{R_1 C_1 T_{\mathrm{s}}}, \tag{4}$$

where T_s is the sampling period, boundary condition $U_{\mathrm{LFO}}[0] = U_{\mathrm{Ref}}$ and $s = \pm 1$. The sign s is changed to the opposite value every time when the LFO output signal exceeds the comparator thresholds

$$U_{\mathrm{LFO}}[n+1] > U_{Ref} + \left(U_{\mathrm{Supp}} - U_{\mathrm{Ref}}\right)\frac{R_2}{R_3} \tag{5}$$

or

$$U_{\mathrm{LFO}}[n+1] < U_{Ref} - \left(U_{\mathrm{Supp}} - U_{\mathrm{Ref}}\right)\frac{R_2}{R_3}. \tag{6}$$

2.2. Clock generation unit

The BBD chip cannot operate only by itself and requires clock signals to control the BBD chip provided by the clock generation unit. In contrast to the LFO circuit, this circuit is often unique for audio effect manufacturers and for specific audio effects. However some general characteristic can be found as well. A triangle signal from the LFO is typically filtered by a low pass filter to shape the LFO signal to a sine wave by filtering higher harmonics. The filtered LFO signal is used to control an astable oscillator generating the clock signal. It could be either an integrated circuit (e.g. MN3101 with an internal or an external oscillator) or it is also possible to generate the clock signal using discrete circuit parts – an astable circuit and a flip-flop chip generating two phase opposite signals as can be seen in Figure 2 which shows the clock generation unit of the Electro-Harmonix Deluxe Electric Mistress effect [5].

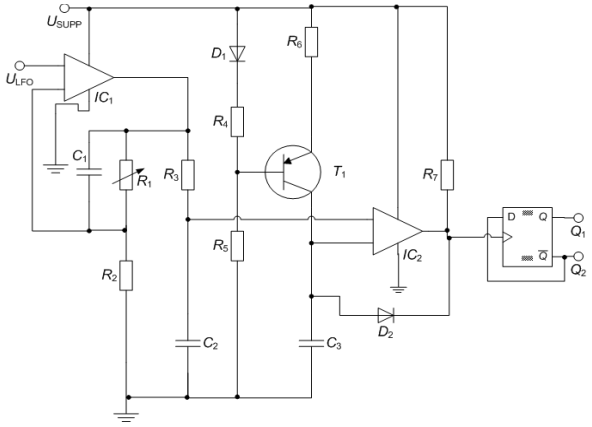


Figure 2: *Clock generator circuit of Electro-Harmonix Deluxe Electric Mistress effect.*

The input low pass filter consists of OPA IC_1, potentiometer R_1, resistors R_2, R_3 and capacitors C_1 and C_2. The transfer function of the low pass filter is given by

$$H(p) = \frac{C_1R_1p+1}{C_1C_2R_1R_3p^2+(C_2R_3+C_1R_1)p+1} \tag{7}$$

and a corresponding digital filter can be designed using the bilinear transform of the transfer function (7).

The second part of the circuit is the clock generator. The main principle is based on charging and discharging of the capacitor C_3. The capacitor is charged via collector current from the transistor T_1 until the capacitor voltage is equal to threshold voltage from the low pass filter, then the comparator IC_2 is switched to low output value and the capacitor C_3 is immediately discharged via the high speed diode D_2 and the comparator output switches back to the original high output value. The whole process is being repeated as can be seen from Figure 3. Simulation of the circuit could be solved in real time like other audio circuit using an appropriate method (e.g. DK-method) but duration of transients is shorter than a typical sampling period and therefore the simulation would have to run at very high sampling frequencies. The most important information is not however the signal itself but rather the time when the signal reaches the threshold value. Supposing an approximation of the step response of the simulated circuit with a step response of a passive RC network given by

$$U_{\mathrm{THD}} = U_{\mathrm{Max}}\left(1 - e^{-\frac{t}{\tau}}\right), \tag{8}$$

with approximation parameters τ (equal to the time constant), U_{Max} (equal to the max voltage which can be reached) and U_{THD} (the voltage to which the capacitor should be charged), the unknown time t can be expressed using

$$t = -\tau \log\left(\frac{U_{\mathrm{Max}} - U_{\mathrm{THD}}}{U_{\mathrm{Max}}}\right), \tag{9}$$

on the condition that the U_{THD} voltage is constant or is being changed very slowly during the integration. For slowly changing voltage U_{THD}, a linear interpolation can be used to formulate the equation

$$\frac{U_{\mathrm{THD}}(t_0+t) - U_{\mathrm{THD}}(t_0)}{t} = U_{\mathrm{Max}}\left(1 - e^{-\frac{t}{\tau}}\right), \tag{10}$$

where t_0 is the boundary condition and t the unknown integration time. Because the time t cannot be simply isolated from equation (10), a numerical method must be used to get the integration time.

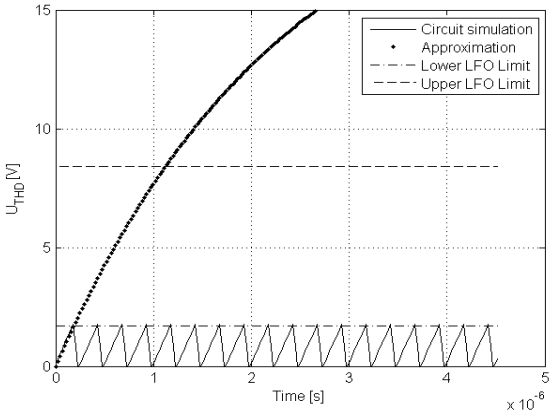


Figure 3: *Step response of the astable circuit and its approximation.*

The astable oscillator output signal is connected to the clock input of the flip-flop chip which reacts on the rising edge, the data port of the flip flop chip is connected to the output port Q_2 in order to flip the output to the opposite state, generating two opposite clock signals for the BBD with the period equal to

$$T_{\mathrm{Clock}} = 2t \tag{11}$$

at Q_1 and Q_2 output ports.

2.3. BBD simulation

There have been used more types of BBD chips from different manufactures (e.g. MNxxx chips made by Panasonic and SADxxx manufactured by Reticon company) in guitar effects. Basically, the internal structure is very similar, consisting of a series of MOS transistors used as switches and capacitors used to hold the electric charge. The simplified circuit topology is shown in Figure 4 [4]. The first transistor and capacitor are used for sampling of an input signal. The rest of the circuit consist of charge holding elements which are separated by DC biased gates enhancing the charge transfer [2]. The neighboring charge holding elements are controlled using two anti-phase clock signals in such way that neighboring switches are in opposite states in order to pass the electric charge only into a subsequent stage.

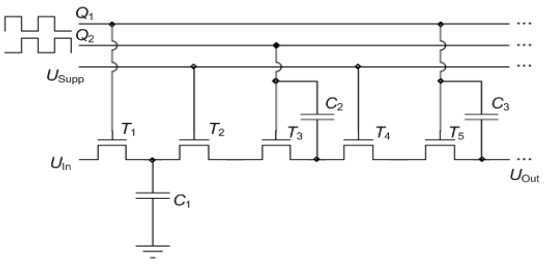


Figure 4: *Simplified BBD circuit topology.*

The crucial difference between the BBD delay line and the variable circular buffer used for implementation of a delay line in digital effects is that the BBD has a fixed number of cells. In case of variable circular buffer, the desired delay is obtained using variable number of delay cells but the input and output signal samples are written and read with a constant sampling period. On the contrary, a variable sampling period is required to obtain the desired delay from the BBD delay line. It means that for the digital implementation of the BBD delay line we have to match the digital effect sampling period with the BBD sampling period. An oversampling can be used to match the sampling periods for static delay times, as has already been suggested in [4]. However, more challenging is when the BBD sampling period is modulated by the LFO. In this case, there is no constant relation to match the sampling periods and an interpolation of the input and output signals is required because the BBD sampling can occur at any time between the input and output signal samples. An algorithm based on this principle has been designed and its flowchart describing processing of one signal sample is shown in Figure 5. The algorithm uses one variable `tTemp` to store the time increments. This variable is initialized to the BBD clock period obtained from (9).While it is greater than a sampling period T_s, it is decremented by T_s and output signal samples are read from the BBD. If the `tTemp` is less than T_s, the new input sample for the BBD is acquired, signal samples stored in the BBD delay line are shifted and new clock period is computed from (9) and added to `tTemp`. Because the BBD sampling occurs between audio signal samples, linear interpolation is used, as shown in Figure 5. The samples in the delay cells are shifted according to $bbd[2n+m+1] = bbd[2n+m]$ for $n \in \langle 0, \frac{N}{2}-1\rangle$ and $m = \mathrm{modulo}_2(m+1)$ where N is number of cells of the BBD chip (512 for SAD1024 BBD used in this effect) and *bbd* is the delay line to store the samples.

The model might be further extended with integrated filters simulating frequency dependent electric charge transfer between the cells. The filtration can be done e.g. using $bbd[2n+m+1] \mathrel{+}= \frac{T_{Clock}}{\tau}(bbd[2n+m] - bbd[2n+m+1])$ with T_{Clock} given by (9) and τ is filter time constant. Ideally, each cell transfer should be further process by a nonlinear function to simulate the transistors nonlinearities.

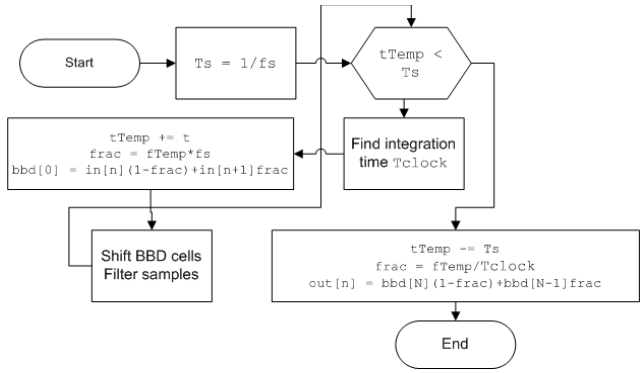


Figure 5: *BBD chip simulation algorithm –processing for one audio sample.*

Both nonlinear processing and filtration for each cell transfer however introduce very high computational demands when real-time processing is considered and both can be replaced by the BBD output sample filtering and nonlinear processing with characteristic e.g. according to paper [4].

2.4. Main signal path

Figure 6 shows the direct signal path consisting of the summation amplifier, antialiasing and reconstruction filters and SAD1024 BBD. The BBD line is connected to the nodes U_{BBDin} and U_{BBDout}. The whole signal path circuit except the BBD circuit can be simulated using different approaches. However, the challenge is the instantaneous feedback causing the flanger effect. The approach using the division of the signal path circuit into subcircuits would require adding a unit delay into the global feedback loop changing the overall frequency response of the circuit, as it was already shown in [6] where the unit delay was used in the tube guitar power amplifier simulation. Therefore simulation of the whole signal path circuit as one block is used. The nodal DK method with incidence matrices defined in [7] is a suitable method. Because the circuit contains operational amplifiers, an extension of the DK method introduced in [6] is used. The whole circuit can be described by the conductance matrix

$$\boldsymbol{S} = \begin{pmatrix} \boldsymbol{N}_{\mathrm{R}}^{\mathrm{T}}\boldsymbol{G}_{\mathrm{R}}\boldsymbol{N}_{\mathrm{R}} + \boldsymbol{N}_{\mathrm{X}}^{\mathrm{T}}\boldsymbol{G}_{\mathrm{X}}\boldsymbol{N}_{\mathrm{X}} & \boldsymbol{N}_{\mathrm{u}}^{\mathrm{T}} & \boldsymbol{N}_{\mathrm{OPA-O}}^{\mathrm{T}} \\ \boldsymbol{N}_{\mathrm{u}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{N}_{\mathrm{OPA-O}} + \boldsymbol{A}\boldsymbol{N}_{\mathrm{OPA-I}} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \quad (12)$$

where $\boldsymbol{N}_{\mathrm{R}}$, $\boldsymbol{N}_{\mathrm{X}}$, $\boldsymbol{N}_{\mathrm{u}}$, are incidence matrices, $\boldsymbol{G}_{\mathrm{R}}$ and $\boldsymbol{G}_{\mathrm{X}}$ resistors and capacitors conductance matrices all defined in [7] and $\boldsymbol{N}_{\mathrm{OPA\text{-}I}}$, $\boldsymbol{N}_{\mathrm{OPA\text{-}O}}$ are incidence matrices and $\boldsymbol{A}$ is the vector of OPAs amplifications all defined in [6]. A linear model of the OPA given by $U_{\mathrm{Out}} = A(U^+ - U^-)$ is used. Except the linearized OPAs, there are no other nonlinear circuit elements and thus the circuit can be described by a linear state space representation given by

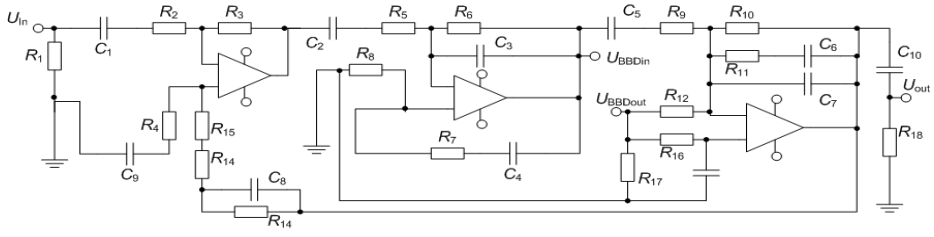


Figure 6: *Circuit schematic of the signal path of the flanger effect.*

$$\begin{aligned} \boldsymbol{X} &= \boldsymbol{AX} + \boldsymbol{BU} \\ \boldsymbol{Y} &= \boldsymbol{DX} + \boldsymbol{EU} \end{aligned} \tag{13}$$

where inputs vector is given by $\boldsymbol{U} = \begin{bmatrix} U_{\mathrm{In}}[n] \\ U_{\mathrm{BBDout}}[n] \end{bmatrix}$, outputs vector is given by $\boldsymbol{Y} = \begin{bmatrix} U_{out}[n] \\ U_{\mathrm{BBDin}}[n] \end{bmatrix}$, matrices ***A, B, D*** and ***E*** are linear system state space matrices derived according to [7] from the conductance matrix ***S*** defined in (12), vector ***X*** is the system state vector and *n* is the sampling period index. The secondary input $U_{\mathrm{BBDout}}[n]$ and output $U_{\mathrm{BBDin}}[n]$ are connected to the BBD delay line implemented using the algorithm from Figure 5.

3. SIMULATION RESULTS

In order to prove the validity of the proposed algorithm, the algorithm was implemented in Matlab and the frequency response of the real flanger effect was measured. Because it is the time variable effect, it is generally difficult to evaluate the simulation only by measurement and comparison of the frequency responses of measured and simulated effect. However this flanger effect has a switch (called filter matrix on the effect box) bypassing the LFO circuit. In this regime, the input low pass filter in the clock generation circuit is fed up with constant voltage 1.7 V, this input voltage is amplified by the low pass filter using potentiometer R_1 in IC_1 feedback yielding the comparator threshold voltages within $U_{\mathrm{THD}} \in (1.7, 8.4)$ V for all settings of the potentiometer R_1 (called „*Range*“) in Figure 3. This static regime allows to measure the frequency response of the effect. Figures 7 and 8 show comparison of the measured and simulated frequency responses. The sampling frequency used for the simulation was equal to 96 kHz to match the frequency responses at higher frequencies otherwise warped by the bilinear transform. The length of the BBD was 512 samples and the integration time was adjusted by resistor R_6 in Figure 3, which has the same function (with wide range of settings) in the original circuit, to match the frequency responses.

Additionally, the algorithm was implemented in C++ language as a VST plugin and was able to run in real-time. The implementation of the BBD line requires shifting and filtering of all stored samples few times during one sampling period which is computationally very demanding for simulation in real time. To decrease the computational demands, the BBD line can by implemented as a fixed delay line using a circular buffer instead of shifting all the delay cells and further the low pass filtering between the delay cells can be omitted. This however can affect the effect frequency responses of the effect at high frequencies.

4. CONCLUSIONS

Simulation of the Electro-Harmonix Deluxe Electric Mistress flanger effect was described in this paper. The main attention was paid to simulation of the clock signal generation circuit and the BBD circuit. In contrast to the standard digital delay line implementations, the BBD delay line consist of fixed number of delay cells and the variable delay is obtained using different speed of reading the data from the delay line. The main signal path of the flanger effect was simulated using the nodal DK-method with incidence matrices. The novelty is use of the incidence matrices for operational amplifiers allowing simulation of the whole signal path circuit including the global feedback without dividing the circuit into blocks. The results showed good match between the simulated and measured frequency responses of the flanger effect proving validity of the proposed simulation algorithm.

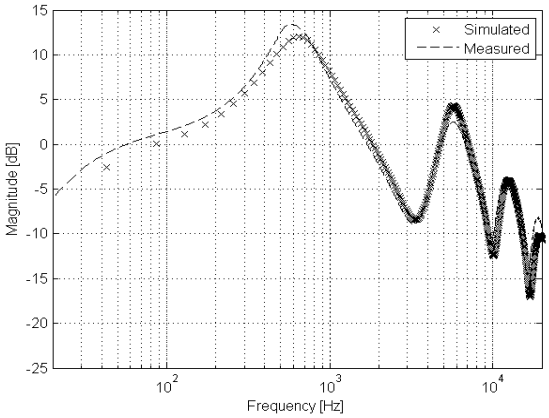


Figure 7: *Frequency responses for min range parameter.*

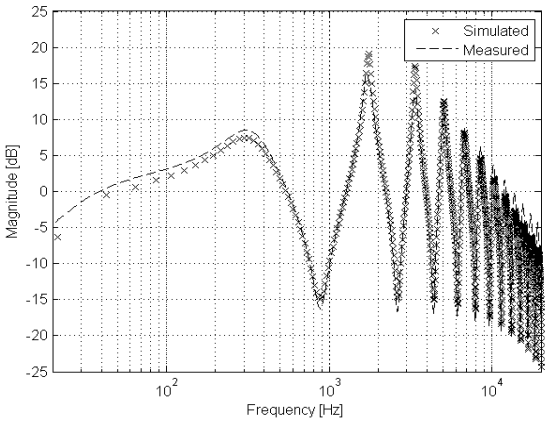


Figure 8: *Frequency responses for max range parameter.*

5. REFERENCES

[1] F. L. J. Sanders and K. Teer, “Bucket-brigade electronics – new possibilities for delay, time axis conversion, and scanning”, *IEEE Journal of Solid State Circuits*, vol. 4, no. 3, pp. 131 – 136, 1969.

[2] F. L. J. Sanders, “Integrated bucket-brigade delay line using MOS tetrodes”, Tech. Rep., Phillips Technical Review, Vol. 31, pp. 226-228, 1970.

[3] U. Zölzer, DAFX - Digital Audio Effects, J. Wiley & Sons, Ltd, New York, 1 edition, 2002.

[4] C. Raffel and J. Smith, “Practical Modeling of Bucket-brigade Devices Circuits,” in *Proceedings of the 13th International Conference on Digital Audio Effects DAFx10*, Graz, Austria, Sept. 6–10 2010.

[5] R. Metzger, "The Mistress Mystery Page" Available from http://www.metzgerralf.de/elekt/stomp/mistress/index.shtml , 2016, [online].

[6] J. Mačák, *Real-time Digital Simulation of Guitar Amplifiers as Audio Effects*, Ph.D. thesis, Brno University of Technology, Brno, 2012.

[7] M. Holters and U. Zölzer, “Physical modelling of a wah-wah effect pedal as a case study for application of the nodal dk method to circuits with variable parts,” in *Proceedings of the 14th International Conference on Digital Audio Effects DAFx11*, Paris, France, Sept. 19–23, 2011.

AUDIO NONLINEAR MODELING THROUGH HYPERBOLIC TANGENT FUNCTIONALS

Adalberto Schuck Jr. [*]

DELET - EE - UFRGS,
Dept. of Electrical Engineering
Engineering School
Federal University of Rio Grande do Sul
Porto Alegre, RS, Brazil
schuck@ufrgs.br

Bardo Ernst Josef Bodmann [†]

DEMEC - EE - UFRGS,
Dept. of Mechanical Engineering
Engineering School
Federal University of Rio Grande do Sul
Porto Alegre, RS, Brazil
bardo.bodmann@ufrgs.br

ABSTRACT

In the present contribution we present the preliminary results of a black box nonlinear system (NLS) modeling. The NLS is composed by a nonlinear sigmoid-type input-output relationship (NLTF) followed by a linear system (LTI), as in a Hammerstein nonlinear system. Here, the used NLTF is derived from a deformation of the Hyperbolic Tangent power expansion. The advantage of using the hyperbolic tangent function is that nonlinearity depends on the linear and cubic terms that measure curvature (and thus nonlinearity) of the transfer function. The hyperbolic tangent model is extended to other types of nonlinear systems by expanding the nonlinear system in linear and increasingly nonlinear contributions, where the expansion parameters are deformed to enhance or suppress specific nonlinear modes of the expansion. Simulations were performed using Matlab 2012a. The preliminary results show fairly good agreement between the system obtained by parametric inference and a reference system, with mean square error (MSE)=0.035.

1. INTRODUCTION

Linear Time Invariant Systems (LTI) have been extensively studied for decades [1], [2]. However, the nonlinearities in audio musical systems that are responsible for specific tone characteristics desired by many musicians [3], [4].

The nonlinearities of the NLS can be weak (i.e the NLS can be represented by a power series expansion of only a few terms) or strong (higher order terms of the power series expansion must be calculated for the modelling system) [5]. If the NLS is very weak, a linear approximation can be used. On the other hand, very strong nonlinear systems as samplers, switches and other systems with discontinuities in the system representation must use the entire terms of its power series representation, which is rather inconvenient for modeling purposes [6]. Then, in accordance to the "strength" of the nonlinearities an appropriate and efficient technique to be used is in order. Guitar and bass vacuum-tube amplifiers can be considered weakly NLS and most of NLS identification techniques make this assumption.

By its own nature, when a weak nonlinear system related with saturation (NLS) has as input a pure sine wave $x(t) = A_1 \cos(2\pi f_1 t + \phi_1)$, higher harmonics related to the input frequency will appear in the output, according with the relation $y(t) = \sum_n B_n \cos(2\pi n f_1 t + \phi_n)$. Accordingly, LTI identification techniques as impulse response, convolution and Laplace or Fourier analysis cannot be used with NLS. Hence, many attempts have been suggested to model such NLS [4].

The main techniques for NLS identification/modeling can be classified in **black box approaches**, **white box approaches** or **analytic modeling techniques** and **grey box approaches**. Black box approaches involve no *a priory* knowledge on the NLS to be modelled/identified. The NLS is excited with a set of test signals and using obtained outputs, the coefficients of a polynomial or power series relation are estimated in such a way as to minimize the error between the true NLS system and the model. In this category one finds the Volterra, Volterra-Wiener and Hammerstein model techniques, for instance, [5], [7], [8], [9], [10], [11], [3], [12], [13]. On the other hand, white box approaches involve total knowledge of the system to be modelled. To achieve this, one needs to know the circuit theory and the schematics of the devices to be modelled. In possession of this information, the nonlinear differential equation set of the circuits involved can be obtained and solved. In this category the models for SPICE Simulation, transient modified nodal analysis, state-space representation and numerical methods for solving nonlinear circuits techniques can be found as for instance in refs. [14], [15], [16], [17], [18], [19], [20]. Grey box approaches use polynomial models as in black box techniques, but incorporating some knowledge about the nonlinear circuits used. Good reviews of NLS modeling techniques can be found in [4], [21].

In this work we propose a black box method where the NLS is composed by a nonlinear sigmoid-type input-output relationship (nonlinear transfer function, NLTF) followed by a LTI, as in Hammerstein nonlinear systems, [12]. This is shown in Fig. 1. The first section shows the calculation of the output from the entire system given a generic input. Then the developments based on the Hyperbolic Tangent series and its coefficients estimation are shown, followed by a practical example.

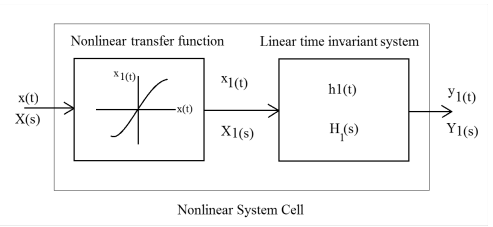


Figure 1: *The entire NLS proposed.*

[*] This work was supported by author own support
[†] This work was supported by author own support

2. A MODEL PROPOSAL FOR THE MODELING SYSTEM

The NLS model (or cell) proposed here (Fig. 1) is half the system proposed in [3] and is appropriate to model weakly NLS as a vacuum tube amplifiers pre-amplifier or power stages and distortion and overdrive devices. It is composed by an NLTF followed by an LTI system as in a Hammerstein model [12]. The modeling procedure consists in applying appropriate test signals in order to estimate both the NLTF and the LTI system that minimize the error between the output of the NLS to be modeled and the NLS model, by some minimization criterium of the error between the true output signal against the NLS model output. Here we used for simplicity the standard minimum square method as specified further down in eq. (10). The modeling procedure is shown in Fig. 2. One of the differences of this work is that the NLTF used

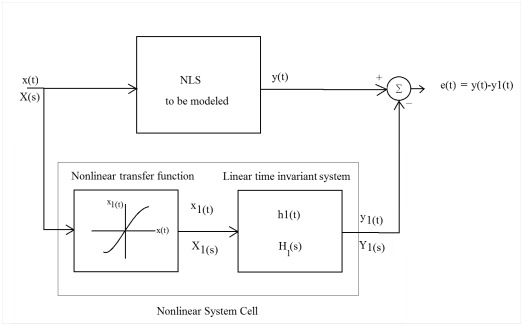


Figure 2: *The modeling system and its variables.*

is derived from a deformation of the Hyperbolic Tangent power expansion. The development and identification of the coefficients of this deformed expansion will be explained further below. As a matter of fact, the harmonics produced for the NLTF are filtered by the LTI system. This circuit section usually models the frequency response of the amplifier stage to be modeled, including the parasitic capacitances, tone controls or a simplified output transformer and speaker model.

Generally speaking, many LTI have the transfer function in Laplace domain described as a quotient of polynomials by ([2], [22]) :

$$H_1(s) = \frac{Y_1(s)}{X_1(s)} = G\frac{\sum_{k=0}^{M} B_k s^{M-k}}{\sum_{k=0}^{N} A_k s^{N-k}}\,, \tag{1}$$

where G is the Global Gain of the LTI and B_k and A_k are the coefficients of the differential equation that rules the LTI.

It is straight forward to show [2] that the differential equation which relates the output $y_1(t)$ with the input $x_1(t)$ is

$$\sum_{k=0}^{N} A_k \frac{\mathrm{d}^{N-k}x_1(t)}{\mathrm{d}t^{N-k}} = G\sum_{k=0}^{M} B_k \frac{\mathrm{d}^{M-k}y_1(t)}{\mathrm{d}t^{M-k}}\,, \tag{2}$$

recalling that $x_1(t)$ is the response of the NLS to an input $x(t)$.

The impulse response of eq. (1) can be analytically evaluated by finding the Poles (roots of the denominator polynomial) of the LTI, expanding eq. (1) in Partial Fractions and finding the inverse Laplace transform to the expansion [2],[22]. Otherwise, a numerical algorithm can be applied to eq. (2) to find a solution for that differential equation.

The output $y_1(t)$ can be represented by the convolution

$$y_1(t) = \int_{-\infty}^{\infty} h_1(\tau)x_1(t-\tau)\,\mathrm{d}\tau\,, \tag{3}$$

where $x_1(t)$ is the output of the NLTF for an input $x(t)$, or formally $x_1(t) = NLTF|_{x(t)}(t)$, so that eq. (3) is

$$y_1(t) = \int_{-\infty}^{\infty} h_1(\tau)NLTF|_{x(t)}(t-\tau)\,\mathrm{d}\tau\,. \tag{4}$$

For instance, if one estimate the NLTF as an arc hyperbolic sine function, like the one that mimics diode distortion pedals and a JCM900 preamp output voltage (ref. [15]), then eq. (4) can be written as

$$y_1(t) = \int_{-\infty}^{\infty} h_1(\tau)\mathrm{arcsinh}|_{x(t)}(t-\tau)\,\mathrm{d}\tau\,. \tag{5}$$

To estimate this sub-system one has to apply an input signal $x(t)$ small enough in order that the NLS to be modeled may be considered a linear system. This is fairly true for many amplifiers and distortion devices, and all techniques already developed for LTI can be used, in time or in frequency domain. In the present case the identification/estimation is performed by $\widetilde{H_1}(\omega) = Y(\omega)/X(\omega)$.

3. THE NONLINEAR SUB-SYSTEM MODEL

As a starting point for modeling the weakly nonlinear properties of an audio-system, we start from a mathematical function, i.e. the hyperbolic tangent

$$\begin{aligned} x_1 &= \tanh\alpha \\ &= \frac{e^{\alpha} - e^{-\alpha}}{e^{\alpha} + e^{-\alpha}}\,, \end{aligned} \tag{6}$$

that in certain limits of α (the input signal and amplification) exhibits predominantly linear properties and beyond these limits, then the full nonlinear characteristics as can be seen in Fig.3. The

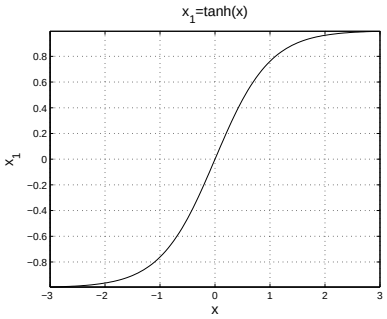


Figure 3: $x_1 = \tanh x$.

choice for this specific function resides in the fact that a representation by exponential functions may be easily implemented computationally. Then, the normalized output signal $y_1(t)$ of the nonlinear sub-system may be described by the input signal $x(t)$ and the response of the nonlinear sub-system

$$x_1(t) = \tanh\left(\underbrace{\kappa(x(t)+x_0)}_{\alpha}\right) - \tanh\left(\kappa x_0\right)\,, \tag{7}$$

where κ plays the role of an amplification factor and for a symmetric NLTF $x_0 \equiv 0$, whereas for asymmetric NLTF $x_0 \neq 0$. Note that besides the translation of the argument of the hyperbolic function there is also necessarily a shift in the function such as to match a zero input signal with a zero output signal. In this kind of approach the nonlinearity is a unique one which may be seen from the constituent differential equation that has the hyperbolic tangent as solution.

$$\frac{\partial^2 \xi(\alpha)}{\partial \alpha^2} = 2\left(\xi^3(\alpha) - \xi(\alpha)\right) \tag{8}$$

The "beauty" of the hyperbolic tangent function is that the second derivative and its dependence on the linear and cubic signal is a direct measure for the curvature of the transfer function and thus for the nonlinear behavior. In order to extend this nonlinear model to other types of nonlinear systems we proceed in two steps, first expand the nonlinear system in linear and increasing nonlinear contributions, and second, extend the expansion by introducing new parameters that allow to tune the balance of the linear and nonlinear character, in other words, allow to enhance or suppress specific nonlinear modes of the expansion. For signals $x(t)$ sufficiently small most of the audio systems show a linear behavior, which corresponds to the first term of a small signal expansion, where including nonlinear terms nevertheless restricting generality we truncate at the fifth term that represents a nonlinear character. If $x_1 = \tanh(\kappa x)$ with $\kappa x = \alpha$, then

$$\begin{aligned}
\xi(\alpha) &= \left(\sum_{n=0}^{\infty} \underbrace{\frac{1}{n!} \left.\frac{\partial^n \tanh(\alpha)}{\partial \alpha^n}\right|_{\alpha=\kappa x_0}}_{a_n} \alpha^n \right) - \tanh(\kappa x_0) \\
&= \left(1 - \tanh^2(\alpha)\right)\Big|_{\alpha=\kappa x_0} \alpha \\
&\quad - \tanh(\alpha)\left(1 - \tanh^2(\alpha)\right)\Big|_{\alpha=\kappa x_0} \alpha^2 \\
&\quad - \left(\frac{1}{3} - \tanh^2(\alpha)\right)\left(1 - \tanh^2(\alpha)\right)\Big|_{\alpha=\kappa x_0} \alpha^3 \\
&\quad + \left(\frac{2}{3} - \frac{1}{2}\tanh^2(\alpha)\right)\tanh(\alpha) \times \\
&\qquad \times \left(1 - \tanh^2(\alpha)\right)\Big|_{\alpha=\kappa x_0} \alpha^4 \\
&\quad + \left(\frac{2}{15} - \frac{21}{30}\tanh^2(\alpha) + \frac{1}{2}\tanh^4(\alpha)\right) \times \\
&\qquad \times \left(1 - \tanh^2(\alpha)\right)\Big|_{\alpha=\kappa x_0} \alpha^5 \\
&\quad + \mathcal{O}(\alpha^6)
\end{aligned} \tag{9}$$

It is noteworthy that all terms of the expansion are linearly independent so that one may modify the original factors $a_n \rightarrow a_n + \delta a_n$ by an increment or decrement δa_n and thus adjust the linear to the nonlinear proportions beyond that given by the hyperbolic tangent function and additionally shape the nonlinearity since $[2\left(y^3(\alpha) - y(\alpha)\right)]$ is no longer the original curvature.

Recalling that we are considering a black box modeling, nevertheless focus on specific audio system characteristics the connection between a system with measured total response function (microphone, amplifier and speaker characteristics) may be determined using parametric inference techniques as laid out for instance in [23]. To this end the input signal $x(t)$ as well as the desired output signal $y(t)$ is discretized in T times t_i ($i \in \{0, \ldots, T\}$) and the factors δa_n are adjusted such as to minimize the difference of the model and the experimental data.

$$\min_{\{x_0, \{\delta a_n\}_{n=1}^N\}} \left(\sum_{i=0}^{T} \left\| y(t_i) - \int_0^{t_i} h(t_i - \tau)\xi(\tau)\, d\tau \right\| \right) \tag{10}$$

Here $||\cdot||$ denotes any convenient norm or semi-norm. In this work we use the Euclidian norm.

A comment is in order here, other approaches make use of orthogonal functional spaces which they use to model nonlinear responses [5]. With respect to this approach ours is not that much different, however, the advantage of the present approach is justified by the fact that all derivatives of the hyperbolic tangent function may again be represented by hyperbolic tangent functions and constants, in other words only one function (represented by exponential functions) is needed to generate the whole expansion, where in the present discussion we showed the linear and five nonlinear terms that appear in the expansion. In the symmetric case the parity even terms in α disappear identically. Moreover, in principle the tuning needs only to be stored in a vector that indicates directly the linear and nonlinear characteristics to be modeled. For example, if the NLTF would be a simple hyperbolic tangent, such as $y_1(t) = \tanh x(t)$, the hyperbolic tangent can be expressed as a power series by [24]

$$\tanh x = x - \frac{x^3}{3} + 2\frac{x^5}{15} - 17\frac{x^7}{315} + \ldots, \tag{11}$$

and then the coefficients a_n for n even would be 0 and $a_1 = 1$, $a_3 = -\frac{1}{3}$, $a_5 = \frac{2}{15}$, $a_7 = -\frac{7}{315}$ and so on. For a case of arc hyperbolic sine as the NLTF, expanding the arc hyperbolic sine gives [24]

$$\operatorname{arcsinh} = x - \frac{x^3}{6} + 3\frac{x^5}{40} + \ldots, \tag{12}$$

and in this case a_n for n even would be 0 and $a_1 = 1$, $a_3 = -\frac{1}{6}$, $a_5 = \frac{3}{40}$ and so on, which is $\delta a_3 = -\frac{1}{2}a_3$ and $\delta a_5 = -\frac{7}{120}$ with respect to the hyperbolic tangent case.

4. METHODS

In order to simulate a system to be identified, all the signals and systems were performed using Matlab 2012a version. The sampling frequency used is $f_s = 120000$ samples per second in order to accommodate the harmonics of the output signal up to the 5^{th} component without aliasing, simulating an A/D system with $f_s = 20000$ samples per second with an anti-alias analogue 6^{th} order low-pass filter with cut-off frequency of 10kHz before the sampling process and after the discretization, the signal is upsampled 6 times. But all the signals involved to be exhibited are downsampled to $f_{s1} = 20000$ samples per second.

For the NLTF to be modeled, we chose the distortion simulation presented in [21], given by

$$x_1 = \operatorname{sign}(x)(1 - e^{-|x|}). \tag{13}$$

The graphic of this NTFS is shown in Fig.4. The output of this block feeds a discrete linear system which is a 2_{nd} order digital Butterworth bandpass filter with cut-off frequencies of 100 Hz and 8000 Hz, normalized to radians. This simulates approximately the frequency response of a $12''$ guitar speaker. The frequency response of this system is shown in Fig.5: The signal $x[n]$ chosen for the LTI identification is a linear chirp from 0Hz at $n = 0$ to 10kHz at $n = 65535$ (or $t = 0.96$ s). In order to keep the nonlinear effects of the NLTF negligible, the signal must be so small that its amplitude does not surpass the linear part of the NLFT. Because of that, the maximal amplitude of the signal was chosen equal 0.01. The periodogram and the spectrogram of this signal is shown in Fig. 6.

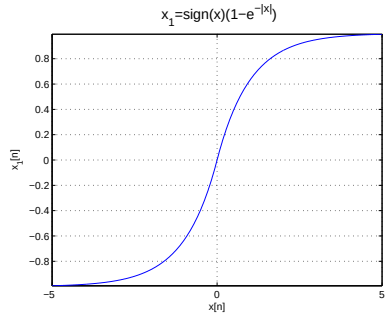


Figure 4: *Graphic of the NLTF* $x_1 = \text{sign}(1 - e^{-|x|})$

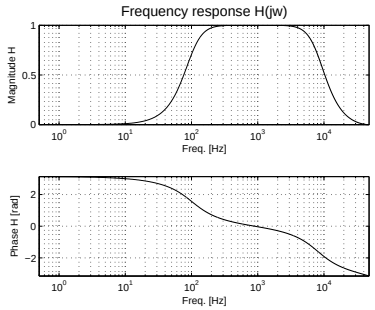


Figure 5: *Frequency response of the LTI system to be identified.*

Injecting the signal $x[n]$ in the system to modeled, the signal $y[n]$ is obtained and assuming the NLTF nonlinear effect is negligible, then $H_1(w)$ can be estimated by

$$\widetilde{H_1}[k] = \frac{Y[k]}{X[k]}\,, \tag{14}$$

where $Y[k]$ and $X[k]$ are the FFTs of the signals $x[n]$ and $y[n]$, respectively. In Fig. 7, can be seen the periodogram and the spectrogram of the signal y[n]. Note that using these signal amplitude, aliasing practically did not occur and the nonlinear effect appears as a very faint 3_{rd} harmonic line in its spectrogram. Just for the sake of exemplification, if the chirp signal used here would have a large amplitude A (eg. $A = 5$), the output of the NLS to be modeled would had the periodogram and spectrogram shown in Fig.8 The frequency response of $\widetilde{H_1}[k]$ obtained is shown in Fig.9: The impulse response estimation of $\tilde{h_1}[n]$ is obtained performing the IFFT over $\widetilde{H_1}[k]$. It's worth noting that this form of estimation will provide a FIR system model, even if the LTI system to be estimated be an IIR system (as our case). But if one choose the appropriate number of coefficients for the FIR system, the error can be made negligible and the computational effort minimized. Using $N = 65536$, then the estimation of the LTI system $\widetilde{H_1}[k]$ impulse response $\tilde{h_1}[n]$ is obtained. The next step is to estimate the NLTF of the system. To do that, a second signal $x[n]$ is injected in the system, now a sine signal with fixed frequency inside the band pass of the system (already identified) but with amplitude large enough so the output signal $y[n]$ be distorted. Here the signal was a sine signal with frequency of 500Hz (around the center frequency of the Bandpass) and amplitude A of 5. The input and output signals can be seen in Fig.12 To remove the phase influence of the LTI sub-system in the output signal, an inverse filter (aka equalizer filter) $g[n]$ obtained by turning the magnitude of

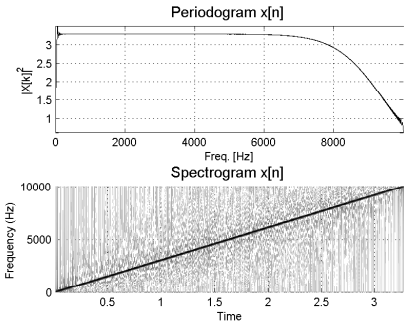


Figure 6: *The Periodogram and the Spectrogram of the signal x[n] for LTI estimation.*

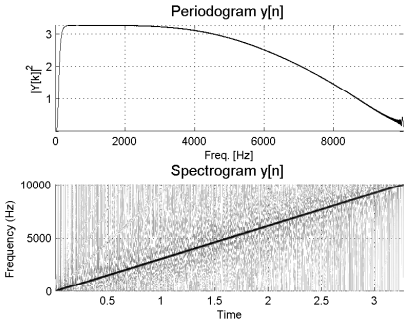


Figure 7: *The Periodogram and the Spectrogram of the signal y[n].*

$\widetilde{H_1}[k] = 1.0$ and then $g[n] = IFFT[\frac{1}{\widetilde{H_1}[k]}]$ as shown in Fig. 10. This will preserve the amplitude of the signal $y[n]$ but will correct the phase only. is applied to the output signal $y[n]$, and giving an equalized signal $y_{eq}[n]$. The scatter plot between $x[n]$ and $y_{eq}[n]$ is shown in Fig.11: As can be seen by the Lissajous curves obtained, the phase wasn't completely corrected but the phase error of $y_{eq}[n]$ is small enough for the next step. Finally, with both signal $x[n]$ and $y_{eq}[n]$, the new coefficients $a_k + \delta a_k$ of eq. (9) can be estimated. Here, we used minimization with Euclidian norm and consequently, least squares fitting techniques, evaluated up to the 10^{th} order. The coefficients obtained for our example are

$$\begin{aligned}
a_0 + \delta a_0 &= 0.000412543580279774\,,\\
a_1 + \delta a_1 &= 0.695903675375691\,,\\
a_2 + \delta a_2 &= 8,42100337873921e-06\,,\\
a_3 + \delta a_3 &= -0.0873565648035508\,,\\
a_4 + \delta a_4 &= -1.51387407429003e-05\,,\\
a_5 + \delta a_5 &= 0.00715137611726394\,,\\
a_6 + \delta a_6 &= 2.73945820531464e-06\,,\\
a_7 + \delta a_7 &= -0.000280736782949899\,,\\
a_8 + \delta a_8 &= -1.67046480776686e-07\,,\\
a_9 + \delta a_9 &= 4.11012487805344e-06\,,\\
a_{10} + \delta a_{10} &= 3.17102262802150e-09\,,\\
&\cdots\,.
\end{aligned}$$

The comparison between the outputs of the actual system and the NLS estimated is shown in Fig.12.

Finally, it is shown on Fig.13 the output of the system to be modeled and the model for all the parameters above, having as input a sine signal of $1kHz$.

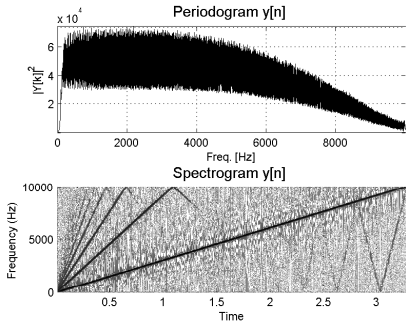


Figure 8: *The Periodogram and the Spectrogram of the signal y[n] for a chirp with* $A = 5$.

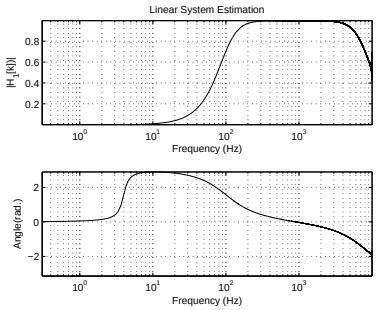


Figure 9: *Frequency response of the LTI system estimated.*

The Mean Square Error (MSE) between $y[n]$ and $y_1[n]$, estimated by

$$MSE = \frac{1}{N}\sum_{n=0}^{N-1}(y[n] - y_1[]n)^2 \tag{15}$$

was around $MSE = 0.035$.

5. CONCLUSIONS

In the present work we discussed a black box approach for a NLS that shall simulate the response of an audio system such as a tube amplifier for musical instruments with its characteristic frequency response of the amplifier, including the parasitic capacitances, tone controls, output transformer and speaker. To this end we used a chirp signal to excite the NLS followed by a LTI and adjusted the coefficients of a deformed hyperbolic tangent power expansion in order to reproduce a desired output by the use of parametric inference technique. The choice for the deformed hyperbolic tangent function resides in the fact that a representation by exponential functions may be easily implemented digitally and allows to describe symmetric as well as asymmetric amplification using translations of the argument and amplitude respectively. Moreover the hyperbolic tangent function has a simple relation to its nonlinearity since the second derivative depends on the linear and cubic function. As a specific example we used an exponential distortion $x_1 = \text{sign}(1 - e^{-|x|})$ followed by a Butterworth 2_{nd} order bandpass filter with cut-off frequencies related to 100Hz and 8kHz to simulate the frequency response of an 12" speaker. For the example used, the Mean Square Error (MSE) between $y[n]$ and $y_1[n]$ was around $MSE = 0.035$.

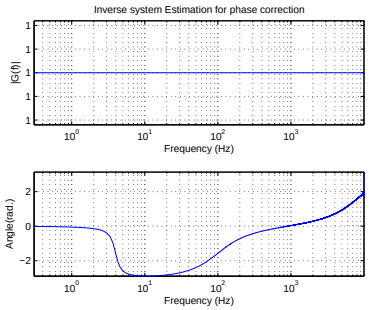


Figure 10: *Frequency response of the Inverse filter estimated.*

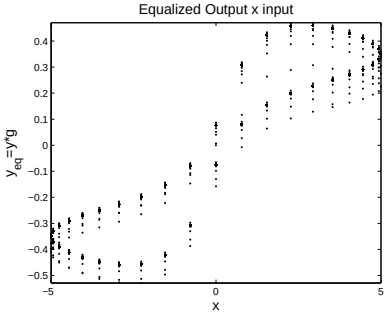


Figure 11: *Scatter plot between* $x[n]$ *and* $y_{eq}[n]$.

The authors of the present work are aware of the fact that there are other approaches similar to the present one, however, they make use of orthogonal functional spaces which they use to model nonlinear responses. We believe, that the advantage of the present approach is justified due to the fact that all derivatives of the hyperbolic tangent function may again be represented by hyperbolic tangent functions and constants. The deformation parameters that were determined to reproduce the desired input - output signal relation showed that the desired system can be reproduced with fairly good fidelity. In future work we intend to apply the proposed procedure to a selection of other amplifiers obtained by measurements and compare quality as well as computational efficiency for simulation applications.

6. REFERENCES

[1] Chi-Tsong Chen, *Linear Systems Theory and Design*, Oxford University Press, Oxford, UK, third edition, 2009.

[2] S. Haykin and B. van Veen, *Signals and Systems*, John Wiley & Sons,Ltd, Hoboken, New Jersey, USA, second edition, 2005.

[3] K. Dempwolf, M. Holters, and U. Zölzer, "The influence of small variations ina a simplified guitar amplifier model," in *Proc. of the 12th Int. Conference on Digital Audio Effects (DAFx-10)*, Como, Italy, Sept. 01-04, 2009, pp. DAFx–1–DAFX–6.

[4] J. Pakarinen and D. T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music Journal*, vol. 33, no. 2, pp. 85–100, 2009.

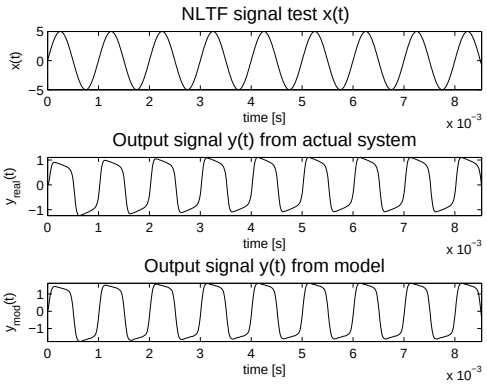


Figure 12: *Comparison between the outputs from the actual system and from the NLS estimated.*

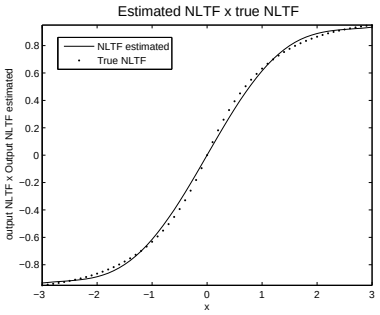


Figure 13: *Estimated NLTF vs. Actual NLTF.*

[5] Martin Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*, John Wiley & Sons,Ltd, New York, USA, 1980.

[6] M. R. Dunn, "The volterra series and its application," M.S. thesis, University of California, 1992.

[7] Martin Schetzen, "Nonlinear system modeling based on the wiener theory," *Proceedings of the IEEE*, vol. 69, no. 12, pp. 1557–1573, 1981.

[8] J. Schattschneider and U. Zölzer, "Discrete-time models for nonlinear audio systems," in *Proc. of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx-99)*, Trondheim, Norway, Dec. 09-11, 1999.

[9] A. Farina, "Simultaneous measurement of impulse response and distortion with swept-sine technique," in *AES 108th Convention*, Paris, France, Feb. 19-24, 2000, pp. 1–24.

[10] A. Farina, A. Bellini, and E. Armelloni, "Non-linear convolution: A new approach for the auralization of distorting systems," in *AES 110th Convention*, Amsterdam, The Netherlands, May 12-15, 2001, pp. 1–4.

[11] Thomas Hélie, "On the use of volterra series for real-time simulations of weakly nonlinear analog audio devices: Application to the moog ladder filter," in *Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx-06)*, Montreal, Canada, Sept. 18-20, 2006, pp. DAFx–7–DAFX–12.

[12] A. Novák, L. Simon, F. Kadlec, and P. Lotton, "Nonlinear system identification using exponential swept-sine signal," *IEEE Trans. Instrum. and Meas.*, vol. 59, no. 8, pp. 2220–2228, 2010.

[13] I. Mezghani-Marrakchi, G. Mahé, S. Djaziri-Larbi, M. Jaïdane, and M. T.-H. Alouane, "Nonlinear audio system identification through audio input gaussianization," *IEEE Trans. Audio, Speech and Language Processing*, vol. 22, no. 1, pp. 41–53, 2014.

[14] S. Möller, M. Gromowski, and U. Zölzer, "A measurement technique for highly nonlinear transfer functions," in *Proc. of the 5th Int. Conference on Digital Audio Effects (DAFx-02)*, Hamburg, Germany, Sept. 26-28, 2002, pp. DAFx–203–DAFX–206.

[15] K. Dempwolf, M. Holters, and U. Zölzer, "Analysis and simulation of an analog guitar compressor," in *Proc. of the 13th Int. Conference on Digital Audio Effects (DAFx-10)*, Graz, Austria, Sept. 06-10, 2010, pp. DAFx–1–DAFX–8.

[16] M. Holters and U. Zölzer, "Physical modelling of a wha-wha effect pedal as a case study for application of the nodal dk method to circuits with variable parts," in *Proc. of the 14th Int. Conference on Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 19-23, 2011, pp. DAFx–31–DAFX–35.

[17] O. Kroning, K. Dempwolf, and U. Zölzer, "Analysis and simulation of an analog guitar compressor," in *Proc. of the 14th Int. Conference on Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 19-23, 2011, pp. DAFx–205–DAFX–208.

[18] F. Eichas, M., M. Holters, and U. Zölzer, "Physical modeling of the mxr phase 90 guitar effect pedal," in *Proc. of the 17th Int. Conference on Digital Audio Effects (DAFx-11)*, Erlangen, Germany, Sept. 01-05, 2014, pp. DAFx–1–DAFX–6.

[19] D. T. Yeh, J. S. Abel, A. Vladimirescu, and J. O. Smith, "Numerical methods for simulation of guitar distortion circuits," *Computer Music Journal*, vol. 32, no. 2, pp. 23–42, 2008.

[20] D. T. Yeh, J. S. Abel, and III J. O. Smith, "Automated physical modeling of nonlinear audio circuits for real-time audio effects-part i:theoretical development," *IEEE Trans. Audio, Speech and Language Processing*, vol. 18, no. 4, pp. 728–737, May 2010.

[21] Udo Zölzer, *Digital Audio Effects*, John Wiley & Sons,Ltd, Chicester, West Sussex, UK, second edition, 2011.

[22] Julius O. Smith III, *Physical Audio Signal Processing for Virtual Musical Instruments and Audio Effects*, W3K Publishing, 2010, http://books.w3k.org.

[23] David Roxbee Cox, *Principles of Statistical Inference*, Cambridge University Press, Cambridge, UK, 2006.

[24] L. Rade and B. Westergreen, *Mathematical Handbook for Science and Engineering*, Springer-Verlag, Berlin/Heidelberg, Germany, fifth edition, 2004.

SIGNAL-MATCHED POWER-COMPLEMENTARY CROSS-FADING AND DRY-WET MIXING

Marco Fink, Martin Holters, Udo Zölzer*

Department of Signal Processing and Communications,
Helmut-Schmidt University Hamburg
Hamburg, Germany
marco.fink@hsu-hh.de

ABSTRACT

The blending of audio signals, called cross-fading, is a very common task in audio signal processing. Therefore, digital audio workstations offer several fading curves to select from. The choice of the fading curve typically depends on the signal characteristics and is supposed to result in a mixed signal featuring power and loudness close to the input signals. This work derives a correlation-based design of the fading curves to achieve exact power consistency to avoid audible fluctuations of the signal's loudness. This principle is extended to the problem of mixing original signals with effect-processed signals using the dry-wet balance. Weighting coefficients for dry and wet signals are derived which realize the desired dry-wet balance but maintain the signal power.

1. INTRODUCTION

The convenience and affordability of todays audio processing tools allows almost any user to work on audio data, especially digitally. Digital audio workstations (DAWs) offer manifold possibilities for recording, mixing, arranging, adding effects, mastering, etc. One of the essential tools within any DAW is the cross-fader which allows to smoothly blend between separate pieces of audio [1]. A cross-fade is realized by multiplying the signals with fading curves, as illustrated in Fig. 1, and summing the weighted signals. Three main applications for the utilization of cross-fading can be emphasized:

1. Transition between songs:
   Many digital audio players allow to cross-fade between the end of the current song and the beginning of the next song to achieve smooth transitions and continuous sound. Quite long transition times of several seconds are applied in general.
2. Blend between different tracks or takes:
   Oftentimes cross-fading is applied to switch between different takes of a recording to combine the best parts of different performances of the instrumentalist. Typical cross-fade times for this application are in the range of $10 - 30$ ms [2]. The transition between different recorded tracks or instruments within a song can also be realized using cross-fades, especially in electronic music.
3. Fading between sources in DJ-ing:
   Many DJs artistically recompose elements from recordings to create new songs. This approach requires cross-fading to blend between sources and beats. In other words, turntables and mixers can even be considered as the DJs actual instrument and underlie a continuous evolution [3].

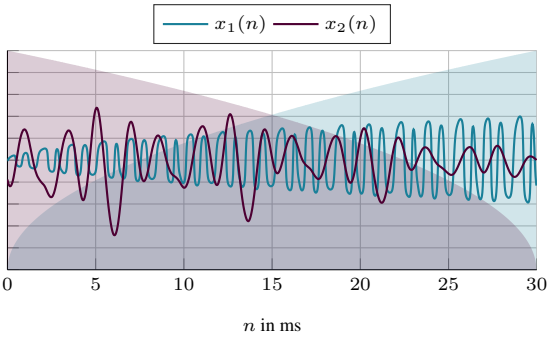


Figure 1: *Exemplary cross-fading of two signals using square-root fading curves*

Another application of cross-fading is to blend intact and synthesized concealment audio streams in the context of packet loss concealment [4, 5]. All mentioned applications share the requirement of preserving the signal's loudness while cross-fading. It is well-known that the cross-faded signal power depends on the correlation of the signals to be mixed. Uncorrelated signals are cross-faded using the so-called *Equal Power Crossfade* featuring a -3 dB weighting in the transition center. In contrast, correlated signals are supposed to be linearly cross-faded corresponding to a weighting of -6 dB [2, 3]. However, these rules of thumb are very practically motivated and not perfectly accurate. Therefore, this work derives fading curves analytically based on the exactly measured correlation of the signals to be cross-faded to achieve perfect power preservation. The perfectly power-complementary fading curves are derived in Sect. 2 based on a simple signal model. The power preservation property of the proposed design is validated in Sect. 3. Additionally, Sect. 4 demonstrates that the proposed fading design can also be utilized in the context of dry-wet mixing which shares several properties with the problem of cross-fading. The work is concluded in Sect. 5.

2. CROSS-FADING

The cross-fading of two signals $x_1(n)$ and $x_2(n)$ of a certain length N can be described as

$$x_{\text{mix}}(n) = w(n)\, x_1(n) + w_i(n)\, x_2(n), \quad n \in [0, \ldots, N-1], \tag{1}$$

* The author is now with Ableton and can be contacted using marco.fink@ableton.com

where $w(n)$ and $w_i(n)$ are the fading and the inverse fading curve. Assuming signals with the same power and hence same signal variance

$$\sigma_x^2 = \mathrm{E}\left[x_1^2\right] = \mathrm{E}\left[x_2^2\right] \tag{2}$$

and identical mean value

$$\mu_x = E\left[x_1\right] = E\left[x_2\right] = 0 \tag{3}$$

allows one to estimate the variance of the faded signal

$$\begin{aligned}
\mathrm{E}\left[x_{\mathrm{mix}}^2\right] &= \mathrm{E}\left[(w\,x_1 + w_i\,x_2)^2\right] && (4)\\
&= \mathrm{E}\left[w^2x_1^2 + 2\,w\,w_i\,x_1\,x_2 + w_i^2x_2^2\right] && (5)\\
&= w^2\mathrm{E}\left[x_1^2\right] + 2\,w\,w_i\,\mathrm{E}\left[x_1x_2\right] + w_i^2\mathrm{E}\left[x_2^2\right] && (6)\\
&= w^2\sigma_x^2 + 2\,w\,w_i\,\mathrm{Cov}(x_1,x_2) + w_i^2\sigma_x^2. && (7)
\end{aligned}$$

Expressing the covariance as the product of the signal variances and the correlation coefficient $\mathrm{Cov}(x_1,x_2) = r_{x_1,x_2}\,\sigma_x^2$ yields

$$\begin{aligned}
\mathrm{E}\left[x_{\mathrm{mix}}^2\right] &= w^2\sigma_x^2 + 2\,w\,w_i\,r_{x_1,x_2}\sigma_x^2 + w_i^2\sigma_x^2 && (8)\\
&= \sigma_x^2(w^2 + 2\,w\,w_i\,r_{x_1,x_2} + w_i^2). && (9)
\end{aligned}$$

It is typically desirable that the cross-faded signal features the same power and hence variance as the input signals

$$\mathrm{E}\left[x_{\mathrm{mix}}^2\right] = \sigma_x^2(w^2 + 2\,w\,w_i\,r_{x_1,x_2} + w_i^2) = \sigma_x^2. \tag{10}$$

Canceling the variance holds

$$w^2 + 2\,w\,w_i\,r_{x_1,x_2} + w_i^2 = 1. \tag{11}$$

In the following, the amplitude ratio of w and w_i shall be described with the function depending on the normalized time index $\alpha = \frac{n}{N-1}$

$$f(\alpha) = \frac{w(\alpha)}{w_i(\alpha)}. \tag{12}$$

Rewriting Eq. (12) to $w(\alpha) = f(\alpha)\,w_i(\alpha)$ and inserting it into Eq. (11) holds

$$w_i^2(\alpha) = \frac{1}{1 + 2\,f(\alpha)\,r_{x_1,x_2} + f^2(\alpha)}. \tag{13}$$

Multiple assumptions concerning $f(\alpha)$ can be made:

1. $f(0) = 0$ since the fading curve w starts with an amplitude of 0
2. $f(1) = \infty$ since the inverse fading curve w_i ends with an amplitude of 0
3. $f(\alpha) = \frac{1}{f(1-\alpha)}$ since the fading curves w and w_i are symmetric and hence feature the same amplitude in the center of the curve $f(0.5) = 1$.

Several functions fulfill these requirements. In the following, the tangent function $f_{\tan}(\alpha) = \tan(\frac{\pi\,\alpha}{2})$ and the function $f_{\mathrm{scl}}(\alpha) = \frac{\alpha}{1-\alpha}$ are utilized and applied in Eq. (13) to hold the inverse fading curves

$$\begin{aligned}
w_{i,\tan}(\alpha) &= \frac{1}{\sqrt{1 + 2\,\tan(\frac{\pi\,\alpha}{2})\,r_{x_1,x_2} + \tan^2(\frac{\pi\,\alpha}{2})}} && (14)\\
&= \frac{\cos(\frac{\pi\,\alpha}{2})}{\sqrt{1 + 2\,r_{x_1,x_2}\,\sin(\frac{\pi\,\alpha}{2})\,\cos(\frac{\pi\,\alpha}{2})}}
\end{aligned}$$

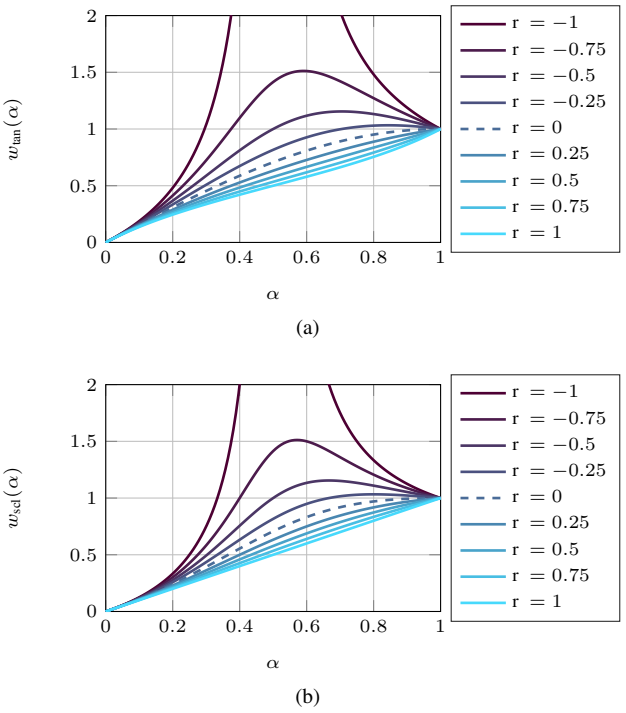


Figure 2: *Power-complementary cross-fading curves for different correlation coefficients*

and

$$\begin{aligned}
w_{i,\mathrm{scl}}(\alpha) &= \frac{1}{\sqrt{1 + 2\,\frac{\alpha}{1-\alpha}\,r_{x_1,x_2} + \frac{\alpha^2}{(1-\alpha)^2}}} && (15)\\
&= \frac{1-\alpha}{\sqrt{1 - 2\,(1 - r_{x_1,x_2})\,\alpha\,(1-\alpha)}}.
\end{aligned}$$

Correspondingly, the fading curves are derived as

$$w_{\tan}(\alpha) = \frac{\sin(\frac{\pi\,\alpha}{2})}{\sqrt{1 + 2\,r_{x_1,x_2}\,\sin(\frac{\pi\,\alpha}{2})\,\cos(\frac{\pi\,\alpha}{2})}} \tag{16}$$

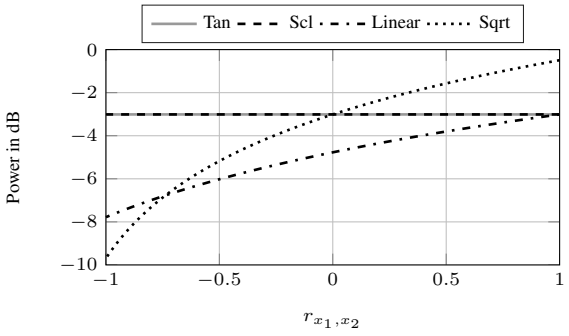
and

$$w_{\mathrm{scl}}(\alpha) = \frac{\alpha}{\sqrt{1 - 2\,(1 - r_{x_1,x_2})\,\alpha\,(1-\alpha)}}. \tag{17}$$

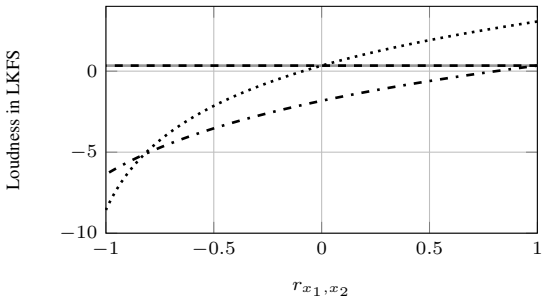
From Eq. (16,17) follows that perfectly power-complementary fading curves can be analytically designed whenever the correlation coefficient

$$r_{x_1,x_2} = \frac{\sum_{n=0}^{N-1}(x_1 - \mu_{x_1})(x_2 - \mu_{x_2})}{\sqrt{\sum_{n=0}^{N-1}(x_1 - \mu_{x_1})^2}\sqrt{\sum_{n=0}^{N-1}(x_2 - \mu_{x_2})^2}}. \tag{18}$$

is computed beforehand. The fading curves are plotted in Fig. 2 for different correlation coefficients. Apparently, the $w_{\tan}$ curve evolves from a sine curve to a slightly S-shaped curve for decreasing correlation values. In contrast, the w_{scl} changes from non-symmetric S-shaped curve to the linear curve. It should also be noted that the proposed cross-fading curve design also works flawlessly for negatively correlated signals. Certainly the amplitude

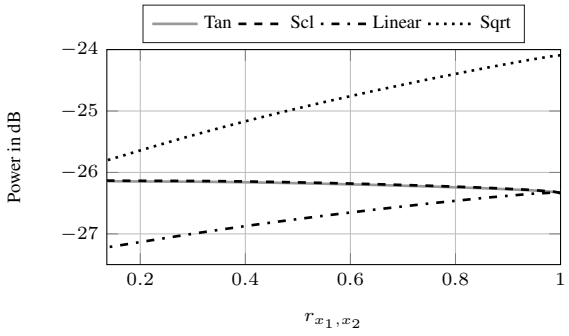


(a)

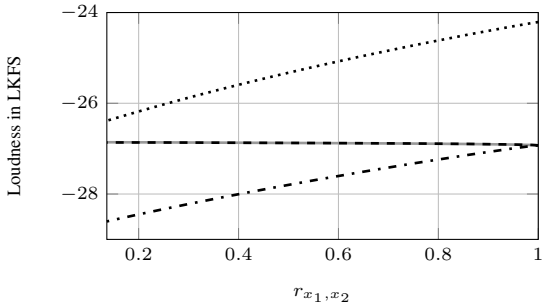


(b)

Figure 3: *Power and loudness according ITU BS-1770 of cross-faded sinusoids featuring different correlation coefficients* r_{x_1,x_2}



(a)



(b)

Figure 4: *Power and loudness according to ITU BS-1770 of cross-faded mono-to-stereo converted signals featuring different correlation coefficients* r_{x_1,x_2}

of the cross-fading curves increases to compensate the power loss caused by destructive interference. Cross-fading completely negatively correlated ($r_{x_1,x_2} = -1$) remains an undesirable scenario since infinite amplification in the transition center is required. However, cross-fading amplitude-inverted signals is improbable in practice anyways. Timbral coloration caused by cancellations of specific signal components remains a typical problem [1].

3. EVALUATION

Two simple experiments shall prove the effectiveness of the proposed fading curves. Two sinusoids of 1 s length featuring the same frequency ω_0 but different phase offsets ϕ

$$x_1(n) = \sin(\omega_0\, n) \tag{19}$$

$$x_2(n) = \sin(\omega_0\, n + \phi), \quad \phi \in [0, \ldots, 2\pi] \tag{20}$$

are cross-faded using the linear curve $w_{\text{lin}}(\alpha) = \alpha$, the square-root curve $w_{\text{sqrt}}(\alpha) = \sqrt{\alpha}$, and the curves from Eq. (16,17). The power of the cross-faded signals is plotted against the correlation coefficient in Fig. 3a). Using the proposed curves yields constant power for all values of r_{x_1,x_2} whereas the linear and square curve produce varying power progressions. The linear and square-root cross-fading is solely power-complementary for fully correlated ($r_{x_1,x_2} = 1$) and uncorrelated signals ($r_{x_1,x_2} = 0$), respectively. In addition to the signal power, the loudness according to ITU BS-1770 [6] was measured and illustrated in Fig. 3b). Basically, the loudness runs the same trend as the power.

In the following, the experiment shall be repeated using a real-world signal. An excerpt from the *Organ Handel* sample of length 1 s from the SQAM database [7] is processed with the mono-to-stereo conversion from [8] resulting in two decorrelated signals. The degree of decorrelation depends on the stereo width parameter and processing band width. The processing band is set from 50 Hz to 16 kHz whereas the stereo width parameter is varied from 0 to 1. Figure 4 illustrates the power and loudness of the cross-faded mono-to-stereo converted signals over the corresponding correlation coefficient. In contrast to the first experiment, the power varies slightly for the proposed fading curves. However, the loudness remains almost constant. Like beforehand, the application of the linear and square-root curves yield correlation-dependent varying results with deviations > 1 dB.

Both experiments validate the effectiveness of this novel approach of power-complementary cross-fading. Fortunately, it is also useful in the context of dry-wet mixing as shown in the following.

4. DRY-WET MIXING

Involving audio effects to shape the sound of a single or multiple instruments is a key task of musicians, audio engineers and producers. Many effect units offer a so-called dry-wet balance knob defining the proportion of unprocessed (dry) and processed (wet) signal in the output. Hence, the resulting mix can be described as a weighted sum

$$x_{\text{mix}}(n) = g_d\, x_d(n) + g_w\, x_w(n), \tag{21}$$

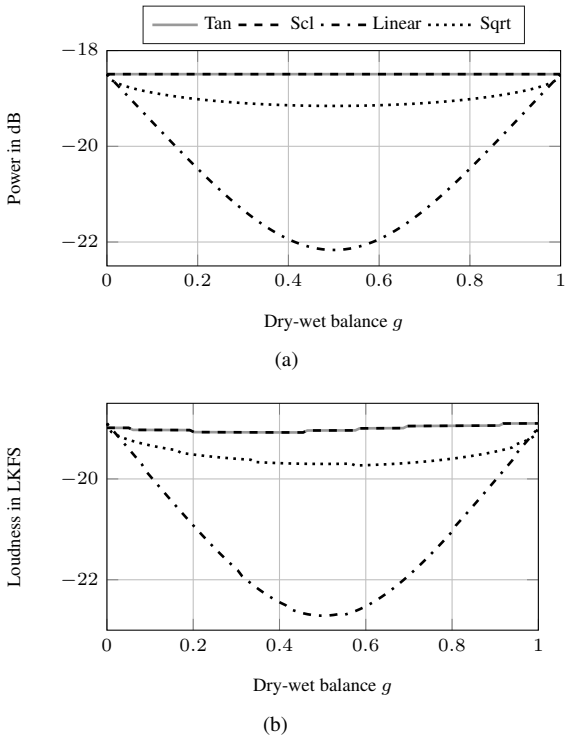


Figure 5: *Power and loudness according to ITU BS-1770 of reverberated and original signal mix using different dry-wet balances*

of dry signal $x_d(n)$ and wet signal $x_w(n)$. The relationship of the weighting coefficients g_d and g_w defines the dry-wet balance g. Typically, the dry-wet balance covers the value range from 0 (full dry) to 1 (full wet). Since the amount of decorrelation added to the dry signal through the effect processing is typically unknown, one has to expect power and resulting loudness variations in the mixed signal for different dry-wet balances similar to the cross-fading problem of the previous section. The design formulas Eq. (14-17) can directly be utilized to obtain the weighting coefficients by inserting the dry-wet balance g instead of the normalized time index α as proven by the following experiment.

An excerpt of the SQAMs *Guitar Sarasate* sample of length 10 s is convolved with the *M7 - 3 Rooms 08 Music Room* impulse response of the Samplicity's Impulse Response Library [9] to realize a reverberation effect. The power and loudness of $x_{\text{mix}}(n)$ is measured for different dry-wet balances g and plotted in Fig. 5. The dry and wet signal feature a correlation coefficient $r_{x_d,x_w} = -0.1417$. Hence, destructive interferences occur and the power of the mixed signal is significantly reduced for the linear and square-root relation of the weighting coefficients. In contrast, obtaining weighting coefficients with the proposed design results in constant power and almost constant loudness for all dry-wet balances. However, applying this approach is solely possible for off-line processing since the correlation coefficient has to be computed for the whole signal length before the dry-wet mix is performed and consequently the property of power preservation is solely achieved in temporal average. Nevertheless, the proposed approach can be utilized in a real-time application when the cross-correlation coefficient is sample-wise estimated as shown in [10].

5. CONCLUSION

Cross-fading is a key tool in audio editing of whatever form. The selection of fading curves by audio professionals is mainly based on experience and hence, is more or less imprecise. This work analytically derives perfectly power-complementary fading curves based on the exactly measured correlation of the signals to be cross-faded. The derivation is based on a simple mixing signal model and some basic assumptions which are met by most audio signals. Experiments on synthetic and real-world signals validate the effectiveness of the fading curve design. The proposed fading curve design can also be utilized in the context of dry-wet mixing due to similar boundary conditions. An additional experiment exposes the power-preservation for mixing dry and reverberated signals for any dry-wet balance using the proposed design.

6. REFERENCES

[1] S. Langford, *Digital Audio Editing: Correcting and Enhancing Audio in Pro Tools, Logic Pro, Cubase, and Studio One*, Digital Audio Editing: Correcting and Enhancing Audio in Pro Tools, Logic Pro, Cubase, and Studio One. Taylor & Francis, 2013.

[2] T. Görne, *Tontechnik*, Hanser, 2008.

[3] Jeffs Rick, "Evolution of the DJ Crossfader," Tech. Rep., Rane Corporation, 1999.

[4] Marco Fink, Martin Holters, and Udo Zölzer, "Comparison of various predictors for audio extrapolation," in *Proceedings of the 16th International Conference on Digital Audio Effects, DAFx-13, Maynooth, Ireland, September 2-5, 2013*, Erlangen, 2013.

[5] Marco Fink and Udo Zölzer, "Low-delay error concealment with low computational overhead for audio over IP applications," in *Proceedings of the 17th International Conference on Digital Audio Effects, DAFx-14, Erlangen, Germany, September 1-5, 2014*, Erlangen, 2014.

[6] "ITU Recommendation ITU-R BS.1770-3, Algorithms to measure audio programme loudness and true-peak audio level," Tech. Rep., International Telecommunications Union, August 2012.

[7] "EBU - TECH2353, Sound Quality Assessment Material recordings for subjective tests," Tech. Rep., European Broadcasting Union, September 2008.

[8] Marco Fink, Sebastian Kraft, and Udo Zölzer, "Downmix-compatibe conversion from mono to stereo in time-and frequency-domain," in *Proceedings of the 18th International Conference on Digital Audio Effects, DAFx-15, Trondheim, Norway, Nov 30-Dec 3, 2015*, Trondheim, 2015.

[9] "Samplicity's Bricasti M7 Impulse Response Library v1.1," `http://www.samplicity.com/bricasti-m7-impulse-responses/`.

[10] Ronald M. Aarts, Roy Irwan, and Augustus J.E.M. Janssen, "Efficient tracking of the cross-correlation coefficient," *Speech and Audio Processing, IEEE Transactions on*, vol. 10, no. 6, pp. 391–402, 2002.

TIME-DOMAIN IMPLEMENTATION OF A STEREO TO SURROUND SOUND UPMIX ALGORITHM

Sebastian Kraft, Udo Zölzer

Department of Signal Processing and Communications
Helmut-Schmidt-University
Hamburg, Germany
sebastian.kraft@hsu-hh.de

ABSTRACT

This paper describes a time-domain algorithm to upmix stereo recordings for an enhanced playback on a surround sound loudspeaker setup. It is mainly the simplified version of a previously published frequency-domain algorithm where the standard short-time Fourier transform is now replaced by an IIR filter bank. The design of complementary filter blocks and their arrangement in a tree structure to form a filter bank are derived. The arithmetic complexity of the filter bank itself and of the complete upmix algorithm is analysed and compared to the frequency-domain approach. The time-domain upmix is less flexible in its configuration but achieves an audio quality comparable to the frequency-domain implementation at a fraction of its computational cost.

1. INTRODUCTION

Upmixing is a process to generate additional channels when an audio source is intended to be played back over a setup with more loudspeakers than source channels. Ideally, a good upmix should redistribute the input signal to all available loudspeakers providing an immersive listening experience without compromising the original character of the stereo track. For example, the azimuth position of sources in a stereo mix as well as the overall timbre and spatial character should be preserved.

Most algorithms in this area are based on the same processing principles applied to a time-frequency-domain representation of the input signal. The channels of a stereo or multi-channel recording are described as a weighted sum of direct signal sources overlaid by an uncorrelated ambient signal [1, 2]. First, the azimuth positions or panning coefficients of the sources are estimated under the assumption that only one dominant source is active at a single time-frequency instant. Next, the direct and ambient components are separated with the knowledge of the panning coefficients. Having the separated signals with their related source positions it is possible to remix the original content considering any different target loudspeaker configuration.

Other algorithms mainly focus on a direct and ambient signal separation [3, 4], where [3] is one of the few examples for a time-domain approach. However, as a normalised least mean squares (NLMS) method is used to adapt an FIR extraction filter with several hundreds of coefficients it is computationally quite demanding. Further examples for time-domain approaches are Dolby Pro Logic I and II [5] which were widely spread in the consumer area a decade ago and were optimised for a cost effective implementation using simple time-domain operations. Basically, only a few subtractions and additions of the left and right channels with additional phase shifts and VCAs (voltage controlled amplifiers) for simple directional steering are required. But due to full-band processing of the input signal the capability to separate multiple concurrent sources is quite limited.

The idea behind the approach presented in this paper and its derivation is similar to the previous work in [6, 7] but all derivations are rewritten to yield an equivalent time-domain formulation. This allows to replace the short-time Fourier transform (STFT), previously used to create a time-frequency representation of the input signal, by a non-subsampling filter bank as in [8]. It is build of complementary IIR filters arranged in a tree structure and allows a perfect allpass reconstruction. In [2, 7] it was already pointed out that the STFT spectra resolution could be reduced to Bark bands without impairments. Hence, a low-resolution filter bank would be well suited to search for an optimal trade-off between complexity and frequency resolution and to analyse the influence of the time-frequency resolution on the quality of the resulting upmix.

The underlying stereo signal model and estimation of source positions as well as the direct and ambient component separation will be introduced in Sec. 2. Afterwards, the design of the filter bank is described in Sec. 3 together with an analysis of its arithmetic complexity. Section 4 shows the application of the proposed separation in the context of a stereo to multi-channel surround sound upmix and compares the results with its frequency-domain counterpart.

2. SIGNAL SEPARATION

2.1. Stereo signal model

The left and right channels of a stereo signal

$$x_L(n) = \left[\sum_{i=1}^{I} g_{L_i} \cdot s_i(n)\right] + a_L(n) \tag{1}$$

$$x_R(n) = \left[\sum_{i=1}^{I} g_{R_i} \cdot s_i(n)\right] + a_R(n) \tag{2}$$

can be described as a weighted sum of I source signals $s_i(n)$ and additive uncorrelated ambient signals $a_L(n)$ and $a_R(n)$ in the left and right channel, respectively. The weightings g_{L_i} and g_{R_i} of the individual sources are called panning coefficients and are bound between zero and one. Their squared sum should be equal to one ($g_{L_i}^2 + g_{R_i}^2 = 1$) to achieve a constant power and loudness of a panned source independent of its current position. By applying a

filter bank, the signal model

$$x_L(b,k) = \left[\sum_{i=1}^{I} g_{L_i} \cdot s_i(b,k)\right] + a_L(b,k) \tag{3}$$

$$x_R(b,k) = \left[\sum_{i=1}^{I} g_{R_i} \cdot s_i(b,k)\right] + a_R(b,k) \tag{4}$$

is transformed into a time-frequency representation. The variables b and k denote the time and band index, whereas b is equal to n for non sub-sampling filter banks and will be used in the following.

Two simplifications are required to invert the signal model and to recover sufficient approximations of the source signals and their panning parameters. First, it is a typical assumption that at a certain time instant n and in a frequency band k only a single dominant source s_u is active and the contribution of other sources is close to zero ($\sum_{\forall i \neq u} |s_i(n,k)| \approx 0$) if the time and frequency resolution is not too small [9]. This allows to summarise the time-frequency representations of the individual sources

$$g_L(n,k) \cdot s(n,k) \approx \sum_{i=1}^{I} g_{L_i} \cdot s_i(n,k) \tag{5}$$

into a single source $s(n,k)$ and panning coefficients $g_{L/R}(n,k)$.

Second, the left and right ambient signal can be expected to sound similar but due to different paths and reflections in the room, they are decorrelated. Hence, the left and right ambient signals

$$a_L(n) = \mathcal{H}_{A_L}\{a(n)\}, \qquad a_R(n) = \mathcal{H}_{A_R}\{a(n)\}$$

originate from a single ambient signal $a(n)$ which has been modified by an abstract filter operation $\mathcal{H}(\cdot)$. Combining both assumptions, a simplified time-frequency signal model

$$x_L(n,k) = g_L(n,k) \cdot s(n,k) + \mathcal{H}_{A_L}\{a(n,k)\}, \tag{6}$$

$$x_R(n,k) = g_R(n,k) \cdot s(n,k) + \mathcal{H}_{A_R}\{a(n,k)\} \tag{7}$$

with a reduced number of unknowns can be obtained.

If a STFT would be used for the time-frequency transform, the abstract filter operation in the signal model could be implemented as a multiplication of $A(n,k)$ with a frequency response

$$H_A(k) = \gamma(k) \cdot e^{j\phi(k)}, \qquad \begin{aligned} 0 < \gamma(k) &< 1 \\ 0 < \phi(k) &\leq \pi \end{aligned} \tag{8}$$

consisting of a band-wise magnitude $\gamma(k)$ and phase $\phi(k)$ coefficient. This leads to a signal model

$$X_L(b,k) = g_L(b,k) \cdot S(b,k) + H_{A_L}(k) \cdot A(b,k) \tag{9}$$

$$X_R(b,k) = g_R(b,k) \cdot S(b,k) + H_{A_R}(k) \cdot A(b,k) \tag{10}$$

in the time-frequency domain. The actual decorrelation filter parameters are usually not known for general music material and are difficult to estimate. However, a coarse approximation of a decorrelation filter response by a random distribution can lead to realistically sounding decorrelated signals [10, 11, 12] and was also successfully used for ambience extraction in [7]. Furthermore, the desired correlation of the left and right ambient signal can be nicely adjusted by the phase angle ϕ where $\phi = \pi/2$ would yield a broadband correlation of 0 and with $\phi = \pi$ the resulting ambient signals are out of phase (correlation is -1).

With a time-domain filter bank, the application of the ambience decorrelation filters $\mathcal{H}_{A_L}$ and $\mathcal{H}_{A_R}$ is not as trivial as in the frequency-domain. An equivalent time-domain formulation of the filter in (8) with a corresponding signal model could look like

$$h_{a_{L/R}}(k) = \gamma(k) \cdot \pm 1, \qquad 0 < \gamma(k) < 1 \tag{11}$$

$$x_L(n,k) = g_L(n,k) \cdot s(n,k) + h_{a_L}(k) \cdot a(n,k) \tag{12}$$

$$x_R(n,k) = g_R(n,k) \cdot s(n,k) + h_{a_R}(k) \cdot a(n,k) \tag{13}$$

where the filter is modelled with a band-wise gain $\gamma(k)$ and a phase shift of π can be achieved by choosing opposite sign coefficients in each channel. Other phase shifts would require a time-domain filtering of each sub-band and would impose the problem of inverse filtering when extracting the direct and ambient signal in Sec. 2.3.

2.2. Estimation of source directions

For typical music mixes the amplitude of the ambient signal $a(n,k)$ can be assumed to be far less than the amplitude of the direct signal $s(n,k)$. This also means that the power of the left and right channels

$$P_{x_L}(n,k) \approx g_L^2(n,k) \cdot P_s(n,k) \tag{14}$$

$$P_{x_R}(n,k) \approx g_R^2(n,k) \cdot P_s(n,k). \tag{15}$$

is mainly depending on the weighted direct signal power. Rearranging and solving equations (14)-(15) with the constraint $g_L^2 + g_R^2 = 1$, the panning coefficients

$$\hat{g}_L(n,k) = \sqrt{\frac{P_{x_L}(n,k)}{P_{x_L}(n,k) + P_{x_R}(n,k)}} \tag{16}$$

$$\hat{g}_R(n,k) = \sqrt{\frac{P_{x_R}(n,k)}{P_{x_L}(n,k) + P_{x_R}(n,k)}}. \tag{17}$$

can be estimated from the power of the left and right stereo channels. A simple estimate of the power of a sub-band signal $x(n,k)$

$$P_x(n,k) = \alpha \cdot P_x(n-1,k) + (1-\alpha) \cdot x(n,k)^2 \tag{18}$$

can be determined by recursive averaging with a coefficient $0 < \alpha < 1$. Other power estimates, in particular with signal adaptive coefficients, may be investigated in the future and could for example improve processing of transients.

The "stereophonic law of sines" [13]

$$\frac{g_L - g_R}{g_L + g_R} = \frac{\sin(\theta)}{\sin(\theta_0/2)} = -\Psi \tag{19}$$

describes the perceived angle θ of a source if its amplitude is weighted by $g_{L/R}$ for playback on a left and right loudspeaker with an angle θ_0 between both. The normalised position index Ψ, ranging from -1 for left and $+1$ for right positions, combines the coefficients $g_{L/R}$ in a single value. From (14)-(15) and (19) one can derive estimates for the position index and angle

$$\hat{\Psi}(n,k) = \frac{\sqrt{P_{x_R}(n,k)} - \sqrt{P_{x_L}(n,k)}}{\sqrt{P_{x_R}(n,k)} + \sqrt{P_{x_L}(n,k)}} \tag{20}$$

$$\hat{\theta} = \arcsin\left(\sin(\theta_0/2) \cdot \hat{\Psi}(n,k)\right) \tag{21}$$

based on the power of the left and right stereo channel.

2.3. Direct and ambient signal separation

When the panning coefficients or their estimates from the previous section are known, the signal model (12)-(13) can be transformed mathematically to get the direct and ambient signal components

$$\hat{s}(n,k) = \frac{h_{a_R}(k)\cdot x_L(n,k) - h_{a_L}(k)\cdot x_R(n,k)}{h_{a_R}(k)\cdot \hat{g}_L(n,k) - h_{a_L}(k)\cdot \hat{g}_R(n,k)} \tag{22}$$

$$\hat{a}(n,k) = \frac{\hat{g}_L(n,k)\cdot x_R(n,k) - \hat{g}_R(n,k)\cdot x_L(n,k)}{\hat{g}_L(n,k)\cdot h_{a_R}(k) - \hat{g}_R(n,k)\cdot h_{a_L}(k)}\,. \tag{23}$$

For low-resolution filter banks as used in the upmix application the random decorrelation filter gains $\gamma(k)$ from (11) would cause an audible band-wise panning instead of the desired diffuse decorrelation. Hence, the decorrelation filters are set to

$$h_{a_L}(k) = 1, \qquad h_{a_R}(k) = -1 \tag{24}$$

and in this case the extraction formula can be further simplified to

$$\hat{s}(n,k) = \frac{x_L(n,k) + x_R(n,k)}{\hat{g}_L(n,k) + \hat{g}_R(n,k)} \tag{25}$$

$$\hat{a}(n,k) = \frac{\hat{g}_L(n,k)\cdot x_R(n,k) - \hat{g}_R(n,k)\cdot x_L(n,k)}{\hat{g}_L(n,k) + \hat{g}_R(n,k)}\,. \tag{26}$$

As already pointed out in [6], the above equations are very similar to a classical mid-side decomposition performed in sub-bands. The main difference is the weighting with the estimated panning coefficients to allow proper separation of ambient and direct components in case the direct signal is not panned to the center.

3. COMPLEMENTARY FILTERBANK

An IIR filter bank as in [8, 14] is used to create a time-frequency representation of the input signal. It consists of complementary filter blocks arranged in a tree structure and additional allpass sections are inserted to achieve an overall allpass reconstruction property. The individual bands will not be downsampled, hence the reconstruction can be a simple summation of the sub-bands.

3.1. Complementary allpass filter structure

The basic building block of the filter bank is a filter pair F and $\tilde{F}$ which split an input signal into a lower and higher complementary band. The two filters are power complementary if their transfer functions satisfy

$$\left|F(e^{j\omega})\right|^2 + \left|\tilde{F}(e^{j\omega})\right|^2 = 1 \tag{27}$$

and if the sum of the transfer functions additionally yields an allpass magnitude response

$$\left|F(e^{j\omega}) + \tilde{F}(e^{j\omega})\right| = 1 \tag{28}$$

they are *doubly complementary* [15]. Therefore, by summation of both filter outputs it is possible to recover the input signal except for a certain phase shift.

The pair of filters could be directly designed with standard methods allowing for above constraints. However, by decomposing a single filter prototype F in two parallel allpass sections it is possible to obtain a second complementary output with just one further subtraction. The detailed derivation can be found in [16] and will be shortly summarised in the following sections.

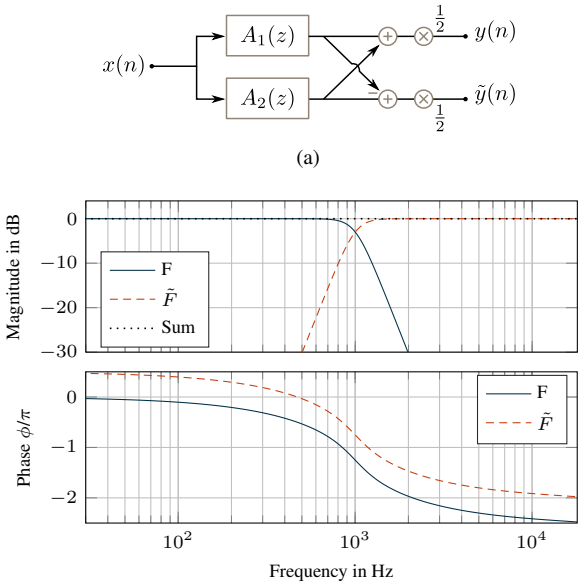


Figure 1: *Doubly complementary allpass filter structure (a) with exemplary magnitude and phase response for a 5th order butterworth lowpass (b).*

3.1.1. Allpass Decomposition

An IIR filter $F(z)$ can be split into a parallel sum

$$F(z) = \frac{P(z)}{D(z)} = \frac{1}{2}\Big(A_1(z) + A_2(z)\Big) \tag{29}$$

of two allpass filters $A_1(z)$ and $A_2(z)$ in case the following requirements are met:

1. the order N of $F(z)$ is odd
2. $P(z)$ is a mirror symmetric polynomial, $P(z^{-1}) = z^N\,P(z)$
3. $F(z)$ has real coefficients and $|F(e^{j\omega})| \leq \infty$ (*bounded real* transfer function)

This holds true for typical IIR filter designs like Butterworth, Chebyshev and elliptic filters. Furthermore, a complementary filter

$$\tilde{F}(z) = \frac{Q(z)}{D(z)} = \frac{1}{2}\Big(A_1(z) - A_2(z)\Big) \tag{30}$$

can be easily obtained by the difference of the allpass filters as depicted in Fig. 1. The summed transfer function of a complementary filter stage formed by (29) and (30)

$$H(z) = F(z) + \tilde{F}(z) = A_1(z) \tag{31}$$

has the required allpass characteristic. This means that perfect magnitude reconstruction is possible but the phase shift and group delay of $A_1(z)$ will remain in the reconstructed signal.

The respective order of the allpass sections is $N_1 = (N-1)/2$ and $N_2 = (N+1)/2$ where $N_1 + N_2 = N$. Table 1 gives an overview of the required instructions per sample for a complementary allpass filter structure compared to a direct form implementation with two separate filters. It can be seen that the complementary filter created with the allpass decomposition requires only little more than half of the instructions as a direct-form implementation with two filters.

	Mult.	Add.	Overall
Compl. Allpass struct.	$N+2$	$2N+2$	$3N+4$
2x Direct-Form filters	$3N+1$	$4N$	$7N+1$

Table 1: *Number of multiplications, additions and overall operations for a single complementary filter stage per sample.*

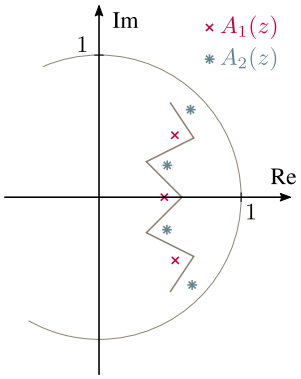


Figure 2: *Alternating selection of poles in the z-plane to create an allpass decomposition.*

3.1.2. Filter design

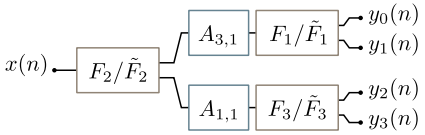
The general form of the two allpass filters

$$A_1(z) = \prod_{k=1}^{N_1} \frac{z^{-1}-p_k}{1-z^{-1}\,p_k}$$
$$A_2(z) = \prod_{k=N_1+1}^{N} \frac{z^{-1}-p_k}{1-z^{-1}\,p_k} \qquad (32)$$

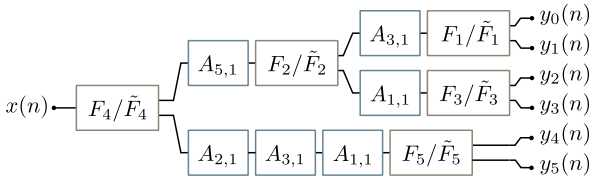
is fully defined by the knowledge of the poles p_k, as the zeros $z_k = 1/p_k$ are the inverse of the poles. From (29) and (30) it is apparent that $F(z)$ or $\tilde{F}(z)$ feature the same poles as the sum or difference of $A_1(z)$ and $A_2(z)$. Hence, the poles of the allpass filters are just subgroups of the poles already contained in $F(z)$.

An algorithm to derive a suitable grouping of the poles from a filter transfer function $F(z)$ is described in [16] where first the polynomial $Q(z)$ is determined and then the zeros of $P(z)+Q(z)$ are calculated. The zeros outside the unit circle will form the first allpass $A_1(z)$ and the zeros inside the unit circle will form the second allpass $A_2(z)$. The algorithm to calculate $Q(z)$ is recursive and requires several polynomial multiplications which may become numerically unstable for high order filters. In particular Butterworth designs lead to very small valued coefficient sets and it may become difficult to find a stable decomposition for filter orders above 5.

A more simple graphical allpass decomposition is given in [17] which directly makes use of the poles typically derived in the IIR design procedure and hence is less prone to numerical errors. The poles of Butterworth, Chebyshev and elliptic lowpass filters are placed on an ellipsoid curve inside the unit circle. An alternating separation as depicted in Fig. 2 yields two groups of poles, whereas the smaller group with $(N-1)/2$ poles is assigned to $A_1(z)$ and the remaining $(N+1)/2$ poles are assigned to $A_2(z)$.



(a) $M = 3$ filters, 4 output bands



(b) $M = 5$ filters, 6 output bands

Figure 3: *Two exemplary filter bank structures consisting of 3 and 5 complementary filters.*

3.2. Filter bank tree structure

In the following, $H_k(z) = Y_k(z)/X(z)$ will denote the transfer function of a filter bank channel k. Overall M complementary filters $F_m(z)/\tilde{F}_m(z)$ are used and $A_{m,1}(z)$ and $A_{m,2}(z)$ are the corresponding allpass decompositions.

The complementary filter stages are cascaded in a tree structure to successively divide the bands and to create $M+1$ outputs. An exemplary filter bank with 4 outputs and 3 complementary filters is shown in Fig. 3 a). The additional allpass sections $A_{3,1}(z)$ and $A_{1,1}(z)$ are required to guarantee an overall allpass transfer function after summing the sub-band outputs for reconstruction. For example, when summing the individual output transfer functions without the allpass sections it appears that

$$\begin{aligned} H(z) &= H_0(z) + H_1(z) + H_2(z) + H_3(z) \\ &= F_2(z)\cdot\Big(F_1(z)+\tilde{F}_1(z)\Big) + \tilde{F}_2(z)\cdot\Big(F_3(z)+\tilde{F}_3(z)\Big) \\ &= F_2(z)\cdot A_{1,1}(z) + \tilde{F}_2(z)\cdot A_{3,1}(z), \end{aligned}$$

is not an allpass. However, by adding additional allpass sections after $F_2/\tilde{F}_2$ the overall transfer function

$$\begin{aligned} H(z) &= F_2(z)\cdot A_{3,1}(z)\cdot A_{1,1}(z) + \tilde{F}_2(z)\cdot A_{1,1}(z)\cdot A_{3,1}(z) \\ &= A_{2,1}(z)\cdot A_{1,1}(z)\cdot A_{3,1}(z) \end{aligned}$$

becomes allpass. Another exemplary filter bank structure to yield six doubly complementary bands is given in Fig. 3 b). If a reconstruction is not required in the desired application and the filter bank is only used for signal analysis, the additional allpass sections can be omitted without altering the power of the sub-band signals.

More details about the general setup of tree-structured recursive filter banks and in particular about the placement of the additional allpass sections to yield allpass reconstruction properties can be found in [14]. In a summary, the basic rules are:

- The overall transfer function for a bank of M filters is

$$H(z) = \sum_{k=0}^{M} H_k(z) = \prod_{m=1}^{M} A_{m,1}(z). \qquad (33)$$

	Mult.	Add.	Overall
Compl. filters	$N+2$	$2\,N+2$	$M\cdot(3\,N+4)$
Add. allpass	N_1	$2\,N_1$	$M_A\cdot 3\,N_1$

Table 2: *Number of multiplications, additions and overall operations per sample for a filter bank with M filters of order N. M_A denotes the required number of allpass sections for perfect magnitude reconstruction.*

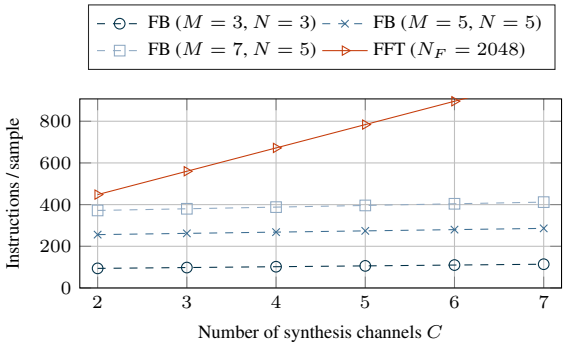


Figure 4: *Arithmetic complexity for filter banks with 2 analysis and C synthesis channels compared to an STFT analysis/synthesis.*

Hence, the overall group delay only depends on the $A_{m,1}(z)$ sections and can be minimised by always applying the lower number of poles to $A_{m,1}(z)$ during the allpass decomposition.

- It has to be assured that every sub-band signal passes all possible allpass sections ($A_{m,1}(z)$, $m=[1\ldots M]$) on its way through the filter bank. Missing allpass sections have to be inserted into the signal path.
- The required number of additional allpass sections can be minimised with a simple rule: At every branch we have to add the corresponding allpass sections of all filters in the opposite branch.

3.3. Arithmetic complexity

The arithmetic complexity in terms of required multiplications and additions for a filter bank with M filters of order N is given in Table 2. It can be seen that the complexity linearly increases with the number of bands and filter order.

The STFT is the standard transform for analysis/synthesis systems and the question is how it would compare to a filter bank based approach in terms of arithmetic complexity. The number of operations to transform a block of length N_F with a standard Cooley-Tukey FFT (radix-2) can be estimated to be in the range of $\sim\ 5\log(N_F)\,N_F$. With a typical overlap of 75 % and real valued time-domain signals this yields $\sim 10\log(N_F)$ instructions per sample and transform.

Figure 4 compares several recursive filter banks and a STFT variant with a block size of $N_F=2048$ samples. The complexity of the filter bank analysis/synthesis system is mainly independent of the number of output channels as the synthesis is a simple summation of all sub-bands. In contrast for the STFT, as well as sub-sampled filter banks, an equal complexity synthesis step is required and the complexity increases linearly with the number of synthesis channels. It can be seen that a filter bank may be in particular advantageous if the application only requires a few bands with low filter orders but arbitrary frequency resolution and if more output than input channels are to be generated.

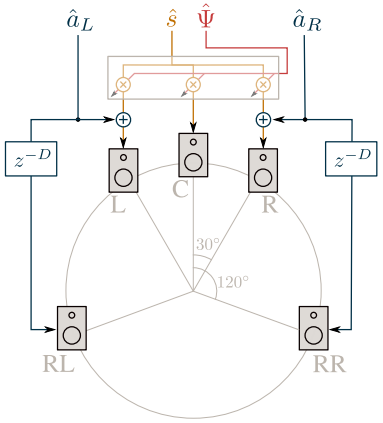


Figure 5: *Stereo to 5 channel upmix.*

	M	M_A	N	Frequencies [Hz]
FB I	3	2	3	220, 1000, 4000
FB II	5	6	3	220, 500, 1000, 2000, 5000
FB III	5	6	5	220, 500, 1000, 2000, 5000
STFT	$N_F=2048$, $N_H=512$			

Table 3: *Chosen filter bank parameters.*

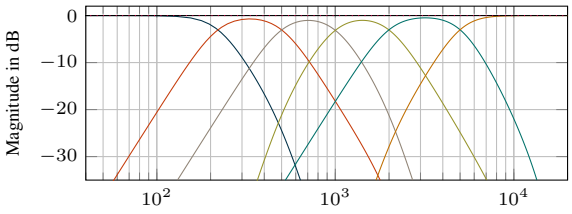
4. UPMIX APPLICATION

The estimated direct signal source positions and the separated direct and ambient signals can be used to create a stereo to surround upmix following a signal flow as depicted in Fig. 5. The direct signal is repanned on the front loudspeakers, for example by using *Vector Base Amplitude Panning* (VBAP) [18], while the ambient signals are added to the corner loudspeakers. To decorrelate the front from the rear ambient signals, a short delay is included but it would also be possible to apply more advanced time-domain decorrelators as described in [19].

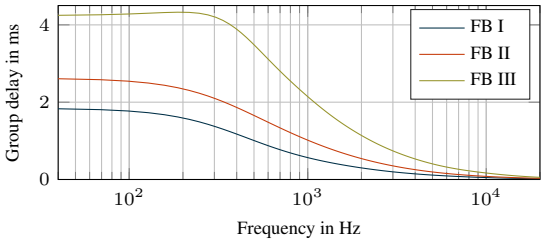
4.1. Filter bank configuration

Several combinations of filter order as well as corner frequencies and number of bands were tested and the corresponding parameters are given in Table 3. The magnitude response for configuration FB II is depicted in Fig. 6 a) and the group delay for all given configurations is plotted in Fig. 6 b). The coefficient α for the recursive power estimation per sub-band was set to $\alpha=5\cdot 10^{-4}$ in the following experiments.

It turned out, that the positions estimated from the lowest and highest band of the filter bank are not reliable as there is too much overlap of individual sources in these frequency regions. Therefore, in the following only filter bank outputs 1 up to $M-1$ will

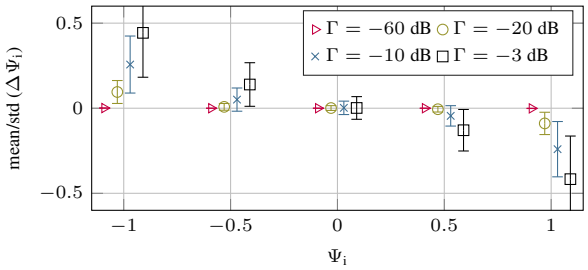


(a) Magnitude response of FB II

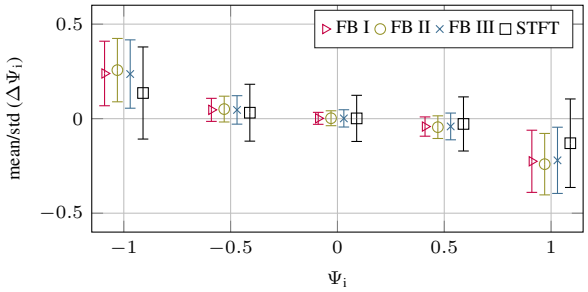


(b) Group delay

Figure 6: *Magnitude response and group delay for several filter bank configurations.*



(a) Filter bank FB II



(b) Comparison of filter banks and STFT at $\Gamma = -10$ dB

Figure 7: *Mean and standard deviation of $\Delta\Psi_i$ for various configurations.*

be processed by the upmix and channel 0 and M will be directly fed to the front left and right loudspeakers. In the end, this corresponds to a band-limiting of the extracted ambient and center channels.

4.2. Evaluation of estimated positions

It was assumed in Sec. 2.2 that a low power ambient signal will not influence the estimation of the panning coefficients. However, the question is how the accuracy of the panning estimation is impaired if the ambient signal power is increased. To further investigate this, a single direct signal has been panned to various positions Ψ_i and ambience was added with a *Large Hall* impulse response from a *Bricasti M7* stereo reverb unit. The ambient to direct power ratio (ADR)

$$\Gamma = 10\ \log_{10}\left(\frac{P_{A_L} + P_{A_R}}{P_S}\right) \tag{34}$$

describes the ratio between the overall ambient and direct signal power where

$$P_x = \sum_n x(n)^2$$

denotes power of a signal $x(n)$. The resulting mean and standard deviation of the position error

$$\Delta\Psi_i(n,k) = \hat{\Psi}_i(n,k) - \Psi_i \tag{35}$$

is plotted for several ADR values in Fig. 7 a) and it can be seen that stronger ambient components directly lead to a higher error. As the ambient signal has near equal power in the left and right channel the estimated positions will be biased towards the center and the error further increases for strongly panned sources. In Fig. 7 b) the error for $\Gamma = -10$ dB is compared between the different filter bank configurations from Table 3 and also to the STFT based method from [6]. It is apparent that the error is relatively independent of the filter bank configuration. Compared to the STFT based position estimation the mean error for strongly panned sources is higher, however, the standard deviation is considerably lower for all source positions.

4.3. Ambient signal quality

Although the extracted ambient signals sound realistic they are too different from the real signals which were added in the mixing process and a direct comparison, e.g. by error energy, is usually not significant. Therefore, only signal characteristics can be compared and it is well known that ambient signals should be diffuse and decorrelated to create an immersive sound field. The diffuseness and uniformity of an ambient signal can be measured by inter-channel cross-correlation (ICC) and inter-channel level differences (ICLD), whereas for real ambient signals both are observed to be close to zero. With a high-resolution frequency-domain implementation it is possible to apply versatile decorrelation filters with little effort as described in detail in [7]. With a low-resolution filter bank this is not feasible and with the simple filters as in (24) the left and right ambient signals are just phase-inverse copies of each other. Hence, the ICC is -1 and the ICLD is 0. Listening to the isolated ambient sound field, the out of phase character creates an impression of width but it can also evoke unpleasant cancellation artefacts in particular during head movements. This is the main difference to the frequency-domain implementation [7] where the resulting ICC was freely adjustable.

4.4. Arithmetic complexity

A detailed analysis of the arithmetic complexity of the frequency-domain upmix algorithm was done in [7] and the numbers in Table 4 a) are based on these findings. However, for a fair comparison with the simplified time-domain approach, the ambient decor-

relation filters in the frequency-domain were set to $H_{A_L}(k) = 1$ and $H_{A_R}(k) = -1$ according to (24) and no further decorrelation between front and rear channels is applied. This partly reduces the arithmetic complexity of the direct and ambient decomposition.

A corresponding analysis of the arithmetic complexity of the filter bank based upmix algorithm is given in Table 4 b). As expected, the base cost for analysis and synthesis is drastically reduced for typical filter bank configurations. In contrast, the number of operations required for the upmix processing has increased as all sub-bands are processed at full rate. Processing an upmix from stereo to 5-channel surround in the frequency-domain requires about 38 MFlops, whereas the corresponding time-domain variant with a low-resolution filter bank is in a range between 8 and 20 MFlops. This could be further reduced by a lowered update rate of the estimated positions and repanning coefficients but was out of the scope of this study due to the large amount of possible solutions and parameters.

It has to be noted that highly optimised FFT implementations (e.g. the FFTW library[1]) are widely available and can easily reduce the processing time for a transform by a factor of three up to five. Of course, similar optimization strategies like code vectorisation could be applied to the filter bank. But as no ready to run solutions are available this would require a comparably high effort and programming expertise.

4.5. Discussion

Informal listening tests confirm that the generated 5-channel surround upmix is convincing and works well with typical commercial studio music recordings. Compared to the stereo input, the source positions are well retained and no timbral coloration or other artefacts are audible. The center loudspeaker successfully stabilises the front image, in particular for listeners outside of the sweet spot and the out of phase artefacts observed with the isolated ambient signal are masked by the usually quite strong direct signal and are not annoying. However, in a direct A/B comparison with the STFT based frequency-domain upmix it is possible to spot subtle differences. The ambient signal is weaker and does not create a sound field as diffuse as experienced from the STFT approach.

First tests with discrete stereo microphone recordings showed good results for coincident microphone arrangements. In contrast, with non-conincident microphone setups a proper estimation of directions and a separation of direct and ambient components is not possible at the same quality. The reason is that phase shifts are introduced in the direct signals between both channels violating the basic signal model assumptions in Sec. 2.1.

Overall, the time-domain implementation upmix offers obvious benefits compared to a simple stereo playback even with the lowest resolution filter bank FB I. The actual configuration of the filter bank does not seem to have a strong influence on the results. More important seems the fact that low and high frequency content is separated by the filter bank and will not interfere with the estimation in the important mid frequency regions.

Another interesting aspect is the filter bank group delay of less than 5 ms which is quite low compared to the blocking delay of a STFT based analysis and synthesis. A STFT configuration with a block size $N_F = 2048$ at 44.1 kHz sample rate would yield about 46 ms delay. Therefore, the time-domain upmix is well suited for real-time applications and implementations on low-cost stream-based DSPs where no block-based processing is possible or FFT implementations are not available.

5. CONCLUSION

The goal of this study was to develop a low-complexity time-domain upmix algorithm. First, an equivalent time-domain formulation to a previously described frequency-domain method for estimation of source positions and separation of direct and ambient signal components has been derived. A filter bank is then used to create a time-frequency representation of the input signal and its design based on complementary IIR filters is described.

The arithmetic complexity of the filter bank and the filter bank-based upmix is compared to a STFT based approach. The time-domain variant is less flexible in its possible configurations but achieves an audio quality comparable to the frequency-domain approach at a fraction of computational cost. This makes it in particular well suited for low-cost and sample-by-sample DSP implementations where no highly optimised FFT implementation is available or for low-delay applications.

Sound examples of both the frequency-domain and time-domain approach can be found on the website of the department[2].

6. REFERENCES

[1] Carlos Avendano and Jean-Marc Jot, "A frequency-domain approach to multichannel upmix," *Journal of the Audio Engineering Society*, vol. 52, no. 7, pp. 740–749, 2004.

[2] Christof Faller, "Multiple-loudspeaker playback of stereo signals," *Journal of the Audio Engineering Society*, vol. 54, no. 11, pp. 1051–1064, 2006.

[3] John Usher and Jacob Benesty, "Enhancement of Spatial Sound Quality: A New Reverberation-Extraction Audio Upmixer," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 2141–2150, sep 2007.

[4] Jianjun He, Ee-Leng Tan, and Woon-Seng Gan, "Linear Estimation Based Primary-Ambient Extraction for Stereo Audio Signals," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 2, pp. 505–517, feb 2014.

[5] Roger Dressler, "Dolby Surround Pro Logic II decoder principles of operation," *Dolby White paper*, 2000.

[6] Sebastian Kraft and Udo Zölzer, "Stereo signal separation and upmixing by mid-side decomposition in the frequency-domain," in *Proc. of the 18th Int. Conference on Digital Audio Effects*, 2015.

[7] Sebastian Kraft and Udo Zölzer, "Low-complexity stereo signal decomposition and source separation for application in stereo to 3D upmixing," in *Proc. of the 140th AES Convention*, 2016.

[8] Alexis Favrot and Christof Faller, "Complementary N-band IIR filterbank based on 2-band complementary filters," in *Proc. of the Intl. Workshop on Acoustic Echo and Noise Control (IWAENC)*, 2010.

[9] Alexander Jourjine, Scott Rickard, and Özgür Yilmaz, "Blind separation of disjoint orthogonal signals: demixing N sources from 2 mixtures," in *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing*, 2000.

[1] `http://www.fftw.org/`

[2] `http://ant.hsu-hh.de/upmix`

a) Frequency Domain

	Operations per block			
	ADD	MULT	SQRT	Overall ($2 \rightarrow C$)
STFT				$2 \cdot 5\ \log_2(N_F) \cdot N_F$
Inv. STFT				$C \cdot 5\ \log_2(N_F) \cdot N_F$
Pos. Estimation	$5 \cdot N_F$	$7 \cdot N_F$	$3 \cdot N_F$	$15 \cdot N_F$
Dir./Amb. Separation	$5 \cdot N_F$	$8 \cdot N_F$		$13 \cdot N_F$
Repanning	$(2\,C_1 + 4) \cdot N_F$	$(2\,C_1 + 9) \cdot N_F$	$2 \cdot N_F$	$(4\,C_1 + 15) \cdot N_F$
Overall per block				$\left[\ C \cdot 5\ \log_2(N_F)\ +\ 4\,C_1\ +\ 10\ \log_2(N_F) + 43\ \right] \cdot N_F$
Overall per sample				$\left[\ C \cdot 5\ \log_2(N_F)\ +\ 4\,C_1\ +\ 10\ \log_2(N_F) + 43\ \right] \cdot 2$
- Sample rate 44.1 kHz, $N_F = 2048$				13.5 MFlops $+$ $\left[C \cdot 4.85 + C_1 \cdot 0.35\right]$ MFlops
- Transforms + Upmix				$\left[(C+2) \cdot 4.85\right]$ MFlops $+$ $\left[3.8 + C_1 \cdot 0.35\right]$ MFlops
- $C = 5, C_1 = 3$				34.0 MFlops $+$ 4.85 MFlops

b) Time Domain

	Operations per sample and band			
	ADD	MULT	SQRT	Overall ($2 \rightarrow C$)
Filterbank				$2 \cdot \left[\ M \cdot (3\,N + 4)\ +\ M_A \cdot 3\,N_1\ \right]$
Power estimation	2	6		$8 \cdot (M-1)$
Synthesis	C			$C \cdot (M-1)$
Pos. Estimation	3	3	3	$9 \cdot (M-1)$
Dir./Amb. Separation	3	4		$7 \cdot (M-1)$
Repanning	$(C_1 + 4)$	$(C_1 + 9)$	2	$(2\,C_1 + 15) \cdot (M-1)$
Overall at sample rate 44.1 kHz, $C = 5, C_1 = 3$				
- FB I + Upmix				5.1 MFlops $+$ 3.3 MFlops
- FB II + Upmix				9.6 MFlops $+$ 6.5 MFlops
- FB III + Upmix				13.8 MFlops $+$ 6.5 MFlops

Table 4: *Required operations for an upmix from* 2 *to* C *channels where direct signal repanning is limited to a subset of* C_1 *loudspeakers (e.g. front loudspeakers only).*

[10] Gary S. Kendall, "The Decorrelation of Audio Signals and Its Impact on Spatial Imagery," *Computer Music Journal*, vol. 19, no. 4, pp. 71–87, 1995.

[11] Tapani Pihlajamäki, Olli Santala, and Ville Pulkki, "Synthesis of spatially extended virtual sources with time-frequency decomposition of mono signals," *Journal of the Audio Engineering Society*, vol. 62, no. 7-8, pp. 467–484, 2014.

[12] Marco Fink, Sebastian Kraft, and Udo Zölzer, "Downmix-compatibe conversion from mono to stereo in time- and frequency-domain," in *Proc. of the 18th Int. Conference on Digital Audio Effects*, 2015.

[13] Benjamin B. Bauer, "Phasor analysis of some stereophonic phenomena," *IRE Transactions on Audio*, vol. 10, no. 1, pp. 143–146, 1962.

[14] Alexis Favrot and Christof Faller, "Designing sets of N doubly complementary IIR filter," in *Proc. of the 130th AES Convention*, 2011.

[15] Sanjit K. Mitra, Yrjö Neuvo, and Palghat P. Vaidyanathan, "Complementary IIR digital filter banks," in *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1985.

[16] Palghat P. Vaidyanathan, Sanjit K. Mitra, and Yrjö Neuvo, "A new approach to the realization of low-sensitivity IIR digital filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 2, pp. 350–361, 1986.

[17] Lajos Gazsi, "Explicit formulas for lattice wave digital filters," *IEEE Transactions on Circuits and Systems*, vol. 32, no. 1, pp. 68–88, 1985.

[18] Ville Pulkki, "Virtual sound source positioning using vector base amplitude panning," *Journal of the Audio Engineering Society*, vol. 45, no. 6, pp. 456–466, 1997.

[19] Franz Zotter, Matthias Frank, Georgios Marentakis, and Alois Sontacchi, "Phantom source widening with deterministic frequency dependent time delays," in *Proc. of the 14th Int. Conference on Digital Audio Effects*, 2011.

ROUNDING CORNERS WITH BLAMP

Fabián Esqueda , Vesa Välimäki*

Dept. of Signal Processing and Acoustics
Aalto University
Espoo, Finland
fabian.esqueda@aalto.fi

Stefan Bilbao †

Acoustics and Audio Group
University of Edinburgh
Edinburgh, UK
s.bilbao@ed.ac.uk

ABSTRACT

The use of the bandlimited ramp (BLAMP) function as an antialiasing tool for audio signals with sharp corners is presented. Discontinuities in the waveform of a signal or its derivatives require infinite bandwidth and are major sources of aliasing in the digital domain. A polynomial correction function is modeled after the ideal BLAMP function. This correction function can be used to treat aliasing caused by sharp edges or corners which translate into discontinuities in the first derivative of a signal. Four examples of cases where these discontinuities appear are discussed: synthesis of triangular waveforms, hard clipping, and half-wave and full-wave rectification. Results obtained show that the BLAMP function is a more efficient tool for alias reduction than oversampling. The polynomial BLAMP can reduce the level of aliasing components by up to 50 dB and improve the overall signal-to-noise ratio by about 20 dB. The proposed method can be incorporated into virtual analog models of musical systems.

1. INTRODUCTION

Nonlinear audio processing introduces frequency components that are not present in the original input signal. When the frequencies of these components exceed half the sampling rate, or Nyquist limit, they are reflected into the baseband through aliasing [1, 2]. Aliasing distortion can cause audible disturbances, such as beating and inharmonicity, and affect the overall performance of an audio system [3, 1]. In fields such as virtual analog modeling of musical systems, the aim is to emulate the harmonic distortion introduced by analog systems while avoiding aliasing distortion [4, 2]. Therefore, it is of great importance to find efficient algorithms that minimize its effect.

A well known previous approach to avoiding aliasing in nonlinear audio processing is oversampling [4, 5, 6, 7]. In oversampling, the input signal is upsampled prior to processing (typically by a low factor) and downsampled back to the original rate after processing. This approach requires access to the original unprocessed signal and, ideally, some knowledge on the order of the nonlinear processing stage. In oversampling, the added computational costs will depend on the oversampling factor and order of the filters used for its implementation. Other techniques available to avoid aliasing in nonlinear processing include the harmonic mixer [8] and reducing the order of the nonlinearity [5]. The latter approach can also be used in distortion synthesis of classical oscillator waveforms [9].

Signal processing operations that introduce discontinuities in the waveform of a signal or its derivatives are major sources of aliasing. These discontinuities require infinite bandwidth to be represented in the digital domain. Attempting to sample them trivially will inevitably introduce aliasing distortion [10, 11]. When a discontinuity is introduced in the first derivative of a signal, a sharp edge or corner is introduced in the actual waveform.

Previous work on alias-reduced synthesis of oscillator waveforms has introduced the concept of quasi-bandlimiting discontinuities found in the waveform [12, 11, 13]. This work further explores this idea by presenting the use of the bandlimited ramp (BLAMP) function to treat any discontinuities found in the first derivative of a signal. This is achieved by quasi-bandlimiting the corners found in the waveform of a signal. The BLAMP function was originally proposed for synthesis of alias-free triangular waveforms [14, 11]. We derive a polynomial approximation of the BLAMP function, or polyBLAMP, which leads to an efficient implementation. Four examples of audio-specific scenarios where corners appear in the waveform of a signal are discussed: synthesis of triangular oscillator waveforms, hard clipping, half-wave and full-wave rectification. Results obtained demonstrate that the polyBLAMP method can effectively reduce the aliasing caused by these corners and the discontinuities they introduce.

This paper is organized as follows. Section 2 derives the analytical form of the BLAMP correction function. Section 3 discusses the computational costs of the BLAMP function and presents the derivation of its polynomial approximation. In Section 4, the performance of the method is evaluated by considering four applications. Finally, concluding remarks appear in Section 5.

2. INTEGRATED BANDLIMITED FUNCTIONS

Analog signals with discontinuities in their waveforms have infinite frequency content and must be bandlimited to less than half the Nyquist limit prior to sampling to avoid aliasing. The full-band nature of discontinuous signals can be observed, for instance, by considering the Fourier series (FS) expansion of a rectangular pulse. The FS for this signal consists of an infinite sum of odd sinusoidal components, with the amplitude of the k^{th} harmonic defined as $1/k$ of the first harmonic or fundamental.

We can model a single discontinuity in the continuous-time domain using the Heaviside unit step function, which is defined as

$$u(t) = \begin{cases} 0 & t < 0 \\ 1 & t \geq 0, \end{cases} \tag{1}$$

where t is time. This function jumps from 0 to 1 at $t = 0$ and is used in system analysis to measure the step response of a system.

In this work we are concerned with aliasing caused by discontinuities occurring not in the waveform of a signal, but in its first

* The work of F. Esqueda is funded by the Aalto ELEC Doctoral School.
† The work of S. Bilbao is supported by the European Research Council, under grant number ERC-StG-2011-279068-NESS.

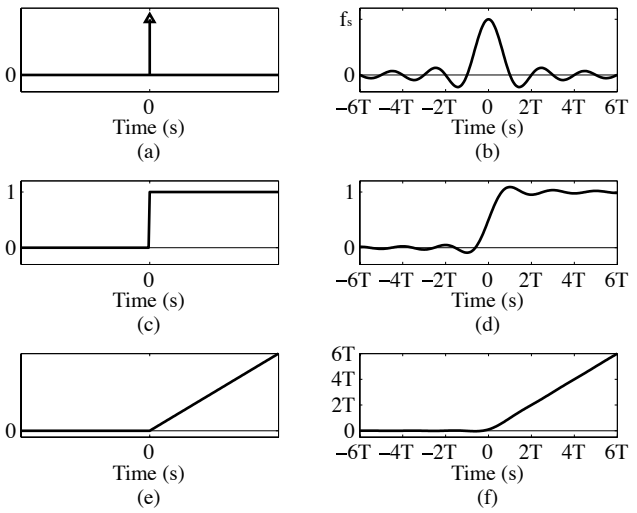


Figure 1: *Time-domain waveform of the (a) impulse, (b) bandlimited impulse, (c) Heaviside unit step, (d) BLEP, (e) ramp, and (f) BLAMP functions. Parameter T is the sampling period.*

derivative. Therefore, we model a discontinuity in the first derivative by evaluating the integral of the Heaviside function (1) as

$$\int_{-\infty}^{t} u(\tau)\mathrm{d}\tau = tu(t) = r(t). \tag{2}$$

Equation (2) is known in the literature as the ramp function [15]. It is characterized by the sharp corner that occurs at $t = 0$ when the function starts to linearly increase.

Signals with discontinuities in their first derivative also have infinite frequency content. One example of this is the triangular waveform, which can be represented using the FS as an infinite sum of odd sinusoidal components. In this series, harmonics decay at a steeper rate than in the case of the rectangular pulse, with the amplitude of the k^{th} harmonic given by $1/k^2$ with respect to the fundamental. In the digital domain, this steeper decay means that the level of aliasing introduced during trivial, non-bandlimited sampling of a sharp corner will be lower than that introduced in discontinuous signals. Nevertheless, when working at audio rates (e.g. 44.1 kHz) the effects of this aliasing may still be perceived, particularly at high fundamental frequencies. Section 4.1 further expands on the issue of aliasing in triangular waveforms.

In order to derive a correction function that can reduce the aliasing introduced by trivial sampling of sharp corners we need to take one step back and evaluate the derivative of the Heaviside unit step function with respect to time. This derivative is defined as the Dirac delta function [15], so that

$$\frac{\mathrm{d}u(t)}{\mathrm{d}t} = \delta(t). \tag{3}$$

Figures 1(a), 1(c) and 1(e) show the continuous-time domain waveforms for the Dirac delta (represented by a single impulse), Heaviside unit step, and ramp functions, respectively. From these waveforms it should become evident that the size of the discontinuity introduced in the first derivative by a sharp corner will depend on the slope of the signal at this corner.

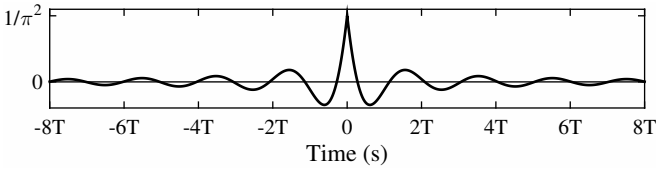


Figure 2: *BLAMP residual function is the difference of the BLAMP and trivial ramp functions. Cf. Figs. 1(f) and 1(e).*

The delta function has a flat unity spectrum, so its bandlimited form can then be obtained by evaluating the inverse Fourier Transform (FT) of an ideal brickwall lowpass filter [16], which yields

$$h^{(0)}(t) = f_\mathrm{s}\mathrm{sinc}(f_\mathrm{s}t), \tag{4}$$

where f_s represents the sampling rate and $\mathrm{sinc}(x) = \sin(\pi x)/\pi x$. Figure 1(b) shows the waveform for this expression.

Following our previous logic, we can derive a bandlimited expression for the ramp function (2) by integrating (4) twice. The integral of the bandlimited unit impulse yields the closed-form equation for the bandlimited step (BLEP) function [11], expressed as

$$h^{(1)}(t) = \frac{1}{2} + \frac{1}{\pi}\mathrm{Si}(\pi f_\mathrm{s}t), \tag{5}$$

where $\mathrm{Si}(x)$ is the sine integral, defined as $\mathrm{Si}(x) = \int_0^x \frac{\sin(t)}{t}\mathrm{d}t$. Previous work in the field of alias-free synthesis of rectangular and sawtooth oscillators has focused on using this expression to bandlimit the inherent discontinuities of these waveforms [17, 11, 18]. Figure 1(d) shows the shape for this function.

Moving on, (5) can be integrated once more using integration by parts, yielding

$$h^{(2)}(t) = t\left[\frac{1}{2} + \frac{1}{\pi}\mathrm{Si}(\pi f_\mathrm{s}t)\right] + \frac{\cos(\pi f_\mathrm{s}t)}{\pi^2 f_\mathrm{s}} \tag{6}$$

$$= th^{(1)}(t) + \frac{\cos(\pi f_\mathrm{s}t)}{\pi^2 f_\mathrm{s}}. \tag{7}$$

This equation gives the closed form expression for the BLAMP function with unit slope, and its shape is shown in Fig. 1(f). At first glance, Figs. 1(e) and 1(f) may appear indistinguishable. However, computing the difference between (7) and (2) quickly proves otherwise, as shown in Fig. 2. This function is referred to as the BLAMP residual function in this study.

In the discrete-time domain, the BLAMP residual can be used to reduce the aliasing caused by a discontinuity in the first derivative by adding it to every sharp edge in the waveform. The first step of this process involves centering the residual function at the exact points in time where the edges occur and sampling it at the nearest integer sample points. These sampled values must then be scaled by the magnitude and direction (i.e. rising or falling edge) of the discontinuity introduced in the first derivative of the signal. The magnitude parameter, as previously stated, can be computed from the slope of the signal at the edge.

3. POLYNOMIAL BLAMP APPROXIMATION

The analytic expression for the BLAMP residual function has two limitations. First, its implementation is computationally expensive due to the presence of the sine integral function. Secondly, the

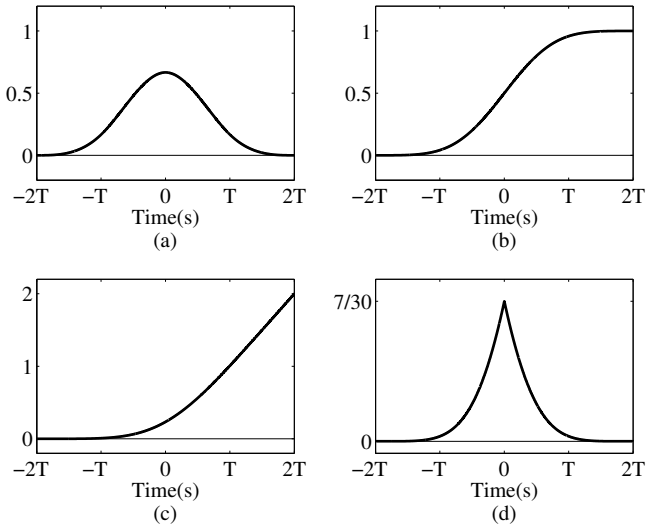


Figure 3: *(a) Cubic B-spline basis function, (b) its first integral, (c) its second integral or polyBLAMP approximation and (d) the polyBLAMP residual, i.e., the difference between the polyBLAMP and the trivial ramp functions.*

Table 1: Third-order B-spline basis functions, its first integral (polyBLEP), its second integral (polyBLAMP), and polyBLAMP residual ($1 \leq D < 2$ and $0 \leq d < 1$) [20].

Span	Third-order B-spline basis function
$[-2T, -T]$	$D^3/6 - D^2/2 + D/2 - 1/6$
$[-T, 0]$	$-D^3/2 + 2D^2 - 2D + 2/3$
$[0, T]$	$D^3/2 - 5D^2/2 + 7D/2 - 5/6$
$[T, 2T]$	$-D^3/6 + D^2 - 2D + 4/3$
Span	First integral: Four-point polyBLEP
$[-2T, -T]$	$D^4/24 - D^3/6 + D^2/4 - D/6 + 1/24$
$[-T, 0]$	$-D^4/8 + 2D^3/3 - D^2 + 2D/3 - 1/6$
$[0, T]$	$D^4/8 - 5D^3/6 + 7D^2/4 - 5D/6 + 7/24$
$[T, 2T]$	$-D^4/24 + D^3/3 - D^2 + 4D/3 + 1/3$
Span	Second integral: Four-point polyBLAMP
$[-2T, -T]$	$D^5/120 - D^4/24 + D^3/12 - D^2/12 + D/24 - 1/120$
$[-T, 0]$	$-D^5/40 + D^4/6 - D^3/3 + 2D^2/3 - D/6 + 1/30$
$[0, T]$	$D^5/40 - 5D^4/24 + 7D^3/12 - 5D^2/12 + 7D/24 - 1/24$
$[T, 2T]$	$-D^5/120 + D^4/12 - D^3/3 + 2D^2/3 + D/3 + 4/15$
Span	Four-point polyBLAMP residual
$[-2T, T]$	$d^5/120$
$[-T, 0]$	$-d^5/40 + d^4/24 + d^3/12 + d^2/12 + d/24 + 1/120$
$[0, T]$	$d^5/40 - d^4/12 + d^2/3 - d/2 + 7/30$
$[T, 2T]$	$-d^5/120 + d^4/24 - d^3/12 + d^2/12 - d/24 + 1/120$

function does not have finite support, it does not vanish. Therefore, its truncation to a finite interval introduces small discontinuities which will produce further aliasing. Both issues can be addressed by storing a windowed precomputed portion of the function in a lookup table. This approach is sometimes used in practical implementations of the BLEP method [17, 12, 18]. In general, the effectiveness and efficiency of a table-based implementation will depend on several variables, including table size, interpolation method used (if any) and type of window.

In this work, we instead propose the use of a B-spline polynomial approximation of the BLAMP function (the polyBLAMP) which can be implemented with minimal computational costs. This polyBLAMP function can correct four samples, two on each side of every sharp edge in the waveform. The four-point polyBLAMP function is derived by first approximating the bandlimited impulse (3) as a piecewise polynomial using the coefficients for the third-order B-spline basis function and following the same steps detailed in the previous section [i.e. integrate twice and subtract (2)]. B-spline interpolating polynomials have been used in this study due to their steep spectral decay which makes them suitable for antialiasing applications [19, 11].

Before moving on to the derivation of the polyBLAMP, we first consider that, in practice, the exact sample points at which the sharp edges occur in a signal (i.e. the points where the derivative of the signal is discontinuous) will most likely not coincide with the sampling intervals of the system and must be estimated. In the four-point case, the process of centering the correction function around a set of four samples can be seen as equivalent to delaying it by $D = D_{\text{int}} + d$ samples, where $D_{\text{int}} = 1$, and $d \in [0, 1)$ is the fractional delay.

The coefficients for the B-spline basis function can be expressed in terms of delay D using the four polynomials shown at the top of Table 1. These polynomial coefficients are derived via the iterative convolution of a rectangular pulse, and the resulting waveform can be seen in Fig. 3(a). From this figure, that loosely resembles the central lobe of the bandlimited impulse [see Fig. 1(b)], we can observe the characteristic bell-shaped curve of B-spline interpolators. Integrating this basis function once yields the B-spline polynomial form of the BLEP function (known as the polyBLEP [12, 11]), and integrating once more results in the four-point polyBLAMP function [20]. The polynomials for these two functions and their corresponding waveforms are shown in Table 1, and Figs. 3(b) and 3(c), respectively. Finally, the bottom four rows of Table 1 show the piecewise polynomial coefficients for the polyBLAMP residual evaluated by substituting $D = d + 1$ and computing difference between the polyBLAMP and the ramp function. A two-point version of the polyBLAMP function can be found in [21]. However, due to its superior performance, this work focuses solely on the four-point method.

Expressing the four-point polyBLAMP residual function in terms of the fractional delay d required to center it around a sharp edge simplifies the procedure of sampling it at the four neighboring sample points. Therefore, parameter d must be estimated to a certain degree of accuracy. First, we consider $s[n]$ to be the discrete-time signal to be antialiased, where $n \in \mathbb{Z}_{\geq 0}$ is the sample index. Next, we define n_{a} and n_{b} as the sample indices of the signal before and after an edge, i.e. the *corner boundaries*. For every edge in the waveform, sample points $n_{\text{a}} - 1$, n_{a}, n_{b} and $n_{\text{b}} + 1$ will be processed by the algorithm. The aim is to fit a polynomial of the form $f(D) = aD^3 + bD^2 + cD + e$ to the signal $s[n]$ at these four points. Lagrange interpolation can be used to find the closed form expressions for coefficients a, b, c, and e. Since the data points are evenly spaced, these coefficients can be written as

$$\begin{aligned} a &= -\tfrac{1}{6}s[n_{\text{a}} - 1] + \tfrac{1}{2}s[n_{\text{a}}] - \tfrac{1}{2}s[n_{\text{b}}] + \tfrac{1}{6}s[n_{\text{b}} + 1] \\ b &= s[n_{\text{a}} - 1] - \tfrac{5}{2}s[n_{\text{a}}] + 2s[n_{\text{b}}] - \tfrac{1}{2}s[n_{\text{b}} + 1] \\ c &= -\tfrac{11}{6}s[n_{\text{a}} - 1] + 3s[n_{\text{a}}] - \tfrac{3}{2}s[n_{\text{b}}] + \tfrac{1}{3}s[n_{\text{b}} + 1] \\ e &= s[n_{\text{a}} - 1]. \end{aligned} \tag{8}$$

After fitting the polynomial, the next step is to obtain the inter-

section of this curve with ρ, the corner parameter. The value of ρ will depend on the particular application. For instance, for corners caused by rectification we need to find the zero-crossings of the polynomial, thus $\rho = 0$. Further details on how this parameter is adjusted for each application are given in Sec. 4. This inverse interpolation problem is equivalent to solving the following equation for D:

$$aD^3 + bD^2 + cD + e - \rho = 0. \tag{9}$$

A solution can be estimated using Newton-Raphson's (NR) iterative method [20], defined as

$$D_{q+1} = D_q - \frac{f(D_q)}{f'(D_q)}, \tag{10}$$

where $q = 0, 1, 2, ..., Q-1$, and Q is the number of iterations required for the ratio $f(D_q)/f'(D_q)$ to become small enough to be neglected, and D_0 is an initial guess [22]. Since the solution to (9) will range between [1,2) due to the restriction on D, an appropriate initial guess would be $D_0 = 1.5$.

We can then estimate the point where the discontinuity in the first derivative occurs as

$$D_{q+1} = D_q - \frac{aD_q^3 + bD_q^2 + cD_q + e - \rho}{3aD_q^2 + 2bD_q + c}. \tag{11}$$

The resulting value D_Q represents the fractional delay associated with a sharp edge or corner. The slope at this point is obtained as a byproduct of the NR method, which is given as

$$\mu(D_Q) = 3aD_Q^2 + 2bD_Q + c. \tag{12}$$

Finally, the value of d can be computed as $d = D_Q - 1$. This represents an estimated sharp edge at $n_a + d$, i.e. $s[n_a + d] = \rho$.

4. POLYBLAMP APPLICATIONS

This section shows how the polyBLAMP correction method can be applied for antialiasing in four audio applications where discontinuities appear in the first derivative of the signal waveform.

4.1. Alias-Free Triangular Oscillator

The first application considered in this study is the synthesis of antialiased triangular oscillator waveforms. This type of geometric waveform is commonly used as a source signal in subtractive synthesis due to its rich harmonic content. As mentioned in Sec. 2, the triangular waveform is composed of odd harmonics only and has the perceptual attribute of being *smoother* to the ears than sawtooth and rectangular waveforms. This characteristic can be attributed to the steep spectral decay of its harmonics, which decay at a rate of about -12 dB per octave (the spectrum of sawtooth and rectangular waveforms decays at a rate of about -6 dB per octave) [3]. This steep spectral decay rate is associated with the discontinuity in its first derivative [10].

Fig. 4(a) shows the continuous-time domain waveform for four periods of a triangular oscillator with fundamental frequency f_0 and period $T_0 = 1/f_0$. Computing the first derivative of this signal results in the square signal shown in Fig. 4(b). The peak-to-peak amplitude of this resulting waveform is determined by 2μ, where μ is the absolute value of the slope of the rising and falling portions of the signal. Since the slope of the falling section is the negative of the slope of the rising section, the signal in Fig. 4(a) is

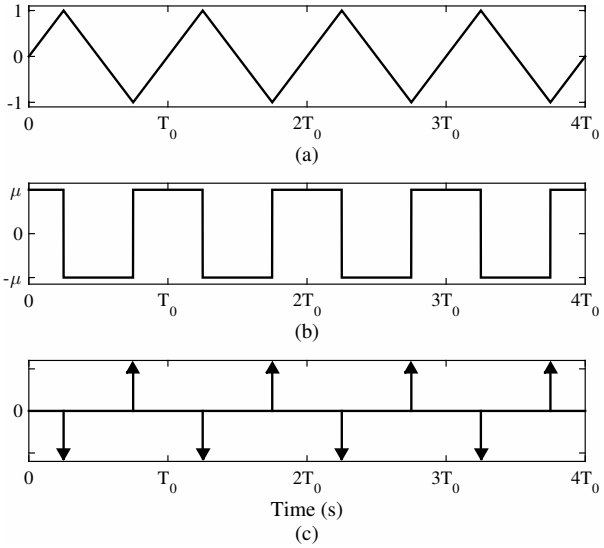


Figure 4: *(a) Continuous-time triangular waveform with arbitrary fundamental frequency f_0, (b) its first and (c) second derivatives.*

formally known as the symmetrical triangular waveform. Finally, evaluating the derivative of Fig. 4(b) yields the alternating impulse train shown in Fig. 4(c).

In theory, an alias-free discrete-time implementation of the triangular waveform can be achieved by replacing the impulses seen in Fig. 4(c) with (4) (note that the polarity of every second pulse has to be inverted) and integrating the function twice [16]. Due to the infinite nature of the bandlimited impulse (4) and the difficulties associated with performing the double integration, this approach is impractical. Instead, we propose adding the four-point polyBLAMP residual function to the actual waveform at the exact points where the impulses would appear in the second derivative, i.e. at the corners. The residual function has to be scaled by 2μ and inverted for positive edges of the waveform [see Fig. 4(c)].

Since this is a synthesis application of the polyBLAMP method, there is no need to estimate the fractional points at which the edges occur or the slope of the signal at those points; these two parameters are readily available. To implement the proposed method we first synthesize a trivial triangular waveform using a bipolar modulo counter $\phi[n]$ that switches its direction every time it reaches $+1$ or -1 [23]. The fractional delay d associated with each corner can be computed every time the polarity of the counter is inverted as

$$d = (T_\phi - \phi[n])/T_\phi, \tag{13}$$

where $T_\phi = 2f_0/f_s$ is the phase step size. The slope parameter is given by $|\mu| = 2T_\phi$.

Fig. 5 shows the waveform and spectrum for a 1661-Hz (MIDI note A6) trivial triangular waveform sampled at 44.1 kHz without and with four-point polyBLAMP correction. These results show that the corrected signal is virtually alias-free below approx. 12 kHz. Due to the inherent steep spectral decay of B-spline polynomials, the polyBLAMP method introduces a frequency droop of approx. -12 dB. This droop begins after the 10 kHz mark and, if necessary, can be compensated using a shelving EQ filter [11]. However, due to it only affecting high frequencies, it can be neglected in most applications. One convenient property of the B-

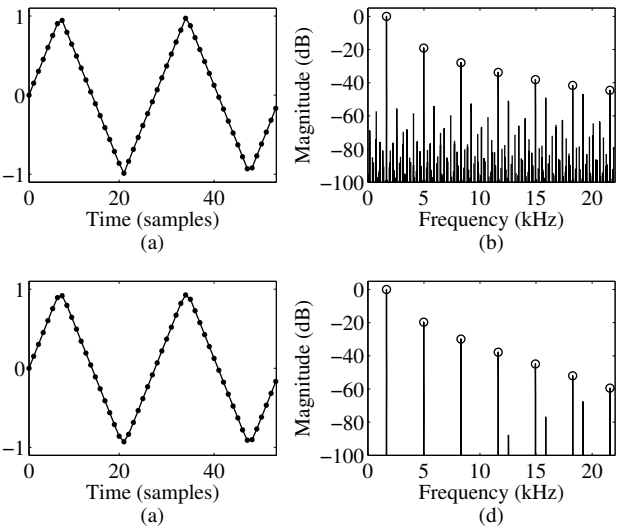


Figure 5: *Waveform and magnitude spectra of a (a)-(b) 1661.2-Hz trivial triangle wave, and (c)-(d) the same signal after four-point polyBLAMP correction. Circles indicate non-aliased components.*

spline polyBLAMP method is that it preserves the original range of signal values, as the correction is performed "inwards", so to speak.

Several methods to synthesize triangular waveforms with reduced aliasing have been proposed. Stilson et al. initially suggested double integration of a bipolar bandlimited impulse train (BLIT) [16, 19]. Välimäki et al. developed a more efficient approach using a differentiated parabolic waveform (DPW) [24]. This approach was later optimized for synthesis of triangle waveforms by Ambrits and Bank [13] using efficient polynomial transition regions (EPTR). The EPTR method was used as a reference to evaluate the performance of the proposed polyBLAMP method.

The signal-to-noise ratio (SNR) of the A6 triangular waveform was measured with and without polyBLAMP correction. In this context, SNR was defined as the power ratio between harmonics and aliasing components. To show the limits of the proposed method, a second measurement was performed on a 4168-Hz (MIDI note C8) signal. This frequency represents the highest fundamental frequency on a piano. For further evaluation, the SNRs obtained using oversampling by factors 2 and 4 were also computed. The top two rows of Table 2 show the results obtained from these measurements. The polyBLAMP method exhibits results comparable to oversampling by 4 at a fraction of the computational costs. Additionally, the resulting SNRs for the EPTR algorithm were 54 dB and 43 dB for the A6 and C8 signals, respectively.

In terms of computational costs, the top row of Table 3 shows the average synthesis times for a 1-second C8 triangular signal using oversampling by 2 and 4, the EPTR method and the four-point polyBLAMP. These results were obtained by porting the algorithm into Python and using the `time` function. The polyBLAMP method yielded the fastest processing times.

Table 2: SNR measurements in dB for test signals of 1661 Hz (A6) and 4186 Hz (A8). The best SNR on each row is bolded.

Signal	Triv.	OS		polyBLAMP
		by 2	by 4	
Triangular A6	42 dB	52 dB	56 dB	**54 dB**
Triangular C8	30 dB	42 dB	46 dB	**45 dB**
Clipping A6	34 dB	42 dB	43 dB	**57 dB**
Clipping C8	24 dB	34 dB	38 dB	**42 dB**
Half-W. Rec. A6	40 dB	43 dB	44 dB	**61 dB**
Half-W. Rec. C8	28 dB	36 dB	38 dB	**48 dB**
Full-W. Rec. A6	32 dB	40 dB	41 dB	**53 dB**
Full-W. Rec. C8	20 dB	28 dB	30 dB	**39 dB**

4.2. Alias-Free Hard Clipping

Hard clipping is another example of an audio application where discontinuities in the first derivative of a signal are introduced [20]. Signal clipping is a form of distortion that limits the values of a signal that lie above or below a predetermined threshold. Symmetric hard clipping can be expressed as

$$f_c(x[n]) = \text{sgn}(x[n])\text{min}(|x[n]|, L), \tag{14}$$

where $x[n]$ is the input signal, $\text{sgn}(\cdot)$ is the sign function, and $L \in (0, 1]$ is the normalized clipping threshold. In practice, signal clipping may be necessary due to system limitations, e.g. to avoid overmodulating an audio transmitter. In discrete systems, it can be caused unintentionally due to data resolution constraints, or intentionally as when simulating an analog system in which signal values are saturated [25].

Fig. 6(a) shows the continuous-time clipped f_0-Hz sinusoid with clipping threshold $L = 0.7$ (solid line) together with the original sine wave (dashed line). Following the same approach as in the previous subsection, we evaluate the first derivative of this signal and observe that this derivative presents discontinuities at the exact points in time where it enters or leaves a saturation [see Fig. 6(b)]. Further derivation of this signal yields the waveform shown in Fig. 6(c), which contains impulses whose polarities depend on the direction of the observed discontinuities.

Implementation of the four-point polyBLAMP correction on an arbitrary input signal requires a polynomial to be fit to the four corner boundaries as described in Sec. 3. Then, the NR method can be used to estimate parameters d and μ by substituting $\rho = \pm L$ in (11). The polarity of ρ will depend on the polarity of the clipping point being corrected, as shown in Fig. 6(c). Fig. 7 shows the

Table 3: *Averaged computation time (in ms) for oversampling by factors of 2 and 4, the EPTR, and 4-point polyBLAMP methods.*

	Oversampling		EPTR	polyBLAMP
Signal	by 2	by 4		
Triangular C8	36 ms	72 ms	45 ms	**18 ms**
Clipping C8	46 ms	102 ms	-	**43 ms**
Half-W. Rect. C8	40 ms	89 ms	-	**18 ms**
Full-W. Rect. C8	41 ms	90 ms	-	**28 ms**

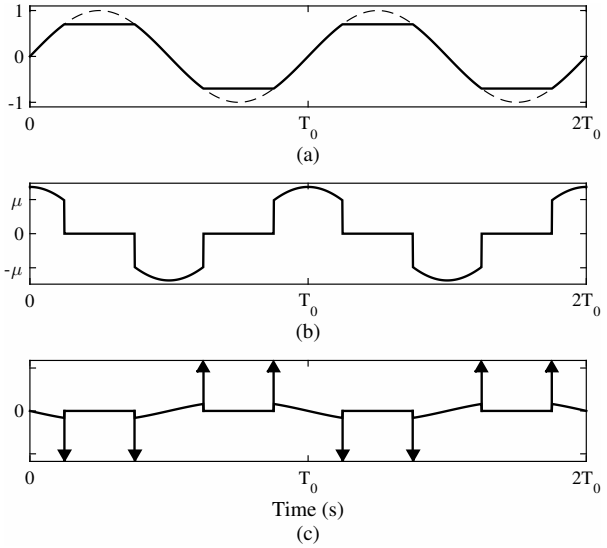


Figure 6: *(a) Continuous-time f_0-Hz sinusoid hard-clipped with clipping threshold $L = 0.8$, (b) its first and (c) second derivatives.*

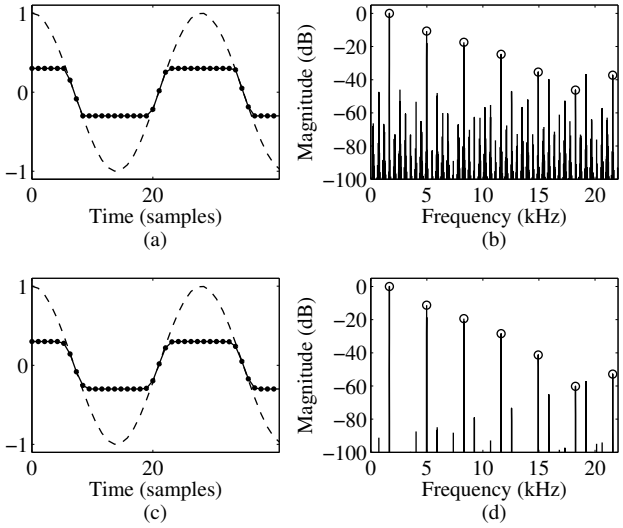


Figure 7: *Waveform and magnitude spectra of a (a)-(b) 1660-Hz trivial hard-clipped sine wave ($L = 0.3$), and (c)-(d) the same signal after four-point polyBLAMP correction.*

waveform and magnitude spectra for a 1661-Hz sinusoid clipped at $L = 0.3$ before and after four-point polyBLAMP correction. Once again, the corrected signal exhibits improved performance in terms of aliasing. For instance, the level of the most prominent aliasing component below the fundamental, at 720 Hz has been attenuated by 43 dB. As before, the clipping threshold is preserved after the correction, since the polyBLAMP method does not introduce any overshoot in the time domain.

Rows 3 and 4 of Table 2 show the SNRs measured for two sinusoidal signals (MIDI notes A6 and C8) trivially-clipped, using oversampling by factors 2 and 4, and after four-point polyBLAMP correction. All these measurements were performed using a clipping threshold $L = 0.3$. In this application, the four-point polyBLAMP method also exhibits better performance than oversampling by low factors, with SNR improvements of 22.6 and 17.4 dB for each respective signal. In terms of computational costs, the polyBLAMP method shows similar costs to those of oversampling by factor 2 but with improved SNR (see Tables 2 and 3).

As a final note on hard clipping, Fig. 6(c) shows that the second derivative of the signal has discontinuities around each impulse. These discontinuities, while small, will contribute to the overall aliasing seen at the output of the clipper. Integrating the BLAMP or polyBLAMP function should, in theory, yield a correction function that further reduces aliasing. This idea is not explored any further in this study and is left as future work.

4.3. Alias-Free Half-Wave Rectification

Signal rectification is a type of memoryless nonlinear processing that can be used to introduce harmonic distortion. In a half-wave rectifier, only positive portions of the waveform are kept, while negative portions are set to zero

$$f_r(x[n]) = \max(x[n], 0). \tag{15}$$

In analog applications, this can be achieved using a diode, which only allows current to flow in one direction. A particular feature of half-wave rectification is that it introduces even harmonics only.

Figs. 8(a) and 8(b) show the continuous-time domain waveform for a half-wave rectified sine wave and its first derivative, respectively. As expected, the corners introduced by the rectifier translate into discontinuities in the derivative. The magnitude of each discontinuity is determined by the slope μ of the original signal at the zero-crossings. Derivating this signal once more yields the positive impulse train depicted in Fig. 8(c).

The polyBLAMP method can be used to round the corners seen in Fig. 8(a) by centering it at the zero crossings. Parameters d and μ can be estimated by replacing $\rho = 0$ in (9) and (11). This is equivalent to finding the zero-crossings of the input signal. Fig. 9 shows the waveforms and magnitude spectra for a 1660-Hz sinusoid without and with polyBLAMP correction. In this example, the removal of the spurious frequency components is evident, with aliases below the fundamental are attenuated by more than 40 dB. Rows 5 and 6 of Table 2 show the proposed method outperforms oversampling by 2 and 4, and increases SNR by more than 20 dB with respect to a trivial implementation of half-wave rectification. In terms of computational costs, this implementation is cheaper than oversampling by a factor 2, as shown in Table 3.

4.4. Alias-Free Full-Wave Rectification

In full-wave rectification, negative portions of the waveform are not zeroed, but inverted, for example by taking the absolute value:

$$f_R(x[n]) = |x[n]|\,. \tag{16}$$

This process introduces both even and odd harmonics. Several analog audio effects incorporate a full-wave rectifier as part of a larger signal processing chain, such as the Octavio Fuzz pedal [26]. It can also be found as a stand-alone effect in modular synthesizer units, e.g. the Malekko 8NU8R [27].

Fig. 10(a) shows the continuous-time waveform for a rectified sine wave. Figs. 10(b) and 10(c) show the first and second derivatives of this waveform, respectively. In this case, the magnitude of the discontinuities that appear in the first derivative are defined

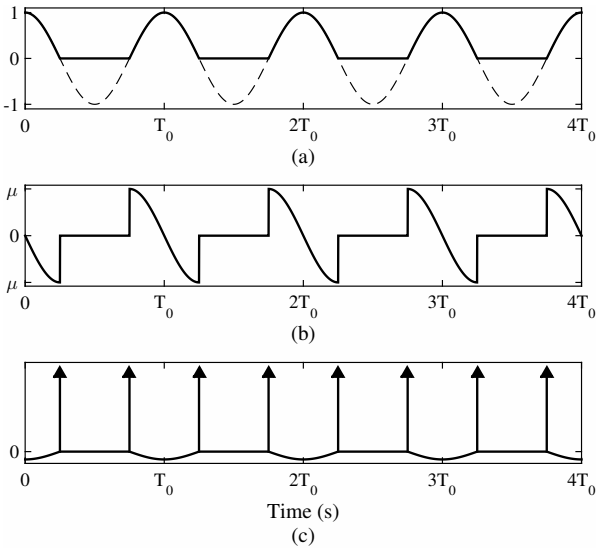


Figure 8: *(a) Half-wave rectified continuous-time sine wave, (b) its first and (c) second derivatives.*

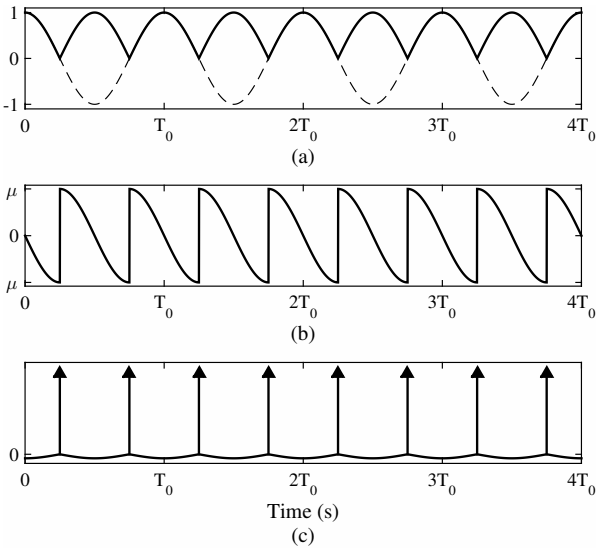


Figure 10: *(a) Full-wave rectified continuous-time sine wave, (b) its first and (c) second derivatives.*

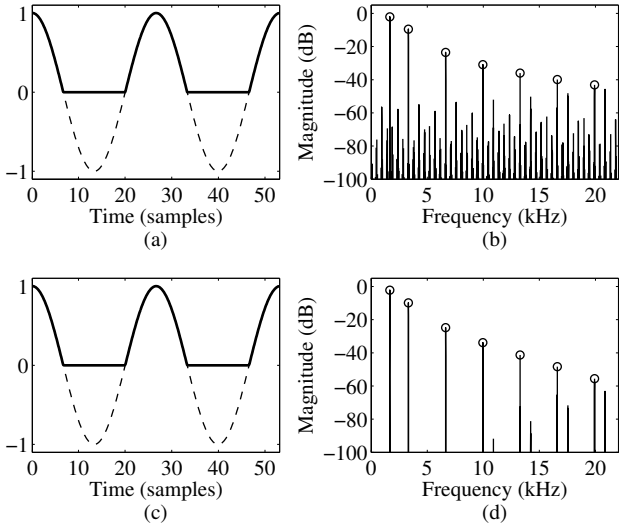


Figure 9: *Waveform and magnitude spectrum of a (a)-(b) 1660-Hz trivial half-wave rectified sine wave, and (c)-(d) the same signal after four-point polyBLAMP correction.*

as twice the slope of the original signal at the zero-crossings. As with hard clipping, both types of rectification also introduce discontinuities in subsequent derivatives of the signal, hinting at the possibility that further correction could be achieved using higher-order bandlimited integral functions.

The polyBLAMP method can be used in the same fashion as with half-wave rectification by scaling the slope parameter by factor 2. The bottom two rows of Table 2 show the measured SNRs for the two sinusoidal test signals discussed in the previous subsections. Once again, the polyBLAMP method outperforms oversampling by factors 2 and 4, offering a nearly 20-dB improvement in SNR with reduced computational costs (cf. Table 3).

Lane et al. [28] have proposed to use a full-wave rectified sine wave (after further linear filtering) to approximate the sawtooth waveform. Välimäki and Huovilainen analyzed this approximation showing that, while it contains considerably less aliasing than the trivial sawtooth, the aliasing can be still be audible at high fundamental frequencies [12]. The polyBLAMP method could now be used to further enhance this sawtooth generation method.

Additionally, to demonstrate that the BLAMP method is applicable to nonlinearly processed arbitrary signals and not just sine waves, Fig. 11 shows the waveform and magnitude spectrum for a synthetic string sound recording before and after full-wave rectification without and with polyBLAMP correction. Overall, aliasing components have been reduced by nearly 20 dB on average.

5. CONCLUSIONS

The corner-rounding capabilities of the polynomial approximation of the BLAMP function, or polyBLAMP, were studied. In addition to alias-free synthesis of triangular waveforms, it can enhance certain nonlinear waveshaping methods, which introduce discontinuities in the first derivative of the signal waveform. The fractional delay and slope at each corner need to be estimated, and then this method can correct a few samples in the neighborhood of each corner. The polyBLAMP method helps implementing alias-free versions of hard-clipping and rectification for arbitrary signals without oversampling, and thus enables enhanced nonlinear audio effects processing.

Supplementary material, including MATLAB code and sound examples, can be found in `http://research.spa.aalto.fi/publications/papers/dafx16-blamp/`.

6. REFERENCES

[1] T. S. Stilson, *Efficiently-Variable Non-Oversampled Algorithms in Virtual-Analog Music Synthesis*, PhD thesis, Stanford University, Stanford, CA, USA, June 2006.

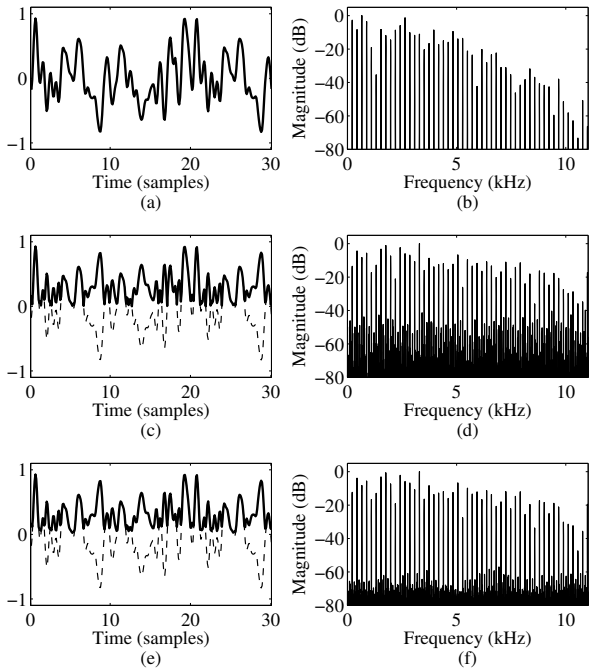


Figure 11: *Waveform and spectrum for a (a)-(b) synthetic string recording and the same signal after full-wave rectification (c)-(d) before and (e)-(f) after four-point polyBLAMP correction.*

[2] D. T. Yeh, *Digital Implementation of Musical Distortion Circuits by Analysis and Simulation*, PhD thesis, Stanford University, Stanford, CA, USA, June 2009.

[3] V. Välimäki and A. Huovilainen, "Oscillator and filter algorithms for virtual analog synthesis," *Computer Music J.*, vol. 30, no. 2, pp. 19–31, 2006.

[4] J. Pakarinen and D. T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music J.*, vol. 33, no. 2, pp. 85–100, 2009.

[5] H. Thornburg, "Antialiasing for nonlinearities: Acoustic modeling and synthesis applications," in *Proc. Int. Comput. Music Conf.*, Beijing, China, Oct. 1999, pp. 66–69.

[6] D. Mapes-Riordan, "A worst-case analysis for analog-quality (alias-free) digital dynamics processing," *J. Audio Eng. Soc.*, vol. 47, no. 11, pp. 948–952, Nov. 1999.

[7] J. Schimmel, "Audible aliasing distortion in digital audio synthesis," *Radioengineering*, vol. 21, no. 1, pp. 57, 2012.

[8] J. Schattschneider and U. Zölzer, "Discrete-time models for nonlinear audio systems," in *Proc. Digital Audio Effects Workshop*, Trondheim, Norway, Dec. 1999, pp. 45–48.

[9] V. Lazzarini and J. Timoney, "New perspectives on distortion synthesis for virtual analog oscillators," *Computer Music J.*, vol. 34, no. 1, pp. 28–40, 2010.

[10] P. Kraght, "Aliasing in digital clippers and compressors," *J. Audio Eng. Soc.*, vol. 48, no. 11, pp. 1060–1064, Nov. 2000.

[11] V. Välimäki, J. Pekonen, and J. Nam, "Perceptually informed synthesis of bandlimited classical waveforms using integrated polynomial interpolation," *J. Acoust. Soc. Am.*, vol. 131, no. 1, pp. 974–986, Jan. 2012.

[12] V. Välimäki and A. Huovilainen, "Antialiasing oscillators in subtractive synthesis," *IEEE Signal Process. Mag.*, vol. 24, no. 2, pp. 116–125, Mar. 2007.

[13] D. Ambrits and B. Bank, "Improved polynomial transition regions algorithm for alias-suppressed signal synthesis," in *Proc. 10th Sound and Music Computing Conf. (SMC2013)*, Stockholm, Sweden, Aug. 2013, pp. 561–568.

[14] A. Huovilainen, "Design of a scalable polyphony-MIDI synthesizer for a low cost DSP," M.S. thesis, Aalto University, Espoo, Finland, 2010.

[15] R. N. Bracewell, *The Fourier Transform and its Applications*, McGraw-Hill, 2nd edition, 1986.

[16] T. Stilson and J. Smith, "Alias-free digital synthesis of classic analog waveforms," in *Proc. Int. Computer Music Conf.*, Hong Kong, 1996, pp. 332–335.

[17] E. Brandt, "Hard sync without aliasing," in *Proc. Int. Computer Music Conf.*, Havana, Cuba, Sep. 2001, pp. 365–368.

[18] W. Pirkle, *Designing Audio Effect Plug-Ins in C++*, Focal Press, 1st edition, Oct. 2013.

[19] J. Nam, V. Välimäki, J. S. Abel, and J. O. Smith, "Efficient antialiasing oscillator algorithms using low-order fractional delay filters," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 773–785, May 2010.

[20] F. Esqueda, S. Bilbao, and V. Välimäki, "Aliasing reduction in clipped signals," *IEEE Trans. Signal Process.*, 2016, accepted for publication.

[21] F. Esqueda, V. Välimäki, and S. Bilbao, "Aliasing reduction in soft-clipping algorithms," in *Proc. European Signal Process. Conf.*, Nice, France, Aug. 2015, pp. 2059–2063.

[22] F. B. Hildebrand, *Introduction to Numerical Analysis*, Dover Publications, 2nd edition, Jun. 1987.

[23] J. Kleimola and V. Välimäki, "Reducing aliasing from synthetic audio signals using polynomial transition regions," *IEEE Signal Process. Lett.*, vol. 19, no. 2, pp. 67–70, Feb. 2012.

[24] V. Välimäki, J. Nam, J. O. Smith, and J. S. Abel, "Alias-suppressed oscillators based on differentiated polynomial waveforms," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 786–798, May 2010.

[25] D. Rossum, "Making digital filters sound analog," in *Proc. Int. Computer Music Conf.*, San Jose, CA, Oct. 1992, pp. 30–33.

[26] Dunlop Manufacturing, Inc., "Jimi Hendrix Octavio Pedal," Product information available at `http://www.jimdunlop.com/product/jhoc1-jimi-hendrix-octavio-effect`, accessed March 19, 2015.

[27] Malekko Heavy Industry Corporation, "8NU8R Dual Analog Attenuator Module," Product information available at `https://malekkoheavyindustry.com/product/8nu8r/`, accessed March 19, 2015.

[28] J. Lane, D. Hoory, E. Martinez, and P. Wang, "Modeling analog synthesis with DSPs," *Computer Music J.*, vol. 21, no. 4, pp. 23–41, 1997.

SYNTHESIS OF SOUND TEXTURES WITH TONAL COMPONENTS USING SUMMARY STATISTICS AND ALL-POLE RESIDUAL MODELING

Hyung-Suk Kim and Julius Smith

Center for Computer Research in Music and Acoustics (CCRMA)
Stanford University, USA
{hskim08|jos}@ccrma.stanford.edu

ABSTRACT

The synthesis of sound textures, such as flowing water, crackling fire, an applauding crowd, is impeded by the lack of a quantitative definition. McDermott and Simoncelli proposed a perceptual source-filter model using summary statistics to create compelling synthesis results for non-tonal sound textures. However, the proposed method does not work well with tonal components. Comparing the residuals of tonal sound textures and non-tonal sound textures, we show the importance of residual modeling. We then propose a method using auto regressive modeling to reduce the amount of data needed for resynthesis and delineate a modified method for analyzing and synthesizing both tonal and non-tonal sound textures. Through user evaluation, we find that modeling the residuals increases the realism of tonal sound textures. The results suggest that the spectral content of the residuals has an important role in sound texture synthesis, filling the gap between filtered noise and sound textures as defined by McDermott and Simoncelli. Our proposed method opens possibilities of applying sound texture analysis to musical sounds such as rapidly bowed violins.

1. INTRODUCTION

Sound *textures* are signals that have more structure than filtered noise, but, like visual textures, not all of the details are perceived by the auditory system. Saint-Arnaud [1] gives a qualitative definition of sound textures in terms of having constant long term characteristics that, unlike music or speech, do not carry a message which can be decoded. Figure 1 illustrates the relative information-bearing potential of music, speech, sound textures, and noise, showing how sound textures lie between music/speech and noise. Examples of sound textures include natural sounds such as water flowing, leaves rustling, fire crackling, or man-made sounds like the sound of people babbling, a crowd applauding or sounds of machinery such as drills. There can be textural components in musical sounds such as fast violin-bowing or guitar-string scraping.

A better understanding of sound textures can provide insights into our auditory process, and what information we extract from auditory inputs. Furthermore, such knowledge can be used to find sparse representations and applied to analysis/synthesis of environmental sounds, sound texture identification, data compression, and gap-filling.

Since Saint-Arnaud's work on sound texture, there has been a gradual increase of interest in this area and various approaches have been explored [2, 3]. One approach that has been extensively used is granular synthesis [4, 5, 6, 7, 8, 9]. In most cases, the general approach is to parse the audio during analysis, usually into sound events and background din, and then recompose the components according to a stochastic rule. The advantage of these approaches is that the original source is used for resynthesis, resulting in output quality as good as the source. This also means, however, that the method is bound by the source signal and that the methods may not be generalizable. Other approaches include applying various metrics and theories such as polyspectra [10], wavelets [11], dynamic systems [12] and scattering moments [13] to analyze sound textures.

Another approach is that of source-filter modeling. One source-filter approach is time-frequency linear predictive coding (TFLPC) also called cascade time-frequency linear prediction (CTFLP) [14, 15]. In TFLPC, time domain linear prediction, which captures the spectral content, is followed by frequency domain linear prediction, which models the temporal envelope of the residuals.

McDermott and Simoncelli [16] propose a source-filter approach using perceptual multiband decomposition, looking at the long term statistics of the multiband signal and its modulations. To evaluate the proposed model, the extracted statistics are imposed onto subband envelopes using an iterative method. The subband envelopes are multiplied with a noise signal to create the synthesized signal. An advantage of this approach is that there are no assumptions regarding the nature of the sound source, as it models how the auditory system processes the sound.

A limitation of the method proposed by McDermott and Simoncelli is that it does not work well for tonal sounds. The resynthesized results of sounds with tonal components such as wind chimes and church bells were perceived to have low realism.

Liao et al.[17] applied McDermott and Simoncelli's approach directly to a short-time Fourier transform (STFT), where marginals and subband correlations are extracted from the STFT of source signal, then iteratively imposed onto a new STFT for resynthesis.

Although there are a set of sounds that are generally accepted as sound textures, such as water flowing, fire crackling, and babble noise, there is not yet a generally quantitative definition for sound textures. Moreover, how sound texture is defined or rather defin-

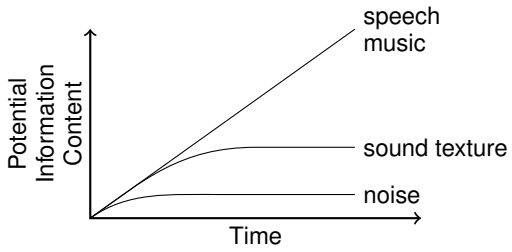


Figure 1: *Potential information content of a sound texture vs. time (from Saint-Arnaud[4])*

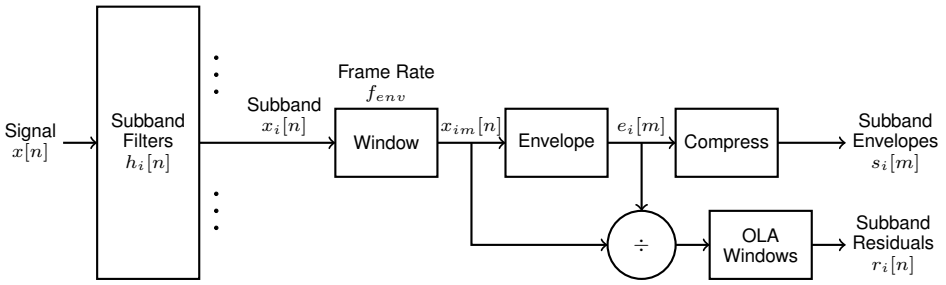


Figure 2: *Sound texture decomposition. The schematic illustrates the sound texture decomposition process for a single subband.*

ing the scope of sound textures, i.e., specifying which sounds are included in a given class of sound textures, in turn affects the approach to analyzing and synthesizing sound textures in that class.

In this paper, we limit our definition of sound textures to what can be synthesized using structured noise, that is, sounds that can be reproduced by shaping noise in a structured way. Despite the limiting definition, this approach can cover a broad range of sounds, as demonstrated by the aforementioned source-filter models. With this definition, sounds with pitch inflections, such as the sound of a baby crying or cars accelerating, will likely not fall into this category and thus will not be considered. Such a definition works well in conjunction with sines+noise synthesis [18] in which sinusoidal modeling handles any tonal components while texture classification, analysis, and synthesis can be applied to the residual signal after the tonal components are removed.

In the following sections, we examine sound textures with tonal components, compare it to non-tonal sound textures, and apply the insights gained from the comparisons to the developement of an analysis/synthesis model that includes tonal components.

2. SOUND TEXTURE DECOMPOSITION

We begin by formulating a method to decompose a sound texture into its subband sideband modulations, which we will call envelopes, and its residuals.[1] The decomposition process is illustrated in Figure 2.

The source sound texture $x[n]$ is first separated into subbands $x_i[n]$ with a subband filter bank equally spaced on an equivalent rectangular bandwidth (ERB) scale $h_i[n]$ [19]. We choose $h_i[n]$ such that its Fourier transform $H_i[k]$ satisfies,

$$\sum_i \left|H_i[k]\right|^2 = \mathbf{1}. \tag{1}$$

Thus, $\{h_i\}$ forms an FIR power-complementary filter bank [20]. The filter bank $h_i[n]$ is applied to the signal for both the analysis and synthesis steps. The analysis step gives

$$x_i[n] = h_i[n] * x[n]. \tag{2}$$

For subband $x_i[n]$, we first apply an analysis window $w[n]$ with 50% overlap on the signal at frame rate f_{env}. The length of the window is $L = 2R = 2/f_{env}$. We choose $w[n]$ to have constant overlap-add (OLA), i.e.,

$$\sum_m w^2[n + mR] = \mathbf{1}. \tag{3}$$

The window $w[n]$, like $h_i[n]$, is applied to the signal for the analysis and synthesis steps. We define the m-th windowed segment of $x_i[n]$ as

$$x_{im}[n] = w[n]x_i[n - mR]. \tag{4}$$

For each subband segment $x_{im}[n]$, we derive the uncompressed envelope of the segment $e_i[m]$ by taking the power within the windowed segment and normalizing it by the squared sum of the window $w[n]$,

$$e_i[m] = \left\{ \frac{\sum_{n=0}^{L-1}(x_{im}[n])^2}{\sum_{n=0}^{L-1}(w[n])^2} \right\}^{\frac{1}{2}} \tag{5}$$

Finally, a compression, simulating basilar membrane compression, is applied to $e_i[m]$ to obtain the subband envelopes $s_i[m]$.

$$s_i[m] = f_{\text{comp}}(e_i[m]) = (e_i[m])^{0.3} \tag{6}$$

Once we have the subband envelopes, we calculate the statistics for the envelope mean, variance, skewness, kurtosis, cross correlation, the envelope modulation power, between subband (C1) modulation correlation and within subband (C2) modulation correlation. The variance of each subband, which is equivalent to the subband power, is also saved.

The residual of segment $x_{im}[n]$ is derived by dividing the segment by the envelope value.

$$r_{im}[n] = x_{im}[n]/e_i[m] \tag{7}$$

The segment residuals are merged to obtain the subband residual $r_i[n]$ and the subband residuals are summed to obtain the signal residual $r[n]$.

$$r_i[n] = \sum_m w[n + mR]r_{im}[n + mR] \tag{8}$$

$$r[n] = \sum_i h_i[n] * r_i[n] \tag{9}$$

While this decomposition process differs from McDermott and Simoncelli [16], the resulting envelope $s_i[m]$ is very similar. The envelope statistics imposing algorithm from McDermott and Simoncelli can be applied with little modification. The advantage of this formulation is that all the residuals are aggregated into one signal $r[n]$.

[1] This follows McDermott and Simoncelli's terminology. The term "modulation" is used to describe the frequency components of the envelopes.

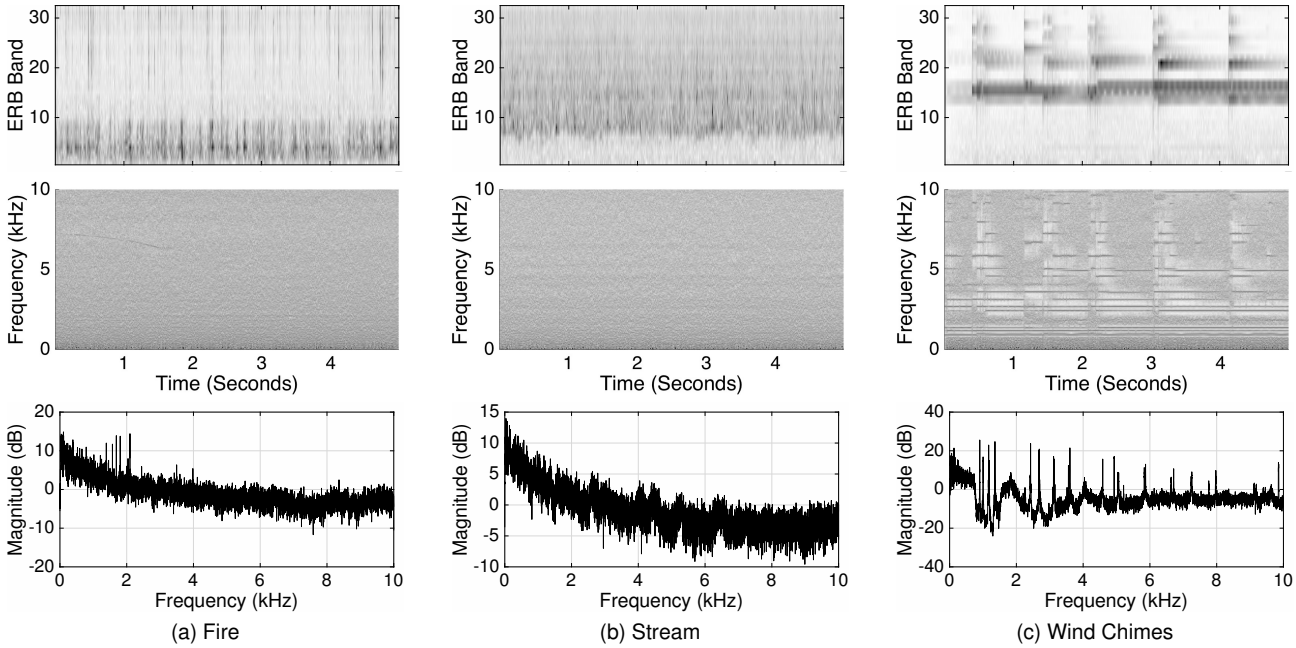


Figure 3: *Examples of sound texture decomposition and residual power spectral density. The first row shows the subband envelopes $s_i[m]$ of each signal. The second row shows the spectrogram of the residual $r[n]$. The third row is the power spectral density of the residual obtained using Welch's method. The y-axis of the envelope plot and the residual plot is different. (ERB scale vs. linear scale)*

3. RESIDUAL ANALYSIS

The envelopes and residuals from the sound texture decomposition can be viewed to have a carrier-modulation relation where the subband envelopes $s_i[m]$ are the amplitude modulations and the residual signal $r[n]$ is a temporally stable carrier signal. The subband envelopes, the spectrogram of the residual signal and the power spectral density (PSD) of the residual signal for example sound textures are shown in Figure 3.

The residual of crackling fire and flowing water is very close to pink (1/f) noise [21]. This is the result of normalizing the subband power over an ERB scale. Replacing the residual with pink noise for synthesis works well.

However, for a tonal sound like wind chimes (Figure 3c), the power spectral density is spiky due to the tonal components. Replacing the residual with pink noise would diffuse the tonal components, exciting the whole subband instead of focusing the signal power on a narrow band.

Inspecting the spectrogram of the residual in Figure 3c, the subband residuals do not look temporally stable, contrary to our assumption of carrier stability. Comparing the carrier spectrogram to the subband envelopes, we see that the subband envelopes have a small value where the tonals are missing. Thus, replacing the residual in Figure 3c with a temporally stable residual would not change the perceived output when merged with the subband envelopes.

Welch's method was used to estimate the power spectral density of the residual signal. We found that the shape of the tonal components is well captured when the averaging period is longer than 0.5 seconds. For the examples in this paper, an averaging period of 1 second was used at a sampling rate of 20 kHz.

4. RESIDUAL MODELING

We can impose the power spectral density directly onto the residuals during the synthesis process to improve the results. Moreover, this will allow synthesis of sounds with tonal components. However, the amount of data for directly imposing the PSD is very large. For our example, 1 second at 20 kHz results in 10001 samples for the PSD. Much of the data is noise, we only need the contour of the PSD. One method of reducing the amount of data needed is by modeling the audio using high order auto-regressive (AR) modeling. High order AR modeling has been used for gap-filling and spectral modeling [22, 14]. The advantage of this approach is that we get high quality results without handling sinusoid components and noise components separately. A similar approach to tonal noise modeling has been covered by Polotti and Evangelista [23].

For non-tonal sounds, a good approximation can be obtained using low order AR models. However, for tonal sounds, it is important to model the tonal components well, especially the peak sharpness. If a tonal peak is modeled too broadly, that tonal component will sound diffused.

In Figure 4a, the residual is modeled evenly at both AR orders 100 and 200. In Figure 4b, the tonal components around 1kHz are not modeled well at an AR order of 100. Increasing the AR order to 200 improves the results.

To find a reasonable AR order, we plot the standard deviation of the magnitude errors against the AR order. For the stream example, there is little improvement with higher orders. For the wind chime example, we see a noticeable improvement between AR order 100 and 200. Examining more examples, an AR order of 200 was sufficient to model the tonal examples used for this paper.

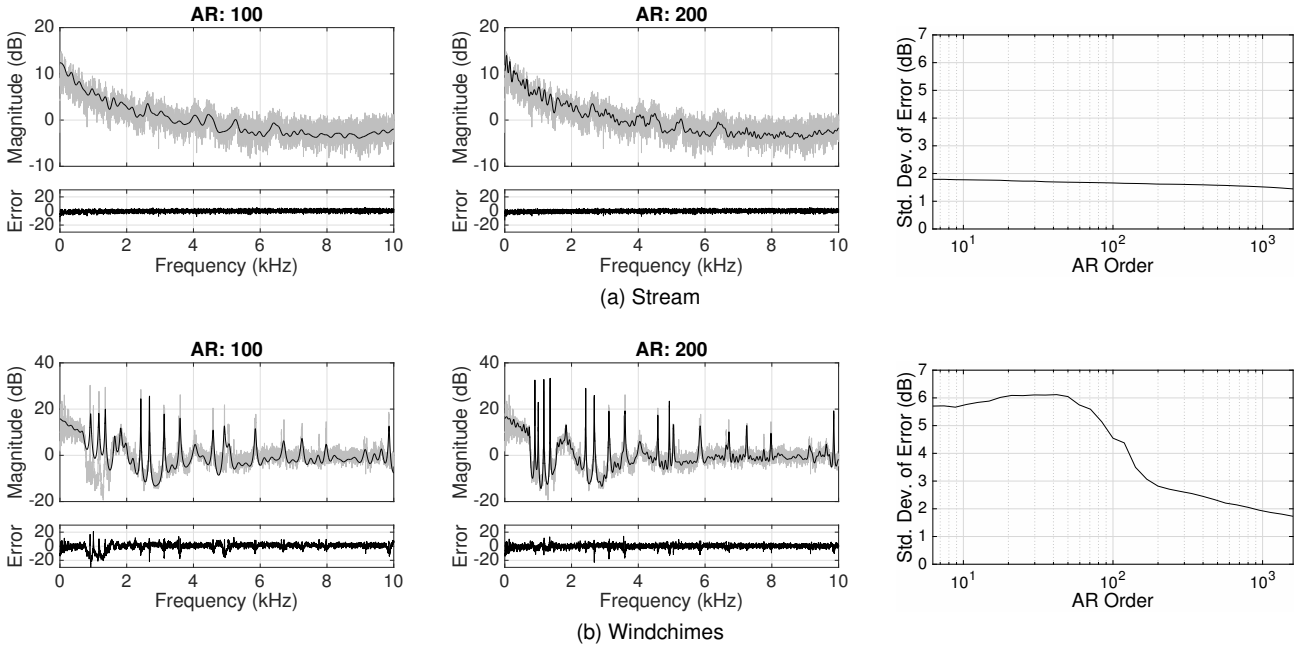


Figure 4: *Residual AR modeling. In the first two columns, the frequency response of an AR model (black) is overlaid on the residual PSD (gray). Below each power spectrum, the error between the actual response and the AR approximation is plotted. In the third column, the standard deviation of the modeling errors is plotted for different AR orders. There is little improvement when increasing the AR order for the stream example. However, for the windchime example, we see a noticeable improvement between AR order 100 and 200.*

5. SOUND TEXTURE RESYNTHESIS

In this section, the process of extracting statistics and features from a source sound texture is covered with a detailed explanation of how the extracted statistics are used for resynthesis. The analysis and synthesis process is illustrated in Figure 5.

5.1. Extracting Sound Texture Statistics

After separating the source into subbands, the variance of each subband is saved. The subband variance is equivalent to the power of each subband signal. The human auditory system has acute sensitivity to the power in each subband, thus imposing the subband power correctly is important. The subband variance is used to "equalize" the subbands when resynthesizing.

Each subband is decomposed into its envelope and residual components as formulated in §3. The subband envelopes are then used to extract a subset of the statistics described in McDermott and Simoncelli [16]. We include modulation statistics in envelope statistics since the modulation statistics are all derived from the subband envelopes. One noticeable difference is that the envelopes are not windowed, windowing is inherently applied in the decomposition step. A detailed description of the statistics used is provided in §9.

The subband residuals are merged back into a full-band single channel residual signal as explained in equation (9). The AR coefficients are estimated from the residual signal (§4) and the AR coefficients are saved for use as an all-pole filter to synthesize a new residual signal.

5.2. Synthesizing from Sound Texture Statistics

For resynthesis, starting with white noise, the envelopes and residuals are synthesized in parallel using the statistics from the analysis process. The two are then merged into subbands which are then equalized using the subband variances. The equalized subbands are then summed to form the final output signal.

5.2.1. Envelope Synthesis

After decomposing the subbands from the white noise signal into envelope and residual components, the residuals are discarded and only the envelopes are used for this step. The statistics imposing method was adapted from McDermott and Simoncelli [16]. For the target envelope statistics T_{env} extracted from the source sound texture and the current envelope statistics S_{env}, the L2 norm of $T_{env} - S_{env}$ is minimized using conjugate gradient descent.

Because the envelope mean is not normalized, it is not imposed in the gradient descent (See statistics formulas in §9). Instead, the envelope mean is imposed separately by adjusting the envelope means afterwards. It is worth noting that the uncompressed envelope $e_i[m]$, defined in equation (5), is proportionate to the power of the windowed segment $x_i m[n]$ and thus the envelope mean is closely related to the subband power. However, because the synthesized residuals may not be spectrally flat, the subband variances are enforced after composing the synthesized residuals and envelopes to ensure the subband powers are correctly equalized.

This process is iterated until the difference between the target statistics and the current statistics is below a certain threshold or

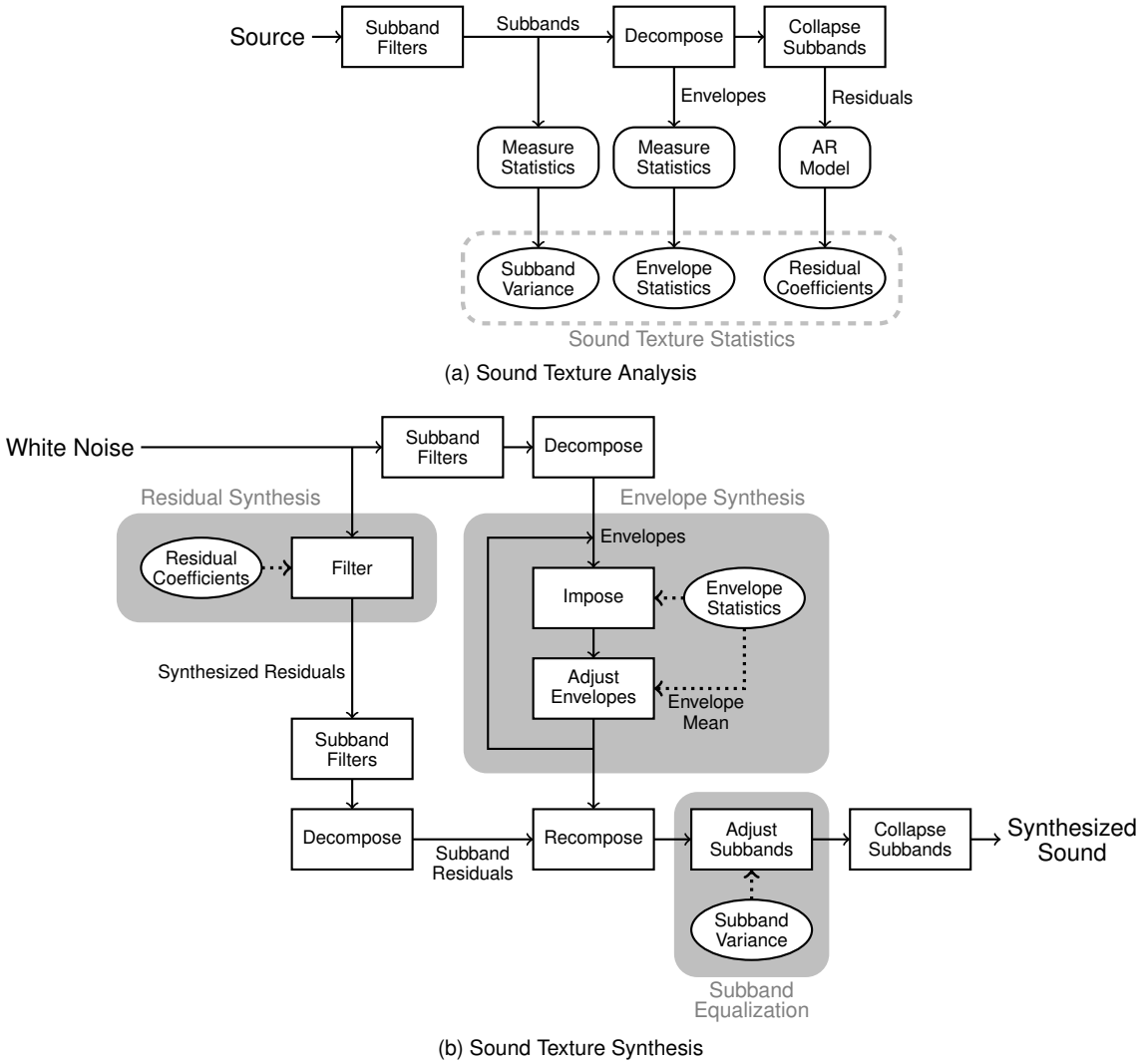


Figure 5: *Schematic of the sound texture analysis and synthesis procedure. The subband variance, envelope statistics and residual coefficients are measured and saved, then used for residual synthesis, envelope synthesis and subband equalization.*

the number of iterations pass a set limit. The process is not guaranteed to converge.

5.2.2. Residual Synthesis

The residual synthesis is straight forward. The input white noise signal is filtered with an all-pole filter composed of the AR coefficients from the analysis process. The synthesized residual is then decomposed into subband residuals and envelope components. The envelope components are discarded and the subband residuals are used for merging with the synthesized envelopes into subbands.

5.2.3. Equalization and Subband Rendering

Before merging the subbands, we adjust the variance of each subband. The equalization has a noticeable effect on the perception of the sound. In the recomposing step and collapse subband step the window $w[n]$ and the subband filters $h_i[n]$ are applied as synthesis windows and filters.

6. RESULTS

To test the effectiveness and validity of our model, we ran a user test where the participants were asked to rate the realism of resynthesized sound textures on a continuous scale from 1 to 7 with 1 being highly unrealistic and 7 being highly realistic.[2] Twelve subjects participated, 9 male, 3 female with a median age of 35. The participants were presented with the reference audio clip from which the statistics were extracted, along with 1 sample audio clip and 3 resynthesized audio clips, presented in random order. All audio clips were 4 seconds long.

[2]Sound samples used for the user tests are provided at `https://ccrma.stanford.edu/~hskim08/soundtextures/residual.html`.

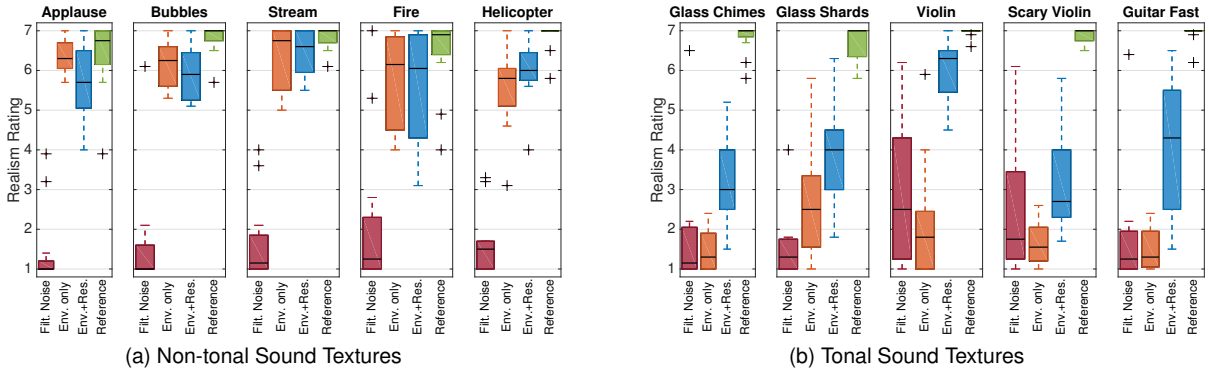


Figure 6: *User test realism ratings. The median for each sample is shown with a thick black line in within the box. The box covers the first to third quartile (25% to 75%). The whiskers cover about 99%. The + symbols represent the outliers. The realism scale provided to the users is 1-Highly unrealistic, 2-Unrealistic, 4-Acceptible, 6-Realistic, 7-Highly Realistic.*

The sample clip was taken from a different part of the same audio file as the reference audio. The three methods of resynthesis were 1) white noise filtered to match the PSD of the source audio, 2) white noise with only the envelope synthesized, and 3) white noise with both envelope synthesis and residual synthesis. The first method simulates residual only resynthesis, while the second case only uses the envelope statistics for resynthesis.

For the non-tonal sound textures, both the envelope only case and the case with both envelope and residual synthesis were perceived to have high realism. Filtered noise was perceived to contain low realism. For these examples, it seems most of the perceived information is in the envelopes and thus synthesizing the envelopes was sufficient to create realistic samples.

For sound textures with tonal components, the effect of the residual synthesis becomes visible. For violin sounds, filtered noise scaled higher realism than the envelope synthesis case, implying that the spectral information was more dominant in the perception of those sounds. In all cases, using both envelopes and residual for synthesis was perceived to be more realistic those that used only one.

Two examples worth noting are that of fire and violin. Despite having no tonal component, the fully resynthesized sample of fire was perceived to have lower realism than the other non-tonal examples. The fully resynthesized sample of violin on the other hand was perceived to have very high realism compared to other tonal examples. We believe this is due to the limitations of the temporal subband shaping modeled with the within subband (C2) modulation correlation. The crackling in fire as well as the attacks of the tonal examples have a noticeable temporal effect. However, we have found that the effects of enforcing the C2 correlation seemed to be limited. When the C2 correlation is easy to match, as in the violin example, we see that our method creates very compelling results.

7. CONCLUSIONS

We presented a method of decomposing a sound texture into its envelope and residual components. Examining the residuals for different examples, it was observed that non-tonal sound textures had residuals with power spectral densities close to pink noise, while that was not the case for tonal sound textures. Thus, the need for residual modeling. Applying high order auto-regression modeling to the residual, it was possible to reduce the data needed by a magnitude of two with little perceived differences. We presented a system for extracting the statistics from a source sound texture and the system for using the statistics to resynthesize new examples. The importance of both the residuals and envelopes was verified by a user test. For non-tonal sounds, a good envelope model was sufficient to synthesize realistic sounds. Adding the residual modeling did not affect the realism. However, for tonal sounds, modeling both the residuals and envelopes gave more realistic results than modeling just the residuals or the envelopes.

Taking a higher point of view, our approach fills the gap between filtered noise and the sound texture analysis presented by McDermott and Simoncelli [16]. Filtered noise captures the short term statistics in the form of power spectral distribution, including tonal components. Meanwhile summary statistics capture the modulations on the order of seconds.[3] Revisiting Saint-Arnaud's comparison of speech, music, noise, and sound textures in Figure 1, our model provides an explanation for the intuition behind the relation between potential information and time. The spectral distribution for noise can be estimated in a few milliseconds, while the subband modulations can be estimated on the order of seconds. The structure of speech and music is defined over a time span greater than that of seconds, usually minutes or longer.

The original objective of the study was to improve the sound texture model of McDermott and Simoncelli to cover tonal sound textures such as wind chimes. Over the course of time, we found that the model could be applied to constant pitch sounds such as a single note on a violin or guitar. We could model the textural aspect of the instrument sound such as fast bowing or tremolo picking. This suggests that the analysis of modulations could be applied to instrument modeling to add textural timbres.

While AR modeling was used to reduce the amount of data needed to synthesize the residuals, a sines+noise like approach could futher reduce the data. We were able to model the residuals of non-tonal sound textures sufficiently using AR orders of 10. By separately modeling the tonal peaks of the PSD, then using AR modeling only on the remaining residuals, it seems possible to reduce the amount of data by another order.

[3]The lowest modulation band used is 0.5Hz. See §9.

During the user test, the limited effectiveness of within subband (C2) modulation correlation enforcement for temporal modeling was observed. Improvements in temporal modeling seems to be an important factor in increasing the realism of the proposed sound texture synthesis method.

For this study, we limited the tonal sound textures to those that do not have variable pitch trajectories. This excludes most cases of vocalizations including bird songs, babies crying and speech. These sounds may require a completely different approach as there may be tonal components that move between subbands which may be challenging to model. Once more a sines+noise decomposition may prove to be useful for such cases, where the tonal components are modeled separately and the noise component, din, could be modeled by our proposed method.

Finally, there is a lack of evaluation metrics for sound textures. Evaluating the samples with PQevalAudio [24], all samples scored a very low objective difference grade, -3.5 or less, on a scale of 0 to -4 where 0 is imperceivable and -4 is very annoying. This seems to be caused by the fact that PQevalAudio compares the audio on a frame to frame basis meaning that it compares short term statistics whereas the synthesis for sound texture enforces long term statistics and as such the short term statistics can be very different. This is likely the case for other perceptual audio evaluation metrics. The short term measurements for sound textures may vary, yet the perception of the sounds are similar [25], suggesting that a different metric would be needed to programatically evaluate the perceived quality of resynthesized sound textures. Validating sound texture models with improved analysis/synthesis results should help make better perceptual evaluation metrics.

8. REFERENCES

[1] Nicolas Saint-Arnaud, "Classification of Sound Textures," M.S. thesis, Massachusetts Institute of Technology, 1995.

[2] Gerda Strobl, Gerhard Eckel, and Davide Rocchesso, "Sound Texture Modeling: A Survey," pp. 61–65, 2006.

[3] Diemo Schwarz, "State of the Art in Sound Texture Synthesis," in *Proc. of the 14th Int. Conference on Digital Audio Effects (DAFx-11)*, Paris, France, September 2011.

[4] Nicolas Saint-Arnaud and Kris Popat, "Analysis and Synthesis of Sound Textures," in *Readings in Computational Auditory Scene Analysis*, 1995, pp. 125–131.

[5] Ziv Bar-Joseph, Ran El-Yaniv, Dani Lischinski, Michael Werman, and Shlomo Dubnov, "Granular Synthesis of Sound Textures using Statistical Learning," in *Proceedings of the International Computer Music Conference*, Beijing, China, 1999.

[6] Lie Lu, Liu Wenyin, and Hong-Ziang Zhang, "Audio Textures: Theory and Applications," *IEEE Transactions on Speech and Audio Processing*, vol. 12, no. 2, pp. 156–167, Mar. 2004.

[7] Ananya Misra, Ge Wang, and Perry Cook, "Musical Tapestry: Re-composing Natural Sounds," *Journal of New Music Research*, vol. 36, no. 4, pp. 241–250, Dec. 2007.

[8] Gerda Strobl, *Parametric Sound Texture Generator*, Ph.D. thesis, Graz University of Technology, 2007.

[9] Martin Fröjd and Andrew Horner, "Sound Texture Synthesis Using an Overlap–Add/Granular Synthesis Approach," *Journal of the Audio Engineering Society*, vol. 57, no. 1/2, pp. 29–37, 2009.

[10] Shlomo Dubnov, Naftali Tishby, and Dalia Cohen, "Polyspectra as Measures of Sound Texture and Timbre," *Journal of New Music Research*, vol. 26, no. 4, pp. 277–314, Dec. 1997.

[11] Shlomo Dubnov, Ziv Bar-Joseph, Ran El-Yaniv, Dani Lischinski, and Michael Werman, "Synthesizing Sound Textures Through Wavelet Tree Learning," *Computer Graphics and Applications, IEEE*, vol. 22, no. 4, pp. 38–48, 2002.

[12] Doug Van Nort, Jonas Braasch, and Pauline Oliveros, "Sound Texture Analysis Based on a Dynamical Systems Model and Empirical Mode Decomposition," in *Audio Engineering Society Convention 129*, Nov 2010.

[13] Joan Bruna and Stéphane Mallat, "Audio Texture Synthesis with Scattering Moments," *arXiv.org*, Nov. 2013.

[14] Marios Athineos and Daniel PW Ellis, "Sound texture modelling with linear prediction in both time and frequency domains," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*. IEEE, 2003, vol. 5, pp. V–648.

[15] Xinglei Zhu and Lonce Wyse, "Sound Texture Modeling and Time-Frequency LPC," in *Proc. of the 7th Int. Conference on Digital Audio Effects (DAFx-04)*, Naples, Italy, October 2004.

[16] Josh H McDermott and Eero P Simoncelli, "Sound Texture Perception via Statistics of the Auditory Periphery: Evidence from Sound Synthesis," *Neuron*, vol. 71, no. 5, pp. 926–940, Sept. 2011.

[17] Wei-Hsiang Liao, Axel Roebel, and Alvin WY Su, "On the Modeling of Sound Textures Based on the STFT Representation," in *Proc. of the 16th Int. Conference on Digital Audio Effects (DAFx-13)*, Maynooth, Ireland, September 2013.

[18] Xavier Serra and Julius Smith, "Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[19] Brian R Glasberg and Brian CJ Moore, "A Model of Loudness Applicable to Time-Varying Sounds," *Journal of the Audio Engineering Society*, vol. 50, no. 5, pp. 331–342, 2002.

[20] Parishwad P Vaidyanathan, *Multirate Systems and Filter Banks*, Prentice-Hall, 1993.

[21] Bruce J West and Michael F Shlesinger, "On the Ubiquity of 1/f Noise," *International Journal of Modern Physics B*, vol. 03, no. 06, pp. 795–819, June 1989.

[22] Paulo AA Esquef and Luiz WP Biscainho, "An Efficient Model-based Multirate Method for Reconstruction of Audio Signals Across Long Gaps," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 4, pp. 1391–1400, 2006.

[23] Pietro Polotti and Gianpaolo Evangelista, "Fractal additive synthesis," *IEEE Signal Processing Magazine*, vol. 24, no. 2, pp. 105–115, March 2007.

[24] Peter Kabal, "An Examination and Interpretation of ITU-R BS. 1387: Perceptual Evaluation of Audio Quality," *TSP Lab Technical Report, Dept. Electrical & Computer Engineering, McGill University*, pp. 1–89, 2002.

[25] Josh H McDermott, Michael Schemitsch, and Eero P Simoncelli, "Summary Statistics in Auditory Perception," *Nature Neuroscience*, vol. 16, no. 4, pp. 493–498, Feb. 2013.

9. APPENDIX: ENVELOPE STATISTICS

The envelope statistics are adapted from McDermott and Simoncelli [16], with minor changes to accommodate the differences in how the subband envelopes $s_i[m]$ were derived. The most noticeable difference is the replacement of the window function $w(t)$ with $1/N$. We formulate the statistics here for completeness.

The envelope statistics can be categorized into subband envelope statistics and envelope modulation statistics. The subband statistics are directly measured from the subband envelopes, whereas the modulation statistics are measured after the subbands are filtered into modulation bands through a constant Q filter bank $\bar{f}_u[m]$ for the modulation power and an octave spaced filter bank $f_u[m]$ for the C1 and C2 correlations.

9.1. Subband Envelope Statistics

We start by defining the envelope moments. Defining the moments help simplify the definitions of the marginals. Precalculating the moments can reduce computation times. For the i-th subband envelope $s_i[m]$, the envelope moments are defined as,

$$\mathrm{m}_1[i] = \mu_i = \frac{1}{N}\sum_{m=1}^{N} s_i[m]$$

$$\mathrm{m}_X[i] = \frac{1}{N}\sum_{m=1}^{N} \{s_i[m] - \mu_i\}^X, \qquad X > 1$$

The standard deviation σ_i is also useful to precalculate.

$$\sigma_i = \sqrt{\mathrm{m}_2[i]}$$

9.1.1. Envelope Marginals

The envelope marginals, except for the envelope mean $\mathrm{M1}_i$, are normalized. This makes the statistics independent from any scaling factors. This is also important when imposing the statistics using optimization. Because the envelope mean is not normalized and tends to have smaller values than all other statistics used, it needs to be enforced separately after the optimization. The envelope marginals help shape the general distribution of the envelopes.

$$\mathbf{M1_i} = \mathrm{m}_1[i] = \mu_i$$

$$\mathbf{M2_i} = \frac{\mathrm{m}_2[i]}{(\mathrm{m}_1[i])^2} = \Big\{\frac{\sigma_i}{\mu_i}\Big\}^2$$

$$\mathbf{M3_i} = \frac{\mathrm{m}_3[i]}{(\mathrm{m}_2[i])^{3/2}} = \frac{1}{N}\frac{\sum_{m=1}^{N}(s_i[m]-\mu_i)^3}{\sigma_i^3}$$

$$\mathbf{M4_i} = \frac{\mathrm{m}_4[i]}{(\mathrm{m}_2[i])^2} = \frac{1}{N}\frac{\sum_{m=1}^{N}(s_i[m]-\mu_i)^4}{\sigma_i^4}$$

9.1.2. Envelope Cross-band Correlation

This is the correlation coefficient of two subband envelopes $s_i[m]$ and $s_j[m]$.

$$\mathrm{C_{ij}} = \frac{1}{N}\sum_{m=1}^{N}\frac{(s_i[m]-\mu_i)(s_j[m]-\mu_j)}{\sigma_i\sigma_j}$$

The envelope cross-band correlation helps enforce the comodulation of the subbands.

9.2. Envelope Modulation Statistics

Each subband envelope is further decomposed into its modulation bands through another filter bank. The modulation bands cover frequencies from 0.5Hz to $f_{env} = 400$Hz. Two different filter banks are used for the modulation power and the C1/C2 modulation correlations. The modulation power is calculated using a constant Q filter bank $\bar{f}_u[m]$.

$$\bar{b}_{i,u}[m] = \bar{f}_u[m] * s_i[m]$$

The C1/C2 modulation correlations are calculated using an octave band filter bank $f_u[m]$. An octave band is chosen because of the formulation of the C2 correlation.

$$b_{i,u}[m] = f_u[m] * s_i[m]$$

9.2.1. Modulation Power

The modulation power $\mathrm{M}_{i,u}$ is the root-mean-square of the modulation band normalized by the variance of the whole subband σ_i^2. The modulation power can be viewed as the distribution of the subband power within the modulation bands.

$$\mathbf{M_{i,u}} = \frac{1}{N}\frac{\sum_{m=1}^{N}(\bar{b}_{i,u}[m])^2}{\sigma_i^2}$$

9.2.2. Between Band Modulation (C1) Correlation

The C1 correlation is the correlation coefficient of two subband modulations $b_{i,u}[m]$ and $b_{j,u}[m]$ where i and j are the subband numbers and u is the modulation band number.

$$\mathbf{C1_{ij,u}} = \frac{1}{N}\sum_{m=1}^{N}\frac{b_{i,u}[m]b_{j,u}[m]}{\sigma_{i,u}\sigma_{j,u}}$$

where,

$$\sigma_{i,u} = \sqrt{\frac{1}{N}\sum_{m=1}^{N} b_{i,u}[m]}$$

The C1 correlation helps enforce the comodulation of subbands within the same modulation band.

9.2.3. Within Band Modulation (C2) Correlation

The C2 correlation enforces the temporal shape of a subband by imposing phase of modulation bands within a subband. To compare the phase of adjacent subbands, the modulation bands are transformed to its analytic signal $a_{i,u}$.

$$a_{i,u}[m] = b_{i,u}[m] + j\mathcal{H}\{b_{i,u}[m]\}$$

Next, the lower octave signal is expanded an octave by squaring the values, then normalized.

$$d_{i,u}[m] = \frac{(a_{i,u}[m])^2}{\|a_{i,u}[m]\|}$$

The correlation coefficient of the two bands is calculated for the C2 correlation.

$$\mathbf{C2_{i,uv}} = \frac{1}{N}\sum_{m=1}^{N}\frac{d_{i,v}^*[m]a_{i,u}[m]}{\sigma_{i,u}\sigma_{i,v}}$$

REDUCING THE ALIASING OF NONLINEAR WAVESHAPING USING CONTINUOUS-TIME CONVOLUTION

Julian D. Parker, Vadim Zavalishin, Efflam Le Bivic

Native Instruments GmbH,
Berlin, Germany
julian.parker@native-instruments.de

ABSTRACT

Nonlinear waveshaping is a common technique in musical signal processing, both in a static memoryless context and within feedback systems. Such waveshaping is usually applied directly to a sampled signal, generating harmonics that exceed the Nyquist frequency and cause aliasing distortion. This problem is traditionally tackled by oversampling the system. In this paper, we present a novel method for reducing this aliasing by constructing a continuous-time approximation of the discrete-time signal, applying the nonlinearity to it, and filtering in continuous-time using analytically applied convolution. The presented technique markedly reduces aliasing distortion, especially in combination with low order oversampling. The approach is also extended to allow it to be used within a feedback system.

1. INTRODUCTION

Nonlinear waveshaping has been part of the toolbox of musical signal processing since the 1960s, when overdrive and fuzz pedals became popular for treating the sound of the electric guitar. Its application in the digital signal processing domain began in the 1970s with exploration of synthesis methods employing waveshaping using Chebyshev polynomials [1, 2]. In more recent times, investigation of waveshaping has mainly been pursued within the domain of virtual analog modelling. A particularly active area of research has been filters with embedded nonlinearities, including the Moog ladder filter [3–7], the diode ladder filter [5, 8] and Sallen-Key based filters [9, 10]. Much work has also been done on the digital emulation of analog overdrive and fuzz pedals [11, 12] and tube amplifiers [13, 14]. Other nonlinear analog devices have been modelled, including the ring-modulator [15, 16] and bucket-brigade based effects [17, 18]. Recent work has applied correction functions usually used for oscillator antialiasing to the problem of antialiasing signals processed by a hard clipper [19], and has also considered more abstract applications of nonlinear waveshaping [20, 21].

One of the primary problems encountered when dealing with nonlinearities is that of aliasing distortion. Aliasing distortion is present in many types of digital signal processing algorithm, and is generally perceived to be disturbing or unpleasant [22]. In this paper, we describe a method for reducing the aliasing produced by processing a signal with a memoryless nonlinearity, as well as describe how the same method can be applied inside a filter. This method is based on forming a continuous-time approximation of the signal, applying the nonlinearity, and analytically deriving the result of applying convolution with a continuous-time lowpass filter kernel. The application of the filtering process in the continuous domain is crucial to working of the method, as it allows suppression of components even beyond the original Nyquist frequency of the system. The method is related to the Differentiated Polynomial Waveform (DPW) approach to antialiasing oscillator waveforms [23–25], which also applies filtering in the continuous-time domain. There is also some relation to methods of antialiasing wavetable playback using integrated wavetables [26, 27].

In Sec. 2, we describe the simplest formulation of the technique. Sec. 3 describes how the method can be extended to use any piecewise polynomial filter kernel, with the example of the triangular or linear interpolation kernel given. In Sec. 4, we discuss some drawbacks of the method—specifically its delay, and filtering effects below Nyquist. Examples of applying the method to a number of nonlinearities are given in Sec. 5. In Sec. 6, we broaden the method to apply to feedback systems, where the delay of the antialiased nonlinearity can be directly compensated by removal of equivalent parts of the filter structure. Finally, in Sec. 7 we draw conclusions about the presented work.

2. APPROXIMATION OF CONTINUOUS DOMAIN NONLINEAR WAVESHAPING

Nonlinear waveshaping in a digital signal processing context is generally applied directly to a discrete-time signal:

$$y[n] = f(x[n]) \tag{1}$$

where y denotes the output, x the input, n the discrete-time sample index, and f is an arbitrary nonlinear function. The waveshaping process is not bandlimited, and therefore depending on the input can generate frequency components exceeding the Nyquist frequency of the system (in many cases, even an infinite series of components). The components are reflected around the Nyquist frequency, and appear within the output spectrum as *aliasing* distortion. This is a well-known and common problem in musical digital signal processing. The most prominent method of reducing aliasing is to oversample the processing of the signal through the nonlinearity. This raises the Nyquist frequency, and hence the point at which the generated harmonics alias. However, this approach is still far from ideal given that the sequence of harmonics can be infinite.

The ideal result of the process, that of applying the nonlinearity to the input signal without any generated aliasing, can be thought of as perfect sampling of the same nonlinearity applied in continuous-time:

$$y(t) = f(\tilde{x}(t)) \tag{2}$$

where $\tilde{x}$ is a continuous-time reconstruction of our input signal, y is the continuous-time output signal, and t is the time variable. Given that we are working within a purely discrete-time context and presumably don't want to pass our signal out to the continuous domain for nonlinear processing, the challenge is to approximate this expression as accurately as possible whilst staying within the discrete-time domain.

2.1. Approximating a continuous-time input signal

The first challenge is to devise a version of the discrete-time input signal which can be treated in the same way as its continuous-time reconstruction. In the case of a known input signal, for example a sinusoid, this can be done trivially. In the case of an arbitrary signal, the problem is more complicated—we need to draw a function through the known sample-points of the input signal. One way of approaching this problem is to utilise a standard interpolation technique to 'fill in the gaps' between sample points—resulting in a piecewise approximation of the ideal continuous-time input signal. In the case of linear interpolation, and assuming a unit sampling interval, this would result in the following:

$$\tilde{x}(t) = \begin{cases} x_1 + \tau(x_0 - x_1), & 0 \leq t < 1 \\ x_2 + \tau(x_1 - x_2), & 1 \leq t < 2 \\ \vdots & \\ x_n + \tau(x_{n-1} - x_n), & (n-1) \leq t < n \end{cases} \tag{3}$$

where $x_n \equiv x[n]$ is shorter notation for the samples of the discrete-time input signal, and $\tau = 1 - (t \mod 1)$, a time variable that runs $1 \ldots 0$ between each sample.

Higher-order interpolation methods could be applied here, with the result of better suppression of the image spectra that repeat infinitely above the original Nyquist frequency. However, approximating the input signal as locally linear allows an analytic solution of the method to be derived, as will be seen in the following section.

2.2. Returning to the discrete-time domain

Now, given an expression for $\tilde{x}(t)$, we are able to calculate $y(t)$ at any specific t via (2). This can be considered to be sampling the approximated continuous-time signal at an arbitrary point. However, if we want our discrete-time output signal to be free of aliasing, we need to apply some kind of filtering to the continuous-time signal before it is sampled to remove components above our original Nyquist frequency. This can be done by applying a continuous-time convolution with some filter kernel h to $y(t)$:

$$\tilde{y}(t) = \int_{-\infty}^{\infty} h(u) y(t-u) du \tag{4}$$

where $\tilde{y}(t)$ is the approximately bandlimited continuous-time output. $\tilde{y}(t)$ can then be trivially sampled at the original sampling times (again assuming unit sample interval):

$$y[n] = \tilde{y}(n) \tag{5}$$

A very simple example of a lowpass kernel is a rectangular function of unit width:

$$h_{\text{rect}}(t) = \begin{cases} 1, & 0 \leq t \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Fig. 1 shows the amplitude response of this kernel, along with its time-domain form. From the amplitude response, we can see that the convolution will significantly attenuate any harmonics which exceed the original Nyquist frequency.

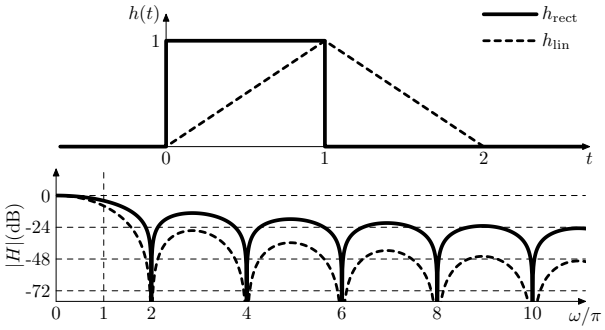


Figure 1: *Rectangular and linear continuous-time convolution kernels and their continuous time amplitude responses.*

We can now write an expression for the output $y[n]$:

$$\begin{aligned} y[n] = \tilde{y}(n) &= \int_{-\infty}^{\infty} h_{\text{rect}}(u) y(n-u) du \\ &= \int_0^1 y(n-u) du \\ &= \int_0^1 f(\tilde{x}(n-u)) du \end{aligned}$$

using (3), and noticing that over this interval $u = \tau$, we can write:

$$\begin{aligned} y[n] &= \int_0^1 f(\tilde{x}) d\tau \\ &= \int_0^1 f(x_n + \tau(x_{n-1} - x_n)) d\tau \end{aligned} \tag{7}$$

From integration by substitution, we can write:

$$\int_0^1 f(\tilde{x}) \frac{d\tilde{x}}{d\tau} d\tau = \int_{x_n}^{x_{n-1}} f(\tilde{x}) d\tilde{x}$$

The piecewise linear nature of $\tilde{x}$ now becomes useful as it means that $\frac{d\tilde{x}}{d\tau}$ is constant over the extent of the τ integration, and can be factored out of the integral to produce:

$$\begin{aligned} y[n] = \int_0^1 f(\tilde{x}) d\tau &= \frac{d\tau}{d\tilde{x}} \int_{x_n}^{x_{n-1}} f(\tilde{x}) d\tilde{x} \\ &= \frac{1}{x_{n-1} - x_n} \int_{x_n}^{x_{n-1}} f(\tilde{x}) d\tilde{x} \end{aligned} \tag{8}$$

Finally, by applying the fundamental theorem of calculus, we produce:

$$y[n] = \frac{F_0(x_n) - F_0(x_{n-1})}{x_n - x_{n-1}} \tag{9}$$

where F_0 is the antiderivative of f.

2.3. Precision and ill-conditioning concerns

In a digital context with finite precision arithmetic, (9) becomes ill-conditioned when $x_n \approx x_{n-1}$ due to a $0/0$-type uncertainty, resulting in precision loss or even a division by zero. Assuming floating point numeric representation the precision loss occurs from the subtraction of two values of the same sign and comparable magnitude. This kind of precision loss in the numerator can be

minimized by choosing the integration constant for F_0 such that $F_0(0) = 0$. In this case the precision loss in the entire formula is determined by that in the denominator.

According to the derivation given in Appendix A:

$$\frac{F_0(x_n) - F_0(x_{n-1})}{x_n - x_{n-1}} = f\left(\frac{x_n + x_{n-1}}{2}\right) + O((x_n - x_{n-1})^2) \tag{10}$$

where $O(\dots)$ denotes the order of the neglected terms, and hence the order of the error. This value can then be substituted for (9) when the value of $x_n - x_{n-1}$ becomes very small.

3. EXTENSION TO HIGHER-ORDER FILTER KERNELS

The filter kernel described in (6) is the simplest possible kernel that can be used in this technique. A more complex kernel can be used, as long as the convolution can be performed analytically. The triangular or linear-interpolation kernel is an example of such a kernel:

$$h_{\text{lin}}(t) = \begin{cases} t, & 0 \le t < 1 \\ 2 - t, & 1 \le t \le 2 \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

Substituting this into (4), and following the same procedure as previously, we obtain:

$$\begin{aligned} \tilde{y}(n) &= \int_{-\infty}^{\infty} h_{\text{lin}}(u) y(n-u) du \\ &= \int_0^1 u y(n-u) du + \int_1^2 (2-u) y(n-u) du \\ &= \int_0^1 \tau f(x_n + \tau(x_{n-1} - x_n)) d\tau + \\ &\quad \int_0^1 (1-\tau) f(x_{n-1} + \tau(x_{n-2} - x_{n-1})) d\tau \end{aligned} \tag{12}$$

The two integrals can be evaluated by noting that over the interval we're considering $\tau = \frac{\tilde{x} - x_n}{x_{n-1} - x_n}$. Therefore, by again applying integration by substitution, we have:

$$\begin{aligned} \tilde{y}(n) = &\int_{x_n}^{x_{n-1}} \frac{\tilde{x} - x_n}{(x_n - x_{n-1})^2} f(\tilde{x}) d\tilde{x} + \\ &\int_{x_{n-1}}^{x_{n-2}} \frac{x_{n-2} - \tilde{x}}{(x_{n-1} - x_{n-2})^2} f(\tilde{x}) d\tilde{x} \end{aligned} \tag{13}$$

As before, we can use the fundamental theorem of calculus to write this expression in terms of antiderivatives of f. Additionally, the antiderivative of $xf(x)$ is needed. We denote this as $F_1(x)$. After some simplification, the expression for the output can be written as:

$$\begin{aligned} \tilde{y}(n) = &\frac{x_n(F_0(x_n) - F_0(x_{n-1})) - (F_1(x_n) - F_1(x_{n-1}))}{(x_n - x_{n-1})^2} + \\ &\frac{x_{n-2}(F_0(x_{n-2}) - F_0(x_{n-1})) - (F_1(x_{n-2}) - F_1(x_{n-1}))}{(x_{n-2} - x_{n-1})^2} \end{aligned} \tag{14}$$

The frequency response of the triangular kernel can be seen in Fig. 1. As expected, the suppression of harmonics above Nyquist will be improved compared to the rectangular kernel. However, the expression needed to compute the anti-aliased output is more complex than in the rectangular kernel case. This is ameliorated somewhat by the fact that many terms can be calculated by storing and re-using the values calculated at previous time steps. Consequently, there should only be one evaluation of F_0 and of F_1 per sample.

The same derivation can be followed for any higher order kernel which consists of piecewise polynomial sections, for example a Hermite interpolator. However as the polynomial order of the kernel grows, the higher-order counterparts of the analytical convolution expressions (9) and (14) become more ill-conditioned. Therefore the precision requirements of the computation grow and so potentially the computational load.

3.1. Precision and ill-conditioning concerns

As in the rectangular window case, when $x_n \approx x_{n-1}$ or $x_{n-1} \approx x_{n-2}$, (14) becomes ill-conditioned. Again, this can be resolved as described in Appendix A. In this case, we must deal with the first and second terms of (14) separately. The first term becomes:

$$\frac{1}{2} f\left(\frac{x_n + 2x_{n-1}}{3}\right) \tag{15}$$

when $x_n \approx x_{n-1}$. Similarly, the second term becomes:

$$\frac{1}{2} f\left(\frac{x_{n-2} + 2x_{n-1}}{3}\right) \tag{16}$$

when $x_{n-1} \approx x_{n-2}$.

4. LINEAR RESPONSE AND GROUP DELAY OF METHOD

The conversion to the continuous-time domain and back can be seen as oversampling to an infinitely large sampling rate. The linear interpolation (3) and convolution (4) in that regard can be seen as lowpass filters used in resampling.

Additionally, many nonlinearities of interest become transparent at very low signal levels: $f(x) \approx x$. In this case the whole system becomes a combination of an upsampler and a downsampler. Therefore the system behaves as an LTI filter, where the overall filtering effect is a combination of the lowpass filters (3) and (4) and the aliasing occuring due to both filters being non-brickwall.

Using (41) of Appendix A we obtain the linear-case version of (9):

$$y[n] = \frac{x_n + x_{n-1}}{2} \tag{17}$$

Therefore at low signal levels (9) can be viewed as a half-sample fractional delay using linear interpolation (i.e. its group delay is 0.5 samples).

In a similar fashion for (14) we obtain

$$y[n] = \frac{x_n + x_{n-2}}{6} + \frac{2}{3} x_{n-1} \tag{18}$$

corresponding to a group-delay of 1 sample.

The respective amplitude responses of (17) and (18) are plotted in Fig. 2. Higher-order kernels can be treated in a similar fashion. Extension to the case with non-unity linear scaling at low signal levels is also trivial.

5. EXAMPLES AND RESULTS

In the following section, we describe how the methods described above can be applied to a number of nonlinearities.

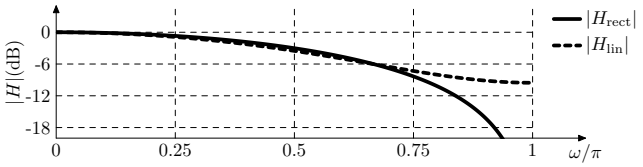


Figure 2: *Low signal-level discrete-time frequency responses of the method when using rectangular and linear continuous-time convolution kernels.*

5.1. Example: $\tanh()$

One of the most common nonlinear functions used in music signal processing is the hyperbolic tangent:

$$f(x) = \tanh(x) \tag{19}$$

the antiderivative is given by:

$$F_0(x) = \log(\cosh(x)) \tag{20}$$

therefore:

$$y[n] = \frac{\log(\cosh(x_n)) - \log(\cosh(x_{n-1}))}{x_n - x_{n-1}} \tag{21}$$

in the case where the rectangular kernel is used.

The antiderivative of $x\tanh(x)$ is given by:

$$F_1(x) = \frac{1}{2}\left(x\left(x + 2\log\left(e^{-2x}+1\right)\right) - \text{Li}_2\left(-e^{-2x}\right)\right) \tag{22}$$

where Li_2 is the dilogarithm function. The full expression for $y[n]$ can be obtained by substituting the expressions for $F_0(x)$ and $F_1(x)$ into (14).

The expression (22) is rather expensive to compute. Therefore it may be beneficial to tabulate it. Due to the ill-conditioned nature of (14) the tabulation needs to be done with high precision. Particularly, usage of piecewise segments of higher than linear order may be advised for the table. Also, warping of the argument scale (for example, making it logarithmic) can further reduce the table size. The unbounded argument range for the tabulated function can be achieved by noting that $\tanh(x) \approx \text{sgn}(x)$ within very high precision for $|x| \gg 1$, thus $F_1(x)$ can be approximated by

$$F_1(x) \approx F_1(x_0\,\text{sgn}(x)) + \frac{x^2 - x_0^2}{2}\,\text{sgn}(x),\ |x| \geq x_0 \gg 1 \tag{23}$$

for some sufficiently large x_0. The tabulation approach also allows us to deal with nonlinearities which cannot be analytically integrated.

Fig. 3 shows spectrograms of a linear sine sweep, processed by a $\tanh()$ nonlinearity with a linear input gain of 5, at two different sample rates, without and with the continuous-time convolution applied. At both 44.1kHz and 88.2kHz, there is a significant reduction in aliasing when the method is applied. At 44.1kHz a small loss of high-frequency content is visible, due to the effective discrete-time frequency response of the method. This effect is explained in Sec. 4. As can be seen, the triangular kernel produces noticeably greater suppression of aliased components than the rectangular kernel. However, the difference is not as large as that between no antialiasing and the rectangular kernel.

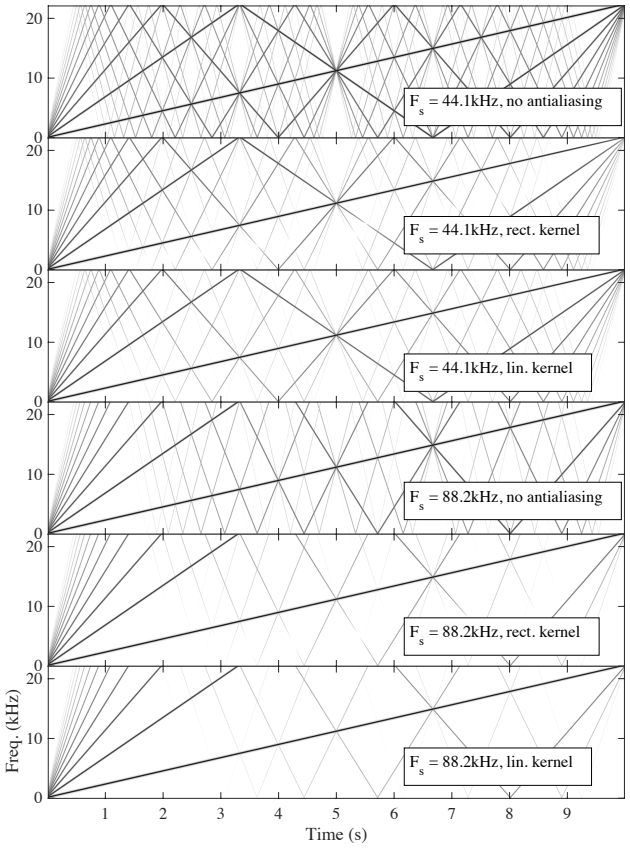


Figure 3: *Spectrogram of linear sine sweep from 0…22kHz, processed with* $\tanh()$ *function with an input gain of 5. The minimum amplitude visible is -80dBFS.*

5.2. Example: Hard Clipper

Another common saturating nonlinearity in music signal processing is the hard clipper, defined by:

$$f(x) = \begin{cases} x, & -1 \leq x \leq 1 \\ \text{sgn}(x), & \text{otherwise} \end{cases} \tag{24}$$

Recent work has considered alias-suppression for this function by applying a correction function to the transition between linear and clipped regions [19].

The antiderivative of (24) is trivially calculated, taking care to set the arbitrary constant of integration so that the function passes through the origin:

$$F_0(x) = \begin{cases} \frac{1}{2}x^2, & -1 \leq x \leq 1 \\ x\,\text{sgn}(x) - \frac{1}{2}, & \text{otherwise} \end{cases} \tag{25}$$

Similarly, the antiderivative of $xf(x)$ can be calculated:

$$F_1(x) = \begin{cases} \frac{1}{3}x^3, & -1 \leq x \leq 1 \\ (\frac{1}{2}x^2 - \frac{1}{6})\,\text{sgn}(x), & \text{otherwise} \end{cases} \tag{26}$$

In order to illustrate the effectiveness of the technique, a test was performed. A linear sine sweep wave processed by the nonlinearity with a very large input gain, in this case 10. The sample

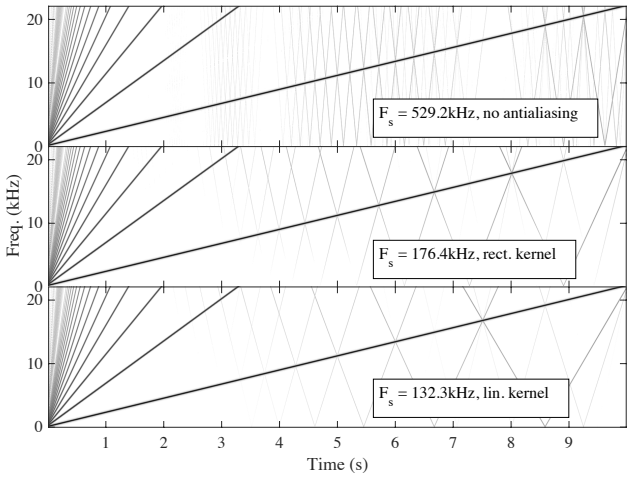


Figure 4: *Spectrogram of linear sine sweep from 0... 22kHz, processed with hard clipping function with an input gain of 10. The minimum amplitude visible is -80dBFS.*

rate of the cases was adjusted until roughly the same amount of aliasing was present. Signal-to-noise ratio (SNR) was calculated by creating an image mask denoting the non-aliased part of the spectrogram and calculating the ratio of the power inside the image mask to that outside the image mask. The mask is created from the spectrogram of an 'ideal' version of the processed signal (in this case calculated at $F_s = 11.2$MHz) by picking bins which have a power greater than –30dB. The results are given in Table 1 and shown in Fig. 4.

Table 1: *Oversampling required for similar aliasing level of hard clipper with input gain of 10.*

Kernel	F_s(kHz)	$F_s/44.1$kHz	SNR(dB)
None	529.2	12	46.7
Rectangular	176.4	4	46.3
Triangular	132.3	3	46.6

6. APPLICATION TO SYSTEMS WITH FEEDBACK

As discussed above in Sec. 4, the antialiasing method introduces delay into the signal path, which poses a problem in systems with feedback. This becomes especially critical in the systems with delayless feedback, which occurs in implicit time-discretization schemes, most prominently in trapezoidal integration. In the following we propose an approach to address this problem.

6.1. Delay elimination

Equation (17) is equivalent to the equation of the FIR part of a DF1 (Direct Form 1) trapezoidal integrator, shown in Fig. 5. Consider a serial connection of an antialiased nonlinearity Φ, as described in (9), and a DF1 trapezoidal integrator. This configuration is shown in Fig. 6. Since, according to (17), the nonlinearity (9) is already implementing the FIR part of the integrator, we can drop this part of the integrator from the structure, thereby eliminating the extra delay. The resulting structure is shown in Fig. 7.

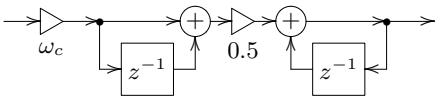


Figure 5: *Direct form 1 trapezoidal integrator (ω_c is the embedded cutoff control gain).*

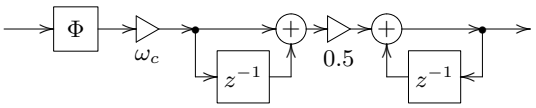


Figure 6: *A serial chain of an antialiased nonlinearity* (9) *(denoted by Φ) and a DF1 trapezoidal integrator.*

The nonlinearity and the integrator do not have to be located immediately next to each other in the feedback loop. However, any structural elements in the intermediate area must be considered. For example, consider a serial chain of a nonlinearity and a 1-pole lowpass filter, as shown in Fig. 8. In this case, as there is a summation node in between the nonlinearity and the integrator we must also introduce the same FIR part in the other path into the summation. This structure is shown in Fig. 9. It is interesting to note that the structure in Fig. 9 is equivalent to a naive 1-pole lowpass filter with an adjusted cutoff, as shown in Fig. 10.

Such manipulations change the topology of the system, and thus have the potential to change its time-varying behaviour. In practice, these changes are not usually severe enough to be a problem, but care must be taken.

The approach of Fig. 7 can be used as long as nonlinearities and integrators are interleaved in the feedback path. In the case where a nonlinearity does not have an associated integrator to absorb its delay, we can insert a very high cutoff (close to Nyquist) 1-pole lowpass filter immediately following (or preceding) it. This allows the delay to be absorbed, whilst hopefully having a minimal effect on the behaviour of the system. However, as the inserted lowpass has the potential to alter the linear (small signal) frequency response of the filter, care must again be taken.

6.2. Solving the antialiased implicit equation

Topologies containing delayless feedback require an implicit equation to be solved in order to be computable. If the implicit equation is transcendental, which is common with the types of saturating nonlinearity often used in musical filters, then linearisation is usually performed. A common approach to this problem is to apply the Newton–Raphson method. To this end, when the systems contains antialiased nonlinearities we need to be able to differentiate the antialiased nonlinearities with respect to their instantaneous in-

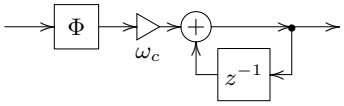


Figure 7: *A version of Fig. 6 with eliminated antialiasing delay.*

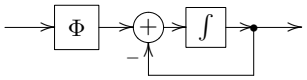


Figure 8: *A serial chain of an antialiased nonlinearity* (9) *and a 1-pole low pass filter.*

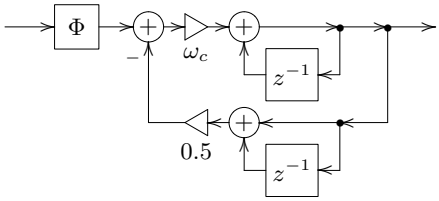


Figure 9: *Application of the delay elimination across a summation node in Fig. 8.*

put signal. Consider $y[n]$ in (9), expressed as a function of x_n:

$$y[n] = \Phi(x_n) = \frac{F_0(x_n) - F_0(x_{n-1})}{x_n - x_{n-1}} \tag{27}$$

Then

$$\frac{d\Phi}{dx_n} = \frac{f(x_n) \cdot (x_n - x_{n-1}) - \big(F_0(x_n) - F_0(x_{n-1})\big)}{(x_n - x_{n-1})^2} \tag{28}$$

Similarly to (9), the equation (28) becomes ill-conditioned when $x_n \approx x_{n-1}$. The ill-conditioned case substitute for (28) can be obtained by differentiating (10) with respect to x_n resulting in:

$$\frac{d\Phi}{dx_n} = \frac{1}{2} f'\left(\frac{x_n + x_{n-1}}{2}\right) + O(x_n - x_{n-1}) \tag{29}$$

6.3. Example: Moog Ladder Filter

Consider the Moog-ladder-like structure [4, 28] in Fig. 11, using trapezoidal integration. By folding the nonlinearity into the first of the four 1-pole lowpasses (as shown in Fig. 9) we eliminate the delay. Then we can apply Newton–Raphson to compute the solution of the implicit equation.

The performance of the approach was tested by driving the filter in Fig. 11 at $k = 8$ (strong selfoscillation) with a 5kHz unit amplitude sine oscillator, while sweeping the filter cutoff from 1Hz to 21kHz. We have plotted the spectrograms for non-antialiased and antialiased versions of filter, where we iterated Newton–Raphson until the convergence reached -60dB or better.

The output is shown in Fig. 12, at 44.1kHz and 88.2kHz sampling rates. The output at high sampling rate (576kHz) without antialiasing is provided as a reference. As can be easily seen, the antialiasing produces a noticeable reduction in aliasing. This has a particularly strong effect on the dynamics of the filter as the frequency is swept, as it prevents the resonant peak from locking on to particular frequencies where aliased components coincide. This change in dynamics is very clearly audible, with the non antialiased versions sounding 'steppy' as their cutoff is swept.

Additionally, the extra lowpass method was tested in the same filter topology. The extra lowpass filter's normalized prewarped cutoff was set to $\omega_c = 30$ (the corresponding unprewarped cutoff $\omega \approx 0.98\pi$). The results are also shown in Fig. 12, and appear

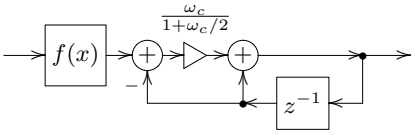


Figure 10: Structure of Fig. 9 with resolved local delayless feedback.

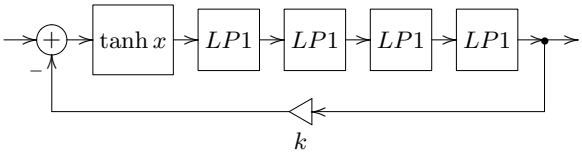


Figure 11: *A Moog-ladder-like filter.*

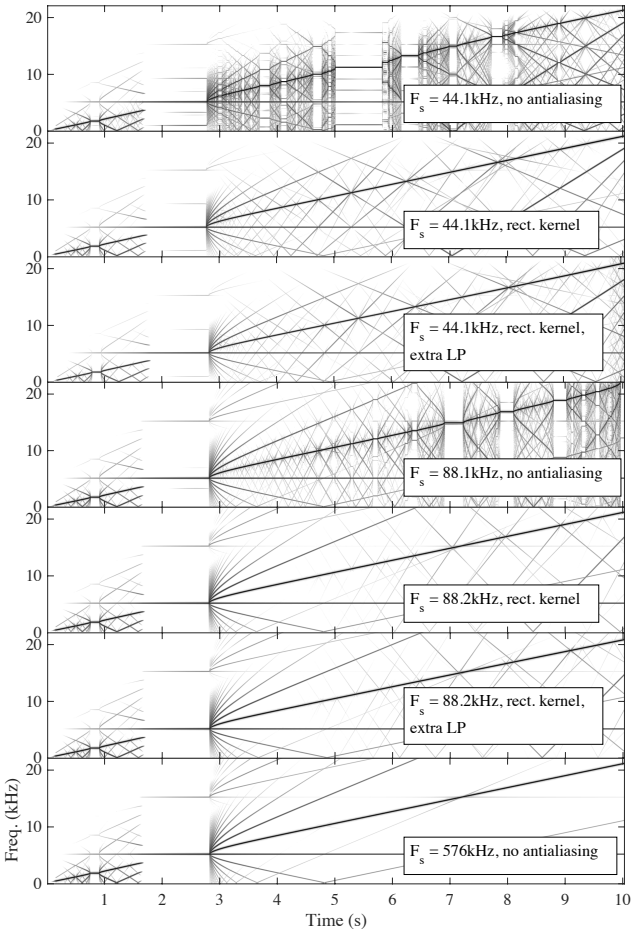


Figure 12: *Cutoff sweep of Moog-ladder-like filter with high resonance and 5kHz unit sinusoidal input. Shown without antialiasing, with antialiasing and integrator delay elimination, and with antialiasing and extra lowpass delay elimination.*

to be very similar to those given by the standard antialiasing tech-

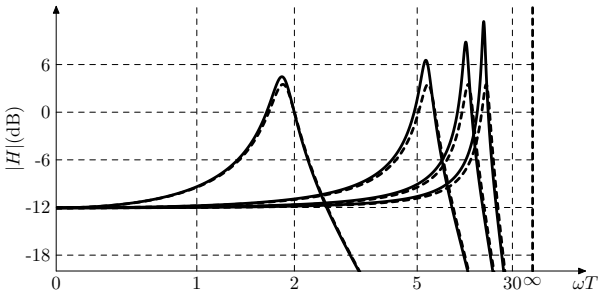


Figure 13: *Effect of the extra lowpass on the amplitude response of the Moog-ladder-like filter at $k = 3$. Dashed curves show the amplitude response in the absence of the extra lowpass. The ω axis is using the prewarped frequency scale (∞ is Nyquist, 2 is half Nyquist frequency, 30 is the extra lowpass's cutoff).*

nique. However, the inserted lowpass increases the filter's resonance and lowers the resonant frequency at high cutoff settings, as illustrated by Fig. 13. This effect is not strongly audible at sampling rates of 88.2kHz or above.

7. CONCLUSION

In this work, we have described a new method of suppressing aliasing when processing a digital signal with a nonlinear waveshaper. The method allows generated harmonics to be suppressed above Nyquist, by constructing a continuous-time approximation of the input signal, applying the waveshaping, and then analytically applying a convolution with a continuous-time filter kernel. The method is especially effective in situations where the nonlinearity generates a large amount of harmonics, such as a saturating nonlinearity with large input gain. In these situations, applying the antialiasing can give a similar improvement to a much larger level of oversampling. Techniques for applying the method within feedback systems were also given.

Sound and code examples illustrating the techniques described in this paper are available at the accompanying website[1].

8. ACKNOWLEDGEMENTS

After the initial submission of this work, the authors were made aware of a private unpublished work by Antti Huovilainen from 2010 that describes a subset of the technique detailed in this paper. The authors were not aware of this work during the writing process, but would like to acknowledge its existence.

[1] `http://github.com/julian-parker/DAFX-AntiAliasing`

9. REFERENCES

[1] M. Le Brun, "Digital waveshaping synthesis," *J. Audio Eng. Soc.*, vol. 27, no. 4, pp. 250–266, 1979.

[2] C. Roads, "A tutorial on non-linear distortion or waveshaping synthesis," *Computer Music Journal*, vol. 3, no. 2, pp. 29–34, 1979.

[3] A. Huovilainen, "Non-linear digital implementation of the Moog ladder filter," in *Proc. 7th Int. Conf. Digital Audio Effects*, Naples, Italy, Oct. 2004, pp. 61–64.

[4] V. Välimäki and A. Huovilainen, "Oscillator and filter algorithms for virtual analog synthesis," *Computer Music J.*, vol. 30, no. 2, pp. 19–31, 2006.

[5] V. Zavalishin, *The art of VA filter design*, Native Instruments, 2012.

[6] S. D'Angelo and V. Välimäki, "Generalized Moog ladder filter: Part I–linear analysis and parameterization," *IEEE/ACM Trans. Audio, Speech, and Lang. Process.*, vol. 22, no. 12, pp. 1825–1832, 2014.

[7] S. D'Angelo and V. Välimäki, "Generalized Moog ladder filter: Part II–explicit nonlinear model through a novel delay-free loop implementation method," *IEEE/ACM Trans. Audio, Speech, and Lang. Process.*, vol. 22, no. 12, pp. 1873–1883, 2014.

[8] F. Fontana and M. Civolani, "Modeling of the EMS VCS3 voltage-controlled filter as a nonlinear filter network," *IEEE Trans. Audio, Speech, and Language Process.*, vol. 18, no. 4, pp. 760–772, May 2010.

[9] A. Huovilainen, "Design of a scalable polyphony-MIDI synthesizer for a low cost DSP," M.S. thesis, Aalto University, 2010.

[10] J. D. Parker and S. D'Angelo, "A digital model of the Buchla low-pass-gate," in *Proc. 16th Int. Conf. Digital Audio Effects (DAFx-13)*, Maynooth, Ireland, 2013, pp. 278–285.

[11] D. T. Yeh, J. S. Abel, and J. O. Smith, "Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinary differential equations," in *Proc. 10th Int. Conf. on Digital Audio Effects (DAFx07)*, Bordeaux, France, 2007, pp. 197–204.

[12] D.T. Yeh, J.S. Abel, and J.O. Smith, "Automated physical modeling of nonlinear audio circuits for real-time audio effects; part I: Theoretical development," *IEEE Trans. Audio, Speech, and Language Process.*, vol. 18, no. 4, pp. 728–737, May 2010.

[13] J. Pakarinen and D. T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music J.*, vol. 33, no. 2, pp. 85–100, 2009.

[14] R. C. D. de Paiva, J. Pakarinen, V. Välimäki, and M. Tikander, "Real-time audio transformer emulation for virtual tube amplifiers," *EURASIP J. Advances Signal Process.*, 2011.

[15] R. Hoffmann-Burchardi, "Digital simulation of the diode ring modulator for musical applications," in *Proc. 11th Int. Conference on Digital Audio Effects (DAFx-08)*, Espoo, Finland, Sept. 2008.

[16] J. D. Parker, "A simple digital model of the diode-based ring-modulator," in *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011, pp. 163–166.

[17] A. Huovilainen, "Enhanced digital models for analog modulation effects," in *Proc. Int. Conf. Digital Audio Effects*, Madrid, Spain, 2005, pp. 155–160.

[18] C. Raffel and J. O. Smith, "Practical modeling of bucket-brigade device circuits," in *Proc. 13th Int. Conf. Digital Audio Effects (DAFx-10)*, Graz, Austria, Sep. 2010, pp. 50–56.

[19] F. Esqueda, V. Välimäki, and S. Bilbao, "Aliasing reduction in soft-clipping algorithms," in *23rd European Signal Processing Conference (EUSIPCO)*, Nice, France, Aug. 2015, pp. 2014–2018.

[20] B. De Man and J. D. Reiss, "Adaptive control of amplitude distortion effects," in *Proc. Audio Eng. Soc. 53rd Int. Conf.*, Jan 2014.

[21] J. Kleimola, V. Lazzarini, J. Timoney, and V. Välimäki, "Vector phase shaping synthesis," in *Proc. 14th Int. Conf. Digital Audio Effects (DAFx-11)*, Paris, France, 2011.

[22] H.-M. Lehtonen, J. Pekonen, and V. Välimäki, "Audibility of aliasing distortion in sawtooth signals and its implications for oscillator algorithm design," *J. Acoust. Soc. Am.*, vol. 132, no. 4, pp. 2721–2733, 2012.

[23] V. Välimäki, "Discrete-time synthesis of the sawtooth waveform with reduced aliasing," *IEEE Signal Process. Letters*, vol. 12, no. 3, pp. 214–217, 2005.

[24] V. Välimäki, J. Nam, J. O. Smith, and J. S. Abel, "Alias-suppressed oscillators based on differentiated polynomial waveforms," *IEEE Trans. Audio, Speech and Language Process.*, vol. 18, no. 4, pp. 786–798, 2010.

[25] V. Välimäki, J. Pekonen, and J. Nam, "Perceptually informed synthesis of bandlimited classical waveforms using integrated polynomial interpolation," *J. Acoust. Soc. Am.*, vol. 131, pp. 974–986, 2012.

[26] G. Geiger, "Table lookup oscillators using generic integrated wavetables," in *Proc. 9th Int. Conf. on Digital Audio Effects (DAFx-06)*, Montreal, Canada, 2006, p. 169.

[27] A. Franck and V. Välimäki, "Higher-order integrated wavetable and sampling synthesis," *J. Audio Eng. Soc.*, vol. 61, no. 9, pp. 624–636, 2013.

[28] V. Zavalishin, *Preserving the LTI system topology in s- to z-plane transforms*, Native Instruments, 2008.

A. DERIVATION OF RESOLVED OUTPUT IN ILL-CONDITIONED AREA AND SYSTEM RESPONSE IN LINEAR CASE

Let us consider one linear segment of $\tilde{x}(t)$, the continuous-time approximation of the input signal, being processed by the nonlinearity f:

$$y(\tau) = f(x_n + \tau(x_{n-1} - x_n)) \tag{30}$$

In order to calculate the output of the continuous-time convolution, we must calculate one or more integrals of the form:

$$\int_0^1 y(\tau)\bar{h}(\tau)\,d\tau \tag{31}$$

where $\bar{h}$ is one particular *unipolar* piecewise segment of the convolution kernel. The analytical expressions for integrals (31) become ill-conditioned at $x_n \approx x_{n-1}$, therefore we wish to obtain alternative expressions that can be substituted for the ill-conditioned case.

(31) can be seen as a weighted average of $y(\tau)$, where $\bar{h}(\tau)$ is the weight function. From the mean value theorem there exists $\tau_0 \in [0, 1]$ such that:

$$\int_0^1 y(\tau)\bar{h}(\tau)\,d\tau = y(\tau_0)\int_0^1 \bar{h}(\tau)\,d\tau \tag{32}$$

For $x_n \approx x_{n-1}$, in principle any $\tau_0 \in [0, 1]$ can be chosen for the approximation. However, we wish to find the most optimal choice. Introducing a normalized weight function:

$$w(\tau) = \bar{h}(\tau) \Big/ \int_0^1 \bar{h}(\tau)\,d\tau \tag{33}$$

we rewrite (32) as:

$$\int_0^1 y(\tau)w(\tau)\,d\tau = y(\tau_0) \tag{34}$$

For $x_n \approx x_{n-1}$ the function $y(\tau)$ can be considered approximately linear on $[0, 1]$:

$$y(\tau) = a + b\tau + O((x_n - x_{n-1})^2) \tag{35}$$

where the error term is assuming the bounded second derivative of $f(x)$. From (34) we then have:

$$\begin{aligned} y(\tau_0) &= \int_0^1 y(\tau)w(\tau)\,d\tau \\ &= \int_0^1 (a + b\tau + O((x_n - x_{n-1})^2))w(\tau)\,d\tau \\ &= (a + O((x_n - x_{n-1})^2))\int_0^1 w(\tau)\,d\tau + b\int_0^1 \tau w(\tau)\,d\tau \\ &= a + bM_1 + O((x_n - x_{n-1})^2) \\ &= y(M_1) + O((x_n - x_{n-1})^2) \end{aligned} \tag{36}$$

where

$$M_1 = \int_0^1 \tau w(\tau)\,d\tau \tag{37}$$

is the first moment of the weight function $w(\tau)$. That is

$$y(\tau_0) = y(M_1) + O((x_n - x_{n-1})^2) \tag{38}$$

and therefore we choose

$$\tau_0 = M_1 = \int_0^1 \tau\bar{h}(\tau)\,d\tau \Big/ \int_0^1 \bar{h}(\tau)\,d\tau \tag{39}$$

Using this τ_0 and (32), we can then compute the integrals (31) in the ill-conditioned case.

Further, for $f(x) = x$ the equation (30) turns into

$$y(\tau) = x_n + \tau(x_{n-1} - x_n)) \tag{40}$$

while the term $O((x_n - x_{n-1})^2)$ vanishes from (35) and the following equations. Using (40) and (39) we turn (32) into

$$\int_0^1 y(\tau)\bar{h}(\tau)\,d\tau = (x_n + (x_{n-1} - x_n)M_1)\int_0^1 \bar{h}(\tau)\,d\tau \tag{41}$$

which allows us to calculate the response of the system in cases where $f(x)$ is approximately transparent.

SOUND MORPHING BY AUDIO DESCRIPTORS AND PARAMETER INTERPOLATION

Savvas Kazazis
CIRMMT,
McGill University
Montreal, Canada
savvas.kazazis@mail.mcgill.ca

Philippe Depalle
CIRMMT,
McGill University
Montreal, Canada
depalle@music.mcgill.ca

Stephen McAdams
CIRMMT,
McGill University
Montreal, Canada
smc@music.mcgill.ca

ABSTRACT

We present a strategy for static morphing that relies on the sophisticated interpolation of the parameters of the signal model and the independent control of high-level audio features. The source and target signals are decomposed into deterministic, quasi-deterministic and stochastic parts, and are processed separately according to sinusoidal modeling and spectral envelope estimation. We gain further intuitive control over the morphing process by altering the interpolated spectrum according to target values of audio descriptors through an optimization process. The proposed approach leads to convincing morphing results in the case of sustained or percussive, harmonic and inharmonic sounds of possibly different durations.

1. INTRODUCTION AND RELATED WORK

Sound morphing plays an important role in many areas including sound design for compositional applications and video games, speech manipulation, and in generating stimuli with specific and controllable acoustic parameters that are used in psychoacoustic experiments [1, 2]. Despite the extensive literature on this topic, there is no consensus on a single definition of audio morphing, and an extensive discussion on different viewpoints can be found in [3]. In this paper we present a strategy for stationary morphing, as opposed to *dynamic morphing*, in which a source sound gets continuously transformed over time into a target sound. We consider *static morphing* as a process that hybridizes a source sound with target sounds, or target audio features, through the independent manipulation of acoustic parameters.

Additive synthesis is one of the most flexible techniques, and as such many morphing strategies rely on interpolating the parameters of a sinusoidal model [4, 5, 6, 7, 8]. Tellman et al. [4] first pair the partials of the two sounds by comparing their frequency ratios to the fundamental frequency, and afterwards they interpolate their frequency and amplitude values. They also time-scale the two sounds to morph between their tremolo and vibrato rates based on assumptions that usually do not hold in the case of most natural sounds. Osaka [5] first performs dynamic time warping (DTW), and then he finds partials' correspondences by dynamic programming. The residual is modeled with short partials and is morphed according to stochastic parameter interpolation with hypothesized distributions. Fitz et al. [6] estimate the parameters of the "Bandwidth Enhanced Model" [9] by reassigned spectrograms, and use morphing envelopes to control the evolution of the frequency, amplitude, bandwidth and noisiness of the morph. Haken et al. [7] use a similar technique to morph in real time between pre-analyzed sounds that are placed in a three-dimensional timbre control space. Boccardi and Drioli [8] use Gaussian Mixture Models (GMM) to morph only the partials' magnitudes, which are derived from Spectral Modeling Synthesis (SMS) [10]. According to Boccardi and Drioli, since the morphing is based only on magnitude transformations, the source and target signals should belong to the same instrument family.

Other morphing strategies rely on interpolating the parameters of a source-filter model. Slaney et al. [11] construct a multidimensional space that encodes spectral shape and fundamental frequency on orthogonal axes. Spectral shape is derived through Mel-Frequency Cepstral Coefficients (MFCC) and fundamental frequency by the residual spectrogram. The optimum temporal match between the source and target sounds is found using DTW based on MFCC distances. The smooth and pitch spectrograms are interpolated separately. Ezzat et al. [12] argue that interpolating the spectral envelopes by simple cross-fading, as in [11], does not account for proper formant shifting. They describe a method for finding correspondences between spectral envelopes so as to encode the formant shifting that occurs from a source to a target sound. The morphing is based on interpolating the warped versions of the two spectral envelopes, and morphing between the residuals is left for future work.

Other authors claim to control synthesis parameters or to morph according to perceptual dimensions by using high-level audio features. Hoffman and Cook [13] propose a general framework for feature-based synthesis according to an optimization scheme that maps synthesis parameters to target feature values. The results are very preliminary: the source sound consists of stationary sinusoids and noise that is spectrally shaped through MFCCs; the target features are limited to spectral centroid, spectral roll-off and fundamental frequency histograms. Park et al. [14] treat single features as modulation signals that are applied to a source sound. According to their proposed scheme, different features cannot be controlled independently and thus the combination of multiple target features leads to unpredictable results. Mintz [15] uses linear constrained optimization on audio descriptors to control the parameters of an additive-plus-noise synthesizer. Williams and Brookes [16] morph using SMS according to verbal attributes that correlate with audio descriptors and in [17] employ a similar technique to morph between prerecorded sounds and sounds captured in real time. Hikichi and Osaka [18] adjust the parameters of a physical model using the spectral centroid as a reference to morph between piano and guitar sounds, and Primavera et al. [19] focus on the importance of decay time when morphing between percussive sounds of the same family. Coleman and Bonada [20] derive analytic relations for the spectral centroid and standard deviation to control adaptive effects for resampling and band-pass equalization. Caetano and Rodet in [21] investigate spectral envelope representations, which lead to linearly varying values of audio descriptors when linearly interpolated according to a morphing factor, and in

[22] use optimization techniques based on genetic algorithms to obtain morphed spectral envelopes that approximate target audio descriptor values.

Other approaches rely strictly on the time domain [23] or on time-frequency representations [24, 25]. Röbel [23] models the signals as dynamical systems using neural networks and morphs by interpolating their corresponding attractors. According to the author, the attractors of the two sounds should be topologically equivalent for achieving a convincing morphing. Ahmad et al. [24] propose a scheme for morphing between transient and non-stationary signals using the discrete wavelet transform (DWT) along with singular value decomposition (SVD) for interpolating the wavelet coefficients. Olivero et al. [25] propose a sound morphing technique without making any presumptions about the nature of the signal or its underlying model. The technique relies on the interpolation of Gabor masks and its penalty-based version is shown to encompass typical cross-synthesis strategies used in computer music applications. Furthermore, the interpretation of one of the strategies in terms of Bregman divergences allows them to include constraints that force morphing intermediates to exhibit a predesigned temporal sequence of centroids. This approach works well only as long as there is overlapping energy between the sounds and in our opinion, certain presumptions about the nature of the signal are necessary for choosing an appropriate morphing strategy.

Table 1 shows a brief comparison between the above-presented methods that are applicable to static morphing and the current approach. In Section 2 we present an overview of our proposed approach. Section 3 describes in detail the morphing process based on parameter interpolation, and Section 4 presents the optimization scheme used for morphing based on higher-level audio features. In Section 5 we present our concluding remarks and future improvements of our method.

2. A HYBRID APPROACH TO SOUND MORPHING

The morphing scheme presented here requires a source sound, to which we apply timbral transformations according to a morphing factor "α" ($0 \leq \alpha \leq 1$), and a target. A value of $\alpha = 0$ corresponds to the source sound and a value of $\alpha = 1$ corresponds to the target sound. The target could consist only of specific audio descriptor values that are obtained according to a morphing factor α_{d} and applied to the source sound, or it could be a different sound from which we extract the audio descriptors that we want to morph accordingly, but we also interpolate between the spectrotemporal fine structures of the two according to a morphing factor α_{p}. Depending on their spectral content, the source and target sounds can be decomposed into three parts as in [5]: a deterministic part, which is related to harmonic and inharmonic qualities; a quasi-deterministic part, which is more related to transients and spectrotemporal irregularities; and a stochastic part, which is related to noise color. The deterministic and quasi-deterministic parts are estimated through sinusoidal modeling from which we obtain the time-varying frequencies, amplitudes and phases of the partials. The stochastic parts are derived by subtracting the deterministic and quasi-deterministic parts from the original signals [10] and are modeled by estimating their spectral envelopes.

In the next step, we compute the time-varying audio descriptors on each of the three parts and for each analysis frame. Audio descriptors that are applicable to the current approach are presented in detail in [26]. For the purposes of this study we have experimented with: spectral centroid and higher order statistical moments of the spectrum including the standard deviation (referred to as spectral spread), spectral skewness, and spectral kurtosis; spectral decrease; and spectral deviation, which is only computed on the deterministic part of the signal. Descriptors that are applicable exclusively to harmonic (or slightly inharmonic) signals, such as tristimulus values and the odd-to-even harmonic ratio, are also applicable. Natural sounds, however, rarely exhibit such well-defined properties, and thus such descriptors would be more suitable in the case of synthetic or simplified natural sounds. Once we calculate the descriptors of the source and target sounds we can compute intermediate values according to the morphing factor α_d, and we interpolate the model parameters of the deterministic, quasi-deterministic and stochastic parts separately. The intermediate values of audio descriptors are applied to the parameter-interpolated signals using the optimization scheme described in Section 4.

We chose to model differently the stochastic part, on the one hand, and the deterministic and quasi-deterministic parts, on the other hand, because not all sounds exhibit a strong formant structure. As such, spectral envelopes would be a poor estimation of the signal, unless they are estimated by the tracked partials, as in [10, 27, 28]. On the other hand, it is well known that if the signal is stochastic-only, sinusoidal modeling usually leads to artifacts and so a morphing scheme based exclusively on this model would degrade the sound quality. The separation into deterministic and quasi-deterministic parts is necessary for improving the estimation of partial-to-partial correspondences, as we discuss in Section 3.1.1. In the following we assume that the source and target sounds are equalized in loudness, have the same fundamental frequencies, and can be of different durations.

3. PARAMETER INTERPOLATION

In this section we describe the interpolation schemes based on the parameters of the sinusoidal model and the parameters that model the spectral envelopes of the residuals.

3.1. Deterministic and quasi-deterministic parts

The following scheme is used for both the harmonic and quasi-harmonic parts. Before interpolating the parameters of the sinusoidal model, it is necessary to find partial-to-partial correspondences between the source and target sounds.

3.1.1. Estimating partial-to-partial correspondences

The deterministic part consists of partials that are long in duration, with respect to the total duration of the analyzed sound, whereas the quasi-deterministic part consists of shorter partials that are generally unstable in frequency (short chirps), have lower amplitude values, and surround the harmonic or inharmonic partials of the deterministic part. Such partials may also occur as artifacts of the sinusoidal analysis algorithm, especially in cases where the sinusoids are of low amplitude and the tracking algorithm fails to perform a reliable peak-to-peak matching.

A one-to-one correspondence between the partials of the source and target sounds is very unlikely to occur unless we limit the number of tracked partials to the most prominent ones with respect to their durations and amplitude thresholds. However, there are

Table 1: *A brief comparison of methods for static morphing.*

Author(s) and papers	**Sound model & morphing strategy**	**Partial matching**	**High-level audio features**
Osaka [5]	Sinusoidal modeling. The residual is modeled with short partials according to hypothesized distributions.	Yes	No
Tellman et al. [4]	Sinusoidal modeling. No treatment of the residual.	Yes	No
Haken et al. [7]	Noise-enhanced sinusoidal modeling.	No	Amplitude and fundamental frequency
Boccardi and Drioli [8]	GMM applied to SMS. No treatment of the residual.	No	No
Caetano and Rodet [22]	Spectral envelopes for the deterministic and stochastic parts.	No	Spectral audio descriptors
Röbel [23]	Dynamical systems.	Not applicable	No
Ahmad et al. [24]	DWT with SVD.	Not applicable	No
Olivero et al. [25]	Gabor transform with constrained Gabor masks.	Not applicable	Arithmetic, harmonic and geometric centroids
Kazazis et al. [present document]	Sinusoidal modeling and spectral envelopes.	Yes	Spectral and harmonic audio descriptors

cases in which even if there is a limit to the number of tracked partials, the assumption of a one-to-one correspondence as described in [21] could be problematic. For example, when morphing from a sound that has odd and even harmonics to a sound that has only odd ones, we would ideally interpolate only the frequency and amplitude values of the odd harmonics of the two sounds to avoid the artifacts that would result from interpolating the odd with both the odd and even harmonics of the two sounds.

For finding correspondences between the partials of the source and target sounds, we use a k-nearest neighbors classifier (k-NN) based on Euclidean frequency proximity, and under the condition that the vector that is to be classified must have the same or a smaller number of partials. Obviously, the k-NN classifier does not return a one-to-one, but rather a many-to-one, mapping, so we choose the closest neighbor in frequency, and we treat the rest of the neighbors as unmatched partials. The unmatched partials retain their original frequencies but are initialized with zero amplitude levels, which gradually increase according to the morphing factor. After experimenting with different sounds, we concluded that such treatment does not lead to perceptual stream segregation, but rather to a seamless partial fade-in effect that facilitates the morphing between inharmonic sounds or between sounds that consist of unequal numbers of partials (see Fig. 1).

3.1.2. Interpolation of partials' breakpoint values

We represent each partial according to its start and end times, and with time breakpoints that are set according to its frequency and amplitude variations. If the source and target sounds have a different number of breakpoints, we simply interpolate the breakpoint values of the shorter one in order to match the number of breakpoints of the longer one. This representation enables us to interpolate the parameters at the level of events, which offers greater control over the morphing process as opposed to parameter interpolation between time frames. If the partials of the source and target sounds differ in duration, we are able to achieve intermediate durations by interpolating the breakpoint values of each partial according to the morphing factor. Interpolating between the start and end times of the partials also allows us to morph their onset asynchrony. We use the following expressions for calculating the interpolated values of partials' frequencies and amplitudes, respectively:

$$f(\alpha_\mathrm{p}) = \alpha_\mathrm{p} f_\mathrm{s} + (1-\alpha_\mathrm{p}) f_\mathrm{t} \tag{1}$$

$$\log_{10}(g(\alpha_\mathrm{p})) = \alpha_\mathrm{p} \log_{10}(g_\mathrm{s}) + (1-\alpha_\mathrm{p}) \log_{10}(g_\mathrm{t}) \tag{2}$$

where the subscripts "s", "t" denote the source and target, respectively, and α_p is the morphing factor related to parameter interpolation. Though Fig. 1 does not show a typical harmonic spectrum of the analyzed sounds because of the very low amplitude detection threshold ($-90\,\mathrm{dB}$) that was used in the partial-tracking algorithm, and which subsequently gave rise to auxiliary harmonic components, it clearly illustrates the estimation of partial-to-partial correspondences and the interpolation of the partials' breakpoint values.

3.2. Stochastic part

For morphing the stochastic part, we first estimate for every analysis frame its spectral envelope using Linear Predictive Coding (LPC), because we assume that the modeled signal is random, which fits exactly the basic assumption of LPC. We then get a temporal sequence of spectral envelopes (one for each frame), which allows us to render a time-varying Power Spectral Density (PSD) of the stochastic part. In order to morph, we interpolate for each time between the spectral envelope of the source and the target at this corresponding time. For a high-quality interpolation of the spectral envelopes, it is necessary to convert the LPC transverse coefficients to an alternative representation, because they do not interpolate well and might lead to unstable filters. Line Spectral Frequencies (LSF), Reflection Coefficients (RC) and Log Area Ratio (LAR) have been shown to interpolate smoothly, lead to stable intermediate filters, and lead to linear variations of audio descriptors

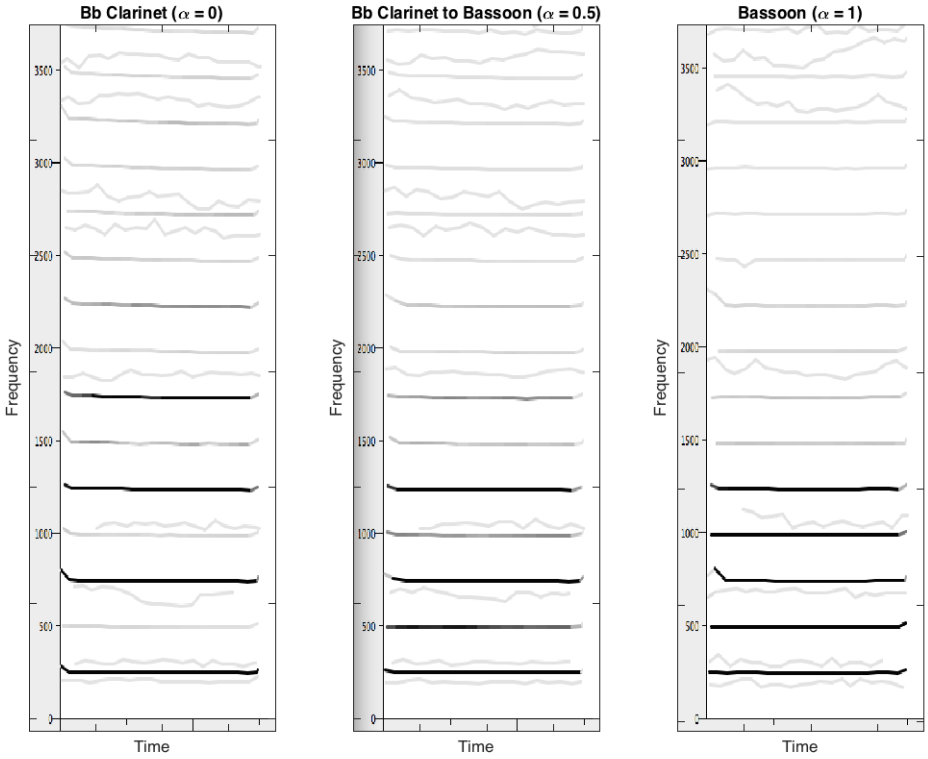


Figure 1: *Partial-to-partial correspondences and parameter interpolation of the deterministic part. Morphing from a clarinet sound to a bassoon with $\alpha_p = 0.5$. Gray-level values correspond to the partials' amplitude values.*

when linearly interpolated [21, 29]. We choose to interpolate the LAR coefficients (Eq. 3) as they both guarantee the filter's stability and have a physical interpretation, which could be specifically useful when trying to morph between sounds that were created by physical modeling synthesis as in [5, 2]. The filters' coefficients are interpolated according to Eq. (2).

$$\text{lar}(r_\alpha) = \alpha_\text{p}\text{lar}(r_\text{s}) + (1 - \alpha_\text{p})\text{lar}(r_\text{t}) \tag{3}$$

where lar is a vector the coefficients of which read:

$$\text{lar}(r)[i] = ln\left(\frac{1 - r(i)}{1 + r(i)}\right), \quad 1 \leq i \leq n \tag{4}$$

and n is the number of reflection coefficients r. The morphed residual is synthesized by filtered white noise after the inversion of the LAR coefficients to LPC coefficients.

3.3. Temporal Energy Envelope

In the present approach, the temporal energy envelope is a consequence of morphing. The parts of the signal that were morphed independently are added together to form the parameter-interpolated signal and thus, the energy envelope is constructed from the time-varying amplitudes of the partials and the gains of the filter.

4. FEATURE INTERPOLATION

The desired values of descriptors along with the interpolated spectrum form an underdetermined system because in theory there are an infinite number of sounds that have the same audio descriptor values. As previously described in Section 2, the target may consist only of target descriptor values D_a, in which case the morphing is based exclusively on high-level features. Fig. 2 shows an example of two sounds exchanging time-varying spectral centroids, where $\alpha_\text{p} = 0$, since the source is the Timpani without any parameter-based morphing, and $\alpha_\text{d} = 1$, because we apply to the source spectrum the spectral centroid values of the Tuba, which is the target. For each time frame, we match the audio descriptor values obtained according to a specific α_d to the interpolated spectrum by optimizing the amplitudes of the sinusoids or FFT bins of the interpolated spectrum x_j under the constraints of the target values of descriptors D_a. More formally this can be expressed as:

$$\min_x \sum_{j=1}^{N} |x_j - g_j| \quad \text{subject to} \quad D(x) = D_a \tag{5}$$

where g_j are the parameter-interpolated amplitude values according to α_p, N is the total number of partials or FFT bins, and D_a is the target value of $D(x)$, which can be one of the following descriptors (Eq. (6) – (11)).

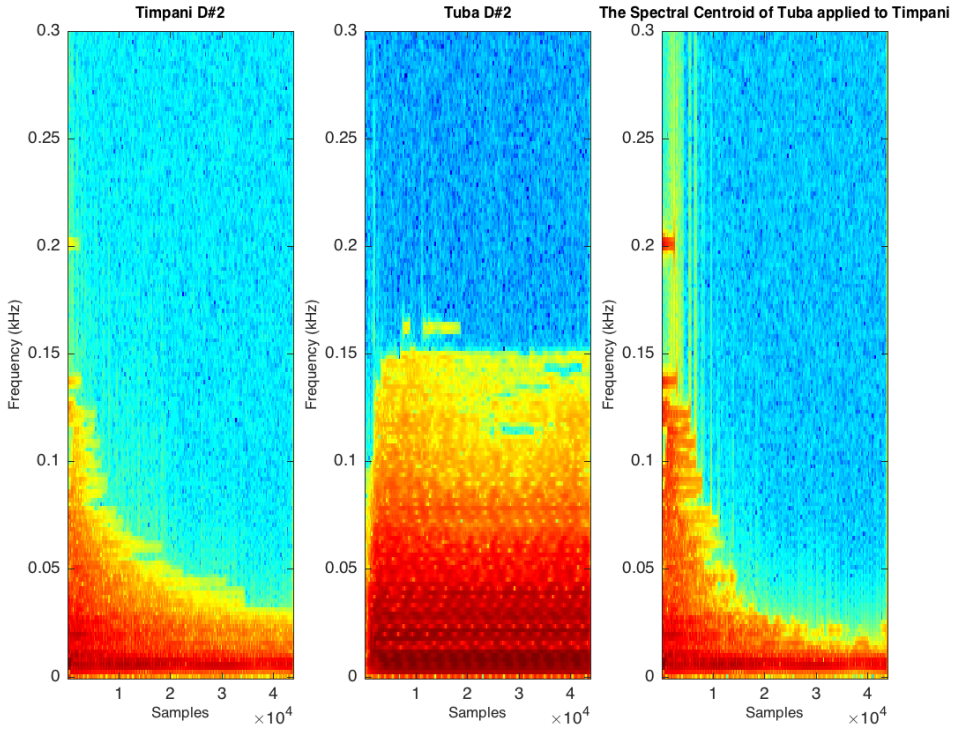


Figure 2: *The spectral centroid time series of a Tuba sound applied to a Timpani (the actual values of the time series are shown in Fig. 3.)*

$$m_1 = \sum_{j=1}^{N} f_j \cdot p_j \tag{6}$$

$$m_2 = (\sum_{j=1}^{N} (f_j - m_1)^2 \cdot p_j)^{1/2} \tag{7}$$

$$m_3 = (\sum_{j=1}^{N} (f_j - m_1)^3 \cdot p_j)/m_2^3 \tag{8}$$

$$m_4 = (\sum_{j=1}^{N} (f_j - m_1)^4 \cdot p_j)/m_2^4 \tag{9}$$

$$decr = \frac{1}{\sum_{j=2}^{N} x_j} \sum_{j=2}^{N} \frac{x_j - x_1}{j-1} \tag{10}$$

$$dev = \frac{1}{N} \sum_{j=1}^{N} (x_j - \mathrm{SE}(f_j)) \tag{11}$$

where p_j are the normalized values of x_j [27]:

$$p_j = \frac{x_j}{\sum_{j=1}^{N} x_j} \tag{12}$$

dev denotes the harmonic spectral deviation and $\mathrm{SE}(f_j)$ is the value of the spectral envelope at frequency f_j, which is estimated by averaging the values of three adjacent partials; $decr$ denotes the spectral decrease; m_1, m_2, m_3 and m_4 denote the spectral centroid, spectral spread, spectral skewness and spectral kurtosis respectively. The optimization is run in Matlab using the "fmincon" function along with the "sqp" method, which are suitable for solving constrained and non-linear problems [30]. Since the audio descriptors have different ranges, it is necessary to normalize them for assessing the convergence of the algorithm. Using this optimization scheme, we are able to set different morphing factors for each descriptor independently, as long as a feasible solution among these values exists. Furthermore, the choice of the objective function (Eq. 5) forces the optimized spectrum to be as close as possible to the interpolated one by keeping its frequency content unchanged and by altering its amplitude values as little as possible. Fig. 3 shows an example of morphing the parameter-interpolated signal according to varying morphing values of spectral centroid and spectral spread while preserving a constant value for the rest. Using a sinusoidal model for the deterministic and quasi-deterministic parts, the optimized values correspond directly to the parameters of additive synthesis, and the residual reaches its target values by altering the energy of the FFT bins. As in Section 3.1.2, if the source and target sounds are of different durations, we simply interpolate the descriptor values of the shorter one in order to match them to the analysis frames of the longer one.

5. CONCLUSIONS AND FUTURE WORK

We presented a hybrid approach to sound morphing based on sinusoid-plus-noise modeling and higher-level audio features. Dividing the signal into deterministic, quasi-deterministic, and stochastic parts and processing them separately allows for finer control of

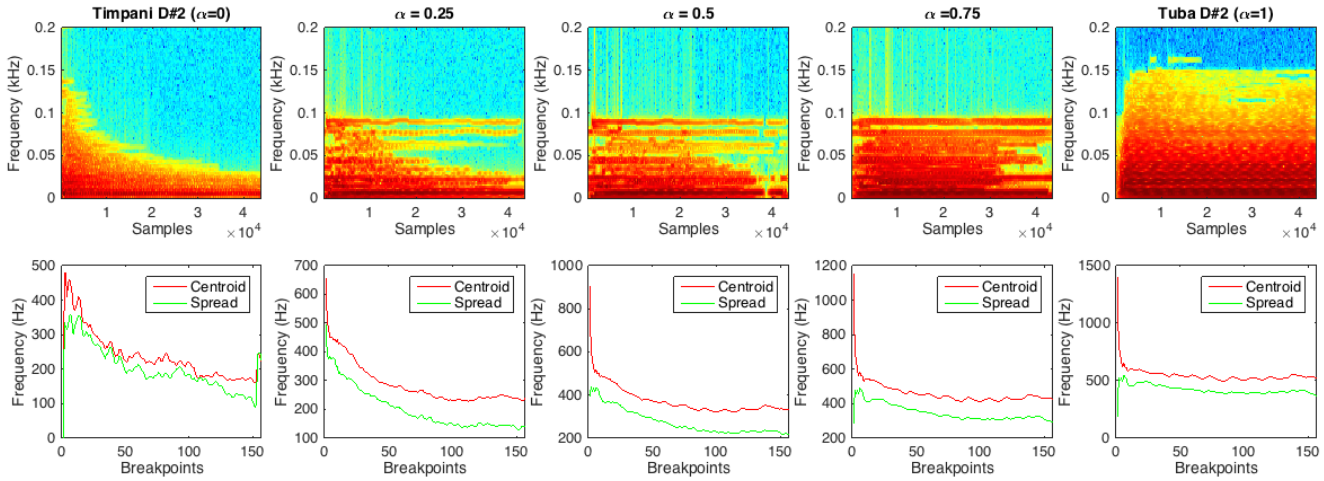


Figure 3: *Morphing the parameter interpolated signal by audio descriptors. Spectral centroid and spectral spread vary according to the morphing factor α. The rest of descriptors preserve constant target values according to their median when interpolated with $\alpha_d = 0.5$.*

the synthesis parameters and also enables us to morph between deterministic and quasi-deterministic signals of different durations. The morphed sound is synthesized using additive synthesis for the deterministic and quasi-deterministic parts and filtered white noise for the stochastic part. The spectrum of the morphed signal is further refined according to target audio descriptor values through an optimization process. We have shown that this process allows us to control accurately and independently several audio features, provided that a feasible solution among them exists. Audio examples are available at: https://www.mcgill.ca/mpcl/resources-0/supplementary-materials. The proposed scheme is more suitable for sustained and percussive sounds, which can either be harmonic or inharmonic, rather than textural sounds. Their residuals however, should be stationary (or pseudo-stationary) as opposed to sound texture, the residual of which is usually non-stationary and may consist of sharp transients. A refinement of our approach would be to find sophisticated ways to interpolate between different tremolo and vibrato rates while preserving the overall spectrotemporal complexity of the partials. Finally, we by no means claim that the use of high-level audio features enables a perceptually based sound morphing. Rather, it offers a more intuitive control over the morphing process, as in the case of adaptive effects [31]. Up to now only spectral centroid and log-attack time have been shown to be significantly correlated with perceptual dimensions, cf. [1, 32]. If and how such audio features collapse to single perceptual dimensions remains to be empirically determined.

6. ACKNOWLEDGMENTS

We would like to thank Philippe Esling for his advice on improving the computational efficiency of our optimization procedure, and the four anonymous reviewers for their helpful comments about improving the clarity of this document. This work was funded by Canadian Natural Sciences and Engineering Research Council grants awarded to Stephen McAdams and Philippe Depalle and a Canada Research Chair awarded to Stephen McAdams.

7. REFERENCES

[1] A. Caclin, S. McAdams, B.K. Smith, and S. Winsberg, "Acoustic Correlates of Timbre Space Dimensions: A Confirmatory Study Using Synthetic Tones," *J. Acoust. Soc. Am.,* 118 (1), pp. 471-482, 2005.

[2] S. Lakatos, P. Cook, and G. Scavone, "Selective attention to the parameters of a physically informed sonic model," *Acoustics Research Letters Online, J. Acoust. Soc. Am.,* March 2000.

[3] M. Caetano, *Morphing isolated quasi harmonic acoustic musical instrument sounds guided by perceptually motivated features,* PhD Thesis , IRCAM, Université Pierre et Marie Curie, Paris, France. June 2011.

[4] E. Tellman, L. Haken, and B. Holloway, "Timbre Morphing of Sounds with Unequal Numbers of Features," *J. Audio Eng. Soc.,* vol. 43, no. 9, pp 678-689, September, 1995.

[5] N. Osaka, "Timbre Interpolation of Sounds Using a Sinusoidal Model," in *Proc. ICMC,* Banff Centre for the Arts, Canada, 1995.

[6] K. Fitz, L. Haken, S. Lefvert, C. Champion, and M. O'Donnell, "Cell-Utes and Flutter-Tongued Cats: Sound Morphing Using Loris and the Reassigned Bandwidth-Enhanced Model," *Computer Music Journal,* 27 (3), 2003.

[7] L. Haken, K. Fitz, and P. Christensen, *Sound of Music: Analysis, Synthesis, and Perception,* "Beyond Traditional Sampling Synthesis: Real-Time Timbre Morphing Using Additive Synthesis," J. W. Beauchamp Ed. Springer-Verlag, Berlin, 2006.

[8] F. Boccardi, and C. Drioli, " Sound Morphing with Gaussian Mixture Models," in *Proc. DAFx,* Limerick, Ireland, 2001.

[9] K. Fitz, L. Haken, and P. Christensen, "A New Algorithm for Bandwidth Association in Bandwidth Enhanced Additive Sound Modeling," in *Proc. ICMC*, Havana, Cuba, 2001.

[10] X. Serra and J. I. Smith, "Spectral Modelling Synthesis," *Computer Music Journal,* 14 (4),1990.

[11] M. Slaney, M. Covell, and B. Lassiter, "Automatic Audio Morphing," in *Proc. ICASSP,* Atlanta, Georgia, 1996.

[12] T. Ezzat, E. Meyers, J. Glass, and T. Poggio, "Morphing Spectral Envelopes using Audio Flow," in *Proc. ICASSP,* Philadelphia, Pennsylvania, 2005.

[13] M. Hoffman,and P. Cook, "Feature-based Synthesis: Mapping from Acoustic and Perceptual Features to Synthesis Parameters" in *Proc. ICMC,* New Orleans, USA, 2006.

[14] T. Park, J. Biguenet, Z. Li, R. Conner, and S. Travis, "Feature Modulation Synthesis," in *Proc. ICMC,* Copenhagen, Denmark, 2007.

[15] D. Mintz, *Toward Timbral Synthesis: a New Method for Synthesizing Sound Based on Timbre Description Schemes,* Master's thesis, Univ. Cal, 2007.

[16] D. Williams, and T. Brookes, "Perceptually-Motivated Audio Morphing: Warmth," *AES 128th Convention,* London, UK, 2010.

[17] D.Williams, P. Randall-Page, E. R. Miranda, "Timbre morphing: near real-time hybrid synthesis in a musical installation," in *Proc. NIME,* London, UK, 2014.

[18] T. Hikichi and N. Osaka, "Sound Timbre Interpolation Based on Physical Modeling" *Acoustical Science and Technology,* 22 (2), 2001.

[19] A. Primavera, F. Piazza and J. D. Reiss, "Audio Morphing for Percussive Hybrid Sound Generation," *AES 45th Conference on Applications of Time Frequency Processing in Audio*, Helsinki, March 2012.

[20] G. Coleman, and J. Bonada, "Sound Transformation by Descriptor Using an Analytic Domain," in *Proc. DAFx,* Espoo, Finland, 2008.

[21] M. Caetano and X. Rodet, "Automatic Timbral Morphing of Musical Instrument Sounds by High-Level Descriptors," in *Proc. ICMC,* New York, USA, 2010.

[22] M. Caetano and X. Rodet, "Independent Manipulation of High-Level Spectral Envelope Shape Features for Sound Morphing by Means of Evolutionary Computation," in *Proc. DAFx,* Graz, Austria, 2010.

[23] A. Röbel, "Morphing Dynamical Sound Models," in *Proc. IEEE Workshop Neural Net Sig. Proc,* 1998.

[24] M. Ahmad, H. Hacihabiboglu, and A. M. Kondoz, "Morphing of Transient Sounds Based on Shift-Invariant Discrete Wavelet Transform and Singular Value Decomposition," in *Proc. ICASSP,* Taipei, Taiwan, 2009.

[25] A. Olivero, P. Depalle, B. Torresáni and R. Kronland-Martinet, "Sound Morphing Strategies Based on Alterations of Time-Frequency Representations by Gabor Multipliers", *AES 45th Conference, Applications of Time-Frequency Processing in Audio,* Helsinki, Finland, March 2012.

[26] G. Peeters, B. L. Giordano, P. Susini, N. Misdariis, and S. McAdams, "The Timbre Toolbox: Extracting Audio Descriptors from Musical Signals," *J. Acoust. Soc. Am.,* 130, pp. 2902-2916, 2011.

[27] A. Röbel, and X.Rodet, "Efficient Spectral Envelope Estimation and its Application to Pitch Shifting and Envelope Preservation," in *Proc. DAFx,* Madrid, Spain, 2005.

[28] T. Galas, and X. Rodet, "An Improved Cepstral Method for Deconvolution of Source-Filter Systems with Discrete Spectra: Application to Musical Sounds," in *Proc. ICMC,* Glasgow, Scotland, 1990.

[29] T. Islam, *Interpolation of Linear Prediction Coefficients for Speech Coding,* Master's thesis, McGill University, 2010.

[30] http://www.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html#bsgppl4/

[31] V. Verfaille and P. Depalle, "Adaptive Effects based on STFT, using a Source-Filter model," in *Proc. DAFx*, Naples, Italy, 2004.

[32] C. W. Wun, A. Horner, and Bin. Wu, "Effect of Spectral Centroid Manipulation on Discrimination and Identification of Instrument Timbres," *J. Audio Eng. Soc.,* 62(9), 575-583, 2014.

REAL-TIME FORCE-BASED SOUND SYNTHESIS USING GPU PARALLEL COMPUTING

Ryoho Kobayashi

Faculty of Environment and Information Studies,
Keio University SFC
Kanagawa, Japan
ryoho@sfc.keio.ac.jp

ABSTRACT

In this paper we propose a real-time sound synthesis method using a force-based algorithm to control sinusoidal partials. This synthesis method can generate various sounds from musical tones and noises with three kinds of intuitive parameters, which are attractive force, repulsive force and resistance. However, the implementation of this method in real-time has difficulties due to a large volume of calculations for manipulating thousands or more partials. In order to resolve these difficulties, we utilize a GPU-based parallel computing technology and precalculations. Since GPUs allowed us to implement powerful simultaneous parallel processing, this synthesis method is made more efficient by using GPUs. Furthermore, by using familiar musical features, which include MIDI input for playing the synthesizer and ADSR envelope generators for time-varying parameters, an intuitive controller for this synthesis method is accomplished.

1. INTRODUCTION

A force-based sound synthesis [1] is an application of sinusoidal partial editing techniques. This synthesis method can generate various sounds from musical tones and noises with three kinds of intuitive parameters, which are attractive forces to a reference spectrum, repulsive forces between partials, and resistances for decreasing the speeds of the partials. However, this method requires a large volume of calculations due to a need of some thousands or more partials to control.

By recent developments of graphics processing units (GPUs) [2] and APIs for handling these processors, an implementation of an efficient simultaneous parallel processing has been realized. GPUs are originally developed for computer graphics and visual image processing, however, they have begun to be used for general purpose due to a popularization of general-purpose computing on graphics processing units (GPGPU) [3]. And they recently have been utilized for sound processing and syntheses [4, 5, 6]. A GPGPU framework is considered of value for a acceleration of force-based sound synthesis, because this synthesis method needs many simple calculations simultaneously.

In this paper we propose a process to accomplish a real-time sound synthesis using a force-based algorithm by utilizing a GPU-based parallel computing technique. All programs presented in this paper are written in Swift [7] and use Metal API [8] for GPU processing.

2. FORCE-BASED SOUND SYNTHESIS

The sound synthesis method presented in this paper uses a force-based algorithm, which is commonly known as a graph drawing algorithm [9, 10]. The fundamental process for the synthesis is described in this section.

2.1. Analysis of a Reference Sound Source

The first step is the analysis of a reference sound source. The Short-Time Fourier Transform (STFT) analysis [11] is used for this step, where amplitudes $A(k)$ for each frequency $F(k)$ are detected by

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-2\pi i k n / N} \tag{1}$$

$$A(k) = |X(k)| \tag{2}$$

$$F(k) = \frac{kR}{N} \tag{3}$$

where $x(n)$ consists of N samples of a windowed waveform and R represents the sampling rate.

2.2. Distribution of Partials

The synthesis phase for the proposed method begins by generating partials in a specific range of frequencies. The amplitudes of the partials are unchangeable. The user specifies the maximum number of partials and the number of active partials is determined in proportion to the amplitude of the reference sound as (4).

$$\nu = \nu_{max} \sum_{k=0}^{N-1} \alpha A(k) \tag{4}$$

In this equation, α represents a constant number for scaling the amplitude. The active or inactive partials are randomly chosen. In this step, since the frequencies of the partials are random, an unpitched sound is typically created.

2.3. Attractive Force

Attractive forces, which are applied to the partials, are generated from the spectrum detected from the reference sound. A partial is attracted to neighboring frequency components. The force is inversely proportional to the square of the distance between the target frequency component and the frequency of the partial.

$$f_a(P(i)) = \sum_{0<|F(k)-P_F(i)|<\tau} \frac{\mathrm{sgn}(F(k) - P_F(i)) g_a A(k)}{|F(k) - P_F(i)|^2} \tag{5}$$

$f_a(P(i))$ represents an attractive force for partial $P(i)$ of which the frequency is $P_F(i)$, g_a is a constant value to adjust the strength of the force, and τ corresponds to the range of the effective frequency components.

2.4. Repulsive Force

To avoid congestion of partials at a small peak in the spectrum, repulsive forces are generated between every pair of partials. The force is inversely proportional to the square of the distance between the partials.

$$f_r(P(i)) = \sum_{P_F(j) \neq P_F(i)} \frac{\mathrm{sgn}(P_F(i) - P_F(j)) g_r}{|P_F(i) - P_F(j)|^2} \tag{6}$$

$f_r(P(i))$ represents a repulsive force for partial $P(i)$. By using all pairs of partials for the calculation, partials depart from condensations.

2.5. Resistance

When the reference sound has a static frequency component, the partials have the risk of periodic vibration around a spectral peak. This is because the attractive forces convert back and forth between potential and kinetic energy. Therefore, the oscillations are inhibited by implementing resistance.

$$f(P(n,i)) = f_a(P(n,i)) + f_r(P(n,i)) \tag{7}$$

$$v(P(n,i)) = r(v(P(n-1,i)) + f(P(n-1,i))) \tag{8}$$

In this equation, $v(P(n,i))$ represents a current speed for partial $P(i)$, n is the current time frame, and r is a resistance value between 0 and 1.

2.6. Synthesis

The forces, which are derived above, are applied to partials at every frame by addition of the forces.

$$P_F(n,i) = P_F(n-1,i) + v(P(n-1,i)) \tag{9}$$

The sound synthesis is accomplished using a common oscillator bank synthesis technique [12, 13] which is realized by

$$y(t) = \sum_{\forall i} A \cos[2\pi P_F(n,i) t + \phi_i] \tag{10}$$

where A represents a constant amplitude for each partial, and ϕ_i is an initial phase.

3. GPU-BASED PARALLEL COMPUTING

3.1. General-purpose computing on graphics processing units

General-purpose computing on graphics processing units (GPGPU) refers to the use of GPUs for general purpose parallel computing, and performs computation in applications traditionally handled by the central processing unit (CPU), outside of computer graphics and image processing. GPU has ability to process multiple tasks simultaneously by having thousands of cores. OpenCL [14] and NVidia's CUDA [15] are two popular frameworks to implement general purpose computations on GPUs. The synthesis method this paper proposes uses Apple's Metal API [8] for low-overhead access to the GPUs.

3.2. Metal API

Metal [8] is a graphics application programming interface (API), which allows low-level and low-overhead access to GPUs. Metal is developed and provided by Apple, and is available to use on iOS and Mac OS X. The previous implementation of this synthesis method was written in Objective-C and ran on Mac OS X. The real-time version, which is proposed in this paper, is rewritten in Swift, and is still implemented for Mac OS, thus, Apple's Metal API is expected to deliver high performance and has good prospects for the future. Furthermore, further possibilities of implementation for mobile devices are expected, because Metal is compatible with iOS.

By using Metal framework, a low-overhead interface, a memory and resource management, integrated support for both graphics and compute operations, and precompiled shaders are provided. For the force-based sound synthesis, the handling of the current frequency, the resistance for speed, and the attractive/repulsive forces are required for acceleration to manipulate each partial component. Therefore, the current frequency and speed need to be stored until the calculations for the next time frame are performed.

4. IMPLEMENTATIONS

We present implementations of GPU-based parallel computing technologies and conceptions for the efficiencies of calculations for force-based sound synthesis.

4.1. Attractive and Repulsive Forces

The attractive force for a partial is proportional to the amplitude for a target frequency component, and inversely proportional to the square of the distance between the target frequency component and the frequency of the partial. The repulsive force is inversely proportional to the square of the distance between the partials.

These two forces are possible to calculate by using the convolutions of the functions below with the reference spectrum for the attractive forces and the spectrum of current partials for the repulsive forces.

$$h_a(F) = \frac{-\mathrm{sgn}(F) g_a}{F^2} \tag{11}$$

F is a frequency, g_a is a value to adjust the strength of the force. and $h_a(F)$ is a calculated function for attractive forces.

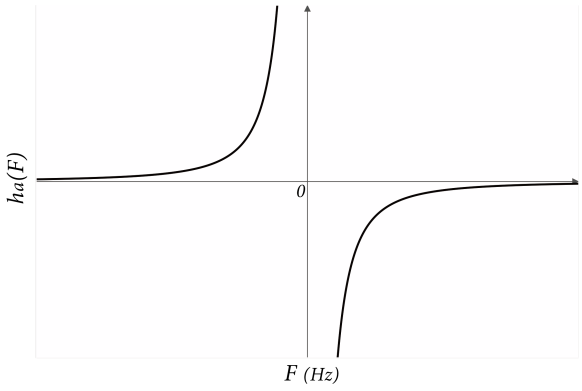


Figure 1: *Function for calculating attractive forces.*

$$h_r(F) = \frac{\mathrm{sgn}(F) g_r}{F^2} \tag{12}$$

F is a frequency, g_r is a value to adjust the strength of forces. and $h_r(F)$ is a calculated function for repulsive forces.

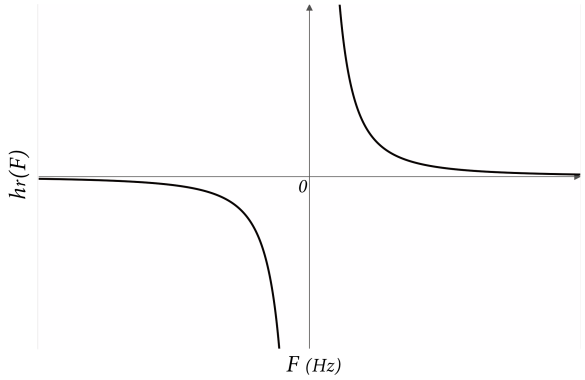


Figure 2: *Function for calculating repulsive forces.*

The summation of these forces are described the following equation 13.

$$\alpha(F) = h_a(F) * A_d(F) + h_r(F) * A_c(F) \tag{13}$$

where $\alpha(F)$ is a summation of the attractive and repulsive forces, $A_d(F)$ is an amplitude for a frequency F from a reference source, and $A_c(F)$ is an amplitude from current partials.

4.2. Updating of Frequencies of Partials

The speed of the frequency change $v(P(n,i))$ and the frequency of a partial $P_F(n,i)$ for the newer frame are calculated by the equasion 8 and 9 in section 2. Since these calculations are achieved by using simple summations and multiplications, parallel computing is applied without difficulties.

5. MIDI INPUT

To use the force-based synthesis method for a musical performance, MIDI note messages [16] are available to input. The fundamental frequency for a reference source spectrum is calculated from the "note number", and the maximum number of partials is proportional to the "velocity".

5.1. Preset Reference Spectra

Efficient computations are accomplished by using GPU-based parallel computing presented in previous sections, however, the volume of calculations is still large. Therefore, a preparation of reference spectral sources and precalculations of the attractive forces are valuable for the achievements of a stable real-time synthesis environment.

The force-based synthesis is possible to generate a variety of timbres by controlling the combination of attractive and repulsive forces and a resistance, thus, simple harmonic series are useful enough. By preparing the reference source, the attractive forces $h_a(F) * A_d(F)$ in equation 13 are also available to prepare.

5.2. ADSR Envelopes

This synthesis method can generate various sounds by adjusting the parameters. In particular, the coefficients for attractive force g_a in the equation 5, repulsive force g_r in the equation 6, and resistance r in the equation 8 are important for controlling the similarity to the reference sound and the quickness of transitions.

By implementing time-varying controls for these parameters, users can dynamically synthesize various sounds. The synthesis method allows users to assign attack time, decay time, sustain value, and release time (ADSR) functions, and minimum and maximum values to the three parameters. The ADSR envelope generator model is generally used in existing synthesizers through the ages, and familiar to musicians. These dynamic controls are simultaneously activated with MIDI note messages and adjusted with MIDI control messages.

5.3. Synthesized Result

In this section, we present a synthesized result of this method.

A spectrogram of a reference sound source for this example is made from sawtooth waves as shown in Figure 4. In this source sound, a fundamental pitch increase, and every overtones are contained.

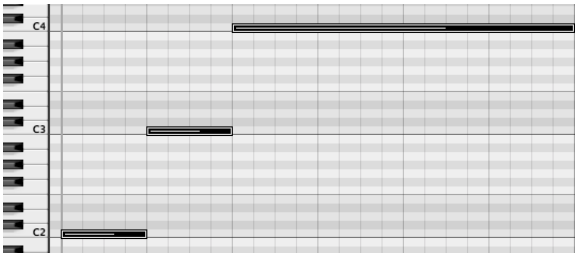


Figure 3: *Input MIDI notes.*

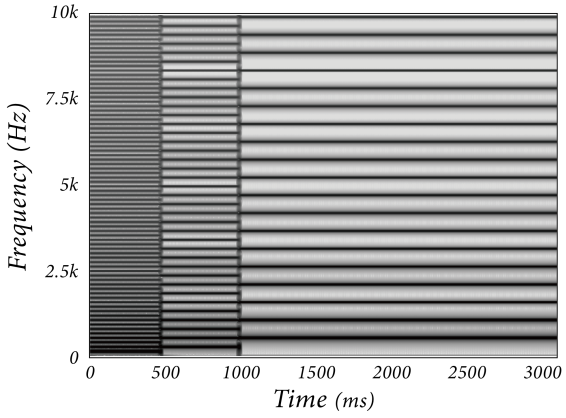


Figure 4: *Reference sound source (sawtooth wave).*

By applying ADSR envelopes in Figure 5 for 5000 partials, a result as shown in Figure 6 is generated. This result shows that a strong repulsive force at an attack makes noise and the partials are gradually attracted to the reference source. At the first note,

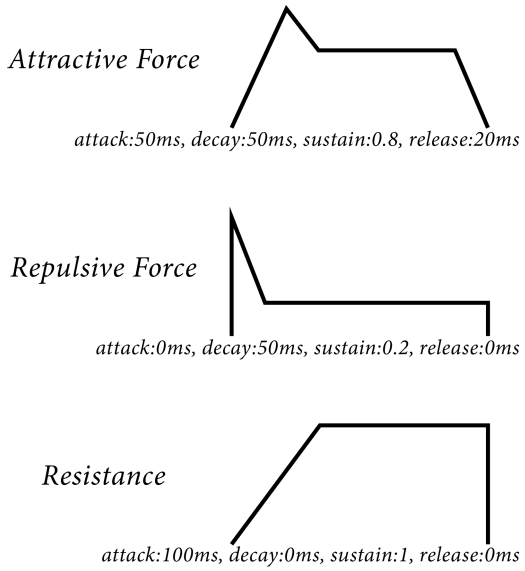


Figure 5: *ADSR envelopes.*

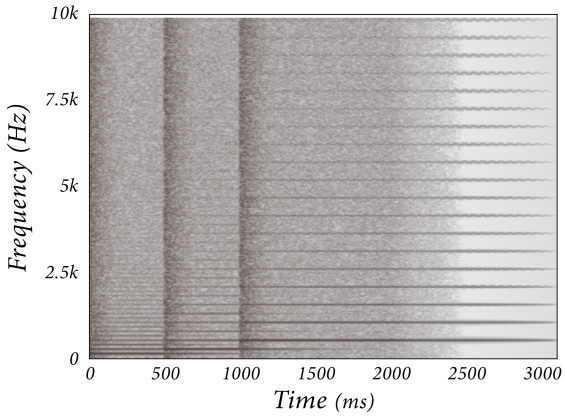


Figure 6: *Synthesized result.*

randomly distributed partials cannot converge on reference overtones during the short duration of 500ms. Since the second and third notes start from partials with density fluctuation, which are made by the previous note, they are easily attracted on the reference overtones.

6. CONCLUSIONS

In this paper, a real-time application for force-based sound synthesis, which is accomplished by using GPU-based parallel computing, is proposed. The utilization of GPUs is powerful and efficient for this synthesis method. Although it is difficult for off-the-shelf personal computers to generate high-quality results in real-time in this time, thus, some ideas for reducing the calculation cost are implemented. In this section, we provide the process for the real-time synthesis.

6.1. Preparations

1. Select one reference source sound which is previously detected frequency components
2. Calculate attractive forces from the reference spectrum
3. Setup the maximum number of partials and prepare the corresponding shader

6.2. Real-Time Processing

1. Receive MIDI note message
2. Scale the attractive force function corresponding to the note number.
3. Distribute partials corresponding to the velocity and ADSR function.
4. Calculate attractive and repulsive forces for each partial
5. Calculate speeds and frequencies for each partial
6. Store the new speeds and frequencies of the partials
7. Synthesize the sound for the current frame by using inverse fast Fourier transform (IFFT) [11]

6.3. Future works

The major limitation for this synthesis method is caused by the limit of the frame-rate, which depends on the capabilities of GPUs. GPUs are generally designed for manipulating visual data, therefore, the upper limit of the frame-rate is not enough for a high-quality sound synthesis. We consider that the development of the interpolation techniques between frames has value, though the improvement of these kind of processors are expected.

7. REFERENCES

[1] Ryoho Kobayashi, "Sinusoidal synthesis using a force-based algorithm," in *Proceedings of the 17th International Conference on Digital Audio Effect (DAFx-14)*, Erlangen, Germany, September 1-5 2014, pp. 19–22.

[2] John D. Owens, Mike Houston, David Luebke, Simon Green, John E. Stone, and James C. Phillips, "GPU computing," *Proceedings of the IEEE*, vol. 96, no. 5, pp. 879–899, 2008.

[3] Mark Harris, "GPGPU: General-purpose computation on GPUs," *SIGGRAPH 2005 GPGPU COURSE*, 2005.

[4] Lauri Savioja, Vesa Välimäki, and Julius O. Smith III, "Audio signal processing using graphics processing units," *Journal of the Audio Engineering Society*, vol. 59, no. 1/2, pp. 3–19, 2011.

[5] Lauri Savioja, Vesa Välimäki, and Julius O. Smith III, "Real-time additive synthesis with one million sinusoids using a GPU," *Audio Engineering Society Convention 128*, 2010.

[6] Pei-Yin Tsai, Tien-Ming Wang, and Alvin Su, "GPU-based spectral model synthesis for real-time sound rendering," in *Proceedings of the 13th International Conference on Digital Audio Effect (DAFx-10)*, Graz, Austria, September 6-10 2010, pp. 1–5.

[7] Apple Inc., "Swift - apple developer," Available at https://developer.apple.com/metal/, accessed March 15, 2016.

[8] Apple Inc., "Metal for developers," Available at https://developer.apple.com/swift/, accessed March 15, 2016.

[9] Thomas M. J. Fruchterman and Edward M. Reingold, "Graph drawing by force-directed placement," *Software: Practice and Experience*, vol. 21, no. 11, pp. 1129–1164, 1991.

[10] John Adrian Bondy and Uppaluri Siva Ramachanda Murty, *Graph Theory with Applications*, The Macmillan Press Ltd., London, 1976.

[11] Jont B. Allen, "Short term spectral analysis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Processing*, vol. 25, no. 3, pp. 235–238, 1977.

[12] G. DiGiugno, "A 256 digital oscillator bank," in *Proceedings of the International Computer Music Conference*, MIT, Cambridge, 1976, pp. 188–91.

[13] F. Richard Moore, *Elements of Computer Music*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1990.

[14] John E. Stone, David Gohara, and Guochun Shi, "OpenCL: A parallel programming standard for heterogeneous computing systems," *Computing in Science and Engineering*, vol. 12, no. 3, pp. 66–73, 2010.

[15] Jason Sanders and Edward Kandrot, CUDA *by Example: An Introduction to General-Purpose* GPU *Programming*, Addison-Wesley Professional, 2010.

[16] MIDI Manufacturers Association, *The Complete MIDI 1.0 Detailed Specification*, 1996.

[17] Pat Hanrahan and Jim Lawson, "A language for shading and lighting calculations," *ComputerGraphics*, vol. 24, pp. 289–295, 1990.

[18] Alécio P. D. Binotto, Joao L. D. Comba, and Carla M. D. Freitas, "Real-time volume rendering of time-varying data using a fragment-shader compression approach," in *IEEE Symposium on Parallel and Large-Data Visualization and Graphics*, 2003, pp. 69–75.

8. APPENDIX: SOUND EXAMPLES

Sound examples are available online at the following address.

http://www.ryoho.com/software/sinfba/

A PHYSICAL STRING MODEL WITH ADJUSTABLE BOUNDARY CONDITIONS

Maximilian Schäfer

Friedrich-Alexander Universität
Erlangen-Nürnberg (FAU)
Multimedia Communications
and Signal Processing
Cauerstr. 7, D-91058 Erlangen, Germany
max.schaefer@fau.de

Petr Frenštátský

Brno University of Technology
Faculty of Electrical Engineering and
Communication
Technická 3058/10, 616 00 Brno,
Czech Republic
petr.frenstatsky@
phd.feec.vutbr.cz

Rudolf Rabenstein

Friedrich-Alexander Universität
Erlangen-Nürnberg (FAU)
Multimedia Communications
and Signal Processing
Cauerstr. 7, D-91058 Erlangen, Germany
rudolf.rabenstein@fau.de

ABSTRACT

The vibration of strings in musical instruments depends not only on their geometry and material but also on their fixing at the ends of the string. In physical terms it is described by impedance boundary conditions. This contribution presents a functional transformation model for a vibrating string which is coupled to an external boundary circuit. Delay-free loops in the synthesis algorithm are avoided by a state-space formulation. The value of the boundary impedance can be adjusted without altering the core synthesis algorithm.

1. INTRODUCTION

The vibrations of strings, bars and other sound generating objects in musical instruments are a well studied subject. Based on the fundamental laws of elasticity, their dynamic behavior is accurately described by differential equations of different kinds.

There is a variety of methods to turn these differential equations into computational models by discretization in time and space or by transformation into the respective frequency domains. The procedure of deriving a real-time algorithm from a physical description of parts of a musical instrument is called physical modeling sound synthesis.

The dynamics of a vibrating body depend not only on properties like shape and material but also on the contact conditions to other parts of the musical instrument or the hands of the musician. These are relatively easy to model if the ends of a string are fixed by frets or bridges or if the ends of a bar in a xylophone are free from external forces.

Other types of contact conditions need more careful consideration. The touch and the movement of the musicians fingers can be modelled only approximately with mathematical equations. The interaction between the strings, the bridge, and the body of a musical instrument requires careful measurement of parameters like body resonances and bridge impedances. Since there is an abundance of research in this field, only a few books and overview articles with extensive references can be addressed here.

The corresponding boundary conditions are described in [1, Chap. 2.12] by conditions for the deflection (fixed end) or its first order space derivative (free end) or by complex impedances at the boundaries. Also [2, Chap. 6.1.9] discusses not only the simple cases of fixed and free ends but also lossy boundary conditions. In finite difference models, the boundary conditions are often formulated after spatial discretization by involving virtual spatial sample points beyond the boundary [2, 3]. Waveguide methods model the influence of the boundary on the reflection of waves by reflection factors or reflection filters (bridge filters) [3, 4]. The string-bridge interaction is described by a mechanical impedance (admittance) in [5].

A particular physical modeling technique is the functional transformation method [6]. It is based on the spatial eigenfunctions of vibrating bodies, which are most easily determined for fixed and for free ends. However [7] discusses also impedance boundary conditions for frequency independent impedances. The more involved case of frequency dependent impedances (typically bridge impedances) is discussed in [8] by incorporating the boundary impedance into the synthesis algorithm.

A conceptually simpler approach has been presented in general terms in [9]. There the synthesis algorithm is kept separate from the boundary model. This way, the impedance in the boundary model can be adjusted during operation without affecting the structure of the string model. The approach is motivated by the plant-controller loop familiar from control theory.

How these adjustable boundary conditions can be applied to a functional transformation model of a string is shown here in detail. Sec. 2 presents the physical model of a string and the spatial transformation is reviewed in Sec. 3. Boundary conditions are discussed in Sec. 4 with the impedance as an adjustable parameter. The occurrence of delay-free loops can be avoided by a state space approach in Sec. 5. Examples show the effect of the resulting synthesis algorithm in Sec. 6. Although the results shown here involve a frequency independent impedance, the state space approach can be extended also to frequency dependent impedances as discussed in Sec. 7.

2. PHYSICAL MODEL OF A STRING

This section describes the physical foundations of a single vibrating string. The partial differential equation describing the oscillation of a string is reformulated to serve as a starting point for the following Functional-Transformation method (FTM). The presentation is an abridged and modified version of the approach described in [10].

2.1. Physical Description

The starting point for a computational model is the partial differential equation (PDE) of a single vibrating string [10, 11]. The deflection $y = y(x, t)$ depends on the position on the string x and time t, $\dot{y}$ represents the time- and y' the space-derivative. Then the PDE of a single vibrating string is given as

$$\rho A\ddot{y} + EIy'''' - T_\mathrm{s}y'' + d_1\dot{y} - d_3\dot{y}'' = f_\mathrm{e}, \qquad (1)$$

with the cross-section area A, moment of inertia I and the length l. The material is characterized by the density ρ and Young's modulus E. T_s describes the tension and d_1 and d_3 are non-frequency and frequency dependent damping [11]. The excitation function is defined as $f_\mathrm{e} = f_\mathrm{e}(x,t)$.

The PDE (1) is reformulated into a vectorized form for subsequent transformations

$$\left[\boldsymbol{C}\frac{\partial}{\partial t} - L\right]\boldsymbol{y}(x,t) = \boldsymbol{f}_\mathrm{e}(x,t), \tag{2}$$

with the differential operator

$$L = \boldsymbol{A} + \boldsymbol{I}\frac{\partial}{\partial x} \tag{3}$$

and the vector of variables

$$\boldsymbol{y}(x,t) = \begin{bmatrix} \dot{y} & y' & y'' & y''' \end{bmatrix}^\mathrm{T}. \tag{4}$$

The system matrices of Eq. (2) are

$$\boldsymbol{C} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ c_1 & 0 & c_2 & 0 \end{bmatrix} \quad \boldsymbol{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ a_1 & 0 & a_2 & 0 \end{bmatrix} \tag{5}$$

with the coefficients

$$a_1 = \frac{d_1}{EI} \quad a_2 = -\frac{T_\mathrm{s}}{EI} \quad c_1 = -\frac{\rho A}{EI} \quad c_2 = \frac{d_3}{EI}. \tag{6}$$

The equivalence between the scalar and the vector representation follows by converting (2) back to the scalar form (1).

2.2. Boundary Conditions

The behavior of a vibrating string depends also on a set of boundary conditions in addition to the PDE (1). They can be described using a boundary matrix $\boldsymbol{F}_\mathrm{b}^\mathrm{H}$, which transforms the vector of variables $\boldsymbol{y}(x,t)$ into a vector of boundary excitations $\boldsymbol{\phi}(x,t)$

$$\boldsymbol{F}_\mathrm{b}^\mathrm{H}\boldsymbol{y}(x,t) = \boldsymbol{\phi}(x,t) \qquad x = 0, l. \tag{7}$$

The superscript H denotes the hermitian matrix. In a first simple case it is assumed, that there are unknown boundary excitations ϕ for deflection and bending moment at the boundaries of the string

$$y(x,t) = \phi_1(x,t) \qquad x = 0, l, \tag{8}$$

$$y''(x,t) = \phi_3(x,t) \qquad x = 0, l. \tag{9}$$

Since the vector of variables in Eq. (4) contains the first time derivative, Eq. (8) is turned into $\dot{y}(x,t) = \dot{\phi}_1(x,t)$ by time differentiation, such that

$$\boldsymbol{F}_\mathrm{b}^\mathrm{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{\phi}(x,t) = \begin{bmatrix} \dot{\phi}_1(x,t) \\ 0 \\ \phi_3(x,t) \\ 0 \end{bmatrix}. \tag{10}$$

2.3. Laplace Transformation

For the following transformation of the string model a Laplace transformation has to be applied to the vectorized PDE (2) and to the boundary conditions (7), which leads to

$$\left[s\boldsymbol{C} - L\right]\boldsymbol{Y}(x,s) = \boldsymbol{F}_\mathrm{e}(x,s) \tag{11}$$

$$\boldsymbol{F}_\mathrm{b}^\mathrm{H}\boldsymbol{Y}(x,s) = \boldsymbol{\Phi}(x,s). \tag{12}$$

The complex frequency variable is s and the Laplace transforms of the time variables are denoted by uppercase letters. This model is used for the further derivations of the FTM.

3. TRANSFORMATION OF THE STRING MODEL

In this section the FTM is applied to the string model. The foundations of the transformation are described in [6, 10]. Therefore, some of the derivations regarding the transformation are skipped in the following.

3.1. Sturm-Liouville Transformation

The first step is the definition of a forward and inverse Sturm-Liouville Transformation (SLT). The forward transformation is defined as [10]

$$\mathcal{T}\left\{\boldsymbol{Y}(x,s)\right\} = \bar{Y}(\mu,s) = \int_0^l \tilde{\boldsymbol{K}}^\mathrm{H}(x,\mu)\boldsymbol{C}\boldsymbol{Y}(x,s)\mathrm{d}x, \tag{13}$$

and the inverse transformation can be written in terms of a sum as

$$\mathcal{T}^{-1}\left\{\bar{Y}(\mu,s)\right\} = \boldsymbol{Y}(x,s) = \sum_\mu \frac{1}{N_\mu}\bar{Y}(\mu,s)\boldsymbol{K}(x,\mu), \tag{14}$$

with the scaling factor

$$N_\mu = \int_0^l \tilde{\boldsymbol{K}}^\mathrm{H}(x,\mu)\boldsymbol{C}\boldsymbol{K}(x,\mu)\mathrm{d}x. \tag{15}$$

The vectors $\boldsymbol{K}(x,\mu)$ and $\tilde{\boldsymbol{K}}(x,\mu)$ are the kernel functions (eigenfunctions) of the transformation, which depend on the problem. The kernel functions fulfill different properties which are shown in [10]. The eigenfunctions $\tilde{\boldsymbol{K}}(x,\mu)$ are adjoint to the primal eigenfunctions $\boldsymbol{K}(x,\mu)$. Additionally, the two sets of eigenfunctions are biorthogonal. The integer index μ can be regarded as a discrete spatial frequency variable.

3.2. Application to the PDE

The transformation from Eq. (13) is now applied to Eq. (11). The properties of the eigenfunctions from Sec. 3.3 lead to the result

$$s\bar{Y}(\mu,s) - s_\mu\bar{Y}(\mu,s) - \bar{\Phi}(\mu,s) = \bar{F}_\mathrm{e}(x,s), \tag{16}$$

with the transformed boundary and excitation term

$$\bar{\Phi}(\mu,s) = \left[\tilde{\boldsymbol{K}}^\mathrm{H}(x,\mu)\boldsymbol{\Phi}(x,s)\right]\Big|_0^l \qquad x = 0, l, \tag{17}$$

$$\bar{F}_\mathrm{e}(\mu,s) = \int_0^l \tilde{\boldsymbol{K}}^\mathrm{H}(x,\mu)\boldsymbol{F}_\mathrm{e}(x,s)\mathrm{d}x, \tag{18}$$

where the values s_μ represent the spatial eigenfrequencies. Solving Eq. (16) for the transformed output signal gives

$$\bar{Y}(\mu,s) = \frac{1}{s - s_\mu}\left[\bar{\Phi}(\mu,s) + \bar{F}_\mathrm{e}(\mu,s)\right]. \tag{19}$$

3.3. Eigenvalue Problems

The two kernel functions of the forward and inverse SLT have to fulfill their eigenvalue problems and boundary conditions, so that the transformation is applicable in Eq. (16)

$$L\boldsymbol{K}(x,\mu) = s_\mu\boldsymbol{C}\boldsymbol{K}(x,\mu) \qquad \boldsymbol{F}_\mathrm{b}^\mathrm{H}\boldsymbol{K}(x,\mu) = \boldsymbol{0}, \tag{20}$$

$$\tilde{L}\tilde{\boldsymbol{K}}(x,\mu) = s_\mu^*\boldsymbol{C}^\mathrm{H}\tilde{\boldsymbol{K}}(x,\mu) \qquad \tilde{\boldsymbol{F}}_\mathrm{b}^\mathrm{H}\tilde{\boldsymbol{K}}(x,\mu) = \boldsymbol{0}, \tag{21}$$

with the adjoint differential operator and boundary matrix

$$\tilde{L} = \boldsymbol{A}^{\mathrm{H}} - \boldsymbol{I}\frac{\partial}{\partial x} \qquad \tilde{\boldsymbol{F}}_{\mathrm{b}}^{\mathrm{H}} = \boldsymbol{I} - \boldsymbol{F}_{\mathrm{b}}^{\mathrm{H}}. \tag{22}$$

Indeed, Eq. 16 follows by applying the SLT from (13) to the PDE (2), observing the properties (20) and (21) and integration by parts. From these eigenvalue problems the primal and the adjoint eigenfunctions are calculated, see Sec. 3.6.

3.4. Kernel functions

To calculate the eigenfunctions $\boldsymbol{K}(x,\mu)$ and $\tilde{\boldsymbol{K}}(x,\mu)$ from the eigenvalue problems, Eqs. (20) and (21) are reformulated as

$$\partial_x \boldsymbol{K}(x,\mu) = \boldsymbol{Q}\boldsymbol{K}(x,\mu), \tag{23}$$

$$\partial_x \tilde{\boldsymbol{K}}(x,\mu) = \tilde{\boldsymbol{Q}}\tilde{\boldsymbol{K}}(x,\mu), \tag{24}$$

with

$$\boldsymbol{Q} = s_\mu \boldsymbol{C} - \boldsymbol{A}, \tag{25}$$

$$\tilde{\boldsymbol{Q}} = \boldsymbol{A}^{\mathrm{H}} - s_\mu^* \boldsymbol{C}^{\mathrm{H}} = -\boldsymbol{Q}^{\mathrm{H}}. \tag{26}$$

The solution of the primal eigenvalue problem for the eigenfunctions can be formulated in terms of a matrix exponential

$$\boldsymbol{K}(x,\mu) = e^{\boldsymbol{Q}x}\boldsymbol{K}(0,\mu), \tag{27}$$

where $\boldsymbol{K}(0,\mu)$ is the boundary vector of the eigenfunctions. The matrix exponential can be calculated using the method from [12], or by any other suitable method. The solution for the adjoint kernel function $\tilde{\boldsymbol{K}}(x,\mu)$ is formulated analogously.

3.5. Eigenvalues and Eigenfrequencies

To calculate the kernel functions with Eq. (27) the eigenvalues λ of matrix $\boldsymbol{Q}$ and the eigenfrequencies s_μ of the string model have to be derived. Therefore the characteristic polynomial $p_{\mathrm{Q}}(\lambda)$ of the matrix $\boldsymbol{Q}$ is calculated. It follows from Eq. (25) as

$$p_{\mathrm{Q}}(\lambda) = \lambda^4 - q_2\lambda^2 - q_1 s_\mu, \tag{28}$$

with the coefficients

$$q_1 = c_1 s_\mu - a_1 \qquad q_2 = c_2 s_\mu - a_2. \tag{29}$$

With relation between the eigenvalues

$$\lambda_2 = -\lambda_1 \qquad \lambda_4 = -\lambda_3, \tag{30}$$

follows for the eigenvalues of the matrix $\boldsymbol{Q}$

$$\lambda_{1/3}^2 = \frac{1}{2}\left(q_2 \pm \sqrt{q_2^2 + 4q_1 s_\mu}\right). \tag{31}$$

By evaluating the boundary conditions for the eigenfunctions in Eqs. (23) and (24), it can be shown that the eigenvalues λ can only adopt values of the form

$$\lambda = \lambda_\mu = j\mu\frac{\pi}{l} = j\gamma_\mu. \tag{32}$$

Setting Eq. (28) to zero and solving for the eigenfrequencies s_μ leads to the dispersion relation

$$s_\mu^2 + \left(\frac{a_1}{c_1} - \lambda^2\frac{c_2}{c_1}\right)s_\mu - \frac{\lambda^2}{c_1}\left(\lambda^2 + a_2\right) = 0. \tag{33}$$

The eigenfrequencies s_μ are the solutions of the dispersion relation (33) for λ from (32). For each value of γ_μ the dispersion relation yields two complex conjugate eigenfrequencies s_μ.

3.6. Solution for the kernels

Using the derivations from the previous sections the eigenfunctions can be calculated from Eq. (27). Skipping many explicit calculations, the eigenfunction $\boldsymbol{K}(x,\mu)$ results in

$$\boldsymbol{K}(x,\mu) = \begin{bmatrix} \frac{s_\mu}{\gamma_\mu}\sin(\gamma_\mu x) \\ \cos(\gamma_\mu x) \\ -\gamma_\mu \sin(\gamma_\mu x) \\ -\gamma_\mu^2 \cos(\gamma_\mu x) \end{bmatrix}, \tag{34}$$

and similar for the adjoint eigenfunction function

$$\tilde{\boldsymbol{K}}(x,\mu) = \begin{bmatrix} q_1^* \cos(\gamma_\mu x) \\ -\frac{s_\mu^* q_1^*}{\gamma_\mu}\sin(\gamma_\mu x) \\ -\gamma_\mu^2 \cos(\gamma_\mu x) \\ \gamma_\mu \sin(\gamma_\mu x) \end{bmatrix}. \tag{35}$$

These solutions can be verified via inserting them into the eigenvalue problems from the Eqs. (23) and (24).

3.7. Output Signal

The results from the previous sections can now be used to construct the synthesis equation for the string, which is based on the inverse SLT from Eq. (14). As mentioned in Sec. 3.5 for one μ-value a complex conjugated pair of eigenfrequencies s_μ arises in Eq. (33). Setting the excitation function $f_{\mathrm{e}}(x,t)$ to zero for brevity, the synthesis equation turns into

$$\begin{aligned} \boldsymbol{Y}(x,s) &= \sum_\mu \frac{1}{N_\mu}\boldsymbol{K}(x,\mu)\bar{Y}(\mu,s) = \\ &= \sum_\mu \frac{1}{N_\mu}\boldsymbol{K}(x,\mu)\frac{1}{s-s_\mu}\bar{\Phi}(\mu,s). \end{aligned} \tag{36}$$

The resulting synthesis system is pictured in Fig. 1. The output $\boldsymbol{Y}(x,s)$ is a superposition of many first-order systems – oscillating with the eigenfrequencies of the system and weighted with the eigenfunctions $\boldsymbol{K}(x,\mu)$. For a more suitable implementation, each pair of complex conjugate first-order systems may be merged into one real-valued second-order system.

3.8. Discrete-time equivalent

This synthesis structure can be transformed into the discrete-time domain to achieve a difference equation for computer implementation. Using e.g. impulse-invariant-transformation [6] turns the transfer function from Eq. (19) into

$$\bar{Y}(\mu,z) = \frac{z}{z-z_\mu}\bar{\Phi}(\mu,z), \tag{37}$$

where $\bar{Y}(\mu,z)$ and $\bar{\Phi}(\mu,z)$ are the z-domain equivalents of $\bar{Y}(\mu,s)$ and $\bar{\Phi}(\mu,s)$. The poles z_μ are defined as

$$z_\mu = \exp\left(s_\mu T\right), \tag{38}$$

with the sampling time T.

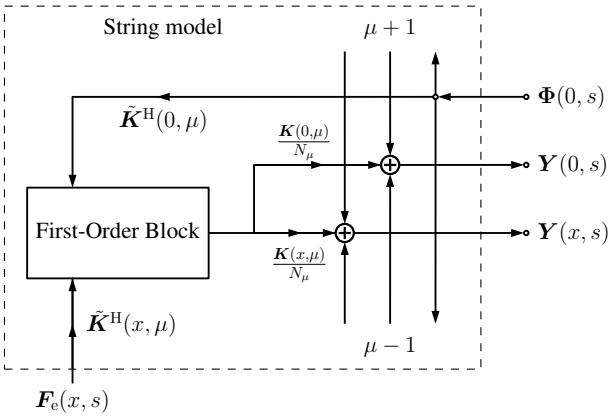


Figure 1: Block diagram of the synthesis system; $\boldsymbol{\Phi}(0,s)$, $\boldsymbol{Y}(0,s)$: Vector of boundary excitations/observations at $x=0$, $\boldsymbol{Y}(x,s)$: vector of output signals at any position x, $\boldsymbol{F}_{\mathrm{e}}(x,s)$: Excitation function.

4. ADJUSTABLE BOUNDARY CONDITIONS

The boundary conditions of a PDE are essential for the transformation with the FTM, but the more complex the boundary conditions are, the more complex becomes the transformation. Using adjustable boundary conditions means to transform the PDE including a simple set of boundary conditions and later adjust them to fulfill any kind of boundary condition [8, 9].

4.1. Simple Boundary Conditions

As starting point, the simple set of boundary conditions from Sec. 2.2 are revisited. The boundary behavior at $x=0$ as described by Eqs. (8) and (9) in terms of the boundary matrix $\boldsymbol{F}_{\mathrm{b}}^{\mathrm{H}}$ and a vector of boundary excitations $\boldsymbol{\Phi}(0,s)$ reads in the frequency domain

$$\boldsymbol{F}_{\mathrm{b}}^{\mathrm{H}}\boldsymbol{Y}(0,s)=\begin{bmatrix}1&0&0&0\\0&0&0&0\\0&0&1&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}sY(0,s)\\Y'(0,s)\\Y''(0,s)\\Y'''(0,s)\end{bmatrix}=\begin{bmatrix}\dot{\Phi}_1\\0\\\Phi_3\\0\end{bmatrix}. \quad (39)$$

In addition an observation matrix $\boldsymbol{F}_{\mathrm{o}}^{\mathrm{H}}$ is defined. It transforms the vector of variables $\boldsymbol{Y}(x,s)$ into a vector of boundary observations $\boldsymbol{Y}_{\mathrm{o}}$, which can be written as

$$\boldsymbol{F}_{\mathrm{o}}^{\mathrm{H}}\boldsymbol{Y}(0,s)=\begin{bmatrix}0&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}sY(0,s)\\Y'(0,s)\\Y''(0,s)\\Y'''(0,s)\end{bmatrix}=\begin{bmatrix}0\\Y_{\mathrm{o2}}\\0\\Y_{\mathrm{o4}}\end{bmatrix}. \quad (40)$$

The boundary behavior of the variables of the string model can be formulated in terms of the boundary and the observation matrix

$$\boldsymbol{Y}(0,s)=\left(\boldsymbol{F}_{\mathrm{b}}^{\mathrm{H}}+\boldsymbol{F}_{\mathrm{o}}^{\mathrm{H}}\right)\boldsymbol{Y}(0,s), \quad (41)$$

and is pictured at the position $x=0$ in Fig. 2. It shows that the boundary excitations Φ can be interpreted as input variables and the boundary observations as output variables at the boundary of the string.

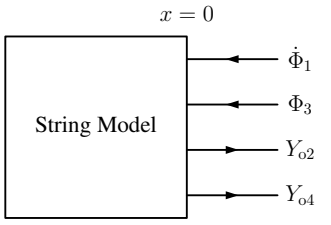


Figure 2: Schematic of the boundary behavior of the string model referring to Eqs. (39) - (40). Y_{o}: Boundary Observations, Φ: Boundary Excitations.

4.2. Impedance Boundary Conditions

In this section, impedance boundary conditions at $x=0$ are investigated (see e.g. [6]). They connect the velocity of the string ($\dot{y}(0,t)$) at the boundary to a force $f(0,t)$ via the mechanical impedance Z_{s} [13]. The force can be exerted e.g. by pressing the ball of the thumb on the bridge to damp the string vibration.

The relation between force and velocity can be formulated in the frequency domain with the Laplace transforms of the velocity $sY(0,s)$ and of the force $F(0,s)$ with the admittance $Y_{\mathrm{s}}=Z_{\mathrm{s}}^{-1}$

$$sY(0,s)-Y_{\mathrm{s}}\,F(0,s)=\Phi_{\mathrm{Z1}}, \quad (42)$$

$$Y''(0,s)=\Phi_{\mathrm{Z3}}, \quad (43)$$

Φ_{Z1} and Φ_{Z3} are the boundary excitations. The force F can be written in the terms of the variables in Eq. (4)

$$F(0,s)=T_{\mathrm{s}}\,Y'(0,s)-EI\,Y''(0,s). \quad (44)$$

The boundary conditions from Eqs. (42) – (43) are rearranged into a boundary matrix, which transforms the variable vector into a vector of boundary excitations

$$\boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}\boldsymbol{Y}(0,s)=\boldsymbol{\Phi}_{\mathrm{Z}}, \quad (45)$$

with

$$\boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}=\begin{bmatrix}1&-Y_{\mathrm{s}}T_{\mathrm{s}}&0&Y_{\mathrm{s}}EI\\0&0&0&0\\0&0&1&0\\0&0&0&0\end{bmatrix}. \quad (46)$$

4.3. Adjustable Boundary Conditions

Using different sets of boundary conditions would mean to recalculate the eigenfunctions from Eqs. (20) and (21) with different boundary conditions and to re-apply the corresponding transformations according to Sec. 3. Furthermore, the determination of the eigenfunctions for impedance boundary conditions does in general not lead to closed form solutions.

A different approach is to use the eigenfunctions for simple boundary conditions from Eqs. (34) and (35) and to feed the boundary observations $\boldsymbol{Y}_{\mathrm{o}}$ from (40) back into the boundary excitations $\dot{\Phi}_1$ and Φ_3 from (39) via the impedance condition (45). This approach had been discussed in general terms in [9] and is now applied to string vibrations.

Starting point is Eq. (45), which can be reformulated using Eq. (41) as

$$\begin{aligned}\boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}\boldsymbol{Y}(0,s) &= \boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}\left(\boldsymbol{F}_{\mathrm{b}}^{\mathrm{H}}+\boldsymbol{F}_{\mathrm{o}}^{\mathrm{H}}\right)\boldsymbol{Y}(0,s)\\ &= \boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}\boldsymbol{F}_{\mathrm{b}}^{\mathrm{H}}\boldsymbol{Y}(0,s)+\boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}\boldsymbol{F}_{\mathrm{o}}^{\mathrm{H}}\boldsymbol{Y}(0,s)=\boldsymbol{\Phi}_{\mathrm{Z}}. \quad (47)\end{aligned}$$

With Eqs. (39) - (40) follows

$$\boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}\boldsymbol{\Phi}=-\boldsymbol{F}_{\mathrm{bZ}}^{\mathrm{H}}\boldsymbol{Y}_{\mathrm{o}}+\boldsymbol{\Phi}_{\mathrm{Z}}. \quad (48)$$

This equation leads to a rule for the transformation of the simple boundary excitations $\boldsymbol{\Phi}$ from Eq. (39) into impedance boundary conditions

$$\begin{bmatrix}\dot{\Phi}_1\\ \Phi_3\end{bmatrix}=Y_{\mathrm{s}}\begin{bmatrix}T_{\mathrm{s}} & -EI\\ 0 & 0\end{bmatrix}\begin{bmatrix}Y_{\mathrm{o}2}\\ Y_{\mathrm{o}4}\end{bmatrix}+\begin{bmatrix}\Phi_{\mathrm{Z}1}\\ \Phi_{\mathrm{Z}3}\end{bmatrix}. \quad (49)$$

The boundary behavior of the string model at $x=0$ is pictured in Fig. 3. It can be seen that the impedance boundary conditions result in a boundary circuit, which adjusts the simple boundary values to fulfill the impedance conditions, according to the transformation rule (49).

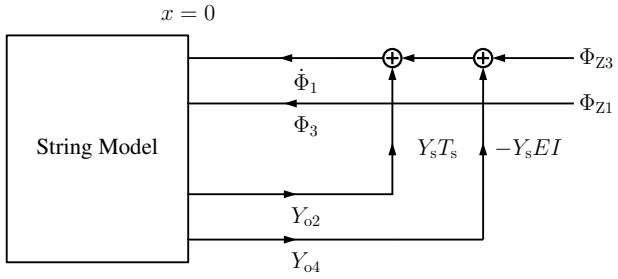


Figure 3: Schematic of the boundary behavior of the string model referring to Eq. (49). Y_{o}: Boundary observations, Φ: Boundary excitations.

With the adjustment of the boundary conditions, one can use a simple model of the string and later adjust the boundaries to match a more complex set of conditions. The adjustment is realized by an external feedback loop between the boundary observations $\boldsymbol{Y}_{\mathrm{o}}$ and the boundary excitations $\dot{\Phi}_1$ and Φ_3.

4.4. Input-Output model for the boundary

This section explains how the output of the string model from Eq. (36) is connected to the boundary circuit from Eq. (49). For the position $x=0$ follows for the eigenfunctions

$$\boldsymbol{K}(0,\mu)=\begin{bmatrix}0 & 1 & 0 & -\gamma_\mu^2\end{bmatrix}^{\mathrm{T}}, \quad (50)$$

$$\tilde{\boldsymbol{K}}^{\mathrm{H}}(0,\mu)=\begin{bmatrix}q_1 & 0 & -\gamma_\mu^2 & 0\end{bmatrix}. \quad (51)$$

Therefore the two non-zero output values of Eq. (41) follow from the z-domain equivalent of (36) as

$$\boldsymbol{Y}_{\mathrm{o}}(z)=\begin{bmatrix}Y_{\mathrm{o}2}(z)\\ Y_{\mathrm{o}4}(z)\end{bmatrix}=\sum_\mu \frac{1}{N_\mu}\bar{Y}(\mu,z)\begin{bmatrix}1\\ -\gamma_\mu^2\end{bmatrix}, \quad (52)$$

where $\bar{Y}(\mu,z)$ is connected to $\bar{\Phi}(\mu,z)$ via Eq. (37).

The boundary term $\bar{\Phi}(\mu,z)$ depends on the boundary conditions $\boldsymbol{\Phi}(0,z)$ and $\boldsymbol{\Phi}(l,z)$ similarly to the continuous-time formulation in (17). Assuming at $x=l$ simple and homogeneous boundary conditions, i.e. $\boldsymbol{\Phi}(l,z)=0$ and at $x=0$ adjustable boundary conditions according to Eq. (49) with $\Phi_3=0$ leaves only a dependency on $\dot{\Phi}_1$. Then $\bar{\Phi}(\mu,z)$ can be expressed using (51) as

$$\bar{\Phi}(\mu,z)=-\bar{\boldsymbol{K}}^{\mathrm{H}}(x,\mu)\boldsymbol{\Phi}(0,z)=-q_1\dot{\Phi}_1(0,z). \quad (53)$$

Now an input-output model for the quantities shown in Fig. 3 can be set up by combining (37) and (52)

$$z\bar{Y}(\mu,z)=z_\mu\bar{Y}(\mu,z)-q_1 z\dot{\Phi}_1(0,z), \quad (54)$$

to obtain the internal variable $\bar{Y}(\mu,z)$ from the boundary input Φ_1 and by using (52) to obtain the boundary output $\boldsymbol{Y}_{\mathrm{o}}$ from $\bar{Y}(\mu,z)$.

However, following the signal flow in (49), (54) and (52) unveils the existence of a delay-free loop. The next section describes how this delay-free loop can be avoided.

5. STATE SPACE MODEL

As described before the adjustment of the boundary conditions according to Eq. (49) causes a delay-free loop in the system. Delay-free loops can be resolved using iterative methods described in [14, 15]. In this paper the delay-free loop is avoided altogether by transformation into a state space model.

5.1. State Equations

At first a state equation is established with the definition of the state variable

$$\bar{W}(\mu,z)=\bar{Y}(\mu,z)+q_1\dot{\Phi}_1(0,z). \quad (55)$$

Since the excitation function $f_{\mathrm{e}}(x,t)$ does not contribute to the boundary feedback and thus to a possible delay-free loop, it is set to zero for brevity.

Rewriting (54) and (52) with the state variable $\bar{W}(\mu,z)$ and collecting the terms depending on μ into vectors and matrices gives finally

$$z\bar{\boldsymbol{W}}(z)=\boldsymbol{A}\bar{\boldsymbol{W}}(z)+\boldsymbol{b}\,\dot{\Phi}_1(0,z), \quad (56)$$

$$\boldsymbol{Y}_{\mathrm{o}}(z)=\boldsymbol{C}_{\mathrm{o}}\bar{\boldsymbol{W}}(z)+\boldsymbol{d}_{\mathrm{o}}\dot{\Phi}_1(0,z). \quad (57)$$

The state and output vectors are given as

$$\bar{\boldsymbol{W}}(z)=\begin{bmatrix}\vdots\\ \bar{W}(\mu,z)\\ \vdots\end{bmatrix}, \qquad \boldsymbol{Y}_{\mathrm{o}}(z)=\begin{bmatrix}Y_{\mathrm{o}2}\\ Y_{\mathrm{o}4}\end{bmatrix}, \quad (58)$$

and the state matrices follow as

$$\boldsymbol{A}=\mathrm{diag}\left(\dots,z_\mu,\dots\right), \quad (59)$$

$$\boldsymbol{b}=\left[\dots,z_\mu q_1(s_\mu),\dots\right], \quad (60)$$

$$\boldsymbol{C}_{\mathrm{o}}=\left[\dots,\frac{1}{N_\mu}\begin{bmatrix}1\\ -\gamma_\mu^2\end{bmatrix},\dots\right], \quad (61)$$

$$\boldsymbol{d}_{\mathrm{o}}=\sum_\mu\frac{q_1(s_\mu)}{N_\mu}\begin{bmatrix}1\\ -\gamma_\mu^2\end{bmatrix}. \quad (62)$$

5.2. Feedback Loop

To fit the boundary behavior in Fig. 3 into the state space equations (56) - (57), the boundary equation (49) is reformulated (with $\Phi_{\mathrm{Z1}} = \Phi_{\mathrm{Z3}} = 0$) as

$$\dot{\Phi}_1(0,z) = Y_\mathrm{s}\, \boldsymbol{r}^\mathrm{T} \boldsymbol{Y}_\mathrm{o}(z), \tag{63}$$

with the vector

$$\boldsymbol{r}^\mathrm{T} = [T_s \; -EI]\,. \tag{64}$$

It is part of a delay-free loop, where the remaining part is formed by the direct path in (57).

To avoid this delay-free loop, Eq. (63) is inserted into (57) and solved for the output vector $\boldsymbol{Y}_\mathrm{o}(z)$

$$\boldsymbol{Y}_\mathrm{o}(z) = \left(\boldsymbol{I} - Y_\mathrm{s}\, \boldsymbol{d}_\mathrm{o} \boldsymbol{r}^\mathrm{T}\right)^{-1} \boldsymbol{C}_\mathrm{o} \bar{\boldsymbol{W}}(z). \tag{65}$$

Now the boundary input can be expressed directly in terms of the state vector $\bar{\boldsymbol{W}}(z)$ with the help of (63) and (65)

$$\dot{\Phi}_1(0,z) = Y_\mathrm{s} \boldsymbol{r}^\mathrm{T} \left(\boldsymbol{I} - Y_\mathrm{s} \boldsymbol{d}_\mathrm{o} \boldsymbol{r}^\mathrm{T}\right)^{-1} \boldsymbol{C}_\mathrm{o} \bar{\boldsymbol{W}}(z) = \boldsymbol{r}_\mathrm{b}^\mathrm{T} \bar{\boldsymbol{W}}(z). \tag{66}$$

The vector $\boldsymbol{r}_\mathrm{b}$ contains the influence of the admittance Y_s at the boundary and can be reformulated using the Sherman-Morisson identity [16, eq. (160)]

$$\boldsymbol{r}_\mathrm{b}^\mathrm{T} = Y_\mathrm{s} \boldsymbol{r}^\mathrm{T} \left(\boldsymbol{I} - Y_\mathrm{s} \boldsymbol{d}_\mathrm{o} \boldsymbol{r}^\mathrm{T}\right)^{-1} \boldsymbol{C}_\mathrm{o} = \frac{1}{Z_\mathrm{s} - \boldsymbol{r}^\mathrm{T} \boldsymbol{d}_\mathrm{o}} \boldsymbol{r}^\mathrm{T} \boldsymbol{C}_\mathrm{o}. \tag{67}$$

5.3. Simulation Model

The final simulation model for adjustable boundary conditions follows from the state space representation (56) and (57) and the description of the outer feedback loop by (66). The structure of the model is shown in Fig. 4.

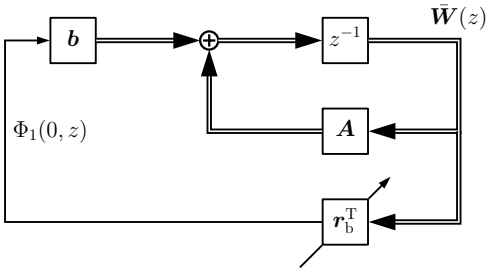


Figure 4: Signal flow of the state space equations from (56) and (57) including the boundary feedback loop of Eq. (66).

There are two loops and both include a time delay z^{-1}. The inner loop is closed by the diagonal matrix $\boldsymbol{A}$ from Eq. (59) and is easy to implement. The outer loop contains the boundary admittance Y_s or impedance Z_s according to (63). So the delay-free loop arising in Sec. 4.4 is avoided using the state space structure. The output-equation (57) calculates the output of the model for the position $x = 0$ (Feedback path in Fig. 4).

The output signal at any other position $x = x_\mathrm{a}$ on the string is calculated by a second output equation

$$\boldsymbol{Y}_\mathrm{a}(z) = \boldsymbol{C}_\mathrm{a} \bar{\boldsymbol{W}}(z) + \boldsymbol{d}_\mathrm{a} \dot{\Phi}_1(0,z), \tag{68}$$

which is basically the same as for $x = 0$. Only two matrices and vectors have to be recalculated

$$\boldsymbol{C}_\mathrm{a} = \left[\ldots, \frac{1}{N_\mu} \boldsymbol{K}(x_\mathrm{a}, \mu), \ldots\right], \tag{69}$$

$$\boldsymbol{d}_\mathrm{a} = \sum_\mu \frac{q_1(s_\mu)}{N_\mu} \boldsymbol{K}(x_\mathrm{a}, \mu). \tag{70}$$

The other matrices are independent of the pick-up position x_a, so they can be directly taken from Eqs. (59) - (60). The complexity of the feedback loop at the position $x = 0$ is hidden in the vector $\boldsymbol{r}_\mathrm{b}^\mathrm{T}$, which is independent of any pick-up point on the string. Thus the feedback path is variable by adjusting the value for the impedance Z_s in (67), see Fig. 4.

6. EXAMPLES

The following section presents simulation results, which are based on the theory from previous sections. The simulation uses the string model from Sec. 3 with a simple set of boundary conditions, which are adjusted to fulfill the impedance boundary conditions using the concept from Sec. 4.3. The model is implemented with the state space representation from Sec. 5 to avoid delay-free loops.

6.1. Basic Parameters

The string model for the simulation is based on the transformation of the PDE of a vibrating string and the subsequent state space representation from Eqs. (56) - (57). The boundary conditions of the string are pictured in Fig. 5. The string has a supported end at $x = l$, which results in homogeneous boundary conditions referring to (7)

$$Y(x,s) = 0, \qquad Y''(x,s) = 0 \qquad x = l. \tag{71}$$

At the position $x = 0$ the string is placed on the bridge and is influenced by an admittance, which results in impedance boundary conditions from Eqs. (42) - (43). The boundary excitations Φ_{Z1} and Φ_{Z3} are set to zero since there are no external forces. The mechanical admittance Y_s is used as an adjustable parameter and is considered as frequency independent.

For the simulations a nylon guitar B string was used. The physical parameters of the string are taken from [6, 17] and are listed in Table 1.

ρ	Density	$1140\,\mathrm{kg/m^3}$
E	Young's modulus	$5.4\,\mathrm{GPa}$
l	Length	$0.65\,\mathrm{m}$
A	Cross section area	$0.5188\,\mathrm{mm^2}$
I	Moment of inertia	$0.141\,\mathrm{mm^4}$
d_1	Freq. indep. damping	$8 \cdot 10^{-5}\,\mathrm{kg/(ms)}$
d_3	Freq. dep. damping	$-1.4 \cdot 10^{-5}\,\mathrm{kgm/s}$
T_S	Tension	$60.97\,\mathrm{N}$

Table 1: Physical parameters of a nylon guitar B string.

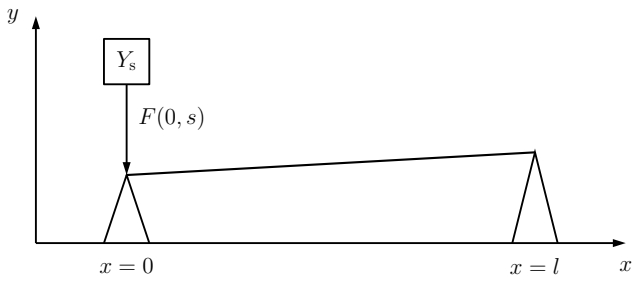


Figure 5: Guitar string influenced by an impedance/admittance caused e.g. the ball of players hand at bridge position $x = 0$ and a supported end at $x = l$.

6.2. Simulation Results

The following section presents the results of the simulation of the string model based on the previous chapters. At first, the influence of the admittance Y_s on the bending $y'(0,t)$ at the position $x = 0$ is shown. Then the influence on the velocity $\dot{y}(x_\mathrm{a},t)$ at a specific pick-up point $x = x_\mathrm{a}$ is presented. In each case the signal is pictured for three admittance values between $Y_\mathrm{s} = 0$ and $Y_\mathrm{s} = 0.125\,\frac{\mathrm{s}}{\mathrm{kg}}$. The string is excited by a single impulse at the position $x_\mathrm{e} = 0.5\,\mathrm{m}$

$$f_\mathrm{e}(x) = \begin{cases} 50\,\mathrm{mN} & x = x_\mathrm{e} \\ 0 & x \neq x_\mathrm{e} \end{cases}. \tag{72}$$

The results are presented by the normalized amplitude spectra of the velocity and bending. At first results for a zero admittance ($Y_\mathrm{s} = 0$) are presented, then the admittance is varied and also a non-zero pick-up position is considered.

Results for zero admittance

In this part the string model from Eq. (36) using $\mu = 1 \ldots 100$ complex eigenfrequencies with the simple set of boundary conditions from Sec. 4.1 is considered. For the implementation the state space representations from Eqs. (56) - (57) is used with $\Phi_1 = 0$. Fig. 6 shows the variation of the bending $y'(x,t)$ over time and pick-up position for a mechanical admittance $Y_\mathrm{s} = 0\,\mathrm{s/kg}$.

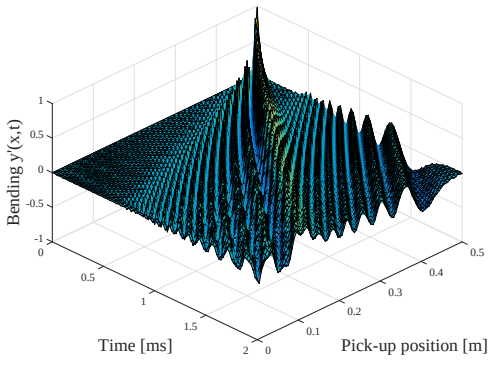


Figure 6: Variation of the bending $y'(x,t)$ over time at different pick-up positions on the string for $Y_\mathrm{s} = 0\,\mathrm{s/kg}$.

The excitation function in Fig. 6 is an impulse according to Eq. (72). It causes the propagation of waves on the string. The results show that the derived string model matches the behavior of a real string for simple boundary conditions [6–8, 10].

Results for non-zero admittance

The following experiments show the behavior of the spring model using the impedance boundary conditions from Sec. 4.2. The implementation uses the state space representation from Eqs. (56) - (57), with boundary term $\dot{\Phi}_1$ from Eq. (66). The admittance is varied between $Y_\mathrm{s} = 0$ and a maximum value of $Y_\mathrm{s} = 0.125\,\mathrm{s/kg}$, which is taken from [7].

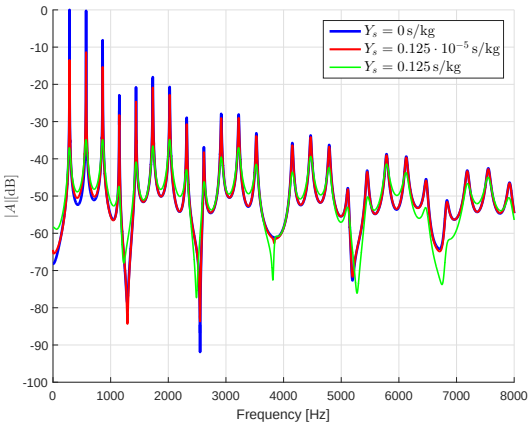


Figure 7: Amplitude spectra of bending $y'(x,t)$ at the bridge position $x = 0$.

Fig. 7 shows the amplitude spectra of bending $y'(x,t)$ at the position $x = 0$ for different admittance values. For an admittance value of $Y_\mathrm{s} = 0$ results the spectrum of a string with simple boundary conditions, as the lower feedback path in Fig. 4 is zero. For an admittance value $Y_\mathrm{s} = 0.125 \cdot 10^{-5}\,\mathrm{s/kg}$ the damping influence of the admittance on the bending can be seen clearly, especially in the low-frequency region. For the maximum value of admittance $Y_\mathrm{s} = 0.125\,\mathrm{s/kg}$ the whole spectrum is damped and is much flatter. Thus the increase of the admittance value leads to the well known effect of a decreased oscillation time of the system. This reproduces the sound of a palm muted playing style.

With the increasing value of mechanical admittance, the poles z_μ of the simple string model are shifted towards the origin of the unit circle by the impedance boundary condition feedback loop. Thus the Euclidean distance of the poles is reduced and the single frequency components of the signal are damped depending on the value of the admittance.

Fig. 8 shows the spectra of velocity $\dot{y}(x,t)$ at the pick-up position $x_\mathrm{a} = 0.4\,\mathrm{m}$ for the same three values of admittance Y_s. The general behavior of the spectra is similar to the behavior in Fig. 7. But especially the influence of the feedback loop for the mid-value of Y_s is not so strong at $x = 0.4\,\mathrm{m}$ as for $x = 0$ in Fig. 7. This makes sense, as the pick-up point is removed by $x = 0.4\,\mathrm{m}$ from the bridge position of the guitar, so the damping influence is not as strong as for the bridge position. For the maximum value of the admittance, the spectrum is similarly flat as before at the position $x = 0$.

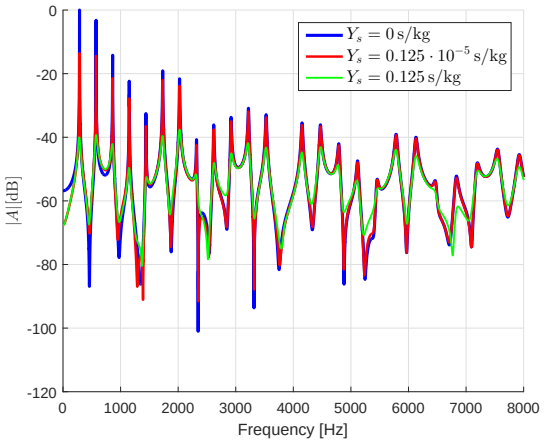


Figure 8: Amplitude spectra of velocity $\dot{y}(x,t)$ at the position $x = 0.4\,\mathrm{m}$.

For both pick-up positions the influence of the admittance damps the harmonics of the output signal. According to different values of the admittance, the signals sounds fully or partially muted.

7. CONCLUSIONS AND FURTHER WORK

This paper proposes a complex string model with general boundary conditions. It is realized by standard state space methods based on the functional transformation method. Then the simple boundary conditions for supported ends are adjusted to impedance boundary conditions, without the need for a recalculation of the eigenfunctions. Using this principle, the simple string model is connected to an admittance at the bridge position.

Here a frequency independent admittance is adopted. It is suitable for modeling a palm muted playing style where the ball of the players hand damps the strings. Other types of boundary conditions require frequency dependent admittances. For example it is well known that the connection of a string to a sound board exhibits strong resonances [17]. They are described by a bridge impedance with multiple poles [4]. Frequency dependent impedances of this type can be modeled with an arrangement similar to Fig. 4, where the feedback path includes a digital filter with complex poles. Another way is to exploit the parallel resonator structure of the functional transformation method by merging the impedance feedback path with the $\boldsymbol{A}$ matrix of the state space representation. Then each diagonal entry of the matrix $\boldsymbol{A}$ can be weighted by a different impedance value as proposed in [8].

8. REFERENCES

[1] Neville H. Fletcher and Thomas D. Rossing, *The Physics of Musical Instruments*, Springer-Verlag, New York, USA, 2nd edition, 1998.

[2] Stefan Bilbao, *Numerical Sound Synthesis*, John Wiley and Sons, Chichester, UK, 2009.

[3] V. Välimäki, J. Pakarinen, C. Erkut, and M. Karjalainen, "Discrete time modeling of musical instruments," *Reports on Progress in Physics*, vol. 69, pp. 1–78, 2006.

[4] Julius O. Smith III, *Physical Audio Signal Processing*, W3K Publishing, 2010, http://www.w3k.org/books.

[5] Cumhur Erkut, *Aspects in Analysis and Model-based Sound Synthesis of Plucked String Instruments*, Ph.D. thesis, Helsinki University of Technology, Espoo, Finland, 2002.

[6] R. Rabenstein and L. Trautmann, "Digital sound synthesis of string instruments with the functional transformation method," *Signal Processing*, vol. 83, no. 8, pp. 1673–1688, August 2003.

[7] Lutz Trautmann and Rudolf Rabenstein, *Digital Sound Synthesis by Physical Modeling using the Functional Transformation Method*, Kluwer Academic Publishers, New York, USA, 2003.

[8] L. Trautmann, S. Petrausch, and M. Bauer, "Simulations of string vibrations with boundary conditions of third kind using the functional transformation method," *The Journal of the Acoustical Society of America (JASA)*, vol. 118, no. 3, pp. 1763–1775, September 2005.

[9] R. Rabenstein and S. Petrausch, "Adjustable boundary conditions for multidimensional transfer function models," in *10th International Conference on Digital Audio Effects (DAFx-07)*, Bordeaux, France, September 2007, pp. 305–310.

[10] S. Petrausch and R. Rabenstein, "A simplified design of multidimensional transfer function models," in *International Workshop on Spectral Methods and Multirate Signal Processing (SMMSP2004)*, Vienna, Austria, September 2004, pp. 35–40.

[11] Julien Bensa, Stefan Bilbao, Richard Kronland-Martinet, and Julius O. Smith, "The simulation of piano string vibration: From physical models to finite difference schemes and digital waveguides," *The Journal of the Acoustical Society of America*, vol. 114, no. 2, pp. 1095–1107, 2003.

[12] U. Luther and K. Rost, "Matrix Exponentials and Inversion of Confluent Vandermonde Matrices," *Electronic Transactions on Numerical Analysis*, vol. 18, pp. 91–100, 2004.

[13] A.D. Pierce, *Acoustics – An Introduction to its physical principles and applications*, Acoustical Society of America, 1991, reprint.

[14] A. Härmä, "Implementation of recursive filters having delay free loops," *1998. Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 3, pp. 1261–1264, May 1998.

[15] G. Borin, G. De Poli, and D. Rocchesso, "Elimination of delay-free loops in discrete-time models of nonlinear acoustic systems," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 5, pp. 597–605, Sep. 2000.

[16] K. B. Petersen and M. S. Pedersen, "The matrix cookbook," http://www2.imm.dtu.dk/pubdb/p.php?3274, Nov 2012, Version 20121115.

[17] O. Christensen, "An oscillator model for analysis of guitar sound pressure response," *Acta Acustica united with Acustica*, vol. 54, no. 5, pp. 289–295, 1984.

A MODAL APPROACH TO THE NUMERICAL SIMULATION OF A STRING VIBRATING AGAINST AN OBSTACLE: APPLICATIONS TO SOUND SYNTHESIS

Clara Issanchou, Jean-Loïc Le Carrou
CNRS, LAM / d'Alembert
Sorbonne Universités, UPMC Univ Paris 06
Paris, France
issanchou@lam.jussieu.fr

Stefan Bilbao
Acoustics and Audio Group
University of Edingurgh
Edinburgh, Scotland
sbilbao@staffmail.ed.ac.uk

Cyril Touzé, Olivier Doaré
IMSIA, ENSTA ParisTech-CNRS-EDF-CEA
Université Paris Saclay
Palaiseau, France
cyril.touze@ensta-paristech.fr

ABSTRACT

A number of musical instruments (electric basses, tanpuras, sitars...) have a particular timbre due to the contact between a vibrating string and an obstacle. In order to simulate the motion of such a string with the purpose of sound synthesis, various technical issues have to be resolved. First, the contact phenomenon, inherently nonlinear and producing high frequency components, must be described in a numerical manner that ensures stability. Second, as a key ingredient for sound perception, a fine-grained frequency-dependent description of losses is necessary. In this study, a new conservative scheme based on a modal representation of the displacement is presented, allowing the simulation of a stiff, damped string vibrating against an obstacle with an arbitrary geometry. In this context, damping parameters together with eigenfrequencies of the system can be adjusted individually, allowing for complete control over loss characteristics. Two cases are then numerically investigated: a point obstacle located in the vicinity of the boundary, mimicking the sound of the tanpura, and then a parabolic obstacle for the sound synthesis of the sitar.

1. INTRODUCTION

In many musical instruments, across various cultures, the interaction between a vibrating structure and an obstacle is a key feature leading to amplitude-dependent timbral modification, and is essential in order to replicate the resulting sound. This contact may arise due to the excitation of the instrument [1, 2], in which case the exciting mechanism may be considered as lumped, or during its consequent vibrations [3]. In the latter case, the contact can be pointwise (e.g. the case of string/fret contact in an electric bass) or distributed (e.g. string/bridge contact in a sitar). Such interactions are strongly nonlinear, which complicates significantly its numerical study.

Following analytical studies of the contact between a string and a rigid obstacle [4, 5, 6], a number of numerical methods have been developed in order to simulate the interaction between a vibrating string and an obstacle. Waveguides are used in [7, 8, 9, 10], coupled with finite difference schemes in [11], where the string is ideal, and [12], where the string is damped and stiff and the obstacle is located at one end of the string. A modal description is presented in [13] for modelling an ideal string vibrating against a parabolic obstacle at one boundary and in [14] for a dispersive lossy string where an obstacle consolidated at the bridge of a tanpura is considered. This instrument is also modelled in [15, 16], where the motion of a stiff damped string against an obstacle is obtained by discretising Hamilton's equations of motion. Finally, a finite difference method is developed in [3], also allowing the simulation of a stiff, damped string with an obstacle having an arbitrary shape. The case of the interaction between a string and a fretboard is in particular described in [17]. This obstacle is also considered in [18] where the Functional Transformation Method is used. However, in these models, eigenfrequencies and damping parameters cannot be arbitrary, but must follow a distribution specified by a small number of tuning parameters.

In this study, we present a conservative numerical scheme to model a stiff damped string vibrating against an arbitrarily shaped obstacle. The key features of the scheme are as follows. A modal expansion is used as a starting point for the linear case (i.e., in the absence of contact). By using an equal number of modes and discretization points, a linear transformation relates the spatial displacement and the modal coordinates, so that the contact force is treated directly with the displacement. Finally a regularized contact force together with an energy-conserving time-stepping scheme are implemented. Eigenfrequencies and damping parameters of the string can be adjusted at ease, and in particular according to experimental measurements so that sound synthesis can be more realistic.

Numerical results of the scheme are illustrated by considering two different obstacles for synthesizing the sounds produced by a string vibrating against a point obstacle and a distributed obstacle, mimicking the bridge of the tanpura and a flat bridge respectively. Sound examples are available at
www.lam.jussieu.fr/Membres/Issanchou/Sounds_DAFx16.html.

2. MODEL SYSTEM

Consider a stiff string of length L (m), tension T (N/m), and with linear mass density μ (kg/m). Stiffness effects in the string are characterised by Young's modulus E (Pa) of the material, and the moment of inertia $I = \pi r^4/4$, where r is the string radius, in m. The string vibrates in the presence of an obstacle assumed not in contact with the string at rest and described by a fixed profile $g(x)$, $x \in [0, L]$. See Figure 1. Under the assumption of small displacements, the dynamics of the string is described by the following equation of motion:

$$\mu u_{tt}(x,t) - T u_{xx}(x,t) + EI u_{xxxx}(x,t) = f(x,t), \quad (1)$$

where $u(x,t)$ is the transverse displacement of the string in a single polarisation perpendicular to the barrier. Partial differentiation with respect to time t and coordinate x are indicated by multiple subscripts. Simply supported boundary conditions at the string endpoints are assumed:

$$u(0,t) = u(L,t) = u_{xx}(0,t) = u_{xx}(L,t) = 0, \qquad \forall t \in \mathbb{R}^+. \quad (2)$$

No damping is considered as yet, and a detailed model of loss will be introduced once modal analysis has been perfomed. See Section 3.1.

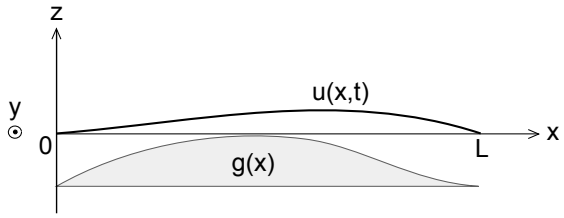


Figure 1: *A string of length L vibrating against an obstacle $g(x)$.*

$f(x,t)$ represents the contact force of the barrier upon the string. In general, it should be positive at times and locations along the string for which the string and barrier are in contact. A power-law expression is selected for the contact force. This choice has already been shown to be successful in the realm of musical acoustics [3, 17, 16, 19, 20]. The interaction force thus reads:

$$f(x,t) = f(\eta(x,t)) = K\left[\eta(x,t)\right]_+^\alpha, \tag{3}$$

where $\alpha \geq 1$ is a constant value, $\eta(x,t) = g(x) - u(x,t)$ is a measure of interpenetration of the string into the barrier, and $[\eta]_+ = 0.5\left(\eta + |\eta|\right)$ is the positive part of η [3]. Such a power-law model of a repelling force due to a collision may be interpreted as a compression of the medium in the case of flexible objects (such as, e.g., a piano hammer [21]). In the present case, it is best thought of as a regularized (smooth) force penalising such interpenetration [22]. As such, the constant K ideally will take on a very large value. This regularized approach contrasts with nonsmooth methods for which no penetration is allowed [23, 24].

One particular advantage of the present choice relies in the fact that the force f derives from a potential ψ:

$$f = \frac{d\psi}{d\eta}, \quad \text{with} \quad \psi(\eta) = \frac{K}{\alpha+1}\left[\eta\right]_+^{\alpha+1}. \tag{4}$$

This property is of special interest in the design of energy-conserving numerical methods. See Section 3.3.

2.1. Energy Balance

Energy techniques play an important role in the construction of numerical methods for highly nonlinear systems, as in the present case of the string in contact with a barrier. For instance, it is used in the present case of distributed collisions in [3], in the case of finite difference schemes. The basic strategy is to associate, with a given numerical method, a numerical conserved or dissipated energy quantity, which is itself a positive semi-definite function of the state. As such, it can be used to bound the dynamics of the system, and to find sufficient numerical stability conditions.

The continuous energy expression associated with (1) is obtained by multiplying (1) by u_t and then employing integration by parts over the spatial domain. It may be written as:

$$\mathscr{H} = \int_0^L \left[\frac{\mu}{2}(u_t)^2 + \frac{T}{2}(u_x)^2 + \frac{EI}{2}(u_{xx})^2 + \psi\right] dx. \tag{5}$$

It satisfies $\mathscr{H} \geq 0$ and the following equality:

$$\frac{d\mathscr{H}}{dt} = 0, \tag{6}$$

implying that energy is conserved. The first three terms in the expression correspond to stored energy due to the effects of inertia, tension and stiffness, respectively. The final term denotes the energy stored in the collision mechanism. Note that, as losses have not yet been introduced, the system is Hamiltonian. When losses are introduced, one should expect a balance of the form

$$\frac{d\mathscr{H}}{dt} + \mathscr{Q} = 0, \tag{7}$$

for some function $\mathscr{Q}(t) \geq 0$, with the interpretation of power loss, implying that

$$0 \leq \mathscr{H}(t) \leq \mathscr{H}(0) \tag{8}$$

for $t \geq 0$. It is not easy to give a simple expression for $\mathscr{Q}$ in the case of realistic models of loss in strings, which are usually expressed in the frequency domain [14], and not in terms of a spatiotemporal PDE system. Thus our expression for power loss is postponed until a modal analysis has been carried out. See Section 3.1.

3. NUMERICAL SCHEME FOR A STRING VIBRATING AGAINST AN OBSTACLE

The main characteristics of the numerical scheme are the following:
1. An exact scheme for a lossy linear oscillator without obstacle, as described in in [19], is used as a building block.
2. Each mode of the string can be described, in isolation, with such an oscillator. Therefore we apply the exact scheme to each mode, ensuring a fine description of frequencies and losses, adjusted for each mode. Then we add a force term F (corresponding to a modal representation of f in equation (1)). At this point, we obtain an equation in terms the modal coefficients of u.
3. Taking as many modes as interior points of the spatial mesh, we can rewrite the equation on u directly, through Fourier transformation. Then the force term is expressed as in [3], in order to obtain a conservative scheme.

3.1. Modal analysis

The modal expansion for the displacement of the string is as follows:

$$u(x,t) = \sum_{j=1}^{\infty} q_j(t)\phi_j(x), \text{ with } \phi_j(x) = \sqrt{\frac{2}{L}}\sin\left(\frac{j\pi x}{L}\right) \tag{9}$$

for simply supported boundary conditions.

Inserting the expansion of u in (1), one obtains:

$$\mu(\ddot{\mathbf{q}} + \mathbf{\Omega}^2\mathbf{q}) = \mathbf{F}, \tag{10}$$

where $\mathbf{q}$ is a vector containing modal coefficients, and $\mathbf{\Omega}$ is a diagonal matrix such that $\Omega_{j,j} = \omega_j = 2\pi\nu_j$.
Eigenfrequencies are given by $\nu_j = j\frac{c}{2L}\sqrt{1+Bj^2}$, where $B = \frac{\pi^2 EI}{TL^2}$ describes the inharmonicity created by taking into account the stiffness of the string. Finally the vector $\mathbf{F}$ represents the modal projection of the contact force, with $F_j = \int_0^L f(x,t)\phi_j(x)\mathrm{d}x$.

Equation (10) describes a lossless string, where the linear part corresponds to the description of a lossless oscillator for each mode. Therefore, losses can now be introduced by associating each mode with a lossy oscillator. Then (10) becomes [19]:

$$\mu(\ddot{\mathbf{q}} + \mathbf{\Omega}^2\mathbf{q} + 2\mathbf{\Sigma}\dot{\mathbf{q}}) = \mathbf{F}, \tag{11}$$

where $\mathbf{\Sigma}$ is a diagonal matrix such that $\Sigma_{j,j} = \sigma_j$. A damping parameter σ_j is now associated to each modal equation, and can be tuned at ease in order to consider any frequency dependence.

Let us now introduce the theoretical model for losses proposed in [14], which will allow us to determine realistic values to damping parameters in (11). This model takes into account the three main dissipation mechanisms in musical strings, namely friction with surrounding air, viscoelastic and thermoelastic behaviour of the material as internal losses. The following expression of the quality factor $Q_j = \frac{\pi\nu_j}{\sigma_j}$ has been given as:

$$Q_j^{-1} = Q_{j,air}^{-1} + Q_{j,ve}^{-1} + Q_{te}^{-1}, \tag{12}$$

where subscripts ve and te refer to viscoelastic and thermoelastic losses, and:

$$Q_{j,air}^{-1} = \frac{R}{2\pi\mu\nu_j}, R = 2\pi\eta_{air} + 2\pi d\sqrt{\pi\eta_{air}\rho_{air}\nu_j},$$
$$Q_{j,ve}^{-1} = \frac{4\pi^2\mu EI\delta_{ve}}{T^2}\nu_j^2.$$

In these expressions, η_{air} and ρ_{air} are, respectively, the dynamic viscosity coefficient and the air density. In the rest of the paper, they are set to the following values, assumed constant: $\eta_{air} = 1.8 \times 10^{-5}$ kg m^{-1}s^{-1} and $\rho_{air} = 1.2$ kg m^{-3}. Finally this loss model depends on two parameters that can be fitted from e.g. experimental measurements, as performed for example in [14, 25, 20]: the viscoelastic losses angle δ_{ve}, and the constant value Q_{te}^{-1} characterizing the thermoelastic damping. It results in a frequency-dependent loss model which accurately takes into account the damping mechanism present in musical string vibrations [14, 25, 20].

3.2. Spatial discretization

The spatial discretization is defined as $x_i = \frac{iL}{N}$, $i \in \{0, ..., N\}$. Boundary conditions are $u(x_0, t) = 0$ and $u(x_N, t) = 0\ \forall t \in \mathbb{R}^+$. In the following, only the values of u on the grid with $i \in \{1, 2, ..., N-1\}$ are thus needed in any calculation.

Considering $N-1$ modes, the following relationship is fulfilled, $\forall i \in \{1, 2, ..., N-1\}$:

$$u_i(t) = \sum_{j=1}^{N-1} q_j(t)\phi_j(x_i) = \sum_{j=1}^{N-1} q_j(t)\sqrt{\frac{2}{L}}\sin\left(\frac{j\pi i}{N}\right). \tag{13}$$

This can be written in matrix form as $\mathbf{u} = \mathbf{S}\mathbf{q}$, where $S_{i,j} = \phi_j(x_i), \forall (i,j) \in \{1, ..., N-1\}^2$. The inverse of $\mathbf{S}$ can easily be calculated: $\mathbf{S}^{-1} = \frac{L}{N}\mathbf{S}^T$. This linear relation between $\mathbf{u}$ and $\mathbf{q}$ will be useful in the following analysis.

3.3. Time discretization

Some notations are first introduced:

$$u_i^n = u(n\Delta t, x_i)$$
$$\delta_{t-}\mathbf{u}^n = \frac{\mathbf{u}^n - \mathbf{u}^{n-1}}{\Delta t}$$
$$\delta_{t+}\mathbf{u}^n = \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t}$$
$$\delta_{t\cdot}\mathbf{u}^n = \frac{\mathbf{u}^{n+1} - \mathbf{u}^{n-1}}{2\Delta t}$$
$$\delta_{tt}\mathbf{u}^n = \frac{\mathbf{u}^{n+1} - 2\mathbf{u}^n + \mathbf{u}^{n-1}}{\Delta^2}$$
$$\langle \mathbf{u}, \mathbf{v} \rangle = \Delta x \sum_{j\in\{1,...,N-1\}} u_j v_j.$$

Let us now consider the following time discretization of (11):

$$\frac{\mu}{\Delta t^2}(\mathbf{q}^{n+1} - \mathbf{C}\mathbf{q}^n + \tilde{\mathbf{C}}\mathbf{q}^{n-1}) = \mathbf{F}^n, \tag{14}$$

where $\mathbf{C}$, $\tilde{\mathbf{C}}$ are diagonal matrices such that :

$$C_{i,i} = e^{-\sigma_i\Delta t}\left(e^{\sqrt{\sigma_i^2 - \omega_i^2}\Delta t} + e^{-\sqrt{\sigma_i^2 - \omega_i^2}\Delta t}\right)$$
$$\tilde{C}_{i,i} = e^{-2\sigma_i\Delta t}.$$

When the contact force is not present, the modal approach may be viewed as an assembly of independent linear oscillators. This temporal integration scheme gives an exact solution in this case, as shown for example in [19], ensuring that at least the linear part is well-approximated (indeed, perfectly). The contact force, through which modes are coupled, remains to be determined. In order to avoid the difficulty linked to this coupling, the dynamical equation for the modal displacements vector $\mathbf{q}$ is rewritten for the physical displacement vector $\mathbf{u}$. Thanks to the linear relationship stated above between $\mathbf{u}$ and the modal displacement $\mathbf{q}$, the discrete equation on u can be written as:

$$\frac{\mu}{\Delta t^2}(\mathbf{u}^{n+1} - \mathbf{D}\mathbf{u}^n + \tilde{\mathbf{D}}\mathbf{u}^{n-1}) = \mathbf{f}^n, \tag{15}$$

where $\mathbf{D} = \mathbf{S}\mathbf{C}\mathbf{S}^{-1}$ and $\tilde{\mathbf{D}} = \mathbf{S}\tilde{\mathbf{C}}\mathbf{S}^{-1}$. The contact force expression used in the present approach is the same as in [3]: $\mathbf{f}^n = -\frac{\delta_{t-}\psi^{n+\frac{1}{2}}}{\delta_{t\cdot}\mathbf{u}^n}$, where $\psi^{n+\frac{1}{2}} = \frac{1}{2}(\psi^{n+1} + \psi^n)$ and $\psi^n = \psi(\eta^n)$. This formulation allows for a conservative scheme when there is no loss, and a dissipative one otherwise.

Therefore, at each time step, the following equation must be solved:

$$\mathbf{r} + \mathbf{b} + m\frac{\psi(\mathbf{r} + \mathbf{a}) - \psi(\mathbf{a})}{\mathbf{r}} = 0, \tag{16}$$

where $\mathbf{r} = \mathbf{u}^{n+1} - \mathbf{u}^{n-1}$ is the unknown, $\mathbf{a} = \mathbf{u}^{n-1}$, $m = \frac{\Delta t^2}{\mu}$ and $\mathbf{b} = -\mathbf{D}\mathbf{u}^n + \tilde{\mathbf{D}}\mathbf{u}^{n-1} + \mathbf{u}^{n-1}$. The Newton-Raphson algorithm may be used to this end [3].

Compared to a finite difference approach, the major advantage is the consideration of damping parameters and frequencies that can be adjusted mode by mode. However, the matrices $\mathbf{D}$, $\tilde{\mathbf{D}}$ are full in the modal case, as opposed to sparse in the case of local finite difference approximations, and computation time increases accordingly (see section 4.2).

Numerical energy analysis and stability conditions for the scheme are provided in the next section.

3.4. Stability analysis

The numerical scheme (15) is in a form close to actual implementation. However, in order to derive the discrete energy associated to the scheme, it is more convenient to rewrite it in terms of discrete temporal operators. Following the given in the case of an oscillator in [19] to get an equivalent scheme, one obtains:

$$\mu \left[\check{\mathbf{C}}_\mathbf{1} \delta_{tt} \mathbf{q}^n + \check{\mathbf{C}}_\mathbf{2} \mathbf{q}^n + \check{\mathbf{C}}_\mathbf{3} \delta_{t.} \mathbf{q}^n \right] = \mathbf{F}^n, \tag{17}$$

with $\check{\mathbf{C}}_\mathbf{1}$, $\check{\mathbf{C}}_\mathbf{2}$ and $\check{\mathbf{C}}_\mathbf{3}$ diagonal matrices satisfying:

$$\check{C}_{1_{ii}} = \frac{1 + (1-\gamma_i)\frac{\omega_i^2 \Delta t^2}{2}}{1 + (1-\gamma_i)\frac{\omega_i^2 \Delta t^2}{2} + \sigma_i^* \Delta t}$$

$$\check{C}_{2_{ii}} = \frac{\omega_i^2}{1 + (1-\gamma_i)\frac{\omega_i^2 \Delta t^2}{2} + \sigma_i^* \Delta t}$$

$$\check{C}_{3_{ii}} = \frac{2\sigma_i^*}{1 + (1-\gamma_i)\frac{\omega_i^2 \Delta t^2}{2} + \sigma_i^* \Delta t}.$$

The coefficients γ_i and σ_i^* may be written as

$$\gamma_i = \frac{2}{\omega_i^2 \Delta t^2} - \frac{A_i}{1 + e_i - A_i}$$

$$\sigma_i^* = \left(\frac{1}{\Delta t} + \frac{\omega_i^2 \Delta t}{2} - \gamma_i \frac{\omega_i^2 \Delta t}{2} \right) \frac{1 - e_i}{1 + e_i}$$

where

$$A_i = e^{-\sigma_i \Delta t} \left(e^{\sqrt{\sigma_i^2 - \omega_i^2} \Delta t} + e^{-\sqrt{\sigma_i^2 - \omega_i^2} \Delta t} \right) \text{ and } e_i = e^{-2\sigma_i \Delta t}. \tag{18}$$

The scheme for the displacement u thus may be written as:

$$\mu \left[\check{\mathbf{D}}_\mathbf{1} \delta_{tt} \mathbf{u}^n + \check{\mathbf{D}}_\mathbf{2} \mathbf{u}^n + \check{\mathbf{D}}_\mathbf{3} \delta_{t.} \mathbf{u}^n \right] = \mathbf{f}^n, \tag{19}$$

where $\check{\mathbf{D}}_\mathbf{1} = \mathbf{S}\check{\mathbf{C}}_\mathbf{1}\mathbf{S}^{-1}$, $\check{\mathbf{D}}_\mathbf{2} = \mathbf{S}\check{\mathbf{C}}_\mathbf{2}\mathbf{S}^{-1}$ and $\check{\mathbf{D}}_\mathbf{3} = \mathbf{S}\check{\mathbf{C}}_\mathbf{3}\mathbf{S}^{-1}$. The force term is expressed as previously.

A discrete energy balance can be obtained by taking the inner product between equation (19) and $\delta_{t.}\mathbf{u}^n$:

$$\delta_{t-} \mathbf{H}^{n+\frac{1}{2}} = -\mu \left\langle \delta_{t.} \mathbf{u}^n, \check{\mathbf{D}}_\mathbf{3} \delta_{t.} \mathbf{u}^n \right\rangle. \tag{20}$$

where:

$$\mathbf{H}^{n+\frac{1}{2}} = \frac{\mu}{2} \left\langle \delta_{t+} \mathbf{u}^n, \check{\mathbf{D}}_\mathbf{1} \delta_{t+} \mathbf{u}^n \right\rangle + \frac{\mu}{2} \left\langle \mathbf{u}^{n+1}, \check{\mathbf{D}}_\mathbf{2} \mathbf{u}^n \right\rangle + \left\langle \psi^{n+\frac{1}{2}}, 1 \right\rangle. \tag{21}$$

$\check{\mathbf{D}}_\mathbf{3}$ is positive semi-definite, so that the scheme is strictly dissipative. Therefore it is stable if the energy is positive.

Since the force potential is non-negative, the stability condition is given by:

$$\left\langle \delta_{t+} \mathbf{u}^n, (\check{\mathbf{D}}_\mathbf{1} - \frac{\Delta t^2}{4} \check{\mathbf{D}}_\mathbf{2}) \delta_{t+} \mathbf{u}^n \right\rangle \geq 0. \tag{22}$$

It is therefore sufficient to have $(\check{\mathbf{D}}_\mathbf{1} - \frac{\Delta t^2}{4} \check{\mathbf{D}}_\mathbf{2})$ positive semi-definite, which is true if $(\check{\mathbf{C}}_\mathbf{1} - \frac{\Delta t^2}{4} \check{\mathbf{C}}_\mathbf{2})$ is positive semi-definite. Consequently, the condition that must be satisfied may be written as:

$$\frac{1 + e_i + A_i}{1 + e_i - A_i} \geq 0 \quad \forall i. \tag{23}$$

Equation (23) is satisfied if $1 + e_i \pm A_i > 0$.This is always true, and hence the scheme is unconditionally stable. The same conclusion is obtained in the limiting case $\sigma_i = 0 \ \forall i$, using the same reasoning leading to 22 with a reduced expression of γ_i.

3.5. Contact losses

Nonlinear losses due to contact can be added in the presented framework, following the considerations developed in [26, 3]. The contact force given by (3) may be modified as:

$$f = \frac{d\psi}{d\eta} - \frac{du}{dt} K \beta [\eta]_+^\alpha, \tag{24}$$

with $\beta \geq 0$.

The energy (5) of the system (with no loss inherent to the string) then satisfies [3]:

$$\frac{d\mathscr{H}}{dt} = -\mathscr{Q}_{contact} \tag{25}$$

where

$$\mathscr{Q}_{contact} = \int_0^L (u_t)^2 K \beta [\eta]_+^\alpha dx. \tag{26}$$

The additional damping term may be discretised using the following expression [3]: $\delta_{t.}\mathbf{u}^n K \beta [\eta^n]_+^\alpha$.

Instead of (16), the equation to be solved at each time step is then:

$$(1 + c)\mathbf{r} + \mathbf{b} + m \frac{\psi(\mathbf{r} + \mathbf{a}) - \psi(\mathbf{a})}{\mathbf{r}} = 0, \tag{27}$$

where $c = \frac{\Delta t}{2\mu} K \beta [\mathbf{g} - \mathbf{u}^n]_+^\alpha$. The discrete energy balance is given by:

$$\delta_{t-} \mathbf{H}^{n+\frac{1}{2}} = -\mu \left\langle \delta_{t.} \mathbf{u}^n, \check{\mathbf{D}}_\mathbf{3} \delta_{t.} \mathbf{u}^n \right\rangle - \left\langle \delta_{t.} \mathbf{u}^n, \delta_{t.} \mathbf{u}^n K \beta [\eta^n]_+^\alpha \right\rangle. \tag{28}$$

Since $\left\langle \delta_{t.} \mathbf{u}^n, \delta_{t.} \mathbf{u}^n K \beta [\eta^n]_+^\alpha \right\rangle \geq 0$, the dissipation in the system is then increased.

4. APPLICATION TO MUSICAL INSTRUMENTS AND SOUND SYNTHESIS

In this section, the numerical scheme presented previously is used to simulate the motion of a string vibrating against an obstacle. The cases of a point obstacle and a distributed obstacle are considered.

The string to be considered here is of length $L = 1.002$ m, under tension $T = 180.5$ N and with linear mass density $\mu = 1.17 \times 10^{-3}$ kg/m. Musical strings are known to have a very small stiffness. Consequently, the inharmonicity coefficient B introduced in Section 3.1 is chosen to be $B = 1.78 \times 10^{-5}$, corresponding to measured values according to the protocol described in [20]. In order to show the ability of the model to incorporate a complex damping law, the theoretical loss model introduced in section 3.1 is used. The values $\delta_{ve} = 0.0045$ and $Q_{te}^{-1} = 0.000203$ have been chosen, and correspond to experimental data, obtained using the method exposed in [20]. This model has a complex frequency dependence and accounts for the main loss mechanisms present in strings.

First, a time step, or sample rate has to be chosen. To this end, a convergence study has been undertaken, as elaborated in the following section.

4.1. Convergence study

Contact problems lead to the spontaneous generation of high frequencies in the system, due to the very short timescales resulting from collisions between very stiff objects, as in the present case. This implies, at least in theory, a need for a high sampling rate in order to obtain accurate results, which should decrease as the contact stiffness does. The presence of damping, particularly at high frequencies, is expected to be an ameliorating factor. In order to highlight such accuracy issues, as they relate to the time step, convergence results are presented here.

Consider a point obstacle located at $x = 6$ mm (see Section 4.3). In Figure 2, time history and spectrograms of simulation output are shown, where output is taken as the velocity at the bridge. For the contact restoring force, $K = 10^{13}$ and $\alpha = 1.3$ in order to obtain a stiff contact, limiting penetration of the string in the obstacle to the range of 1×10^{-8} m. With $K = 10^9$ for instance, the penetration attains about 1×10^{-5} m and the resulting sound richness is significantly altered. The string is initially plucked at $x = L/2$, with a maximal displacement of 1.8 mm.

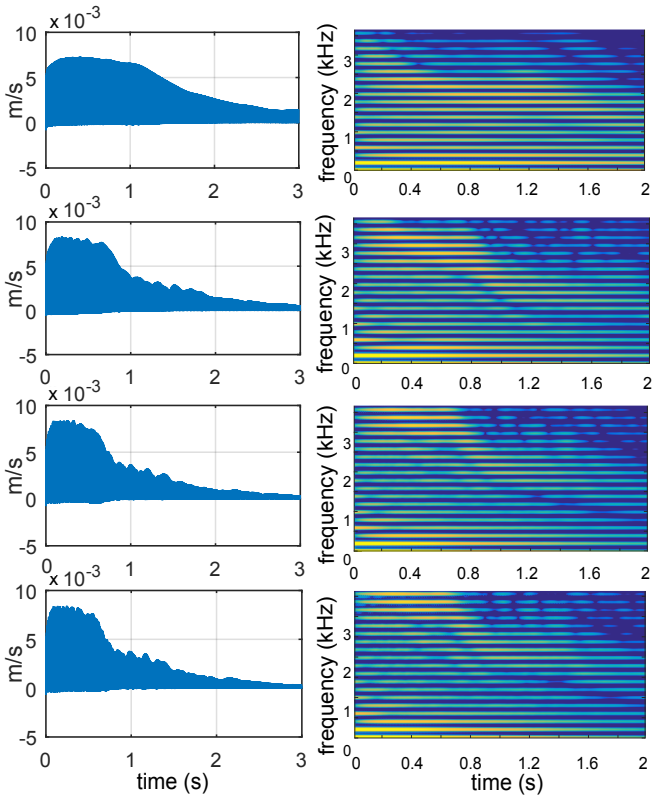


Figure 2: *At left : time-domain output signals. At right : spectrograms. From top to bottom :* $F_s = 128, 256, 512$ *and* 1024 *kHz.*

Figure 2 clearly highlights that for a sample rate $F_s = 1/\Delta t = 128$ kHz, the simulated sound is far from convergence (it is substantially different from the result with $F_s = 1024$ kHz), and thus not reliable. From $F_s = 256$ kHz, the general shape of the time history, together with the spectral dynamics, seems to be correctly reproduced. However, detailed view inside the signal and auditory comparisons definitely evidenced that the high frequencies components are not well reproduced, such that this sample rate is still insufficient. Finally, careful inspection crossed with listening tests shows that a sample rate of at least $F_s = 1$ MHz is necessary for convergence of numerical results. To be confident with the accuracy of the simulations presented below, the sampling rate has been fixed for all the simulations to $F_s = 2$ MHz. Note also that careful comparisons with experimental results have been presented in [20], for which a sample rate of 2 MHz was also necessary to obtain very satisfactory results over long simulation times.

4.2. Computation cost

In this part, computation times of the numerical scheme are discussed. The computations have been realized with MATLAB on a single CPU with a clock at 2.4 GHz, and time costs are given in table 1 for the computation of one second of sound, rounded up to the nearest minute when $N = 1001$. Steps which are controlled are the Newton-Raphson loop, the computation of $\mathbf{b}, \mathbf{a}, \mathbf{r}^n$ in (16), the computation of the energy given in (21) and finally the total time is given. It clearly appears that the most costly steps are

$N-1=500$				
F_s	44.1 kHz	88.2 kHz	176 kHz	1 MHz
Newton-Raphson	0.7	1.3	2.6	9.8
$\mathbf{b}, \mathbf{a}, \mathbf{r}^n$ in (16)	1.5	3.1	6	31.4
Energy	0.8	1.7	3.5	16.8
Total time	3.1	6.4	12.5	59.5
$N-1=1001$				
F_s	44.1 kHz	88.2 kHz	176 kHz	1 MHz
Newton-Raphson	1	2	4	14
$\mathbf{b}, \mathbf{a}, \mathbf{r}^n$ in (16)	5	7	14	119
Energy	3	6	12	54
Total time	10	16	30	189

Table 1: *Computation times, in minutes, for* $N - 1 = 500$ *and* $N - 1 = 1001$.

the computation of $\mathbf{b}, \mathbf{a}, \mathbf{r}^n$ and the energy, i.e. products of matrices and vectors, most probably because involved matrices are full. Therefore, the computation time is mostly driven by N, so that a judicious way of reducing computation cost may be to adapt the spatial grid to the obstacle, making the space grid finer around the obstacle and larger elsewhere.

4.3. Point obstacle: case of the tanpura

The tanpura is an Indian instrument which is played by plucking open strings. The strings are connected to a curved bridge over which a thread is carefully installed makes the sound very specific to this instrument. This bridge and its thread (fully modelled in [15]) can mostly be described as a two point bridge [14]. Therefore in this part, this two point bridge is considered: assuming that the bridge is at $x = 0$, a point obstacle is located at $x = 6$ mm. Two initial conditions are considered, by plucking the string at $x = L/5$ and $x = 4L/5$, and the velocity of the string is computed at the bridge.

Parameters of the contact force are $K = 10^{13}$ and $\alpha = 1.3$ as in Section 4.1, and the spatial grid is such that $N = 1001$. At the initial time a smoothed triangle with a maximal amplitude of about 1 mm is imposed as the initial displacement, with no velocity. This small value has been chosen in order to show that even for very

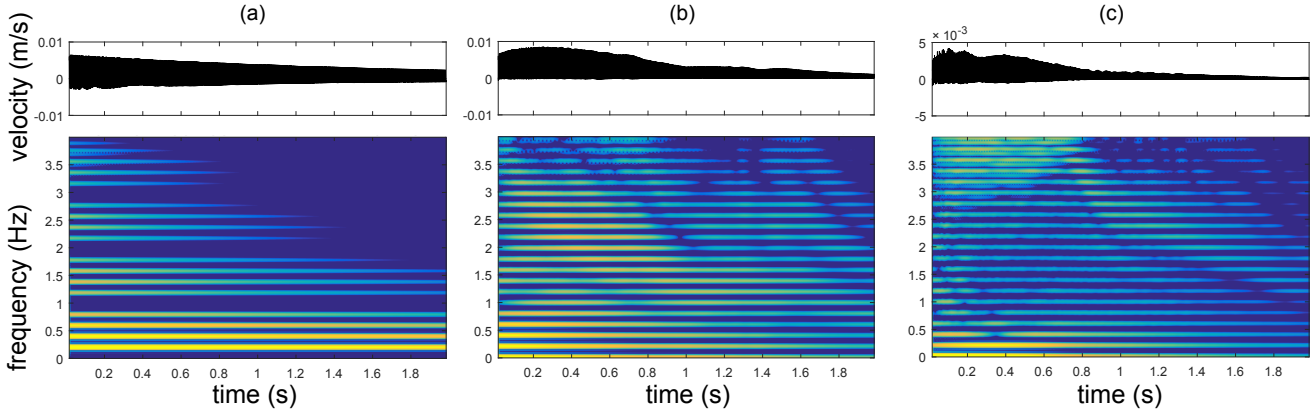


Figure 3: *Spectral evolution of velocities at the bridge (a) no obstacle, plucking at $x = L/5$ (b) point obstacle, plucking at $x = L/5$ (c) point obstacle, plucking at $x = 4L/5$.*

small displacements, the contact can introduce strong nonlinear effects which are clearly audible and visible on signals.

When there is no obstacle, one can observe rejection of modes which are not excited by the initial condition, as well as a faster damping of high frequencies compared to low frequencies (see Figure 3). When the obstacle is added, all modes are present and energy is transmitted from modes to others. Spectrum tendencies are similar to those encountered in [15] (numerical results) and [14] (experimental results).

The velocity signal for the string with an obstacle is mostly positive (see Figure 4), this is due to the close position of the measure position to the obstacle. Moreover, the effect of the string stiffness implies dispersion which is clearly visible on temporal signals, and constitutes a precursor. When the plucked point is at $x = 4L/5$ rather than $x = L/5$, the precursor needs more time to reach the obstacle, therefore it is much more developed when arriving, which explains the high frequencies richness of the temporal signal in Figure 4.

Moreover, according to [14], the frequency sliding depends on the plucking position, which is also observed here.

The temporal evolution of energy $\mathbf{H}^{n+\frac{1}{2}}$ is presented in Figure 5. It decreases faster when there is an obstacle, probably because energy is transmitted to higher modes, which are more damped. For a plucking point at $x = 4L/5$, the energy decreases faster than at $x = L/5$, since more high frequencies are generated by the bridge.

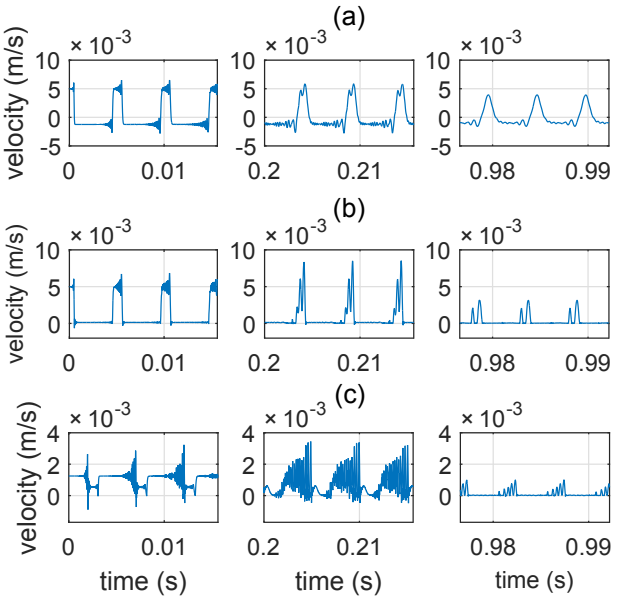


Figure 4: *Temporal evolution of velocities at the bridge for a point obstacle mimicking the case of the tanpura (a) no obstacle, plucking at $x = L/5$ (b) point obstacle, plucking at $x = L/5$ (c) point obstacle, plucking at $x = 4L/5$.*

4.4. Distributed obstacle

In this part, a distributed parabolic obstacle is considered at one end of the string, that could mimic the case of a sitar. Its shape is as follows:

$$g(x) = -ax^2, \tag{29}$$

where we set $a = 0.0065$. The length of the obstacle is 19 mm (see Figure 6). Such a value, small compared to realistic ones (see for instance [13], where a measure gives $a = 0.5778$), is chosen in order to make the contact between the string and the full length of the bridge arise. No particular adjustment is made in terms of inclination, however interesting observations can already be made.

Numerical parameters K, α and the initial condition are the same as previously. Temporal and spectral evolutions are presented in Figures 7 and 8, as well as the energy of the signal. Because of the obstacle, the velocity signal at the bridge as a minimum value close to 0.

Similarly to the case of the point obstacle, a precursor is visible from the first periods and high frequencies are highly excited by the contact, which results in narrow peaks on temporal signals (Figure 7). As mentioned in [12] for experimental results on a complete instrument, the absence of rejection and a descending formant can clearly be observed in Figure 8. Moreover, as in the point obstacle case, the energy decrease is faster than when there is no obstacle, which may be due to the transfer of energy from lower to higher modes, combined with larger damping at high frequencies.

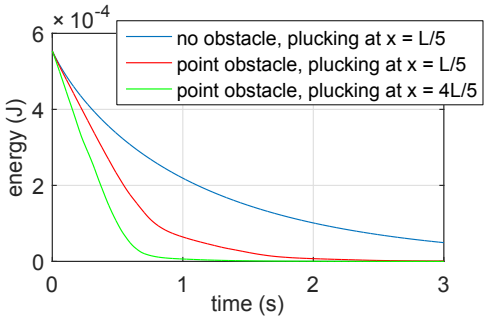


Figure 5: *Energy decay for different configurations.*

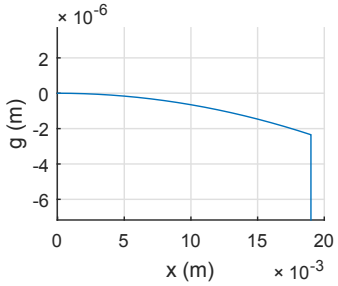


Figure 6: *Shape of the distributed obstacle.*

5. CONCLUSION

A numerical method combining a modal approach and an energy-conserving scheme for the case of the string in contact with a barrier has been introduced. Due to the modal approach, the linear parameters of the strings (eigenfrequencies and damping coefficients) can be set independently for each mode, allowing for flexible control over frequency-dependent loss. In particular, linear characteristics of a measured string can be used in order to obtain very realistic sounds, as proposed in [20]. Such complete control over damping rates for the string constitutes one of the major advantages of a modal approach to synthesis. The scheme itself, though with modal characteristics, operates ultimately in the spatial domain, much like a finite difference scheme, though with the special property of being exact under linear conditions. Such a method can thus be viewed as a type of spectral method [27], accompanied by a time-stepping method which is tuned according to the modal frequencies (unlike, e.g., more typical methods such as leap-frog, or members of the Runge-Kutta family). As with spectral methods, though the updates are no longer sparse, as in the case of FD schemes, it may be possible to employ fast transforms (such as a variant of the FFT) in order to perform the updates (which in this case, follows directly from the sinusoidal structure of the matrix $\mathbf{S}$).

It has been shown that a high sample rate is necessary in order to obtain reliable results for simulations over a long duration. Computation time, which does not allow a real-time simulation at present on a standard machine, could be improved by considering a variable spatial step, finer in the region surrounding the obstacle, and possibly a variable time step [28], since high temporal precision is only necessary when the string is in contact with the obstacle. In audio applications, though, such algorithmic refinements must be treated with care—the use of a variable grid spacing will entail a loss of structure in the resulting update matrices, and the use of a variable time step must necessarily be accompanied by some form of on-line sample rate conversion which is perceptually transparent.

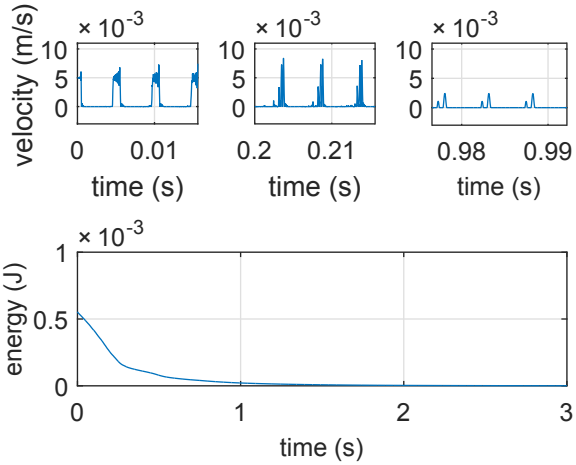


Figure 7: *Temporal evolution of the velocity at the bridge and its energy*

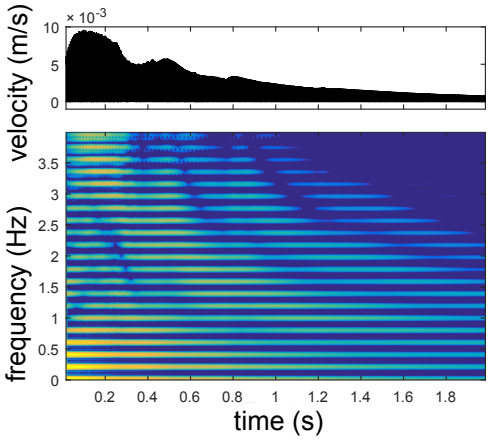


Figure 8: *Spectral evolution of the velocity at the bridge, distributed obstacle*

The next steps of this research will consider more closely the comparison between the outcomes of these numerical methods and measurements realized on a real string and/or on real instruments. To this end, and also in the interest of higher quality synthesis, a parametric study on the contact force parameters should be carried out, and the incorporation of various additional features of the string/barrier system would be of interest. One such feature is the extension of string vibration to two polarisations, as recently explored in the context of bowed string synthesis [29] and in case of the tanpura [30]. At present, there is not a model of an excitation mechanism, and a refined plucking model could be included, perhaps modeling the dynamics of the player's fingers [31]. Ultimately, a complete instrument will require a model of coupling to the instrument body, acoustic radiation, and possibly sympathetic strings [32].

6. ACKNOWLEDGMENTS

This work was supported by the European Research Council, under grant number ERC-2011-StG-279068-NESS.

7. REFERENCES

[1] X. Boutillon, "Model for piano hammers: Experimental determination and digital simulation," *J. Acoust. Soc. Am.*, vol. 83, no. 2, pp. 746–754, 1988.

[2] A. Chaigne, P. Joly, and L. Rhaouti, "Numerical modeling of the timpani," in *European Congress on Computational Methods in Applided Sciences and Engineering, Barcelona*, 2000.

[3] S. Bilbao, A. Torin, and V. Chatziioannou, "Numerical modeling of collisions in musical instruments," *Acta Acustica united with Acustica*, vol. 101, pp. 155–173, 2015.

[4] H. Cabannes, "Cordes vibrantes avec obstacles," *Acustica*, vol. 55, pp. 14–20, 1984.

[5] M. Schatzman, "A hyperbolic problem of second order with unilateral constraints: The vibrating string with a concave obstacle," *Journal of Mathematical Analysis and Applications*, vol. 73, pp. 138–191, 1980.

[6] R. Burridge, J. Kappraff, and C. Morshedi, "The sitar string, a vibrating string with a one-sided inelastic constraint," *SIAM Journal of Applied Mathematics*, vol. 42, no. 6, pp. 1231–1251, 1982.

[7] E. Rank and G. Kubin, "A waveguide model for slapbass synthesis," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, 1997, vol. 1, pp. 443–446.

[8] G. Evangelista and F. Eckerholm, "Player-instrument interaction models for digital waveguide synthesis of guitar: Touch and collisions," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 4, pp. 822–832, 2010.

[9] D. Kartofelev, A. Stulov, H.-M. Lehtonen, and V. Välimäki, "Modeling a vibrating string terminated against a bridge with arbitrary geometry," in *Proceedings of the Stockholm Music Acoustics Conference*, 2013.

[10] D. Kartofelev, A. Stulov, and V. Välimäki, "Pitch glide effect induced by a nonlinear string-barrier interaction," in *AIP Conference Proceedings*, 2015.

[11] A. Krishnaswamy and J. O. Smith, "Methods for simulating string collisions with rigid spatial objects," in *Proc. IEEE Workshop of Applications of Signal Processing to Audio and Acoustics*, 2003, pp. 233–236.

[12] S. Siddiq, "A physical model of the nonlinear sitar string," *Archives of acoustics*, vol. 37, no. 1, pp. 73–79, 2012.

[13] C. P. Vyasarayani, S. Birkett, and J. McPhee, "Modeling the dynamics of a vibrating string with a finite distributed unilateral constraint: Application to the sitar," *J. Acoust. Soc. Am.*, vol. 125, no. 6, pp. 3673–3682, 2009.

[14] C. Valette and C. Cuesta, *Mécanique de la corde vibrante*, Hermès, 1993.

[15] V. Chatziioannou and M. van Walstijn, "Numerical simulation of tanpura string vibrations," *ISMA*, pp. 609–614, 2014, Le Mans, France.

[16] V. Chatziioannou and M. van Walstijn, "Energy conserving schemes for the simulation of musical instrument contact dynamics," *Journal of Sound and Vibration*, vol. 339, pp. 262–279, 2015.

[17] S. Bilbao and A. Torin, "Numerical simulation of string/barrier collisions: the fretboard," in *Int. Conference on Digital Audio Effects (DAFx-14)*, 2003.

[18] L. Trautmann and R. Rabenstein, "Multirate simulations of string vibrations including nonlinear fret-string interactions using the functional transformation method," *EURASIP Journal on Applied Signal Processing*, vol. 7, pp. 949–963, 2004.

[19] S. Bilbao, *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*, Wiley, 2009.

[20] C. Issanchou, S. Bilbao, O. Doaré, J.-L. Le Carrou, and C. Touzé, "Méthode modale mixte pour le contact unilatéral corde / obstacle : application au chevalet de la tampoura," in *Congrès Français d'Acoustique, Le Mans, France*, 2016.

[21] A. Chaigne and A. Askenfelt, "Numerical simulations of piano strings. i. a physical model for a struck string using finite difference methods," *J. Acoust. Soc. Am.*, vol. 95, no. 2, pp. 1112–1118, 1994.

[22] D. Harmon, "Robust, efficient, and accurate contact algorithms," PhD Thesis, Department of Computer Science, Columbia University, 2010.

[23] A. Banerjee, A. Chanda, and R. Das, "Historical origin and recent development on normal directional impact models for rigid body contact simulation: A critical review," *Archives of Computational Methods in Engineering*, pp. 1–26, 2016.

[24] B. Brogliato and V. Acary, *Numerical Methods for Nonsmooth Dynamical Systems. Applications in Mechanics and Electronics*, Springer Verlag, 2008.

[25] A. Paté, J.-L. Le Carrou, and B. Fabre, "Predicting the decay time of solid body electric guitar tones," *J. Acoust. Soc. Am.*, vol. 135, no. 5, pp. 3045–3055, 2014.

[26] K. Hunt and F. Crossley, "Coefficient of restitution interpreted as damping in vibroimpact," *J. Applied Mech.*, pp. 440–445, 1975.

[27] L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, 2000.

[28] P. Flores and J. Ambrósio, "On the contact detection for contact-impact analysis in multibody systems," *Multibody Syst Dyn*, vol. 24, pp. 103–122, 2010.

[29] C. Desvages and S. Bilbao, "Two-polarisation finite difference model of bowed strings with nonlinear contact and friction forces," in *Int. Conference on Digital Audio Effects (DAFx-15)*, 2015.

[30] J. Bridges and M. Van Walstijn, "Investigation of tanpura string vibrations using a two-dimensional time-domain model incorporating coupling and bridge friction," in *Proc. of the third Vienna Talk on Music Acoustics*, 2015.

[31] D. Chadefaux, J.-L. Le Carrou, and B. Fabre, "A model of harp plucking," *J. Acoust. Soc. Am.*, vol. 133, no. 4, pp. 2444–2455, 2013.

[32] J.-L. Le Carrou, F. Gautier, N. Dauchez, and J. Gilbert, "Modelling of sympathetic string vibrations," *Acta Acustica united with Acustica*, vol. 91, no. 2, pp. 277–288, 2005.

A REAL-TIME SYNTHESIS ORIENTED TANPURA MODEL

Maarten van Walstijn, Jamie Bridges, and Sandor Mehes

Sonic Arts Research Centre
School of Electronics, Electrical Engineering, and Computer Science
Queen's University Belfast, UK
{m.vanwalstijn,jbridges05,smehes01}@qub.ac.uk

ABSTRACT

Physics-based synthesis of tanpura drones requires accurate simulation of stiff, lossy string vibrations while incorporating sustained contact with the bridge and a cotton thread. Several challenges arise from this when seeking efficient and stable algorithms for real-time sound synthesis. The approach proposed here to address these combines modal expansion of the string dynamics with strategic simplifications regarding the string-bridge and string-thread contact, resulting in an efficient and provably stable time-stepping scheme with exact modal parameters. Attention is given also to the physical characterisation of the system, including string damping behaviour, body radiation characteristics, and determination of appropriate contact parameters. Simulation results are presented exemplifying the key features of the model.

1. INTRODUCTION

Among mechanically-induced sound effects naturally afforded by musical instruments, the generation of overtones in tanpura drone playing is one of the more spectacular and intriguing examples. In Indian musical tradition, the phenomenon is known as *jvari* (meaning 'life-giving'), and arises from the impactive interaction of the vibrating string with a hard-surfaced bridge. The player fine-tunes the effect by carefully positioning a thin thread between the bridge and the string (see Fig. 1).

As a vibrational phenomenon, the jvari effect has attracted scientific interest for almost a century, starting with the musical acoustics poineering work by Raman [1]. Several ways of analysing and modelling the vibrations of the tanpura and other 'flat-bridge' instruments such as the sitar, veena, and biwa have been proposed since, with the aims ranging from theoretical understanding (usually relying on simplifying assumptions regarding the nature of the interaction [2, 3, 4]) to more practical discrete-time simulation [5, 6] including several synthesis oriented studies [7, 8]. The problem also naturally bears some resemblance to various other cases involving collisions, including string-fingerboard contact in the guitar [9, 10], violin [11], and bass guitar [12, 13, 14].

Despite these advances, efficient and realistic synthesis of the sound of flat-bridge string instruments appears to have remained a somewhat elusive goal. One of the original difficulties, namely that of potential instability when incorporating collision forces, has recently been addressed more widely within a finite-difference context, by construction of time-stepping schemes that respect the energy balance inherent to the underlying continuous-domain model [15, 16, 17, 18]. Tanpura models based on such energy methods can reproduce the jvari effect by simulating distributed string-bridge collisions [17, 19, 20]. However the algorithms that implement these tanpura models are not particularly suited to sound synthesis because of the high computational burden resulting from the reliance on iterative solvers and from the high sample rates needed to alleviate numerical dispersion.

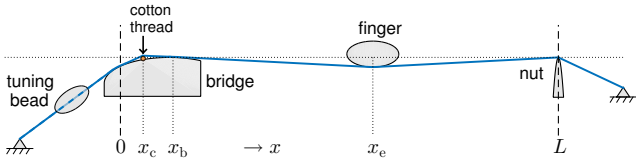


Figure 1: Schematic depiction of the tanpura string geometry (with altered proportions for clarity). The string termination points of the model are indicated with the vertical dashed lines.

This paper aims to formulate a leaner discrete-time tanpura string model requiring significantly reduced computational effort, but retaining much of the key sonic features of the instrument. Two aspects that distinguish this challenge somewhat from other cases of string-barrier interaction are (a) the sustained nature of the impactive interaction (with a high potential for audible high-frequency artefacts, including aliasing) and (b) the sensitivity of the jvari to some of the system parameters and to discretisation errors. The key features of the proposed model, presented in Section 2, can be summarised as follows:

- the spatially distributed string-bridge collision forces are suppressed to a single variable, which - in conjunction with neglecting contact damping and using a unity exponent in the contact law - allows updating the numerical system without the use of an iterative solver;
- the thread interaction, which effects a 'softer' string termination, is explicitly modelled as a local spring-damper connection;
- a modal expansion approach is utilised, which allows formulating a numerical model with exact modal frequencies and damping;
- discretisation is performed on a first-order partial derivative form of the modal differential equations, which facilitates the use of a two-point discrete gradient for discretisation of the bridge contact force;
- numerical stability is independent of the system parameters and the temporal step; the only numerical constraint is that the mode series is truncated at Nyquist in order to avoid mode aliasing.

For realistic synthesis of tanpura drones, one also needs to determine appropriate system parameters, including those related to string damping, bridge and thread contact, and sound radiation; this is discussed in Section 3. Exemplifying simulation results are then presented and discussed in Section 4, followed by concluding remarks and perspectives in Section 5.

2. TANPURA STRING MODEL

2.1. Model Equations

The transversal displacement $y(x,t)$ of the string depicted in Fig. 1, with the spatial domain defined as $x \in [0, L]$ and t denoting time, may be described by:

$$\rho A \frac{\partial^2 y}{\partial t^2} = T\frac{\partial^2 y}{\partial x^2} - EI\frac{\partial^4 y}{\partial x^4} - \gamma(\beta)\frac{\partial y}{\partial t} + \mathcal{F}_\mathrm{c}(x,t) + \mathcal{F}_\mathrm{b}(x,t) + \mathcal{F}_\mathrm{e}(x,t), \quad (1)$$

in which ρ, A, T, E, and I are mass density, cross-sectional area, tension, Young's modulus, and moment of inertia, respectively. Assuming simply supported ends, the boundary conditions are

$$y(x,t)\Big|_{x=0,L} = 0, \quad \frac{\partial^2 y}{\partial x^2}\Big|_{x=0,L} = 0. \quad (2)$$

Frequency-dependent string damping is incorporated by defining the parameter $\gamma(\beta)$ in (1) as:

$$\gamma(\beta) = 2\rho A\Big[\sigma_0 + \left(\sigma_1 + \sigma_3\beta^2\right)|\beta|\Big], \quad (3)$$

where β is the wave number and $\sigma_{0,1,3}$ are fit parameters. The interactions with the cotton thread, the bridge and a plucking finger are modelled using the force densities $\mathcal{F}_\mathrm{c}(x,t)$, $\mathcal{F}_\mathrm{b}(x,t)$ and $\mathcal{F}_\mathrm{e}(x,t)$, respectively. These are defined here in a simplified form by pre-determining their spatial distributions, hence modelling each as ($z = \mathrm{c}, \mathrm{b}, \mathrm{e}$):

$$\mathcal{F}_z(x,t) = \psi_z(x) F_z(t), \quad (4)$$

where $\psi_z(x)$ are spatial distribution functions of the form

$$\psi_z(x) = \begin{cases} \frac{\pi}{2w_z}\cos\left[\frac{\pi}{w_z}(x - x_z)\right] & : \quad x \in D_z \\ 0 & : \quad \text{otherwise} \end{cases} \quad (5)$$

in which $D_z = [x_z - \frac{1}{2}w_z, x_z - \frac{1}{2}w_z]$ denotes a spatial domain of width w_z and centre position x_z. Equation (5) is a good approximation to the force profile typically observed in the initial vibrations as computed with distrubuted contact models (see the left plot of Fig. 2). In addition this form provides a convenient way of exciting mainly the first mode of the string (by setting $x_\mathrm{e} = \frac{1}{2}L$, $w_\mathrm{e} = L$). The impactive contact with the bridge is modelled here using a lossless contact law with unity exponent and elasticity constant k_b:

$$F_\mathrm{b}(t) = k_\mathrm{b}\lfloor h_\mathrm{b} - y_\mathrm{b}(t)\rfloor, \quad (6)$$

in which $\lfloor y \rfloor$ denotes $u(y)\cdot y$, where $u(y)$ is the unit step function. Given that the string never detaches from the cotton thread, this interaction can be modelled as a simple spring-damper connection

$$F_\mathrm{c}(t) = k_\mathrm{c}\left[h_\mathrm{c} - y_\mathrm{c}(t)\right] - r_\mathrm{c}\frac{\partial y_\mathrm{c}}{\partial t}, \quad (7)$$

where r_c is a damping parameter. The term $y_z(t)$ ($z = \mathrm{b}, \mathrm{c}$) in equations (6) and (7) represents a spatially averaged value of the string displacement at x_z:

$$y_z(t) = \int_{x_z - w_z/2}^{x_z + w_z/2} \psi_\mathrm{z}(x)\, y(x,t)dx. \quad (8)$$

The contact potential energies are

$$V_\mathrm{c}(y_\mathrm{c}) = \frac{k_\mathrm{c}}{2}\left[h_\mathrm{c} - y_\mathrm{c}\right]^2, \quad V_\mathrm{b}(y_\mathrm{b}) = \frac{k_\mathrm{b}}{2}\lfloor h_\mathrm{b} - y_\mathrm{b}\rfloor^2, \quad (9)$$

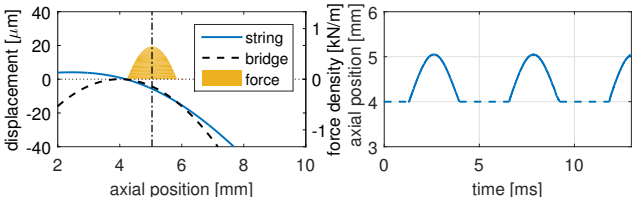


Figure 2: Left: string motion snapshot obtained with a model simulating distributed bridge contact [28]. The profile of the orange surface indicates force density, and the dash-dot line indicates the corresponding instantaneous central contact point. Right: Variation of the central contact point over the first 12ms. The flat dashed lines indicates periods of no contact.

where both height constants h_c and h_b are normally zero to ensure grazing contact at equilibrium. From the second equation it is straightforward to derive that

$$F_\mathrm{b}(t) = -\frac{\partial V_\mathrm{b}}{\partial y_\mathrm{b}}. \quad (10)$$

The forces exerted by the string at the left-end termination ('o') and the nut end ('n') are

$$F_\mathrm{o}(t) = T\frac{\partial y}{\partial x}\Big|_{x=0} - EI\frac{\partial^3 y}{\partial x^3}\Big|_{x=0}, \quad (11)$$

$$F_\mathrm{n}(t) = -T\frac{\partial y}{\partial x}\Big|_{x=L} + EI\frac{\partial^3 y}{\partial x^3}\Big|_{x=L}. \quad (12)$$

Since $F_\mathrm{o}(t)$ is generally much smaller than $F_\mathrm{c}(t)$ and $F_\mathrm{b}(t)$, the total force exerted by the string on the bridge can be calculated as

$$F_\mathrm{d}(t) = -F_\mathrm{c}(t) - F_\mathrm{b}(t). \quad (13)$$

Approximations to the emitted sound can be found by filtering $F_\mathrm{d}(t)$ and $F_\mathrm{n}(t)$, where the filters have transfer functions that approximate measured body radiation responses (see Section 3.3).

The plucking force is specified here in highly simplified form:

$$F_\mathrm{e}(t) = \begin{cases} a_\mathrm{e}\sin^2\left[(\pi/\tau_\mathrm{e})\, h_\mathrm{e}(t - t_\mathrm{e})\right] & : \quad t \in \mathcal{T}_\mathrm{e} \\ 0 & : \quad \text{otherwise} \end{cases}, \quad (14)$$

where $\mathcal{T}_\mathrm{e} = [t_\mathrm{e}, t_\mathrm{e} + \tau_\mathrm{e}]$ and with

$$h_\mathrm{e}(t) = \frac{1}{2}\left[t + \frac{\tau_\mathrm{e}\sinh(\beta_\mathrm{e} t/\tau_\mathrm{e})}{\sinh(\beta_\mathrm{e})}\right]. \quad (15)$$

As seen in Fig. 3, the parameter $\beta_\mathrm{e} > 0$ controls the attack and release slopes of the plucking function, which allows mimicking the gentle style in which tanpura strings are generally plucked. The other control parameters are the amplitude a_e and the overall pluck signal timespan τ_e. More sophisticated plucking models have been proposed (see, e.g. [9]) but may not be needed given that the characteristics of the tanpura sound are heavily dominated by the nonlinearity of the string-bridge interaction, with relatively little dependence on the intricacies of the finger-string interaction.

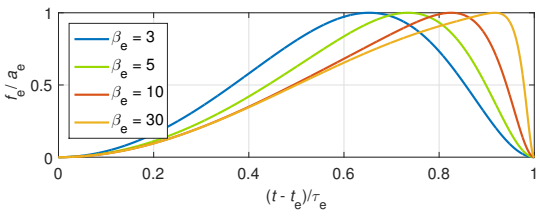


Figure 3: Normalised plucking force signal for four values of β_e.

2.2. Modal Expansion

The solution of (1) can be expressed as a superposition of the normal modes of the string (indexed with i):

$$y(x,t) = \sum_{i=1}^{M} v_i(x)\, \bar{y}_i(t), \tag{16}$$

where $\bar{y}_i(t)$ denotes the mode displacement and $v_i(x) = \sin(\beta_i x)$ is the corresponding mode shape (spatial eigenfunction) for the boundary conditions given in (2), with $\beta_i = i\pi/L$. After substitution of (16) into (1), then multiplying with $v_i(x)$ and applying a spatial integral over the length of the string, one obtains that the dynamics of each of the modes is governed by:

$$m\frac{\partial^2 \bar{y}_i}{\partial t^2} = -k_i \bar{y}_i(t) - r_i \frac{\partial \bar{y}_i}{\partial t} + \sum_{z=\mathrm{c,b,e}} \bar{F}_{\mathrm{z},i}(t), \tag{17}$$

in which $m = \frac{1}{2}\rho A L$ is the modal mass (which is the same for all modes), and where $k_i = \frac{1}{2}L\left(EI\beta_i^4 + T\beta_i^2\right)$ and $r_i = \frac{1}{2}L\gamma(\beta_i)$ are the elastic and damping constants of the mode, respectively. Within the constraint $r_i < 2\sqrt{k_i m}$, the modal frequencies are $\omega_i = \sqrt{k_i/m - \alpha_i^2}$, where (in accordance with (3))

$$\alpha_i = r_i/(2m) = \sigma_0 + \sigma_1\beta_i + \sigma_3\beta_i^3 \tag{18}$$

are the modal decay rates. The force terms in (17) are

$$\bar{F}_{z,i}(t) = \int_0^L v_i(x)\psi_z(x)F_z(t)dx = g_{z,i}F_z(t), \tag{19}$$

where[1]:

$$g_{z,i} = \frac{\pi^2 \sin(\beta_i x_z)\cos(\beta_i w_z/2)}{\pi^2 - \beta_i^2 w_z^2}. \tag{20}$$

In modal form, the spatially averaged string displacement at $x = x_z$ ($z = \mathrm{c, b}$) is

$$y_z(t) = \int_{x_z - w_z/2}^{x_z + w_z/2} \psi_z(x) \sum_{i=1}^{M} v_i(x)\bar{y}_i(t)dx = \sum_{i=1}^{M} g_{z,i}\bar{y}_i(t), \tag{21}$$

and the nut force can be expanded as

$$F_{\mathrm{n}}(t) = \sum_{i=1}^{M}\Big[-Tv_i'(L) + EIv_i'''(L)\Big]\bar{y}_i(t), \tag{22}$$

where $v_i'(x)$ and $v_i'''(x)$ denote the first and third spatial derivative of $v_i(x)$, respectively.

2.3. Discretisation in Time

We may re-formulate (17) in first-order form as follows:

$$\frac{\partial \bar{y}_i}{\partial t} = \frac{\bar{p}_i}{m}, \tag{23}$$

$$\frac{\partial \bar{p}_i}{\partial t} = -k_i\bar{y}_i - r_i\frac{\partial \bar{y}_i}{\partial t} + \sum_{z=\mathrm{c,b,e}} g_{z,i}F_z(t), \tag{24}$$

in which $\bar{p}_i$ represents the modal momentum. Gridding time by denoting $y^n \equiv y(n\Delta t)$, where $\Delta t = {f_\mathrm{s}}^{-1}$ is the temporal step, we introduce the difference and sum operators

$$\delta y^n = y^{n+\frac{1}{2}} - y^{n-\frac{1}{2}} \approx \Delta t\, \frac{\partial y}{\partial t}\Big|_{t=n\Delta t}, \tag{25}$$

$$\mu y^n = y^{n+\frac{1}{2}} + y^{n-\frac{1}{2}} \approx 2\, y\Big|_{t=n\Delta t}. \tag{26}$$

[1] Note that care has to be taken in evaluating (20) for $\beta_i = w_z/\pi$, using $\lim_{\beta_i \to w_z/\pi} g_{z,i} = \frac{\pi}{4}\sin(\beta_i x_z)$.

The mid-point-in-time discretisation of equations (23) and (24) can then be written as

$$\frac{\delta \bar{y}_i^{n+\frac{1}{2}}}{\Delta t} = \frac{\mu \bar{p}_i^{n+\frac{1}{2}}}{2m}, \tag{27}$$

$$\frac{\delta \bar{p}_i^{n+\frac{1}{2}}}{\Delta t} = -k_i\frac{\mu \bar{y}_i^{n+\frac{1}{2}}}{2} - r_i\frac{\delta \bar{y}_i^{n+\frac{1}{2}}}{\Delta t} + \sum_{z=\mathrm{c,b,e}} g_{z,i}F_z^{n+\frac{1}{2}}, \tag{28}$$

which is equivalent to applying the trapezoidal rule. The external force is calculated as $\frac{1}{2}\mu F_\mathrm{e}^{n+\frac{1}{2}}$, and the cotton thread force is obtained by discretising (7):

$$F_\mathrm{c}^{n+\frac{1}{2}} = k_\mathrm{c}\left(h_\mathrm{c} - \tfrac{1}{2}\mu y_\mathrm{c}^{n+\frac{1}{2}}\right) - r_\mathrm{c}\frac{\delta y_\mathrm{c}^{n+\frac{1}{2}}}{\Delta t}. \tag{29}$$

Special care has to be taken regarding stability in discretising the bridge contact force, due to its non-analytic form [15, 16]. A suitable numerical term is obtained by discretising (10), which yields the two-point discrete gradient:

$$F_\mathrm{b}^{n+\frac{1}{2}} = -\frac{\delta V_\mathrm{b}^{n+\frac{1}{2}}}{\delta y_\mathrm{b}^{n+\frac{1}{2}}} = -\frac{V_\mathrm{b}(y_\mathrm{b}^{n+1}) - V_\mathrm{b}(y_\mathrm{b}^n)}{y_\mathrm{b}^{n+1} - y_\mathrm{b}^n}. \tag{30}$$

Using the scaled momentum value $\bar{q}_i^n = (\Delta t/(2m))\bar{p}_i^n$, the system equations (27,28) can be written more conveniently as

$$\delta \bar{y}_i^{n+\frac{1}{2}} = \mu \bar{q}_i^{n+\frac{1}{2}}, \tag{31}$$

$$\delta \bar{q}_i^{n+\frac{1}{2}} = -a_i\mu \bar{y}_i^{n+\frac{1}{2}} - b_i\delta \bar{y}_i^{n+\frac{1}{2}} + \xi \sum_{z=\mathrm{c,b,e}} g_{z,i}F_z^{n+\frac{1}{2}}, \tag{32}$$

where $\xi = \Delta t^2/(2m)$, $a_i = \frac{1}{4}k_i\Delta t^2\, m^{-1}$, and $b_i = \frac{1}{2}r_i\Delta t m^{-1}$. Once the force terms $F_z^{n+\frac{1}{2}}$ are known, the dynamics of each mode can be simulated by solving for $\bar{y}_i^{n+1}$ and $\bar{q}_i^{n+1}$ at each time step. However, unless a very small temporal step is used, this procedure would lead to severe numerical dispersion. Therefore the coefficients in (31,32) are replaced by the values (a_i^*, b_i^*) below, which ensures that the modes have exact modal frequencies and damping:

$$a_i^* = \frac{1 - 2R_i\Omega_i + R_i^2}{1 + 2R_i\Omega_i + R_i^2}, \quad b_i^* = \frac{2\left(1 - R_i^2\right)}{1 + 2R_i\Omega_i + R_i^2}, \tag{33}$$

where $R_i = \exp(-\alpha_i\Delta t)$ and $\Omega_i = \cos(\omega_i\Delta t)$. This is readily verified by testing the (31,32) without the force terms for the ansatz

$$\bar{y}_i^n = \exp(s_\mathrm{d} n\Delta t), \quad \bar{q}_i^n = C\exp(s_\mathrm{d} n\Delta t), \tag{34}$$

where $s_\mathrm{d} = \mathrm{j}\omega_\mathrm{d} - \alpha_\mathrm{d}$ is the complex resonance frequency of the discretised mode, with $\mathrm{j} = \sqrt{-1}$, and C is a complex constant. This leads to a characteristic equation in $z_\mathrm{d} = \exp(s_\mathrm{d}\Delta t)$:

$$z_\mathrm{d}^2 - 2\left(\frac{1 - a_i}{1 + a_i + b_i}\right)z_\mathrm{d} + \left(\frac{1 + a_i - b_i}{1 + a_i + b_i}\right) = 0. \tag{35}$$

After substituting (33) this becomes

$$z_\mathrm{d}^2 - 2R_i\cos(\omega_i\Delta t)z_\mathrm{d} + R_i^2 = 0, \tag{36}$$

which has the solution $z_\mathrm{d} = R_i\exp(\omega_i\Delta)$, from which it follows that $s_\mathrm{d} = \mathrm{j}\omega_i - \alpha_i$, i.e. the discrete system has exact modal frequency and damping. The above coefficient replacement can be thought of in terms of the following adapted elasticity and damping values:

$$k_i \to k_i^* = \frac{4ma_i^*}{\Delta t^2}, \quad r_i \to r_i^* = \frac{2mb_i^*}{\Delta t}. \tag{37}$$

An accompanying restriction - necessary to avoid aliased modes - is that none of mode frequencies exceeds the Nyquist frequency, i.e. $\omega_i < \pi/\Delta t$, which defines the largest possible truncation of the modal series expansion in (16). A seemingly simpler way to avoid numerical dispersion is to discretize (17) using the impulse invariant transform and defining (30) in three-point form. However the resulting model would not be provably stable due to mode dependency of the modal force coefficients, which precludes defining an energy balance with non-negative energy.

2.4. A Vector-Matrix Update Form

If we stack the variables in column vector form, for M modes we may write the system equations as

$$\delta\bar{\mathbf{y}}^{n+\frac{1}{2}} = \mu\bar{\mathbf{q}}^{n+\frac{1}{2}}, \tag{38}$$

$$\delta\bar{\mathbf{q}}^{n+\frac{1}{2}} = -\mathbf{A}\mu\bar{\mathbf{y}}^{n+\frac{1}{2}} - \mathbf{B}\delta\bar{\mathbf{y}}^{n+\frac{1}{2}} + \xi \sum_{z=\mathrm{c,b,e}} \mathbf{g}_z F_z^{n+\frac{1}{2}}, \tag{39}$$

where $\mathbf{A}$ and $\mathbf{B}$ are diagonal matrices with diagonal entries $A_{ii} = a_i^*$ and $B_{ii} = b_i^*$, and where column vectors $\mathbf{g}_z$ hold the values defined with (20). A convenient vector update is then found by using (38) to define $\bar{\mathbf{s}} = \delta\bar{\mathbf{y}}^{n+\frac{1}{2}} = \mu\bar{\mathbf{q}}^{n+\frac{1}{2}}$, and substituting

$$\bar{\mathbf{y}}^{n+1} = \bar{\mathbf{s}} + \bar{\mathbf{y}}^n, \quad \bar{\mathbf{q}}^{n+1} = \bar{\mathbf{s}} - \bar{\mathbf{q}}^n, \tag{40}$$

in (39), which allows solving for $\bar{\mathbf{s}}$ with

$$\bar{\mathbf{s}} = \bar{\mathbf{u}} + \mathbf{e}_\mathrm{b} F_\mathrm{b}^{n+\frac{1}{2}} + \mathbf{e}_\mathrm{c} F_\mathrm{c}^{n+\frac{1}{2}}, \tag{41}$$

where $\mathbf{e}_\mathrm{b} = \xi\mathbf{J}^{-1}\mathbf{g}_\mathrm{b}$ and $\mathbf{e}_\mathrm{c} = \xi\mathbf{J}^{-1}\mathbf{g}_\mathrm{c}$, with $\mathbf{J} = \mathbf{I} + \mathbf{A} + \mathbf{B}$, and where

$$\bar{\mathbf{u}} = \mathbf{J}^{-1}\Big[2\left(\bar{\mathbf{q}}^n - \mathbf{A}\bar{\mathbf{y}}^n\right) + \xi\mathbf{g}_\mathrm{e} F_\mathrm{e}^{n+\frac{1}{2}}\Big]. \tag{42}$$

Hence once the contact forces $F_\mathrm{b}^{n+\frac{1}{2}}$ and $F_\mathrm{c}^{n+\frac{1}{2}}$ are known, eq. (41) immediately yields the step $\bar{\mathbf{s}}$, after which both $\bar{\mathbf{y}}^{n+1}$ and $\bar{\mathbf{q}}^{n+1}$ can be updated using (40). A further update comprises evaluating the nut force $F_\mathrm{n}^{n+\frac{1}{2}}$ with the vector form of (22). Note that since the matrices $\mathbf{J}$ and $\mathbf{A}$ are diagonal, the matrix operations in (42) can be implemented very efficiently via componentwise multiplication/division.

2.5. Solving for the Contact Forces

In order to solve for the forces $F_\mathrm{b}^{n+\frac{1}{2}}$ and $F_\mathrm{c}^{n+\frac{1}{2}}$, (41) is pre-multiplied with $\mathbf{g}_\mathrm{b}^\mathsf{T}$, which (also using $s_\mathrm{b} = y_\mathrm{b}^{n+1} - y_\mathrm{b}^n$) yields the scalar equation

$$s_\mathrm{b} = u_\mathrm{b} + \theta_\mathrm{bb} F_\mathrm{b}^{n+\frac{1}{2}} + \theta_\mathrm{bc} F_\mathrm{c}^{n+\frac{1}{2}}, \tag{43}$$

where $u_\mathrm{b} = \mathbf{g}_\mathrm{b}^\mathsf{T}\bar{\mathbf{u}}$, $\theta_\mathrm{bb} = \mathbf{g}_\mathrm{b}^\mathsf{T}\mathbf{e}_\mathrm{b}$ and $\theta_\mathrm{bc} = \mathbf{g}_\mathrm{b}^\mathsf{T}\mathbf{e}_\mathrm{c}$. Similarly, one can pre-multiply (41) with $\mathbf{g}_\mathrm{c}^\mathsf{T}$, giving

$$s_\mathrm{c} = u_\mathrm{c} + \theta_\mathrm{bc} F_\mathrm{b}^{n+\frac{1}{2}} + \theta_\mathrm{cc} F_\mathrm{c}^{n+\frac{1}{2}}, \tag{44}$$

with $u_\mathrm{c} = \mathbf{g}_\mathrm{c}^\mathsf{T}\bar{\mathbf{u}}$ and $\theta_\mathrm{cc} = \mathbf{g}_\mathrm{c}^\mathsf{T}\mathbf{e}_\mathrm{c}$. Setting $\phi_\mathrm{c} = k_\mathrm{c}/2 + r_\mathrm{c}/\Delta t$, the numerical thread force in equation (29) can be written in terms of s_c as

$$F_\mathrm{c}^{n+\frac{1}{2}} = k_\mathrm{c}\left(h_\mathrm{c} - y_\mathrm{c}^n\right) - \phi_\mathrm{c} s_\mathrm{c}. \tag{45}$$

Combining with (44) to eliminate s_c then gives

$$F_\mathrm{c}^{n+\frac{1}{2}} = v_\mathrm{c} - \beta_\mathrm{c} F_\mathrm{b}^{n+\frac{1}{2}}, \tag{46}$$

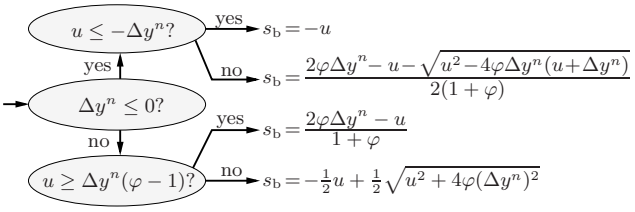


Figure 4: Analytic solution of equation (48), where $\varphi = \theta k_\mathrm{b}/2$ and $\Delta y^n = h_\mathrm{b} - y_\mathrm{b}^n$.

where

$$v_\mathrm{c} = \frac{k_\mathrm{c}\left(h_\mathrm{c} - y_\mathrm{c}^n\right) - \phi_\mathrm{c} u_\mathrm{c}}{1 + \phi_\mathrm{c}\theta_\mathrm{cc}}, \quad \beta_\mathrm{c} = \frac{\phi_\mathrm{c}\theta_\mathrm{bc}}{1 + \phi_\mathrm{c}\theta_\mathrm{cc}}. \tag{47}$$

Now substituting this into (43) and evaluating the bridge force term with (30) one obtains the nonlinear scalar equation

$$s_\mathrm{b} + u + \theta\frac{V_\mathrm{b}(s_\mathrm{b} + y_\mathrm{b}^n) - V_\mathrm{b}(y_\mathrm{b}^n)}{s_\mathrm{b}} = 0, \tag{48}$$

where $\theta = \theta_\mathrm{bb} - \beta_\mathrm{c}\theta_\mathrm{bc}$ and $u = -u_\mathrm{b} - \theta_\mathrm{bc} v_\mathrm{c}$. This equation is analytically solvable according to four distinct cases [18] as shown in Fig. 4. Once s_b is known the bridge force can be calculated accordingly and the thread force is updated with (46).

2.6. Numerical Energy Balance and Stability

The energy of the numerical model can be calculated for any time instant n by summing up the mode energies and adding the potentials of the bridge and thread interaction:

$$\begin{aligned} H^n &= \sum_{i=1}^{M}\left[\frac{1}{2m}(\bar{p}_i^n)^2 + \frac{k_i^*}{2}(\bar{y}_i^n)^2\right] + V_\mathrm{c}(y_\mathrm{c}^n) + V_\mathrm{b}(y_\mathrm{b}^n) \\ &= \frac{(\bar{\mathbf{q}}^n)^\mathsf{T}\bar{\mathbf{q}}^n + (\bar{\mathbf{y}}^n)^\mathsf{T}\mathbf{A}\bar{\mathbf{y}}^n}{\xi} + \frac{k_\mathrm{c}}{2}\left[h_\mathrm{c} - y_\mathrm{c}^n\right]^2 + \frac{k_\mathrm{b}}{2}\lfloor h_\mathrm{b} - y_\mathrm{b}^n\rfloor^2. \end{aligned} \tag{49}$$

Multiplying the left-hand side of (39) with $(\mu\bar{\mathbf{q}}^{n+\frac{1}{2}})^\mathsf{T}$ and the right-hand side with $(\delta\bar{\mathbf{y}}^{n+\frac{1}{2}})^\mathsf{T}$ yields, after a few further algebraic manipulations, the energy balance

$$\frac{\delta H^{n+\frac{1}{2}}}{\Delta t} = P^{n+\frac{1}{2}} - Q^{n+\frac{1}{2}}, \tag{50}$$

where

$$P^{n+\frac{1}{2}} = \tfrac{1}{2}\Delta t^{-1}\mathbf{g}_\mathrm{e}^\mathsf{T}\delta\bar{\mathbf{y}}^{n+\frac{1}{2}}\mu F_\mathrm{e}^{n+\frac{1}{2}} \tag{51}$$

is the input power and

$$Q^{n+\frac{1}{2}} = \frac{(\delta\bar{\mathbf{y}}^{n+\frac{1}{2}})^\mathsf{T}\mathbf{B}\delta\bar{\mathbf{y}}^{n+\frac{1}{2}}}{\xi\Delta t} + \left(\frac{r_\mathrm{c}}{\Delta t^2}\right)\left(\delta y_\mathrm{c}^{n+\frac{1}{2}}\right)^2 \tag{52}$$

is the dissipated power. From inspecting (33), it follows that the diagonal matrices $\mathbf{A}$ and $\mathbf{B}$ can only contain real-valued positive elements and are thus positive definite. This implies unconditional numerical stability, as both H^n and $Q^{n+\frac{1}{2}}$ are consequently guaranteed to be non-negative (i.e. the total energy can increase only through external force excitation). It is worthwhile pointing out that the energy balance in (50) relies not only on the use of the discrete gradient in (30), but also on the fact that equations (4) and (8) utilise the same spatial distribution function, which ensures that the vector terms $\mathbf{g}_\mathrm{b}$ are eliminated in the calculation process.

3. PHYSICAL CHARACTERISATION

3.1. String Parameters

The string parameters L, A, E and ρ are readily available for a given string, and the tension T can be set accordingly such that the correct fundamental frequency results. The moment of inertia is calculated directly from the string radius r as $I = \frac{1}{4}\pi r^4$. The damping coeffcients $\sigma_{0,1,3}$ in (18) can be estimated from a plucked string signal measured with a freely vibrating string (i.e. no bridge contact), for example as in [21]. Given the importance of the role of string damping in the jvari effect, it is worthwhile noting that these coefficients can be expressed directly in terms of the physically motivated fit parameters used by Woodhouse in eq. (8) of [21] for characterising guitar strings:

$$\sigma_0 \approx \tfrac{1}{2}\eta_A, \quad \sigma_1 \approx \tfrac{1}{2}c\,\eta_F, \quad \sigma_3 \approx \tfrac{1}{2}c\,\lambda\,\eta_B, \tag{53}$$

with $c = \sqrt{T/(\rho A)}$ and $\lambda = EI/T$, and where η_A, η_F, and η_B represent "air", "friction", and "bending" damping, respectively. Woodhouse's modal decay rate form, which was adapted from [22], generally allows an excellent low-parameter fit to measured string data over a wide frequency range [22, 21]. For this reason equation (18) is preferred here over the even-order form $\alpha_i = \sum_j \sigma_j \beta_i^{2j}$ often used in sound synthesis.

Table 1: Physical parameter values used for a C_3 string.

string parameters			contact parameters		
L	1.0	[m]	x_b	10×10^{-3}	[m]
ρA	4.83×10^{-4}	[kg/m]	w_b	2.0×10^{-3}	[m]
T	33.1	[N]	k_b	4.39×10^{8}	[N/m]
EI	6.03×10^{-5}	[Nm^2]	x_c	[4.0 - 8.0]$\times10^{-3}$	[m]
σ_0	0.6	[s^{-1}]	w_c	1.2×10^{-3}	[m]
σ_1	6.5×10^{-3}	[m/s]	k_c	1.2×10^{5}	[N/m]
σ_3	5.0×10^{-6}	[m^3/s]	r_c	1.2	[kg/s]

3.2. Contact Parameters

The rationale for using a contact law with unity exponent is based on an analogy to contact between two cylinders with parallel axes. From Hertzian contact theory, the force F is proportional to the depth of indentation d for this case [23]:

$$F = \frac{\pi}{4} E^* l\, d, \tag{54}$$

where l is the contact length, with the contact modulus given as

$$E^* = \left(\frac{1-\nu_1^2}{2E_1} + \frac{1-\nu_2^2}{2E_2} \right)^{-1} \tag{55}$$

for materials with Young's moduli E_1, E_2 and Poisson's ratios ν_1, ν_2. This law is independent of the radii of the two cylinders and thus is applicable also to contact between a cylindrical string and a nearly flat surface. By loosely equating d and $\Delta y = h_\mathrm{b} - y_\mathrm{b}$ an estimate of the elastic constant in (6) is found as $k_\mathrm{b} = \frac{1}{4}\pi E^* w_\mathrm{b}$, with E^* evaluated using the values for steel (or bronze when applicable) and ivory in eq. (55). The cotton thread contact parameters are not as easily determined and are empirically set here as $k_\mathrm{c} = w_\mathrm{c} \cdot 10^8$ N/m and $r_\mathrm{c} = k_\mathrm{c} \cdot 10^{-6}$ N/m. The parameters used for a one metre long male tanpura string (steel, radius = 0.14mm) when tuned to the note C_3 ($f_1 = 130.81$Hz) are listed in Table 1.

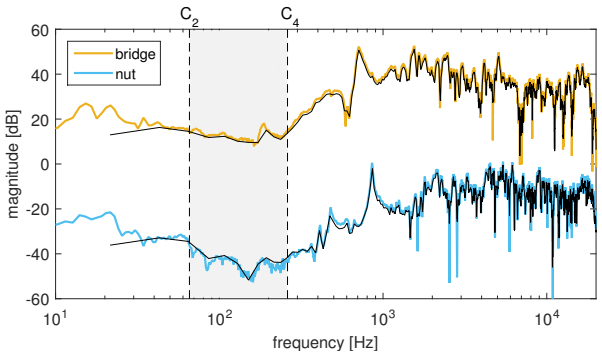


Figure 5: Tanpura body magnitude response measured with an impact hammer applied at the bridge (top) and the nut (bottom), and with a microphone positioned at 80cm from the instrument. For clarity, the responses have been offset by 50dB. The grey shaded area indicates the range in which the fundamental frequency of a tanpura string would normally fall.

3.3. Body Radiation Filters

Fig. 5 shows the body magnitude responses of a tanpura as obtained with impact hammer experiments. The fundamental frequency of a tanpura string will generally fall in the range between that of a C_2 string ($f_1 = 65.4$Hz) and a C_4 string ($f_1 = 198$Hz). As the plot shows, the body radiates less powerfully in this frequency range, to a first approximation acting as a high-pass filter. The black thin solid lines in Fig. 5 indicate the responses of 2048-tap FIR filters obtained by truncating the measured body impulse responses. Significantly more efficient body filters models can be achieved in IIR form (see, e.g. [24]).

4. SIMULATION RESULTS

4.1. Simulation of a C_3 String

Fig. 6 shows a small selection of string profile snapshots obtained with the simulation of a C_3 string, using the parameters listed in Table 1, with $x_\mathrm{c} = 5$mm. To exemplify the role of the bridge and the thread, the string was excited using $a_\mathrm{e} = -0.4$N, $\tau_\mathrm{e} = 50$ms, $\beta_\mathrm{e} = 30$, $x_\mathrm{e} = L/2$ and $w_\mathrm{e} = L$, effectively initialising the string approximately to the first mode shape. As seen in Fig. 6(a) the string motion becomes progressively Helmholz-like, as also found in earlier studies [22, 19]. The zoomed view in Fig. 6(b) shows that a small level of string motion is allowed at the thread position. The bridge on the other hand is far less compressive, approximating a one-sided constraint. The role of the cotton thread connection can be summarised as follows: the thread damping provides additional attenuation of the high-frequency standing waves along the string length between the left termination and the bridge, which in combination with the lower elasticity constant effects a 'soft' termination (somewhat similar to a violin string stopped by a finger rather than the nut). This avoids the harsher sound that would result with a (near) rigid termination.

Fig. 7 shows how the displacement at the bridge evolves over time. The string-bridge compression is extremely small (less than 3×10^{-8}m over the whole duration). The plots demonstrate the emergence of a high-frequency wavepacket in the waveforms. Due to string stiffness this precursive wave arrives back at the bridge before the lower frequencies, thus escaping the periodic closing of

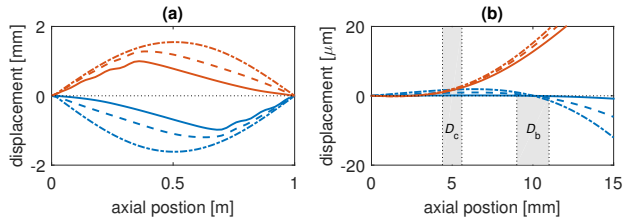


Figure 6: Snapshots of the profile of a C_3 string excited using the parameters $x_e = L/2$, $w_e = L$. The sampling frequency is 44.1kHz. Left: full string profile. Right: zoomed view of the thread/bridge region; the grey shaded rectangles indicate the force-active regions of the thread (D_c) and the bridge (D_b). For both plots, the time instants are t = 49.0ms (dash-dot blue), t = 52.6ms (dash-dot red), t = 208.4ms (dashed blue), t = 212.0ms (dashed red), t = 412.7ms (solid blue), t = 416.3ms (solid red).

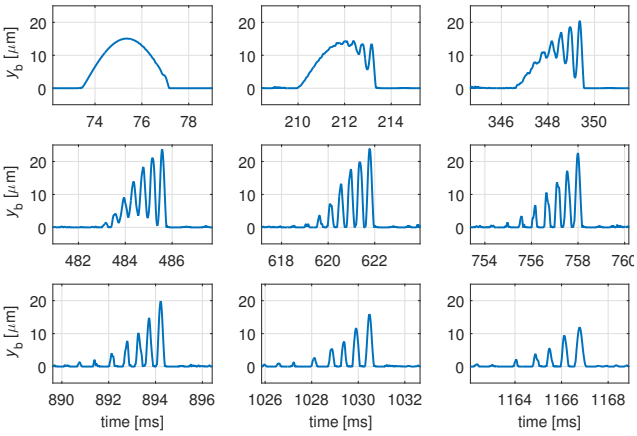


Figure 7: Evolution of the displacement (y_b) at the bridge.

the gap between the string and the bridge [3]. Running the simulation with $EI = 0$ confirms that the precursor, which is the principal oscillatory manifestation of the jvari effect, disappears in the absence of string stiffness. As can be gleaned from the plots, the spectral centroid of the precursor gradually diminishes over time, which is due to the string damping being frequency-dependent. The rate at which the centroid descends also depends on the distance between the bridge and the thread [3]; this is one of the sonic features that tanpura players control when, in preparation of a performance, they adjust the thread position in search of a desired jvari.

Fig. 8(a) shows the global evolution of the driving forces F_d and F_n. Comparison with the corresponding pressure signals p_d and p_n shown in Fig. 8(b) highlights the difference in terms of the initial attack transient, which is almost completely absent from the radiation signals; this is because they contain much less of the low-frequency excitation components and are dominated by the frequencies that make up the precursor. The aural impression is therefore that the sound grows in amplitude over the initial 500ms, which is contrary to our normal experience of plucked strings. These results suggest that the high-pass nature of the body responses (explained earlier in Section 3.3) is a key ingredient in producing this effect, which is further enhanced by the frequency-dependent sensitivity of human hearing [3].

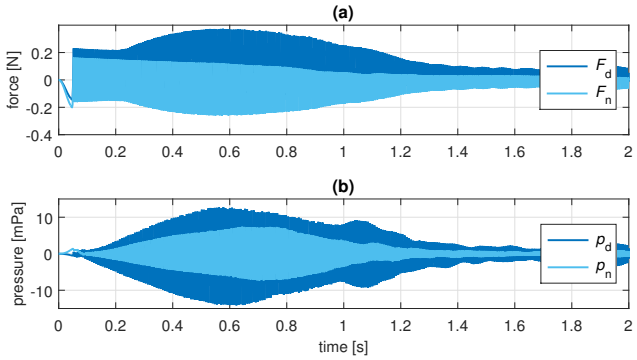


Figure 8: Waveforms of the bridge and nut driving force signals (a) and the corresponding radiation pressure signals (b).

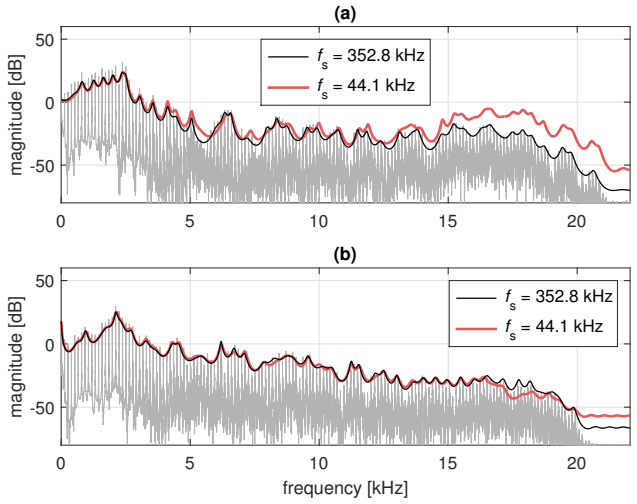


Figure 9: Spectral envelopes of the bridge radiation pressure signal (a) and the nut radiation signal (b), for different sampling frequencies. In each plot, the magnitude spectrum obtained with eight times oversampling (f_s =352.8kHz) is plotted in light grey.

4.2. Convergence

In a sound synthesis context, it is natural to use a standard audio rate. However mainly due to the nonlinear phenomena, the model will not give exactly the same result as for higher sampling frequencies, even when the number of modes is kept the same. Fig. 9 shows the magnitude spectra (light grey line) of the radiation signals p_d and p_n, as obtained with for a C_3 string using eight times oversampling. In each of the plots the thin black line represents the corresponding spectral envelope while the thicker, red line indicates the envelope of the spectrum obtained using $f_s = 44.1$kHz. For both sampling frequencies, the number of modes was set to 133, which corresponds to truncating the mode series at 20kHz. The plots show that while there is little difference between the nut radiation signals, the 44.1kHz model bridge signal is artificially strong in amplitude at high frequencies ($f > 15$kHz). Perceptually, the discrepancy is very small but nevertheless noticeable. Informal listening experiments indicated that oversampling by a factor two (i.e. using $f_s = 88.2$kHz) is sufficient to reduce any differences to almost unnoticeable levels.

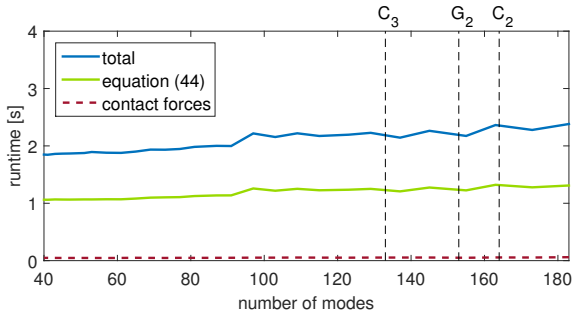


Figure 10: `Matlab` computation times for one second of simulated sound when using a 44.1kHz sampling frequency. The results were obtained using a machine with an i7 CPU 2.93GHz processor and 4GB RAM.

4.3. Algorithmic Efficiency

The `Matlab` runtime for one second of simulation (see Fig. 10) is only mildly dependent on the number of modes included and does not exceed 2.5 seconds for any practical string (compared to 2 minutes with the model in [19], which requires four times oversampling in order to alleviate numerical dispersion). A significant speed-up is achievable with an optimised C implementation, bringing the proposed model in range for real-time application on standard processors. Note that the filtering of the nut and bridge driving signals is not included in the computational operations that are time-measured. As is evident from the plot, more than half of the computational effort comprises updating equation (42), while solving for the contact forces takes up an almost negligible fraction.

4.4. Drone Synthesis

In order to get a first glimpse of what a virtual-acoustic tanpura would sound like, four separate string models were tuned to produce a *panchamam* raga pattern of G_2-C_3-C_3-C_2. The total bridge and nut driving forces are now

$$\hat{F}_\mathrm{d} = -\sum_{j=1}^{4} \left(F_{\mathrm{c},j} + F_{\mathrm{b},j}\right), \quad \hat{F}_\mathrm{n} = \sum_{j=1}^{4} F_{\mathrm{n},j}, \qquad (56)$$

where subscript j indexes the strings. These are filtered as before with the body radiation filters, with the filter outputs p_d and p_n assigned to the left and right channel of an audio signal; a sound example can be found on the accompanying webpage[2] alongside further supporting material. Fig. 11 shows the spectrogram of the total bridge force $\hat{F}_\mathrm{d}$ for a single drone cycle. The plot illustrates the migration of energy from the lower partials to higher frequency components. The precursors are manifested in the time-frequency representation as formants with descending centre frequencies. Also noticeable is that the precursors generated with the C_3 and C_2 strings together construct a joint, seamless jvari pattern, which adds to the sense of continuity of the drone [3].

5. CONCLUDING REMARKS AND PERSPECTIVES

The proposed model permits simulation of tanpura drones with significantly increased efficiency compared to previous models,

[2] `www.socasites.qub.ac.uk/mvanwalstijn/dafx16a/`

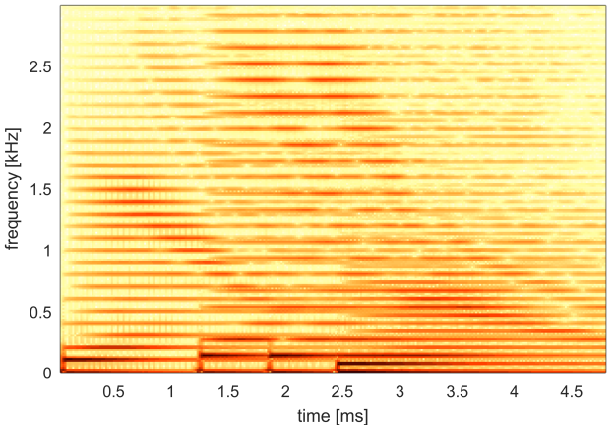


Figure 11: Spectrogram of a four-string tanpura bridge driving force signal. The strings were excited at $t_e = 0$, 1.2, 1.8, and 2.4s.

as such opening up possibilities for real-time implementation on standard processors. The modal approach taken results in spectral accuracy of the string resonance behaviour, and also facilitates efficient implementation due to the diagonality the system matrices.

Modal expansion is, of course, not a new concept in sound synthesis, originating some decades ago as *modal synthesis* [25]. It also underpins the more formalised *functional transformation method* [26, 13] and other modular approaches [27]. What is different in the approach presented here is that collision forces are incorporated in a provably stable manner, which is of particular importance in a real-time sound synthesis context.

The model could also be formulated in finite difference form, for example using a parametric implicit three-point scheme for the string [18] which can be tuned to closely approximate spectral accuracy. However for the relatively simple model proposed here this approach holds no efficiency advantages, since matrices would be sparse rather than diagonal. In addition, the three-point form of (30) is a less accurate approximation of the bridge contact force.

Among several possible model extensions, either in modal or finite difference form, probably the most important ones are (a) the simulation of sympathetic vibrations by also modelling the string-body coupling and (b) the re-introduction of distributed bridge collisions [28]. Because the latter effects a periodic modulation of the central contact point between the string and the bridge (see Fig. 2(b)), it would be sensible to consider this in conjunction with tension modulation, which is potentially relevant given that tanpura strings are loosely strung. The literature already contains various techniques for incorporating tension modulation into modal schemes [26, 29, 27] and finite difference formulations [30], but the proposed two-point scheme requires a re-formulation. Such extensions would pave the way for a rigorous evaluatory comparison between the proposed model, more complete models, and measurement data. However they will also increase the complexity and computational load of the model, meaning real-time synthesis may be viable only via hardware acceleration, such as graphical processing units [31] or field programmable arrays [32].

Besides model refinements and extensions, the most pertinent advance may be the implementation of a real-time virtual-acoustic tanpura that affords real-time control options. This would allow fine-tuning of the jvari effect by ear through on-line adjustment of the parameters of the string, thread, and bridge, much in the same way as real-world tanpura players set up their instrument.

6. REFERENCES

[1] C. Raman, "On some indian stringed instruments," in *Proc. of the Indian Assoc. for the Cultivation of Science*, 1921, vol. 7, pp. 29–33.

[2] R. Burridge, J. Kappraff, and C. Morshedi, "The sitar string, a vibrating string with a one-sided inelastic constraint," *SIAM J. Appl. Mathematics*, vol. 42, no. 6, pp. 1231–1251, 1982.

[3] C. Valette, C. Cuesta, C. Besnainou, and M. Castellengo, "The tanpura bridge as a precursive wave generator," *Acustica*, vol. 74, pp. 201–208, 1991.

[4] C. Vyasarayani, S. Birkett, and J. McPhee, "Modeling the dynamics of a vibrating string with a finite distributed unilateral constraint: Application to the sitar," *J. Acoust. Soc. Am.*, vol. 125, no. 6, pp. 3673–3682, 2009.

[5] T. Taguti, "Dynamics of simple string subject to unilateral constraint: A model analysis of sawari mechanism," *Acoust. Sci. Technol.*, vol. 29, no. 3, pp. 203–214, 2008.

[6] D. Kartofelev, A. Stulov, H. Lehtonen, and V. Välimäki, "Modeling a vibrating string terminated against a bridge with arbitrary geometry," in *Stockholm Musical Acoustics Conf.*, 2013.

[7] A. Krishnaswamy and J. Smith, "Methods for simulating string collisions with rigid spatial obstacles," in *Proc. IEEE WASPAA*, New York, 2003.

[8] S. Siddiq, "A Physical Model of the Nonlinear Sitar String," *Archives of Acoustics*, vol. 37, no. 1, Jan. 2012.

[9] G. Evangelista and F. Eckerholm, "Player-instrument interaction models for digital waveguide synthesis of guitar: Touch and collisions," *IEEE Trans. Audio, Speech and Lang. Proc.*, vol. 18, no. 4, pp. 822–832, 2010.

[10] S. Bilbao and A. Torin, "Numerical modeling and sound synthesis for articulated string/fretboard interactions," *J. Audio Eng. Soc.*, vol. 63, pp. 336–347, 2015.

[11] C. Desvages and S. Bilbao, "Two-polarisation finite difference model of bowed strings with nonlinear contact and friction forces," in *Proc. of the 17th Int. Conf. on Digital Audio Effects (DAFX-15)*, 2015.

[12] E. Rank and G. Kubin, "A waveguide model for slapbass synthesis," in *IEEE Int. Conf. on Acoust., Speech, and Sig. Proc.*, 1997, vol. 1, pp. 443–446.

[13] L. Trautmann and R. Rabenstein, "Multirate simulations of string vibrations including nonlinear fret-string interactions using the functional transformation method," *Appl. Signal Proc.*, vol. 7, pp. 949–963, 2004.

[14] P. Kramer, J. Abeßer, C. Dittmar, and G. Schuller, "A digital waveguide model of the electric bass guitar including different playing techniques," in *Proc. Int. Conf. Acoust. Speech Sig. Proc.*, 2012, pp. 353–356.

[15] V. Chatziioannou and M. van Walstijn, "An energy conserving finite difference scheme for simulation of collisions," in *Sound and Music Computing (SMAC-SMC 2013)*, 2013.

[16] S. Bilbao, A. Torin, and V. Chatziioannou, "Numerical modeling of collisions in musical instruments," *Acta Acustica united with Acustica*, vol. 101, no. 1, pp. 155–173, 2015.

[17] V. Chatziioannou and M. van Walstijn, "Energy conserving schemes for the simulation of musical instrument contact dynamics," *J. Sound and Vibration*, vol. 339, pp. 262–279, 2015.

[18] S. Bilbao, "Numerical Modeling of String/Barrier Collisions," in *Proc. of the Int. Symp. Musical Acoustics (ISMA)*, 2014.

[19] M. van Walstijn and V. Chatziioannou, "Numerical simulation of tanpura string vibrations," in *Int. Symp. on Musical Acoustics (ISMA)*, 2014.

[20] J. Bridges and M. van Walstijn, "Investigation of tanpura string vibrations using a two-dimensional time-domain model incorporating coupling and bridge friction," in *Third Vienna Talk on Music Acoustics, At Vienna, Austria*, 2015, pp. 126–131.

[21] J. Woodhouse, "Plucked guitar transients: Comparison of measurements and synthesis," *Acta Acustica united with Acustica*, vol. 90, no. 5, pp. 945–965, 2004.

[22] C. Valette and C. Cuesta, *Mécanique de la corde vibrante*, Hermes, Paris, 1993.

[23] V. L. Popov, *Contact Mechanics and Friction: Physical Principles and Applications*, Springer, Heidelberg, New York, 2010.

[24] B. Bank, "Direct design of parallel second-order filters for instrument body modeling," in *Int. Comp. Music Conf. (ICMC)*, 2007, pp. 458–465.

[25] J.M. Adrien, "The missing link: Modal synthesis," in *Representations of Musical Signals*, Giovanni De Poli, Aldo Piccialli, and Curtis Roads, Eds., chapter 8, pp. 269–298. MIT Press, Cambridge, MA, USA, 1991.

[26] L. Trautmann and R. Rabenstein, *Digital Sound Synthesis by Physical Modelling Using the Functional Transformation Method*, Kluwer Academic / Plenum Publishers, 2003.

[27] F. Avanzini and R. Marogna, "A modular physically based approach to the sound synthesis of membrane percussion instruments," *IEEE Trans. Audio, Speech, and Lang. Proc.*, vol. 18, no. 4, pp. 891–902, 2010.

[28] M. van Walstijn and J. Bridges, "Simulation of distributed contact in string instruments: a modal expansion approach," in *Proc. Europ. Sig. Proc Conf (EUSIPCO2016)*, Accepted for publication, 2016.

[29] S. Bilbao, "Modal type synthesis techniques for nonlinear strings with an energy conservation property," in *Proc. Int. Conf. Digital Audio Effects (DAFx-04)*, 2004, pp. 119–124.

[30] S. Bilbao and J. O. Smith, "Energy-conserving finite difference schemes for nonlinear strings," *Acta Acustica u/w Acustica*, vol. 91, no. 2, pp. 299–311, 2005.

[31] S. Bilbao, A. Torin, P. Graham, J. Perry, and G. Delap, "Modular Physical Modeling Synthesis environments on GPU," in *Int. Comp. Music Conf. (ICMC)*, 2014, pp. 1396–1403.

[32] F. Pfeifle and R. Bader, "Real-time finite difference physical models of musical instruments on a field programmable gate array (FPGA)," in *Proc. Int. Conf. Digital Audio Effects (DAFx-12)*, 2012.

ASSESSING APPLAUSE DENSITY PERCEPTION USING SYNTHESIZED LAYERED APPLAUSE SIGNALS

Alexander Adami

International Audio Laboratories*,
Friedrich-Alexander Universität Erlangen
Erlangen, Germany
alexander.adami@audiolabs-erlangen.de

Sascha Disch

Fraunhofer IIS,
Erlangen, Germany
sascha.disch@iis.fraunhofer.de

Garri Steba

Ostbayerische Technische Hochschule
Amberg-Weiden, Germany
g.steba@oth-aw.de

Jürgen Herre

International Audio Laboratories*,
Friedrich-Alexander Universität Erlangen
Erlangen, Germany
juergen.herre@audiolabs-erlangen.de

ABSTRACT

Applause signals are the sound of many persons gathered in one place clapping their hands and are a prominent part of live music recordings. Usually, applause signals are recorded together or alongside with the live performance and serve to evoke the feeling of participation in a real event within the playback recipient. Applause signals can be very different in character, depending on the audience size, location, event type, and many other factors. To characterize different types of applause signals, the attribute of 'density' appears to be suitable. This paper reports first investigations whether density is an adequate perceptual attribute to describe different types of applause. We describe the design of a listening test assessing density and the synthesis of suitable, strictly controlled stimuli for the test. Finally, we provide results, both on strictly controlled and on naturally recorded stimuli, that confirm the suitability of the attribute density to describe important aspects of the perception of different applause signal characteristics.

1. INTRODUCTION

Recordings of applause signals can be very different in their sound character, depending on many factors including audience size, location, event type, recording setup, etc. Several publications in the past have shed some light on the nature of applause signals, providing a basic clap anaysis [1], attempting to synthesize applause through physical modeling of individual claps [2,3], through sound texture synthesis [4–7] or morphing of granular sounds [8]. Also the phenomenon of rhythmic applause [9, 10] was already subject of scientific curiosity. Yet, to our best knowledge, nobody has actually looked into how to perceptually characterize different types of applause signals. We propose to use the attribute 'density' to capture the predominant character of different applause signals.

To subjectively assess the suitability of the attribute density, we designed a dedicated listening test that is presented in this paper in the evaluation section.

To achieve a controlled variation of density within different applause stimuli without changing other factors like timbre, spatial properties, etc., we generated all test stimuli through layering from dry studio recordings of individual persons applauding. To mimic the geometric and physical conditions of a gathered audience, we applied a simple model of the same while layering. This is discussed in the next sections.

2. APPLAUSE DENSITY

Existing, well established perceptual attributes like loudness, pitch and timbre do not describe applause properties very well. To characterize different types of applause signals, the novel attribute of 'density' appears to be suitable. The concept of density might be rightfully attributed to all kinds of sounds and sound textures that predominantly consist of sufficiently dense transient events being distributed in a pseudo-random way like rain and fireworks.

In previous work, the idea of defining an 'impact rate' to further characterize environmental sound textures has already been roughly sketched, e.g., for describing the sound of falling raindrops [11]. However, the author concludes that such an impact rate appears to be more of a physical measure than a psychoacoustic one. Notwithstanding, the author suggests to further conduct psychoacoustic experiments to clarify this assumption. In our present paper, we are presenting the results of such experiments.

Kawahara, for example, measured a set of simple parameters of different applauses at a recording site in order to efficiently recreate similar applause sounds at a receiver site, controlled by the proposed set of transmitted parameters. These parameters include, among other things, the average time interval between two clapping events [12]. Again, the proposed parameters are related much closer to a physical measure than to a perceptual quality.

For sure, the perception of applause signals is influenced by several factors, whereas the most obvious one is certainly the number of people clapping simultaneously. Still, we suggest that the sensation of density is an abstract psychoacoustic quantity in its own right albeit closely coupled to the actual physics behind the generation of such sound textures. Additionally, parameters like spatial properties, room acoustics, and distance of a listener to the applauding crowd have an impact on the resulting applause impression: the amount of reverb determines how much individual claps get smeared and possibly blended, whereas the distance to the crowd influences the perceived near-by to far-off clap ratio, i.e., the ratio of individually distinguishable foreground claps and

* A joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer IIS, Germany

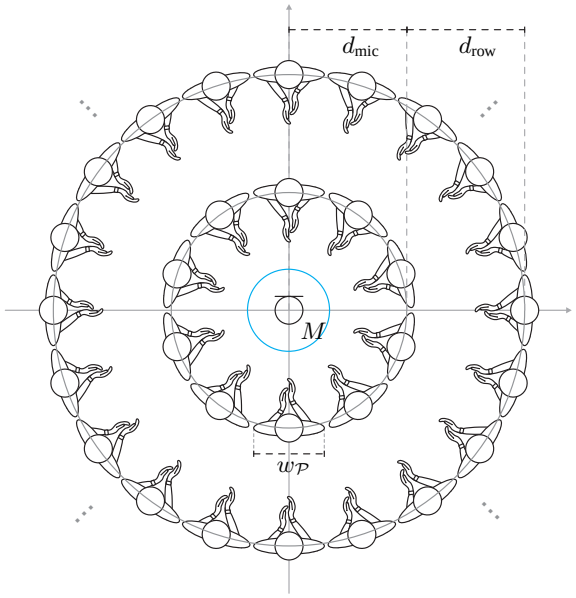


Figure 1: *Illustration of the basic applause model: virtual clapping people with width $w_{\mathcal{P}}$ at rows with the inter-row distance d_{row} and d_{mic} denoting the distance between the microphone M and the first row.*

the more homogenous background noise floor resulting from many claps being superimposed. Applause density appears to be a suitable choice for an attribute which accommodates all mentioned properties additional to the impact rate. Consequently, we investigate in this paper if there is a perceptual attribute like applause density and if it is consistently perceived.

3. SYNTHESIZING APPLAUSE SIGNALS

To assess applause density perception, it is necessary to have control over as many parameters of the applause signals as possible. For that reason, we came up with a simple model which allows to synthesize applause signals using layering of individual recordings of single people clapping. Note that the layering approach is not aiming at synthesizing the most realistic applause signals but providing a tool producing sufficiently natural and plausible applause to investigate density as a suitable perceptual attribute. In this section, the basic physically motivated applause model and some assumptions which are used to synthesize the applause signals are described. Furthermore, the algorithm will be introduced.

3.1. Basic Applause Model

Figure 1 depicts the assumed physical model for the applause synthesis. In the model, the virtual applauding people $\mathcal{P}_{r,p}$ are arranged on concentric circles, which will be called rows, around a virtual microphone M in the center, where r and p denote the one-based row and person indices. Every virtual person produces a corresponding clap signal $C_{r,p}$. The distance between the microphone and the first row of virtual people is given by d_{mic}, and the inter-row distance, i.e., the distance between consecutive rows is given by d_{row}. Varying d_{mic} will result in different near-by to far-off clap ratios, i.e., a small d_{mic} will result in many intensive and individually distinguishable near-by claps whereas a large d_{mic} will result in a more homogenous and noise-like signal.

The maximum number of virtual people allowed on a specific row is determined by the space a virtual person consumes and the radius of the circle the row is placed on. In the r-th row, the maximum number of virtual persons is given by

$$P_r = \text{floor}\left(\frac{2\pi(d_{\text{mic}} + (r-1)\cdot d_{\text{row}})}{w_{\mathcal{P}}}\right), \tag{1}$$

with $r = 1...R$ and R denoting the total number of rows. This also determines the angular spacing of the virtual persons on that row, i.e., a person in row r is placed every $\Delta_{\varphi,r} = \frac{360^\circ}{P_r}$ degree. Assuming all spots in a row are occupied, the overall number of persons P_Σ is then determined by

$$P_\Sigma = \sum_{r=1}^{R} P_r. \tag{2}$$

Since the sound pressure is attenuated on its way traveling from a virtual person at the r-th row to the microphone in the center, an attenuation factor has to be applied to it. Propagation losses are modeled by the distance law leading to the row-dependent amplitude attenuation factor

$$a_r = \frac{d_0}{d_{\text{mic}} + (r-1)\cdot d_{\text{row}}}, \tag{3}$$

where d_0 represents a reference distance which corresponds to the microphone distance used for the actual applause sample capturing. Note that a_r can also become greater than one, if the distance to a row is smaller than the reference distance.

There are also frequency dependent effects which need to be taken into account. For instance, people on the inner rows act as an obstacle to the inwards propagating sound waves and also the air itself absorbs sound energy. Both results in a frequency dependent attenuation. These shading and absorption effects can be modeled by applying a lowpass or treble shelving filter $h_r(t)$ to the signals. Since the filter characteristics are distance dependent, the filter has a dependency on the row index r.

In this model, we assume the microphone's pickup pattern M_{PU} to be spatially uniform which corresponds to an omnidirectional characteristic:

$$M_{\text{PU}}(\phi) = 1, \tag{4}$$

where $\phi = 0...360^\circ$ denotes the sound waves' incident angles. This is indicated by the blue circle around the microphone in Figure 1.

With the constant pickup pattern, the microphone signal can then be written as

$$M(t) = \sum_{r}^{R}\left(a_r \cdot \sum_{p}^{P_r}\left(M_{\text{PU}}\cdot C_{r,p}(t)\right) * h_r(t)\right), \tag{5}$$

where $*$ denotes a convolution.

3.2. The Layering Algorithm

Applause consists of up to thousands of individual clap signals naturally superimposed on their way traveling from the hands of the clapping person to the ears of the receiver or to a microphone. By

Parameter	Default	Description
$\widehat{P}_\Sigma$	-	desired number of people clapping
d_{mic}	15 m	distance of microphone to first row
d_{row}	1 m	distance between rows
$w_\mathcal{P}$	0.5 m	space a virtual person consumes
δ_{const}	1.15 s	constant relative signal shift
δ_{rand}	0.01 s	random relative signal shift

Table 1: *The applause model's controllable parameters*

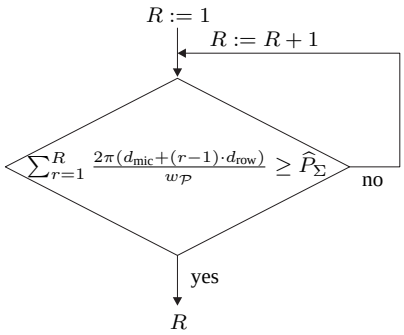


Figure 2: *Determination of the number of rows R.*

layering the captured base signals and applying a simple physical model, the natural behavior is mimicked.

The layering algorithm is designed such that it accepts the desired number of people clapping $\widehat{P}_\Sigma$ as input and produces the corresponding applause signal for the given parameters. A list of the model's controllable parameters is given in Table 1. Starting with the desired number of people clapping, the number of rows has to be determined. This can be done by increasing the number of rows until the expression in (6) comes true, i.e., the overall number of persons P_Σ possible for the given number of rows exceeds or equals the desired number of people clapping. This is also illustrated in Figure 2.

$$\sum_{r=1}^{R} \frac{2\pi(d_{\text{mic}} + (r-1)\cdot d_{\text{row}})}{w_\mathcal{P}} \geq \widehat{P}_\Sigma \tag{6}$$

In the next step, the number of people in each row has to be determined. Except for the last row, this can be done by using Equation (1). Since the last row does not necessarily need to be fully occupied, it has to be treated separately and can be computed using:

$$P_R = \widehat{P}_\Sigma - \sum_{r=1}^{R-1} P_r. \tag{7}$$

To compute the row-dependent attenuation factor a_r, Equation (3) can be applied directly. For the lowpass filter, a first order Butterworth filter with adaptive cut-off frequency was used to match the spectral tilt of an arbitrarily chosen reference applause recording.

Finally, the actual applause signals to be layered have to be generated from the captured base files $c_b(t)$, where $b = 0...B-1$ and B denotes the number of available base signals. Since there is only a limited number of base signals available, they have to be treated in a way such that they can be used multiple times without

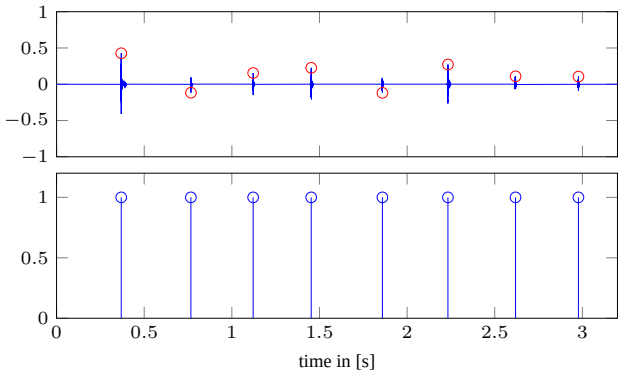


Figure 3: *Annotated base signal and corresponding metadata signal. Top plane: local peak picking; bottom plane: meta data signal*

creating perceivable artifacts due to correlations. This could be accomplished by using circularly time-shifted copies of the base signals. With δ_t denoting the time-shift, the time-shifted applause signals are obtained by

$$C_{\widehat{p}}(t) = \begin{cases} c_b(t + j\delta_t), & \text{if} \quad 0 \leq t < T - j\delta_t \\ c_b\big(t - (T - j\delta_t)\big), & \text{if} \quad T - j\delta_t < t \leq T \end{cases}, \tag{8}$$

where $\widehat{p} = 1...\widehat{P}_\Sigma$, $b = (\widehat{p}-1)\%B$, $j = \text{ceil}(\frac{\widehat{p}}{B} - 1)$, $\%$ denoting the modulo operator, and T denoting the uniform duration of the base signals. Please note, that the time-shifted clap signals were computed using the index $\widehat{p}$ for the sake of simplicity of notation. The relation between the $\widehat{p}$-based and the row- and persons-on-a-row-based indexing is given by $\widehat{p} = \sum_{\hat{r}=1}^{r-1} P_{\hat{r}} + p$. In order to add some variability to the layered applause signals, the time shift δ_t was designed to be a superposition of a constant and a random-based time offset, such that $\delta_t = \delta_{\text{const}} + \Delta_{\text{rand}}$, where Δ_{rand} denotes a function which returns a uniformly distributed random value from the interval $-\delta_{\text{rand}}...\delta_{\text{rand}}$. All random offsets were determined during the algorithm's initialization and invariant during the layering. The maximum magnitude δ_{rand} of the random part of the time-shift can be chosen arbitrarily but should be small compared to the constant part δ_{const} to avoid introducing artifacts. The theoretical maximum number of virtual clapping people possible without risking that the circularly shifted version of a signal is the signal itself again is determined by

$$P_{\max} = \text{floor}\left(\frac{T}{\delta_{\text{const}} + \delta_{\text{rand}}}\right) \cdot B. \tag{9}$$

Please note that this is an estimate for the worst case scenario.

With the microphone's omnidirectional pickup pattern, the determined parameters and signals can be inserted into Equation (5) yielding the microphone signal or final layered applause signal, respectively.

3.3. Metadata

In order to be able to evaluate additional attributes like clap rate, each captured base signal was manually annotated with respect to individual claps. This was done by local peak picking, i.e., a marker was set to every point in time of maximum absolute amplitude of an individual clap. For each base signal a corresponding metadata signal $\gamma_b(t)$ can be generated containing ones at the

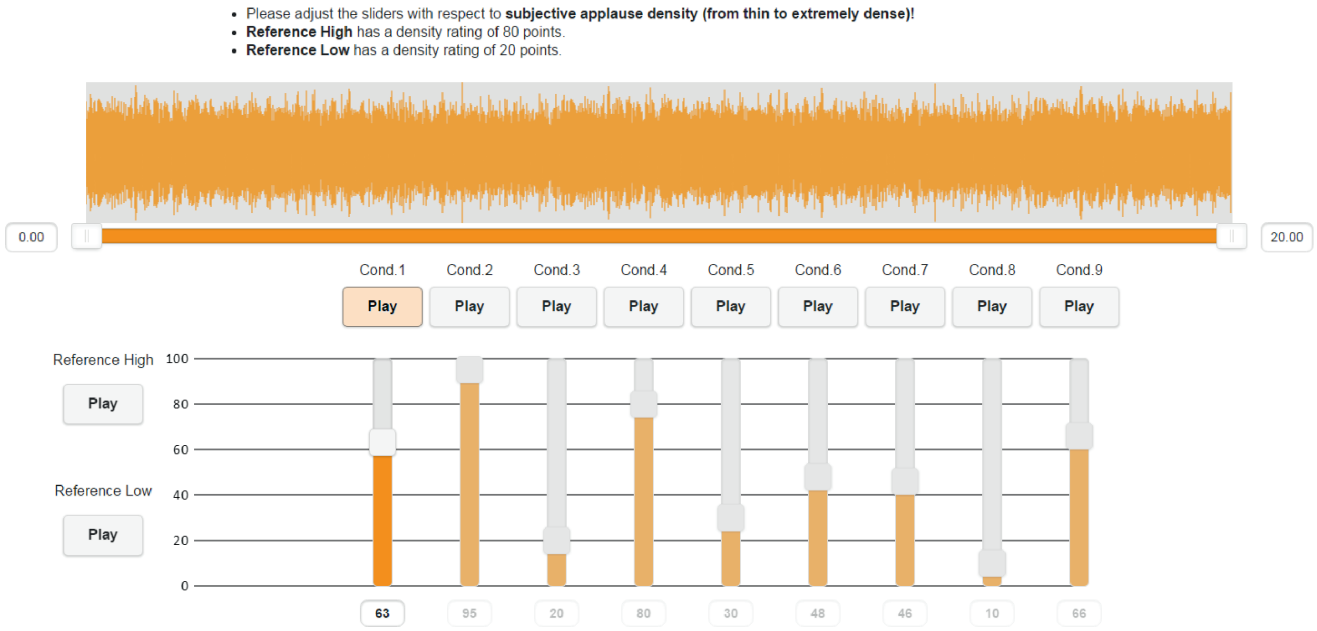


Figure 4: *Customized webMUSHRA [13] graphical user interface.*

marker positions and zeros everywhere else (see Figure 3). This allows for treating the metadata the same way as the base files and therefore layering the metadata almost the same way: attenuation factors and the microphone's directivity pattern can be omitted since the focus is simply on the question whether a clap is present or not and the microphone is assumed to have a constant directivity of one for all incident angles. The layered metadata signals can be described as

$$\Gamma(t) = \sum_{r}^{R} \sum_{p}^{P_r} \gamma_{r,p}(t). \tag{10}$$

The average clap rate of a synthesized applause signal is given by

$$\rho = \frac{1}{T} \sum_{t}^{T} \Gamma(t). \tag{11}$$

4. EVALUATION

4.1. Synthesized applause signals (Test 1)

To assess the perception of applause density, eight mono applause signals with increasing number of virtual people clapping were generated ranging from rather loose to quite dense applause. The individual clap signals were captured in the acoustically optimized sound lab 'Mozart' at Fraunhofer IIS [14]. This room has a mean reverberation time of $\mathrm{T}_m = 0.33\,\mathrm{s}$ and was designed to fulfill the strict recommendations of ITU-R BS 1116-1 [15]. For every single person, three clap signals with a uniform length of $T = 20\,s$ were recorded. Each person was placed individually in a distance of $d_0 = 1\,\mathrm{m}$ in front of a Neumann KM184 directional microphone (cardioid) and successively shown a picture of applauding people at three different venues and asked to clap their hands as if they were part of this crowd. For the layering, the signals were pooled yielding an overall number of $B = 24$ base signals.

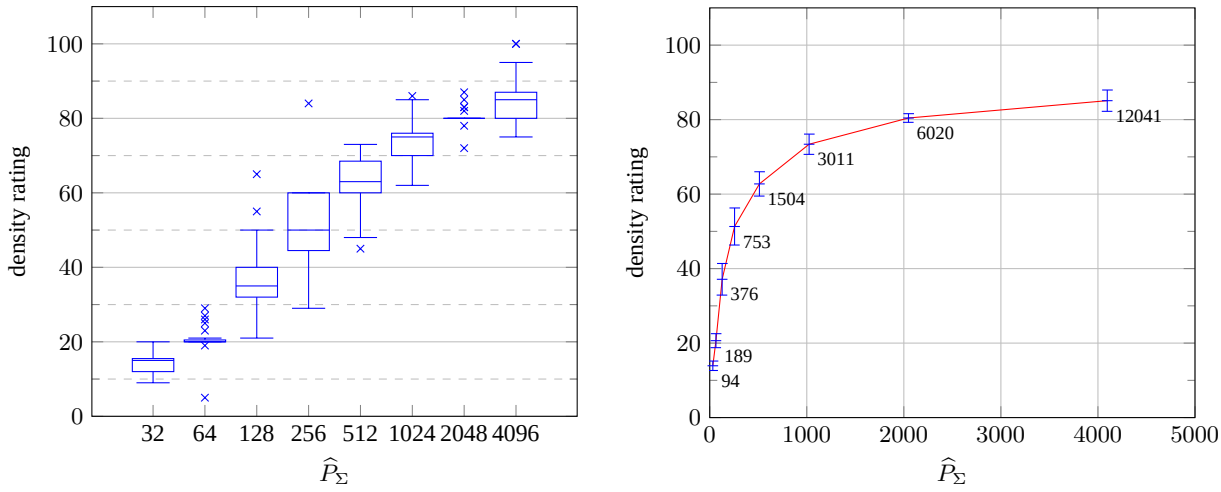
In order to be able to generate applause signals with a higher number of people clapping, the constant time shift δ_{const} had to be decreased compared to the default value in Table 1. The actual values for the constant time shift for a given number of virtual people as well as the corresponding theoretical limit P_{max} according to Equation (9) is given in Table 2. Please note that it was made sure by three persons during informal listening that the generated applause files do not contain artifacts due to the circular time shifting even if the number of virtual people exceeds the theoretical bound. The synthesized applause signals are available at [16].

$\widehat{P}_\Sigma$	32	64	128	256	512	1024	2048	4096
$\delta_{\mathrm{const}}[s]$	1.15	1.15	1.15	1.15	1.15	0.599	0.3	0.15
P_{max}	384	384	384	384	384	672	1200	1920

Table 2: *Parametrization of the layering algorithm for a given number of virtual people and theoretical limit P_{max} according to Equation (9).*

The stimuli were presented to the listening test participants side by side in a multi-stimulus test and as blinded conditions in randomized order. They were to be judged according to their perceived subjective applause density on a scale ranging from 0 to 100 density points and relative to each other. The conditions corresponding to $\widehat{P}_\Sigma = 64$ and $\widehat{P}_\Sigma = 2048$ were used as references corresponding to density levels of 20 and 80 density points, respectively. They were placed next to these values on the density scale and could be listened to any time for reference. However, they were also hidden among the test conditions. This means, the participants also had to identify these hidden references among the stimuli as equally dense compared to the respective reference signals and put them to the corresponding density values.

The test was conducted using Sennheiser HD 650 headphones. As graphical front end, a customized version of the webMUSHRA



(a) *Boxplot of density ratings depending on the number of virtual people clapping ($\widehat{P}_\Sigma = 64$ and $\widehat{P}_\Sigma = 2048$ correspond to the two references).*

(b) *Mean and t-distribution-based 95% confidence intervals. Corresponding clap rate $\rho\left[\frac{claps}{s}\right]$ is written next to confidence intervals.*

Figure 5: *Visualization of participants' responses for the listening test using synthesized applause signals.*

[13] tool was used. A screen shot of the graphical user interface is depicted in Figure 4.

4.2. Naturally recorded applause signals (Test 2)

Applicability to naturally recorded applause signals was assessed in a second listening test. The general test procedure was kept as in the first test but naturally recorded signals with different levels of applause density served as test and reference conditions. The following items were used, where *BBC Applause* and *ARL Applause* were part of the MPEG Surround item test set [17], *Klatschen* was part of the item set used in [18], *SmallCrowdClapping 2* [19] and *Initial Applause* [20] were taken from the freesound web site, *Intro3* and *17Exerc7* were taken from Frank Zappa's 'The Yellow Shark' where the first is an excerpt of the last applause at the end of the first track and the latter is an excerpt of the applause at the end of the 17th track of that record. Applause signals were looped and passively downmixed where necessary to obtain mono signals with a uniform length of 20 s. Additionally, two synthetically generated applause signals with $\widehat{P}_\Sigma = 32$ and $\widehat{P}_\Sigma = 1024$ where included to establish a connection between both listening tests. An overview of the used items and the condition number mapping is given by Table 3. In this test, conditions 2 and 7 served as 20 and 80 density point references, respectively.

Condition	Name
1	synthesized ($\widehat{P}_\Sigma = 32$)
2	klatschen
3	SmallCrowdClapping 2
4	Initial Applause
5	ARL Applause
6	synthesized ($\widehat{P}_\Sigma = 1024$)
7	BBC Applause
8	Intro3
9	17Exerc7

Table 3: *Mapping of item name and condition number of the natural applause signals.*

Test 1									
$\widehat{P}_\Sigma$	32	64	128	256	512	1024	2048	4096	
mean	13.91	20.65	37.13	51.30	62.74	73.39	80.43	85.09	
sd	2.94	4.35	9.84	11.53	7.57	6.31	2.69	6.63	
Test 2									
cond	1	2	3	4	5	6	7	8	9
mean	16.65	20.12	30.82	56.94	58.00	70.94	80.35	91.41	93.18
sd	8.10	2.89	17.51	12.34	14.21	9.16	3.90	8.24	12.80

Table 4: *Mean ratings and standard deviations for synthesized signals (upper table) and natural recordings (bottom table) including corresponding number of people clapping or condition number, respectively.*

5. RESULTS

5.1. Synthesized applause signals

In the first listening test, 23 participants, among which 20 male and 3 female, with an average age of 26.6 years (SD[1]=8.1) ranging from 18 to 53 years took part. Figure 5a shows boxplots of the raw data grouped by the number of people clapping. The very small to non-existent inter-quartile ranges of the reference conditions ($\widehat{P}_\Sigma = 64$ and $\widehat{P}_\Sigma = 2048$) indicate that most participants could easily identify the references and put the slider exactly to the desired value. The apparently high number of outliers for these conditions might result from the slider's measurements which itself cover a range of about 10 density points and therefore add some inaccuracies. If responses which lie within a ± 10 density point tolerance region are considered to have met the reference condition, only one response for $\widehat{P}_\Sigma = 64$ is to be considered as a true miss. An approximately logarithmic increasing perception

[1] SD = standard deviation

of applause density can be observed (approximately linear in the logarithmic presentation of the people clapping).

Figure 5b depicts means and t-distribution-based 95% confidence intervals of the participants' responses in a linear domain. All confidence intervals are non-overlapping. Also the distance from the upper confidence interval limit for $\widehat{P}_\Sigma = 2048$ to the lower limit for $\widehat{P}_\Sigma = 4096$ is 0.6 density points. This indicates that all conditions were perceptually well distinguishable. The red line, connecting the mean values, illustrates the quasi-logarithmic connection of people clapping and perceived density and also indicates that density cannot increase infinitely but saturates at some level. As a reference, the average clap rate is written next to the confidence intervals. It provides information of how many discrete events per second an applause signal consists of. The clap rate increases roughly linearly with the number of people clapping. In general, the results show that applause density can be rated consistently and density perception grows roughly logarithmically with increasing number of people.

5.2. Naturally recorded applause signals

Figure 6 shows boxplots of the responses of the listening test with naturally recorded applause signals. 17 participants among which 14 male and 3 female with an average age of 27.6 years (SD=9.1) ranging from 19 to 53 years were asked to rate applause density of naturally recorded applauses. Except for one response, the reference conditions (conditions number 2 and 7, respectively) can be considered to be recognized within the same ± 10 density points range of tolerance. The two synthesized applause signals correspond to condition 1 ($\widehat{P}_\Sigma = 32$) and 6 ($\widehat{P}_\Sigma = 1024$), respectively. The plot shows that it is possible, based on participants' responses, to produce a plausible ranking of the stimuli according to mean perceived density. It also shows that the spread of density ratings per stimulus is much wider if rating naturally recorded applause signals instead of synthesized ones, which indicates higher uncertainty of the participants. This is also supported by considering the higher standard deviations of the density ratings in the second listening test as given in Table 4. Between the first and the second test, the mean standard deviation increases from 6.48 to 9.91. Also, the average time a participant needed to pass through the test increased around 60% from 2.52 min (SD=1.93) to 4.17 min (SD=2.30) although the second test had only one additional item. This leads to the conclusion that applause density can also be rated consistently for naturally recorded applause signals but participants need more time, i.e., it appears to be harder, and the responses include more uncertainty.

6. CONCLUSION

This paper investigated the perceptual property that may be attributed to sound textures consisting of dense pseudo-random transient events like applause signals. Specifically, we proposed a novel perceptual attribute 'density' to characterize such sound textures. In order to exemplary verify the viability of this attribute on applause signals, a set of applause stimuli of varying densities were created. The generating procedure based on layering started from a set of dry clap recordings and placed these signals in a simple model of a virtual space where virtual clapping people are located in circular rows centered around the virtual microphone considering distance dependent level and timbre. A dedicated listening test methodology was designed based on a multi-stimulus

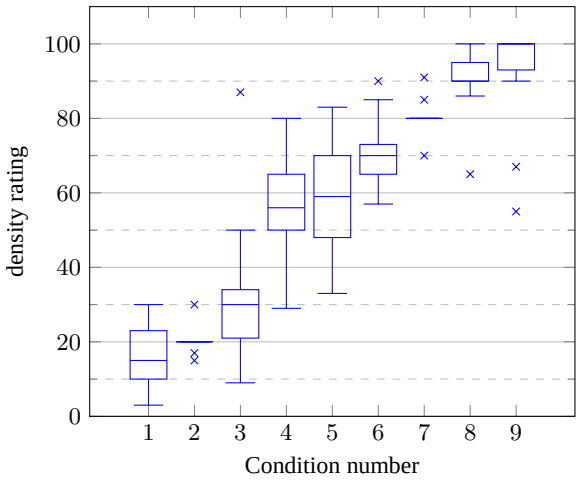


Figure 6: *Boxplots of the participants' responses in the listening test using naturally recorded applause signals (condition 2 and 7 correspond to the 20 and 80 density point references).*

test. Two tests were performed for assessment of the subjective applause density, one using the layered stimuli and the other testing natural applause recordings. The results showed that listeners reacted consistently and were able to reliably distinguish between different applause densities. It was shown that the increase in subjective density points roughly follows the logarithm of the number of clapping people. Therefore, it is evident that density is indeed a psychoacoustic property closely related to the physical measure of an impact rate, but more meaningful in terms of perception. Moreover, the applicability of the applause density attribute to naturally recorded applause signals was confirmed in a second listening test.

7. ACKNOWLEDGMENT

The authors would like to thank Michael Schoeffler for his technical support while customizing the webMUSHRA listening test environment.

8. REFERENCES

[1] B. H. Repp, "The Sound of two hands clapping: An exploratory study," *Journal of the Acoustical Society of America*, vol. 81, no. 4, pp. 1100–1109, 1987.

[2] L. Peltola, C. Erkut, P. R. Cook, and V. Välimäki, "Synthesis of Hand Clapping Sounds," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 3, pp. 1021–1029, 2007.

[3] A. Jylhä and C. Erkut, "Inferring the Hand Configuration from Hand Clapping Sounds," in *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx-08)*, Espoo, Finnland, 2008.

[4] D. Schwarz, "State of the Art in Sound Texture Synthesis," in *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx-11)*, Paris, France, 2011, pp. 221–231.

[5] J. H. McDermott and E. P. Simoncelli, "Sound Texture Perception via Statistics of the Auditory Periphery: Evidence

from Sound Synthesis," *Neuron*, vol. 71, no. 5, pp. 926–940, 2011.

[6] W.-H. Liao, A. Roebel, and A. W. Su, "On the Modeling of Sound Textures Based on the STFT Representation," in *Proceedings of the 16th International Conference on Digital Audio Effects (DAFx-13)*, Maynooth, Ireland, 2013.

[7] J. Brümmerstedt, R. McWalter, and T. Dau, "Analysis and synthesis of environmental sound based on auditory principles," in *DAGA*, Nuremberg, Germany, 2015, pp. 1452–1455.

[8] S. Siddiq, "Morphing of Granular Sounds," in *Proceedings of the 18th International Conference on Digital Audio Effects (DAFx-15)*, Trondheim, Norway, 2015, pp. 4–11.

[9] Z. Néda, E. Ravasz, T. Vicsek, Y. Brechet, and A.-L. Barabási, "Physics of the rhythmic applause," *Phys. Rev. E*, vol. 61, no. 6, pp. 6987–6992, 2000.

[10] Z. Néda, E. Ravasz, Y. Brechet, T. Vicsek, and A.-L. Barabási, "The sound of many hands clapping: Tumultuous applause can transform itself into waves of synchronized clapping," *Nature*, vol. 403, pp. 849–850, 2000.

[11] A. Masurelle, "Towards a gestural control of environmental sound texture synthesis," Master's thesis, Pierre and Marie Curie University (UPMC), Sorbonne Universités, Paris, France, 2011.

[12] K. Kawahara, Y. Kamamoto, A. Omoto, and T. Moriya, "Evaluation of the Low-Delay Coding of Applause and Hand-Clapping Sounds Caused by Music Appreciation," in *138th Convention of the AES*, Warsaw, Poland, 2015.

[13] M. Schoeffler, F.-R. Stöter, B. Edler, and J. Herre, "Towards the Next Generation of Web-based Experiments: A Case Study Assessing Basic Audio Quality Following the ITU-R Recommendation BS.1534 (MUSHRA)," in *1st Web Audio Conference*, Paris, France, 2015.

[14] A. Silzle, S. Geyersberger, G. Brohasga, D. Weninger, and M. Leistner, "Vision and Technique Behind the New Studios and Listening Rooms of the Fraunhofer IIS Audio Laboratory," in *126th Convention of the AES*, Munich, Germany, 2009.

[15] ITU-R BS.1116-1, "Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems," 1997.

[16] Audio Examples of Synthesized Layered Applause Signals. [Online]. Available: https://www.audiolabs-erlangen.de/resources/2016-DAFx-ApplauseLayering/

[17] J. Breebaart, J. Herre, C. Faller, J. Rödén, F. Myburg, S. Disch, H. Purnhagen, G. Hotho, M. Neusinger, K. Kjörling, and W. Oomen, "MPEG Spatial Audio Coding / MPEG Surround: Overview and Current Status," in *119th Convention of the AES*, New York, USA, 2005.

[18] A. Kuntz, S. Disch, T. Bäckström, and J. Robilliard, "The Transient Steering Decorrelator Tool in the Upcoming MPEG Unified Speech and Audio Coding Standard," in *131st Convention of the AES*, New York, USA, 2011.

[19] Small Crowd Clapping 2. [Online]. Available: https://freesound.org/people/snakebarney/sounds/138114/

[20] Initial Applause. [Online]. Available: https://freesound.org/people/unfa/sounds/207728/

TIME DOMAIN ASPECTS OF ARTIFACT REDUCTION IN POSITIONING ALGORITHM USING DIFFERENTIAL HEAD-RELATED TRANSFER FUNCTION

Dominik Storek *

Department of Radioelectronics, Faculty of Electrical Engineering,
Czech Technical University in Prague
Prague, Czech Republic
storedom@fel.cvut.cz

ABSTRACT

This paper focuses on consequences of artifact reduction in virtual sound source positioning method based on Differential Head-Related Transfer Function (DHRTF). As resulted from previous experiments, spatial performance of this experimental method is very promising. However, under specific circumstances, artifacts may occur in the virtually positioned sound. Effective methods for artifact reduction were introduced before. This work discovers impact of the reducing algorithm in the time domain in order to understand phenomena occurring in the process. The cause of artifact presence results from narrow band peak(s) present in the DHRTF magnitude, which causes periodical character of the impulse response in the time domain.

1. INTRODUCTION

The HRTF contains localization cues for a human listener, i.e. Interaural Time Difference (ITD), Interaural Level Differences (ILD) and spectral cues [1, 2, 3]. To obtain virtually positioned sound by the HRTF method, a convolution of the original signal with appropriate HRIR pair is required. Many articles have already dealt with more effective measuring [4, 5] or rendering of the HRTFs [3]. However, simplifying the positioning process focused on reduction of the computational resources is not a well-explored issue yet. The effort was put on finding an algorithm with better spatial performance than simple amplitude panning and with less processing requirements than the HRTF method. This paper aims at time domain aspects of virtual sound source positioning method primarily focused on reducing the computational costs, called *Differential Head-Related Transfer Function* (DHRTF) [6, 7]. The essence of the DHRTF algorithm lies in introducing the frequency dependent ILD and ITD to the stereo signal not by separate filtering of both channels of the binaural signal (as is in the HRTF method [8]), but by filtering only one channel in a way that the same inter-channel differences will occur in the stereo sound, as when the HRTF method is applied. Therefore, only one channel is processed (always the contra-lateral), while the other one remains completely untouched. The DHRTF can be obtained from existing pair of HRTFs as

$$H_D(\vartheta,\omega) = \frac{|H_C(\vartheta,\omega)|}{|H_I(\vartheta,\omega)|}\cdot e^{j(\psi_C-\psi_I)} = \tag{1}$$

$$= \mathrm{A}_D(\vartheta,f)\cdot e^{j\cdot\Psi_D(\vartheta,f)},$$

where indices C and I stand for contra-lateral (farther) and ipsi-lateral (closer) ear. Equivalent of the DHRTF in time domain is called Differential Head-Related Impulse Response (dHRIR) and can be defined as

$$h_c(\vartheta,t) = \mathfrak{F}^{-1}\Big\{H_D(\vartheta,\omega)\Big\} \tag{2}$$

In some HRTF pairs, a specific phenomenon occurs. In unfavorable constellation the attenuation of the ipsi-lateral channel may be greater than in the contra-lateral (against expectation) for particular frequencies, having a character of a narrow-band notch. From the definition of the DHRTF, the same ILD is present. Therefore, the artifacts are caused by presence of sharp peak in the DHRTF (spike) exceeding the level of 0 dB. Since the property of the DHRTF lies in frequency-dependent attenuation and time delay of the contra-lateral channel, positive value of the DHRTF results in boosting the specific band in this channel.

The phenomenon was named Negative Interaural Level Difference (NILD) [9]. The NILD occurs especially within DHRTF derived from a real measured set. The origin of its occurrence results from a unique constellation of the HRTFs in a pair. As shown in Fig. 1 (a), the transfer function of the ipsi-lateral channel can have a spectral notch, which crosses the transfer function of the contra-lateral channel, i.e. the contra-lateral gain is lower than the ipsi-lateral for specific frequency band. This results in spectral spike presence in the DHRTF magnitude, as shown in Fig. 1 (b), black line. Panel (b) contains DHRTF derived from two different sets of HRTF corresponding to the same position. The black line refers to the inappropriate one. However, the presence of the spike in the DHRTF is neither determined for specific spectral bands, nor for specific positions and appears to occur chaotically. For instance, the gray line in the figure represents DHRTF for the same position obtained from another subject. No spectral spike is observable here. Its occurrence is different among various DHRTF sets [9].

Figure 2 shows the extracted NILD spikes in one DHRTF set constructed from a measured set of HRTF [10]. This set of DHRTF is heavily distorted by the NILD spectral features. As can be observed, the NILD occurs mostly in form of narrow spikes. However, even wider frequency bands can appear in particular DHRTFs, especially around frontal axis, i.e. positions $\vartheta = 0^\circ$ and $\vartheta = 180^\circ$. Due to the principle of the method, the artifacts are generally likely to occur around these positions. The Negative ILD may generate noticeable disturbing artifacts of the positioned sound. Perception effect of the mentioned spectral spike is a pure tone character disturbance in the contra-lateral channel. This undesirable phenomenon can be avoided either by selection of appropriate DHRTF set (which actually limits personalization) or by adjusting of the selected individual set of the DHRTF. This adjustment is

* This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS14/ 204/OHK3/ 3T/ 13.

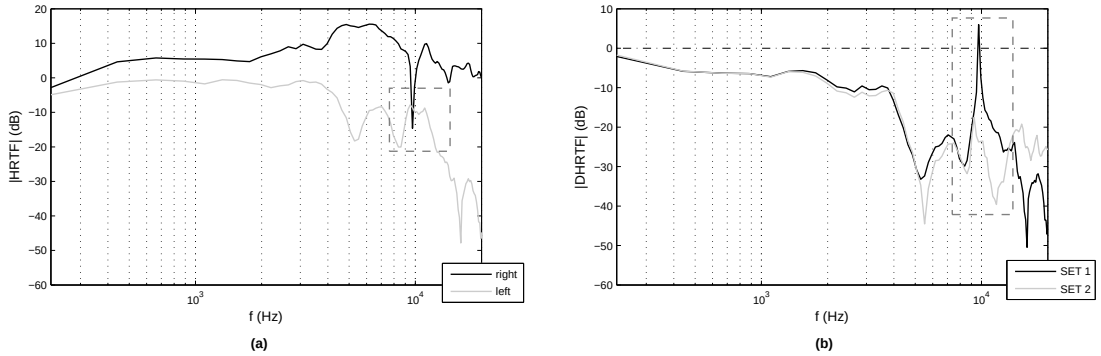


Figure 1: **Artifact origin.** Magnitude of left and right HRTF for $\vartheta = 70^\circ$ and $\vartheta = 0^\circ$ (a). When a notch attenuation of the HRTF for the ipsi-lateral ear crosses HRTF of the contra-lateral, a spike-like peak in DHRTF occurs (b). Spectral peaks and notches vary uniquely in accordance with source position and selected HRTF set.

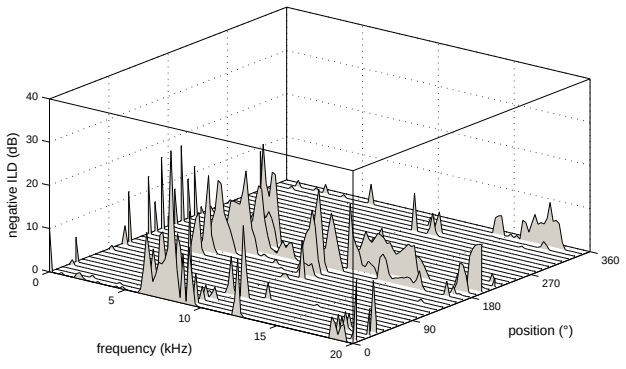


Figure 2: **Negative ILD in the DHRTF.** Extracted spectral spikes of the Negative ILD ($|H_d[\Omega]| > 0$) extracted from DHRTF. Notice highest spike occurrence around positions $\vartheta = 0^\circ$ and $\vartheta = 180^\circ$.

considered to be performed prior to practical use. Previous work published in [9] focused primarily on design of a simple artifact reduction method and verification of its efficiency by subjective listening tests. This paper extends [9] by presenting time domain aspects of application of the artifact reduction method that were investigated thereafter.

2. ARTIFACT REDUCTION

Several artifact reduction approaches were introduced in [9]. According to performed subjective tests, the most effective algorithm contains the following steps.

2.1. Spectral amplitude limiting

The first step performs hard limitation of the DHRTF magnitude curve, where every spectral sample exceeding the threshold (i.e. the negative ILD spike) is reduced to this reference level. This algorithm can be written as

$$\left|\widehat{H}_D^{\vartheta}[\Omega]\right| = \begin{cases} \varepsilon & \text{for } \left|H_D^{\vartheta}[\Omega]\right| > \varepsilon \\ \left|H_D^{\vartheta}[\Omega]\right| & \text{elsewhere,} \end{cases} \tag{3}$$

where Ω denotes spectrum of a discrete signal, $\widehat{H}_D^{\vartheta}[\Omega]$ represents adjusted DHRTF for particular position ϑ, $H_D^{\vartheta}[\Omega]$ represents initial DHRTF, and ε stands for the threshold level. The reference level was considered 0 dB in the logarithmic scale, which is equivalent to 1 in the linear scale. The tests showed that the simple *spectral limiting* may sometimes appear insufficient due to remaining local maximums in the transfer function *under* the reference level ε. Therefore, the algorithm shall be extended by smoothening of the DHRTF magnitude while preserving ILD localization cues.

2.2. Spectral smoothing

This step of smoothing the DHRTF curve by moving average, when each sample of the transfer function is obtained as an average of its neighbors. This process can be also interpreted as convolution of the DHRTF with convolution kernel $M_A[\Omega]$, as described below.

$$\left|\bar{H}_D^{\vartheta}[\Omega]\right| = \left|\widehat{H}_D^{\vartheta}[\Omega]\right| * M_A[\Omega] \tag{4}$$

The $M_A[\Omega]$ is defined as a series of uniformly weighed coefficients of length K with amplitude of $1/K$, as shown here:

$$M_A[\Omega] = \begin{cases} 1/K & \text{for } \Omega \in (0, K) \\ 0 & \text{elsewhere.} \end{cases} \tag{5}$$

The $M_A[\Omega]$ can be comprehended as coefficients of impulse response of a low-pass FIR filter. This filtering procedure ensures that the magnitude of the DHRTF will not cross the reference level ε again. Block diagram representing the signal flow in the artifact reduction method and obtaining the pre-processed dHRIR ready for virtual sound source positioning is summarized in Fig. 3. As can be seen in the diagram, all the processing is done within the DHRTF magnitude. In the first step, the phase of the DHRTF is extracted as

$$\Psi_c^{\vartheta}[\Omega] = arg\Big(H_D^{\vartheta}[\Omega]\Big) \tag{6}$$

and after the processing algorithms, the original phase is returned to the signal.

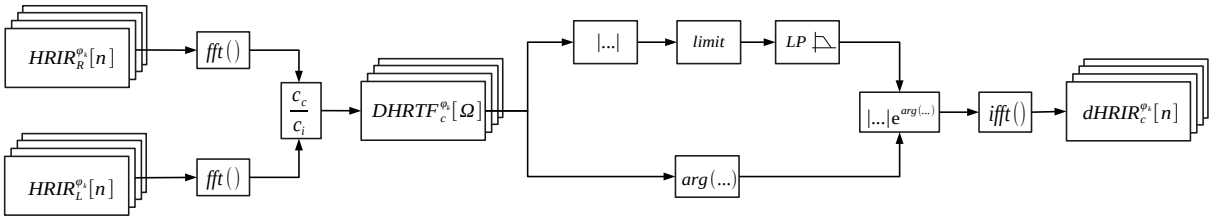


Figure 3: **Processing diagram.** A block scheme of constructing dHRIR with recommended pre-processing for reduction of the artifact occurrence. Once the DHRTF is obtained from the HRTF pair, the phase is extracted, the module is finally limited and smoothed, and merged with the original phase. Inverse Fourier transform is the last step.

3. TIME ANALYSIS

The impact of the introduced algorithms in spectral domain has been already presented in details in [9]. This section focuses on the final effect of the artifact reduction in the time domain. The phase characteristics contains the information about the ITD. For demonstration, see actual values of both HRIR's onsets and resulting ITD obtained from a set of HRIRs measured on acoustic manikin available in [10], which are shown in Fig. 4. The black line denotes onset of the left channel response, the red line denotes onset of the right channel response, and the dashed line shows resulting inter-channel ITD in regular shape of two triangles. The onset time was obtained by thresholding of the impulse response energy. The time information was quantized to particular samples in standard sampling rate of 44.1 kHz. The maximum ITD usually corresponds to approximately 29-33 samples in dependence on the head proportions. The ITD is symmetrical with almost linear character. Figure 5 shows a pure set of dHRIRs for an azimuth in range of $\vartheta \in (0, 360)$ with step of 5 degrees. The dHRIR data were constructed from the author's own measured HRTF set. The particular spike-like spectral features similar to the one demonstrated in Fig. 1 results in pseudo-periodical character of the dHRIR. Several selected responses corresponding to heavily distorted DHRTFs are marked in red. The biggest distortion is observable for positions close to frontal axis, i.e. near $\vartheta \in \{0, 180\}$.

For demonstration of the final effect of the artifact reduction algorithm on the dHRIR response, see Fig. 6. After application of the reduction algorithm, the pseudo-periodicity of the particular responses is suppressed as the spike-like spectral features in the DHRTF magnitude are removed. Notice also decreased noise level in the dHRIRs. Application of the low-pass filter performed by moving average also reduces the *noisy tale* of the response. Another important feature in the dHRIR set is a pair of clearly visible triangular shapes in the horizontal plane indicating the onset of the dHRIR. The shift corresponds to the ITD and the triangle profile is the same as presented in Fig. 4.

As stated above, the dHRIR (DHRTF) contains information about relative time shift and relative attenuation of the contralateral (farther) channel in relation to the ipsi-lateral (closer). The ITD is visible through the onset and the attenuation is observable through energy of the response. For positions close to the axis of $\vartheta \in \{0, 180\}$, the response energy is much higher than for the side positions $\vartheta \in \{90, 270\}$, where the attenuation of the contralateral channel is the highest.

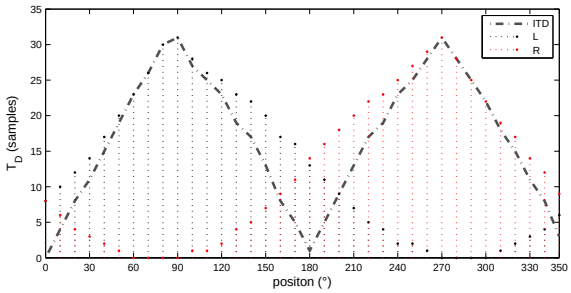


Figure 4: **ITD from HRTF set.** Interaural Time Delay extracted form a real HRTF set by detecting signal energy onset of each impulse response. The dotted black and red lines corresponds to the time shift of left and right HRIRs, respectively. Dashed triangles represent resulting ITD.

4. CONCLUSIONS

This paper presents time domain scope of the effect of artifact reduction algorithm for DHRTF-based positioning and extends previous work published in [9], where also perceptual analysis by listening tests is present. The basis of the algorithm is hard limiting of the DHRTF magnitude and smoothening of the magnitude by low pass filtering performed by moving average convolution kernel. The most affected dHRIRs are around the front-back axis. As the artifact reduction algorithm removes local maxima (peaks) in the DHRTF magnitude, the equivalent in the time domain is removing the periodicity of the impulse response. The algorithm preserves identical time domain localization cues (ITD). The ILD information is modified; however, the main features in the frequency domain are still preserved. Future work will be focused on the origin of the Negative ILD in the data sets as it is present primarily in the measured HRTF sets.

5. ACKNOWLEDGMENTS

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS14/ 204/OHK3/ 3T/ 13. Thanks to C. Fialova for fruitful discussions.

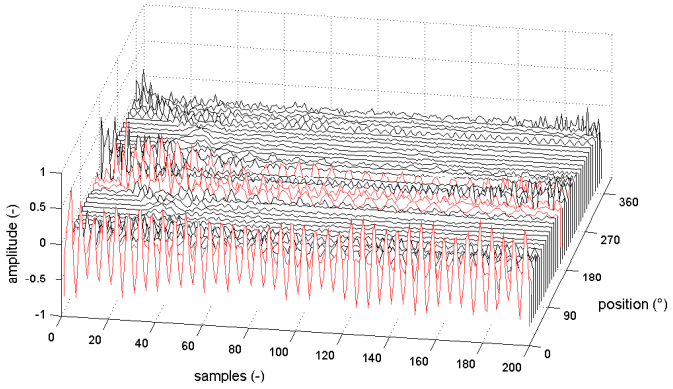


Figure 5: **An original dHRIR set.** A set of $dHRIR[n]$ derived from author's own set of $HRTF$. Particular responses affected subjected to artifact occurrence resulting from spectral negative ILD spikes are highlighted by red color.

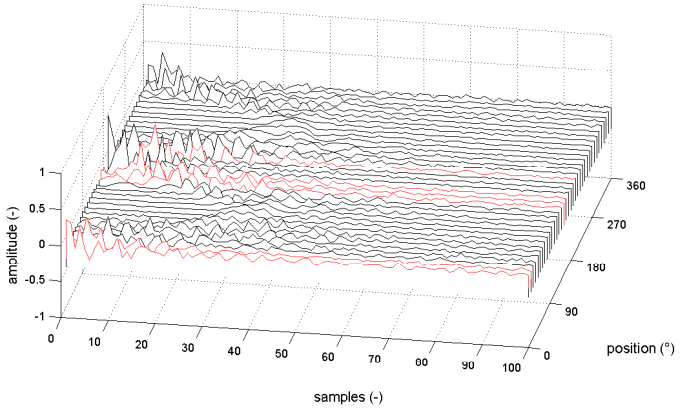


Figure 6: **A processed dHRIR set.** The effect of processing focused on artifact reduction is well observable. The problematic responses highlighted in red looses their pseudo-periodical character and extensive amplitude.

6. REFERENCES

[1] Jens Blauert, *Spatial hearing: the psychophysics of human sound localization*, MIT press, 1997.

[2] Jens Blauert, *The technology of binaural listening*, Springer Verlag, Berlin, 2013, eBook.

[3] Robert Baumgartner, Piotr Majdak, and Bernhard Laback, "Modeling sound-source localization in sagittal planes for human listeners," *The Journal of the Acoustical Society of America*, vol. 136, no. 2, pp. 791–802, 2014.

[4] Areti Andreopoulou, Agnieszka Rogińska, and Hariharan Mohanraj, "A database of repeated head-related transfer function measurements," 2013.

[5] Chris Oreinos and Jörg M Buchholz, "Measurement of a full 3D set of HRTFs for in-ear and hearing aid microphones on a head and torso simulator (HATS)," *Acta Acust. United Ac.*, vol. 99, no. 5, pp. 836–844, 2013.

[6] Dominik Storek and Frantisek Rund, "Differential head related transfer function as a new approach to virtual sound source positioning," in *Radioelektronika, 2012 22nd International Conference*. IEEE, 2012, pp. 1–4.

[7] Dominik Storek, "Virtual sound source positioning by differential head related transfer function," in *Audio Engineering Society Conference: 49th International Conference: Audio for Games*. Audio Engineering Society, 2013.

[8] Udo Zölzer (ed.), *DAFX: Digital Audio Effects*, vol. 1, John Wiley & Sons, 2002.

[9] Dominik Storek, Jaroslav Bouse, Frantisek Rund, and Petr Marsalek, "Artifact reduction in positioning algorithm using differential HRTF," *Journal of the Audio Engineering Society*, vol. 64, no. 4, pp. 208–217, 2016.

[10] V Ralph Algazi, Richard O Duda, Dennis M Thompson, and Carlos Avendano, "The CIPIC HRTF database," in *IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, 2001, pp. 99–102.

DETECTION OF CLICKS IN ANALOG RECORDS USING PERIPHERAL-EAR MODEL

František Rund
Department of Radioelectronics,
Czech Technical University in Prague
Prague, Czech Republic
xrund@fel.cvut.cz

Václav Vencovský
Department of Radioelectronics,
Czech Technical University in Prague
Prague, Czech Republic
vaclav.vencovsky@gmail.com

Jaroslav Bouše
Department of Radioelectronics,
Czech Technical University in Prague
Prague, Czech Republic
bousejar@fel.cvut.cz

ABSTRACT

This study describes a system which detects clicks in sound (audible degradations). The system is based on a computational model of the peripheral ear. In order to train and verify the system, a listening test was conducted using 89 short samples of analog (vinyl) records. The samples contained singing voice, music (rock'n'roll), or both. We randomly chose 30 samples from the set and used it to train the system; then we tested the system using the 59 remaining samples. The system performance expressed as a percentage of correct detections (78.1%) and false alarms (3.9%) is promising.

1. INTRODUCTION

Any undesirable changes in the audio signal are considered as its degradation. According to [1] the degradations can be classified into two groups: global degradations (e.g. background noise, nonlinear distortion, wow and flutter), and localized degradations. Localized degradations are discontinuities in the waveform present only in some samples, such as impulse noise (clicks, crackles, pops, ticks etc.) In this study, we used the term "click" to classify the localized degradations perceived by the listeners as the characteristic noise which is mainly associated with vinyl records. These degradations very often occur in analog records, for example, in historical records, or as a result of damage during the manufacturing process.

Detection of clicks is a principal part of the restoration process (e.g. [1]) or can be used for the audio quality assessment, for example, output quality check during the manufacturing process of the audio records. Manufactures, such as GZmedia [2], usually perform quality control by listening tests with trained employees, however, it is very cost and time demanding. The existing algorithms for impulse detection are based on time domain modeling of the signal (e.g. [1, 3, 4, 5]), or on wavelet transform approach (e.g. [6, 7]). In all of the aforementioned approaches, a detection threshold has to be set. The choice of the threshold is very often empirical, depends on the type of the signal, parameters of the algorithm, etc. Inappropriate threshold setting leads to false detection or missed clicks.

As stated in [1], it is necessary to focus mainly on perceptible degradations. Therefore in this study, we propose a click-detection system based on a computational model of the peripheral ear. The peripheral ear model consists of a physical cochlear model which we previously used for other purpose related with perception [8]. We trained and tested the system by using 89 short sound samples of real music, which contained perceptible clicks. The music samples, which were provided by the vinyl record manufacturer GZmedia, were extracted from four different songs of rock'n'roll music. We conducted a listening test in order to measure the presence of audible clicks.

2. SYSTEM BASED ON A PHYSICAL COCHLEAR MODEL

Figure 1 shows a diagram of the system. The first two blocks represent algorithms simulating the function of the peripheral ear. The remaining blocks process the output signal of the peripheral ear model – this signal is called "internal representation" of the analyzed sound – and give the answer whether the sound contains audible click(s) and temporal position of the click(s) in the signal.

The model of the peripheral ear is composed of two parts, both adapted from the literature. The first part simulates the transformation of the acoustic wave at the entrance of the outer ear into the vibrations of the stapes (the input of the inner ear). The model was adapted from the system called Matlab Auditory Periphery [9]. Resonances of the outer-ear canal are modeled by two parallel 1st-order Butterworth bandpass filters: the first with a gain of 10 dB, lower cutoff frequency of 2.5 kHz, and higher cutoff frequency of 4 kHz; and the second with a gain of 25 dB, lower cutoff

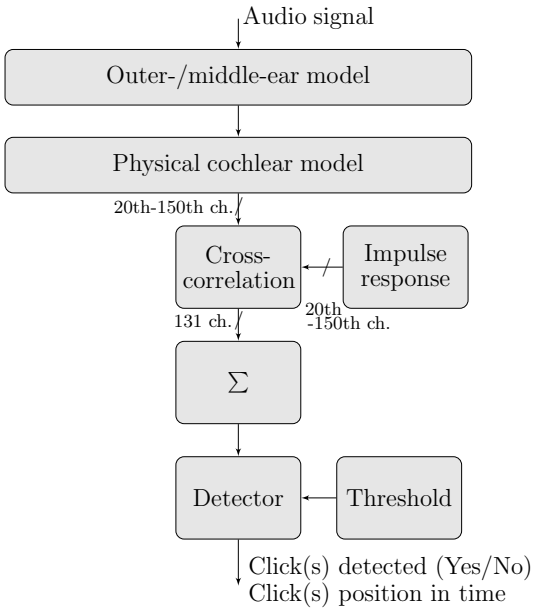


Figure 1: *Diagram of the click detection system*

frequency of 2.5 kHz, and higher cutoff frequency of 7 kHz. The input acoustical signal is first filtered by the two parallel band-pass filters. Then the both filtered signals and the input signal are summed together. The created signal is then processed by a middle ear model transforming it into the stapes displacement. The middle-ear model is composed of two cascaded first order Butterworth filters: a high-pass filter with a cutoff frequency of 50 Hz, and a low-pass filter with a cutoff frequency of 1 kHz. The signal at the output of the second filter is then multiplied by a constant of 45×10^{-9}, which transforms the signal into the displacement (in meters) of the stapes.

The second part of the peripheral ear model transforms the stapes vibrations into the vibrations of the longitudinal segments of the basilar membrane inside the cochlea. The cochlea conducts spatial-frequency analysis – the high frequencies of the incoming sound excite the basilar membrane near the basal site, whereas the low frequencies near the opposite (apical) site. In this paper, the function of the cochlea is simulated by a physical cochlear model described in [10]. The model approximates the basilar membrane by an array of 300 oscillators coupled via surrounding fluid. Therefore the model output is a multichannel signal – the signal in each channel represents displacement of one oscillator. This displacement ξ_i of the i-th oscillator is given by

$$\begin{aligned} &m_i\ddot{\xi}_i(t) + h_i\dot{\xi}_i(t) + s_i[2\dot{\xi}_i(t) - \dot{\xi}_{i-1}(t) \\ &- \dot{\xi}_{i+1}(t)] + k_i\xi_i(t) = f_{H_i}(t) + f_{\mathrm{OHC}_i}[\eta_i(t)], \end{aligned} \tag{1}$$

where m_i, h_i, s_i and k_i are mass, positional viscosity, sharing viscosity and stiffness of the basilar membrane, respectively. Each oscillator is driven by force $f_{H_i}(t)$ given by

$$f_{H_i}(t) = -G_{S_i}a_{S_i}(t) - \sum_{j=1}^{N} G_i^j\ddot{\xi}_j(t), \tag{2}$$

where $a_{S_i}(t)$ is acceleration of the stapes, $\ddot{\xi}_i(t)$ is acceleration of the oscillators, G_{S_i} and G_i^j are transfer functions obtained by solving wave equations. The second force term f_{OHC_i} represents the cochlear amplifier. In the model, the tectorial membrane connected with stereocilia of the outer hair cells is simulated by another array of oscillators. The stereocilia deflection η_i is given by the differential equation

$$\bar{m}_i\ddot{\eta}_i(t) + \bar{h}_i\dot{\eta}_i(t) + \bar{k}_i\eta(t) = -D_i\ddot{\xi}_i(t), \tag{3}$$

where $\bar{m}_i$ is mass, $\bar{h}_i$ is viscous damping and $\bar{k}_i$ is stiffness, and D_i is a constant. The active force f_{OHC_i} is then calculated from η_i, which is first transformed by a sigmoidal nonlinear function, which attenuates the amplification at high intensities [10]. The model parameters are same as those used in [8]. The characteristic frequencies (CFs) of the model channels, i.e., the frequencies of 10-dB pure tone causing the highest excitation in the given channel, were distributed roughly between 30 Hz and 17 kHz. As well as in [10, 8], the model was implemented in the time domain using the implicit Euler method. The accuracy of this method depends on the sampling frequency – rises with increasing sampling frequency. Therefore by assuming that input stimuli have a sampling frequency at least 44.1 kHz, the input signal to the cochlear model was 10-times upsampled before the processing and then the output signal in each model channel was 10-times downsampled.

Figure 2, panel A shows the internal representation (output signal of the peripheral ear model) of a music sound sample which contains a click. The click created a very distinct pattern in the internal representation, which is easily visible especially at high CFs (above about 2.5 kHz); the pattern is in panel A marked by the ellipse. This internal representation is very similar to the internal representation of an impulse (generated as unit impulse with an amplitude of 2 Pa), which is shown in Figure 2, panel B. Therefore the system detects click(s) in sound samples by calculating cross-correlation of the internal representations obtained in response to an analyzed sound sample and to an impulse. The cross-correlation is calculated between the internal representations in corresponding channels. Then the cross-correlations in the individual channels are summed, which creates a signal with a distinct peak indicating a presence of click. Figure 2, panel C shows the absolute value of the summed cross-correlation between the internal representations shown in the same figure. The cross-correlations were summed between the 20th and 150th channels; the CFs of these channels were 14189 Hz and 2419 Hz, respectively. We have chosen this range experimentally. Notice that the channel numbering is inverted – the first channel has the highest CF (see Fig. 2). This order is given by the physical cochlear model [10] in which the first channel simulates the segment of the basilar membrane which is closest

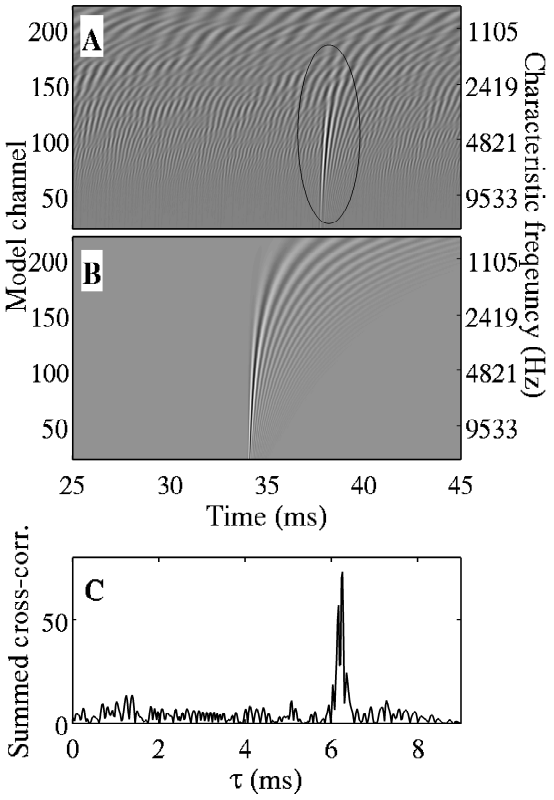


Figure 2: *(A) The auditory model response (internal representation) to a musical signal degraded by a click. The internal representation of the click is marked by the ellipse. (B) The internal representation of an impulse with an amplitude of 2 Pa. (C) Absolute value of summed cross-correlation between the internal representations shown in panels A and B. The cross-correlations were summed across the 20th to 150th model channel (as is indicated in Figure 1).*

to the stapes (input to the cochlea); this segment has the highest CF. The designed system then detects the presence of click(s) by comparing the amplitude of the summed cross-correlation with a previously defined threshold value; the threshold value used in the system was set during the system training described in the next section. If the amplitude is higher than the threshold value, the system detects click(s) (see Figure 1).

Using this method also allows accurate detection of click position in the time domain. This information could be used for a click removal. A disadvantage of the proposed method is that it is computationally demanding, especially the cochlear model. Therefore it cannot be used in real time.

3. LISTENING TEST, SYSTEM TRAINING AND TESTING

In order to measure the presence of click(s) in sound samples, we conducted a listening test. Since the system compares the summed cross-correlation with a previously defined threshold value, it was necessary to set the threshold value by using some of the results of the listening test (to train the system). The remaining results were then used to test the system.

3.1. Listening test

3.1.1. Stimuli

The stimuli were 800 ms long musical samples shaped on its onset and offset with 80 ms long raised cosine ramps. The samples were extracted from wav files with four different songs; the wav files were converted from analog (vinyl) records containing impulse degradations. The samples contained a singing voice, music (rock'n'roll), or both. All the samples were provided by the vinyl record manufacturer GZmedia. The level of the individual samples was not scaled to the same value for all the samples in order to preserve the dynamic range of the music; the samples were presented with a sound pressure level given by its content – if it contained a 1-kHz pure tone with maximum possible amplitude in the wav file, the presented sound pressure level was 94 dB. For further description about the calibration procedure please refer to [11].

3.1.2. Listeners

Four (all males) listeners aged between 24–33 years participated in the experiment. All of them had normal hearing according to their pure tone hearing thresholds – the thresholds between 250 Hz and 8 kHz were within a range of 20 dB of hearing level. The listeners had no prior experience with this type of experiment.

3.1.3. Procedure and equipment

The experiment was divided into three sessions, each session followed by a compulsory pause. The first session provided the listeners a chance to learn the procedure of the experiment. It consisted of 25 stimuli presented three times in a random order. After the first session we asked the listeners whether they had problems with the procedure or with the detection of clicks in the stimuli; none of them rated the experiment or the detection task difficult. The results of the first session were discarded from the evaluation. Finally the second and third session consisted of 89 stimuli presented in random order and the data from these sessions were taken as the results of the listening test.

The listening test was conducted in a sound insulated booth placed in our laboratory. The listeners were sitting in front of a computer monitor (EIZO S2000) and external soundcard (RME Fireface UC), both connected to a computer placed outside of the booth. The listeners controlled the listening test using a mouse and graphical user interface (GUI) programmed in Matlab, and the stimuli were presented via Sennheiser HD 650 headphones. The GUI consisted of three large buttons labeled as "YES", "REPEAT" and "NO". The listeners were presented with a sound sample and asked to press "YES" or "NO" button based on whether they had perceived click(s) in the stimulus. The listeners had a possibility to repeat the presented stimulus as many time as they desired by pressing the "REPEAT" button. After the response and a subsequent 2-second pause, a next stimulus was presented. No feedback was given to the listeners, except the number of the presented sample within the set.

3.2. System training and testing

The overall set of 89 sound samples was randomly divided into a training set with 30 samples and a testing set with 59 samples. During the training, the system was presented with the samples from the training set and the listening tests results of these samples were used to set a threshold value. This threshold value was then used to predict the occurrence of clicks in the samples within the testing set.

As is shown below in Section 4, some of the samples were not rated clearly, for example, fifty percent of the ratings suggested the presence of click(s) and fifty percent not. For the system training it was necessary to set a rule according to which the sample will be claimed to contain click(s). We claimed to contain click(s) those samples which were rated at least in: (1) 75%, or (2) 50% of the all answers across the sessions and listeners. For these samples, we calculated the summed cross-correlations – the cross-correlations were summed between the 20th and 150th model channel; the CFs of these channels were 14189 Hz and 2419 Hz, respectively. We then calculated the maximum absolute value of the summed cross-correlations for each of the sample and then the minimum value of these maxima across the samples. This gave us a threshold value which was then used during the system testing.

After the threshold value was set by using the training sequence, the system performance was evaluated by using the 59 test samples. Based on the results, each sample was labeled either to contain a click or not. The results are given below together with the results of the listening test.

4. RESULTS AND DISCUSSION

Figure 3 shows the results of the listening test and the system predictions. The bars show the percentage of the positive answers – click was detected – across the all listeners and sessions. The circles above each bar indicate those samples for which the click was predicted by the system. The threshold value used in the system was set by using the listening test results on the 30 training sound samples. We assumed that the samples from the training sequence contained a click if the percentage of positive – click was detected – answers was higher or equal to (1) 75%, or to (2) 50%. In fact, both conditions gave the same threshold value (41.5). Therefore the predictions for both conditions are the same. The predictions are shown in Figure 3 by circles.

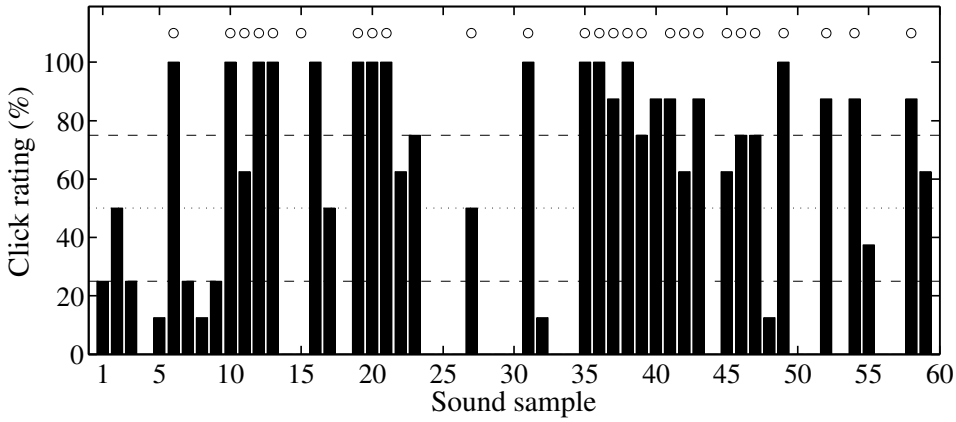


Figure 3: *Results of the listening test and predictions of the click detection system. The bars show the percentage of the click occurrence across the all listeners and responses. Each bar represents one stimulus. Circles indicate the samples in which the system detected a click(s).*

In order to analyze the model performance, we used the "sensitivity measures" for "Yes-No" experiments as is described in [12]. The results of the listening tests were interpreted using two conditions: (1) the samples were marked to contain a click ("Detected") if the rating was equal or higher than 75%, and without a click ("Not detected") if the rating was equal or less than 25% (see the horizontal dashed lines in Figure 3); and (2) the samples were marked "Detected" if the rating was equal or higher than 50%, and "Not detected" if the rating was less than 50% (see the horizontal dotted line in Figure 3). Table 1 shows the calculated model performance. The correct response ("Hit") percentage was calculated by dividing the number of correctly detected clicks by the overall number of samples marked as "Detected" in the listening tests. The "False alarm" percentage was calculated by dividing the number of samples in which the click was predicted and which were marked as "Not detected" by the listeners. The performance of the click detection system is promising; mainly the false alarm rate is very small which may indicate that the threshold value could be smaller to increase the percentage of correct detections.

Table 1: *The system performance.*

	Hit	False alarm
Det. $\geq$75%; Not det. $\leq$ 25%	87.5%	3.9%
Det $\geq$50%; Not det. $<$ 50%	78.1%	3.8%

The clicks were in some of the samples hardly distinguishable from music, e.g., the sound of the plectrum in some samples was very similar to clicks. This contributed to the decreased performance of the detection system. In addition to results of listening tests, in future work we plan to evaluate the system performance against the results of an objective system which uses a reference.

The visual analysis of the model outputs in response to the analyzed sound samples revealed that in some cases the high frequency portion of the model responses may be masked by the musical content. Therefore summing the cross-correlations across a large number of model channels may not be the ideal method for click detection. We plan to focus on these details in future work.

5. CONCLUSIONS

A system allowing to detect the degradation of audio records by clicks (localized degradation caused, for example, by scratches on vinyl records) was designed in this study. The system employs a preprocessing part composed of a computational model of the peripheral ear. The peripheral ear model accounts for the transfer function of the outer and middle-ear, and for the function of the cochlea. The system does not use any reference; it detects clicks by calculating the cross-correlation between the peripheral ear model responses to the analyzed sound and to an impulse. The system sums the cross-correlations across several model channels (at frequencies above about 2.5 kHz) and compares the absolute maximum value with a previously defined threshold value. In order to define the threshold value and then to test the system performance, we used 89 short sound samples (containing signing voice and rock'n'roll music). We evaluated the samples by a listening test in which the listeners were asked whether the samples were degraded by clicks (contained clicks). We randomly chose 30 of the samples for the system training – for setting the threshold value; and then tested the system performance using the remaining 59 samples. The system performance was promising: the correct response rate which depends on the chosen method for the interpretation of the listening test results was higher than 78.1% and the false alarm rate also dependent on the method was smaller than 3.9%. We plan to improve the system accuracy in future work.

6. ACKNOWLEDGMENTS

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS14/204/OHK3/3T/13. The samples of real damaged records were provided by GZmedia company. Authors thank the volunteers who participated in the listening test, and the anonymous reviewers for comments on the previous version of the manuscript.

7. REFERENCES

[1] S. J. Godsill and P. J. W. Rayner, *Digital audio restoration*, Springer-Verlag, London, 1998.

[2] GZmedia, "Pressing GZ Vinyl," Available at http://www.gzvinyl.com/Manufacturing/Pressing.aspx ?sec=Quality-control, accessed March 10, 2016.

[3] S. J. Godsill and P. J. W. Rayner, "A Bayesian approach to the restoration of degraded audio signals," *IEEE Trans. on Speech and Audio Processing*, vol. 3, no. 4, pp. 267–278, 1995.

[4] Ch. Chandra, M. S. Moore, and S. Mitra, "An efficient method for the removal of impulse noise from speech and audio signals," in *Proc. of the 1998 IEEE Intl. Symposium on Circuits and Systems (ISCAS'98).* IEEE, 1998, vol. 4, pp. 206–208.

[5] M. Niedźwiecki and M. Ciołek, "Sparse vector autoregressive modeling of audio signals and its application to the elimination of impulsive disturbances," *IFAC-PapersOnLine*, vol. 48, no. 28, pp. 1208–1213, 2015.

[6] P. Rajmic and J. Klimek, "Removing crackle from an LP record via wavelet analysis," in *Proc. of the 7th Intl. conf. on digital audio effects (DAFx04)*, 2004, pp. 100–103.

[7] R. C. Nongpiur and D. J. Shpak, "Impulse-noise suppression in speech using the stationary wavelet transform," *The Journal of the Acoustical Society of America*, vol. 133, no. 2, pp. 866–879, 2013.

[8] V. Vencovský and F. Rund, "Using a physical cochlear model to predict masker phase effects in hearing-impaired listeners: A role of peripheral compression," *Acta Acustica united with Acustica*, vol. 102, no. 2, pp. 373–382, 2016.

[9] Essex Hearing Research Laboratory, "Auditory modelling at Essex University," Available at http://www.essex.ac.uk/psychology/department/hearinglab/modelling.html, accessed March 10, 2016.

[10] R. Nobili, A. Vetešnik, L. Turicchia, and F. Mammano, "Otoacoustic emissions from residual oscillations of the cochlear basilar membrane in a human ear model," *Journal of the Association for Research in Otolaryngology*, vol. 4, no. 4, pp. 478–494, 2003.

[11] J. Bouše, "Headphones measurement tool implemented in Matlab," in *Proc. of the 19th International Scientific Student Conf. POSTER 2015*, Prague, 2015, pp. 1–5, Czech Technical University in Prague.

[12] N. A. Macmillan and C. D. Creelman, *Detection theory: A user's guide*, Lawrence Erlbaum associates, publishers, Mahwah, New Jersey, London, 2005.

PERCEPTUAL AUDIO SOURCE CULLING FOR VIRTUAL ENVIRONMENTS

Ali Can Metan, Hüseyin Hacıhabiboğlu

Graduate School of Informatics,
Middle East Technical University
Ankara, TR-06800, Turkey
canmetan@engineer.com, hhuseyin@metu.edu.tr

ABSTRACT

Existing game engines and virtual reality software, use various techniques to render spatial audio. One such technique, binaural synthesis, is achieved through the use of head-related transfer functions, in conjunction with artificial reverberators. For virtual environments that embody a large number of concurrent sound sources, binaural synthesis will be computationally costly. The work presented in this paper aims to develop a methodology that improves overall performance by culling inaudible and perceptually less prominent sound sources in order to reduce performance implications. The proposed algorithm is benchmarked and compared with distance-based, volumetric culling methodology. A subjective evaluation of the perceptual performance of the proposed algorithm for acoustic scenes having different compositions is also provided.

1. INTRODUCTION

Virtual environments create the perception of being physically present in a non-physical world via the presentation of synthetic audiovisual stimuli. With today's technology, creation of vivid environments require significant computational power. As such, optimisation of any existing process needs to make better use of the limited resources.

Sound scenes encountered in virtual environments may contain many sound sources. Spatialising all of these sources will impose a performance penalty for the underlying hardware. This problem is aggravated for computationally heavy spatialisation methods such as binaural synthesis which is widely used in VR systems.

In order to deal with the performance implications, most game engines and virtual environment software that are in use today, employ volumetric culling [1]. For volumetric culling, volumes (such as spheres or cubes) are defined for each sound source. Only those sources for which the listener is in the active volume, are rendered. While these methodologies are effective in reducing the amount of sound sources, perceived richness of the resulting scene may also be degraded as a result. Here, we define *auditory richness* as the perceived quantity of sound sources in an real or virtual acoustic scene.

There are also methodologies that incorporate various perceptual approaches. Some of these incorporate crossmodal effects such as culling sound sources that reside outside the view frustum of the listener [2]. This method is based on the assumption that audiovisual correspondence will have a major affect on the perception of sound sources. Other methods will cull or spatially cluster sound sources according to their predicted audibility [3][4].

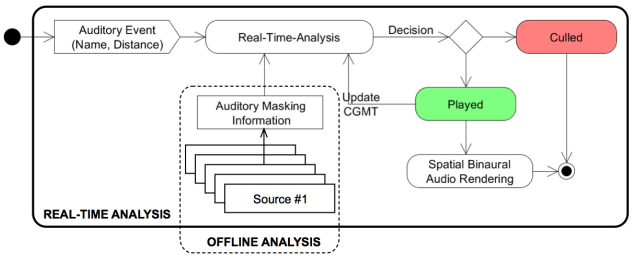


Figure 1: *Overall pipeline for the culling algorithm*

Majority of games and virtual environments include spatially separated, high-energy, broadband sound sources which may also be concurrently active. Explosions, gunshots, engine sounds and other sound sources may render each other inaudible due to auditory masking. If sources that are inaudible can be identified and eliminated from the rendering pipeline, computational savings may be made without degrading the overall quality of the rendered scene. The aim of this paper is to present an algorithm which can select a subset of all sources in the sound scene based on their audibility. The audibility is calculated using an existing model of simultaneous masking which also forms the basis of perceptual audio coding. The sound sources are evaluated and selected according to their perceptual salience due to monaural masking model used in MPEG-1 Layer I. The sound sources selected by using the proposed procedure can then be rendered without significant perceptual degradation of the richness of the auditory scene.

This paper is organised as follows. Sec. 2 presents the proposed algorithm. A performance analysis of the proposed algorithm is given in Sec. 3. The results of a subjective evaluation comparing the proposed culling method with volumetric culling is presented in Sec. 4. Sec. 5 concludes the paper.

2. PERCEPTUAL SOURCE CULLING

The algorithm proposed in this paper, consists of two parts: offline analysis and real time analysis. Extraction of the masking properties of audio sources contained within a scene, is carried out during the offline analysis. During real time analysis, information extracted from the previous stage is used to determine whether a given audio source will be rendered or not. Fig. 1 summarises the algorithm.

2.1. Offline analysis

Offline analysis is performed prior to rendering the actual scene on sound files which may be used in a scene. In this step, masking thresholds of sound source signals are calculated and stored in persistent memory. The model used for calculating the masking thresholds is the MPEG-1 Layer I psychoacoustical model [5]. This application is similar to how the same model is applied in sound synthesis [6]. Since the masking thresholds obtained this way are additive, they need not be calculated at run-time. Details of offline analysis is explained in the remaining parts of this section.

2.1.1. MPEG-1 Audio Layer I Masking Model

MPEG-1 Audio Layer 1 is an audio coding standard incorporating a psychoacoustical model to reduce the bitrate [5]. In the proposed method, this psychoacoustical model is used to determine whether a requested audio event will be audible inside the existing scene or not. The model calculates the masking thresholds of individual sound sources by estimating their tonal and non-tonal components [7]. After relevant maskers have been identified among these components, local masking thresholds are calculated and stored. Details of how the auditory masking thresholds are calculated, is explained briefly below.

As a first step, samples from the audio signal are divided into frames of 512 samples and each individual frame goes under psychoacoustic evaluation. Two separate frequency-domain representations are used. The first representation uses a 32-channel polyphase filter bank for emulating the frequency selectivity of the auditory periphery. The second representation uses a 512-point FFT to determine tonal and noise maskers. Each of the 32 channels generate frames of size 12 resulting in a total of 384 samples. Appropriate time shifting is applied to correct the time delay induced by the filter bank. The frequency resolution of the FFT for a sampling rate of $F_s = 44.1$ kHz is 86.13 Hz. An overlapping Hann window is used in the calculations for obtaining an estimate of the power spectrum, $X(k)$, $k = 0 \dots N/2$ of the frame.

Then, the sound pressure level (SPL) in subband $l = 0 \dots 31$ is computed with respect to a reference of 96 dB. The sound pressure level, $L_{sb}(l)$, is computed for every subband l and stored.

MPEG-1 Audio Layer I psychoacoustical model separates the tonal and noise-like components of the audio signal. The reason for this is the different spreading characteristics of simultaneous masking by these different types of maskers.

The tonal components $X(k)$, are identified based on the local maxima which are determined as the peaks that satisfy the following condition:

$$|X(k)| > |X(k-1)| \qquad \text{and} \qquad |X(k)| >= X(k+1)$$

Subsequently, for each critical band, the remaining noise-like components are summed into a single non-tonal masker component. A local peak is added to a list of tonal maskers if:

$$|X(k)| - |X(k+j)| >= 7 \text{ dB},$$

where j is chosen differently for different frequency bands such that:

$$\begin{array}{cc} j = -2, +2 & \text{for} \quad 2 < k < 63 \\ j = -3, -2, +2, +3 & \text{for} \quad 63 <= k < 127 \\ j = -6, ..., -2, +2, ..., +6 & \text{for} \quad 127 <= k <= 250 \end{array}$$

The remaining spectral lines are identified as non-tonal components.

Less powerful maskers are determined and eliminated in order to obtain a global masked threshold. Two conditions are used for this purpose: 1) tonal or non-tonal component which generates masking thresholds that are below the absolute threshold of hearing are eliminated as these components will not be audible [5], and 2) less powerful tonal components within the distance of less than 0.5 Bark from a powerful tonal masker are eliminated. This way, components with the highest power are kept in the list of tonal components and others are eliminated.

The global masking threshold for each frequency index is derived from the individual masking thresholds of calculated tonal and non-tonal maskers as well as hearing threshold in quiet.

After every frame has been processed, masking thresholds of the 32 subbands are stored in a binary file. For a three second audio signal that has a sampling rate of 44.1 kHz, output will have $\lfloor (44100 * 3)/384 \rfloor = 344$ frames because of overlapping windowing. This results in a file size of 86 KB for the storage of this auxiliary information. This is approximately one fourth of the original sound file. With contemporary hardware specifications, storing this additional information on volatile memory would not cause any issues.

Offline analysis is concluded by storing the masking thresholds for each frame. Global masking threshold of the entire scene will be calculated during the real time analysis.

2.2. Infeasibility of Precalculated Decision Making

One could argue that precalculating and storing global masking threshold of a *dynamic* scene is also a feasible option. For a given scene, masking can only occur after at least two concurrent sound sources are played, which is what is of interest. While at least two sources are necessary for masking to occur, there is no theoretical upper limit on the number of sound sources that can be played at a given time. Even if such a limit is imposed on the number of distinct sound files/sources in order to limit the number of different combinations, since the same sound file can be played infinitely many times, infinitely many possible combinations exist for a given scene. Also, each sound event would have a different onset making precalculation infeasible.

Let there be a scene with 30 sound files that are 4 seconds long each. There are $\lfloor (44100/384) * 4 \rfloor = 459$ frames per sound file. If there are only two unique sound files played at a time, there would be $\binom{30}{2} * (459 + 459 - 1) = 398\,895$ possible combinations just for two overlapping sound sources. In this calculation, onsets of each sound event are constrained to align with the starting pointer of a frame. If there are only three unique sound files played at a time, there would be $\binom{30}{3} * (459 + 459 - 1)^2 = 6\,828\,018\,680$ combinations. This analysis excludes the possibility that same sound source (e.g. a gunshot sound) can be activated more than once in the given duration.

The possible combinations are increasing exponentially and the memory required to store all number of concurrent sound sources becomes infeasible with the contemporary hardware.

2.3. Real-Time Analysis

After the offline analysis is performed on the audio sources contained within the scene, and the masking data stored in memory, real-time analysis can be performed during the program execution. The main purpose of the real-time analysis, is to calculate

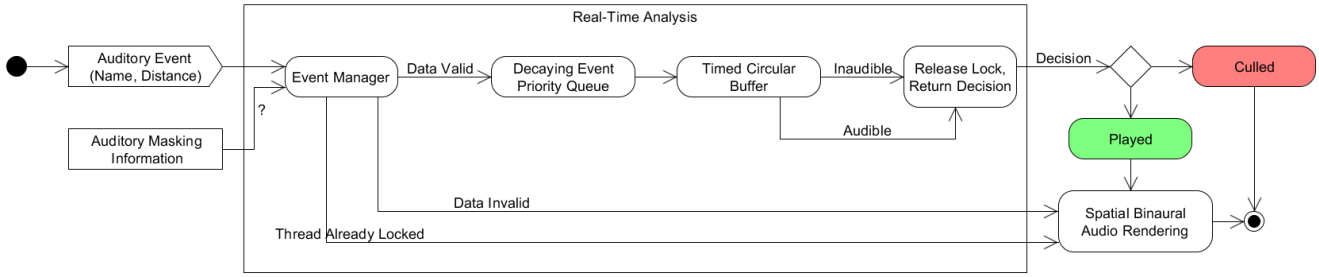


Figure 2: *Overall pipeline for the real time analysis*

the global masking threshold for the dynamic scene on the fly and identify whether a new sound event will be audible or not given the existing events already being rendered. Determining this allows culling inaudible sound sources, which in turn reduces the computational overhead due to rendering spatial audio. At the offline stage simultaneous masking thresholds are calculated for monaural listening conditions. However we assume that the global masking threshold, which is an attribute of monaural hearing is also an indicator of the perceptual salience for binaural hearing. Visual representation of the real time analysis, for handling an auditory event is shown in the Fig. 2.

Real-time analysis involves several stages: Firstly, when an auditory event is triggered, the event manager (EM) does the preliminary operations required to handle this event. EM then places or re-prioritizes this event in the decaying event priority queue (DEPQ) in order to reduce hard drive access delay. After the data is successfully stored in the heap, timed circular buffer (TCB) attenuates masking thresholds according to their distance and adds their masking values to the global masking threshold. Then, the *audibility ratio* of the individual audio source is calculated. If this ratio is larger than a selected threshold value, that audio source is culled.

2.3.1. Concept of Sound Events and Event Manager

For the proposed algorithm, the audio rendering component of the game engine needs to be encapsulated. For every spatial audio render request, sound events are generated, later to be handled by the sound event manager (EM). Each audio event includes:

- An **audio identifier** which denotes both the actual audio file path and the binary file which contains the masking information.
- **Distance** from the listener position

The event manager that was mentioned above has two responsibilities: 1) handling sound event requests, and 2) handling *end-of-audio events* received from the game engine. Event manager runs on its own thread with a single `mutex` that locks each time an event is received. Receiving end-of-audio events require the same `mutex` lock acquisition as receiving an auditory event. Hence they are synchronized, even if multiple threads call them simultaneously. One key difference between them is that, receiving an end-of-audio event is a blocking operation whereas sound event handling is not. Details of each operation are explained below.

Sound Event Handling: Whenever a sound event is triggered, the event manager tries to acquire the thread lock (via the only `mutex` it has). If it fails to acquire the lock, it simply returns without doing any operations and the audio will be rendered. If it acquires the lock, meaning that this is the only running real time analysis, it checks whether the masking information for the given audio signal exists or not. If there is no masking information present, audio will be automatically rendered, without doing any further operations. This could be the case, for example, for background music which should not be subject to culling or for sources which are marked as immutable by the game audio designer.

If the lock is acquired and the audio/masking data are valid, event manager proceeds with the remaining operations. After all the operations and calculations are carried out and the decision is made, the event manager releases the thread lock and returns the result to the calling thread. After that, if the audio is marked as *inaudible*, it will be culled and no further operations will be necessary. If the audio is marked as *audible*, game engine proceeds with spatial binaural audio rendering. A visual representation of the workflow is displayed in Fig. 3.

End-Of-Audio Event Handling: End-of-audio events are signals that indicate that a single audio file has finished playing, and the information of which audio file is finished is not conveyed. These events are required in the execution of the circular masking threshold which will be explained below.

When an *end-of-audio* event is received, lock must be acquired. After the program counter is in the critical section, event manager decrements the active audio count. If there is no active sound in the scene, timed circular buffer is notified and the lock is released. When all sound events have finished playing, TCB is purged. A visual representation of the workflow is displayed in Fig. 4.

2.3.2. Decaying Event Priority Queue

As explained above, auditory masking information for each sound is stored in persistent data storage such as HDDs or SSDs. Compared to the data that was allocated dynamically from heap or stack, accessing data is significantly slower from these resources. In order to counteract this effect, the retrieved auditory masking information is temporarily stored in a decaying event priority queue (DEPQ).

DEPQ is a dynamically allocated priority queue that stores a fixed amount of past auditory event information. Priorities for each event are set according to their arrival order. As the name suggests, priorities for each event are decremented every time a new event comes. When the queue is full and an event that is not stored in queue arrives, the event with the lowest priority (the least active event) is replaced with the newer one.

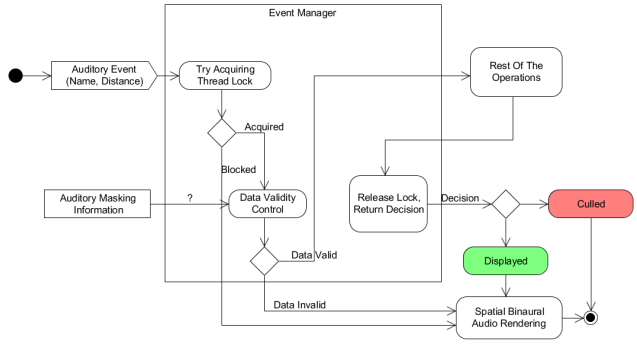


Figure 3: *Event manager workflow for handling auditory events*

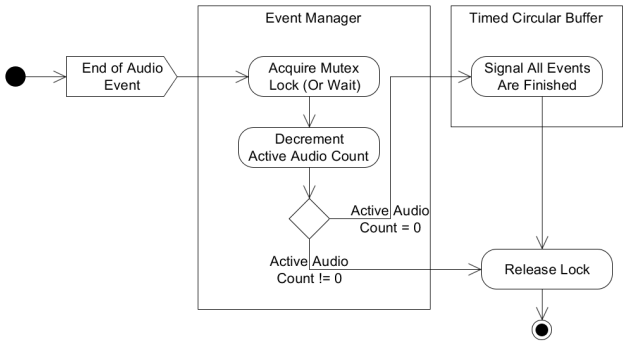


Figure 4: *Event manager workflow for handling end-of-audio events*

2.3.3. Timed Circular Buffer

Timed circular buffer (TCB) contains the global masking threshold and incorporates new masking values into it. All of the masking values come from binary files that are stored during the initial offline analysis stage. TCB also decides whether a given sound will be audible or not by calculating audibility level as a ratio of the number of audible frames to the number of inaudible frames of the audio signal to be played. TCB then returns its assessment to the event manager. The timed circular buffer is a variation of circular buffer with four additional variables besides the size of the buffer and the buffer itself:

1. *Start Index, inclusive*; Points to the first valid data of the buffer. Updated according to time, stored as unsigned integer.
2. *End Index, exclusive*; It points to the first empty index of the buffer. Stored as unsigned integer.
3. *Valid Index, exclusive*; This points to the end of previous sound data and index from which the array needs to be cleared from. One could also say it is this first index after the last valid data. Stored as unsigned integer.
4. *Audibility Percentage*; For a given audio and an existing scene, determines how much of the given auditory event will be audible.

Details of how each index works in conjunction with the algorithm is explained below:

MPEG-1 Layer I uses 12 samples per subband and there are 32 subbands as explained above. Size of the buffer, in the case that the buffer will store 5 s[1] of masking information, can be calculated as follows:

```
bufferSize = Size of buffer for 5 seconds
                        + 1 extra empty frame
           = floor(5000 / Frame Time) * 32 + (32)
           = floor(574.21875) * 32 + 32
           = 18400 Doubles
```

[1]Size of buffer is not constrained to 5 seconds and can be shorter or longer depending on user requirements.

where the duration of a single frame of audio is given as:

```
Frame Time = (Frame Size * 1000 ms) / Sampling Rate
                  = (384 * 1000 ms) / 44100 ≈ 8.7 ms
```

The start index is only modified by the actual time difference. It is updated every time a new auditory event arrives. Calculation of start index, when requested, is implemented as follows:

```
startIndex += ⌊ elapsedTime / frameDuration ⌋ / 32;
            startIndex = startIndex % circBfrSize;
```

Prior to calculating the global masking values for the dynamic scene, the individual masking values are attenuated according to distance using the inverse-square law.

$$\Delta L_p = 10 \log(\frac{r_1}{r_2})^2 \text{ dB}$$

While determining the audibility for a given audio event, a threshold value called *audibility ratio* is defined:

$$\textit{Audibility ratio} = \frac{\textit{number of audible frames}}{\textit{total number of frames}}$$

To determine whether a frame is audible or not, masking values are first compared with the current global masking values. If the given masking value is greater than the existing global masking threshold for any of the 32 subbands, that frame of the audio will be considered audible.

When handling masking values, TCB follows the following steps of execution:

1. Update starting frame according to time,
2. Determine estimated end index and clear deprecated data,
3. Attenuate masking values according to source distance,
4. Calculate the audibility ratio within the existing scene,
5. If audibility ratio is greater than a preset threshold, add these masking values to TCB and return.

3. PERFORMANCE ANALYSIS

The primary purpose of the proposed algorithm is to provide a performance advantage with minimum perceptual degradation. As such, the computational cost of real time analysis should be lower than synthesizing binaural audio. Performance of the proposed algorithm, as well as factors that affect analysis duration is explained below.

3.1. Algorithmic Complexity of Real-Time Analysis

Real time analysis consists of three components: event manager (EM), decaying event priority queue (DEPQ), and timed circular buffer (TCB). All operations of the event manager have a computational complexity of $O(1)$, and the algorithmic complexity depends on the other two components, DEPQ and TCB. In this section, we will assume that the size of the priority queue is m and the size of the masking value sequence is assumed to be n.

Decaying event priority queue (DEPQ) has two main functionalities; handling sound events and handling end-of-audio events. End of audio events are of complexity $O(1)$. Handling audio events on the other hand, not only requires interaction with the queue itself but also the auditory masking values in the file system. Firstly, if the size of the queue is m, searching through the queue has a complexity of $O(m)$. Secondly, fetching the masking values from the file system and placing them into the queue has a complexity of $O(n)$.

For the timed circular buffer (TCB), the sequence of input values is traversed twice. Once for attenuating the masking values according to distance and second time for adding the masking values to the global masking threshold. So the complexity of this operation is $O(n)$.

Three factors determine the computational cost of the analysis stage:

1. Length of the sound signal:
   (a) In the case that the file is not already stored in DEPQ, read time (excluding the seek time) is proportionate to the length of the offline analysis file.
   (b) Each frame coming from the analysis file are compared to the global masking threshold in the TCB to determine whether a given audio is audible or not. Hence number of frames affects the duration of the real-time analysis.
2. Type of persistent data storage: Retrieval from a physical drive takes longer in comparison with a solid-state drive.
3. Storage in DEPQ: If data is not stored in the program stack, time costs of seek time, data transfer rate, and rotational latency (for HDDs) are added to the time cost of finalising the analysis. [8]

A limiting factor in the applicability of the proposed method is the type of persistent data storage from which offline analysis data is retrieved. HDDs incur spin-up delays, seek time delays and slow data transfer rates compared to SSDs. In a system that highly depends on time, such issues may render the system useless due to hardware delays. If the end user were to use HDDs, increasing DEPQ size and storing the whole masking information into the queue would be a feasible option.

3.2. System Performance

An assessment of the performance of the proposed method is presented in this section. Time values listed here are hardware dependent. In order to smooth out peaks due to higher-priority operating system processes, the metrics presented in this section use an average of 10 runs.

The hardware used in the calculation of the reported values includes Intel $i5$-3570K CPU running at 3.40GHz, 8GBs of RAM, 120GBs of SSD, 500GBs of HDD and a GeForce GTX 670 graphics card.

3.2.1. Performance of Real Time Analysis

The time it takes to complete real time analysis is affected by how the pre-calculated data is accessed. Different storage media have different data transfer overheads which in turn affects the duration of real time analysis. This section describes the individual cost of the culling algorithm, including any delays associated with it but excluding the cost of the subsequent (binaural) rendering operations. Table 1 shows the average times to reach a decision, based on the conditions listed.

Different hardware architectures would have different results.

Table 1: *Average time required to complete real time analysis*

Input Size	Storage in DEPQ	Memory Type	Delta Time
1 Frame	Stored	RAM	≈ 0 ms
574 Frames	Stored	RAM	0.51 ms
1 Frame	Absent	HDD	15.626 ms
574 Frames	Absent	HDD	15.918 ms
1 Frame	Absent	SSD	≈ 0 ms
574 Frames	Absent	SSD	0.53 ms

3.2.2. Performance Gained by Culling a Single Audio Source

For a sound signal that was stored in DEPQ, average time it takes to render a 5 second audio is 0.51 milliseconds. During this time, a single CPU core is fully utilized. We can express the total cost of a program execution as:

$$CostOfExecution = Average\,CPU\,Utilization * Time\,In\,Milliseconds$$

Since audio rendering incurs a performance penalty in the beginning of each frame, over the time of execution, we can see whether this methodology is profitable or not. Since we are not considering the memory delay for retrieving the actual audio data from memory, we are not considering the retrieval of the masking values from memory either. For an audio signal that is already inside the DEPQ, and for audio data that was already stored in RAM, the performance metrics are listed in Table 2.

Table 2: *Execution cost comparison*

Input Size	Culling Assessment	Cost Of Execution
1 Frame	No Culling Algorithm	$9\text{ms} * \%0.44 = 3.96$
1 Frame	With Culling Algorithm	$9\text{ms} * \%0.46 = 4.14$
574 Frames	No Culling Algorithm	$5000\text{ms} * \%0.45 = 2250$
574 Frames	With Culling Algorithm	$5000\text{ms} * \%0.49 = 2450$

As shown in the table, the culling algorithm has 8% more performance overhead compared to the existing audio synthesis pipeline. However in the case that the audio will be culled, we will save 92% of the clock cycles. In other words, in order for the proposed algorithm to provide any computational gain, it should cull at least 92% of all the frames. One disadvantage of the culling algorithm, however, is that it will block a CPU core completely until the analysis is done.

At the cost of degrading the overall composition of the scene, a volumetric culling methodology would save %100 percent of the clock cycles instead of our %92 but does not guarantee the retention of perceptually most prominent sources. While the proposed methodology does not provide the best overall performance, it provides a good balance between perceived richness and performance as will be shown in the next section.

4. SUBJECTIVE EVALUATION

Audio produced with sound source culling, regardless of the methodology, is dependent on the user determined variables. Resulting auditory richness produced by the aforementioned methodologies, provide a crude approximation to the original scene. In order to find out which parameters provide sufficient perceived richness, a subjective evaluation was performed.

4.1. Chosen Test Methodology

The ITU-R Recommendation BS.1534, proposes proposes a subjective evaluation method called "MUlti Stimulus test with Hidden Reference and Anchor
(MUSHRA)" which is appropriate for assessing intermediate audio quality [9]. While this paper is not directly related to audio coding or quality, we use MUSHRA to test intermediate levels of auditory richness given a hidden reference and anchor.

4.2. Test Procedure

4.2.1. Presentation of Stimuli

In the MUSHRA test method, a high quality reference signal, a low quality anchor signal and other signals that fall in between them in terms of quality are evaluated [9]. In any given test, there is a single reference and a single anchor that the user is expected to find. Each prerecorded sound can be played as many times as the listener desires. Presentation of both the different experiments and the stimuli contained within the experiments are randomized. So not only each listener listens experiments in a different order but each listener faces a randomized presentation order of the stimuli. For the experiments, a MATLAB interface called MUSHRAM was used [10].

4.2.2. Grading

The whole test procedure is comprised of giving ratings to test signals which are displayed in random sequence. Listeners were instructed to score the presented samples in comparison with the explicitly provided reference signal. The scores that are given, can range between 0 and 100. Listeners were asked to find and rate the reference and the anchor signals as 100 and 0, respectively. For any of the remaining stimuli, listeners were asked to give their scores according to the perceived richness. They were also encouraged to listen to the available reference signal, prior to giving scores on the stimuli.

Table 3: *Contents of Test Scenes*

Experiment	Contents (ss = sound sources)
Impulsive Scene	13 impulsive ss.
Impulsive + Speech + Music	6 impulsive ss., 6 speech ss., 1 Music sound
Impulsive + Sound Effects + Music	6 sound effect ss., 6 impulsive ss., 1 Music sound
Sound Effects + Speech	7 speech ss., 6 sound effect

4.3. Experiment Details

The MUSHRA test applied for this paper is comprised of four different experiments. Each experiment involves a combination of different sound sources of different categories: impulsive sounds, music, static sound effects and speech signals. From these sound sources, we arranged four different psychoacoustical experiments, involving sounds under different categories. In each of these scenes, there were a total of 13 sound sources in the case of reference signals. The applied culling methodology served to reduce the number of sound sources at each rendered scene. Composition of each scene is given in the Table 3.

The test scenes were adjusted so that the maximum length of a stimulus does not exceed 2 s.

4.3.1. Selection of Sound Source Locations

Since both systems need to be tested with the same conditions, a scene that is appropriate to both culling methodologies was required. In order to achieve this, sound sources were distributed along the horizontal plane of the listener. Assuming one source will be directly in front of the listener, the axis needed to be divided into $180/(13-1) = 15$ degree segments. Listener was placed at the coordinate location $(700, 700)$, facing towards the negative x axis.

We used Unreal Engine 4 to test the proposed algorithm. In order to achieve a controllable volumetric culling methodology, each of the sound sources were placed at a fixed radius away from the listener. They were grouped into pairs and were placed at an equal distance and symmetrical angles with respect to the front direction of the listener. Coordinates of sound sources were calculated according to the formula:

$$X = r\cos(\theta) + 700\,(\text{cm})$$

$$Y = r\sin(\theta) + 700\,(\text{cm})$$

Top down view that demonstrates the positions of the sound sources is given in Fig. 5.

For example, a sound source with a culling radius of 200 cm will not be rendered when a listener is positioned 300 cm away from the source. This way of adjusting test scene enables precise control over which sound sources are going to be culled because of the spherical culling radii. Note that these sources were not triggered in the same time frame so that their onsets do not coincide.

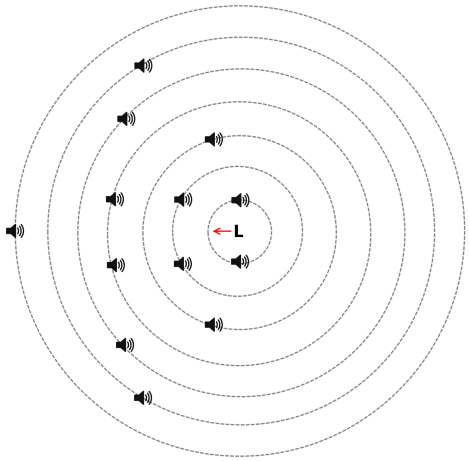


Figure 5: Positioning of the sound sources for the tested scenes

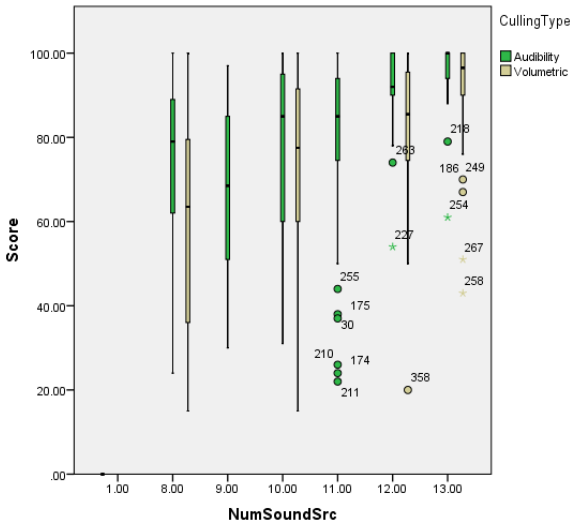


Figure 6: *Box plot of subjective scores from listening tests. Circles denote outliers and stars denote extreme values with their corresponding sample index*

4.3.2. Generation of Test Cases

There are nine stimuli in each experiment. The reference signal includes all 13 sounds. The anchor signal involves only a single sound source to achieve the lowest richness in a scene. The remaining signals are results of scenes with culling methodologies applied. For the perceptual culling methodology, various audibility ratios were tested. For volumetric culling methodology, various radii were tested. In total, there were 4 scenes with 9 culling settings making a total of 36 test cases.

At each run, among the 7 stimuli, 4 of them were produced with perceptual culling and 3 of them were produced with volumetric culling. For perceptual culling, %0, %10, %20 and %30 audibility ratios were used. For distance based culling, 650, 550 and 450 cm culling radii were used. Due to the configuration of the scene, chosen auditory stimuli and culling methodology, test cases that have 9 and 11 sound sources were only available to audibility based culling methodology.

4.4. Statistical Analysis

Eleven participants participated in the test (7 male, 4 female). Fig. 6 shows the results of the scores from the test. As can be observed, scenes that have 8, 10, 12 and 13 sources have results for both of the culling methodologies.

In order to see whether the proposed methodology has any advantage over volumetric culling for a given scene, responses given to scenes with the same number of sources but obtained using different culling strategies can be compared. Fig. 6

Independent-samples t-tests were performed to find out whether mean responses given to stimuli obtained using different culling methods are significantly different. Differences in mean values of subjective responses for 8, 10, and 12 sound source scenes using different culling methodologies is statistically significant at $\alpha = 0.05$ level with auditory culling outperforming volumetric culling for these cases. For the 8 source case, the difference between mean responses was statistically significant with $t(75) = 2.965$, $p = 0.004$ (Variances between groups is significant). For the 10 source case, the difference between mean responses was statistically significant with $t(152) = 2.767$, $p = 0.006$ (Variances between groups is significant). For the 12 source case, the difference between mean responses was statistically significant with $t(59.268) = 2.270$, $p = 0.027$ (Variances between groups is not significant)[2]. For the 13 sources case, the differences between mean responses were not significant. This result was expected as no source was culled through culling methodologies for this case.

5. CONCLUSIONS

A perceptual sound source culling methodology based on models of simultaneous masking was proposed in this paper. The algorithm uses a prioritisation approach based on the audibility of sound sources and those sources which will be less audible than the others are culled.

The algorithm was evaluated with respect to its computational as well as its perceptual performance. Performance evaluations showed that while the culling algorithm itself causes some computational overhead, it may still provide savings when it is realised that culled sources are not processed further for binaural spatialisation and that binaural processing has a higher computational overhead in general. The subjective evaluations involved a comparison of the proposed culling algorithm with volumetric culling where sources are culled based on their distance from the listener. It was observed that for the same number of sources culled using the auditory culling and volumetric culling methods, auditory culling provided higher scores in terms of auditory richness.

While the computational and perceptual performance of the proposed method for culling sources in more complex scenes including a higher number of sources remains to be investigated, the proposed culling algorithm provides a promising approach that

[2] In order to check for the normality assumption, Levene's test was used. The degrees of freedom of the test is modified in order to account for the cases where the assumption that homogeneity of variances does not hold and is called Welch's t-test.

could make it possible to render better spatial audio on low-end devices.

6. REFERENCES

[1] Richard Stevens and Dave Raybould, *Game audio implementation: A Practical Guide Using the Unreal Engine*, Focal Press, 2015.

[2] David Grelaud, Nicolas Bonneel, Michael Wimmer, Manuel Asselot, and George Drettakis, "Efficient and practical audio-visual rendering for games using crossmodal perception," in *Proceedings of the 2009 symposium on Interactive 3D graphics and games*. ACM, 2009, pp. 177–182.

[3] Nicolas Tsingos, Emmanuel Gallo, and George Drettakis, "Perceptual audio rendering of complex virtual environments," in *ACM Transactions on Graphics (TOG)*. ACM, 2004, vol. 23, pp. 249–258.

[4] S Spencer Hooks and Nicolas R Tsingos, "Encoding and rendering of object based audio indicative of game audio content," Aug. 6 2013, US Patent App. 14/414,877.

[5] ISO / IEC, *Coding Of Moving Pictures And Associated Audio For Digital Storage Media At Up To About 1.5 Mbit/S Part 3 Audio*, June 1993.

[6] Mathieu Lagrange and Sylvain Marchand, "Real-time additive synthesis of sound by taking advantage of psychoacoustics," in *Proceedings of the Digital Audio Effects (DAFx01) Conference*, 2001, pp. 249–258.

[7] Davis Pan, "A tutorial on mpeg/audio compression," *IEEE multimedia*, , no. 2, pp. 60–74, 1995.

[8] Hamid D Taghirad and Ehsan Jamei, "Robust performance verification of adaptive robust controller for hard disk drives," *Industrial Electronics, IEEE Transactions on*, vol. 55, no. 1, pp. 448–456, 2008.

[9] ITU-R, *Method for the Subjective Assessment of Intermediate Quality Level of Audio Systems*, bs.1534-3 edition, 10 2015.

[10] E Vincent, "Mushram: A matlab interface for mushra listening tests," *Online] http://www. elec. qmul. ac. uk/people/emmanuelv/mushram*, 2005.

AUTOMATIC VIOLIN SYNTHESIS USING EXPRESSIVE MUSICAL TERM FEATURES

Chih-Hong Yang, Pei-Ching Li, Alvin W. Y. Su

Scream Lab., Department of CSIE,
National Cheng-Kung University,
Tainan Taiwan
P76034193@mail.ncku.edu.tw,
P78021015@mail.ncku.edu.tw,
alvinsu@mail.ncku.edu.tw

Li Su, Yi-Hsuan Yang

MAC Lab., Research Center for Information Technology
Innovation, Academia Sinica,
Taipei Taiwan
lisu@citi.sinica.edu.tw
yang@citi.sinica.edu.tw

ABSTRACT

The control of interpretational properties such as duration, vibrato, and dynamics is important in music performance. Musicians continuously manipulate such properties to achieve different expressive intentions. This paper presents a synthesis system that automatically converts a mechanical, deadpan interpretation to distinct expressions by controlling these expressive factors. Extending from a prior work on expressive musical term analysis, we derive a subset of essential features as the control parameters, such as the relative time position of the energy peak in a note and the mean temporal length of the notes. An algorithm is proposed to manipulate the energy contour (i.e. for dynamics) of a note. The intended expressions of the synthesized sounds are evaluated in terms of the ability of the machine model developed in the prior work. Ten musical expressions such as *Risoluto* and *Maestoso* are considered, and the evaluation is done using held-out music pieces. Our evaluations show that it is easier for the machine to recognize the expressions of the synthetic version, comparing to those of the real recordings of an amateur student. While a listening test is under construction as a next step for further performance validation, this work represents to our best knowledge a first attempt to build and quantitatively evaluate a system for EMT analysis/synthesis.

1. INTRODUCTION

Expression plays an important role in music performance. For the same musical score, different performers would interpret the score with their personal understandings and experiences and instill their feelings and emotions into it, thereby creating large variations in their actual performances. These variations can be observed in interpretational properties like timing, modulation, and amplitude. Therefore, in automatic music synthesis, an important step is to characterize and to control such expressive parameters.

Expressive music performance has been studied in the last few decades [1, 2, 3, 4, 5, 6]. For example, Bresin *et al.* [7] synthesized music of six different emotions by using performance rules such as duration contrast, punctuation, and phrase arch. Maestre *et al.* [8] characterized dynamics and articulation parameters related to the expressivity of saxophone. D'Incà *et al.* [9] considered four sensorial adjectives (*hard*, *soft*, *heavy*, and *light*) and four affective adjectives (*happy*, *sad*, *angry*, and *calm*) based on a set of audio cues. Grachten *et al.* [10] used both predictive and explanatory framework to model three categories of dynamics markings in piano. Erkut *et al.* [11] captured information of guitar performance such as damping regimes and different pluck styles used in a plucked-string synthesis model. More recently, Perez *et al.* [12] combined the modeling of the characteristics of the performer (i.e., pitch, tempo, timbre, and energy), the sound as well as the instrument in order to render natural performances from a musical score.

Surprisingly, among all the elements of expressive synthesis, the *expressive musical terms* (EMT) that describe feelings, emotions, or metaphors in a piece of music have been rarely discussed, even though they have been widely used in Western classical music for hundreds of years. To fill this gap, in a prior work [13] we presented a computational analysis of ten EMTs — including *Tranquillo* (calm), *Grazioso* (graceful), *Scherzando* (playful), *Risoluto* (rigid), *Maestoso* (majestic), *Affettuoso* (affectionate), *Espressivo* (expressive), *Agitato* (agitated), *Con Brio* (bright), and *Cantabile* (like singing) — using a new violin solo dataset called SCREAM-MAC-EMT. [1] The dataset contains ten classical music pieces, with each piece being interpreted in six versions (five EMTs and one mechanical, deadpan version denoted as *None*) by eleven professional violinists, totaling 660 excerpts [13]. With this dataset, we built supervised machine learning models to recognize the EMT of a music excerpt from audio features. We compared the performance of two types of features for the classification task. The first type of features includes a set of audio features characterizing the three interpretational factors, dynamics, duration, and vibrato, whereas the second type of features are standard timbre, rhythm, tonal, and dynamics features such as Mel-frequency cepstral coefficients (MFCC) extracted from the MIRtoolbox [14]. Our evaluation shows that the first feature set, which has clearer music meanings, achieves better classification accuracy than the standard features do, showing the importance of these interpretational features in characterizing EMTs.

Extending from this prior work, we investigate in this paper the use of a small set of such interpretational features for synthesizing music of different EMTs. Specifically, we aim to manipulate features of vibrato, dynamics, and duration to synthesize expressive sounds from a mechanical interpretation. The way of manipulation is learned from a training set SCREAM-MAC-EMT. To quantitatively evaluate the performance of the proposed expressive synthesis method, we make use of the classification model developed in our prior work [13] again to see if the intended EMT of the synthesizer can be correctly recognized. Specifically, we recruit a professional violinist and an amateur student to record new data in accordance with the collection method of the SCREAM-MAC-EMT dataset. That is, both of them perform the sixty classical excerpts (i.e. six different versions of the ten pieces) individually. Then, we compare the performance of the real and synthetic versions by means of the same EMT analysis and classification procedure. In other words, the objective evaluation of the ten EMTs

[1] https://sites.google.com/site/pclipatty/scream-mac-emt-dataset

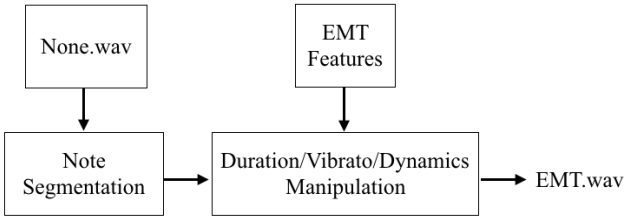


Figure 1: *Flowchart of the EMT synthesis system.*

recognition is applied to these outside data through the classification models constructed from the preliminary work [13]. Therefore, we can observe not only the differences between the two people who have distinct musical trainings and skills from their violin performances, but also the result of synthesis based on the two unique sources. The synthesized sound samples can be found online. [2] This paper is organized as follows. Section 2 describes the expressive synthesis method, together with the EMT features, and the setting of classification. In Section 3, the experimental results are presented. Finally, we conclude in Section 4.

2. METHOD

Figure 1 shows the EMT synthesis diagram, whose goal is to convert a mechanical interpretation of a music piece into an expressive one. We refer to the mechanical interpretation as the "*None*" signal. As the manipulation process is usually done for each note, at the first stage note segmentation is applied to the *None* audio file. Then the manipulations of duration, vibrato, and dynamics of each segmented notes are performed according to the pre-learned parameters of the target EMT (see Sections 2.1–2.3 for details). Lastly, when concatenating all the manipulated notes back into a complete music piece, we adopt fade-in and fade-out operations to eliminate the crackled sounds.

To synthesize expressive sounds, the parameter values of the ten EMTs are calculated by averaging over the corresponding music pieces because each EMT is interpreted in five different excerpts. Moreover, as we have the performance from eleven musicians for each music piece and each EMT, the parameters will be averaged again across the violinists. The *EMT feature set* listed in the Table 1 is used in the proposed synthesis system. It includes seven relevant features, namely `vibRatio`, `ND-`C_M, `4MD-`C_M, `FPD-`C_M, `D-M-`C_M, `D-Max-`C_M, and `D-maxPos-M`, as well as two fundamental features of vibrato, `VR-M-M` and `VE-M-M`. The first seven features are found to be more important than other possible interpretational features for classifying EMTs [13]. The last two features are found less useful in classifying EMTs, but they are still needed to manipulate added vibrato to a note.

The manipulations of duration and vibrato are implemented by means of the phase vocoder, which is a mature technique of time stretching and pitch shifting [15, 16, 17]. Given an audio input, the short-time Fourier transform (STFT) converts the signal from time domain into a time-frequency representation. The *time stretching* (expansion/compression) is achieved by modifying the hop size and then by performing the inverse STFT with the overlap-add method. The *pitch shifting* is accomplished by resampling the time stretched signal back to the original length. More details of the synthesis method, together with the meaning of the features listed in the EMT feature set, and the setting of EMT classification, are introduced in the following sections.

In what follows, we assume that all the audio excerpts are sampled at 44.1 kHz.

[2] `http://screamlab-ncku-2008.blogspot.tw/2016/03/music-files-of-expressive-musical-term-experiment.html`

Table 1: *The EMT feature set used in the synthesis system. The terms 'M' and '*C_M*' denote mean and contrast of mean, respectively. Please refer to [13] for details.*

Features	Abbreviation	Description
Vibrato	`vibRatio`	percentage of vibrato notes in a music piece
	`VR-M-M`	mean vibrato rate
	`VE-M-M`	mean vibrato extent
Dynamics	`D-M-`C_M	mean energy
	`D-Max-`C_M	maximal energy
	`D-maxPos-M`	relative time position of the energy peak in a note
Duration	`ND-`C_M	mean length of every single note
	`4MD-`C_M	mean length of a four-measure segment
	`FPD-`C_M	mean length of a full music piece

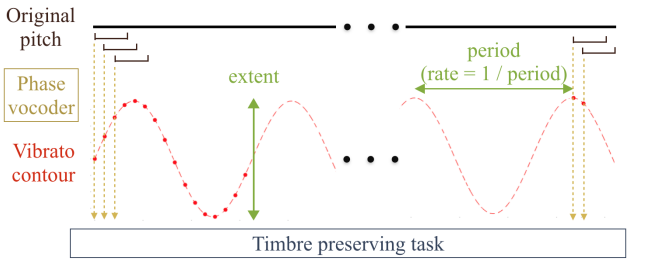


Figure 2: *Illustration of adding vibrato to a non-expressive note. The note is divided into a sequence of fragments whose pitches will be individually shifted by means of the phase vocoder. The timbre preservation is applied to each fragment. The vibrato contour is sampled at sixteen times per cycle to avoid artifacts.*

2.1. Vibrato Features

Vibrato is an essential factor in violin performance and its analysis/synthesis has been studied for decades [18, 19]. Vibrato is defined as a frequency modulation of F0 (fundamental frequency) and is typically characterized by the rate and extent [20]. The vibrato *rate* means the number of periodic oscillations per second while the vibrato *extent* specifies the amount of frequency deviation. In the EMT feature set, `VR-M-M`, `VE-M-M` and `vibRatio` are related to vibrato. The first two are defined as the mean value of the vibrato rate and extent, and the last one means the ratio of the number of vibrato notes over total notes in a music piece. The detailed criteria of determining whether a note is vibrato could be found in [13]. Vibrato is a common interpretation in violin performance, but the `VR-M-M` and `VE-M-M` are found to have weak discrimination power in classifying EMTs [13], possibly due to their subtle difference. The mean values of the two features among the

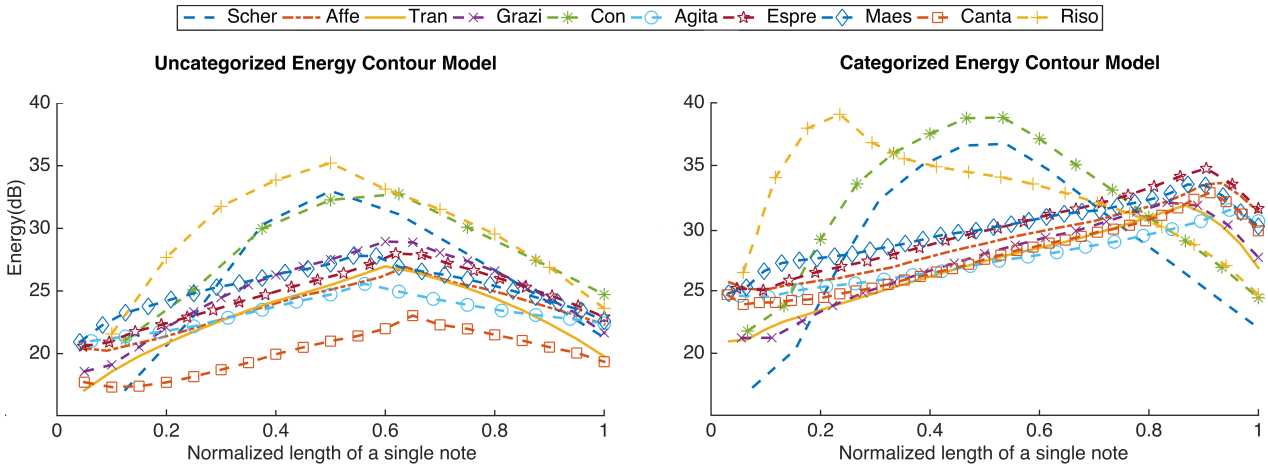


Figure 3: *The energy contours are modeled by means of the EMT parameter (UEC; left) and the categorized method (CEC; right).*

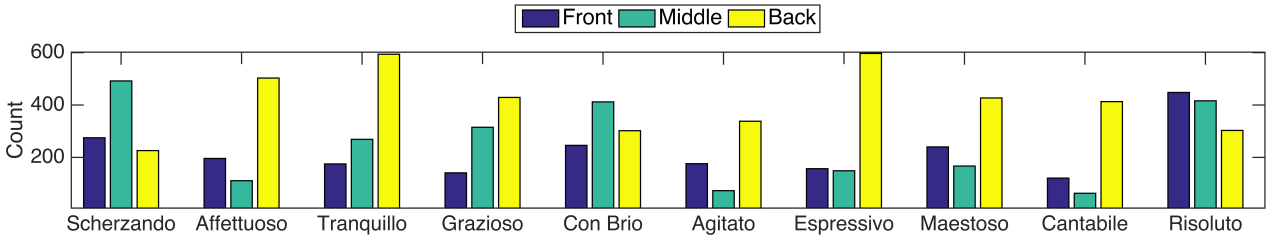


Figure 4: *The number of notes in the three categories of the ten EMTs*

ten expressive musical terms are between 6.3–6.8 Hz (STD=0.15) and 0.27–0.38 semitones (STD=0.03) separately. In contrast, the `vibRatio` has strong discrimination power and its mean values are between 52–74% (STD=6.82) among the ten expressions. Assuming that there are no vibrato at all in the *None* signal, to implement `vibRatio` we need to determine how many vibrato notes there should be, and which notes should be manipulated to have vibrato. Firstly, the amount of vibrato notes is easy to calculate and is expressed by:

$$\#\,\text{Vibrato notes} = \#\,\text{notes in a violin piece} \times \text{vibRatio}\,, \tag{1}$$

where the value of vibRatio is set differently for different EMTs and it's set according to its average value in the training set (i.e. SCREAM-MAC-EMT) per EMT. Secondly, according to our observation, a note with longer duration will more likely have vibrato. Hence, we sort all the notes in descending order of duration and add vibrato to the top longest ones (the exact number of notes is determined by equation (1)).

Moreover, we remark that the continuity of the pitch contour is important to obtain a naturally synthesized vibrato. Therefore, we use a sequence of short fragments to model the modulation of the original frequency of a non-expressive note. Specifically, we shift the pitch of each fragment through the phase vocoder to fit a vibrato contour. For the purpose of avoiding weird artifacts, we sample at sixteen times per cycle of a vibrato contour so that the sampling period is approximately 2.4 milliseconds (410 samples). Accordingly, the first step of the vibrato manipulation process shown in Figure 2 is that partitioning a non-expressive note into a sequence of fragments of 2,048 samples with an 80% overlap (1,638 samples). Next, given a fragment and its corresponding pitch on the particular vibrato contour generated by the `VR-M-M` and `VE-M-M`, the pitch shifting is carried out with a Hanning window of 256 samples as well as a hop size of 64 samples. Then, the timbre preserving method proposed by Röbel and Rodet [21] is adopted. According to this method, both the spectral envelope, measured by a true envelope estimator, and the pre-warping factor of the original fragment are calculated before the pitch shifting takes place. The timbre preservation is therefore realized by means of the multiplication of the pre-warping factor and the pitch-shifted fragment. Finally, we overlap and add the fragments to achieve the synthesized vibrato note.

2.2. Dynamic Features

One of the most prominent characteristics to distinguish expression is dynamics. According to Gabrielsson and Juslin [22], the dynamics and the temporal envelopes of individual notes are different for distinct expressions. The EMT feature set has three dynamic features, `D-M-C`$_M$, `D-Max-C`$_M$ and `D-maxPos-M`, which indicate the mean energy, the maximal energy, and the relative time position of the maximal energy peak in a note (denoted as *maxPos*), separately. To utilize these features for synthesis, we need to know the dynamic envelops of the ten EMTs ahead. However, the specific envelopes are still unknown within these features so we need to model the energy contour, which characterizes the instantaneous energy as a function of time. To make the energy contour as close to the real acoustic envelope as possible, we consider the data of the three consultants, who helped us in the cre-

ation of SCREAM-MAC-EMT [13], and the dynamic level function, which is calculated by summing the spectrum over the frequency bins and expressing in dB scale with frames of 1,024 samples at increments of 256 samples [13]. According to the ways of deciding the values of *maxPos* for each EMT, we implement two types of energy contour model: one is directly using the parameter, `D-maxPos-M`, in the EMT feature set (denoted as *UEC*), while the other is based on a categorized method (denoted as *CEC*).

The UEC modeling of dynamics is implemented as follows:

STEP 1 Calculating the dynamic levels of all the notes among the five excerpts corresponding to a particular EMT across the three consultants (15 excerpts in total).

STEP 2 Resampling all the dynamic levels so that the values of *maxPos* are equal to the `D-maxPos-M` parameter.

STEP 3 Averaging the whole dynamic levels.

The *UEC* model of the ten EMTs is shown in the left side of the Figure 3. We see that *Scherzando*, *Con Brio*, and *Risoluto* have relatively large variation of the energy contours, while the remaining ones have relatively flat ones. Besides, we observe that the values of *maxPos* for all EMTs lie in the interval of 40–70%. However, this phenomenon is unfortunately not consistent with our observation, as the maximal energy would not always lie in the middle of a note. Some notes have strong attacks and others have maximal energy in the back even within a music piece with a particular EMT. The `D-maxPos-M` falling into the middle portion is probably due to the fact that we have taken average on all the notes in the dataset. This motivates us to take the following alternative model.

The CEC modeling of dynamics classifies the notes into three categories among the 15 excerpts of each EMT:

$$\text{note} \in \begin{cases} \text{Front}, & \text{if } \textit{maxPos} < 0.33 \\ \text{Middle}, & \text{if } 0.33 \leq \textit{maxPos} < 0.66 \\ \text{Back}, & \text{otherwise.} \end{cases} \tag{2}$$

After doing this, we count the number of notes for each category. Certainly, as seen in Figure 4, the dominant one of each EMT is different. We simply select the relative category of each EMT, i.e., discarding the remaining ones, to construct a new energy contour model. For example, as most of notes are classified into the back category in the *Tranquillo* case, we take such notes to modeling its own energy contour.

Accordingly, the *CEC* model is realized as follows:

STEP 1 Computing the amount of notes in the front/middle/back category of the ten EMTs using the equation (2).

STEP 2 Selecting the relative majority category of each EMT and taking the notes belonging to this particular group for the dynamic level calculation.

STEP 3 Repeating the three steps of the *UEC* model.

The right-hand side of the Figure 3 shows the estimation of energy contours for each EMT based on the *CEC* model. We notice that *Risoluto* has a strong attack, and *Scherzando* as well as *Con Brio* still has a maximal energy in the middle. Besides, the others have slowly increasing curves which reach the highest energy in the end of a note. The performance of these two models will be evaluated in the classification experiment. Ultimately, we carried out the dynamic manipulation by means of applying a particular energy contour to each note, together with the multiplication of the mean/maximal energy of every note and the parameter, `D-M-`$\mathrm{C_M}$/`D-Max-`$\mathrm{C_M}$.

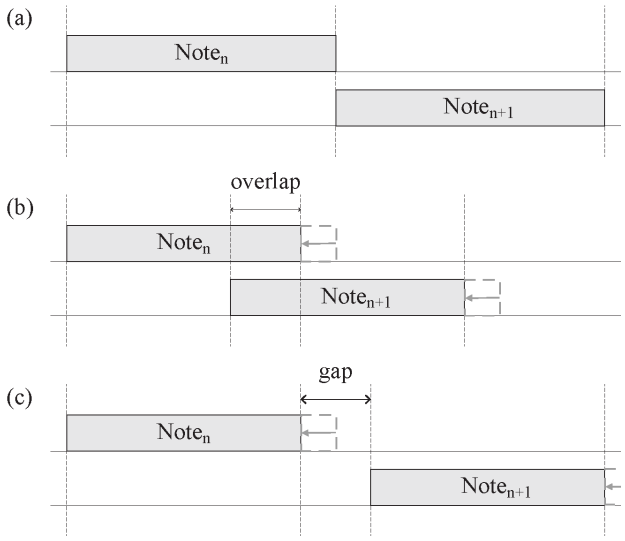


Figure 5: *Illustration of time shortening: (a) The original duration of two consecutive notes, (b) The shortened Note$_n$ followed by an overlapping Note$_{n+1}$, (c) A silent gap between the shortened Note$_n$ and Note$_{n+1}$ if ND–C_M < FPD–C_M.*

2.3. Duration Features

The deviation of timing is also an important expressive factor used by performers [8]. In the EMT feature set, we use `4MD-`$\mathrm{C_M}$, `FPD-`$\mathrm{C_M}$ and `ND-`$\mathrm{C_M}$, defined as the mean length of a four-measure segment, of a full music piece, and of every single note, respectively. Firstly, we stretch a non-expressive note through the phase vocoder and the time-scaling factor is according to the parameter, `ND-`$\mathrm{C_M}$ for each EMT. The length of synthesized note is described as follows:

$$\mathrm{ND_{Synthesis}} = \mathrm{ND_{None}} \times \mathrm{ND\text{–}C_M}\,. \tag{3}$$

Next, we take the `FPD-`$\mathrm{C_M}$ into account and calculate the reasonable onset position of stretched note by the following equation:

$$\mathrm{Onset_{Synthesis}} = \mathrm{Onset_{None}} \times \mathrm{FPD\text{–}C_M}\,. \tag{4}$$

In general, there is an overlap between two consecutive notes in the time shrinking case. However, an abrupt and silent gap may occur in some expressions such as *Tranquillo* if `ND-`$\mathrm{C_M}$ < `FPD-`$\mathrm{C_M}$. This is illustrated in Figure 5. In such a case, the synthesized tone can not keep the temporal continuity of sound. To address this issue, the `FPD-`$\mathrm{C_M}$ will be set equal to the `ND-`$\mathrm{C_M}$ in such condition. Moreover, we stretch every four-measure segment according to the value of `4MD-`$\mathrm{C_M}$ for each EMT. A Hann sliding window of 1,024 samples and a fine hop size of 100 samples are adopted in the phase vocoder module.

2.4. Classification

To evaluate the performance of the synthesis result, we take advantage of the classification models constructed from the prior work [13] for the machine recognition of the ten EMTs. Specifically, the radial-basis function (RBF) kernel Support Vector Machine (SVM) implemented by LIBSVM [23] is adopted for classification. In the training process, we use SCREAM-MAC-EMT

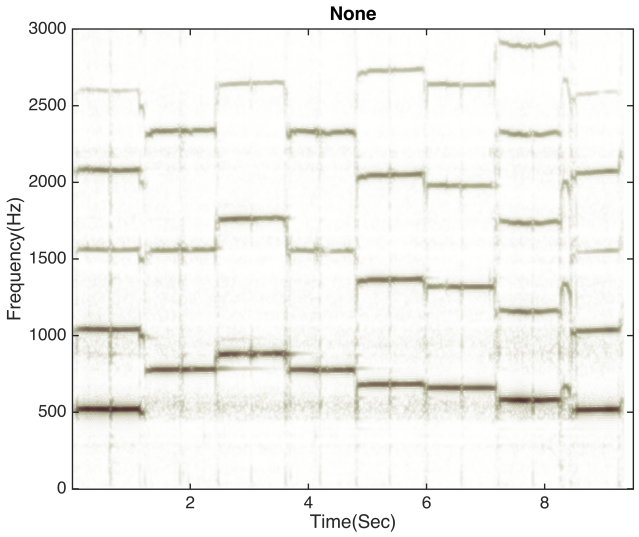


Figure 6: *The first phrase of Mozart's Variationen with a mechanical interpretation performed by an amateur student.*

and take 11-fold cross validation, that is, leave-one-violinist-out in each fold. Besides, the feature selection process is performed by using the ReliefF routine of the MATLAB statistics toolbox [24]. In order to obtain optimized SVM models, the parameters c and γ of the SVM and the top-n' most relevant features are taken based on the highest average accuracy across the 11 folds. In the testing process, each of the outside data, two real recordings collected in this study and six synthetic versions (see Section 3.1 for details), need to be normalized prior to classification. Then the data are fed into the eleven SVM models in conjunction with corresponding relevant features produced in each fold, and the accuracy is computed by averaging over the eleven results. According to [13], the values of c, γ for the SVM classifier and the optimal feature number $n_{\rm opt}$ are set to 1, 2^{-6}, and 36 separately.

3. RESULTS

3.1. Synthesis Results

In this paper, we consider three different sources of the ten non-expressive music pieces, that is, MIDI, amateur student, and professional violinist, in order to observe the differences between the people who have distinct violin trainings and skills. The last two data are recorded in accordance with the collection method of the SCREAM-MAC-EMT. To compare the real recordings with the synthetic versions, both of them perform not only the mechanical interpretation of the ten classical music pieces but also the EMT versions according to the settings of the dataset. In other words, the two people record all the sixty excerpts one by one in a real-world environment. Similarly, to evaluate the proposed synthesis method, each non-expressive excerpt is synthesized in five distinct expressive versions. Moreover, based on the two energy contour models, all the three sources have two types of synthesized sounds. In sum, we have two original and six synthetic data, and each data has sixty excerpts. The following figures, restricting spectrograms from 0 to 3 kHz, illustrate the variations in vibrato, dynamics, and duration. Figure 6 shows an example of mechanical interpretation performed by the amateur, while the three particular expressive versions are demonstrated in Figure 7 which contains original and corresponding synthesized versions in the upper and lower rows respectively. Comparing to the original, we notice that *Scherzando* has a little faster tempo but both *Risoluto* and *Maestoso* have slower one. Besides, all the three synthetic results have more powerful dynamics and stronger vibrato.

Table 2: *The average accuracy compared between the original and synthetic versions which utilize the uncategorized and categorized energy contour models (UEC and CEC, respectively), across three distinct sources.*

Data	MIDI	Amateur	Expert
Original	—	0.293	0.605
UEC	0.578	0.656	0.595
CEC	0.482	0.687	0.615

Table 3: *F-scores of the ten EMTs compared with the original as well as the synthetic version based on the categorized energy contour (CEC) model.*

EMT	MIDI	Amateur		Expert	
	CEC	Original	CEC	Original	CEC
Scherzando	0.878	0.407	0.857	0.653	0.738
Affettuoso	0.317	0.270	0.503	0.436	0.561
Tranquillo	0.923	0.711	0.835	0.838	0.866
Grazioso	0.252	0.250	0.542	0.468	0.516
Con Brio	0.381	0.047	0.770	0.442	0.606
Agitato	0.524	0.397	0.981	0.764	0.922
Espressivo	0.472	NaN	0.231	0.588	NaN
Maestoso	0.483	0.345	0.519	0.815	0.557
Cantabile	0.132	NaN	0.667	0.610	0.459
Risoluto	0.500	0.286	0.855	0.549	0.660

3.2. Classified Results

The objective evaluation of the machine recognition of the ten EMTs is applied to these outside data via the classification models built up from the preliminary work. Hence, the average accuracy predicted by the eleven SVM models among original and synthetic data across the three sources is listed in the Table 2. Additionally, the performances of the two energy contour models are also displayed. Firstly, the MIDI achieves higher classified accuracy when using the *UEC* model. However, all the other synthetic versions have better performance than MIDI. Secondly, the amateur attains an accuracy less than 30% based on the original data but more than 60% among the synthetic ones. There are highly significant differences on both the synthetic data from the original one as validated by a one-tailed t-test (p<0.00001, d.f.=20). In particular, the *CEC* version, using the categorized method to model the dynamic envelopes, achieves the highest accuracy of 0.687, showing a slight improvement from the *UEC* version (p<0.05). Finally, the original data of the expert attain a great performance and the average accuracy comes to 0.605. Besides, both the synthetic versions have nearly the same classified results as the original. In addition, the average F-scores of the ten EMTs comparing between the original and the *CEC* synthetic version are listed in the Table 3. *Espressivo* and *Cantabile* have unrepresentable values, NaN, in the synthetic version of expert and the original version of

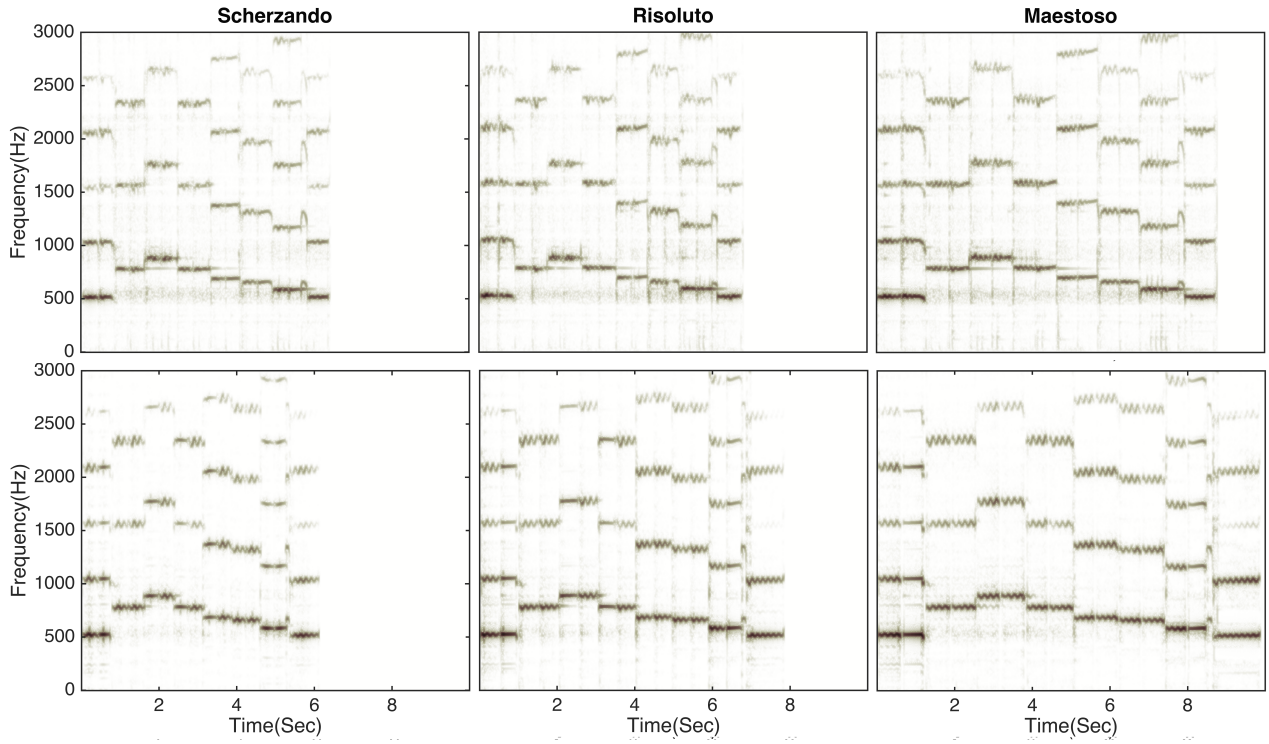


Figure 7: *The first phrase of Mozart's Variationen with three particular EMTs. The upper row shows the original recordings of an amateur while the lower one displays the corresponding synthetic versions based on the specific non-expressive version (see Figure 6).*

amateur respectively because their true positives are zero. Apart from this exception, we find that all the ten EMTs attain higher F-scores in the synthetic version compared with the original of the amateur. Moreover, *Scherzando*, *Tranquillo* and *Agitato* are easily recognized among the five data probably because the first two have lighter dynamics than other EMTs and the last one has faster tempo in most cases.

3.3. Discussion

According to the experimental results, the synthetic data produced by means of the proposed system attain high performances. Specifically speaking, almost all the synthetic versions achieve more than 50% accuracy in the EMT classification task. Particularly, the *CEC* synthetic version of the amateur has significant difference than the original, implying that the virtual simulated violinist is closer to the model of eleven violinists than the amateur. However, the average results are based on the 11 SVM models and corresponding relevant features, which are derived from the 11-fold cross validation from the prior work [13]. We adopt this criterion in order to not only carry on the work but also evaluate the performance of synthesis system via a objective method. In the real application, we will use all the training data to generate a unified model.

Although the synthetic versions obtain great accuracy in classifying the ten EMTs, we could not judge that they have the same expressiveness as the original, or even better than that in the amateur case, by means of the machine recognition. Especially, we only use the nine average features in the synthesis system so both the subtle deviation and the diverse interpretation in violin performance could not be modeled. Hence, the human recognition of expressions is necessary. This work represents an important part of our EMT analysis/synthesis project. A listening test is under construction for it is interesting to know how subjects perform in this regard when original and synthesized sounds are presented. It is expected that such results can be beneficial to this study.

4. CONCLUSION AND FUTURE WORK

In this study, we have presented an automatic system for expressive synthesis from a mechanical sound by means of a small set of interpretational factors derived from the preliminary analysis results. The synthetic data coming from three distinct sources with the dynamic, vibrato, and duration manipulations achieve more than 50% accuracy in the expressive musical term classification task. The performance of two energy contour models is also reported. Specifically, the synthetic versions based on the non-expressive excerpts of an amateur student are closer to the classified models than the original, providing insights into the application of computer-aided music education such as performance calibration in pitch, tempo, and articulation. For future work, we will consider to adopt other features for generating more expressive versions and to conduct a listening test for subjective evaluation.

5. ACKNOWLEDGMENTS

The authors would like to thank the Ministry of Science and Technology of Taiwan for its financial support of this work, under contract MOST 103-2221-E-006-140-MY3.

6. REFERENCES

[1] M. Barthet, P. Depalle, R. Kronland-Martinet, and S. Ystad, "Analysis-by-synthesis of timbre, timing, and dynamics in expressive clarinet performance," *Music Perception*, vol. 28, no. 3, pp. 265–278, 2011.

[2] A. Friberg, "Digital audio emotions-an overview of computer analysis and synthesis of emotional expression in music," in *Proc. of the 11th International Conference on Digital Audio Effects*, 2008.

[3] G. De Poli, A. Rodà, and A. Vidolin, "Note-by-note analysis of the influence of expressive intentions and musical structure in violin performance," *Journal of New Music Research*, vol. 27, no. 3, pp. 293–321, 1998.

[4] G. Widmer and W. Goebl, "Computational models of expressive music performance: The state of the art," *Journal of New Music Research*, vol. 33, no. 3, pp. 203–216, 2004.

[5] R. Bresin and G. U. Battel, "Articulation strategies in expressive piano performance analysis of legato, staccato, and repeated notes in performances of the andante movement of mozart's sonata in g major (k 545)," *Journal of New Music Research*, vol. 29, no. 3, pp. 211–224, 2000.

[6] R. Ramirez, E. Maestre, and X. Serra, "A rule-based evolutionary approach to music performance modeling," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 1, pp. 96–107, 2012.

[7] R. Bresin and A. Friberg, "Synthesis and decoding of emotionally expressive music performance," in *Proc. of the IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 1999, vol. 4.

[8] E. Maestre and E. Gómez, "Automatic characterization of dynamics and articulation of expressive monophonic recordings," in *Proc. of the 118th Audio Engineering Society Convention*, 2005.

[9] G. D'Incà and L. Mion, "Expressive audio synthesis: From performances to sounds," in *Proc. of the 12th International Conference on Auditory Display*, 2006.

[10] M. Grachten and G. Widmer, "Linear basis models for prediction and analysis of musical expression," *Journal of New Music Research*, vol. 41, no. 4, pp. 311–322, 2012.

[11] C. Erkut, V. Välimäki, M. Karjalainen, and M. Laurson, "Extraction of physical and expressive parameters for model-based sound synthesis of the classical guitar," in *Proc. of the 108th Audio Engineering Society Convention*, 2000.

[12] Alfonso Perez and Rafael Ramirez, "Towards realistic and natural synthesis of musical performances: Performer, instrument and sound modeling," in *Proc. of the Third Vienna Talk on Music Acoustics*, 2015.

[13] P.-C. Li, L. Su, Y.-H. Yang, and A.W.Y. Su, "Analysis of expressive musical terms in violin using score-informed and expression-based audio features," in *Proc. of the 16th International Society for Music Information Retrieval Conference*, 2015.

[14] O. Lartillot and P. Toiviainen, "A matlab toolbox for musical feature extraction from audio," in *Proc. of the 10th International Conference on Digital Audio Effects*, 2007.

[15] J.L. Flanagan and R.M. Golden, "Phase vocoder," *Bell System Technical Journal*, vol. 45, no. 9, pp. 1493–1509, 1966.

[16] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of audio," *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 3, pp. 323–332, 1999.

[17] A. Röbel, "A new approach to transient processing in the phase vocoder," in *Proc. of the 6th International Conference on Digital Audio Effects*, 2003.

[18] M. Mellody and G.H. Wakefield, "The time-frequency characteristics of violin vibrato: Modal distribution analysis and synthesis," *The Journal of the Acoustical Society of America*, vol. 107, no. 1, pp. 598–611, 2000.

[19] L. Yang, K. Rajab, and E. Chew, "Vibrato performance style: A case study comparing erhu and violin," in *Proc. of the 10th International Conference on Computer Music Multidisciplinary Research*, 2013.

[20] J. Sundberg, "Acoustic and psychoacoustic aspects of vocal vibrato," *Vibrato*, pp. 35–62, 1995.

[21] A. Röbel and X. Rodet, "Efficient spectral envelope estimation and its application to pitch shifting and envelope preservation," in *Proc. of the 8th International Conference on Digital Audio Effects*, 2005.

[22] A. Gabrielsson and P.N. Juslin, "Emotional expression in music performance: Between the performer's intention and the listener's experience," *Psychology of Music*, vol. 24, no. 1, pp. 68–91, 1996.

[23] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27, 2011.

[24] M. R. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Machine Learning*, vol. 53, no. 1-2, pp. 23–69, 2003.

CONCATENATIVE SOUND TEXTURE SYNTHESIS METHODS AND EVALUATION

Diemo Schwarz and Axel Roebel

UMR STMS IRCAM - CNRS - UPMC
Paris, France
firstname.lastname@ircam.fr

Chunghsin Yeh and Amaury LaBurthe

AudioGaming
Toulouse, France
firstname.lastname@audiogaming.net

ABSTRACT

Concatenative synthesis is a practical approach to sound texture synthesis because of its nature in keeping realistic short-time signal characteristics. In this article, we investigate three concatenative synthesis methods for sound textures: concatenative synthesis with descriptor controls (CSDC), Montage synthesis (MS) and a new method called AudioTexture (AT). The respective algorithms are presented, focusing on the identification and selection of concatenation units. The evaluation demonstrates that the presented algorithms are of close performance in terms of quality and similarity compared to the reference original sounds.

1. INTRODUCTION

Sound texture synthesis is an emerging research topic. It inspires to explore the physics and signal characteristics of sounds other than music and speech, and it also has a great potential to applications in the film, broadcast and video game industries. Sound textures are generally understood as sound that is composed of many micro-events but have features that are stable on a larger time-scale, such as wind, rain, fire, stream, insects, crowd cheering or applause. In this work, we will focus on sounds generated by real-world physics such as environmental, mechanical, and crowd sounds. The imaginary ambiance sounds or creative texture-like sounds are therefore not within the context of this work.

Among existing sound texture synthesis methods [16], granular synthesis is a relatively practical approach as it makes use of snippets of a sound recording, and thus inherits the short-time signal's timbre which provides a shortcut to naturalness. Concatenative synthesis can be seen as a particular kind of granular synthesis [15] and we will be using this term to distinguish it from many commercial granular synthesis products that usually generate sounds of different timbre than that of the original sounds. Since 2012, the authors have been working together on the French national project PHYSIS[1]: an industrial project focused on the modeling, transformation and real-time synthesis of diegetic sounds for interactive virtual worlds and augmented reality. The developed concatenative synthesis methods include: Concatenative Synthesis with Descriptor Controls (CSDC) controls transitions between segments using audio descriptors, thus preventing artefacts (section 2.1); Montage Synthesis (MS) analyzes energy evolution in multiple sub-bands and re-synthesizes a sound texture by means of replacing similar atoms to re-create the sequence of events (section 2.2); AudioTexture (AT) is an algorithm used in a commercial car engine sound synthesizer AudioMotors, which is originally designed to synthesize tire-rolling sound (section 2.3).

In this article, the main emphasis is laid on perceptual evaluation of concatenative synthesis methods for sound textures. Perceptual audio evaluation is a well understood topic [1], but the latest comprehensive survey of methods for sound textures [16] found only some previous work involving evaluation by perceptual listening tests, e.g. [4, 6, 8]. Since then, listening tests have been more systematically carried out in the literature for sound textures [5, 7, 13, 18, 20] and general sound synthesis for virtual reality and gaming [10–12]. The evaluated use-case in this work is example-based sound textures extending an environmental sound texture recording for an arbitrary amount of time, even from varying (non-stable) recordings or periodic sounds, where looping would be easily detectable.

The article is organized as follows. We briefly introduce and compare the algorithms and discuss the respective advantages and disadvantages in section 2. Then, we present the listening test database and setup in section 3. The evaluation results are analyzed and discussed w.r.t. the quality and similarity compared to the reference original sounds in section 3.3. Finally we draw conclusions and present perspectives in section 4. Since this article focuses on the evaluation part, we will only describe the methods in general; readers are invited to consult the implementation details in the respective references.

2. METHODS

The principle of concatenative synthesis is to concatenate sound units in a random or controlled order. The sound units can be defined either by a fixed size (granular synthesis) or by more sophisticated analysis methods. The concatenation between two selected units is carried out by cross-fading using an analysis window such as hanning. The cross-fade shall result in smooth transition provided that the selected units are of similar timbre characteristics at the boundary. For sound texture synthesis, the underlying events are usually evolving (energy, phase, modulation, etc.). Assuming that the events can be identified as consecutive units, we propose to study the identification and selection of sound units which help to reconstruct sound textures that preserve the perceptual quality of the original timbre and the underlying events.

2.1. Concatenative Synthesis with Descriptor Controls

The CSDC method [18] is based on randomized granular playback with control of the similarity between grains using a timbral distance measure based on audio descriptors. In previous work [18], this distance measure has been validated as generally superior to an MFCC-based timbral distance and to uncontrolled purely randomized playback. CSDC is based on corpus-based concatenative synthesis (CBCS) [15], that can be seen as a content-based extension of granular synthesis, which allows unit (grain) selection controlled by audio descriptors.

[1] https://sites.google.com/site/physisproject

In order to synthesize a varying texture without audible repetitions nor artefacts such as abrupt timbral or loudness changes, we use a timbral distance measure between the last played grain and all other grains as candidates, and randomly select a successor grain from the timbrally closest grains, thus generating a random walk through the timbral space of the recording, that never takes too far a step, but that potentially still traverses the whole space of variety of the recording.

The timbre is determined by the audio descriptors suggested by Schwarz and Caramiaux [17] with the addition of pitch. This choice has been validated by Schwarz and O'Leary [18]. The 6 instantaneous descriptors *Loudness*, *FundamentalFrequency*, *Noisiness*, *SpectralCentroid*, *SpectralSpread*, *SpectralSlope* are extracted with the IRCAMDESCRIPTOR library [14] and averaged over all frames of size 23 ms. To avoid too regular triggering of new grains, the duration and time of the played grains are randomly drawn within a 600–1000 ms range, and a random start offset of +/- 200 ms is applied to each grain. Grains are overlapped by 200 ms, and an equal-power sinusoidal cross-fade is applied during the overlap.

2.2. Montage Synthesis

The MS algorithm [13] looks to exploit regions of similarity in the original texture to inform the sequencing of sampled elements. There are two levels to the synthesis model. Longer term sections, called segments, are used to model the higher level structure of textures. These segments are synthesized from the concatenation of shorter sections, called atoms.

In the analysis phase, sub-band energy envelopes are extracted based on perceptual criteria by using the ERB (Equivalent Rectangular Bandwidth) scale and loudness-scale approximation. A time-frequency atom is defined by a duration of 100 ms such that it is long enough to enable the comparison of envelopes and short enough to allow variations in the synthesis phase. An envelope difference measure is proposed to measure the similarity between atoms (local texture structure) and to derive the segments (long term evolution). Based on a statistical model, the sequences of both the segments and atoms are automatically re-synthesized to avoid repetition. A new overlap-add method is also proposed based on frequency-dependent cross-fading length and position in the spectral domain. In principle, the cross fade region is taken to be 4 times the inverse of the bin center frequency, and the possibly different cross-fade position for each bin minimize phase discontinuities. This enables concatenation with short overlap without introducing perceptible modulations.

2.3. AudioTexture

The goal of AudioTexture (AT) is to allow sound designers to make use of any sound texture recordings available and re-create the same sound textures (with semantic controls) synchronous to video images for film/TV post-production and video games.

Principle: Similar to the concept of MS, we view sound textures as composed of two levels of events: micro events (atoms) and macro events (segments). The assumption made here—that a segment boundary represents a new macro event—is essential to identify for good concatenation quality. The micro events are more difficult to handle for complicated textures like crowd cheering/applauding, which will be addressed in future work. We assume that macro events will result in dominant energy variation and thus can be identified from the local maxima of the energy envelope. The boundary between two macro events (assumed to be deterministic) is then defined by the corresponding local minima. Since micro events may result in slight energy variation, the proposed AT algorithm aims to identify prominent local extrema as macro event units.

Method: Similar to several PSOLA (Pitch-Synchronous Overlap–Add) marker analysis algorithms [3], the analysis stage is based on detecting prominent local (energy) maxima as the positions of macro events (glottal pulses in the case of speech). The density of local maxima can be understood as how often a macro event occurs, which in fact is related to the physical behaviors of sound textures. Since the event occurrence frequency varies for different sounds, we simply define a user parameter, minimum macro event duration, to avoid selecting spurious local extrema. Once the macro event units are identified, the synthesis stage uses the common cross-fade method with waveform similarity to refine the cross-fade position [21].

Implementation: In practice, we have found that, by means of low-pass filtering the signal using a biquad filter (gradual attenuation after the cutoff frequency), it is sufficient to obtain a smooth energy envelope of which the local maxima approximate the locations of macro event positions. According to our experimental tests, using a biquad filter of cutoff frequency at 20Hz and of bandwidth 0.3 seems to generalize well over a variety of sound textures. The other practical reason is that the components in the low frequency tend to evolve slowly and are thus more often related in phase for a long-term evolution. The unit identification based on low-frequency emphasized signal seems to produce less perceptually-disturbing phase discontinuity (similar to the idea of MS's frequency-dependent overlap-add treatment). The search of local maxima starts from the beginning of the processed signal. For each local maximum detected, the algorithm selects the largest local maxima within the intervals of minimum macro event duration. The minimum duration of 1 s seems to be sufficiently large to generalize. This parameter should in future work be learned from annotated databases. To concatenate two units during synthesis, the cross-fade region is defined by a quarter of the unit size based on the shape of hanning window.

To summarize, the algorithm (1) searches for local amplitude maxima of the low-passed signal as the macro event positions (2) marks the related local minima (one-to-one correspondence) in the original signal as the macro event boundaries (3) concatenates by cross-fading the macro event units in a random order (or selected order). The algorithm is implemented in a commercial DAW (Digital Audio Workstation) plugin AudioMotors Pro[2] for tire-rolling mode synthesis. Although the product has been made available since 2013, we found that this simple idea has also been suggested in a recent granular synthesis algorithm [19]. AudioTexture is scheduled to be released as a sound design product with preset parameters adapted to different kinds of sound textures.

2.4. Baseline method

We have implemented a baseline method RND similar to Fröjd and Horner's approach [4], which randomly concatenates sound units of sizes randomly drawn between 600 ms to 1 second with 200 ms overlap and equal-power sinusoidal cross-fade. This method is

[2] http://lesound.io/product/audiomotors-pro, free trials are available for download.

Algorithm	Units	Analysis	Synthesis
CSDC	grains	fixed size of 800ms without overlap	descriptor similarity
MS	atoms/segments	sub-band energy envelope difference	sequence model
AT	segments	energy evolution + minimum duration 1 s	random order
RND	grains	random sizes between 600 ms to 1 s	random order

Table 1: This table compares the analysis and synthesis phases of the proposed methods.

used to compare with the three proposed methods to evaluate the effectiveness of sound unit identification and selection.

2.5. Algorithm comparison

An overview of the three sound texture synthesis methods is shown in Table 1. The scale of the units are of the relation RND $\approx$ CSDC < MS < AT. Here we consider the scale of MS defined by its envelope because it imposes a constraint on the selection of the atoms. Like the usual granular synthesis, CSDC uses grain units of a fixed size that is sufficiently large to preserve the local structure generated by micro events. The principal functionality of CSDC is unit selection based on descriptor similarity. RND randomizes both the unit size (close to the grain size of CSDC) and the unit selection. AT is using the largest unit scale ($\geq$ 1 s) for macro events and there is no control strategy applied during the synthesis phase. MS models both micro events by atom units of a fixed size and macro events by segments. A statistical model of atom/segment sequencing is further used at the synthesis stage. The order of complexity of the presented algorithms is RND < AT < CSDC < MS.

3. EVALUATION

The evaluation is carried out by web-based listening tests (see Figure 1). In addition to the concatenative synthesis methods, we have added a signal-model based method SDIS based on spectral domain imposition of statistics [7]. This method is a more efficient implementation of the state-of-art signal-model based method proposed by McDermott and Simoncelli [8]. Since we assume that concatenative synthesis methods generally have the advantage over signal-model based methods in terms of quality, we have included this method for evaluation to verify if all the proposed methods do demonstrate their advantages.

3.1. Experiment setup

The algorithms presented above are evaluated in an ongoing listening test accessible online[3]. The test database contains 27 sound texture examples with equal duration of 7 seconds:

- 14 sounds used by McDermott et. al. in their previous studies on sound texture perception [8]
- 13 sounds contributed by the PHYSIS project partner *GameAudioFactory*[4]

They are carefully selected to cover a wide range of sound textures generated by human, transportation, mechanical gears, animals and natural elements (air, water and fire). Some of the sounds contain explicitly non-uniform environmental sound textures, i.e. containing some variation in texture and timbre, but not clearly perceived as outlier events. They are meant to test an algorithm's capability to re-synthesize slow evolution such as wind blowing. There are also periodic sounds that serve to test the algorithms' capability to preserve the periodic structure without introducing jitter and phase discontinuities.

In the listening test, for each of the 27 sound examples in randomized order, the original is presented to the subject, and then 6 stimuli in randomized order: the resyntheses generated by the five algorithms (CSDC, MS, AT, RND and SDIS) and the original (ORIG) as hidden anchor. Subjects are asked to use a numerical slider between 0 and 100 to rate the stimuli according to the two criteria:

Quality: Presence of artefacts, such as abrupt loudness or timbral changes, cuts, repetitions, loops, distortions, etc. The scale is further divided into 5 levels: bad, poor, fair, good and excellent.

Similarity: Does the resynthsis sounds sufficiently credible like the variation of the original sounds ? The scale is further divided into 5 levels: very dissimilar, somewhat dissimilar, somewhat similar, quite similar and very similar.

3.2. Results

At the time of writing, 17 responders took the test (with 2 only providing partial data). All but 2 reported being familiar with perceptive listening tests, none reported hearing impairment. Age and gender information were not gathered. Figure 2 shows the ratings of quality and similarity (without any scaling). For each algorithm, the rating statistics are calculated over all responses and sounds. We also analyze ratings with respect to different sound classes according to local structure (stable, varying, periodic) in Figure 4 and sound content characteristics (noisy, pitched) in Figure 3:

stable: local structure does not vary along time such as heavy rain sounds (13 sounds)

varying: local structure slightly varies along time such as wind whistling sounds (11 sounds)

periodic: global structure is a repetition of local structure such as helicopter sounds (3 sounds)

noisy: sound does not contain pitched or harmonic components such as gas stove sounds (22 sounds)

pitched: sound contains pitched components such as crowd cheering sounds (5 sounds)

In general, ORIG ranks the best, which validates that the testers are doing a proper job. All the three concatenative synthesis methods are rated in a close range around 80 (between quite similar to very similar). Surprisingly, RND is rated quite well and appears very competitive by the mean similarity measure. All concatenative methods are rated much better than SDIS, which confirms the expected advantages in sound texture synthesis. This seems to

[3] http://ismm.ircam.fr/sound-texture-synthesis-evaluation

[4] http://gameaudiofactory.com

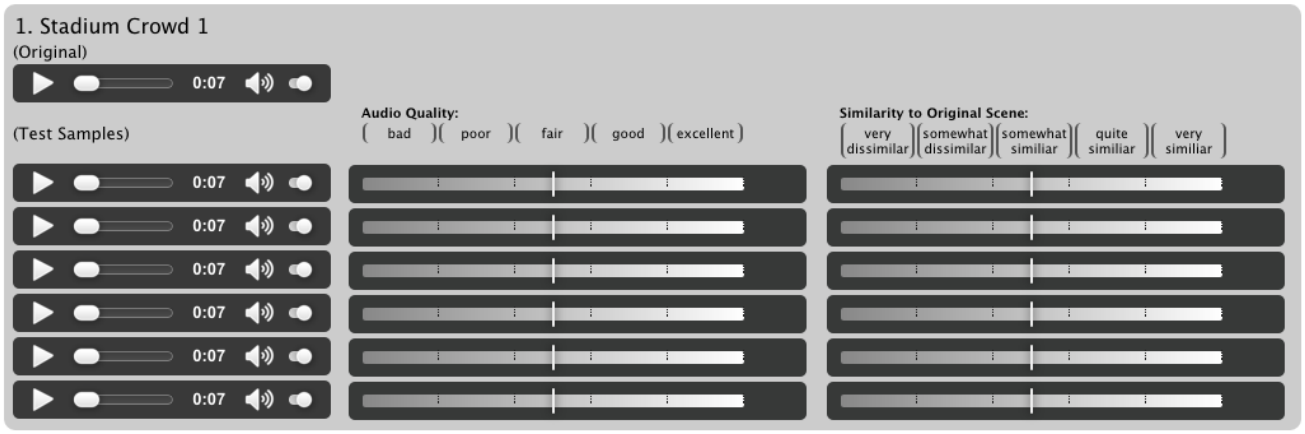


Figure 1: An example of listening test web interface.

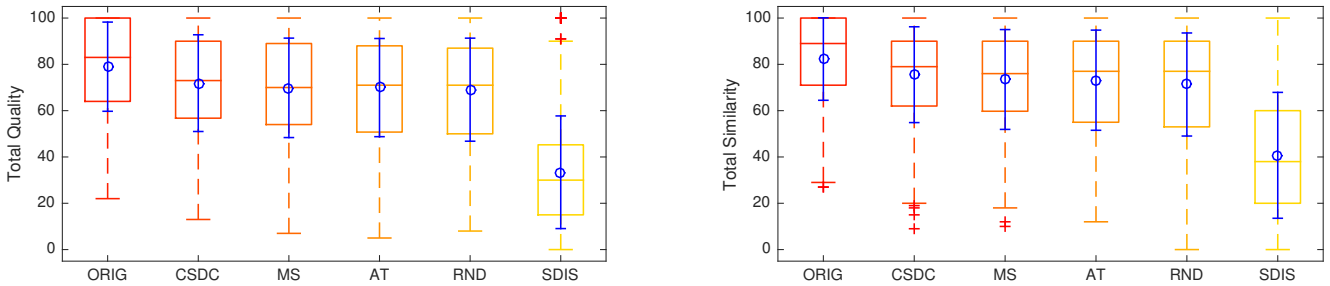


Figure 2: Box plots of the quality and similarity ratings per type of stimulus, showing the mean and standard deviation (blue circle and error bar), median (middle line), quartile range (box), min/max (whiskers), and outliers (crosses).

align with the result obtained in [4] where a concatenative synthesis method seems to be generally rated better than the signal-model based method based on wavelet trees [2].

To test if the observed differences of ratings are significant or simply due to chance, further statistical analysis has been carried out. As Figure 5 shows, the ratings are not normally distributed, so that the Kruskal-Wallis non-parametric method [9] has been applied instead of ANOVA (analysis of variance).[5] Here the null hypothesis H_0 is that the ratings come from the same distribution (and differences in means are thus due to chance), and the alternative hypothesis H_A is that the data comes from different distributions. The significance levels of the p-values for each pair of comparisons are given in Tables 2–7 for the quality and similarity ratings. The significance level depending on the p-value is habitually represented by a number of stars as follows:

Level	*	**	***	****
$p \leq$	0.05	0.01	0.001	0.0001

[5] However, McDonald [9] argues that one-way ANOVA is not very sensitive to non-normal distributions, and indeed, ANOVA with Bonferroni correction gives very similar results in terms of significance of differences of pairs of means: The p-values are generally lower with ANOVA, but only very few passed under the significance threshold of 5%. We report here the more conservative Kruskal-Wallis results.

total	**ORIG**	**CSDC**	**MS**	**AT**	**RND**	**SDIS**
ORIG	—	****	****	****	****	****
CSDC	****	—				****
MS	****		—			****
AT	****			—		****
RND	****				—	****
SDIS	****	****	****	****	****	—

Table 2: Significance level for each pair of differences of means on quality (upper triangle) and similarity (lower triangle) ratings for total results.

stable	**ORIG**	**CSDC**	**MS**	**AT**	**RND**	**SDIS**
ORIG	—			*		****
CSDC		—				****
MS			—			****
AT	**			—		****
RND					—	****
SDIS	****	****	****	****	****	—

Table 3: Significance level for each pair of differences of means on quality (upper triangle) and similarity (lower triangle) ratings for stable sounds.

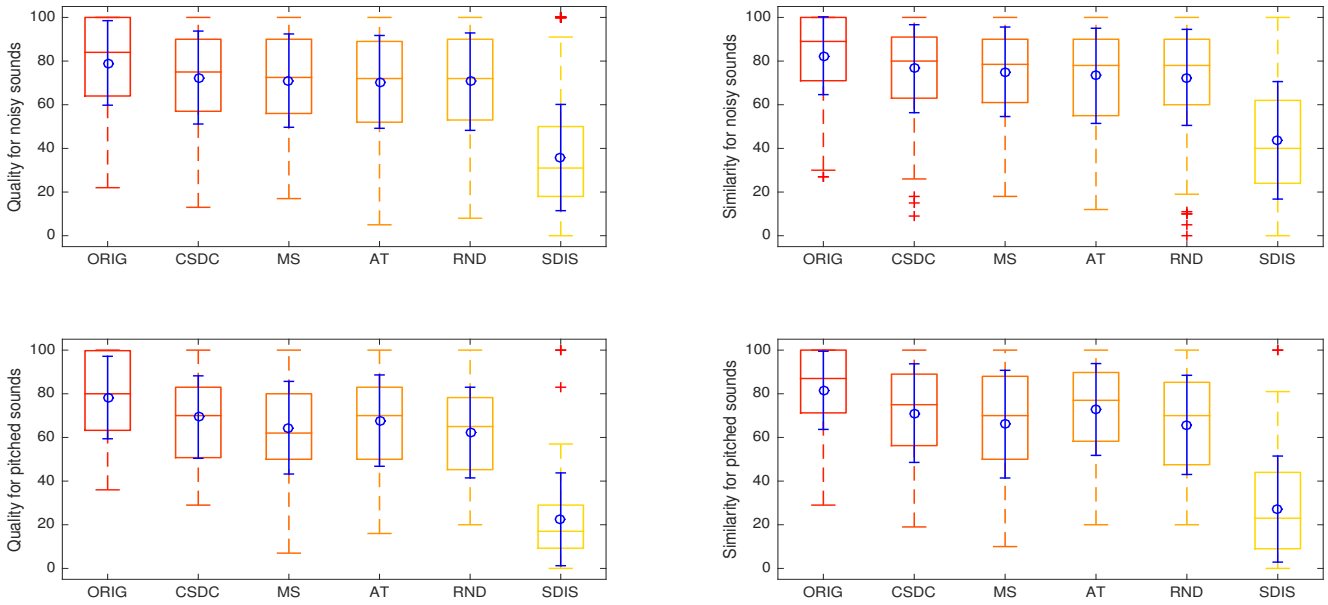


Figure 3: Box plots of the quality and similarity ratings for each sound character (noisy, pitched) per type of stimulus, showing the mean and standard deviation (blue circle and error bar), median (middle line), quartile range (box), min/max (whiskers), and outliers (crosses).

3.3. Discussion

The global results given in Table 2 show that, in general, all resyntheses can be distinguished from the original. However, none of the concatenative synthesis based algorithms can be distinguished amongst each other (the null hypothesis that the differences in ratings are due to randomness can not be rejected). Yet, all concatenative synthesis algorithms can be reliably distinguished from the SDIS algorithm with $p < 0.0001$. The two latter points hold for all subsets of sounds in Tables 3–7.

For *stable* sounds (Table 3, Figure 4 top), the granular resyntheses (with the exception of AT but including RND) can not be distinguished from the original. For *varying* sounds (Table 4, Figure 4 middle), CSDC is significantly better than RND, as well as AT for the similarity rating. For *periodic* sounds (Table 5, Figure 4 bottom), only RND can be distinguished from the original, as well as AT for the similarity rating. However, there are only 3 sounds in this class, so the results should be taken with care.

Noisy sounds (Table 6, Figure 3 top) share the interpretation

varying	ORIG	CSDC	MS	AT	RND	SDIS
ORIG	—	*	****	**	****	****
CSDC	**	—			*	****
MS	****		—			****
AT	**			—		****
RND	****	*		*	—	****
SDIS	****	****	****	****	****	—

Table 4: Significance level for each pair of differences of means on quality (upper triangle) and similarity (lower triangle) ratings for varying sounds.

noisy	ORIG	CSDC	MS	AT	RND	SDIS
ORIG	—	**	****	****	****	****
CSDC	**	—				****
MS	****		—			****
AT	****			—		****
RND	****				—	****
SDIS	****	****	****	****	****	—

Table 6: Significance level for each pair of differences of means on quality (upper triangle) and similarity (lower triangle) ratings for noisy sounds.

periodic	ORIG	CSDC	MS	AT	RND	SDIS
ORIG	—				**	****
CSDC		—				****
MS			—			****
AT	**			—		****
RND	***				—	***
SDIS	****	****	****	***	**	—

Table 5: Significance level for each pair of differences of means on quality (upper triangle) and similarity (lower triangle) ratings for periodic sounds.

pitched	ORIG	CSDC	MS	AT	RND	SDIS
ORIG	—		**		***	****
CSDC		—				****
MS	**		—			****
AT				—		****
RND	***				—	****
SDIS	****	****	****	****	****	—

Table 7: Significance level for each pair of differences of means on quality (upper triangle) and similarity (lower triangle) ratings for pitched sounds.

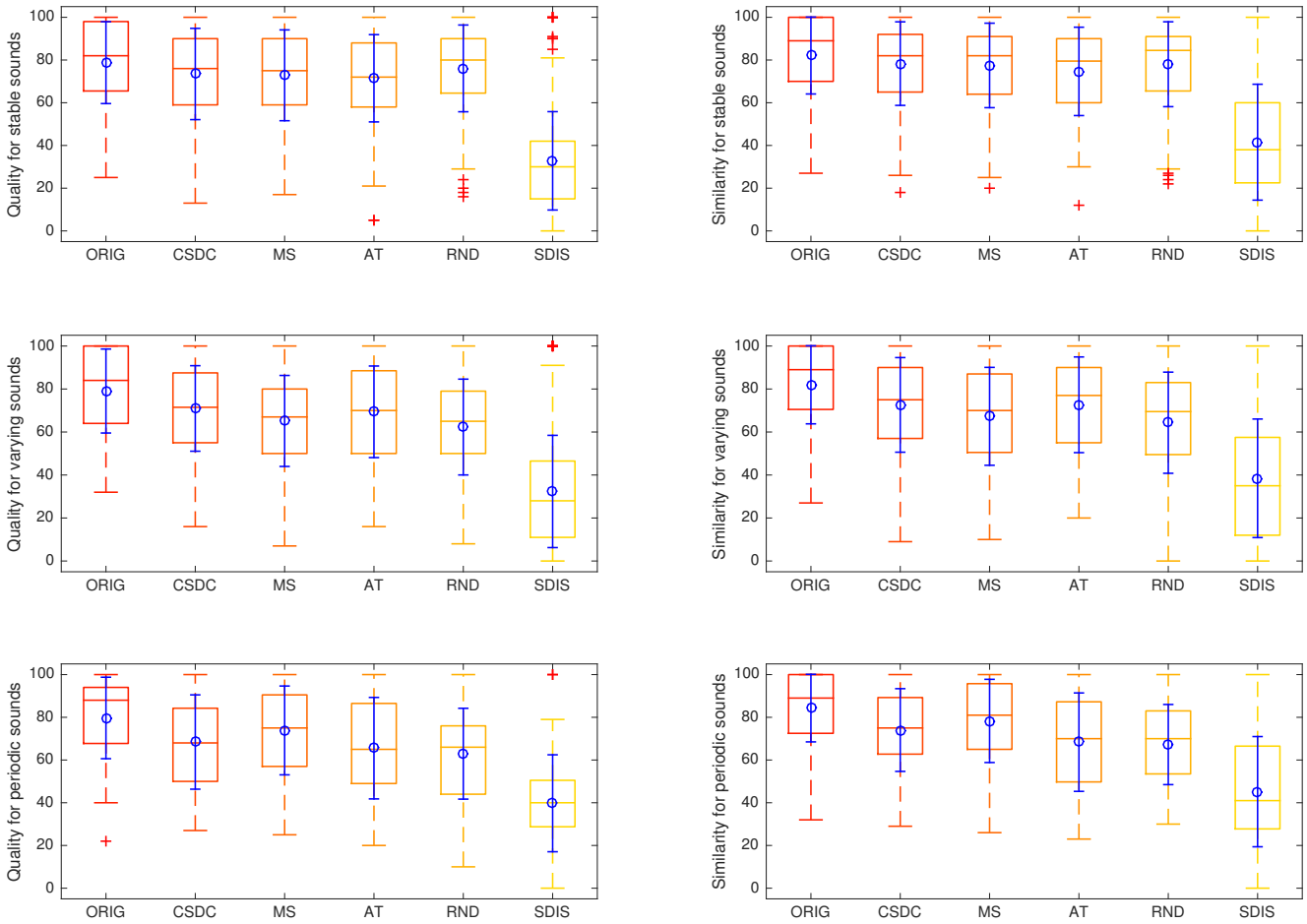


Figure 4: Box plots of the quality and similarity ratings for each sound class (stable, varying, periodic) per type of stimulus, showing the mean and standard deviation (blue circle and error bar), median (middle line), quartile range (box), min/max (whiskers), and outliers (crosses).

for global results: all resyntheses can be distinguished from the original and from SDIS. For *pitched* sounds (Table 7, Figure 3 bottom), CSDC and AT can not be significantly distinguished from the original.

In general, the proposed concatenative synthesis methods obtain slightly better ratings than RND, although the differences are not significant, except for the sound classes of varying sounds, where CSDC and AT (for similarity only) have significantly (with $p < 0.05$) better ratings. We may summarize the advantages of the concatenative algorithms as follows:

CSDC: Unit selection based on descriptor similarity is very effective provided that the unit size is fixed. That is, the descriptors characterize well the units such that the selected units follow a credible sequence even for time-varying sound textures: it shows its strength for stable, periodic, and pitched sounds, where it can not be distinguised significantly from the original, and for varying sounds, where the distinction is less significant than for the other methods. For varying sounds, it is significantly better than RND.

MS: The statistical sequence modeling is very effective for stable and periodic (almost identical to ORIG) sounds with a fixed atom size and varying segment length. However, it tends to have less favorable rating for *varying* sounds. This could possibly be improved by parameter refinement to allow longer evolution of macro events (segments).

AT: The unit analysis is quite promising provided its simplicity. Since there is no treatment to handle unit selection, it may result in less satisfying quality for *varying* sounds such as lapping waves and crowd cheering sounds, and its statistically significant difference to the original for stable sounds shows that there is room for improvement.

4. CONCLUSION

We have evaluated the proposed concatenative methods for sound texture synthesis, each of different degrees of complexity (RND: simple random choice, AT: random choice with simple unit identification, CSDC: unit selection by sound descriptors, MS: unit (segment) identification, sequence modeling and matching). Using a database of sound texture examples relevant to gaming and multimedia applications, the evaluation results had little difference in their mean ratings. The proposed three concatenative synthesis

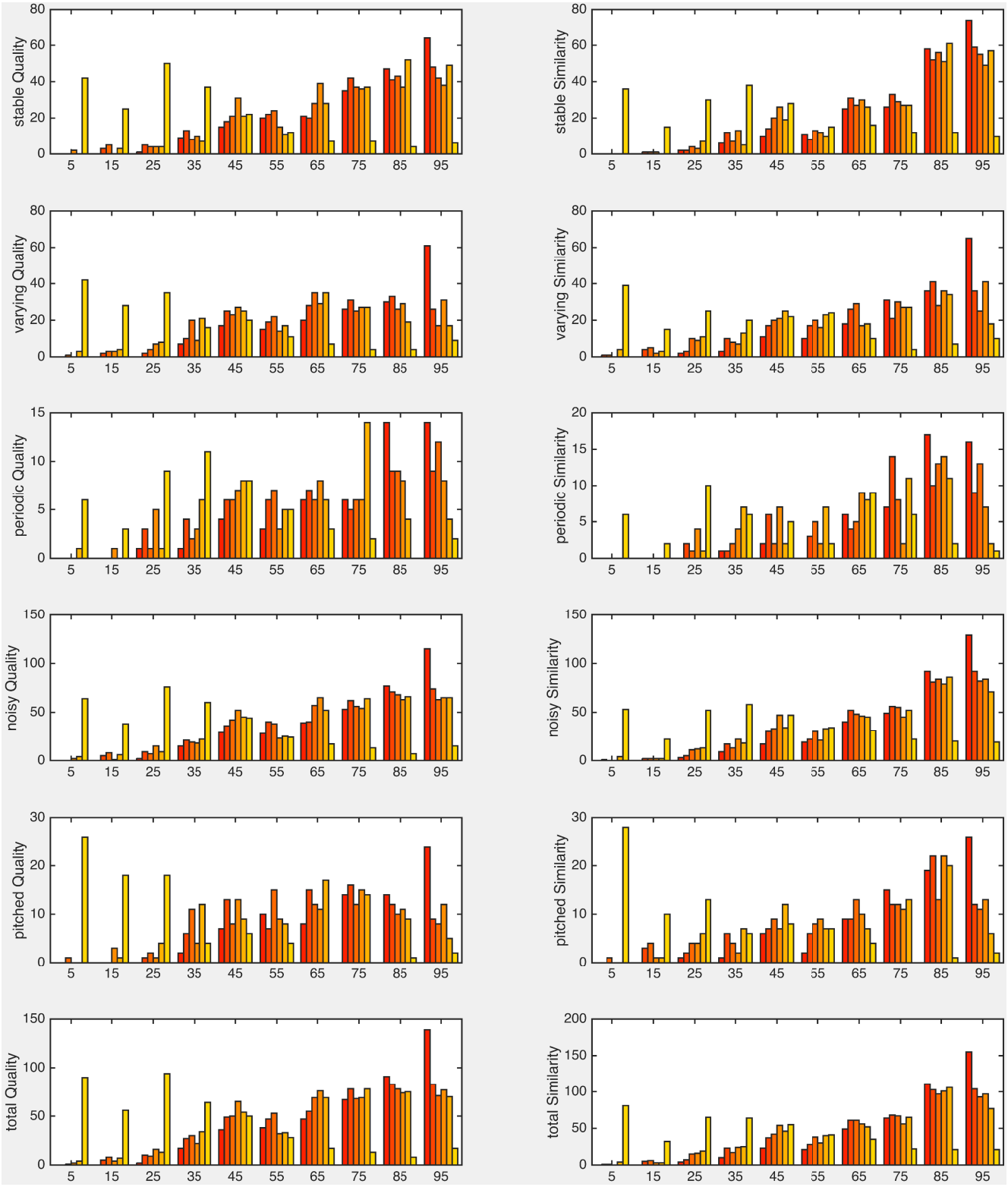


Figure 5: Histograms of ratings per bins of 10 rating points. Order of bars as in the previous figures: ORIG, CSDC, MS, AT, RND, SDIS.

methods appear to be slightly advantageous to the baseline method RND. A finer analysis of the results sound-by-sound is yet to be carried out, which may reveal edge cases that lead to more insights on what treatment is necessary for different types of sound textures and eventually an improvement of the design of the evaluation database. Since the differences in quality/similarity among proposed methods are generally not statistically significant, one may need to design objective evaluation measures. For example, a good concatenative synthesis should generate high quality sound textures while shuffling a lot the units that do not follow their original order. That is, the resynthesis shall have as many as possible re-ordered grains compared to the original sounds.

The concatenative methods evaluated in this article do demonstrate their advantages over the signal model based method SDIS. Notice that we do not draw a conclusion here that the concatenative synthesis methods are better than the signal model based methods but we do confirm certain observations like the results obtained in [4]. However, it is true that it is generally more difficult to develop a signal-model based method that compete in quality with the concatenative methods for sound texture synthesis.

To further improve the algorithms, the principal ideas of certain algorithms can benefit each other. However, the common challenge to sound texture algorithms are varying sounds as shown in Figure 2. We believe that it is essential to model a long-term evolution (one cycle of lapping waves) as well as physically coherent behavior (cycles of lapping waves). Based on the same algorithm, for instance, one may adapt the analysis, control and synthesis parameters to each kind of sound textures. The possible advantages of parametric synthesis methods, such as based on a signal or physical model, or physically-informed [16], in terms of controllability and adaptability to a given temporal evolution are beginning to be attained by recent interactive concatenative methods, e.g. [17].

5. ACKNOWLEDGMENTS

The research presented here is funded by the French *Agence Nationale de la Recherche* (ANR) within the project *PHYSIS*, ANR-12-CORD-0006.

REFERENCES

[1] S. Bech and N. Zacharov. *Perceptual audio evaluation-Theory, method and application*. John Wiley & Sons, 2007.

[2] S. Dubnov, Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman. Synthesis of audio sound textures by learning and resampling of wavelet trees. *IEEE Computer Graphics and Applications*, 22(4):38–48, 2002.

[3] T. Ewender and B. Pfister. Accurate pitch marking for prosodic modification of speech segment. In *Proc. INTERSPEECH*, Makurari, Japan, Sept. 26–30 2010.

[4] M. Fröjd and A. Horner. Sound texture synthesis using an overlap-add/granular synthesis approach. *Journal of the Audio Engineering Society*, 57(1/2):29–37, 2009.

[5] T. Heittola, A. Mesaros, D. Korpi, A. Eronen, and T. Virtanen. Method for creating location-specific audio textures. *EURASIP Journal on Audio, Speech and Music Processing*, 2014.

[6] S. Kersten and H. Purwins. Sound texture synthesis with hidden markov tree models in the wavelet domain. In *Proc. Sound and Music Computing (SMC)*, Barcelona, Spain, July 2010.

[7] W.-H. Liao, A. Roebel, and W.-Y. Su. On the modeling of sound textures based on the STFT representation. In *Proc. Digital Audio Effects (DAFx)*, Maynooth, Ireland, Sept. 2013. URL http://architexte.ircam.fr/textes/Liao13a/.

[8] J. H. McDermott and E. P. Simoncelli. Sound texture perception via statistics of the auditory periphery: evidence from sound synthesis. *Neuron*, 71(5):926–40, Sept. 2011.

[9] J. H. McDonald. *Handbook of Biological Statistics*. Sparky House Publishing, Feb. 2015. URL http://www.biostathandbook.com/.

[10] L. Mengual, D. Moffat, and J. D. Reiss. Modal synthesis of weapon sounds. In *Proc. Audio Engineering Society Conference: 61st International Conference: Audio for Games*. Audio Engineering Society, 2016.

[11] E. Murphy, M. Lagrange, G. Scavone, P. Depalle, and C. Guastavino. Perceptual evaluation of rolling sound synthesis. *Acta Acustica united with Acustica*, 97(5):840–851, 2011.

[12] R. Nordahl, S. Serafin, and L. Turchet. Sound synthesis and evaluation of interactive footsteps for virtual reality applications. In *2010 IEEE Virtual Reality Conference (VR)*, pages 147–153, March 2010.

[13] S. O'Leary and A. Roebel. A two level montage approach to sound texture synthesis with treatment of unique events. In *Proc. Digital Audio Effects (DAFx)*, Germany, Sept. 2014.

[14] G. Peeters. A large set of audio features for sound description (similarity and classification) in the Cuidado project. Technical Report version 1.0, Ircam – Centre Pompidou, Paris, France, Apr. 2004.

[15] D. Schwarz. Corpus-based concatenative synthesis. *IEEE Signal Processing Magazine*, 24(2):92–104, Mar. 2007. Special Section: Signal Processing for Sound Synthesis.

[16] D. Schwarz. State of the art in sound texture synthesis. In *Proc. Digital Audio Effects (DAFx)*, Paris, France, 2011.

[17] D. Schwarz and B. Caramiaux. *Proc. Computer Music Multidisciplinary Research (CMMR) 2013*, volume 8905 of *Lecture Notes in Computer Science (LNCS)*, chapter Interactive Sound Texture Synthesis through Semi-Automatic User Annotations. Springer International Publishing, 2014.

[18] D. Schwarz and S. O'Leary. Smooth granular sound texture synthesis by control of timbral similarity. In *Proc. Sound and Music Computing (SMC)*, Maynooth, Ireland, July 2015. URL https://hal.archives-ouvertes.fr/hal-01182793.

[19] S. Siddiq. Morphing of granular sounds. In *Proc. Digital Audio Effects (DAFx)*, Norway, Nov. 2015.

[20] G. Tiger. *Synthése sonore d'ambiances urbaines pour les applications vidéo-ludiques*. PhD thesis, Conservatoire National des Arts et Metiers (CNAM), 2014.

[21] W. Verhelst and M. Roelands. An overlap–add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech. In *Proc. of Intl. Conf. on Acoustics, Speech, and Signal Processing, ICASSP-93.*, pages 554–557 vol.2, April 1993.

SIGNAL DECORRELATION USING PERCEPTUALLY INFORMED ALLPASS FILTERS

Elliot Kermit-Canfield and Jonathan Abel

Center for Computer Research in Music and Acoustics,
Stanford University, Stanford, CA 94305 USA
[kermit|abel]@ccrma.stanford.edu

ABSTRACT

When a monophonic source signal is projected from two or more loudspeakers, listeners typically perceive a single, phantom source, positioned according to the relative signal amplitudes and speaker locations. While this property is the basis of modern panning algorithms, it is often desirable to control the perceived spatial extent of the phantom source, or to project multiple, separately perceived copies of the signal. So that the human auditory system does not process the loudspeaker outputs as a single coherent source, these effects are commonly achieved by generating a set of mutually decorrelated (e.g., statistically independent) versions of the source signal, which are then panned to make an extended source or multiple, independent source copies.

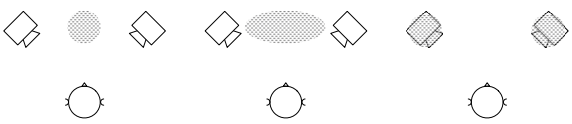
In this paper, we introduce an approach to decorrelation using randomly generated allpass filters, and introduce numerical methods for evaluating the perceptual effectiveness of decorrelation algorithms. By using allpass filters, the signal magnitude is preserved, and the decorrelated copies and original signal will be perceptually very similar. By randomly selecting the magnitude and frequency of the poles of each allpass biquad section in the decorrelating filter, multiple decorrelating filters may be generated that maintain a degree of statistical independence. We present results comparing our approach (including methods for choosing the number of biquad sections and designing the statistics of the pole locations) to several established decorrelation methods discussed in the literature.

1. INTRODUCTION

Signal decorrelation is an important tool for audio upmixing, spatialization, and auraliztion. In the simplest case, when two coherent audio signals are played through loudspeakers, a listener will perceive a single sound source located somewhere between the two speakers, controlled by the relative amplitudes and time delay of the signals. Although this is one of the principles on which stereophonic panning algorithms rely, it is not without problems. For example, when the signals are presented to the listener over headphones, the location of the source is often perceived to be within the listener's head. Additionally, the perceived source width is often reduced. These problems also exist in systems with more than two loudspeakers.

Our goal in decorrelating signals is to reduce the phase coherence of a given signal while maintaining perceptual transparency. That is to say a monophonic file run though our decorrelation algorithm presented on a single speaker should be indistinguishable from the original file. When the same file is run through multiple independent decorrelation filters and presented on multiple speakers, the signals should no longer sum perceptually to a single point and the apparent source width should appear to be extended. See Fig. 1 for a graphical depiction of a phantom source between two speakers and decorrelated phantom sources. In the current work, we are concerned with applications for multichannel and surround sound applications where one might want many decorrelated copies of a signal panned in space.

In the following sections we will briefly review other common decorrelation techniques followed by a description of the measures we use to evaluate perceptual transparency and independence. We will then introduce our approach to signal decorrelation through perceptually-weighted random allpass filters before showing the results of evaluating our approach in comparison to other popular techniques. We will conclude with some recommendations for using decorrelation filters in real-time systems and the implications of this work for further research.



(a) *Phantom image* (b) *Widened ASW* (c) *Independent sources*

Figure 1: *Multiple speakers producing the same signal usually produce a single phantom image (1a) while decorrelation can produce a phantom image with a wider apparent source width (1b) or multiple, statistically independent sources (1c).*

2. STANDARD DECORRELATION TECHNIQUES

Perhaps the simplest decorrelation algorithm is based on convolving the source signal with a short sequence of random samples, scaled to have unit power. The longer we make this sequence the more decorrelation we can achieve, but at the expense of smearing the original signal out in time. For this reason, we typically constrain the length to be shorter than $30\,\mathrm{ms}$ so the decorrelation is not perceived to add reverberation.

In a second approach Kendall proposes an "allpass filter" technique formed by taking the IDFT of a transform constrained to have unit magnitude at the DFT bins with a random phase uniformly distributed on the interval $(-\pi, \pi)$ [1, 2]. Even though this processes creates a filter with unit magnitude *on* the DFT bins, it does not guarantee a flat magnitude response. Additionally, because the DFT is a periodic transform, discontinuities at the start and end of the signal can cause audible artifacts. It is possible to refine this technique by limiting how quickly the phase can change between consecutive samples, but this does not solve the issues related to the magnitude response being only flat on average. Due to the random nature of these algorithms, both techniques work when trying to decorrelate a signal into more than two channels.

Another broad approach involves passing the signal through a filterbank and applying delays to each band of frequencies [3–5]. Researchers have experimented with the number of frequency bands—ranging from as few as three to greater than 24—the spacing of the bands of the filterbank, and the amount of delay in each band. Depending on the complexity of the filterbank, this technique can suffer from frequency cancellation at the band edges during reconstruction, and can be computationally expensive. Furthermore, depending on the algorithm, some these techniques are not necessarily effective for multichannel applications as they take advantages of stereo, complementary processing.

Cabrera proposes a method of sinusoidal modeling with frequency and amplitude modulation to decorrelate signals [6,7]. This method introduces latency due to the signal analysis and can be computationally expensive.[1]

At the additional cost of higher complexity, many of these techniques can be further refined by separating the signal-to-be-processed into steady state and transient components, so that the decorrelation effects can be applied only to the steady state portion. This is done to ensure that transients do not get smeared out in time and become perceptual artifacts.

In addition to the aforementioned techniques, Gardner and Schroeder suggested methods of expanding the spatial extent of a signal using delays and comb filters, but these methods impart strong coloration on the signal [8, 9].

The literature is diverse on decorrelation for upmixing and resynthesizing ambiance with applications for perceptual audio coders. Using a down-mixed, monophonic signal and side-chain information (binaural cue coding), Faller and Baumgarte suggest a perceptually-weighted frequency-domain modification using random sequences for each channel to preserve time and level differences [10, 11]. The MPEG Surround standard includes specification for an allpass filter approach [12, 13].

Valin proposes decorrelating frequencies above 2 kHz using shaped comb-allpass filters and lower frequencies by injecting psychoacoustically masked noise [14]. Zotter et al. take an approach using deterministic allpass filters [15].

Several mono-stereo upmixing techniques rely on leveraging complementary filters or other intrinsic properties of decorrelating only two channels [16–20]. While it is possible that these techniques can be extended for multichannel systems by cascading the algorithms with different parameters, this will not work for all systems (e.g., those require placing signals 90 degrees out of phase).

3. ALLPASS FILTERS FOR DECORRELATION

Allpass filters are useful for signal decorrelation because they maintain a flat frequency response while effecting the phrase and group delay. Digital allpass filters contain poles inside the unit circle matched with zeros at reciprocal magnitudes and at the same frequencies as the pole. In other words, the position of the zeros are reflected across the unit circle. The class of filters we consider are biquad allpass filters that produce a real output signal and have the z transform

$$H(z) = \frac{\rho + z^{-1}}{1 - \overline{\rho} z^{-1}} \cdot \frac{\overline{\rho} + z^{-1}}{1 + \rho z^{-1}} \tag{1}$$

$$= \frac{z^{-2} - 2\Re(\rho) z^{-1} + |\rho|^2}{1 - 2\Re(\rho) z^{-1} + |\rho|^2 z^{-2}}\,, \tag{2}$$

where

$$\rho = \alpha e^{2\pi\omega j}\,, \tag{3}$$

in which α controls the distance of the poles from the unit circle and ω is the angle. This can also be written as the difference equation

$$\begin{aligned} &y[k] - 2\Re(\rho) y[k-1] + |\rho|^2 y[k-2] \\ &\quad = x[k-2] - 2\Re(\rho) x[k-1] + |\rho|^2 x[k]\,, \end{aligned} \tag{4}$$

where $y[k]$ is the output and $x[k]$ is the input at sample k. These allpass filters exhibit conjugate symmetry, which causes their output to be real. This is important for processing real signals as it maintains the magnitude at all real frequencies. Here we propose cascading multiple, randomly generated allpass sections to generate decorrelating filters. Because they are randomly generated, we can create multiple, mutually decorrelated impulse responses with different phase responses, and therefore multiple decorrelated copies of signals.

In the following sections, we will explore the parameterization of this filter structure.

3.1. Pole-Zero Angle

The angle of the pole position controls the frequency at which the slope of the phase response changes fastest. When constructing our cascade of filters, one easy technique for choosing the pole's angle would be to place them randomly. While this is a valid technique, we have chosen to warp the random pole angles by equivalent rectangular bands (ERB) in order to maintain a relatively constant pole density across the critical bands of human hearing. As it turns out, decorrelation is difficult at low frequencies due to long wavelengths. This closely resembles the difficulty humans have localizing low frequency sound sources.

At high frequencies, the human auditory system primarily uses level differences to localize sound sources and decorrelating signals with time delays is potentially wasted [21]. High frequency signals have short wavelengths. Time delays could potentially realign signals' peaks and nulls offset by one or more periods. By warping the random distribution by ERBs, the majority of the poles will be placed in frequency ranges that will have the largest perceptual effect for sound localization.

3.2. Pole-Zero Radii

While the pole angle controls the frequency at which the largest phase change occurs, the pole's radius—the distance between the pole and the unit circle—controls the amount of the phase distortion. Each pole added to the system adds a cumulative π amount of phase to the system. As the pole's radius approaches the center of the unit circle, the phase across all frequencies approaches a linear slope and the response is identical to the phase response of a unit delay. As the pole radius approaches the unit circle, the phase distortion becomes concentrated in a smaller frequency region. Additionally, as the pole approaches the unit circle, the group

[1] Currently this does not run in real-time, but rather performs the analysis offline.

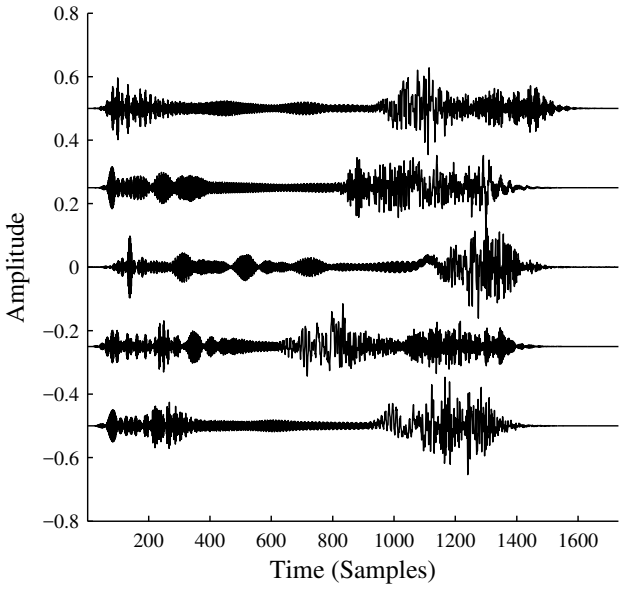


Figure 2: *Five allpass cascade impulse responses (1024 biquad sections), offset vertically.*

delay also becomes larger at that frequency. The pole radius will always be between $0 \leq r < 1$, and in the current approach, we select radii randomly within the limits

$$0.5 \leq r < \beta\,, \tag{5}$$

where r is the radius and β is computed from

$$\beta = \frac{\tau_g}{f_s}\,, \tag{6}$$

in which τ_g is the maximum allowable group delay and f_s is the sampling rate. We computed β to keep the group delay associated with any given pole below a perceptual threshold of $30\,\mathrm{ms}$ to minimize audible artifacts.

3.3. Filter Delay

When cascading the allpass filters, we can replace the unit delay, z^{-1}, with a longer delay, z^{-n}, in order to cause the phase to wrap around the unit circle n times faster. We randomly choose one, or small, prime or near prime delays so the cumulative effect of multiple delays do not align precisely.

Fig. 2 shows two example impulse responses generated by the above allpass filter cascade approach.

4. EVALUATION AND RESULTS

In order to evaluate the success of a decorrelation algorithm, we must consider the degree to which it achieves decorrelation, the amount it perceptually alters the original signal, and its computational efficiency. Correlation can be investigated mathematically through various applications of the cross-correlation function and by studying coherence. Perceptual similarity and the effects on localization are studied with empirical data from informal listening tests.

4.1. Cross-Correlation Metrics

The primary metric for evaluating the correlation between two discrete time signals is the cross-correlation function defined

$$\Phi_{xy}[l] = \sum_{m=0}^{N-1} x[m]y[l+m]\,, \tag{7}$$

where x and y have been zero padded to be the same length and N is the number of samples in one of the signals. This function, measures the similarity between two signals as they are slid by each other at increasing time lags, l. When $x = y$, this function is called the autocorrelation function and will exhibit a maximum peak at lag zero ($l = 0$) with a height equivalent to the signal power. In the cross-correlation case, the more similar signals x and y are, the larger a peak will be seen in the cross-correlation. Additionally, when signals are similar but offset in time from one another, the maximum peak will shift away from lag zero by the amount of delay between the signals. It is important to note that correlation is only valid if the signals have similar features and are worth comparing. Finding the correlation of two unrelated signals will likely show that the signals are decorrelated but the result is not meaningful.

When we apply our decorrelation kernels to a signal, we hope to spread the energy of the signal out in time by different amounts across frequencies. Since our decorrelation algorithm is a LTI system, we can directly compare impulse responses generated by our allpass filter cascade. Fig. 3 shows an example of the autocorrelation and cross-correlation of two allpass decorrelation kernels. As we would expect, the autocorrelation of both impulse responses are highly correlated at lag zero and are not well correlated at any other delay. The cross-correlation of the two impulse responses, on the other hand, have no single point where they line up, and have no large spikes in their correlation.

Because the decorrelation is frequency-dependent, and most listeners have two spatially separated ears, it is imperative that we consider correlations at lags other than zero. Moreover, as we slide signals by each other in the computation of the cross-correlation, it is only at lag zero where signals will overlap entirely and all samples will be contributing to the cross-correlation function. To address these issues, we propose a cross-correlogram, computed by chunking the input signals into windows and performing 50% overlap-add cross-correlations on the windowed signals. This metric allows us to easily display cross and autocorrelations in a manner similar to the interaural cross-correlation (IACC).

In order to compare auto and cross-correlations, we normalize the range of the autocorrelation by the signal power and the cross-correlation by

$$\frac{1}{2}\Big(\max(|\Phi_{xx}|) + \max|\Phi_{yy}|)\Big)\,. \tag{8}$$

This effectively scales the autocorrelation for both signals and the cross-correlation of the signals to each be within the range $[-1 \leq \Phi \leq 1]$, where -1 is perfect negative correlation and 1 is perfect correlation.

For visual clarity, let us consider a linear, sinusoidal chirp from $20\,\mathrm{Hz}$–$20\,\mathrm{kHz}$, seen in Fig. 4. Due to the periodicity of the signal increasing over time, there is a clear pattern imprinted in the autocorrelogram. Fig. 6 shows the auto and cross-correlogram of the present decorrelation technique as well as the noise and Kendall approaches. The distortion of the correlation patterns from the

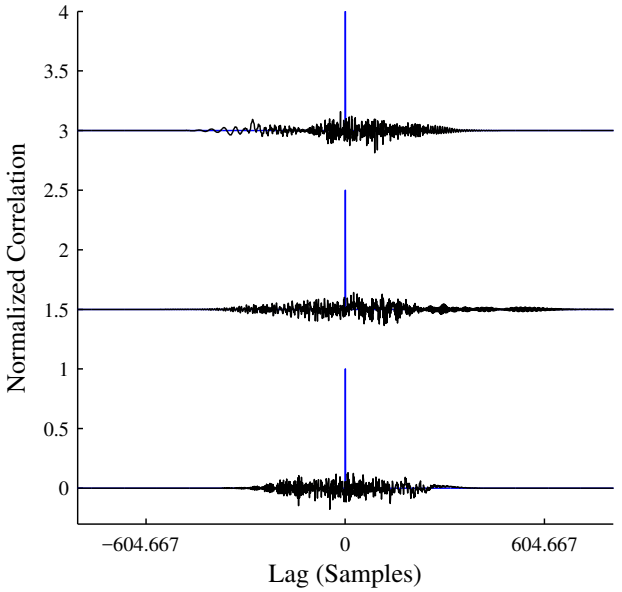


Figure 3: *Cross-correlation (black) and autocorrelation (blue) of three pairs of allpass cascade impulse responses (each 1024 biquads), offset vertically. Note the autocorrelation is a Dirac pulse.*

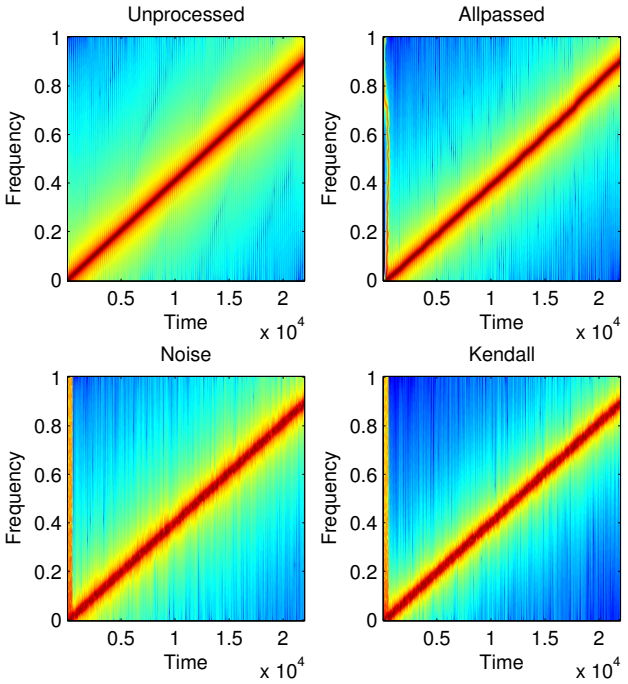


Figure 4: *Spectrogram of* 20 Hz–20 kHz *linear sinusoidal chirp depicting the unprocessed signal (a), allpass cascade (b), noise sequence (c) and Kendall sequence (d).*

auto-correlogram are clearly seen in the cross-correlogram plots. It is important to note that the present technique preserves original input signals better than the other techniques, and this can be seen in the auto-correlogram by the fact that it is much more similar to the unfiltered auto-correlogram seen in Fig. 5.

4.2. Coherence

Coherence is defined as the maximum value of the cross-correlation function. Although coherence distills the correlation sequence to a single number, it is not directly useful as it treats all frequencies the same and does not take the human auditory system into consideration. Instead, we present coherence values for octave bands, computed using a 4th order zero-phase Butterworth filterbank. Ideally, the average coherence values across frequencies would be near zero. Fig. 7 shows coherence per octave band for the allpass filter cascade approach as well as the noise and Kendall methods. Because of the perceptually informed filter design, our technique achieves low coherence above 60 Hz but does not perform well near DC.

4.3. Efficiency

Allpass filters are inherently IIR filters because of the feedback associated with having poles. The random allpass filter cascade is precomputed and the filter is applied at runtime. While the class of decorrelation filter described in this paper could simply be implemented using their difference equations, we did most of our computation by approximating the filter's impulse response by thresholding it once it dropped below −90dB. Below this threshold, we assume the filter's output would be below the noise floor and indistinguishable from roundoff error. We applied the filter impulse response as a convolution kernel (implemented with multiplication in the frequency domain). These impulse responses are quite short, having at most a few thousand taps and are suitable for real-time applications. For most applications, the computational requirements of this decorrelation algorithm will be negligibly small compared to other necessary signal processing constraints.

4.4. Listening Results

We performed informal listening tests to compare the allpass cascade decorrelation technique to the Kendall and noise approaches.[2] The authors and several colleagues familiar with audio and with normal hearing participated in a short listening test to evaluate the amount of decorrelation and the perceptual transparency of the decorrelation techniques. We presented critical listening material including castanets, glockenspiel, and male German speech from the EBU Sound Quality Assessment Material (SQAM) CD as well as recordings of unprocessed electric guitar and vocals from Tom's Dinner [22]. Each recording was presented over headphones and stereo studio monitors with listeners both in the sweet spot and off axis in a quiet room. The various conditions for the test included stereo presentations of the sound examples converted to mono and processed with 250, 500, 1000, 2000, and 3000 allpass biquad sections. As a control, we also processed the material using the Kendall and noise techniques with 512, 1024, and 2048 length sequences.

All three decorrelation algorithms achieved satisfactory decorrelation of the test stimuli. The noise and Kendall approaches decorrelated the signals slightly better than the allpass cascade, but the apparent source width of the signals for all three was significantly larger than the unprocessed "big" mono presentation. Like

[2] We intend to run formal listening tests in the future.

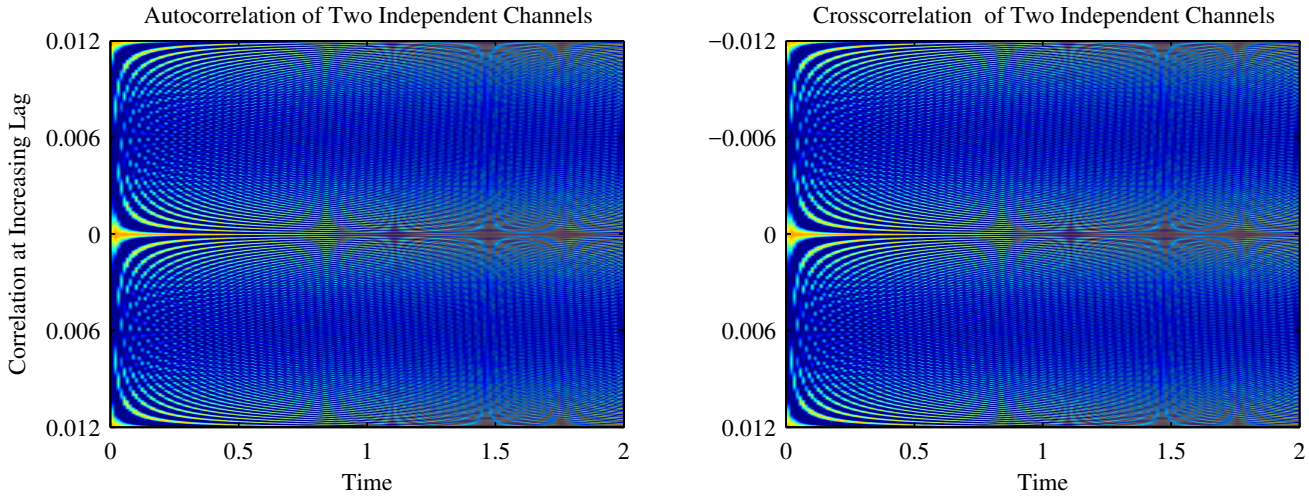


Figure 5: *Autocorrelelogram and cross-correlelogram for unprocessed* 20 Hz–20 kHz *linear sinusoidal chirp signal.*

when multiple violins play the same part in an orchestra and the similar sonic components fuse together to create the impression of a source with a larger apparent source width, the decorrelation algorithms broadened the spatial extent of the monophonic audio examples.

While all three algorithms effectively decorrelated the test material, the allpass cascade sounded much less colored than the other approaches. This is primarily due to the fact that the allpass filter maintain the desired flat frequency response while the Kendall and noise approaches do not. Presented in mono, the allpass cascade approach sounds much more similar to the original audio files and is therefore significantly more perceptually transparent. Presented in stereo, the Kendall and noise methods sound more similar to playing an audio signal combined with an inverted phase version of itself and the sonic coloration can be audible and unpleasant compared to the allpass filter cascade.

The optimal number of biquad sections for the allpass filter technique varies with the source material. In general, using more biquad sections reduces the correlation between channels. However, strong transients such as the onsets of the castanets and glockenspiel can suffer from chirp-like artifacts. In our tests, we found that the castanets were effected when more than 500 biquads were used and the glockenspiel at 1000. All the other material sufficiently masked the displeasing artifacts beyond 2000 biquad sections. Additionally, since the decorrelation filters are randomly generated, their effectiveness varies based on the frequency content of the material and the specific decorrelation filters used.

5. CONCLUSIONS

In this paper, we have described an approach for signal decorrelation that uses cascades of allpass filter biquad sections. Allpass filters are useful for decorrelation because they maintain a flat magnitude frequency response while adding frequency specific phase delays. We have introduced several methods for displaying and evaluating decorrelation, mainly the cross-correlogram, octave band coherence measure, and listening tests. While techniques like Kendall's "allpass filter" and convolution with noise can achieve a higher amount statistical independence, the present algorithm introduces less distortion to the original signal and sounds better. This decorrelation method is well suited for real-time applications for spatializtion, auralization, and upmixing.

In the current approach, the pole locations for the allpass filters are chosen in a naïve way and nothing prevents multiple decorrelation filters from having poles placed in the same locations. In the future, we would like to design the filters in a way that we can achieve the same or greater amounts of decorrelation with a shorter number of biquad sections by reducing the amount of competition between independent decorrelation kernels. Furthermore, the high transient content of sound files like the castanet recording expose the necessity of transient detection in decorrelation algorithms.

6. REFERENCES

[1] Garry Kendall, "The effects of multi-channel signal decorrelation in audio reproduction," in *Proceedings of the International Computer Music Conference*, 1994.

[2] Garry Kendall, "The decorrelation of audio signals and its impact of spatial imagery," *Computer Music Journal*, vol. 19, no. 4, pp. 71–87, Winter 1995.

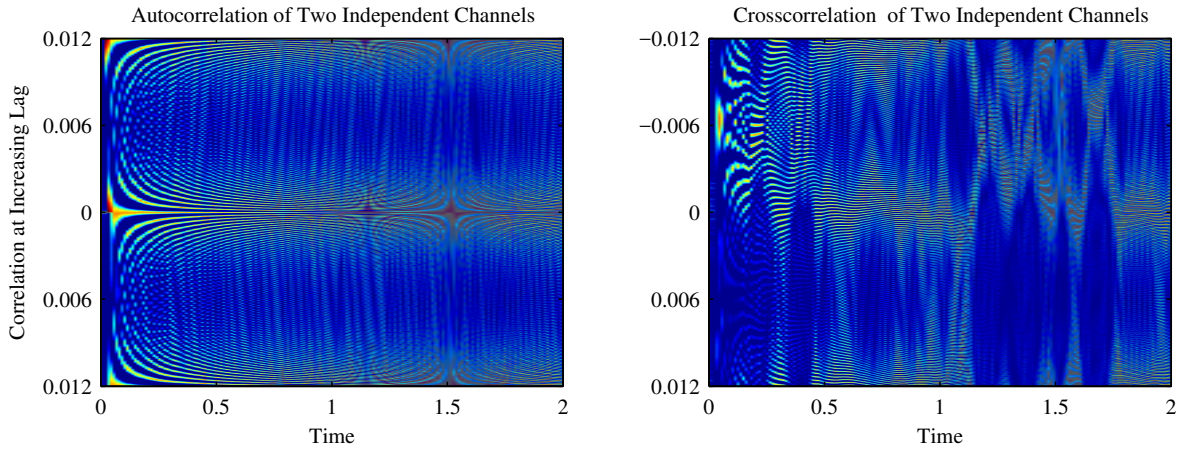
[3] Maurice Boueri and Chris Kyirakakis, "Audio signal decorrelation based on a critical band approach," in *Proceedings of the 117 Audio Engineering Society Convention*, 2004.

[4] Ross Penniman, "A general-purpose decorrelation algorithm with transient fidelity," in *Proceedings of the 137th Audio Engineering Society Convention*, 2014.

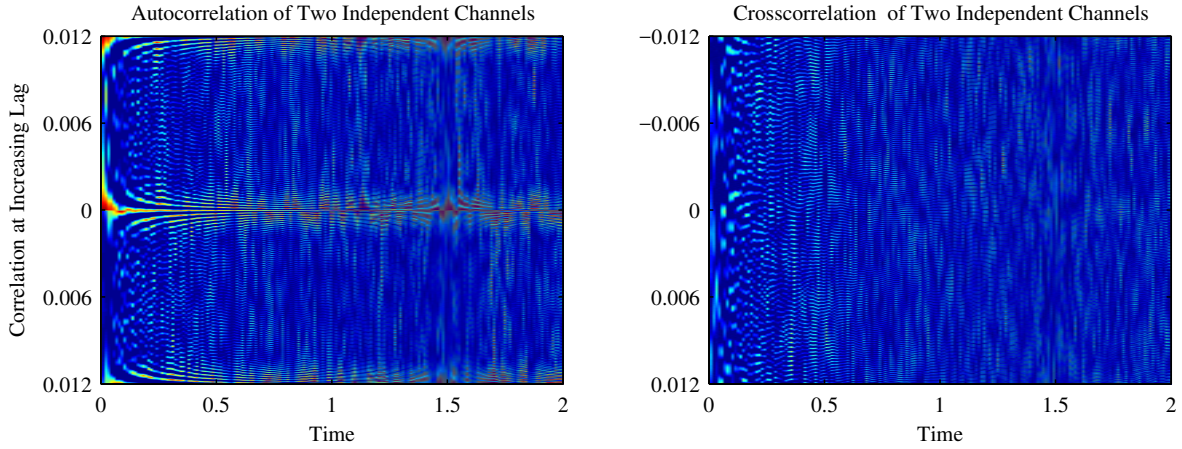
[5] Guillaume Potard and Ian Burnett, "Decorrelation techniques for the rendering of apparent sound source widths in 3d audio displays," in *Proceedings of the 7th International Conference on Digital Audio Effects*, 2004.

[6] Andres Cabrera and Garry Kendall, "Multichannel control of spatial extent through sinusoidal partial modulation (spm)," in *Proceedings of the Sound and Music Computing Conference*, 2012.

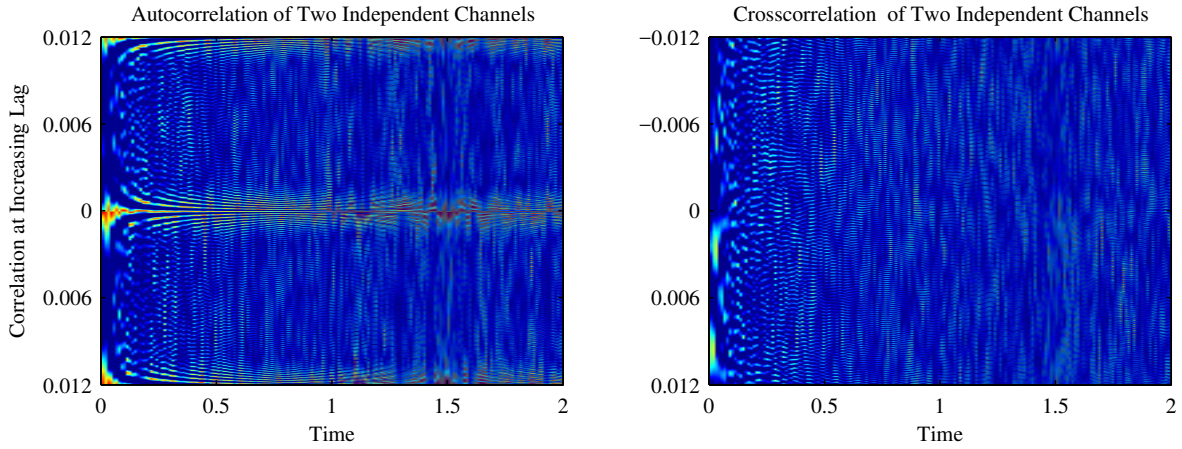
[7] Andre Cabrera, *Control of Source Width in Multichannel Reproduction Through Sinusoidal Modeling*, Ph.D. thesis, Queen's University Belfast, 2012.



(a) *1024 allpass biquad sections*



(b) *1024 sample length noise sequence*



(c) *1024 sample length Kendall sequence*

Figure 6: *Auto-correlelograms (left) and cross-correlelograms (right) for* 20 Hz–20 kHz *linear sinusoidal chirp signal processed with allpass filter cascades (6a), noise sequences (6b), and Kendall filters (6c).*

[8] Mark Gardner, "Image fusion, broadening, and displacement in sound location," *Journal of the Acoustical Society of America*, vol. 46, no. 339, pp. 339–349, January 1969.

[9] Manfred Schroeder, "Natural sounding artificial reverberation," *Journal of the Audio Engineering Society*, vol. 10, no. 3, pp. 219–223, 1962.

[10] Frank Baumgarte and Christof Faller, "Binaural cue coding part i: Psychoacoustic fundamentals and design principles," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 6, pp. 509–519, November 2003.

[11] Frank Baumgarte and Christof Faller, "Binaural cue coding part ii: Psychoacoustic fundamentals and design principles," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 6, pp. 520–531, November 2003.

[12] Jonas Engdegård, Heiko Purnhagen, Jonas Rödén, and Lars Liljeryd, "Synthetic ambience in parametric stereo coding," in *Proceedings of the 116th Audio Engineering Society Convention*, May 2004.

[13] Jürgen Herre, Kristofer Kjörling, Jeroen Breebaart, Christof Faller, Sascha Disch, Heiko Purnhagen, Jeroen Koppens, Johannes Hilpert, Jonas Rödén, Werner Oomen, Karsten Linzmeier, and Kok Seng Chong, "Mpeg surround: The iso/mpeg standard for efficient and compatible multichannel audio coding," *Journal of the Audio Engineering Society*, vol. 11, no. 56, pp. 932–955, November 2008.

[14] Jean-Marc Valin, "Channel decorrelation for stereo acoustic echo cancellation in high-quality audio communication," in *Proceedings of Workshop on the Internet, Telecommunications and Signal Processing*, 2006.

[15] Franz Zotter, Matthias Frank, Georgios Marentakis, and Alois Sontacchi, "Phantom source widening with deterministic frequency dependent time delays," in *Proceedings of the 14th International Conference on Digital Audio Effects*, 2011.

[16] Sebastian Kraft and Udo Zölzer, "Stereo signal separation and upmixing by mid-side decomposition in the frequency-domain," in *Proceedings of the 18th International Conference on Digital Audio Effects*, 2015.

[17] Achim Kuntz, Sascha Disch, Tom Bäckström, and Julien Robilliard, "The transient steering decorrelator tool in the upcoming mpeg unified speech and audio coding standard," in *Proceedings of the 131st Audio Engineering Society Convention*, Oct 2011.

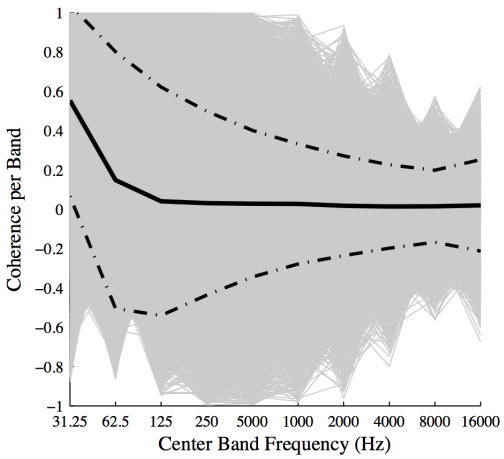
[18] Mathieu Lagrange, Luis Gustavo Martins, and George Tzanetakis, "Semi-automatic mono to stereo up-mixing using sound source formation," in *Proceedings of the 122nd Audio Engineering Society Convention*, May 2007.

[19] Robert Orban, "A rational technique for synthesizing pseudo-stereo from monophonic sources," *Journal of the Audio Engineering Society*, vol. 18, no. 2, pp. 157–164, 1970.

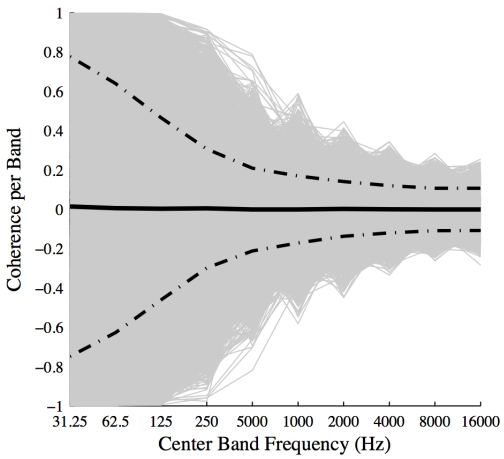
[20] Robert Orban, "Further thoughts on "a rational technique for synthesizing pseudo-stereo from monophonic sources"," *Journal of the Audio Engineering Society*, vol. 18, no. 4, pp. 443, 444, 1970.

[21] Jens Blauert, *Spatial Hearing: The Psychophysics of Human Sound Localization.*, The MIT Press, Cambridge MA, revised edition edition, 1996.

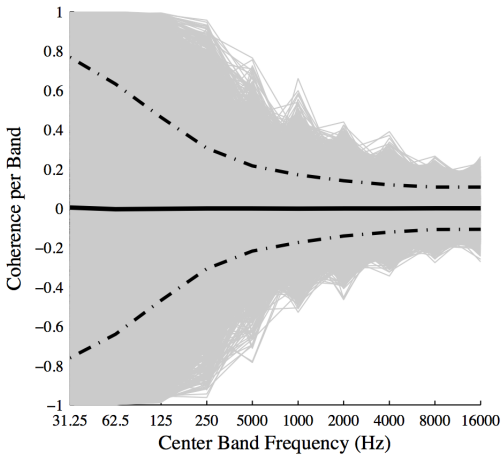
[22] "Ebu tech 3253 companion cd sound quality assessment material recordings for subjective tests (sqam)," `https://tech.ebu.ch/publications/sqamcd`, October 2008.



(a) *1024 allpass biquad sections*



(b) *1024 sample length noise sequence*



(c) *1024 sample length Kendall sequence*

Figure 7: *Coherence in octave bands for allpass cascade (7a), noise sequence (7b), and Kendall filter (7c). The dark lines show the means, dotted lines the standard deviation, and gray lines individual one-vs-one coherences (1000 pairs).*

COMPLEXITY SCALING OF AUDIO ALGORITHMS: PARAMETRIZING THE MPEG ADVANCED AUDIO CODING RATE-DISTORTION LOOP

Pablo Delgado, Markus Lohwasser

Fraunhofer IIS
Realtime Audio Processing Group
Erlangen, Germany
pablo.delgado@iis.fraunhofer.de

ABSTRACT

Implementations of audio algorithms on embedded devices are required to consume minimal memory and processing power. Such applications can usually tolerate numerical imprecisions (distortion) as long as the resulting perceived quality is not degraded. By taking advantage of this error-tolerant nature the algorithmic complexity can be reduced greatly. In the context of real-time audio coding, these algorithms can benefit from parametrization to adapt rate-distortion-complexity (R-D-C) trade-offs. We propose a modification to the rate-distortion loop in the quantization and coding stage of a fixed-point implementation of the Advanced Audio Coding (AAC) encoder to include complexity scaling. This parametrization could allow the control of algorithmic complexity through instantaneous workload measurements using the target processor's task scheduler to better assign processing resources. Results show that this framework can be tuned to reduce a significant amount of the additional workload caused by the rate-distortion loop while remaining perceptually equivalent to the full-complexity version. Additionally, the modification allows a graceful degradation when transparency cannot be met due to limited computational capabilities.

1. INTRODUCTION

Development of low-power, fast implementations of perceptual audio codecs is always challenging due to the complex nature of the signal processing involved. In addition, the necessity of operating these algorithms on computationally-restricted architectures for mass production and low delay requirements present an additional constrain on workload. The goal in this case is to optimize execution speed and power consumption while remaining perceptually indistinguishable from a possible reference implementation on a more powerful platform. If this goal cannot be met for complexity reasons, the degradation in quality should be gradual according to perceptual rules.

In order to achieve this, a major task consists on porting a floating-point code to a fixed-point version suited for low-power platforms. It has been already shown that the same overall perceived quality (according to listening tests and objective measurements) of AAC and mp3 codecs can be preserved even with the precision loss associated with porting code from floating-point arithmetic to a 32-bit, fixed-point representation [1]. These results were achieved by using proper scaling of audio signal energies and masking thresholds at various points in the psychoacoustic model, adapting a fractional arithmetic to pure integer processors and using a logarithmic representation of signals in order to transform costly division operations to subtractions, and to attain a suitable mapping of the dynamic range.

On a more general plane, there have been recent discussions on the implementation of multimedia processing algorithms and the role of fixed complexity boundaries on error-tolerant applications [2]. Due to limitations of silicon CMOS technology in providing further increase in speed at acceptable fault-rate and energy-dissipation, current research suggests a closer look at multimedia applications in terms of precision and resilience requirements [3]. Particularly, parametric adaptation of rate-distortion-complexity curves [4] of audio coding algorithms at different stages can greatly help in optimally exploiting the capabilities of each target platform. Moreover, this parametrization also accelerates the process of achieving optimal performance on new processors, independently of hardware optimizations.

The MPEG 2/4 AAC codec is present among the most prominent examples of perceptual audio coding (PAC) technologies. PAC techniques usually convert the time domain input samples into a frequency domain representation in order to remove redundancies and irrelevancies of the signal for efficient transmission and/or storage. In doing so, the coding noise power is adapted to a hearing threshold provided by a human perceptual model [5] in a way that the noise is as less disturbing as possible. Some latest examples include the Unified Speech and Audio Coding (USAC) [6] and the recent MPEG-H Audio [7] standards. Complexity and workload are of particular importance for encoding and decoding audio in real-time due to their application on low-power mobile devices. Versions of the AAC codec specially fit for this task are AAC Low Delay (AAC-LD) [8] and AAC Enhanced Low Delay (AAC-ELD) and AAC-ELD version 2 (AAC-ELDv2) [9] and 3GPP Enhanced Voice Service (EVS) [10]. These variations feature shorter transform lengths in order to accommodate low delay requirements and eliminate or greatly limit bit reservoir techniques for maintaining a constant time delay ([11],[12]), the trade-off being loss of frequency resolution and an increase in workload for a fixed sampling rate.

The AAC encoder carries the most algorithmically complex modules of the codec [1]. Particularly, the psychoacoustic model block including the time-to-frequency transform and the quantization and coding block are the most demanding in terms of computations [13]. Since, for MPEG codecs, only the AAC decoder is standardized, modifications on the AAC encoder to achieve better performance are possible as long as the produced encoded output produces a valid bitstream. This work will focus on the quantization block by studying a possible parametrization of the internal algorithm to scale its complexity according to the available processing power.

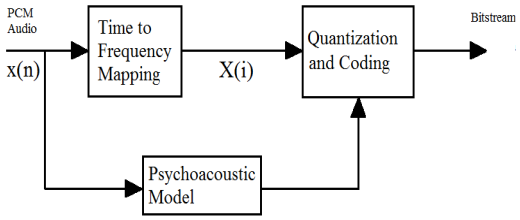


Figure 1: *Simplified block diagram of the AAC Encoder*

1.1. Comparison to previous approaches

Although previous efforts were made in order to identify and optimize the most sensitive steps in terms complexity of the quantization stage ([14], [15]), the proposed algorithms offer efficient implementations with varying compromises in quality, but offer no parametrization of complexity factors except for setting a maximum limit to the number of iterations for rate and distortion loops.

In [14] the authors propose a scheme based on an estimation of the non-uniform quantization stage of the AAC encoder in order to implement a loop-less bit-allocation. This approach results in a significant reduction in complexity (reported 80% to 90%) to the expense of a significant reduction in audio quality. Nevertheless, it is not clear how this approach can benefit from parametrization in cases where the same algorithm is implemented on more powerful architectures and a better audio quality can be allowed.

In this regard, the authors of [15] also propose an hybrid approach where loop-less bit-allocation is used as a starting point for a loop-based method. They claim at least a factor of 10 in complexity reduction with respect to the standard approach to bit-allocation in AAC by decoupling rate and distortion loops. Since only AAC audio decoders -and not encoders- are standardized, the proposed method in the AAC standard is only provided as a reference and is far from meeting the complexity requirements of a low-power and/or real-time implementation. The encoding approach can vary significantly as long as the produced bit-streams are valid. Consequently, many of today's available solutions -including the implementation of AAC used for our work- also shows similar performance with respect to this approach [12], [1].

To the best of our knowledge, the transition between a loop-less approach and a higher quality loop-based bit-allocation is not studied previous works. An equivalent method to that in [14] based on estimation of the quantization noise is already present in our implementation of the AAC encoder in addition to a loop-based bit-allocation method for higher quality. Our work proposes a closer look to the quality-complexity trade-off and a solution for a smoother transition between the two modes of operation.

2. SYSTEM OVERVIEW

The quantization and coding stage of the AAC encoder quantizes the spectral data provided by a Modified Discrete Cosine Transform (MDCT) of a Pulse Code Modulated (PCM) audio signal, in a way that the quantization noise satisfies the demands of the psychoacoustic model [5](Figure 1). On the other hand, the number of bits needed to quantize the spectrum must be below a certain limit, typically the average number of available bits for a block of audio data. The way in which this trade-off is approached is the core of the coding strategy. This strategy is usually carried out in an Analysis-by-Synthesis manner, in which the resulting quantized and coded spectral lines are evaluated and re-quantized until an optimal solution within a range is reached.

The AAC quantization process involves the gain adjustment of groups of spectral values (scale factor bands) which are then processed by a power-law quantizer and a Huffman Coder. The scale factor amplification is used to take advantage of the non-uniform distribution of the coding noise provided by the power-law quantizer to better accommodate perceptual requirements [16]. The amplification factors (scale factors or scaling factors) applied to the scale factor bands are differentially coded also using Huffman Coding. The goal of the quantization stage is to determine the set of scaling factors that better accommodate the psychoacoustic model requirements and the bit availability at a certain rate.

The scale factor amplification and non-uniform quantization is carried out according to

$$\bar{X}(i) = sign(X(i)) \cdot nint\left(\left(\frac{|X(i)|}{\sqrt[4]{2}^{q_k}}\right)^{0.75} - 0.0946\right) \quad (1)$$

where $\bar{X}(i)$ is the value of the quantized *i-th* spectral line, $X(i)$ the *i-th* input MDCT spectral line and q_k the scale factor (amplification factor) associated to the *k-th* scale factor band. The span of $i : 0 < i < I_k - 1$ determines the amount of spectral lines per scale factor band (bandwidth), where I_k ranges from 4 to 52 for each *k-th* scale factor band. Expressions $sign()$ and $nint()$ represent the sign of the argument and the rounding operation to the closest integer respectively [17]. The power, root and division operations present in (1) conform one of the main bottlenecks on the computational load for low power devices [1].

The Fraunhofer AAC Encoder [18] features two different quality modes for the quantization stage: Fast Quality, where an open-loop solution is used to set each scale factor q_k based on *a priori* estimations of the non-uniform quantizer noise and the hearing thresholds provided by the psychoacoustic model, and High Quality, where the scale factors are set by refining the open loop estimation by Analysis-by-Synthesis (AbS) using two stages of iterative search. The AbS method can be seen as an AAC distortion-rate loop [15] and is described below.

2.1. Scale factor band optimization

The optimization method is based on a simple iterative search in the neighborhood of the initial scale factor value, in which a suitable scaling factor q_k that minimizes the quantity

$$D_{sfb}(q_k) = \sum_{\forall i \in \text{sfb}} |X(i) - \tilde{X}(i, q_k)|^2 \quad (2)$$

in each scale factor band (sfb) is chosen. The values of $\tilde{X}(i)$ represent the reconstructed spectral values after quantization using the inverse of (1). The quantity D_{sfb} given by equation (2) is defined as the distortion (or noise power) of each scale factor band caused by the quantization process and is used as a cost function for a series of iterative searches. This minimization is nevertheless restricted to keeping the noise-to-mask ratio (*nmr*) [19] just below a certain limit, and not much lower. If the *nmr* is too low, bits would be wasted in coding a band whose coding noise is already complying with the psychoacoustic rules. The method also considers this case when the starting distortion value is too low and tries to adapt it in

order to save bits in coding. After a first calculation of distortion for the first estimation of q_k, the algorithm has two main branches:

First branch (adjust distortion): If the noise-to-mask ratio determined from the distortion is more than 1.25 (estimated experimentally), then try to improve it by searching for a q_k that minimizes D_{sfb} within $\mathbf{q_k} = [q_k - \nu_l, \ldots, q_k - 1, q_k, q_k + 1, \ldots, q_k + \nu_h]$, where ν_l, ν_h are the lower and higher limits of the search vector respectively.

Second Branch (adjust bitrate): The power-law quantizer of equation (1) provides coarser quantization steps for greater scale factors, coarser step sizes will need less bits to be coded. If the calculated *nmr* is less or equal to 1.25, the scale factor used can be increased in order to spare some bits and still comply with quantization noise masking rules. The search vector becomes then $\mathbf{q_k} = [q_k, q_k + 1, \ldots, \phi_h]$, where ϕ_h is the higher limit and the search stops when the resulting *nmr* is greater than 1.25.

Due to the fact that this iterative search requires the repeated re-quantization of the spectral lines $X(i)$ -via (1) and its inverse quantization counterpart [20]- to get $\tilde{X}(i)$ for each value of q_k the choice of ν_l, ν_h and ϕ_h significantly impacts on the computational complexity.

2.2. Inter-band scale factor assimilation

Once the first set of scale factors has been determined, further iterative searches try to decrease the range of scale factor values across the scale factor bands in order to save bits on differential encoding. Given a set of scale factor bands, their difference in value is reduced and also adjusted to produce the smaller value of (2). Then, for further coding efficiency, additional smoothing along sets of scale factor bands is applied among these previously selected scale factors with the same criteria. In all cases, the same complexity considerations apply and the number of re-quantization operations grow further.

3. PARAMETRIZATION OF THE RATE-DISTORTION LOOP

Prior AAC encoder profiling [18] with the AbS feature switched on showed that the routines of distortion calculation, quantization and inverse quantization were the most cycle-demanding because of the many routine calls to (1) and (2). Distortion calculation and inverse quantization are only performed when the AbS mode is active. The most significant increase in cycle consumption takes place with the switch from no AbS to active AbS at around 40-60% extra cycles depending on the target platform.

3.1. Modification of the distortion calculation

As mentioned above, the evaluation of (2) has many calls throughout the code. It is worth noting that the exact absolute value of D_{sfb} is not important. A certain amount of error is permitted as long as the search algorithm ends at the right value of q_k (i.e. the scale factor that adjusts D_{sfb} to the better trade-off) within the search vector remains guaranteed. Based on this remark, the conditions on the calculation of (2) can be relaxed in favor of lesser computational cost.

3.1.1. Adaptive Threshold

One approach for reducing the amount of re-quantization can be calculating the distortion for a subset of lines on each scale factor band. The whole spectrum needs to be quantized at the end with the right scaling factor q_k, but for the purpose of the iterative search, only a representative portion of the spectrum can be used for estimating (2). This can greatly reduce the computational burden.

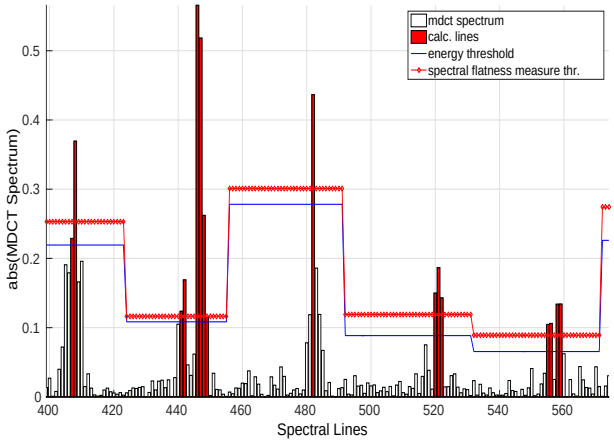


Figure 2: *Extract of the absolute value of the MDCT spectrum of a sample input signal. Lines of the spectrum that are considered for Analysis-by-Synthesis recalculation according to equation (5) ($j \in K_\tau$) and adaptive threshold τ_k (continuous line) for each sfb ($g_h = 0.5$) are coloured in dark red. Threshold based on classical spectral flatness measure [19] for reference (diamond line).*

Since the power-law quantizer aims to evenly distribute the SNR across the dynamic range of the signal by applying a companding function to the quantization steps [16], the spectral lines with higher relative energy will contribute more to the round-off error than the lines with lesser energy.

It is well known that tonal signals concentrate the majority of their energy on narrow portions of their frequency spectrum. The lines corresponding to tonal portions (higher relative energy) will be the ones which contribute the most to the calculation of (2) within a scale factor band. It is therefore advantageous to rely on spectral flatness measures or tonality indices per band in order to identify these sections. The commonly used spectral flatness measure is the ratio of the geometric mean μ_g and the arithmetic mean μ_a of the power spectral density of each scale factor band k, $\text{SFM}_k = \frac{\mu_g}{\mu_a}$ [19]. A value of the SFM close to 1 means a flat "non-tonal" spectrum, whereas a value close to 0 means the presence of strong harmonic components corresponding to a tonal signal.

The selection of the subset of lines can be carried out with the aid of an adaptive threshold. This threshold is a fraction of the maximum value of $X(i)$ for a specific spectral band and is based on the estimated tonality for that particular region. The threshold τ_k is defined as:

$$\tau_k = g_h \cdot \max_{i \in \text{sfb}}[X(i)] \cdot (1 - \frac{nl_k}{I_k}) \tag{3}$$

where I_k is the scale factor band width in spectral lines and nl_k is the number of lines in each scale factor band that will effectively be above some minimum quantization error value, marked as "relevant lines":

$$nl_k = \frac{\sum_{i=0}^{I_k-1} \sqrt{|X(i)|}}{(\frac{e_k}{I_k})^{0.25}} \quad (4)$$

where e_k is the total energy of the spectral band k.

The encoder already calculates nl_k for other purposes and its re-utilization saves processor cycles. For the case of the power-law quantizer of equation (1), an estimation of the quantization error can be made that depends on the quantizer step size q_k and a form factor of the spectral band expressed by the ratio between the geometric mean and the arithmetic mean [5] [14].

The spectral lines with amplitudes below the threshold τ_k will not take part on the distortion calculation. The more tonal the scale factor band, the higher the threshold (Figure 2). This means that only lines corresponding to the strongest harmonics will be calculated. From the figure it can also be seen that, -albeit with different scaling- both classical spectral flatness measures SFM_k and τ_k given by equation (3) are equivalent for this purpose. The proposed tonality measure is used instead of classical approaches because most of the elements of equation (3) are already calculated for other purposes within the encoder. As a consequence some processing power is saved by not estimating tonality again in a different way.

The parameter g_h is an ad-hoc correction gain factor, hand-tuned in a way that the resulted calculated lines after the algorithm approaches the amount predicted by the number of relevant lines nl_k, restricted to the condition that $g_h \cdot (1 - \frac{nl_k}{I_k}) < 1$ so that at least one spectral line is calculated per sfb . It must be noted that other measurements of tonality can be used according to encoder implementation and the available data, and the tuning of the threshold gain g_h can accordingly vary.

3.1.2. Reformulation

The computational cost of calculating (2) for all the spectral lines in narrow bands (few frequency bins) becomes comparable to the one derived from using the threshold implementation. It was determined experimentally that the threshold implementation is more suitable if only used for spectral bands that have more than $I_k = 12$ lines because there is also a cost to implementing the threshold decision within the loop that scans every spectral line. Besides, skipping the calculation of some spectral bands for an already small set can lead to significant errors in (2) that will affect the convergence. Equation (2) then is reformulated as:

$$D_{\text{sfb}}(q_k) = \begin{cases} \sum_{\forall j \in K_\tau} |X(j) - \tilde{X}(j, q_k)|^2 & \text{if } I_k \geq 12 \\ \sum_{\forall i \text{sfb}} |X(i) - \tilde{X}(i, q_k)|^2 & \text{if } I_k < 12 \end{cases} \quad (5)$$

where $K_\tau \in$ sfb is the subset of spectral lines for which $X(i) > \tau_k$.

3.2. Taking advantage of signal stationarity

Another approach for avoiding the excessive re-quantization can be taking advantage of the relative stationarity of audio signals [19]. Once a given set of optimal scaling factors has been determined for an audio time frame, it is possible that the same set of optimal scale factors is already close to optimal for the next frame, given that the signal does not change significantly in that frequency region (Figure 3). Furthermore, this same principle can be applied

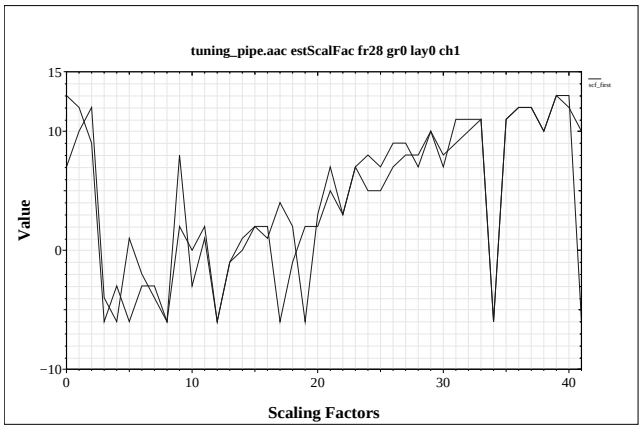


Figure 3: *Sets of scaling factors for coding the spectrum of a tuning pipe recording. Two consecutive audio frames are superimposed. The difference between frames greatly diminishes around the scale factor 28 and up (around 5 kHz and up at a frame size of 1024 and sampling rate of 48 kHz).*

to stereo signals that do not feature a significant inter-channel difference [21]. For two consecutive audio time or channel frames f and $f + 1$, a preset threshold τ_s can be set to the condition:

$$|q_k^{(f+1)} - q_k^{(f)}| < \tau_s \quad (6)$$

for q_k factors of the same scale factor band k (frequency region). Only when the scale factor difference between two frames is bigger than the threshold will the AbS procedure described in section 2 take place. Otherwise, the scaling factor is considered close enough to optimal and the already quantized spectrum can be used. This method can be further refined to implement higher order temporal smoothing techniques, to the expense of additional memory usage: increasing the order F of the smoothing filter requires storing the complete set of scaling factors for each of the F previous frames.

4. RESULTS

This section encompasses results regarding trade-offs of the parameter tuning, complexity measurements and quality measurements.

4.1. Parameter Tuning

The parameters g_h and τ_s introduced in Sections 3.1 and 3.2 have limiting values. On one hand, the values can be set very low, so they do not have any influence on the re-quantization simplification and the complexity remains the same. In fact, these parameters can be set to generate a bit-exact stream with the reference version if set to $g_h = 0$, all the lines will then take part in the re-quantization of equation (5). By setting $\tau_s = 0$, the AbS routine takes place even if the scale factors remain the same within two consecutive frames.

On the other hand, if these parameters are set too high, the simplifications on the distortion calculation account for a coarse -and not so frequent- approximation of (2). This can result in audible artefacts with respect to the full AbS version, where all spectral lines are re-quantized in every audio frame. Figure 4 shows the

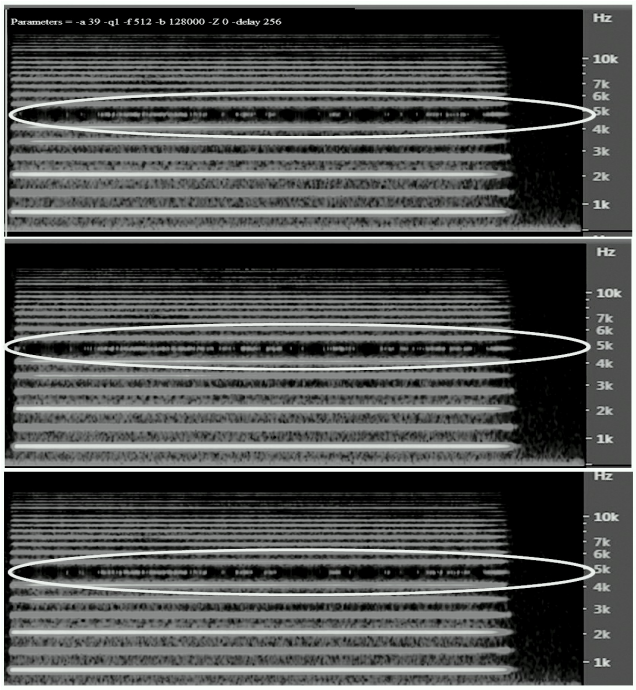


Figure 4: *Spectrogram of a coded tuning pipe mono audio signal at 48 kHz and 48 kbps with AAC-ELD. Above: Unmodified Analysis-by-Synthesis scale factor estimation. Middle: Scale factor smoothing in time (section 3.2) with $\tau_s = 3$. Below: Scale factor smoothing in time with $\tau_s = 1$.*

case where the time smoothing of section 3.2 is implemented with a value of τ_s that is too high (middle spectrogram). In comparison with the full AbS reference (upper spectrogram), more discontinuities can be seen in the spectral line of 5 kHz, which account for audible artefacts. This shows that the stationarity assumption can be overdone and inhibiting the AbS procedure can affect how the encoder works in small details. The lower spectrogram shows an optimal value of $\tau_s = 1$, where a lower complexity is still reached without affecting the sound quality.

Similar assumptions and procedures can be made with the threshold implementation of equation (5), where the optimal value was deemed to be around $g_h = 0.7$

4.2. Complexity

Table 1 shows two representative cases of complexity measurements for the different proposed modifications of section 3. The platforms tested are based on an ARM Cortex-A57 64-bit core and a Texas Instruments C6424 32-bit processor. All MHz values include all memory wait state and cache miss cycles and have been obtained with activated data and program cache. The different encoders do not contain any particular optimization that favors one particular version over the other. The selected encoding variant was AAC-ELD due to the shorter audio processing block length and therefore with tighter complexity requirements [9], [12]. A stereo file at 48 kHz sampling rate was encoded with a frame size (block length) of 512 samples. Encoder tools not relevant to the measurement were turned off in order to minimize the influence of other modules on the measurements. As a reference for a lower bound, workload was also measured for the encoder when no re-quantization routine is used (Condition 5). The frame smoothing procedure of equation (6) was used with a parameter of $\tau_s = 1$ and $g_h = 0.7$ for the adaptive threshold method.

Texas Instruments figures were obtained on a TMS320C6424 EVM evaluation board at a clock speed of 600 MHz.

ARM Cortex-A57 figures were obtained by running the software on a NVIDIA Jetson TX1 module running Linux Ubuntu (GNU/Linux 3.10.67 Kernel aarch64). We considered as the true reproducible value the minimum workload number for processing an audio frame during 10 consecutive encoder executions in order to filter out any influence of the operating system layer. Only one core was active and all power scaling functions were deactivated. The processor core at full performance was running at a CPU frequency of 1.91 GHz.

Indeed, Table 1 shows a general workload reduction for both architectures using one of the two methods, or a combination of both. Relative improvements might change in the presence of architecture-specific optimizations. For example, the significant jump in complexity between condition 5 and the other four conditions for the TI board is due to the poor handling the loops present on the AbS algorithm when no pragma directives are available [22]. As it will be shown in the next section, further reduction in workload can be achieved for mono files, where the encoder does not make use of joint stereo coding techniques [17] that might mitigate the workload reduction impact.

4.2.1. Complexity Scaling

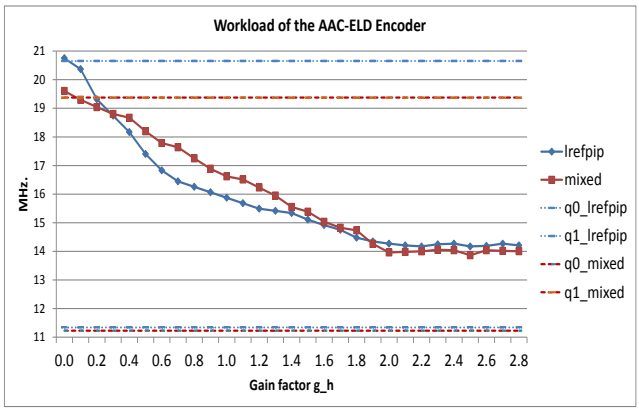


Figure 5: *Parametric workload curves for Cortex-A57 showing the influence of parameter g_h on encoding complexity. Two mono files (lrefpip: tuning pipe, mixed: castanets and guitars) at 48 kHz encoded with a block length of 512 samples. Terms $q0$ and $q1$ used to denote workload figures for no re-quantization at all and full unmodified re-quantization respectively.*

To further illustrate the influence of the proposed modifications on complexity, figure 5 shows parametric workload measurements performed on two different mono signals encoded with the same configurations as before. In contrast to the previous measurements, some compiler optimizations were active (-o3 switch on aarch64-linux-gnu-gcc 4.8.3) in order to provide insight on a real usage case scenario. The signal marked as *lrefpip* corresponds to the recording of a tuning pipe with a predominantly tonal structure, whereas the signal marked as *mixed* contains the recording of

Table 1: *Encoder Workload Measurements (Unoptimized).*

Condition	Encoder variant	Workload (MHz.)	Workload (MHz.)
		ARM Cortex-A57	TI C6424
1	Unmodified Re-quantization	41	74
2	Frame Smoothing	36	67
3	Adaptive Threshold	33	64
4	Fr. Smooth. and Adapt. Thr.	32	62
5	No Re-quantization	30	45

guitars and castanets, showing in this case an hybrid transient-like and tonal structure.

For $g_h = 0$ it can be seen that the workload corresponds to the full re-quantization case where all the lines are calculated. Already the full re-quantization shows varying complexity according to the signal (q1 lines). In the case of the hybrid signal, the content allows some relaxation on the iterative search for finding the minimum distortion, given that part of the coding noise will be masked by the loud transients. Conversely, the tonal structure of *lrefpip* and its relative quietness imposes a stricter requirement for reaching the minimal distortion on each scale factor band, therefore taking more time per frame and increasing workload. There is a small overhead for $g_h = 0$ with respect to the workload of the unmodified encoder on full re-quantization due to the calculation of τ_k.

As g_h increases, the number of spectral lines calculated for each iteration of the AbS algorithm diminishes according to (5). The signal characteristics also influence the way the total complexity is reduced: the encoding of the tonal signal shows a steeper reduction in workload when g_h is increased. This is because the modified re-quantization algorithm presented in section 3.1 only calculates the significant harmonics in each scale factor band. Such harmonics account for a few spectral lines per band that contain the most energy, and therefore convergence to the fitting scaling factor is guaranteed within a few calculations. On the contrary, the selectivity of the algorithm decreases for signals with a greater number of non tonal spectral bands, as is the case of the mixed signal. Most of the lines need to be calculated for non-tonal regions of the spectrum, a smaller workload reduction is achieved when increasing g_h.

The workload reduction reaches a limit for higher values of g_h. According to (5), only scale factor bands that have less than 12 spectral lines will be calculated in its entirety, and the rest of the bands will only be re-quantized with only one line per band in each iteration of the AbS search. Accordingly, this accounts for an offset with respect to the state where no re-quantization at all -at any of the bands- takes place (marked as q0 lines). In addition, even if the AbS iterative search is carried out re-quantizing as few lines as possible in each scale factor band, the whole spectrum needs to be re-quantized with the best suitable scale factors in the end. This final re-quantization of each band with its best scaling factor also contributes to the difference in workload with respect to the q0 operating points of the encoder for each signal.

4.3. Objective and Subjective Quality Grading

In order to evaluate the audio quality of the proposed parametrization, a Perceptual Evaluation of Audio Quality (PEAQ) [23] test automatically rated audio items from a database of 278 entries - music and speech recordings- coded with the different proposed encoder variants ($\tau_s = 1$ and two extreme values for g_h) and encoding at various bit rates and decoded using the same reference decoder.

Table 2: *Total average PEAQ degradation.*

Bitrate	Variant			
/chan	FS	AT ($g_h = 0.2$)	AT ($g_h = 1.9$)	NR (q0)
24000	0.032	0.007	0.014	0.108
32000	0.017	0.026	0.031	0.118
48000	0.180	0.010	0.098	0.344
96000	0.031	0.020	0.050	0.146

Table 3: *Tonal signals average PEAQ degradation.*

Bitrate	Variant			
/chan	FS	AT ($g_h = 0.2$)	AT ($g_h = 1.9$)	NR (q0)
24000	0.067	0.035	0.032	0.177
32000	0.027	0.031	0.093	0.332
48000	0.523	0.038	0.208	0.602
96000	0.092	0.027	0.108	0.291

Table 4: *Audio MUSHRA test items.*

Item	Description
35_short	xylophone
Hanco	jazz music
lrefhrp	harpsichord
lrefpip	tuning pipe
Mahle	orchestra recording
si02	castanets

Tables 2 and 3 show the average differential Objective Difference Grade (ODG) points with respect to the normal unmodified re-quantization operation, corresponding to the q1 operating point described in section 4.2. The encoder versions are, as before with frame smoothing (FS), adaptive threshold (AT) and no re-quantization at all (NR) as a lower quality bound, corresponding

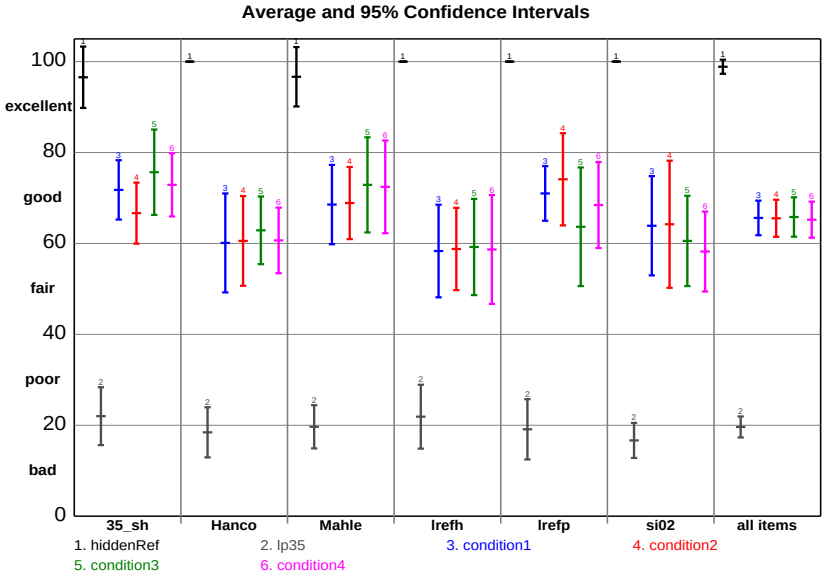


Figure 6: *Listening Test results for the different encoder modifications. Ten test subjects, seven expert listeners. Stereo files encoded with AAC-ELD, 96kbps (48 kbps per channel) at a sampling rate of 48 kHz.*

to the q0 operating point. When comparing Table 2 and 3 it can be seen that ODG differences are greater when only tonal-like signals are considered. The differential ODG values are transcribed without sign for the sake of clarity, but in all cases show a degradation in PEAQ scores with respect to the q1 operating point.

The items which presented the most degradation from the encoder reference were selected for taking part of a MUltiple Stimuli with Hidden Reference and Anchor (MUSHRA) listening test [24]. A short description of the audio items can be found in Table 4. The stereo files were encoded at a bitrate of 48 kbps per channel (96 kbps stereo) again using the AAC-ELD codec.

Figure 6 shows the results of the MUSHRA test for the different encoder variants with re-quantization active. An anchor item that corresponds to a low-pass filtered version of the original reference at 3.5 kHz is marked as "lp35". Condition 5 -no re-quantization- was not included in the listening test since it is considered to operate at a lower quality range, and not meant to be perceptually equivalent to the other versions. There were 10 test subjects, 7 of which are expert listeners. For reproducing the sound, Stax electrostatic headphones and amplifier were used in an acoustically controlled environment. As it can be seen from the test data, from particular items and total, there is no significant perceptual difference between encoder versions.

5. CONCLUSIONS

The error-resilient nature of audio coding algorithms permits a greater headroom for complexity reduction than other signal processing algorithms that do not make use of perceptual rules. In this case, we have shown that the perceived quality of the AbS algorithm in the quantization stage of our AAC-Encoder version [18] does not significantly change, even when the algorithm complexity is notably reduced (up to 20% of the total complexity or 80% of added complexity by AbS on a stereo file, without any hardware optimization). As already discussed in [1] and confirmed here, bit-exactness of the encoder output between versions is not the best figure of merit for conditioning the optimization work. Objective and subjective perceptual evaluation should also be performed in workload reduction strategies aimed to low power implementations. This "perceptually aware" optimization possibility is usually not recognized in later stages of implementation, where all reference algorithms are considered to be optimally tuned, even when computational requirements have not yet been thoroughly assessed.

The implementation stage of algorithms that make use of human perceptual models must be thoroughly evaluated, even during the final stages where usually only architectural optimizations take place. This approach can make a considerable difference when all architecture specific improvements are not enough in order to reach strict workload requirements.

On the case of mono files where joint stereo coding methods are not present, the relative workload reduction is around 30% for the AbS algorithm within perceptual equivalence to the unmodified version. Nevertheless, parametrizing algorithms under complexity-distortion trade-offs can be considered as extra flexibility on top of the work done already if perceptual equivalence does not need to be met. Future work includes trying to design a self-scaling algorithm that implements automated control on these parameters based on instantaneous workload measurements, given experimentally determined limits in section 4.1 and further investigation on parametrizing other modules of the encoder.

6. REFERENCES

[1] Markus Lohwasser, Marc Gayer, and Manfred Lutzky, "Implementing MPEG Advanced Audio Coding and Layer-3 encoders on 32-bit and 16-bit fixed-point processors," in *Audio Engineering Society Convention 115*, Oct 2003.

[2] Y. Andreopoulos, "Error tolerant multimedia stream processing: There's plenty of room at the top (of the system stack),"

Multimedia, IEEE Transactions on, vol. 15, no. 2, pp. 291–303, Feb 2013.

[3] M.A. Anam, P.N. Whatmough, and Y. Andreopoulos, "Precision-energy-throughput scaling of generic matrix multiplication and discrete convolution kernels via linear projections," in *Embedded systems for real-time multimedia (ESTIMedia), 2013 IEEE 11th Symposium on*, Oct 2013, pp. 21–30.

[4] V.K. Goyal and M. Vetterli, "Computation-distortion characteristics of block transform coding," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, Apr 1997, vol. 4, pp. 2729–2732 vol.4.

[5] M. Bosi and R. Goldberg, *Introduction to digital audio coding and standards*, chapter Psychoacoustic models for audio coding, pp. 179–200, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[6] Max Neuendorf et al., "MPEG Unified Speech and Audio Coding - The ISO/MPEG standard for high-efficiency audio coding of all content types," in *Audio Engineering Society Convention 132*, Apr 2012.

[7] Jürgen Herre, Johannes Hilpert, Achim Kuntz, and Jan Plogsties, "MPEG-H audio - the new standard for universal spatial/3D audio coding," *J. Audio Eng. Soc*, vol. 62, no. 12, pp. 821–830, 2015.

[8] Eric Allamanche, Ralf Geiger, Juergen Herre, and Thomas Sporer, "MPEG-4 low delay audio coding based on the AAC codec," in *Audio Engineering Society Convention 106*, May 1999.

[9] Manfred Lutzky, María Luis Valero, Markus Schnell, and Johannes Hilpert, "AAC-ELD V2 - the new state of the art in high quality communication audio coding," in *Audio Engineering Society Convention 131*, Oct 2011.

[10] 3GPP TS 26.445, "Codec for Enhanced Voice Services (EVS); detailed algorithmic description," Tech. Rep., 3GPP Technical Specification (Release 15), 2015.

[11] Ralf Geiger, Manfred Lutzky, Markus Schmidt, and Markus Schnell, "Structural analysis of low latency audio coding schemes," in *Audio Engineering Society Convention 119*, Oct 2005.

[12] Johannes Hilpert, Marc Gayer, Manfred Lutzky, Thomas Hirt, Stefan Geyersberger, and Josef Hoepfl, "Real-time implementation of the MPEG-4 Low-Delay Advanced Audio Coding algorithm (AAC-LD) on Motorola's DSP56300," in *Audio Engineering Society Convention 108*, Feb 2000.

[13] Xiaopeng Hu, Guiming He, and Xiaoping Zhou, "An efficient low complexity encoder for MPEG Advanced Audio Coding," in *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, Feb 2006, vol. 3, pp. 1500–1505.

[14] E. Alexandre, A. Pena, and M. Sobreira, "Low-complexity bit-allocation algorithm for MPEG AAC audio coders," *Signal Processing Letters, IEEE*, vol. 12, no. 12, pp. 824–826, Dec 2005.

[15] S. Nithin, T. V. Sreenivas, and Kumaraswamy Suresh, "Low complexity bit allocation algorithms for MP3/AAC encoding," in *Audio Engineering Society Convention 124*, May 2008.

[16] M. Bosi and R. Goldberg, *Introduction to digital audio coding and standards*, chapter Quantization, pp. 13–34, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[17] M. Bosi and R. Goldberg, *Introduction to digital audio coding and standards*, chapter MPEG2-AAC, pp. 346–350, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

[18] Fraunhofer IIS, "A standalone library of the Fraunhofer FDK AAC code for Android," Available at https://android.googlesource.com/platform/external/aac/+/master, Accessed February 19, 2016.

[19] T. Painter and A. Spanias, "Perceptual coding of digital audio," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 451–515, April 2000.

[20] ISO/IEC 14496-3:2009, "Information technology - Coding of audio-visual objects - Part 3: Audio," Tech. Rep., ISO/IEC, 2009.

[21] C.R. Helmrich, P. Carlsson, S. Disch, B. Edler, J. Hilpert, M. Neusinger, H. Purnhagen, N. Rettelbach, J. Robilliard, and L. Villemoes, "Efficient transform coding of two-channel audio signals by means of complex-valued stereo prediction," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, May 2011, pp. 497–500.

[22] Texas Instruments, *Introduction to TMS320C6000 DSP Optimization (app. note SPRABF2)*, Oct 2011, Application Report.

[23] ITU-R BS.1387-1, "Method for objective measurements of perceived audio quality," Tech. Rep., ITU,Geneva, Switzerland, 2001.

[24] ITU-R BS.1116-3, "Methods for the subjective assessment of small impairments in audio systems," Tech. Rep., ITU,Geneva, Switzerland, 2015.

NON-LINEAR IDENTIFICATION OF AN ELECTRIC GUITAR PICKUP

Antonin NOVAK, Leo GUADAGNIN, Bertrand LIHOREAU, Pierrick LOTTON, Emmanuel BRASSSEUR, Laurent SIMON

Laboratoire d'Acoustique de l'Université du Maine,
UMR CNRS 6613, Le Mans, France
antonin.novak@univ-lemans.fr

ABSTRACT

Physical models of electric guitars are still not very widespread in the scientific literature. Especially, the description of the non linear behavior of pickups still requires some refinements. This paper deals with the identification of pickup non linearities based on a Hammerstein representation, by means of a specific experimental set-up to drive the pickup in a controlled way. A comparison with experimental results shows that the model succeeds in describing the pickup when used in realistic conditions.

1. INTRODUCTION

An electric guitar pickup is a sensor that captures the string vibrations and translates them into an electric signal. It is basically composed of a set of permanent magnets surrounded by an electric coil. A ferromagnetic string vibrating in the vicinity of the pickup results in a variation of the magnetic flux through the coil. According to the Faraday's law, an electrical voltage is then induced across the coil.

A few models of pickup are available in the literature. Some of them are based on integral equations leading to the variation of magnetic flux at the coil location [1]. These models principally show, first, that vertical oscillations of the guitar strings produce a stronger effect than horizontal ones and, second, that there is a noticeable distortion of the electric signal generated by both oscillations. An overview of the modeling issues related to magnetic pickups is available in [2]. It concerns effects of both pickup position and pickup width on the pickup timbre, as well as the effect of the pickup internal impedance. In [2], the magneto-electric conversion done by the pickup is modeled using static non-linearity followed by a simple derivative (Fig. 1). The static non-linearity represents the non-linear relation between the string displacement and the magnetic flux which can be evaluated using computer simulations and implemented as an exponential or N-th order polynomial [3].

On the other hand, studies on non-linear modeling have led to many nonparametric non-linear models. Among these non-linear models, the Volterra series representation is usually considered as an effective one. Nevertheless, it lays down the calculation of multidimensional kernels and in practice, most applications are limited to the second or the third order. Simplified Volterra-based models, namely Hammerstein model (static nonlinear function followed by a linear filter) or Wiener model (linear filter followed by a static nonlinear function) [4], are then often preferred in the case of open-loop systems because of their simpler structure and lower computational cost. Furthermore, for a better accuracy of the estimation, these simple models can be extended to so-called generalized models, such as the generalized Hammerstein model, as shown in Fig. 2. This generalized Hammerstein model is made up of N parallel branches, with each branch consisting of a linear filter $G_n(f)$ preceded by an N-th order power static non-linear function, for $n = 1, N$, and has been successfully tested in [5, 6].

The goal of this paper is to proceed with the identification of pickup linearities based on a generalized Hammerstein representation of the pickup. For this purpose, a specific experimental set-up is used to drive the pickup in a controlled way, and a technique is carried out to get rid of non-linearities due to the driver. One of the aims of this study is to find out if it is meaningful, or not, to use a simple Hammerstein structure given in Fig. 1, as it usually done in modeling the pick-up nonlinearities [1, 2, 3], or if more complex model such as the Generalized Hammerstein one is necessary. The answer is given through the measurement provided in sections 2 and 3 and a comparison between theoretical and experimental results in the case of a realistic use of the pickup is given in section 4.

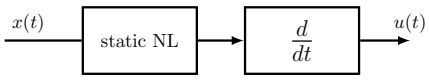


Figure 1: *Non-linear system usually used to model non-linearities of a guitar pickup [2].*

2. MEASUREMENT OF THE PICK-UP NON-LINEARITIES

The first goal of this paper is to identify the pickup in terms of Generalized Hammerstein model (Fig. 2). The output of such a model is governed by the following equation

$$u(t) = \sum_{n=1}^{N} x(t)^n * g_n(t), \tag{1}$$

where $g_n(t)$ is the inverse Fourier transform of $G_n(f)$ and where $*$ stands for convolution.

Since the pickup is an electromagnetic transducer that converts string vibration into an electrical output signal, its experimental characterization is not straightforward. Usually, when measuring a linear or a non-linear device, excitation signal is a controlled one (impulses, swept-sine, pseudo-random sequences) so that output signal can be used to identify the system in terms of a frequency response function (FRF) for a linear system, or in terms of a set of describing functions when dealing with a non-linear system. For a pickup, the excitation signal is the displacement of a plucked string exhibiting a multi-modal and non stationary behavior. Such an excitation is useful for a study in real conditions [7] but can hardly be used to get a FRF or to identify non linearities.

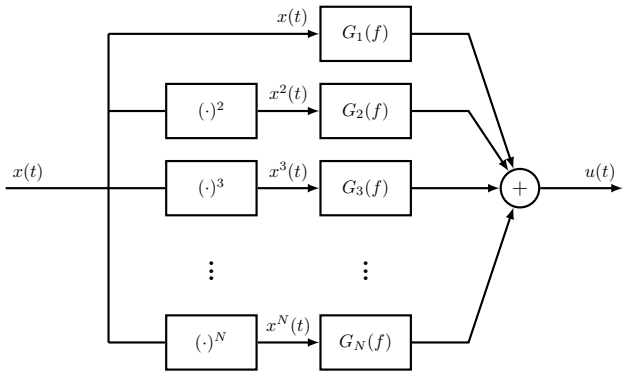


Figure 2: *Generalized Hammerstein model for identifying the non-linearities of the pickup; $x(t)$ and $u(t)$ represent the displacement of the guitar string and the output voltage of the pickup, respectively.*

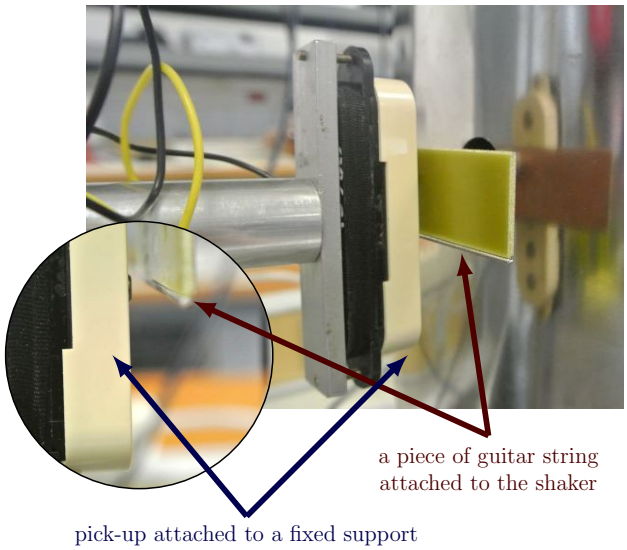


Figure 3: *Measurement device used to characterize the non-linearities of the pickup. The string portion is attached to a shaker; the pickup is attached on a fixed support.*

To control the string displacement, we use the system shown in Fig. 3. A piece of steel string (diameter 1.42 mm) is fixed on a non-magnetic support, itself fixed to a shaker which imposes a string displacement perpendicular to the pickup [8]. The pickup under test is set on a precision movement device which allows adjusting the distance at rest d_0 between the string and the magnet. For this experiment, d_0 is set to $d_0 = 5\ mm$ corresponding approximately to the distance set on a real guitar.

The shaker is driven by a Synchronized Swept Sine signal [9] so that the non-linearities of the whole system, that is the shaker and the pickup in series, can be easily identified using a Generalized Hammerstein model [5].

The synchronized swept-sine is generated using [9]

$$x(t) = \sin\left[2\pi f_1 L \exp\left(\frac{t}{L}\right)\right]. \tag{2}$$

where

$$L = \frac{T}{\ln\left(\frac{f_2}{f_1}\right)}, \tag{3}$$

and where f_1 and f_2 are initial and final frequency respectively and T is duration of the swept-sine. Note that the definition of the exponential swept-sine (Eq. (2)) does not contain the "-1" term contrary to the usual definition [10]. For more details about why the term "-1" should not appear in the exponential swept-sine definition please refer to [9].

To protect the shaker from a possible destruction due to excessive displacement or current, the frequency range is furthermore limited to the span 15 Hz - 500 Hz. The excitation signal is pre-filtered using a linear filter so as to obtain a displacement whose amplitude is almost constant over the frequency span. The peak amplitude is set here to 1 mm. The displacement of the string portion (that is the displacement of the shaker) is measured by means of a vibrometer pointing at the string. The electrical output of the pickup is then connected to an acquisition card which exhibits a high input impedance ($470\ k\Omega$). Consequently, the measured output voltage corresponds to the open-circuit voltage which does not take into account the effect of pickup output impedance.

The Higher Harmonic Frequency Responses (HHFRs) of both the string displacement and the pickup output voltage calculated using the Synchronized Swept Sine method [9] are given in Fig. 4. The method consists in de-convolving the measured signals with a so-called inverse filter as

$$h(t) = \mathcal{F}^{-1}\left[\mathcal{F}[y(t)]\tilde{X}(f)\right], \tag{4}$$

where $y(t)$ is the acquired response of the nonlinear system (displacement or voltage signal) to the synchronized swept-sine, and where the Fourier transform of inverse filter $\tilde{X}(f)$ is given analytically as

$$\tilde{X}(f) = 2\sqrt{\frac{f}{L}}\exp\left\{-j2\pi f L\left[1 - \ln\left(\frac{f}{f_1}\right)\right] + j\frac{\pi}{4}\right\}. \tag{5}$$

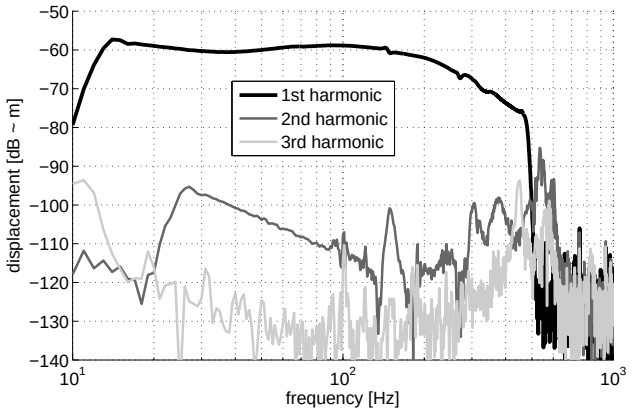
The impulse response $h(t)$ then consists of time-delayed higher harmonic impulse responses, separated by time delays

$$\Delta t_n = L\ln(n), \tag{6}$$

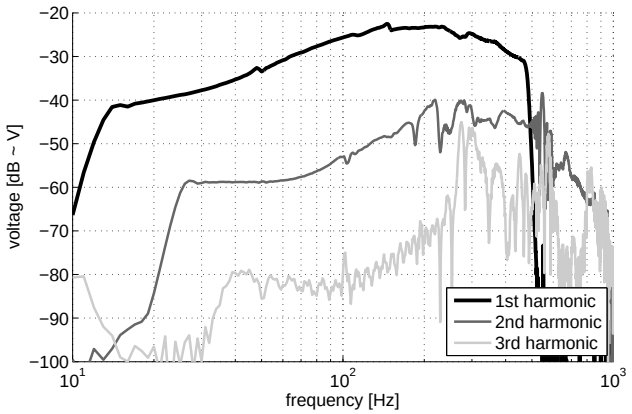
that can be windowed and represented in the frequency domain as The Higher Harmonic Frequency Responses (HHFRs). For more details see [9].

The fundamental harmonic of the string displacement (Fig. 4a) is not flat despite the pre-filtering and the second harmonic reaches -40 dB relative to fundamental harmonic. It is thus rather difficult, or almost impossible, to estimate which part of the HHFRs of the pickup output voltage (Fig. 4b) is due to the pickup behavior and which part is due to the shaker behavior.

To overcome this problem, we use a technique detailed in [11] in which a non-linear system can be identified even if it is preceded by another unknown non-linear system. According to this technique, the non-linear system under test is then described by an N-th order Generalized Hammerstein model, as shown in Fig. 2. For the pickup under test, the magnitude values of the estimated linear filter $G_n(f)$ of the Generalized Hammerstein model are depicted in Fig. 5.



(a) HHFRs of displacement of the string excited by the shaker



(b) HHFRs of the output voltage of the pickup

Figure 4: *Higher Harmonic Frequency Responses (HHFRs) of (a) displacement of the string excited by the shaker and (b) the output voltage of the pickup.*

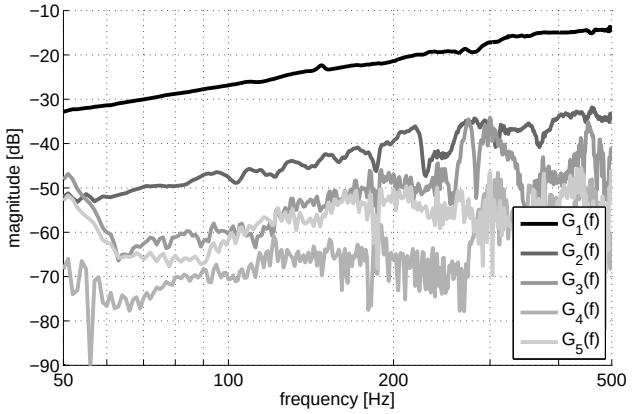
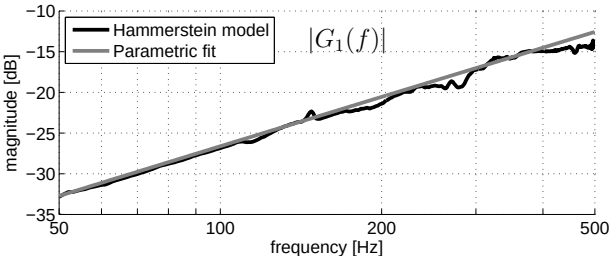
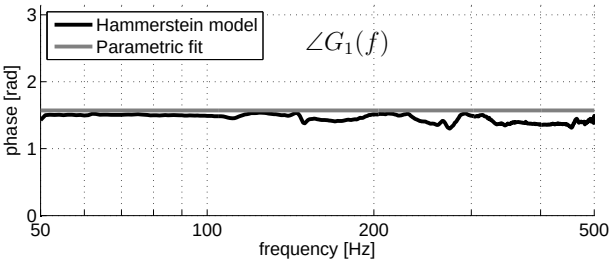


Figure 5: *Magnitude values of the estimated filters $G_n(f)$ of the Generalized Hammerstein model (Fig. 2) of the pickup.*

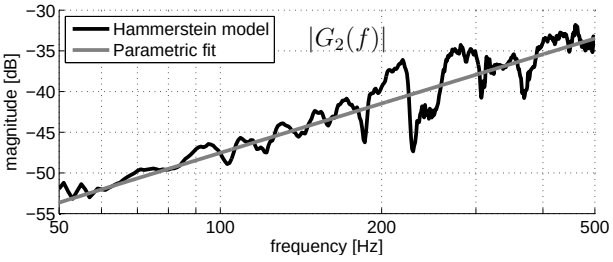


(a)

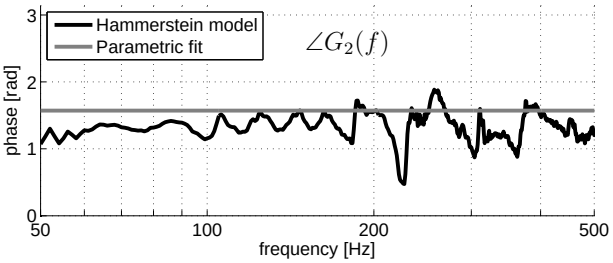


(b)

Figure 6: *Modulus (a) and phase (b) of the first filter $G_1(f)$ of the identified Generalized Hammerstein model of the pickup under test and its parametric fit.*



(a)



(b)

Figure 7: Modulus (a) and phase (b) of the second filter $G_2(f)$ of the identified Generalized Hammerstein model of the pickup under test and its parametric fit.

Table 1: *Coefficients α_n of the estimated parametric Hammerstein model.*

α_1	7.50e−02
α_2	6.75e−03
α_3	2.11e−03
α_4	4.75e−04
α_5	8.31e−04

3. NON-LINEAR PARAMETRIC MODEL OF THE PICKUP

Observing the estimated filters of the Generalized Hammerstein model depicted in Fig. 5, one can note that the dependence on frequency for all filters is approximately 6 dB/oct. Such a slope corresponds to $j2\pi f$ in the frequency domain or to a simple derivative d/dt in the time domain.

It is thus tempting to fit all the filter responses $G_n(f)$ with a function $\alpha_n j2\pi f$ in order to replace each branch of the Generalized Hammerstein model by a multiplicative coefficient α_n in series with a derivative function d/dt. A fit of the first two filters $G_1(f)$ and $G_2(f)$ (magnitude and phase) is depicted in Figs. 6 and 7. It is interesting to note that the estimated phases of both filters are very close to $\pi/2$ within the whole frequency range.

The estimated Generalized Hammerstein model can thus be parametrized and simplified to the following relation

$$u(t) = \sum_{n=1}^{N} \alpha_n \frac{d\, x(t)^n}{dt} = \frac{d}{dt}\left(\sum_{n=1}^{N} \alpha_n x(t)^n\right). \qquad (7)$$

This relation being a time derivative of a Taylor series, we can simplify the Generalized Hammerstein model to a Hammerstein model consisting of a static non-linearity followed by a linear filter (Fig. 1). In this particular case, the static non-linearity is represented by a simple Taylor series with coefficients α_n and the linear filter is represented by a time domain derivative, as shown in Fig. 1. This result tends to confirm the model previously proposed by [2]. The fitted coefficients α_n for the pickup under test are given in Table 1 and the corresponding input-output characteristic is depicted in Fig. 8.

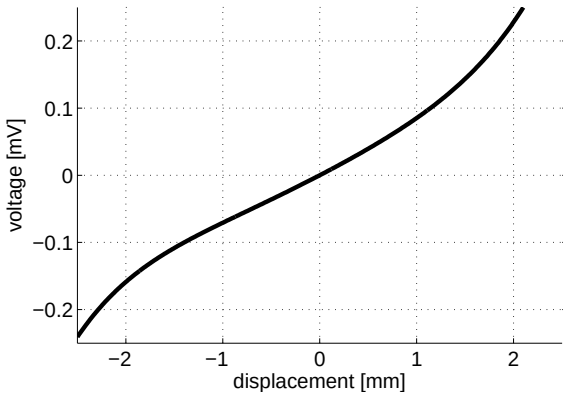


Figure 8: *Input-output graph of the static nonlinerity based calculated as a power series developpement with coefficinets α_n given in Table 1.*

Figure 9: *Picture of the second experiment in which the pickup is placed under a vibrating string.*

4. MODEL VS. REAL GUITAR-STRING SIGNAL

To test the validity of the identified Hammerstein model, we set up a different experiment corresponding to a realistic use of the pickup. For that purpose, we use a lab guitar prototype [8]. A guitar string is fixed on an wooden beam. The string is tuned as open low E (f_0 = 82 Hz). The pickup under test is set on a mechanical arm on which some precision movement pieces are fixed. Thanks to this system, the pickup position under the string can be adjusted along the 3 axes. For this experiment, the pickup is set at 1/4 of the total length of the string corresponding approximately to the neck position on a real guitar and the distance at rest between the string and the pickup is set at 5 mm. A detail of the experiment is shown in Fig. 9. The string is struck using an impact hammer. A vibrometer pointing at the string at the pickup location allows the measurement of the string displacement in the vertical plane. Temporal evolution of both string displacement and pickup output voltage are recorded simultaneously and depicted in Figs. 10 and 11. A zoom on a few periods of both experimental signals is given for three different time lags along the time-varying response. As expected, the string displacement is distorted just after the excitation. It becomes less and less distorted as the harmonics of higher orders fade with time. The output signal of the pickup exhibits the same kind of behavior with time. One can notice that the output voltage is more distorted due to pickup non-linearities.

The displacement signal measured with the vibrometer is then used as the input of estimated parametric Hammerstein model of the pickup and the both measured and synthesized pickup outputs are compared on the same graph (Fig. 11). The difference between estimated and experimental signals is plotted on Fig. 11, showing that the model succeeds in describing the non linear behavior of the pickup when used in realistic conditions.

5. DISCUSSION

The results presented in this paper shows that a simple Hammerstein model seems to be sufficient for the pick-up modeling and that using a Generalized Hammerstein model is not necessary. However, several hypotheses have been put forward simplifying the problem that might be at the origin of small differences between the measured and modeled pick-up outputs compared in Figs. 10

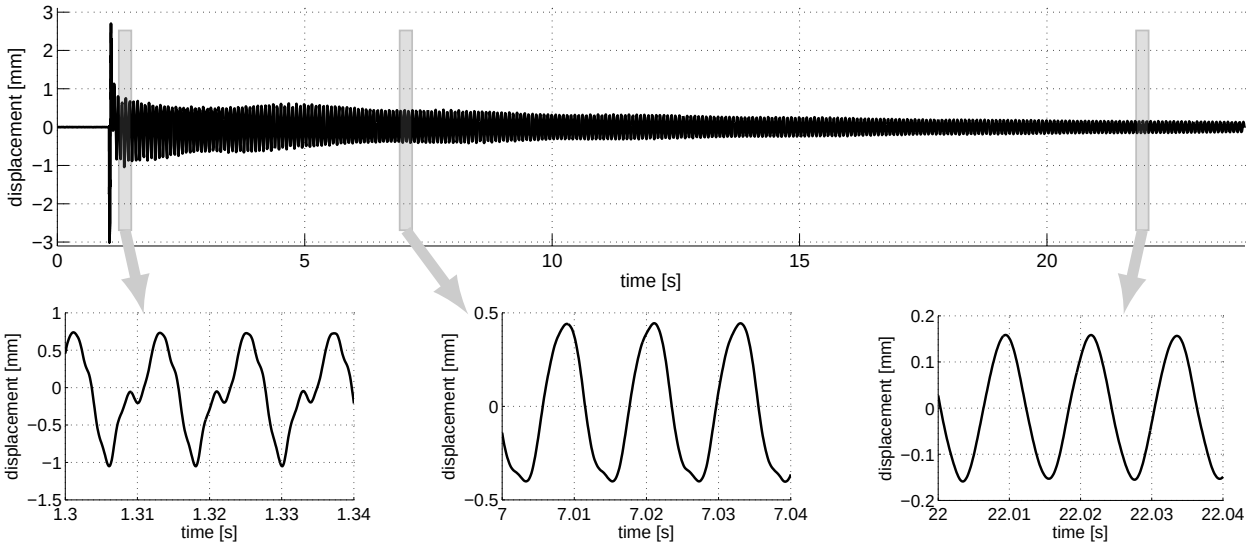


Figure 10: Recorded signals of the vibrating string.

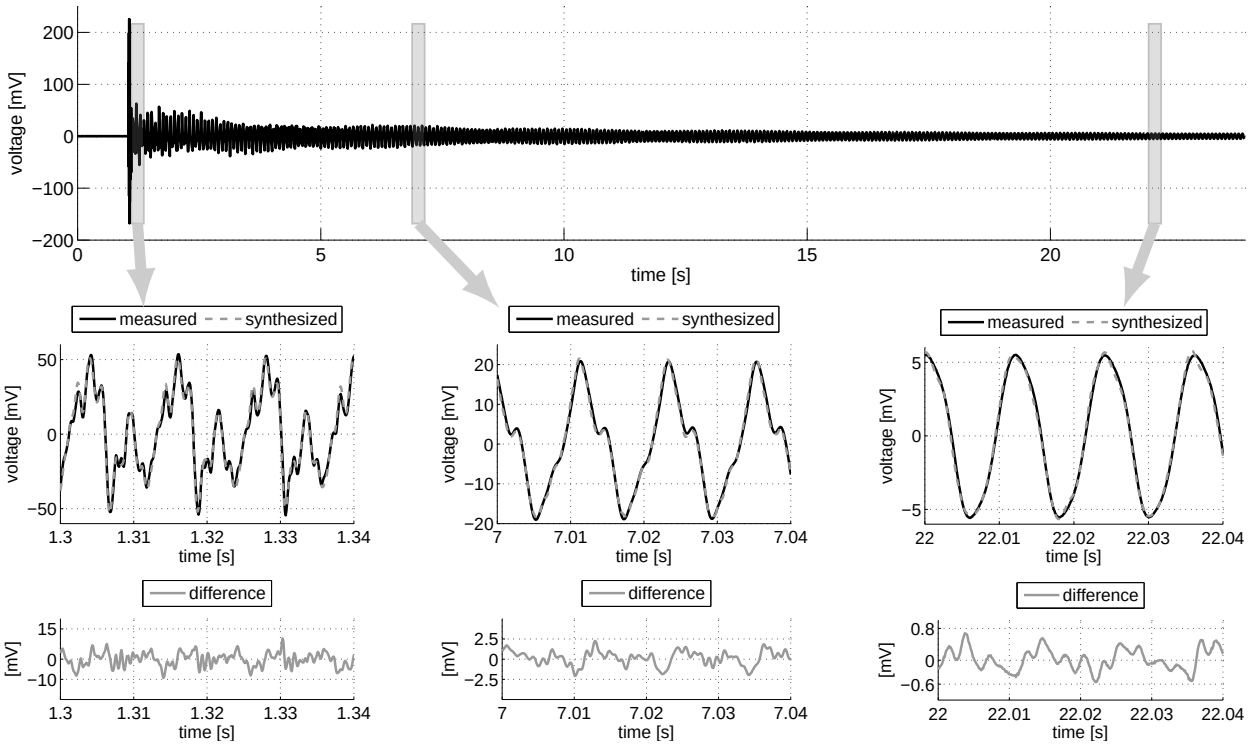


Figure 11: Recorded and synthesized signals of the voltage from the pickup.

and 11.

First, the frequency range of the excitation signal is bellow 500 Hz due to the capacities of the shaker. Using larger frequency range might have been beneficial. The nonlinearities of the pick-up may differ at higher frequencies, and thus, in such a case, a complete Hammerstein Generalized model might be useful. A supplementary study would be necessary to draw a meaningful conclusion.

Next, the movement of the rigid string attached to the shaker exhibits only z-axis polarization, whereas it is known that the string being played by a guitar player exhibits rather an ellipsoid type motion in both y and z directions [12, 13]. A hammer-like impact has been used to excite the string in order to approach the z-axis motion of the string in the comparative measurements whose results are provided in Figs. 10 and 11. However, the real-word pick-up behavior might be influenced by 2D movement of the string, even though it is known that the y-axis contribution is rather negligible [14, 8]. Moreover, the piece of the rigid string attached to the shaker is of finite length and does not exhibit any deformation compared to a string attached on guitar.

Even though these phenomena have been neglected, the results presented in this paper show a very good agreement between the output predicted by the model and the output obtained from the experimental measurement.

6. CONCLUSIONS

This paper presents a parametric model of guitar pickup whose parameters are directly estimated experimentally. The validity of this model has been verified for a pickup operating in a realistic way. In future work, this model can be used to synthesize different kinds of existing pickups (single coil pickups, humbuckers). It can also be extended to the synthesis of augmented pickups by artificially modifying the parameters of the model.

7. REFERENCES

[1] Nicholas G Horton and Thomas R Moore, "Modeling the magnetic pickup of an electric guitar," *American Journal of Physics*, vol. 77, no. 2, pp. 144–150, 2009.

[2] Rafael CD Paiva, Jyri Pakarinen, and Vesa Välimäki, "Acoustics and modeling of pickups," *Journal of the Audio Engineering Society*, vol. 60, no. 10, pp. 768–782, 2012.

[3] Luca Remaggi, Leonardo Gabrielli, R de Paiva, Vesa Välimäki, and Stefano Squartini, "A pickup model for the clavinet," in *Digital Audio Effects Conference 2012 (DAFx-12)*, 2012.

[4] Martin Schetzen, "The volterra and wiener theories of nonlinear systems," 1980.

[5] Antonin Novak, Laurent Simon, František Kadlec, and Pierrick Lotton, "Nonlinear system identification using exponential swept-sine signal," *Instrumentation and Measurement, IEEE Transactions on*, vol. 59, no. 8, pp. 2220–2229, 2010.

[6] Marc Rébillat, Romain Hennequin, Etienne Corteel, and Brian FG Katz, "Identification of cascade of hammerstein models for the description of nonlinearities in vibrating devices," *Journal of Sound and Vibration*, vol. 330, no. 5, pp. 1018–1038, 2011.

[7] Mirko Mustonen, Dmitri Kartofelev, Anatoli Stulov, and Vesa Välimäki, "Experimental verification of pickup nonlinearity," in *Proceedings International Symposium on Musical Acoustics (ISMA 2014), Le Mans, France*, 2014, vol. 1.

[8] Pierrick Lotton, Bertrand Lihoreau, and Emmanuel Brasseur, "Experimental study of a guitar pickup," in *Proceedings International Symposium on Musical Acoustics (ISMA 2014), Le Mans, France*, 2014, vol. 1.

[9] Antonin Novak, Pierrick Lotton, and Laurent Simon, "Synchronized swept-sine: Theory, application, and implementation," *Journal of the Audio Engineering Society*, vol. 63, no. 10, pp. 786–798, 2015.

[10] Angelo Farina, Alberto Bellini, and Enrico Armelloni, "Nonlinear convolution: A new approach for the auralization of distorting systems," in *AES 110th convention*, Amsterdam, May 2001.

[11] Antonin Novak, Balbine Maillou, Pierrick Lotton, and Laurent Simon, "Nonparametric identification of nonlinear systems in series," *Instrumentation and Measurement, IEEE Transactions on*, vol. 63, no. 8, pp. 2044–2051, 2014.

[12] Jim Woodhouse, "Plucked guitar transients: Comparison of measurements and synthesis," *Acta Acustica*, vol. 90, no. 5, pp. 945–965, 2004.

[13] Montserrat Pàmies-Vilà, Ibrahim Atakan Kubilay, Dmitri Kartofelev, Mirko Mustonen, Anatoli Stulov, and Vesa Välimäki, "High-speed linecamera measurements of a vibrating string," in *Proceeding of Baltic-Nordic Acoustic Meeting (BNAM), Tallinn, Estonia*, 2014, vol. 1.

[14] Arthur Paté, Jean-Loïc Le Carrou, and Benoît Fabre, "Predicting the decay time of solid body electric guitar tones," *The Journal of the Acoustical Society of America*, vol. 135, no. 5, pp. 3045–3055, 2014.

MONOPHONIC PITCH DETECTION BY EVALUATION OF INDIVIDUALLY PARAMETERIZED PHASE LOCKED LOOPS

Johannes Böhler, Udo Zölzer

Department of Signal Processing and Communications
Helmut Schmidt University
Hamburg, Germany
johannes.boehler@hsu-hh.de

ABSTRACT

This paper describes a new efficient and sample based monophonic pitch tracking approach using multiple phase locked loops (PLLs). Hereby, distinct subband signals traverse pairs of individually parameterized PLLs. Based on the relation of the instantaneous pitch sample of respective PLLs to one another, relevant features per pitch candidate are derived. These features are combined into pitch candidate scores. Pitch candidates which exhibit the maximum score per sampling instance and exceed a voicing threshold, contribute to the overall pitch track. Evaluations with up to date datasets show that the tracking performance, compared to implementations which use only one PLL has significantly improved and nearly approaches the scores of a state of the art monophonic pitch tracker.

1. INTRODUCTION

Pitch is a perceptual feature which is still subject to discussion and lacks an explicit mathematical definition. In the presented approach pitch is therefore considered to be the momentarily present fundamental frequency. In 3.2 the definition of pitch is discussed further with regard to the comparison with an alternative pitch tracking technique. If pitch information is extracted from an audio signal, it can be used to control further audio signal processing in many possible ways. Until today various monophonic pitch tracking techniques have been developed. Some of these approaches deliver robust and satisfying results. However, most of these systems employ block-based analysis and their implementation can be computational expensive. A block-based approach, named PYIN, applying the difference function paired with probabilistic evaluation and post-processing [1] yields the best results to date. This paper introduces a new sample-based approach for monophonic pitch extraction using multiple PLLs which is computationally efficient and suitable for implementations on low-power processors with limited resources. PLLs have been used for music information retrieval purposes as beat tracking [2] and monophonic pitch detection before. A pitch tracker combining numerous phase locked loops, involving a lot of redundancy leading to high computational cost, is presented in [3]. In another approach a single, modified PLL is used for pitch extraction [4]. This leads to satisfying results for input signals where the respective overtone energy is lower than the energy of the fundamental frequency. Otherwise octave errors might occur and the single PLL locks to overtone frequencies. From here on this algorithm is referenced throughout this paper as Single PLL while the algorithm which is presented in the following text will be referred to as Multi PLL. If instead of a single PLL multiple PLLs with equal parametrization are applied in differing, slightly overlapping subbands, one can observe merging pitch tracks of neighboring PLLs [5]. This observation leads to the idea to combine distinct subbands and variably configured PLLs in a new way in order to exploit the occurring concurrence of pitch tracks on periodic monophonic audio input. The following paper is structured as follows. Section 2 gives a system overview by presenting how multiple bandpass filters and phase locked loops are configured and combined. It shows how particular PLL pitch samples are interpreted in order to extract significant features regarding the instantaneous fundamental frequency of the monophonic audio signal. Based on these features the derivation of PLL-dependent pitch candidate scores and the successive selection of a candidate is described. Section 3 compares the pitch tracking performance of the approach presented in this paper with the original Single PLL pitch detector and the PYIN algorithm by means of a guitar-based dataset. Section 4 summarizes the findings of this study and discusses possible future enhancements.

2. SYSTEM OVERVIEW

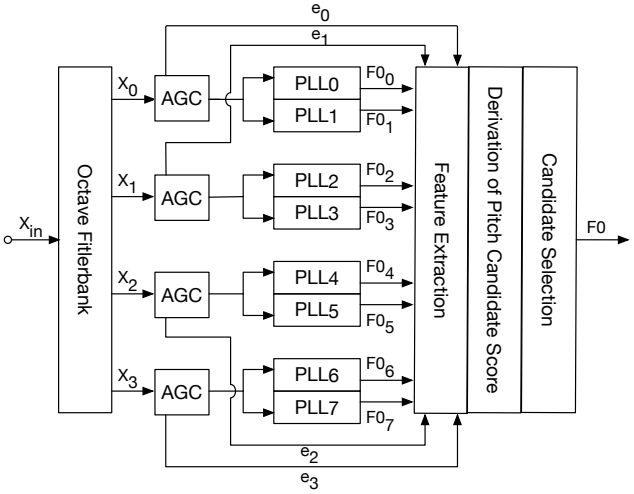


Figure 1: *Block diagram of the Multi PLL algorithm*

First, the input audio signal is filtered using a decimation filter in order to conduct a successive downsampling to a sample frequency of 11 kHz. A filter bank then divides the input audio signal into 4 octave bands. For each subband signal an envelope is calculated in order to generate a constant envelope signal of unity amplitude (AGC block in Fig. 1). This dynamic pre-processing allows the PLLs to achieve an optimal tracking performance. After

the subband signals have passed the gain control stage they are fed to the respective PLL pairs. Output samples of all 8 PLLs are interpreted in order to extract 5 features per PLL. Each feature is based on a different relation as pitch pair deviation, the number of pitch candidates assigned to relative subtones/overtones, number of close pitch candidates and pitch slope. A pitch candidate score for each PLL output sample is derived by combining these features. The pitch candidate scores reflect signal properties as periodicity, harmonicity and pitch slope. Hereby the PLL itself covers the feature extraction which is related to periodicity, while the combination of particular pitch tracks into features pursues amongst others the quantification of harmonicity. The pitch candidate with the highest score is selected and contributes to the overall pitch track F0, provided that a certain voicing threshold is exceeded.

2.1. Filterbank

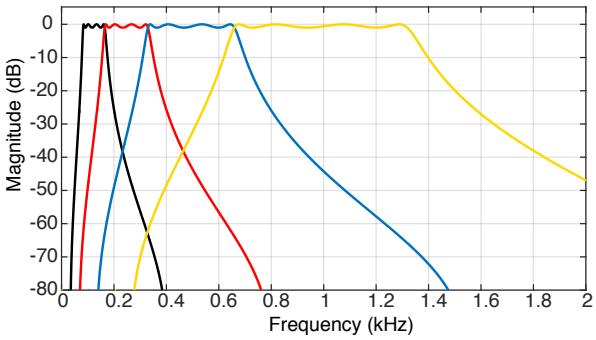


Figure 2: *Absolute frequency responses of utilized 8th order elliptic filters*

The filter bank is composed of $I = 4$ bandpass filters as depicted in Fig. 2. Each sub band channel $i \in [0, \ldots, I-1]$ spans over a region of one octave. 8th order elliptic bandpass filters with a stop band attenuation of 80 dB and a passband ripple of 1 dB are used to facilitate a sufficient edge steepness. This parametrization results in a 63 dB/Octave roll-off, which supports the isolation of fundamentals as illustrated in Fig. 4. The subbands are enclosed by the ripple frequencies

$$f^r_{i,j} = 80.06 \text{ Hz} \cdot 2^{i+j}, \tag{1}$$

where $j = 0$ denotes the lower ripple frequency and $j = 1$ denotes the upper ripple frequency. The magnitude frequency responses $|H_i(e^{j\omega})|$ of the filters are shown in Fig. 2. By filtering the input signal in both the forward and the reverse direction, a phase distortion of the output signal can be prohibited.

The approach described in this study mainly aims at providing the best possible tracking performance of the overall system under ideal performance of the subsystems. Ideal performance with regard to the filterbank subsystem means, the attainment of a certain edge steepness of respective subband filters without the emergence of phase distortions. This ensures the isolation of the fundamental frequency without the presence of the first overtone in the target octave band. At the same time a frequency dependent delay of pitch tracks is prohibited. The above mentioned characteristics can be achieved in the simplest way if the processing of the filterbank is conducted offline. The implementation of the filterbank is the only reason why the overall system is bound to offline processing. All succeeding modules can be adapted for real-time processing without much effort and worth mentioning side effects. In order to implement the whole system in a real-time-capable fashion an alternative design for the filterbank has to be considered.

Within each channel a temporal envelope $e_i(n)$ is generated whose inverse value is used to generate the constant envelope signal

$$\bar{x}_i(n) = \frac{x_i(n)}{e_i(n) + e_{min}}. \tag{2}$$

$e_i(n)$ is computed using the smoothed decoupled peak detection algorithm [6] with attack $\tau_a = 50$ ms and release $\tau_r = 100$ ms. In the denominator e_{min} is added as a constant offset in order to prevent divisions by zero. The chosen parametrization depicts a trade off between minimum-time envelope tracking and the containment of arising nonlinearities within the subband. The further usage of $e_i(n)$ for feature extraction is described in 2.3.

2.2. Phase locked loop

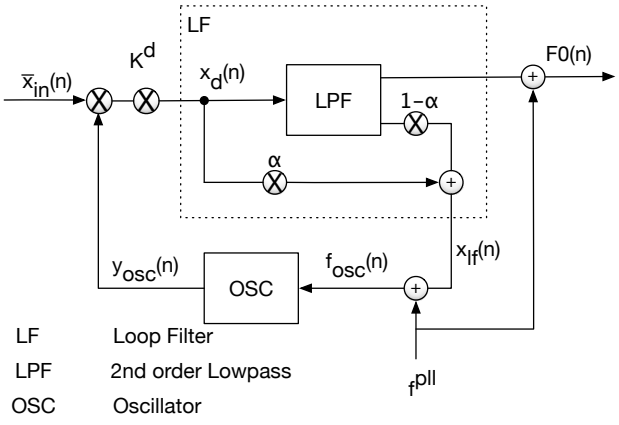


Figure 3: *Block diagram of the modified PLL*

In order to track partials which are present in a respective subband, two third order PLLs operating in a nonlinear mode are installed. The phase detector is implemented as a simple multiplier which uses the constant envelope signal $\bar{x}_{in}(n)$ and the oscillator output $y_{osc}(n)$ as input signals (Fig. 3). The resulting signal contains the difference-frequency and the sum-frequency of the real valued input signals. This signal is in succession amplified by the constant gain factor K^d which controls the frequency range of detection and the sensitivity of the PLL towards changes of the phase difference between the two signals. $x_d(n)$ then traverses a 2nd order lowpass filter (LPF in Fig. 3) with a cutoff frequency of 23 Hz, which is also part of the loop filter, in order to eliminate the sum frequency component of the multiplier output. The forward path is continued by adding the constant PLL specific center frequency f^{pll} to the lowpass filter output. The resulting output signal denotes the F0 track of the PLL. This F0 track is then filtered irrespective of the loop filter in the forward and reverse direction using a first order recursive moving average filter with non recursive coefficient 0.05 and recursive coefficients [1, -0.95] in order to further attenuate undesired high frequency oscillations.

In known implementations the lowpass filter (LPF in Fig. 3) alone represents the loop filter in the feedback path. Because

this implementation requires an immediate tracking of varying frequencies over a big range without the presence of a constant carrier frequency [7], modifications to the loop filter have to be applied in order to obtain satisfying tracking results. This modification is realized using a low frequency shelving filter in the feedback path formed by the weighted sum of x_d(n) and the lowpass-filtered phase detector output [4]. For α a value of 0.35 is chosen. In the feedback path

$$f_{osc}(n) = f^{pll} + x_{lf}(n) \tag{3}$$

controls the instantaneous frequency of the oscillator according to which the phase is incremented. The oscillator emits the real-valued signal

$$y_{osc}(n) = \cos\big(\phi(n)\big) \tag{4}$$

with the wrapped phase

$$\phi(n) = \left(\phi(n-1) + 2\pi\frac{f_{osc}}{f_s}\right) \bmod 2\pi. \tag{5}$$

By feeding $y_{osc}(n)$ and $x_{in}(n)$ to the multiplier, the loop is closed. The basic concept behind the pairwise positioning of the PLLs is the idea that a fundamental frequency within one subband must be tracked by both, the upper and the lower PLL as illustrated in Fig. 4. The parametrization of the 8 PLLs differs in the used values for the gain of the phase detector output K^d, and the PLL center frequency f^{pll}. The gain of the phase detector

$$K_i^d = 600 \cdot (i+1) \tag{6}$$

is adapted per band in a linear fashion and has been determined experimentally. Each subband i is enclosed by a respective PLL pair with lower

$$f_{i,l}^{pll} = 80.06 \text{ Hz} \cdot 2^{i-\frac{1}{12}} \tag{7}$$

and upper

$$f_{i,u}^{pll} = 80.06 \text{ Hz} \cdot 2^{i+\frac{13}{12}} \tag{8}$$

center frequency. For the following steps the octave indexes

$$i \rightarrow p = 2i + j \tag{9}$$

are mapped to PLL indexes p where $j = 0$ refers to the lower and $j = 1$ to the upper PLL.

PLL pairs assigned to subbands that contain merely overtone energy tend to track distinct overtones, while PLL pairs that track the fundamental coincide. If there is no sufficient periodic signal portion apparent in the respective octave band, the F0 track falls back to the PLLs center frequency. This behavior, which supports the differentiation between the fundamental and higher order partials, is shown in Fig. 5. The same principle applied to a guitar lick recording is depicted in Fig. 6. The spectrogram is overlayed with candidate pitch tracks.

2.3. Feature Extraction

For every sampling instance a single pitch sample for each of the 8 individually parameterized PLLs ($F0_p(n)$) is emitted. Based on these samples, the following 5 features are extracted and described in detail. Each feature is determined by the mapping of a calculated difference Δ or a count of pitch candidates N that fulfill certain constraints.

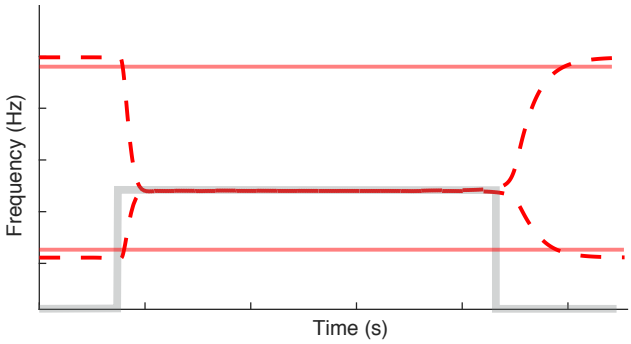


Figure 4: *The PLL pair that corresponds to the channel in which the fundamental is contained concurs (dashed lines). The continuous lines depict the ripple frequencies of the elliptic bandpass filter and the grey line represents the ground truth.*

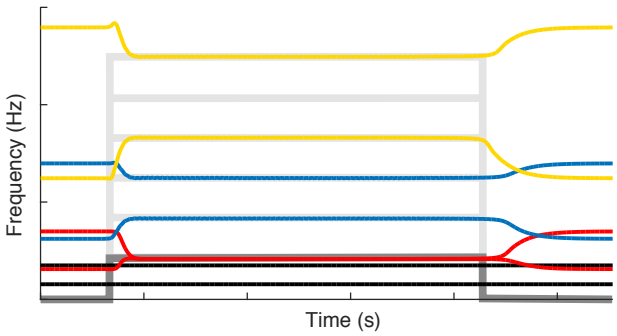


Figure 5: *Optimal tracking of the fundamental and corresponding overtones. Only the PLL pair that tracks the fundamental coincides (red), while all other PLLs remain on their center frequency (black) or track distinct overtones (blue, yellow). The grey lines denote the partial frequencies.*

2.3.1. Pitch pair deviation

We assume that PLL pairs which process the same subband signal coincide if the fundamental is contained within this channel (Fig. 4). In this context

$$\Delta f_p(n) = \left|1200 \cdot \log_2\left(\frac{F0_{\lfloor \frac{p}{2} \rfloor \cdot 2}(n)}{F0_{\lfloor \frac{p}{2} \rfloor \cdot 2+1}(n)}\right)\right| \tag{10}$$

quantifies the distance between $F0_{\lfloor \frac{p}{2} \rfloor \cdot 2}(n)$ and $F0_{\lfloor \frac{p}{2} \rfloor \cdot 2+1}(n)$, the particular pitch pair residing in subband $i = \lfloor \frac{p}{2} \rfloor$, on a logarithmic scale. This leads to the feature

$$F_p^{\Delta f}(n) = \frac{100 - \Delta f_p(n)}{100} \tag{11}$$

with $-\infty < F_p^{\Delta f}(n) \leq 1$. The negative range of $F_p^{\Delta f}(n)$ expresses the importance of coinciding pitch pairs when seeking the fundamental frequency.

2.3.2. Pitch candidates assigned to relative subtones

For a PLL pitch sample $F0_p(n)$ which is assigned to a fundamental, no other pitch candidate $F0_q(n)$ may refer to a relative subtone. There are two criteria a potential subtone $F0_q(n)$ of $F0_p(n)$ has to fulfill in order to be classified accordingly. The absolute difference in Cent between $F0_p(n)$ and an integer multiple of $F0_q(n)$ must fall below ΔC, and the relative subband energy

$$E_i(n) = \frac{e_i(n)^2}{\sum_{j=0}^{3} e_j(n)^2}, \tag{12}$$

must exceed 3%. ΔC is set to 40 Cent in order to consider inherent oscillations of the F0 tracks as well as inaccuracies and inconsistencies of the particular F0 tracks which can occur during the attack of a tone. The subband energy is considered before the signal passes the automatic gain control stage (AGC block in Fig. 1) in order to prohibit false subtone detections caused by the amplification of noise components. A false subtone detection would lead to an incorrect exclusion of pitch candidate $F0_p(n)$ in the following scoring. The number of PLL pitch samples $F0_q(n)$ that refer to a relative subtone of a particular PLL pitch sample $F0_p(n)$ is defined as

$$N_p^{st}(n) = \sum_{q=0}^{p-1} \sum_{o=2}^{8} \begin{cases} 1 & \left| 1200 \cdot \log_2 \left(\frac{F0_p(n)}{F0_q(n) \cdot o} \right) \right| < \Delta C \wedge E_{\lfloor \frac{q}{2} \rfloor}(n) > 3\% \\ 0 & \text{else.} \end{cases} \tag{13}$$

$N_p^{st}(n)$ is mapped to the mandatory feature

$$F_p^{st}(n) = \begin{cases} 1 & N_p^{st}(n) = 0 \\ 0 & \text{else.} \end{cases} \tag{14}$$

If a relative subtone of a pitch candidate $F0_p(n)$ has been identified, its overall score is set to zero as noted in Eq. (21).

2.3.3. Pitch candidates assigned to relative overtones

Pitch candidates $F0_q(n)$ which reside near integer multiples of the currently examined PLLs pitch sample $F0_p(n)$ reinforce the assumption that $F0_p(n)$ is a fundamental frequency. The number of pitch candidates $F0_q(n)$ which are classified as overtones of $F0_p(n)$ is defined as

$$N_p^{ot}(n) = \sum_{q=p+1}^{7} \sum_{o=2}^{8} \begin{cases} 1 & \left| 1200 \cdot \log_2 \left(\frac{F0_p(n) \cdot o}{F0_q(n)} \right) \right| < \Delta C \\ 0 & \text{else,} \end{cases} \tag{15}$$

from which the feature

$$F_p^{ot}(n) = \frac{N_p^{ot}(n)}{7} \tag{16}$$

is derived. This feature favors PLLs with lower-order indexes in order to prevent overtone errors.

2.3.4. Close pitch candidates

A high number of PLL pitch samples $F0_q(n)$ that reside near pitch sample $F0_p(n)$

$$N_p^{cp}(n) = -1 + \sum_{q=0}^{7} \begin{cases} 1 & \left| 1200 \cdot \log_2 \left(\frac{F0_p(n)}{F0_q(n)} \right) \right| < \Delta C \\ 0 & \text{else} \end{cases} \tag{17}$$

exhibits that $F0_p(n)$ comparatively carries a lot of energy. This is represented by the feature

$$F_p^{cp}(n) = \frac{N_p^{cp}(n)}{N_{max}^{cp}}, \tag{18}$$

with N_{max}^{cp} = 4 based on analyzed $N_p^{cp}(n)$ outputs.

2.3.5. Pitch slope

After the attack phase of a tone has passed, its pitch is assumed to be stable with little variation in time. Therefore, a low order

$$\Delta t_p(n) = \left| 1200 \cdot \log_2 \left(\frac{F0_p(n)}{F0(n-1)} \right) \right|, \tag{19}$$

denoting a flat pitch slope, increases the feature

$$F_p^{\Delta t}(n) = \begin{cases} \frac{3 - \Delta t_p(n)}{3} & F0(n-1) \neq 0 \wedge \frac{3-\Delta t_p(n)}{3} > 0 \\ 0 & \text{else.} \end{cases} \tag{20}$$

2.4. Pitch candidate selection

The extracted features are combined into a final pitch candidate score

$$S_p(n) = \frac{F_p^{\Delta f}(n) + F_p^{ot}(n) + F_p^{cp}(n) + F_p^{\Delta t}(n)}{4} \cdot F_p^{st}(n). \tag{21}$$

It is assumed that the correct instantaneous fundamental frequency equals at least one of the PLL output samples, provided that the signal carries enough periodic and harmonic portions. Hence, the output sample with the highest pitch candidate score is determined

$$k = \underset{p \in [0,\ldots,7]}{\arg\max}\, S_p(n) \tag{22}$$

and added to the overall pitch track F0 if the score exceeds a certain voicing threshold

$$F0(n) = \begin{cases} F0_k(n) & S_k(n) > T^v_{\lfloor \frac{k}{2} \rfloor} \\ 0 & \text{else.} \end{cases} \tag{23}$$

If none of the pitch samples exceeds the threshold, the audio signal is assumed to be unvoiced. In this case a zero is appended to the overall pitch track. Outliers caused by overtones, which appear when overtone energy is present before the energy of the fundamental, are smoothed using a nonlinear median filter of order 200. A F0 track, extracted by the Multi PLL, is depicted by the blue line in Fig. 7.

In future implementations the median filter could be replaced by a statistical model for post-processing purposes like a hidden

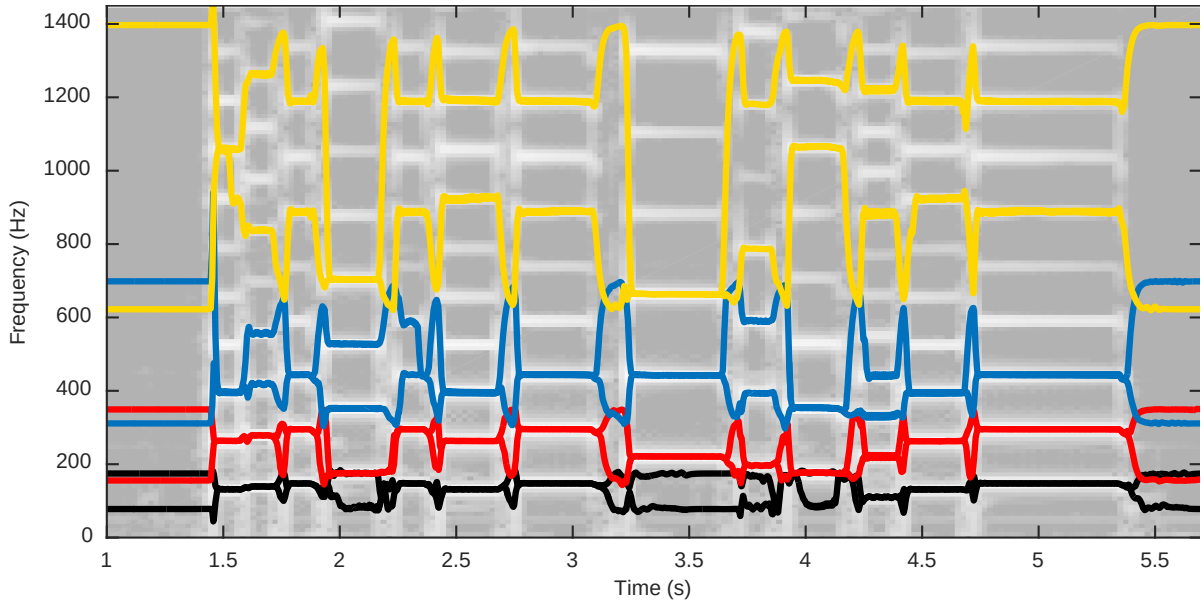


Figure 6: *Spectrogram overlayed with PLL candidate pitch tracks. Pitch tracks with identical color originate from a PLL pair and depend on the same subband signal.*

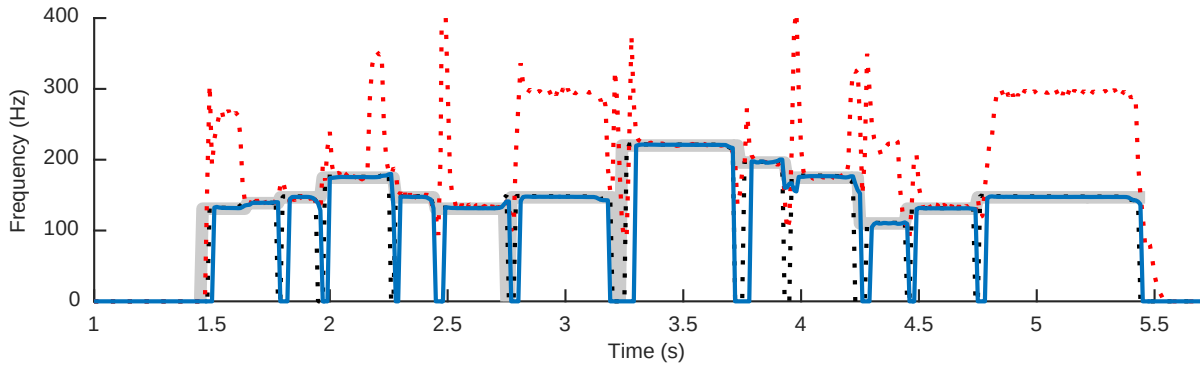


Figure 7: *Plot of the the annotated ground truth (grey), Multi PLL pitch track (blue), Single PLL pitch track (dotted red) and PYIN pitch track (dotted black).*

markov model. The first 4 extracted features could therefore be used to derive observation probabilities, while the pitch slope could be considered by given transition probabilities. In addition to that the algorithm could be optimized by an explicit weighting of the extracted features.

3. EVALUATION

The presented pitch tracker, its predecessor [4] and the PYIN [1] algorithm (with Beta distribution mean of 0.15) are evaluated using the IDMT-SMT-GUITAR dataset [1]. All monophonic guitar lick recordings (one channel RIFF WAVE format, at 44.1 kHz, 24 Bit) and corresponding annotations of dataset 2, which are tagged with playing style fingered, picked and muted as well as expression style normal, are used for the comparison. The annotations deliver pitch information wrapped in the form of note events. A note event contains data such as pitch quantized to midi notes, note onset and note offset times. According to this data, a reference pitch track (ground truth) is generated by

$$f_m = 2^{\frac{(m-69)}{12}} \cdot 440 \text{ Hz}, \tag{24}$$

in order to convert midi notes to frequency values. The tracks of all pitch detectors and the annotation are downsampled to a sampling frequency of 100 Hz. In Fig. 7 pitch tracks of the annotation and the 3 estimators are overlayed. The PYIN pitch tracker shows the fastest reaction to onset events and is capable of determining the fundamental frequency even during the attack phase of a tone. The Multi PLL algorithm avoids the overtone errors of its predecessor but exhibits delayed onset detections compared to the PYIN.

[1] `http://www.idmt.fraunhofer.de/en/business_units/smt/guitar.html`

3.1. Detection rates

Measures of binary classification as Precision and Recall cannot be applied offhand in this case. Therefore they have to be defined appropriately. In dependence on [1] we define Recall as the proportion of actually voiced samples (according to ground truth), which the extractor recognizes as voiced and tracks with a maximum deviation of ± 50 Cent. Precision is defined as the proportion of pitch samples marked by the extractor as voiced which have a maximum deviation of ± 50 Cent from reference pitch. F-Measure can be derived as the geometrical mean of Precision and Recall.

Table 1: *Detection scores for the examined pitch trackers based on the IDMT-SMT-GUITAR dataset.*

Pitch tracker	F-Measure	Precision	Recall
Multi PLL	82.63%	92.24%	75.87%
Single PLL	56.37%	55.83%	57.22%
PYIN	88.23%	90.36%	86.41%

Table 1 shows that the F-Measure of the Multi PLL has increased by approximately 26% compared to its predecessor. Especially Precision has risen by 36.41% due to less octave errors and increased pitch stability. Recall has improved by 18.65% . However, the Multi PLL algorithm misses the F-Measure of the PYIN by 5.60%. This is mainly due to a Recall which lies 10.54% below the PYINs counterpart. Solely the Precision score of the PYIN has been exceeded by 1.88%. Overall, the detection rates of the Multi PLL show that most of the errors are missing detections which are reflected in the low Recall score. These errors are mainly caused by delayed onset recognitions and undetected tones of muted licks.

The PYINs F-Measure of 88.23% for the IDMT-SMT-GUITAR dataset is lower than expected compared to other findings [1]. Recall mainly suffers from unvoiced detections for note on/off stages and muted tones. Precision is decreased by false voiced detections. Nevertheless, the PYIN algorithm depicts the best pitch tracker to date and persuades with good detection rates and exactness which is the reason why it is used as a comparative basis in the following subsection.

3.2. Exactness of pitch estimates

The examined pitch trackers should not only be able to quantize pitch to a semitone grid. If a pitch tracker is applied to audio signals emitted by vocal chords or instruments that enable intonation in between the semitone grid (e.g. fretted and fret less stringed instruments, winds etc.), it can be necessary to estimate the fundamental frequency as precise as possible. The annotations provided by the database don't qualify for the evaluation of exactness of pitch estimates due to its coarse frequency quantization. Because of this and the fact that the PYIN algorithm has already been applied to define the pitch ground truth for the Medley DB [8], its pitch track is in the following regarded as comparative basis.

In order to determine the exactness of the PLL-based pitch trackers, solely pitch samples which have been tagged as voiced and correct (in ± 50 Cent range) for all three pitch trackers are used. For each of the pitch samples of the Single PLL and the Multi PLL, the deviation in Cent to the corresponding samples of the PYIN pitch tracks is determined. Based on these deviations, the mean, the standard deviation (STD) and the median are calculated and a histogram is generated which is depicted in Fig. 8.

Table 2: *Statistical measures concerning the deviation between the PLL-based pitch trackers and PYIN.*

Pitch tracker	Mean (Cent)	STD (Cent)	Median (Cent)
Multi PLL	-2.57	7.21	-1.75
Single PLL	-2.02	16.44	-1.31

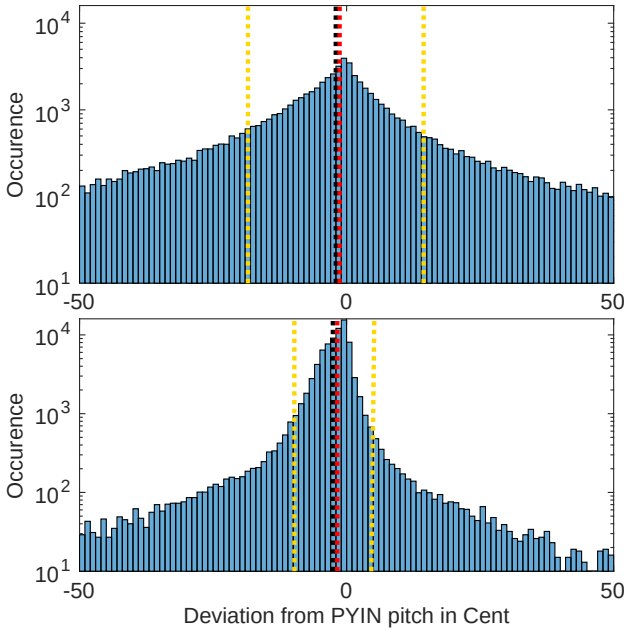


Figure 8: *Histogram of the pitch deviations in the range of ± 50 Cent between PYIN and Single PLL (top) as well as PYIN and Multi PLL (bottom). The dotted lines depict mean (black), median (red) and standard deviation (yellow).*

Table 2 reveals that the mean of the pitch deviation vector is negatively biased for both pitch trackers, which indicates a tendency towards understated pitch estimates. This tendency is more pronounced in the pitch estimates of the Multi PLL algorithm, which is also supported by the median. The standard deviation of the Single PLL is more than twice as high as the standard deviation of the Multi PLL which indicates a considerably pronounced spread around the mean value. The difference concerning the spread is clearly visible in the histograms. The slope of the Multi PLL histogram is much steeper to both sides than the slope of the Single PLL histogram. This deviating characteristic is caused by heavier oscillations of the Single PLLs pitch track which are related to the bigger frequency shift accounted for by the larger K^d value. Consequently phase errors are amplified which lead to a more sensitive and unstable tracking.

Further the histogram indicates that 77.84 % of Multi PLL pitch samples lie below the reference pitch track. It is assumed that the reason for this is a combination of the pitch trackers workings and the inharmonic nature of the instrument whose emitted

audio signal is analyzed. If an instrument has inharmonic properties, its overtones are non integer multiples of the fundamental frequency. This is particularly recognizable for stringed instruments like guitar and depends on the stiffness of actual strings [9]. The PYIN algorithm is based on the difference function which comprises the autocorrelation function [10]. Therefore, the periodicity of the whole signal, including overtones, is evaluated in order to determine the corresponding frequency. This frequency is slightly higher than the frequency of the fundamental depending on the present level of inharmonicity. The thicker the core of the string, the bigger the resulting inharmonicity [9]. Therefore, especially for the bass strings of a guitar the pitch results of the PYIN algorithm tend to be higher than the actual frequency of the fundamental. One can state that the definition of pitch varies between PLL- and autocorrelation-based pitch trackers. While the PLL-based implementations track the frequency of the fundamental, the PYIN algorithm tracks the frequency which is characterized by the periodicity of the overall signal.

The biases of the PLL-based estimators are similar and most certainly emerge from the differing pitch definition of the PYIN estimator. Therefore, the exactness of pitch estimates of PLL-based algorithms depends mainly on the spread, which is less definitive for the Multi PLL. As a result the approach described in this paper provides more precise pitch estimates than its predecessor. In order to evaluate the exactness of PYIN and Multi PLL pitch estimates the nature of pitch has to be specified more precisely and the dataset to be used needs to provide a ground truth with a finer frequency quantization.

4. CONCLUSION

The goal of this study was to develop an efficient monophonic pitch tracker, utilizing multiple PLLs, which delivers improved robustness against overtone errors and enhanced pitch track stability. The access to multiple, variably parameterized PLLs allows a much more comprehensive view of the presence, intensity and positioning of partials than a single PLL could deliver. Based on this information conclusions can be made that result in a substantial improvement of the detection rate. For the IDMT-SMT-GUITAR dataset the Multi PLL estimator has achieved a F-Measure of 82.63 %, which corresponds to an improvement of 26.26% compared to the results of its predecessor.

In future implementations especially the Recall could be improved by optimizing the individual PLL parameters and the weighting of features in order to enhance the timing and rate of the voiced detections. A problem that remains is the detection of pitch for tones with missing fundamentals. Pitch trackers which exploit the periodicity of the overall signal like autocorrelation-related approaches are capable of detecting the perceived pitch of these tones. The approach presented in this study however, requires additional logic which considers the frequency spacing of partials in order to provide this functionality. In addition to that a statistical model could be implemented to further improve the correctness and continuity of the overall pitch track. Finally, to provide real-time-capability for future implementations, the filterbank needs to be modified. An extension of the presented approach for application to polyphonic audio signals seems not practical since the configuration of the filterbank and the PLLs exploits the frequency composition of monophonic audio signals.

5. REFERENCES

[1] M. Mauch and S. Dixon, "PYIN: A fundamental frequency estimator using probabilistic threshold distributions," in *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014.

[2] Edward W. Large, "Beat tracking with a nonlinear oscillator," 1995.

[3] P. Pelle and C. Estienne, "A robust pitch detector based on time envelope and individual harmonic information using phase locked loops and consensual decisions," in *Proc. of the IEEE Int. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014.

[4] U. Zölzer, S.V. Sankarababu, and S. Moller, "PLL-based pitch detection and tracking for audio signals," in *Proc. of the Int. Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, July 2012.

[5] U. Zölzer, "Pitch-based digital audio effects," in *Proc. of the Int. Symposium on Communications Control and Signal Processing (ISCCSP)*, May 2012.

[6] Dimitrios Giannoulis, Michael Massberg, and Joshua D Reiss, "Digital dynamic range compressor design–A tutorial and analysis," *Journal of the Audio Engineering Society*, vol. 60, no. 6, pp. 399–408, 2012.

[7] Floyd Martin Gardner, *Phaselock techniques; 2nd ed.*, Wiley, 1979.

[8] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "MedleyDB: A multitrack dataset for annotation-intensive MIR research," in *Proc. of the Int. Society for Music Information Retrieval Conference*, Oct. 2014.

[9] Manfred Zollner and Hochschule Regensburg, "The physics of e-guitars: Vibration, voltage, sound wave, timbre (Physik der Elektrogitarre)," Available in german language at https://hps.hs-regensburg.de/~elektrogitarre/pdfs/gesamt.pdf, accessed February 26, 2016.

[10] Alain de Cheveigné and Hideki Kawahara, "YIN, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

PIECEWISE DERIVATIVE ESTIMATION OF TIME-VARYING SINUSOIDS AS SPLINE EXPONENTIAL FUNCTIONS

Wen Xue
Samsung Electronics

xue.wen@samsung.com

ABSTRACT

This paper discusses the estimation of non-stationary sinusoidal parameters. We formulate a piecewise version of the distributive derivative algorithm, which is used to analyse non-stationary sinusoidal signals and estimate their frequencies and log amplitude derivatives over a long duration as spline functions, and apply this algorithm for the estimation of instantaneous frequencies, amplitudes and phase angles. Test results show that the piecewise derivative algorithm provides better estimation than the previous non-piecewise version at lower computation cost.

1. INTRODUCTION

The sinusoidal modelling technique [1][2] uses slow-varying sinusoids to model the "deterministic" parts of audio and speech, including harmonic partials of human/animal vocals, string, wind and brass instruments, and harmonic or inharmonic partials of percussion and electric instruments.

Complex exponential functions, or *complex sinusoids*, are of the form $e^{r(t)}$, where $r(t)=p(t)+j\varphi(t)$, $p(t)$, $\varphi(t)\in\mathbf{C}^1(\mathbf{R})$ [1], is the *exponent*. $e^{p(t)}$ and $\varphi(t)$ are known as the amplitude and phase angle, and $\omega(t)=d\varphi(t)/dt$ the angular frequency. We say $e^{r(t)}$ is slow-varying if $p(t)$ and $\omega(t)$ vary slowly with time t. Slow-varying sinusoids have narrow short-time bandwidths [3], allowing concurrent sinusoids be accessed independently via adequate bandpass mechanism, as long as their frequencies stand apart. In particular, the real sinusoid $e^{p(t)}\cos\varphi(t)$ can be accessed via $e^{r(t)}$.

Many sinusoid estimators derived in the past estimate sinusoidal parameters at a point from waveform data in its close vicinity. Several early estimators assuming short-term stationarity of amplitude and frequency were summarized in [4]. As real-world sinusoids are rarely stationary, more complex short-term parametric models, e.g. [5]-[9], were proposed, leading to the highly flexible estimation of arbitrary complex polynomial exponential functions [8][9], and more recently to an even higher degree of freedom by allowing an arbitrary complex polynomial multiplier on top of it [10].

A second family of algorithms addresses long-term amplitude and frequency modulations, e.g. with the spline model [11][12]. While most short-term algorithms engage closed-form computation, the long-term methods depend on iterative optimization, and are likely to suffer high computation cost and convergence to local optima. However, upon successful convergence the long-term constraints help to fight overfitting and improve robustness.

In this paper we show that the distributive derivative approach of [9] can be formulated to address long-term amplitude and frequency modulations using a spline exponential model. This leads to a non-iterative algorithm for the long-term estimation of sinusoids, which combines the simplicity of the derivative method with the robustness of long-term parametric modelling.

The rest of this paper is arranged as follows. Section 2 briefly reviews the distributive derivative algorithm; section 3 derives the piecewise formulation of the derivative algorithm, which estimates frequency and log amplitude derivative as splines; section 4 presents an amplitude and phase handling scheme that helps make long-term and local estimates consistent; section 5 presents test results on a synthesized test set, and section 6 presents a real world example.

2. THE DISTRIBUTIVE DERIVATIVE METHOD

The distributive derivative algorithm, or derivative algorithm for short, estimates a time-varying exponent $r(t)$ of $s(t)=e^{r(t)}$ by taking the derivative of $s(t)$. Early examples of the method include [13] and [14] used for estimating stationary sinusoids and linear chirps. Later the method was generalized by current author and Sandler [8] and Betser [9] to estimate $r(t)$ as the linear combination of differentiable functions. This section gives a brief review of this algorithm, following the formulation of [9].

2.1. General framework (after [9])

Let $h_1(t)$, …, $h_M(t)\in\mathbf{C}^1(\mathbf{R})$ be M linearly independent complex functions and $v_1(t)$, …, $v_I(t)\in\mathbf{C}^1(\mathbf{R})$ be I linearly independent complex functions, and let all these functions have a common compact support $D=[d_1,d_2]$, d_1, $d_2\in\mathbf{R}$. We consider a complex sinusoid

$$s(t)=e^{r(t)} \tag{1}$$

on the interval D, so that the derivative of $r(t)$ is a linear combination of $h_1(t),\ldots h_M(t)$:

$$r'(t)=\sum_{m=1}^{M}\lambda_m h_m(t)\equiv\mathbf{h}(t)^{\mathsf{T}}\boldsymbol{\lambda}\,,\ t\in D \tag{2}$$

where $\mathbf{h}(t)=(h_1(t),\ \ldots,\ h_M(t))^{\mathsf{T}}$, $\boldsymbol{\lambda}_M=(\lambda_1,\ \ldots,\ \lambda_M)^{\mathsf{T}}\in\mathbf{C}^M$ and the superscript $^{\mathsf{T}}$ denotes matrix and vector transpose. Now we consider this problem: given $s(t)$ and $\mathbf{h}(t)$, how do we find $\boldsymbol{\lambda}$?

We take the derivatives of both sides of (1) and substitute (2):

$$s'(t)=s(t)\cdot\mathbf{h}(t)^{\mathsf{T}}\boldsymbol{\lambda} \tag{3}$$

Taking the inner products of both sides of (3) with functions v_1, …, v_I we get

$$\left\langle s',v_i\right\rangle_{\mathrm{c}}=\left\langle s\mathbf{h}^{\mathsf{T}},v_i\right\rangle_{\mathrm{c}}\boldsymbol{\lambda}\,,\ i=1,\ \ldots,\ I \tag{4}$$

or

$$\left\langle s',\mathbf{v}\right\rangle_{\mathrm{c}}=\left\langle s\mathbf{h}^{\mathsf{T}},\mathbf{v}\right\rangle_{\mathrm{c}}\boldsymbol{\lambda}\,, \tag{5}$$

[1] $\mathbf{C}^m(\mathbf{R})$: space of real functions with continuous m^{th}-order derivatives.

where $\mathbf{v}(t)=(v_1(t), \ldots, v_l(t))^\mathsf{T}$ and the continuous inner product operator $\langle \bullet,\bullet \rangle_\mathrm{c}$ is defined for functions and function vectors as

$$\langle x, y \rangle_\mathrm{c} = \int y^*(t)x(t)dt \text{ and } \langle \mathbf{x}^\mathsf{T}, \mathbf{y} \rangle_\mathrm{c} = \int \mathbf{y}^*(t)\mathbf{x}^\mathsf{T}(t)dt\,, \tag{6}$$

respectively, where the superscript * denotes complex conjugate. Comparing (3) and (5) we see that $\langle s',\mathbf{v}\rangle_\mathrm{c}$ is the linear combination of vectors $\langle sh_m,\mathbf{v}\rangle_\mathrm{c}$, m=1, …, M, with the *same coefficients* as $r'(t)$ is that of $h_m(t)$. This converts the decomposition of $r'(t)$ in a function space $\{\, \mathbf{h}(t)^\mathsf{T}\boldsymbol{\lambda} \,|\boldsymbol{\lambda}\in\mathbf{C}^M\}$ to that of $\langle s',\mathbf{v}\rangle_\mathrm{c}$ in a vector space without the need for extracting $r'(t)$ explicitly. We call the entries of $\mathbf{h}$ *basis functions* as they form a basis of the vector space above, and the entries of $\mathbf{v}$ *test functions* after [9].

2.2. Discrete computation

If we define discrete inner products for functions and function vectors as

$$\langle x, y \rangle = \sum_{n=0}^{T-1} y^*(n)x(n)\,,\ \langle \mathbf{x}^\mathsf{T}, \mathbf{y} \rangle = \sum_{n=0}^{T-1} \mathbf{y}^*(n)\mathbf{x}^\mathsf{T}(n)\,, \tag{7}$$

then the following discrete version of (5) holds:

$$\langle s', \mathbf{v} \rangle = \langle s\mathbf{h}^\mathsf{T}, \mathbf{v} \rangle \boldsymbol{\lambda} \tag{8}$$

One issue of using (8) to compute $\boldsymbol{\lambda}$ is that $s'(t)$ is not available as input. Both [8] and [9] suggested that if $\mathbf{v}(t)$ is differentiable and vanishes at both ends of D, then $\langle s',\mathbf{v}\rangle_\mathrm{c}$ can be computed as $-\langle s,\mathbf{v}'\rangle_\mathrm{c}$. For discrete computation however, $-\langle s,\mathbf{v}'\rangle$ approximates $\langle s',\mathbf{v}\rangle$ with an error given in [9] as

$$-\langle s, \mathbf{v}' \rangle = \langle s', \mathbf{v} \rangle - \sum_n (s\mathbf{v}^*)'(t)\,|_{t=n} \tag{9}$$

In [8] we pointed out that this error equals the total (Shannon) sampling alias of $(s\mathbf{v}^*)(t)$. To keep the error term in (9) low, $(s\mathbf{v}^*)(t)$ must have negligible spectral energy density above the Nyquist frequency. Practically this is satisfied by choosing $v_i\in\mathbf{C}^2(\mathbf{R})$, $\forall i$, so that $s(t)v_i(t)^*$ is a base-band signal: for example, a Hann-windowed sinusoid tuned to the central frequency of $s(t)$.

In practice the coefficients of (8) may be contaminated by noise in $s(t)$ (observation noise) and $r'(t)$ (modelling noise). As a remedy it is often solved in a least-square sense using

$$\boldsymbol{\lambda} = \left(\langle \mathbf{v}^\mathsf{T}, s\mathbf{h} \rangle \langle s\mathbf{h}^\mathsf{T}, \mathbf{v} \rangle \right)^{-1} \langle \mathbf{v}^\mathsf{T}, s\mathbf{h} \rangle \langle s', \mathbf{v} \rangle\,, \tag{10}$$

provided that $\langle \mathbf{v}^\mathsf{T},s\mathbf{h}\rangle\langle s\mathbf{h}^\mathsf{T},\mathbf{v}\rangle$ is invertible.

The discrete inner products can also be written as matrix multiplications. Define $\mathbf{t}=(0,\ldots,T\text{-}1)^\mathsf{T}$, $\mathbf{s}=\mathrm{s}(\mathbf{t})=(\mathrm{s}(0),\ldots,\mathrm{s}(T\text{-}1))^\mathsf{T}$, $\mathbf{s}'=s'(\mathbf{t})$, $\mathbf{H}=\mathbf{h}(\mathbf{t}^\mathsf{T})=(\mathbf{h}(0),\ldots,\mathbf{h}(T\text{-}1))$, $\mathbf{V}=\mathbf{v}(\mathbf{t}^\mathsf{T})$, then (8) has the matrix formulation

$$\mathbf{V}^*\mathbf{s}' = \mathbf{V}^* diag(\mathbf{s})\mathbf{H}^\mathsf{T}\boldsymbol{\lambda}\,. \tag{11}$$

2.3. Amplitude and phase angle

The derivative algorithm only estimates $r'(t)$. To complete the estimation of $s(t)=e^{r(t)}$ we still need to estimate $r(0)$, which represents global amplification and phase shift. The least-square estimate of $r(0)$ is computed by correlation with a unit-amplitude zero-phase sinusoid with exponent derivative $r'(t)$:

$$r(0) = \log \frac{\langle s, \tilde{s} \rangle}{\langle \tilde{s}, \tilde{s} \rangle}\,,\ \tilde{s}(t) = \exp \int_0^t \mathbf{h}(\tau)^\mathsf{T}\boldsymbol{\lambda} d\tau \tag{12}$$

The instantaneous angular frequency, amplitude and phase angle at 0 are $\mathrm{Im}\{r'(0)\}$, $e^{\mathrm{Re}\{r(0)\}}$ and $\mathrm{Im}\{r(0)\}$, respectively.

3. PIECEWISE DERIVATIVE METHOD

The derivative algorithm above assumes that $r'(t)$ follows the same parametric model over the whole duration. [9] took $\mathbf{h}(t)$ as a polynomial basis, i.e. $\mathbf{h}(t)=(t^{M-1},t^{M-2},\ldots,1)^\mathsf{T}$, and asserted the validity of the signal model (2) by the Taylor expansion of $r'(t)$. Since Taylor expansions are usually accurate only in the neighbourhood of the origin, this model is not suitable for long duration. On the other hand, piecewise polynomials, or splines, can model arbitrary functions of arbitrary lengths using translations of the same local model. In this section we adapt the derivative algorithm to estimate $r'(t)$ as a spline function. To distinguish between the original and adapted versions, we call them *local* and *piecewise* derivative algorithms, respectively.

3.1. General framework

We limit our discussion to splines with uniformly placed *knot points*, and let them be 0, T, …, LT. We assume that we know the waveform of a slow-varying sinusoid on $[0,LT]$ and roughly know its instantaneous frequencies at 0, T, … LT, both of which can be obtained using a sinusoid tracker, e.g. [1].

We formulate $r'(t)$ as a spline function by expressing it as a $(M\text{-}1)^{\text{th}}$-order polynomial on each $[lT,lT+T)$, l=0, …, L-1:

$$r'(lT+t) = \mathbf{h}(t)^\mathsf{T}\boldsymbol{\lambda}_l\,,\ 0\le t<T,\ l=0, \ldots, L\text{-}1, \tag{13}$$

where $\mathbf{h}(t)=(t^{M-1},t^{M-2},\ldots,1)^\mathsf{T}$, $\lambda_l\in\mathbf{C}^M$. Vectors $\boldsymbol{\lambda}_0$, ..., $\boldsymbol{\lambda}_{L-1}$ contain the polynomial coefficients on the L segments, which are constrained by boundary conditions specific to the spline type. For example, the continuity of $r'(t)$ at lT requires

$$\mathbf{h}(T)^\mathsf{T}\boldsymbol{\lambda}_{l-1} = \mathbf{h}(0)^\mathsf{T}\boldsymbol{\lambda}_l\,,\ l=1, \ldots, L\text{-}1. \tag{14}$$

In this paper we consider the linear interpolative formulation[2] of splines, which expresses $\boldsymbol{\lambda}_l$ as a linear function of the spline samples at the knot points, i.e. $\mathbf{r}'=r'(\mathbf{t})=(r'(0),\ldots,r'(LT))^\mathsf{T}$:

$$\boldsymbol{\lambda}_l=\mathbf{A}_l\,\mathbf{r}',\ l=0, \ldots, L\text{-}1. \tag{15}$$

$\mathbf{A}_0$, …, $\mathbf{A}_{L-1}$ are real matrices depending on L, T and the spline type. Substituting (15) into (13) we get

$$r'(lT+t) = \mathbf{h}(t)^\mathsf{T}\mathbf{A}_l\mathbf{r}'\,,\ 0\le t<T,\ l=0, 1, \ldots, L\text{-}1. \tag{16}$$

We call (13)~(16) the spline exponential model. A specific spline interpolator has a linear interpolative formulation (15) as long as all its boundary conditions are linear in terms of $r'(t)$ and its derivatives. The matrices $\mathbf{A}_0$, …, $\mathbf{A}_{L-1}$ for linear, quadratic and cubic spline interpolators are derived in the Appendix.

We further define a function vector $\boldsymbol{\rho}(t)=(\rho_0(t),\ldots,\rho_L(t))^\mathsf{T}$ by

$$\boldsymbol{\rho}(lT+t)^\mathsf{T} = \mathbf{h}(t)^\mathsf{T}\mathbf{A}_l\,,\ 0\le t<T,\ l=0, 1, \ldots, L\text{-}1. \tag{17}$$

If both $\mathbf{h}(t)$ and $\mathbf{A}_l$ are known then so is $\boldsymbol{\rho}(t)$. The independent coefficients $\mathbf{r}'$ contribute to $r'(t)$ through the entries of $\boldsymbol{\rho}(t)$:

$$r'(t) = \boldsymbol{\rho}(t)^\mathsf{T}\mathbf{r}'\,,\ 0\le t<LT. \tag{18}$$

Eq. (18) expresses the linear piecewise exponential model as a special case of the linear exponential model (2), with $\boldsymbol{\rho}(t)$ replacing $\mathbf{h}(t)$ as the basis. We can therefore construct a linear system similar to (8)

$$\langle s', \mathbf{v} \rangle = \langle s\boldsymbol{\rho}^\mathsf{T}, \mathbf{v} \rangle \mathbf{r}'\,. \tag{19}$$

where $\mathbf{v}(t)=(v_1(t), \ldots, v_{\dim(\mathbf{v})}(t))^\mathsf{T}$ is the vector of test functions. If $\langle \mathbf{v}^\mathsf{T},s\boldsymbol{\rho}\rangle\langle s\boldsymbol{\rho}^\mathsf{T},\mathbf{v}\rangle$ is invertible, then (19) has a least square solution

[2] B-spline formulation is also a possibility. The interpolative expression is chosen because it preserves the task's degree of freedom and allows us draw links with previous methods, like (18).

$$\mathbf{r}' = \left(\left\langle \mathbf{v}^{\mathsf{T}}, s\boldsymbol{\rho}\right\rangle\left\langle s\boldsymbol{\rho}^{\mathsf{T}}, \mathbf{v}\right\rangle\right)^{-1}\left\langle \mathbf{v}^{\mathsf{T}}, s\boldsymbol{\rho}\right\rangle\left\langle s', \mathbf{v}\right\rangle . \tag{20}$$

Equations (19) and (20) give the *piecewise derivative algorithm* for estimating sinusoids with model (16). Notice that although we focus on spline exponentials in this paper, the algorithm itself does not require $\mathbf{h}(t)$ to be a polynomial basis, and therefore can be applied to a larger class of piecewise models of $r'(t)$, as long as they can be formulated as (16).

3.2. Computing coefficient matrix $\langle s\boldsymbol{\rho}^{\mathsf{T}},\mathbf{v}\rangle$

While one can always compute $\boldsymbol{\rho}(t)$ explicitly with (17), the actual estimation of $\mathbf{r}'$ only requires computing the matrix $\langle s\boldsymbol{\rho}^{\mathsf{T}},\mathbf{v}\rangle$, which has a piecewise implementation:

$$\begin{aligned}\left\langle s\boldsymbol{\rho}^{\mathsf{T}}, \mathbf{v}\right\rangle &= \sum_{t=0}^{LT-1}\mathbf{v}^*(t)s(t)\boldsymbol{\rho}(t)^{\mathsf{T}} = \sum_{l=0}^{L-1}\sum_{t=0}^{T-1}\mathbf{v}^*(lT+t)s(lT+t)\boldsymbol{\rho}(lT+t)^{\mathsf{T}} \\ &= \sum_{l=0}^{L-1}\sum_{t=0}^{T-1}\mathbf{v}_{[l]}^*(t)s_{[l]}(t)\mathbf{h}(t)^{\mathsf{T}}\mathbf{A}_l = \sum_{l=0}^{L-1}\left\langle s_{[l]}\mathbf{h}^{\mathsf{T}}, \mathbf{v}_{[l]}\right\rangle\mathbf{A}_l\end{aligned} \tag{21}$$

where $\mathbf{v}_{[l]}(t)=\mathbf{v}(lT+t)$ and $s_{[l]}(t)=s(lT+t)$ represent the parts of $\mathbf{v}(t)$ and $s(t)$ sampled over interval $[lT,lT+T)$.

3.3. Separate models for amplitude and frequency

We let $\mathbf{h}(t)$ be real and replace (15) with

$$\boldsymbol{\lambda}_l = \mathbf{B}_l\mathbf{p}' + j\mathbf{C}_l\boldsymbol{\omega}\,,\quad l=0,\ \ldots,\ L\text{-}1, \tag{22}$$

where $\mathbf{p}'=\mathrm{Re}\{\mathbf{r}'\}$ and $\boldsymbol{\omega}=\mathrm{Im}\{\mathbf{r}'\}$ contain the amplitude growth rates and angular frequencies at 0, T, ..., LT, respectively, and $\mathbf{B}_l$ and $\mathbf{C}_l$, l=0, ..., L-1, are real matrices that implement linear interpolations of $\mathbf{p}'$ and $\boldsymbol{\omega}$ via (13). This formulation allows the amplitude and frequency be modelled with independent spline types, and leads to the real implementation of the piecewise derivative method:

$$\begin{pmatrix}\mathrm{Re}\left\langle s\boldsymbol{\rho}_{\mathbf{B}}^{\mathsf{T}}, \mathbf{v}\right\rangle & -\mathrm{Im}\left\langle s\boldsymbol{\rho}_{\mathbf{C}}^{\mathsf{T}}, \mathbf{v}\right\rangle \\ \mathrm{Im}\left\langle s\boldsymbol{\rho}_{\mathbf{B}}^{\mathsf{T}}, \mathbf{v}\right\rangle & \mathrm{Re}\left\langle s\boldsymbol{\rho}_{\mathbf{C}}^{\mathsf{T}}, \mathbf{v}\right\rangle\end{pmatrix}\begin{pmatrix}\mathbf{p}' \\ \boldsymbol{\omega}\end{pmatrix} = \begin{pmatrix}\mathrm{Re}\left\langle s', \mathbf{v}\right\rangle \\ \mathrm{Im}\left\langle s', \mathbf{v}\right\rangle\end{pmatrix}, \tag{23}$$

where

$$\boldsymbol{\rho}_{\mathbf{B}}(lT+t)^{\mathsf{T}}=\mathbf{h}(t)^{\mathsf{T}}\mathbf{B}_l,\ \boldsymbol{\rho}_{\mathbf{C}}(lT+t)^{\mathsf{T}}=\mathbf{h}(t)^{\mathsf{T}}\mathbf{C}_l,\ 0\le t<T,\ l=0,\ \ldots,\ L\text{-}1. \tag{24}$$

A least square solution to (23) is computed in the same way as to (8) using real arithmetic only. $\langle s\boldsymbol{\rho}_{\mathbf{B}}^{\mathsf{T}}, \mathbf{v}\rangle$ and $\langle s\boldsymbol{\rho}_{\mathbf{C}}^{\mathsf{T}}, \mathbf{v}\rangle$ are computed using (21) from the same set of intermediate results $\langle s_{[l]}\mathbf{h}^{\mathsf{T}},\mathbf{v}_{[l]}\rangle$, l=0, …, L-1.

3.4. Framing of test functions v(t)

In this section we present a specific construction of the test functions $\mathbf{v}(t)$ using overlapping frames. We wish the interval $[0,LT]$ be uniformly covered by $\mathbf{v}(t)$, so that no part of $s(t)$ is ignored or overemphasized. It is intuitive to divide this interval into uniformly spaced frames and apply the same subset of test functions to every frame. And as test functions must vanish at both ends, it is necessary to have overlapping frames. In this paper we place frame centres at the spline knots, i.e. T, $2T$, …, $(L\text{-}1)T$, with 50% overlap between adjacent frames. This gives L-1 frames of length $2T$ over the whole duration (Fig.1a). Given this framing scheme we can write $\mathbf{v}(t)^{\mathsf{T}}=[\mathbf{v}_1(t)^{\mathsf{T}}\ \mathbf{v}_2(t)^{\mathsf{T}}\ \ldots\mathbf{v}_{L-1}(t)^{\mathsf{T}}]$, in which all entries of $\mathbf{v}_l(t)$, l=1, …, L-1, are supported on $[lT\text{-}T,lT+T]$, and are time-shifted versions of the same *local* test functions:

$$\mathbf{v}_l(lT+t)=\mathbf{v}_1(T+t),\ l=1,\ ...,\ L\text{-}1,\ -T\le t\le T. \tag{25}$$

Eq.(25) reduces the design of $\mathbf{v}(t)$ to that of $\mathbf{v}_1(t)$, for which the test functions in the local derivative method (section 2), e.g. windowed Fourier atoms, can be used unchanged.

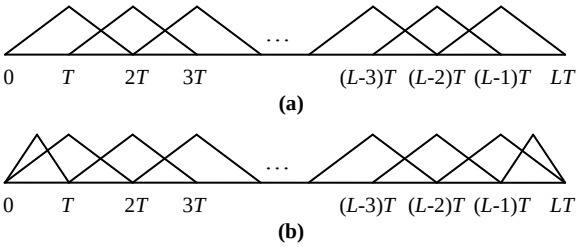


Figure 1 . Framing without and with boundary frames
(a) without boundary frames; (b)with boundary frames

As Figure1(a) shows, the vanishing requirement of test functions inevitably leads to poor frame coverage near 0 and LT. To make better use of data in these parts we add two *boundary frames* supported on $[0,T]$ and $[LT\text{-}T,LT]$, respectively, and construct test functions on these frames as

$$\mathbf{v}_0(t)=\mathbf{v}_L(LT\text{-}T+t)=\mathbf{v}_1(2t), \tag{26}$$

as illustrated in Figure1(b).

3.5. Computational complexity

We examine the number of complex multiplications in the general formulation of the piecewise derivative algorithm, finishing with solving (20) using boundary frames as shown in Figure1(b). Let I be the number of test functions per frame. The computation considered includes computing the coefficients of (19), and solving (19) using (20).

Using boundary frames, $2I$ test functions are non-zero on each of the L sections of size T. Accordingly, $2TI(M+1)$ multiplications are needed to compute $\langle s_{[l]}\mathbf{h}^{\mathsf{T}},\mathbf{v}_{[l]}\rangle$ for each l. $L(L+1)IM$ more multiplications are needed to compute $\langle s\boldsymbol{\rho}^{\mathsf{T}},\mathbf{v}\rangle$ using (21). Computing $\langle s',\mathbf{v}\rangle$ requires $2LTI$ multiplications, i.e. the total length of all test functions. The total number of multiplications for computing (19) sums up to $2LT(M+2)I+L(L+1)MI$. Computing $\langle \mathbf{v}^{\mathsf{T}},s\boldsymbol{\rho}\rangle\langle s',\mathbf{v}\rangle$ and $\langle \mathbf{v}^{\mathsf{T}},s\boldsymbol{\rho}\rangle\langle s\boldsymbol{\rho}^{\mathsf{T}},\mathbf{v}\rangle$ require LNI and $L(L+1)^2I/2$ multiplications respectively; solving (20) by Gaussian elimination requires $L(L+1)(2L+7)/6$.

In comparison, if we apply the local derivative algorithm over $[0,LT]$ with boundary frames, i.e. applying the algorithm to each of the L-1 frames of size $2T$ plus 2 frames of size T, then the coefficients of (8) are computed L+1 times for a total signal duration of $2LT$, and the linear system (8) is solved L+1 times.

Table 1. Complexity of piecewise and local derivative algorithms

	piecewise derivative alg.	local derivative alg.
computing coefficients	$2LTI(M+2)+L(L+1)MI$	$2LTI(M+2)$
solving linear system(s)	$L^3(\mathrm{I}/2+1/3)$	$LM^2(I+M/3)$

Table 1 compares the complexity of the two derivative methods, in which we have ignored less significant terms. In typical applications M and I are small numbers (usually below 10), T is a few orders of magnitude larger, while L has a flexible range. As long as L is not too large, the complexity of both algorithms are dominated by $2LTI(M+2)$. In this case the piecewise derivative algorithm saves computation by allowing I be as small as 1, while $I\ge M$ must be satisfied in the local derivative algorithm. This benefit will be lost when L grows near $M\sqrt{2T}$, as the extra computation spent on solving (20) eventually outgrows the saving.

4. AMPLITUDE AND PHASE

Like its local counterpart, the piecewise derivative algorithm only estimates $r'(t)$. An alternative algorithm, such as the correlation method (12) in 2.3, must be employed to determine the global amplification and phase shift.

While in theory estimating $r(t)$ at any point, e.g. $r(0)$, is enough for reconstructing $r(t)$ by integrating $r'(t)$, doing so accumulates potential errors over time. As piecewise models are designed for long signals, the accumulated error can get dramatic. For this reason we propose to estimate $r(t)$ locally from short intervals at various measurement points, then adjust the $r'(t)$ estimates to fit the local $r(t)$ estimates. We present the details using the separated formulation (22).

4.1. Local estimation

We estimate $r(t)$ at knots lT, l=0, …, L, by applying the correlation method (12) to a short interval near lT:

$$r_l = \tilde{r}(lT) + \log\frac{\langle s, w_l\tilde{s}\rangle}{\langle \tilde{s}, w_l\tilde{s}\rangle}, \quad l=0, \ldots, L, \tag{27}$$

where r_l is the local estimate of $r(lT)$, w_l is a window function on [lT-T,lT+T] for l=1, .., L-1, on [0,T] for l=0, and on [LT-T,LT] for l=L, and

$$\tilde{r}(t) = \int_0^t \mathbf{h}(t)^{\mathsf{T}}(\mathbf{B}_l\mathbf{p}' + j\mathbf{C}_l\boldsymbol{\omega})d\tau\,, \quad \tilde{s}(t) = \exp\tilde{r}(t)\,. \tag{28}$$

While these estimates can be used as they are, in applications like additive synthesis it is desirable that the local estimates of $r(t)$ be coherent with the piecewise estimate of $r'(t)$, i.e. $\exists k_l \in \mathbf{Z}$, l=0, …, L-1, so that

$$\int_{lT}^{lT+T} r'(t)dt = \int_0^T \mathbf{h}(t)^{\mathsf{T}}(\mathbf{B}_l\mathbf{p}' + j\mathbf{C}_l\boldsymbol{\omega})dt = r_{l+1} - r_l + j2k_l\pi\,. \tag{29}$$

This is achievable using an adjustment step described below, which applies a fine tuning to ($\mathbf{p}'$,$\boldsymbol{\omega}$) to satisfy (29).

4.2. Adjustment of ω and p'

Let $\boldsymbol{\omega}$ be the original frequency estimate and $\boldsymbol{\omega}$+$\boldsymbol{\psi}$ be the adjusted estimate. Define

$$\mathbf{u}_l = \int_0^T \mathbf{C}_l^{\mathsf{T}}\mathbf{h}(t)dt\,, \quad l=0, ..., L\text{-}1. \tag{30}$$

Taking the imaginary part of both sides of (29) we get

$$\mathbf{u}_l^{\mathsf{T}}(\boldsymbol{\omega}+\boldsymbol{\psi}) = \mathrm{Im}\{r_{l+1} - r_l\} + 2k_l\pi\,, \quad l=0, \ldots, L\text{-}1. \tag{31}$$

We start with a phase unwrapping process similar to that of [1] to implicitly determine k_l in (31), which we rewrite as

$$\mathbf{u}_l^{\mathsf{T}}\boldsymbol{\psi} = \mathrm{Im}\{r_{l+1} - r_l\} - \mathbf{u}_l^{\mathsf{T}}\boldsymbol{\omega} + 2k_l\pi\,, \quad l=0, ..., L\text{-}1. \tag{32}$$

$\mathbf{u}_l^{\mathsf{T}}\boldsymbol{\psi}$ is the integral of an interpolation of $\boldsymbol{\psi}$ over [lT,lT+T]. As $\boldsymbol{\psi}$ represents a fine adjustment we choose the $k_l \in \mathbf{Z}$ that minimizes the right side of (32), which becomes

$$\mathbf{u}_l^{\mathsf{T}}\boldsymbol{\psi} = res(\mathrm{Im}\{r_{l+1} - r_l\} - \mathbf{u}_l^{\mathsf{T}}\boldsymbol{\omega}, 2\pi)\,, \quad l=0, ..., L\text{-}1, \tag{33}$$

where $res(x,2\pi)$ is the minimal-absolute residue of x modular 2π. Notice that this choice of k_l coincides with that of [1] motivated by maximizing phase smoothness.

Define $b_l = res(\mathrm{Im}\{r_{l+1} - r_l\} - \mathbf{u}_l^{\mathsf{T}}\boldsymbol{\omega}, 2\pi)$, $\mathbf{b}$=$(b_0, ..., b_{L-1})^{\mathsf{T}}$, and $\mathbf{U}$=($\mathbf{u}_0$,...,$\mathbf{u}_{L-1}$), then (33) is simplified to

$$\mathbf{U}^{\mathsf{T}}\boldsymbol{\psi} = \mathbf{b}. \tag{34}$$

This is a linear system of $\boldsymbol{\psi}$ with L equations and L+1 unknown variables. Since $\boldsymbol{\psi}$ is expected to be small, a simple choice is the minimal-norm solution, given by

$$\boldsymbol{\psi} = \mathbf{U}(\mathbf{U}^{\mathsf{T}}\mathbf{U})^{-1}\mathbf{b}. \tag{35}$$

Finally the adjustment of $\boldsymbol{\omega}$ is completed by

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + \boldsymbol{\psi}\,. \tag{36}$$

It is trivial to verify the adjusted $\boldsymbol{\omega}$ satisfies (31).

Adjustment of $\mathbf{p}'$ follows the same procedure as above, except that the phase unwrapping step is not needed.

5. EXPERIMENTS

We test the proposed algorithm on synthesized test signals and compare it to the local derivative algorithm [9], the QUASAR estimator [11], and original quadratic-interpolated fast Fourier transform magnitude (QIFFT) method [15]. The piecewise derivative algorithm is tested with cubic and linear splines (labelled PD^3 and PD^1), the local derivative algorithm is tested with cubic and linear polynomials (LD^3 and LD^1), while the QUASAR estimator (Q) is piecewise linear by design (as "quadratic phase" means linear frequency).

We use each estimator to estimate parameters of a slow-varying sinusoid, then reconstruct the sinusoid from the estimates with a paired synthesizer. Errors are computed for estimated parameters and for sinusoids synthesized from them. The local estimators LD^3, LD^1 and QIFFT (QIF) do not come with "native" synthesizers for more than one frame. We pair LD^3 and LD^1 with natural cubic and linear spline interpolators respectively, and QIFFT with the original McAulay-Quatieri phase-aligned synthesizer [1]. For the reconstruct errors we also test a few *reference* systems that synthesize from the true parameters. Reference system using cubic and linear spline synthesizers are labelled R^3 and R^1 respectively. [11] provides its own least-square interpolator, which we also use as a reference ("R^Q").

Both derivative algorithms use boundary frames. From 3.4, the standard frame size is 2T; standard frame hop and boundary frame size are T. Two test functions per frame are used in the piecewise derivative algorithms (PD^3, PD^1); four per frame are used in the local derivative algorithms (LD^3, LD^1).

For each test signal we use its waveform and reference frequencies at measurement points 0, T, …, LT as inputs. Test signals over [0,LT] are supplied to the derivative algorithms; test signals over [-T,LT+T] are supplied to the Q and QIF estimators which do not fully handle boundary frames. Reference frequencies are rough frequency estimates to tell the estimators where in the T-F plane to expect the sinusoids. We provide these by rounding ground truth frequencies to nearest whole DFT bins, 1bin=1/2T. The QUASAR estimator (Q) uses a single reference frequency, which we supply with the average ground value.

For test functions of both derivative algorithms we follow [9] and use Hann-windowed complex sinusoids whose frequencies are tuned to the whole DFT bins closest to the reference frequencies.

5.1. Test set

Frequency and amplitude-modulated sinusoids are used as test signals. The frequencies are modulated by a sinusoidal modulator of amplitude A_M and period T_M; the amplitudes are modulated by passing the modulated frequency through one of three real transfer functions: a linear function $H_1(f)$ that assigns the central frequency a medium amplitude, and two quadratic functions $H_2(f)$ and $H_3(f)$ that assign the central frequency either the minimal or

maximal amplitude. We summarize the test set by (37a)~(37f), where f, f_0, A_M are given in bins, and T_M in frames, where 1 bin=1/(2T), 1 frame=T. Constant coefficients in (37d)~(37f) are chosen so that the amplitude modulation depth is 2/3. For all tests we use L=10 and T=1024, so that the length of each test sample is 10240. For this choice of L and T the complexity of both derivative algorithms is dominated by computing $\langle s\boldsymbol{\rho}^{\mathsf{T}},\mathbf{v}\rangle$, so that by using I=2 instead of 4 the piecewise algorithm saves nearly half the computation.

Table 2. Synthesized test set

$$f(t) = f_0 + A_M \cos(\varphi_M + 2\pi t / TT_M), \quad (37\,a)$$

$$\varphi(t) = \frac{\pi}{T} f_0 t + \frac{T_M A_M}{2} \sin(\varphi_M + 2\pi t / TT_M), \quad (37b)$$

$$a(t) = H_i(f(t)), \ i=1, 2, 3; \quad (37c)$$

$$H_1(f) = (f - f_0 + 1.5A_M)/A_M, \quad (37d)$$

$$H_2(f) = 0.5 + 2(f - f_0)^2 / A_M^2, \quad (37e)$$

$$H_3(f) = 2.5 - 2(f - f_0)^2 / A_M^2. \quad (37f)$$

For the tests we sample T_M uniformly at 6 positions between 5 and 15 frames, A_M logarithmically at 6 positions between 1 and 32 bins, φ_M uniformly at 6 positions between 0 and 5π/6, f_0 uniformly at 10 positions between 155 and 155.9 bins. This makes a total of 6480 test signals. Apart from clean sinusoids, we also run tests on sinusoids contaminated by concurrent sinusoids or white noise. Reference systems are not tested with contaminated signals.

5.2. Results

We present the test results as accuracy measurements against selected parameters (T_M, A_M, estimator) averaged over all relevant test results. The measurements are amplitude accuracy (E_a), frequency accuracy (E_f) and signal-to-error ratio (*SER*). E_a is computed from the ground truth log amplitude derivatives p_0, …, p_L and their estimates $\tilde{p}_0$, …, $\tilde{p}_L$ as

$$E_a = -10\log_{10} \frac{1}{L+1} \sum_{l=0}^{L} (p_l - \tilde{p}_l)^2. \quad (38a)$$

E_f is computed from the ground truth angular frequencies ω_0, …, ω_L and their estimates $\tilde{\omega}_0$, …, $\tilde{\omega}_L$ as

$$E_f = -10\log_{10} \frac{1}{4\pi^2 (L+1)} \sum_{l=0}^{L} (\omega_l - \tilde{\omega}_l)^2. \quad (38b)$$

SER is computed from the test signal $s(t)$ and resynthesized sinusoid $\tilde{s}(t)$ as

$$SER(dB) = -10\log_{10} \frac{\sum (s(t) - \tilde{s}(t))^2}{\sum \tilde{s}(t)^2}. \quad (38c)$$

Higher values of E_a, E_f and *SER* indicate better amplitude, frequency and reconstruction accuracy, respectively.

5.2.1. Clean sinusoids

Figure 2 compares results of tested estimators on clean signals. Each curve presents the result from one estimator, as *SER*, E_f or E_a against FM period T_M in (a)(c)(e), and against FM extent A_M in (b)(d)(f), averaged over all φ_M, f_0 and transfer functions $H_1(f)$~$H_3(f)$ used in (37c). *SER* results of reference systems are included in (a) and (b).

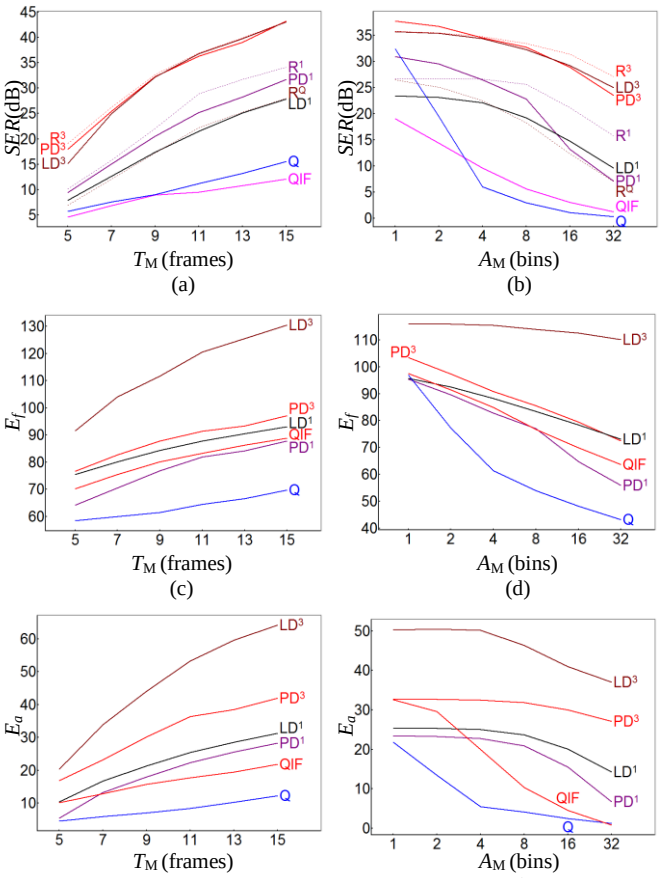


Figure 2 . Comparing estimators on FM signals (cycle 1)

As may be expected, derivative estimators using cubic polynomials outperform those using linear polynomials. Among estimators using the same polynomial orders, the local derivative algorithm does best in estimating log amplitude and frequency. This can be attributed to that fact that it uses more parameters to model the sinusoid (LD^1 uses 5 per frame, LD^3 uses 9, QIFFT use 3, all others use less than 2.1). However, this advantage is not observed in the *SER* results, which treat the whole duration equally instead of focusing on the measurement points. Since the test signals are off-model, better parameter estimation alone does not guarantee better modelling accuracy. This is even more distinctively observed in Fig. 2(b), which shows that the reference systems, in spite of holding the ground truth at measurement points, do not always have the highest *SER*. On the whole the two derivative algorithms provide similar *SER*s, and both PD^3 and LD^3 come close to R^3. Good results are observed from QUASAR estimator (Q) only if both A_M and $1/T_M$ are low. This can be traced back to the heterodyne filtering technique it uses to obtain its preliminary estimates, which cannot handle large frequency modulations.

5.2.2. Test in the presence of other sinusoids

In the presence of concurrent sinusoids, the main influence on the estimation comes from those with closest frequencies. In our experiments the test signal $s(t)$ is mixed with two concurrent sinusoids, with the same amplitude and initial phase as $s(t)$ and frequencies at $\pm B$ bins from $s(t)$. The frequency gap B is sampled logarithmically at 6 positions between 4 and 128 bins.

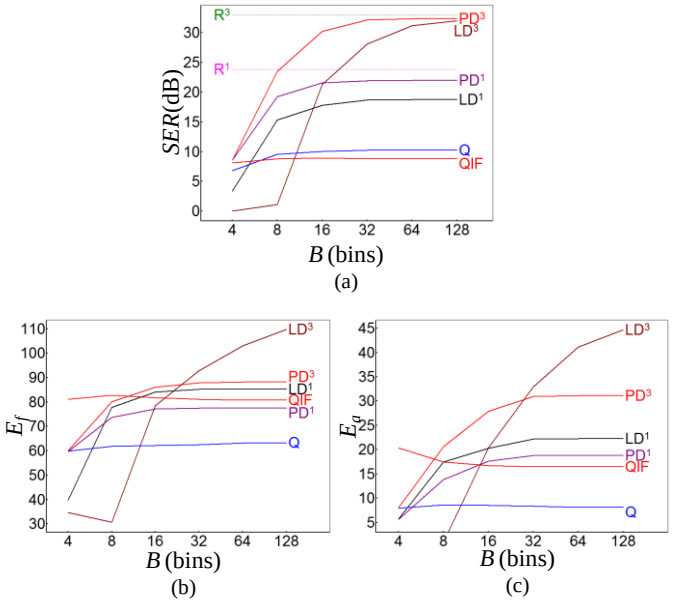


Figure 3 .Comparing estimators on FM signals with additional sinusoids

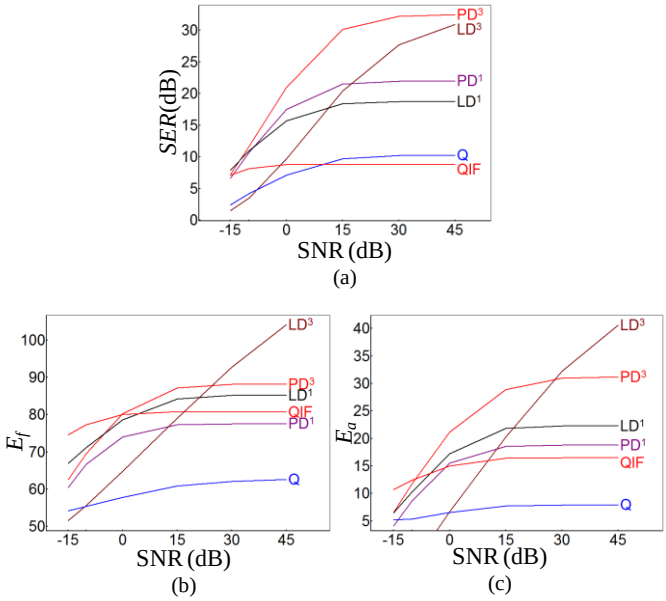


Figure 4 . Comparing estimators on FM signals with white noise

Figure 3 compares the results of all tested estimators, each curve presenting the result from one estimator, as *SER*, E_f or E_a against frequency gap *B*, averaged over all T_M, A_M, φ_M, f_0 and transfer functions $H_1(f)$~$H_3(f)$. While E_f and E_a results are averaged as they are, negative *SER*s are treated as 0 when computing the average. Results of direct synthesis R^3 and R^1 are included for reference. We see that concurrent sinusoids have significant impact on all tested estimators except QIF. For small values of *B* (i.e. strong interference) the piecewise derivative algorithm shows clear advantage in the presence of concurrent sinusoids: estimator PD^1 using a linear spline is already comparable to LD^3 using trinomials when $B \leq 16$. This can be attributed to the fact that the piecewise derivative algorithm uses less free parameters, and therefore is less likely to overfit. This explanation is also confirmed by LD^1 outperforming LD^3 under strong interference. The apparent resistance of QIF to disturbance is related to the construction of the test signals, as the choice of two symmetrical sinusoids on either side of $\omega(t)$ by whole DFT bins helps to minimize their total impact on the quadratic interpolation.

5.2.3. Test in the presence of noise

In this part we mix the test signals with Gaussian white noise at 6 levels, specified by 6 signal-to-noise ratios (SNRs) from -15dB to 45dB. Typical Cramer-Rao bounds for parameter estimation from noisy data are not included because our test signals are not synthesized to fit the parametric models.

Figure 4 compares the results of all tested estimators , each curve presenting the result from one estimator, as *SER*, E_f or E_a against input SNR, averaged over all T_M, A_M, φ_M, f_0 and transfer functions $H_1(f)$~$H_3(f)$. Negative *SER*s are treated as 0 when computing the average. Again, for small values of SNR (i.e. strong noise) the piecewise derivative algorithms shows consistent advantage over their local counterparts, and LD^1 once more outperforms LD^3. In the lower end of SNR the QIFFT method returns the best frequency estimate. This is the result of the estimate being explicitly bounded near the reference frequency no more than 0.5 bin from the ground truth, without much chance of large deviation even in extraordinary noise.

5.3. Summary

We have tested sinusoidal parameter estimation accuracy based on the piecewise derivative and local derivative algorithms, as well as the QUASAR estimator, using signals synthesized independent of the parametric models. For clean sinusoids the local derivative algorithm has the best parameter estimation accuracy, but the piecewise algorithm has the best reconstruction *SER*. The QUASAR estimator has similar performance to PD^1 only for very low modulation, and degrades quickly for high modulation. For sinusoids contaminated with noise or concurrent sinusoids, the piecewise derivative algorithm shows consistent improvement in estimation accuracy, which we attribute to long-term modelling and the reduction of the number of parameters. Notably, these improvements are achieved with half the computation cost of the local derivative method, making the piecewise version even more favourable.

Although we have treated the non-iterative and iterative estimators as mutually exclusive in the experiments, in reality they are not. A good non-iterative method, such as the piecewise derivative algorithm, can help provide an iterative algorithm an initial estimate close to global optimum, therefore relieves the latter of such potential disadvantages as heavy computation and convergence at local optimum.

6. REAL-WORLD EXAMPLE

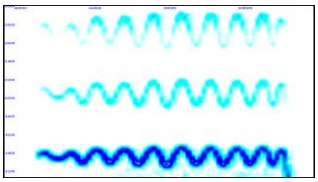
In this section we provide a real-world audio example using the two derivative algorithms to estimate frequencies and frequency slopes of sinusoidal partials.

We run the local and piecewise derivative algorithms on a pure vocal recording of a soprano singing /a:/. The spectrogram of the three lowest partials is given in Figure 5(a). We use linear frequency for the local derivative algorithm and linear spline frequency for the piecewise derivative algorithm.

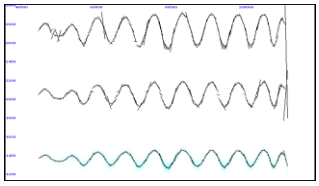
For the three lowest partials we draw the frequency results in Figures 5(b) and 5(c). The local derivative algorithm estimates the frequency over individual frames as short line segments, each covering the duration of 2*T*. The piecewise derivative algorithm estimates it over the whole duration as a linear spline. While no

frequency ground truth is available for real-world recordings, the disagreement of local estimates from adjacent frames provides an indication of the accuracy of the local derivative algorithm, while the smoothness of the frequency trajectories provides an indication of that of the piecewise algorithm. For the clean vocal signal both algorithms provide good frequency estimates.

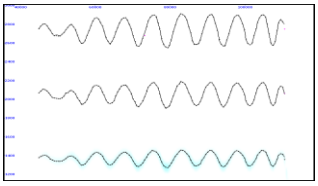
Figures 5(d) and 5(e) shows the results when 0dB white noise is added to the signal. In the presence of this noise only the strongest fundamental partial retains good estimates. For the weaker partials both algorithms suffer performance loss. Although the drawings of Figs. 5(d) and 5(e) are not directly comparable, 5(e) does show more stable frequency slope than 5(d), in which the frequency slope errors are large enough to turn local frequency trajectories into near-vertical spikes. Since the piecewise algorithm relies on a global signal model, it is reasonable to expect higher noise tolerance, because the global constraints help to reduce the number of free parameters and prevent overfitting to local noise sources.



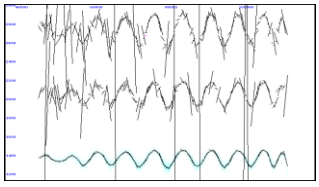
(a) spectrogram



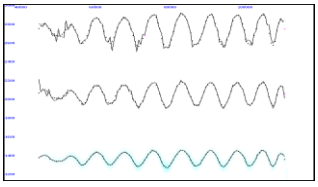
(b) local estimation, clean



(c) piecewise estimation, clean



(d) local estimation, noisy



(e) piecewise estimation, noisy

Figure 5 . Local and piecewise frequency estimation of vocal /a:/

7. CONCLUSION

In this paper we have discussed the theory and implementation of a piecewise derivative algorithm, which approximates a slowly varying complex sinusoid as a spline exponential function. This algorithm is based on the recently developed distributive derivative algorithm, currently the state-of-the-art for non-stationary sinusoid estimation on short intervals. By adapting the derivative approach to piecewise models, we are able to estimate time-varying sinusoids of flexible length. The algorithm is simple, non-iterative, and more robust against noise than the (local) distributive derivative algorithm.

Apart from as the analyser of a sinusoidal modelling system, the proposed estimator may find applications wherever observation of sinusoidal parameters at a sequence of frames is required. For example, musicologists are interested in accurate tracing of vocal and instrumental vibratos and tremolos. In the frequency sweep test popular with audio, the proposed algorithm will allow faster sweep rate (since it does not assume stationarity) and will automatically return the result as a spline function.

The piecewise derivative algorithm has no source separation capacity, therefore does not solve the more difficult problem of estimating sinusoids overlapping other sinusoids or noise in both time and frequency. However, our tests show that the piecewise algorithm is less susceptible to concurrent sinusoids and noise than the local derivative algorithm.

Although we have focused our discussions on the spline exponential model, the piecewise derivative algorithm is also applicable to other linear piecewise parameterizations of the complex exponent, as long as they are compliant to (16).

8. ACKNOWLEDGEMENT

Part of this work was finished when the author was with Queen Mary, University of London and Professor Mark Sandler, with support from the Centre for Digital Music platform grant.

9. REFERENCES

[1] R.J. McAulay and T.F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol.34, no.4, 1986, pp.744-754.

[2] X. Serra, "Musical sound modeling with sinusoids plus noise," *Musical Signal Processing*, Swets & Zeitlinger Publishers, 1997, pp.91-122.

[3] Wen X. and M. Sandler, "On the characterization of slowly varying sinusoids," *EURASIP Journal on Audio, Speech and Music Processing*, vol. 2010, article 941732, 7 pages.

[4] F. Keiler and S. Marchand, "Survey on extraction of sinusoids in stationary sounds," in *Proc.5th International Conference on Digital Audio Effects (DAFx)*, Hamburg, 2002.

[5] P. Masri and N. Canagarajah, "Extracting more detail from the spectrum with phase distortion analysis," in *Proc. 1st COST-G6 Workshop on Digital Audio Effects (DAFx)*, Barcelona, 1998.

[6] A. Röbel, "Frequency slope estimation and its application for non-stationary sinusoidal parameter estimation," in *Proc. 10th International Conference on Digital Audio Effects (DAFx)*, Bordeaux, 2007.

[7] M. Abe and J. O. Smith III, "AM/FM rate estimation for time-varying sinusoidal modeling," in *Proc. International Conference on Acoustics, Speech, and Signal Processing*, Philadelphia, 2005.

[8] Wen X. and M. Sandler, "Notes on model-based non-stationary sinusoid estimation using derivatives," in *Proc. 12th International Conference on Digital Audio Effects (DAFx)*, Como, 2009.

[9] M. Betser, "Sinusoidal polynomial parameter estimation using the distribution derivative," *IEEE Transactions on Signal Processing*, vol.57 no.12, 2009.

[10] S. Muševič and J. Bonada, "Distribution derivative method for generalized sinusoid with complex amplitude modulation," in *Proc. 18th International Conference on Digital Audio Effects (DAFx)*, Trondheim, 2015.

[11] Y. Ding and X. Qian, "Processing of musical tones using a combined quadratic polynomial-phase sinusoid and residual (QUASAR) signal model," *Journal of the Audio Engineering Society*, vol.45, no.7/8, 1997.

[12] A. Röbel, "Adaptive additive modeling with continuous parameter trajectories," *IEEE Transactions on Audio, Speech, and Language Processing*, vol.14 no.4, 2006.

[13] M. Desainte-Catherine and S. Marchand, "High-precision Fourier analysis of sounds using signal derivatives," *Journal of the Audio Engineering Society*, vol.48 no.7/8, 2000.

[14] S. Marchand and P. Depalle, "Generalization of the derivative analysis method to non-stationary sinusoidal modeling," in *Proc.*

11th International Conference on Digital Audio Effects (DAFx), Espoo, 2008.

[15] M. Abe and J. O. Smith, "Design criteria for simple sinusoidal parameter estimation based on quadratic interpolation of FFT magnitude peaks," in *Proc. AES 117th Convention*, San Francisco, 2004.

10. APPENDIX: PIECEWISE MATRIX FORMULATIONS OF LINEAR, QUADRATIC AND CUBIC SPLINES

We denote a spline by $x(t)$ and its samples at the knot points by $\mathbf{x}=(x_0, \ldots, x_L)^{\mathsf{T}}$, $x_l=x(lT)$, $\forall l$. The linear interpolative formulation expresses $x(t)$ as linear functions of $x_0, \ldots, x_L$. Accordingly, the polynomial coefficients over $[lT,lT+T)$, i.e. $\boldsymbol{\lambda}_l$, can be given as a linear transformation of $\mathbf{x}$:

$$\boldsymbol{\lambda}_l=\mathbf{A}_l\mathbf{x},\ \forall l. \tag{A1}$$

10.1. Linear spline

A *linear spline* is specified by

$$\begin{cases} x(lT+t)=\lambda_{l,1}t+\lambda_{l,0},\ 0\le t<T,\ l=0,\cdots,L-1, \\ x(lT)=x_l,\ l=0,\cdots,L. \end{cases} \tag{A2}$$

It has the closed-form expression

$$x(lT+t)=x_l(1-t/T)+x_{l+1}\cdot t/T\ . \tag{A3}$$

For this spline type we can immediately write

$$\mathbf{A}_l=\begin{pmatrix}\mathbf{0}_{2\times l} & \begin{pmatrix}-1/T & 1/T\\ 1 & \end{pmatrix} & \mathbf{0}_{2\times(L-l-1)}\end{pmatrix}. \tag{A4}$$

10.2. Quadratic spline

A quadratic spline is specified by

$$\begin{cases} x(lT+t)=\lambda_{l,2}t^2+\lambda_{l,1}t+\lambda_{l,0},\ 0\le t<T,\ l=0,\cdots,L-1, \\ x(lT)=x_l,\ l=0,\cdots,L, \\ x'_-(lT)=x'_+(lT),\ l=1,\cdots,L-1. \end{cases} \tag{A5}$$

The standard approach for quadratic spline interpolation computes the polynomial coefficients from an intermediate vector $\mathbf{z}=(z_0, \ldots, z_{L-1})^{\mathsf{T}}$, $z_l=y'_l$, by

$$\begin{pmatrix}\lambda_{l,2}\\ \lambda_{l,1}\\ \lambda_{l,0}\end{pmatrix}=\begin{pmatrix}-1/T\\ 1\\ 0\end{pmatrix}z_l+\begin{pmatrix}-1/T^2 & 1/T^2\\ 0 & 0\\ 1 & 0\end{pmatrix}\begin{pmatrix}x_l\\ x_{l+1}\end{pmatrix} \tag{A6}$$

or

$$\boldsymbol{\lambda}_l=(\mathbf{0}_{3\times l}\ \ \mathbf{A}_Z\ \ \mathbf{0}_{3\times(L-l-1)})\mathbf{z}+(\mathbf{0}_{3\times l}\ \ \mathbf{A}_C\ \ \mathbf{0}_{3\times(L-l-1)})\mathbf{x}\ , \tag{A7}$$

where $\mathbf{A}_Z$ and $\mathbf{A}_C$ replace the two matrices in (A6) for briefness. $\mathbf{z}$ is related to $\mathbf{x}$ by the linear system

$$\begin{pmatrix}1 & 1 & & & \\ & 1 & 1 & & \\ & & \cdots & \cdots & \cdots \\ & & & 1 & 1\end{pmatrix}\mathbf{z}=\frac{2}{T}\begin{pmatrix}-1 & 1 & & & 0\\ & -1 & 1 & & 0\\ & & \cdots & \cdots & \cdots\\ & & & -1 & 1\end{pmatrix}\mathbf{x}. \tag{A8}$$

Eq.(A8) is underdetermined by one equation. We introduce an additional constraint by minimizing the square norm of the spline's derivative, i.e.

$$I=\int_0^{LT}\left(x'(t)\right)^2dt=\frac{T}{3}\sum_{l=0}^{L-1}\left(z_l^2-2z_l\frac{x_{l+1}-x_l}{T}+4\frac{(x_{l+1}-x_l)^2}{T^2}\right). \tag{A9}$$

We call the specific quadratic spline that minimizes (A9) the minimal-variation quadratic spline. Let $dI/d\mathbf{z}=0$, we get a new linear equation

$$T\sum_{l=0}^{L-1}(-1)^l z_l+x_0+2\sum_{l=1}^{L-1}(-1)^l x_l+(-1)^L x_L=0\ . \tag{A10}$$

Eq.(A10) pads up the matrix on the left of (A8) to $L\times L$ and the on the right to $L\times(L+1)$, after which (A8) takes the form $\mathbf{M}_L\mathbf{z}=\mathbf{M}_R\mathbf{x}$, $\mathbf{M}_L$ being square and invertible. Substituting $\mathbf{z}=\mathbf{M}_L^{-1}\mathbf{M}_R\mathbf{x}$ into (A7) we get

$$\boldsymbol{\lambda}_l=((\mathbf{0}_{3\times l}\ \ \mathbf{A}_Z\ \ \mathbf{0}_{3\times(L-l-1)})\mathbf{M}_L^{-1}\mathbf{M}_R+(\mathbf{0}_{3\times l}\ \ \mathbf{A}_C\ \ \mathbf{0}_{3\times(L-l-1)}))\mathbf{x}\ . \tag{A11}$$

This gives the formulation of $\mathbf{A}_l$ in (A1) as

$$\mathbf{A}_l=(\mathbf{0}_{3\times l}\ \ \mathbf{A}_Z\ \ \mathbf{0}_{3\times(L-l-1)})\mathbf{M}_L^{-1}\mathbf{M}_R+(\mathbf{0}_{3\times l}\ \ \mathbf{A}_C\ \ \mathbf{0}_{3\times(L-l-1)}). \tag{A12}$$

10.3. Cubic spline

A cubic spline is specified by

$$\begin{cases} x(lT+t)=\lambda_{l,3}t^3+\lambda_{l,2}t^2+\lambda_{l,1}t+\lambda_{l,0},\ 0\le t<T,\ l=0,\cdots,L-1, \\ x(lT)=x_l,\ l=0,\cdots,L, \\ x'_-(lT)=x'_+(lT),\ x''_-(lT)=x''_+(lT),\ l=1,\cdots,L-1. \end{cases} \tag{A13}$$

The standard approach for cubic spline interpolation computes the polynomial coefficients from an intermediate vector $\mathbf{z}=(z_0,\ldots, z_L)^{\mathsf{T}}$, $z_l=y''_l$, by

$$\begin{pmatrix}\lambda_{l,3}\\ \lambda_{l,2}\\ \lambda_{l,1}\\ \lambda_{l,0}\end{pmatrix}=\begin{pmatrix}-1/6T & 1/6T\\ 1/2 & 0\\ -T/3 & -T/6\\ 0 & 0\end{pmatrix}\begin{pmatrix}z_l\\ z_{l+1}\end{pmatrix}+\begin{pmatrix}0 & 0\\ 0 & 0\\ -1/T & 1/T\\ 1 & 0\end{pmatrix}\begin{pmatrix}x_l\\ x_{l+1}\end{pmatrix} \tag{A14}$$

or

$$\boldsymbol{\lambda}_l=\begin{pmatrix}\mathbf{0}_{4\times l} & \mathbf{A}_Z & \mathbf{0}_{4\times(L-l-1)}\end{pmatrix}\mathbf{z}+\begin{pmatrix}\mathbf{0}_{4\times l} & \mathbf{A}_C & \mathbf{0}_{4\times(L-l-1)}\end{pmatrix}\mathbf{x} \tag{A15}$$

where $\mathbf{A}_Z$ and $\mathbf{A}_C$ replace the two matrices in (A14) for briefness. $\mathbf{z}$ is related to $\mathbf{x}$ by the linear system

$$\begin{pmatrix}1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & \cdots & \cdots & \cdots \\ & & & 1 & 4 & 1\end{pmatrix}\mathbf{z}=\frac{6}{T^2}\begin{pmatrix}1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & \cdots & \cdots & \cdots \\ & & & 1 & -2 & 1\end{pmatrix}\mathbf{x}. \tag{A16}$$

This is an underdetermined system. Two more constraints are needed to uniquely specify $\mathbf{z}$ from $\mathbf{x}$. The constraints typically concern the behaviour of $\mathbf{z}$ near 0 and L, referred to as the boundary mode, e.g.

a) natural mode: $z_0=z_L=0$;

b) quadratic run-out mode: $z_0=z_1$, $z_L=z_{L-1}$;

c) cubic run-out mode: $z_0=2z_1-z_2$, $z_L=2z_{L-1}-z_{L-2}$.

Whichever mode we choose, the constraints pad up the matrices on both sides of (A16) to $(L+1)\times(L+1)$, after which (A16) takes the form $\mathbf{M}_L\mathbf{z}=\mathbf{M}_R\mathbf{x}$, $\mathbf{M}_L$ being invertible. Substituting $\mathbf{z}=\mathbf{M}_L^{-1}\mathbf{M}_R\mathbf{x}$ into (A15) we get

$$\boldsymbol{\lambda}_l=((\mathbf{0}_{4\times l}\ \ \mathbf{A}_Z\ \ \mathbf{0}_{4\times(L-l-1)})\mathbf{M}_L^{-1}\mathbf{M}_R+(\mathbf{0}_{4\times l}\ \ \mathbf{A}_C\ \ \mathbf{0}_{4\times(L-l-1)}))\mathbf{x}. \tag{A17}$$

This gives the formulation of $\mathbf{A}_l$ in (A1) as

$$\mathbf{A}_l=(\mathbf{0}_{4\times l}\ \ \mathbf{A}_Z\ \ \mathbf{0}_{4\times(L-l-1)})\mathbf{M}_L^{-1}\mathbf{M}_R+(\mathbf{0}_{4\times l}\ \ \mathbf{A}_C\ \ \mathbf{0}_{4\times(L-l-1)}). \tag{A18}$$

THE FENDER BASSMAN 5F6-A FAMILY OF PREAMPLIFIER CIRCUITS—A WAVE DIGITAL FILTER CASE STUDY

W. Ross Dunkel[*], *Maximilian Rest*[†*], *Kurt James Werner*[*], *Michael Jørgen Olsen*[*], *Julius O. Smith III*[*]

[*]Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, California, USA
[†]Fakultät Elektrotechnik und Informatik, Technische Universität Berlin, Berlin, Germany
[chigi22|mrest|kwerner|mjolsen|jos]@ccrma.stanford.edu

ABSTRACT

The Fender Bassman model 5F6-A was released in 1958 and has become one of the most revered guitar amplifiers of all time. It is the progenitor of a long line of related Fender designs in addition to inspiring Marshall's first amplifier design. This paper presents a Wave Digital Filter study of the preamplifier circuit of 5F6-A-based amplifiers, utilizing recent theoretical advances to enable the simultaneous simulation of its four nonlinear vacuum tube triodes. The Dempwolf triode model is applied along with an iterative Newton solver to calculate the scattering at the 25 port R-type adapter at the root of the WDF tree. Simulation results are compared to "ground truth" SPICE data showing excellent agreement.

1. INTRODUCTION

The Fender Bassman amplifier was first introduced in 1952, undergoing multiple revisions before culminating in the seminal 5F6-A version in 1958. This model is one of the most revered and imitated amplifier circuits of all time [1], inspiring countless related designs including the first Marshall—the JTM 45. Introduced in 1962 by London music store owner Jim Marshall, the JTM 45's original design borrowed heavily from the 5F6-A, a popular seller and employee favorite [2]. The JTM 45 quickly became a successful and much-praised amplifier in its own right and was a crucial ingredient in the heady brew of hard rock, blues, and psychedelic music that was the sound of 1960s London. Our goal is to develop efficient, re-configurable digital emulations of the 5F6-A, JTM 45 and related circuits to serve as a research and design tool for DIY amp-designers, circuit benders, and sonic historians considering the evolution of this historic circuit.

A number of different approaches have been taken to the numerical simulation of vacuum tube guitar amplifier circuits over the years. For an historical overview as well as an introduction to the physical principles of operation of triode tubes see [3]. Recent years have seen the increased use of Wave Digital Filters (WDFs) [4] as an effective tool for virtual analog modeling of audio circuits [5]. New theoretical advances have enabled WDFs to model circuits with complex topologies [6] and multiple nonlinearities [7], a category that includes many audio circuits of interest. In this paper we apply these techniques to the construction of a WDF model of Bassman 5F6-A-derived preamplifiers. These circuits provide an ideal case study—they contain multiple vacuum tube triodes in a complex circuit topology but with a modest part count, limiting overall system complexity to a reasonable level.

The remainder of this work is structured as follows: Sec. 2 explores the circuit in detail, Sec. 3 reviews necessary background information and derives the structure of the WDF simulation including the novel application of the Dempwolf triode model and a Newton-based root finder, Sec. 4 presents simulation results, and Sec. 5 summarizes future work and conclusions.

2. BASSMAN 5F6-A PREAMPLIFIER CIRCUITS

This work defines the 5F6-A's "preamplifier circuit" as consisting of the following stages: the 12AY7 preamp, 12AX7 voltage amp, and 12AX7 cathode follower (as in [1]) driving a nominal load resistance R_L. The circuit schematic is shown in Figure 1 with component values listed in Table 1. The tone stack is not included in the simulation as it is buffered from the preceding sections by the cathode follower stage. The tone stack has already been considered in the context of WDFs [8, 9] and in an earlier non-WDF study [10]. Nominal values cited in the remainder of Sec. 2 are taken from [1] and are presented solely to provide insight into the circuit's operation.

Edge	Comp	Value	Edge	Comp	Value
1	R_{GB1A}	68 kΩ	14	R_{VBP}	αR_{vol}
2	R_{GB1B}	68 kΩ	15	R_{VBN}	$(1-\alpha)R_{\text{vol}}$
3	R_{IB}	1 MΩ	16	R_{VNN}	$(1-\alpha)R_{\text{vol}}$
4	R_{K1}	1 MΩ	17	R_{VNP}	αR_{vol}
5	C_{K1}	250 µF	18	C_{B1}	100 pF
6	R_{GN1A}	68 kΩ	19	R_{GB2}	270 kΩ
7	R_{GN1B}	68 kΩ	20	R_{GN2}	270 kΩ
8	V_{in}	–	21	R_{P2}	100 kΩ
9	R_{IN}	1 MΩ	22	R_{K2}	820 Ω
10	R_{PB}	100 kΩ	23	R_L	100 kΩ
11	R_{PN}	100 kΩ	24	V_P	310 V
12	C_{ON}	20 nF	25	C_{B2}	556 pF
13	C_{OB}	20 nF	–	R_{vol}	1 MΩ

Table 1: *Circuit Components*

2.1. The Preamp Section

The preamp's first stage is a dual channel inverting amplifier powered by a 12AY7 tube consisting of triodes T_{1B} and T_{1N} for bright and normal channels respectively. The grid stopper resistors R_{GB1A}, R_{GB1B}, R_{GN1A}, R_{GN1B} combine with the triodes' Miller capacitances [11] to form a low-pass filter that suppresses RF interference. Each channel has two inputs: high sensitivity #1 with input resistances R_{IB} and R_{IN} and voltage gain of -32.2 and low sensitivity #2 with input resistances of $R_{GB1A} + R_{GB1B}$ and $R_{GN1A} + R_{GN1B}$ and voltage gain of -16.1 (−6 dB attenuation relative to #1). With no cables inserted in the input jacks, their tip and switch terminals are shorted as shown on the left-hand side of Figure 1. When a cable is inserted, this connection is broken and

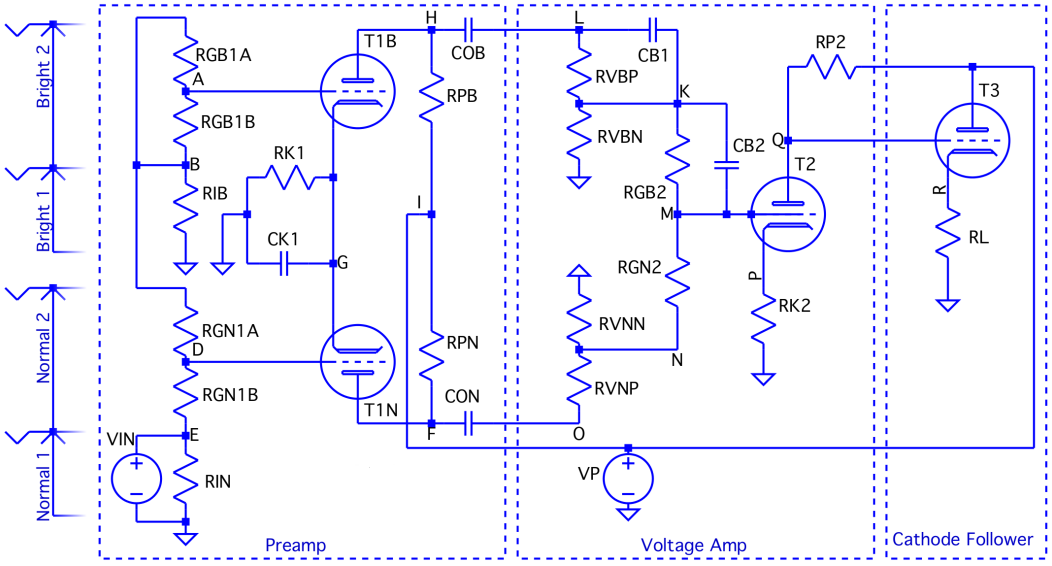


Figure 1: *5F6-A/JTM 45 preamplifier circuit with no inputs (left) and using Normal 1 input with a jumper between Normal 2 and Bright 1 (right). Circuit element values are listed in Table 1.*

the tip terminal is routed into the larger circuit as shown on the right. The cathode resistor R_K sets the bias point of the triodes and is fully bypassed by the capacitor C_K at audio frequencies. The output voltage of this section is developed across plate resistors R_{PB1} and R_{PB2} and is transmitted to the next stage through the output coupling capacitors C_{OB} and C_{ON}.

2.2. The Voltage Amp Section

The second stage consists of a 12AX7 triode T_2 in an inverting amplifier configuration with a net voltage gain of -20.7 at maximum volume. Bright and normal volume levels are set via 1 MΩ potentiometers parameterized by α_B and α_N respectively with $0 \leq \alpha \leq 1$, sub-dividing the pots into resistors $R_{VBN} = (1-\alpha) \cdot R_{\text{vol}}$, $R_{VBP} = \alpha \cdot R_{\text{vol}}$, $R_{VNN} = (1-\alpha) \cdot R_{\text{vol}}$, and $R_{VNP} = \alpha \cdot R_{\text{vol}}$. In practice these resistors are restricted to $R \geq 1\,\Omega$ and we consider only the case $\alpha_B = \alpha_N = \alpha$. Capacitors C_{B1} and C_{B2} provide the treble boost that gives the bright channel its name and R_{GB1} and R_{GB2} are again grid stopper resistors. Cathode resistor R_{K2} sets T_2's bias, with average plate current on the order of 1 mA. As will be seen in the results section, this triode's grid current can grow to an appreciable fraction of the overall cathode current at higher volume settings.

2.3. The Cathode Follower Section

The third section employs a 12AX7 triode T_3 in a non-inverting cathode follower configuration with slightly below unity gain (≈ 0.984). This stage presents a low output impedance of 615 Ω, effectively decoupling it from the tone stack and making it an ideal final stage in our WDF simulation. Even more dramatically than in the case of T_2, T_3's grid current can become appreciable at higher volumes. This has a significant impact on the preamplifier's output waveform, leading to asymmetric clipping and its concomitant even order harmonics. For this reason, models that fail to take grid current into account are unable to satisfactorily reproduce the sound of an over-driven guitar preamplifier.

2.4. Marshall JTM 45

The pre-amp of the JTM 45 is nearly identical to the 5F6-A with three important exceptions. First, a higher gain ECC83 (12AX7) replaces the Bassman's 12AY7 triodes T_{1B} and T_{1N} in the preamp section producing more overall gain and slightly different loading characteristics. Second, the lead version of the JTM 45 includes an additional bypass capacitor C_{B2} across the bright input to T_2 that is not present in the 5F6-A, leading to a more pronounced treble boost for the bright channel. Third, the plate supply voltage V_P is reduced from 325 V to 310 V [1]. For reasons discussed in Sec. 3.4, this work will specifically consider the JTM 45 preamplifier.

2.5. Jumpers

It is common for guitar players to explore different tones by employing a jumper cable between two of the unused input channels. This practice can appreciably alter the amplifier's overall response and further complicate the circuit's already complex topology. This case study specifically considers only one of the most common configurations [12] using the Normal #1 input with a jumper between Normal #2 and Bright #1 inputs as shown in Figure 1. A more general model incorporating real-time switchable jumpers will be the subject of future work.

3. WDF CIRCUIT SIMULATION

3.1. Previous Work

Wave Digital Filters (WDFs) are efficient, modular filter structures with attractive numerical properties including low sensitivity to coefficient round-off and guaranteed incremental passivity of passive

circuit elements (capacitors, inductors, resistors, etc.) when simple rounding rules are applied [4]. Using other numerical methods these "passive" elements can become locally active with rounding error pumping energy into the system if care is not taken to ensure energy conservation. WDFs were invented to facilitate the design of digital emulations of analog ladder and lattice filters, which not coincidentally also feature low sensitivity to component value variance. Recent years have seen steady growth in the application of WDFs to other areas of study removed from their original intended use, including virtual analog modeling of audio circuits [5].

Early WDF tube amplifier studies [13] often modeled triodes as one port devices, ignoring grid current completely and using grid voltage as a cross-control on the plate port current. The classic Fairchild 670 compressor was modeled [14] using a similar approach with additional unit delays to decouple the push/pull amplifier sections. An enhanced model [15] more accurately captured triode behavior, modeling the plate-to-cathode port via a nonlinear resistor parametrized by Koren's model [16] and grid current via a diode model. However, this approach still relies on unit delays to yield computable structures. A case study by Pakarinen et al. [17] applies this model to the output chain of a vacuum tube amplifier. An iterative secant-method based solver has also been employed [18] to simultaneously solve the coupled equations of the Cardarilli triode model [19].

Two recent theoretical developments have enabled the use of WDFs to model vacuum tube circuits without making any of the ad-hoc assumptions of earlier work. First, an approach informed by Modified Nodal Analysis (MNA) [20] was developed to model complex circuit topologies that cannot be decomposed into simple series and parallel combinations [6]. Second, the use of this approach was extended to circuits containing multiple non-adaptable nonlinear ports [7]. Crucially these new developments separate nonlinearities from topology, ensuring that the most challenging part of the problem (solving for the scattering at the root) scales with the number of nonlinear ports instead of the total system size.

3.2. WDF Simulation Structure

Following the methodology of [6] the first step is to translate the schematic into the biconnected graph shown in Figure 2 with nodes representing physical circuit nodes and edges representing circuit components. The three unlabeled nodes in Figure 2 have been added to group the nonlinearities into a nonseparable replacement graph. The separation methods described in [21] can then be applied to reduce the graph into the set of split components show in Figure 3. These split components are used to construct the SPQR-tree representation shown in Figure 4 with the $\mathcal{R}$-type node at the root of the tree, connected via real edges to the $\mathcal{Q}$-type nodes and virtual edges to the $\mathcal{S}$-type and $\mathcal{P}$-type nodes.

This representation directly yields the WDF structure shown in Figure 5 with the $\mathcal{R}$-type node at the root and 17 child sub-trees. The problem then becomes solving for the scattering of incoming waves $\boldsymbol{a}$ to outgoing waves $\boldsymbol{b}$ at the root $\mathcal{R}$-type node governed by the equation

$$\boldsymbol{b} = \mathbf{S}\boldsymbol{a}\,. \tag{1}$$

The topological scattering matrix $\mathbf{S}$ can be calculated [6] using

$$\mathbf{S} = \mathbf{I} + 2\begin{bmatrix}\mathbf{0} & \mathbf{R}\end{bmatrix}\mathbf{X}^{-1}\begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}^{\mathsf{T}}\,, \tag{2}$$

where $\mathbf{R}$ is the diagonal matrix of port resistances and $\mathbf{X}$ is the MNA matrix. Omitted here due to space constraints, a full derivation of $\mathbf{X}$ for the JTM 45 can be found in the supplemental materials online [22]. Note that the nonlinearities have been absorbed into the $\mathcal{R}$-type node and are considered to be connected to the topological scattering via "internal ports" while "external ports" connect the root element to its sub-trees. Internal incoming and outgoing wave variables are denoted as $\boldsymbol{a}_I$ and $\boldsymbol{b}_I$ respectively, external waves as $\boldsymbol{a}_E$ and $\boldsymbol{b}_E$. We can then decompose equation (1) in terms of the vector nonlinear function $\boldsymbol{a}_I = f(\boldsymbol{b}_I)$ that maps $\boldsymbol{a}_I$ to $\boldsymbol{b}_I$ and the partitioned $\mathbf{S}$-matrix as

$$\text{wave nonlinearity} \quad \left\{\boldsymbol{a}_I = f(\boldsymbol{b}_I) \right. \tag{3}$$

$$\text{scattering} \quad \begin{cases}\boldsymbol{b}_I = \mathbf{S}_{11}\boldsymbol{a}_I + \mathbf{S}_{12}\boldsymbol{a}_E & (4)\\ \boldsymbol{b}_E = \mathbf{S}_{21}\boldsymbol{a}_I + \mathbf{S}_{22}\boldsymbol{a}_E\,. & (5)\end{cases}$$

These equations comprise a non-computable, delay-free loop as the outgoing internal wave variables depend instantaneously on the incoming waves that in turn depend instantaneously on $\boldsymbol{b}_I$.

The vast majority of nonlinear circuit elements lack a closed-form wave domain description and are therefore most commonly and straightforwardly defined in the Kirchhoff domain. We denote the vector Kirchhoff nonlinear function that maps voltage into current as $h(\boldsymbol{v}_C)$. In the case of the 5F6-A preamplifier this function is eight dimensional, consisting of four identical two-port triode models of two equations each (discussed in 3.4). When utilizing a Kirchhoff nonlinearity a wave-to-Kirchhoff conversion matrix $\mathbf{C}$ must be included and is partitioned into internal and external components as with the scattering matrix $\mathbf{S}$. These further definitions yield the following set of equations:

$$\begin{cases} & \mathbf{H} = (\mathbf{I} - \mathbf{C}_{22}\mathbf{S}_{11})^{-1}\\ \boldsymbol{i}_C = h(\boldsymbol{v}_C) & \mathbf{E} = \mathbf{C}_{12}(\mathbf{I} + \mathbf{S}_{11}\mathbf{H}\mathbf{C}_{22})\,\mathbf{S}_{12}\\ \boldsymbol{v}_C = \mathbf{E}\boldsymbol{a}_E + \mathbf{F}\boldsymbol{i}_C \quad \text{with} & \mathbf{F} = \mathbf{C}_{12}\mathbf{S}_{11}\mathbf{H}\mathbf{C}_{21} + \mathbf{C}_{11}\\ \boldsymbol{b}_E = \mathbf{M}\boldsymbol{a}_E + \mathbf{N}\boldsymbol{i}_C & \mathbf{M} = \mathbf{S}_{21}\mathbf{H}\mathbf{C}_{22}\mathbf{S}_{12} + \mathbf{S}_{22}\\ & \mathbf{N} = \mathbf{S}_{21}\mathbf{H}\mathbf{C}_{21}\,. \end{cases} \tag{6}$$

By setting up the problem in this way, the delay-free loop has been confined to a set of equations whose size equals the number

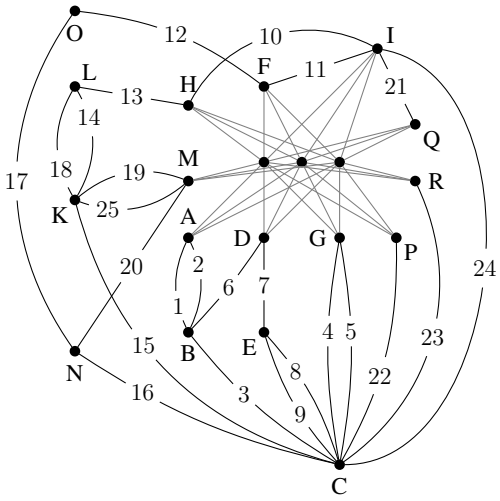


Figure 2: *Biconnected circuit graph. Node labels correspond to those in figure 1, edge numbers are defined in table 1.*

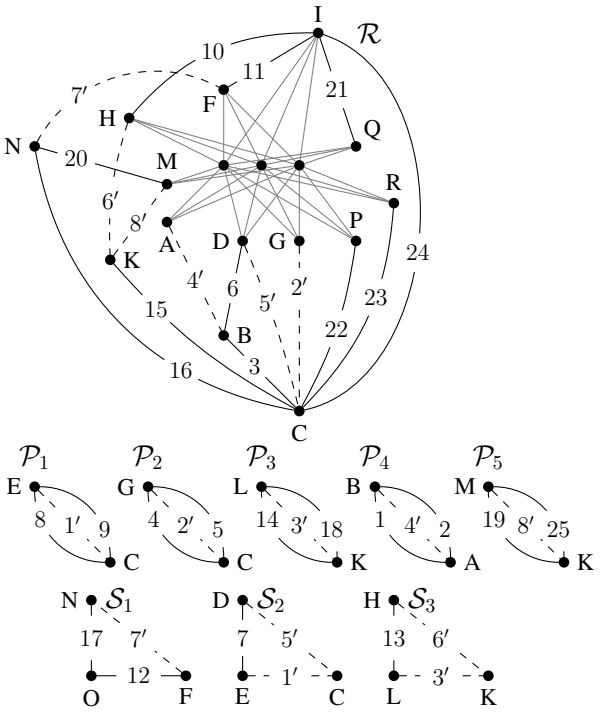


Figure 3: *$\mathcal{S}$-, $\mathcal{P}$-, and $\mathcal{R}$-type split components of the biconnected circuit graph.*

of nonlinear ports. Werner et al. chose to resolve the delay-free loop by applying the K-method to shear the nonlinearity [6]. As the sheared nonlinearity has no closed-form solution, its values are tabulated and interpolated at run-time. This approach has the drawback that as the number of nonlinear ports in the circuit grows a compromise between table resolution and an increasingly rapidly ballooning memory footprint must be made.

3.3. Newton's Method with Backtracking

For this study we take a different approach, directly applying a Newton solver to the set of nonlinear equations (6), eliminating the need to store and interpolate tables. Newton solvers have the further benefit of giving the algorithm designer access to the tradeoff space between accuracy and computation time via a single tunable

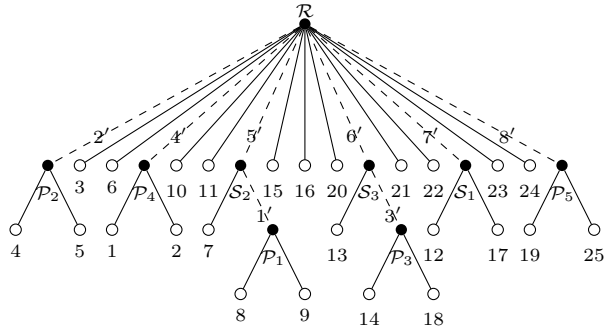


Figure 4: *SPQR tree representation of the circuit graph.*

tolerance parameter. The relevant entries of (6) can be re-arranged to make it clear that the problem can be stated in terms of finding the roots of the eight-dimensional equation

$$f(\boldsymbol{v}_C) = \mathbf{E}\boldsymbol{a}_E - \boldsymbol{v}_C + \mathbf{F}h(\boldsymbol{v}_C). \tag{7}$$

Specifically a form of Newton's method is used that includes backtracking to improve convergence robustness for a wide range of operating conditions. Initial guesses for the port voltages at the beginning of each audio sample's Newton iterations are calculated using the current value of the incoming external waves and the previous value of the internal port currents according to

$$\boldsymbol{v}_{C0}[n] = \mathbf{F}\boldsymbol{i}_C[n-1] - \mathbf{E}\boldsymbol{a}_E[n]. \tag{8}$$

A more detailed discussion of the use of Newton's method with backtracking in the context of WDFs is presented in [23]. Again, it is emphasized that the dimensionality of the iterative Newton root-finding in the current work is limited to the number of nonlinear ports.

3.4. Triode Model

The final step required to arrive at a working WDF simulation is to define an appropriate triode model. We consider the three-terminal electrical device as a two-port nonlinear WDF element with port voltages V_{PK} and V_{GK} and corresponding port currents I_{PK} and I_{GK}. We adopt the convention of amplifier texts such as [1] with the letter "P" referring to plate, "G" to grid and "K" to cathode. Further, we use a two-letter naming convention to make explicit that all voltages are referenced to the cathode voltage. See Figure 6) for a graphical representation of these port definitions.

Previous WDF triode circuit studies have employed the Cardarilli model [19] which defines grid and plate currents in a piece-wise manner above and below a critical grid voltage $V_{GK} = 0.2\,\mathrm{V}$ as shown in Figure 7. The piece-wise nature of the model can lead to poor performance with Newton-based root finders near this critical voltage. Therefore this study employs the Dempwolf model [24] which features a smooth transition across the critical voltage. Cathode, grid, and plate currents are defined as follows:

$$\begin{aligned} I_K &= G \cdot \left(\log\left(1 + \exp\left(C \cdot \left(\frac{1}{\mu} \cdot V_{PK} + V_{GK}\right)\right)\right) \cdot \frac{1}{C}\right)^{\gamma} \\ I_{GK} &= G \cdot \left(\log\left(1 + \exp\left(C_g \cdot V_{GK}\right)\right) \cdot \frac{1}{C_g}\right)^{\xi} \\ I_{PK} &= I_K - I_{GK}. \end{aligned} \tag{9}$$

The model parameters are perveances G, G_g, adaption parameters C, C_g and (positive) exponents γ, and ξ. For a discussion of the physical interpretation of these parameters and how to extract them from measured triode data see [24]. As the paper presents only 12AX7 triode parameters we choose to begin our investigation of the Bassman family of preamplifiers with the JTM 45. Evaluation of the perceptual accuracy of the Dempwolf model in the current work will be limited as it is impossible without physical measurements with which to compare. A comparison of various triode models to measurements of a physical amplifier will be the subject of future work.

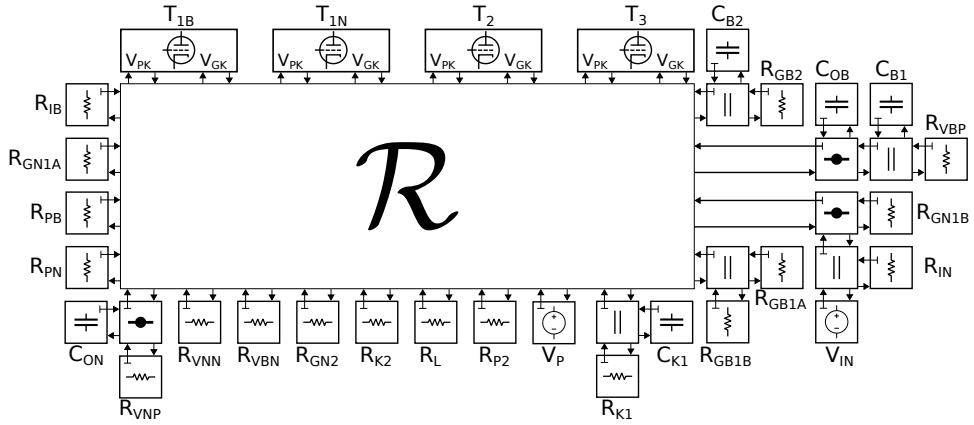


Figure 5: *Preamplifier WDF structure. The root of the tree is the large $\mathcal{R}$-type node, represented here as a topological scattering matrix connected to the four nonlinear triodes via eight "internal ports" and to seventeen linear, adapted sub-trees via its "external ports."*

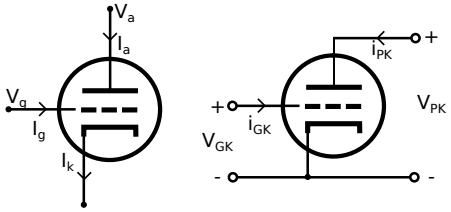


Figure 6: *Dempwolf triode current and voltage definitions (left) and port variable definitions (right).*

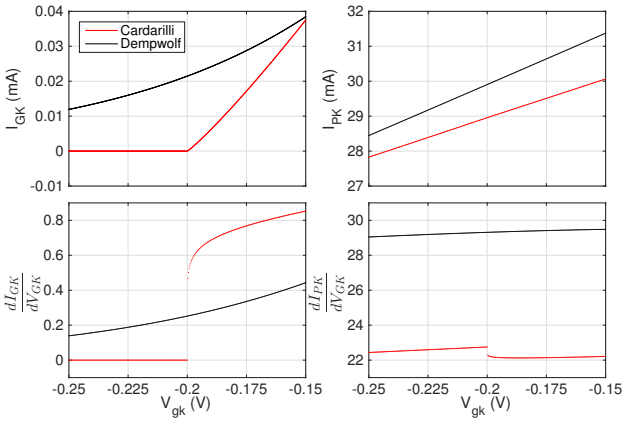


Figure 7: *Comparison of Cardarilli and Dempwolf grid (I_{GK}, left) and plate (I_{PK}, right) currents (above) and their derivatives (below) with respect to V_{GK} for fixed plate voltage V_{PK} = 150 V.*

4. RESULTS

For rendered example audio output see the supplemental materials online [22]. All simulation results presented here were performed in MATLAB at $4\times$ oversampling of a typical base audio sampling rate of 44.1 kHz (176.4 kHz) using 1 kHz, 125 mV peak-to-peak sinusoids as input signals unless otherwise noted. This voltage was chosen as a middle ground between the output level of a hot single coil and a lower-output humbucker pickup [25]. WDF results are compared to LTspice simulations carried out with an identically parameterized Dempwolf triode model. Time domain results for α values of 0.25, 0.5, 0.75, and 1.0 are presented in Figure 8, showing excellent agreement with SPICE results for all volume settings. For $\alpha = 0.25$, the output waveform is still highly sinusoidal but as volume is raised to $\alpha = 1.0$ soft clipping distortion characteristic of tube-based amplifiers becomes increasingly apparent. As α raised further the soft-clipping of the output waveform becomes increasingly asymmetric with positive excursions being noticeably flatter than negative excursions.

Figure 9 provides a more detailed view of the time domain simulation results for $\alpha = 1.0$, showing the output voltage waveform, currents I_{PK} and I_{GK} for triodes T_2 and T_3, error signal (defined as the sample-by-sample difference between the resampled SPICE and WDF data), and per-sample Newton iteration and backtrack counts. While T_{1B} and T_{1N} draw negligible grid current for all volume settings and input voltages studied T_2 and T_3's grid currents reach appreciable fractions of the corresponding total cathode currents. Furthermore, the maximum values of these two grid currents are achieved with a 180° relative phase shift. This makes sense since the inputs to these two stages are 180° out of phase, the voltage amp stage involving T_2 being an inverting stage. The regions where T_3's grid current reaches its maximum are the more heavily clipped positive output excursions. This asymmetric clipping is crucial to the generation of even order harmonics and emphasizes the importance of including the effects of grid currents in triode simulations if they are to capture this expected behavior.

The maximum error signal is approximately 30 mV for an output signal with a peak-to-peak voltage of approximately 165 V, representing an error of less than 0.2%. The SPICE results have been resampled onto the regular time grid of the WDF simulation in order to calculate the error signal. The Newton solver shows a remarkably low 3.03 average iterations and 0.08 backtracks per audio sample for $\alpha = 1.0$.

The WDF and SPICE simulation results are compared in the frequency domain for $\alpha = 1.0$ in Figure 10. Again, the SPICE results have been resampled onto the WDF simulation's time grid ($f_s = 176.4$ kHz) to enable spectral analysis and a Blackman window has been applied. Additionally, the SPICE data have been offset horizontally by 100 Hz to improve intelligibility, as other-

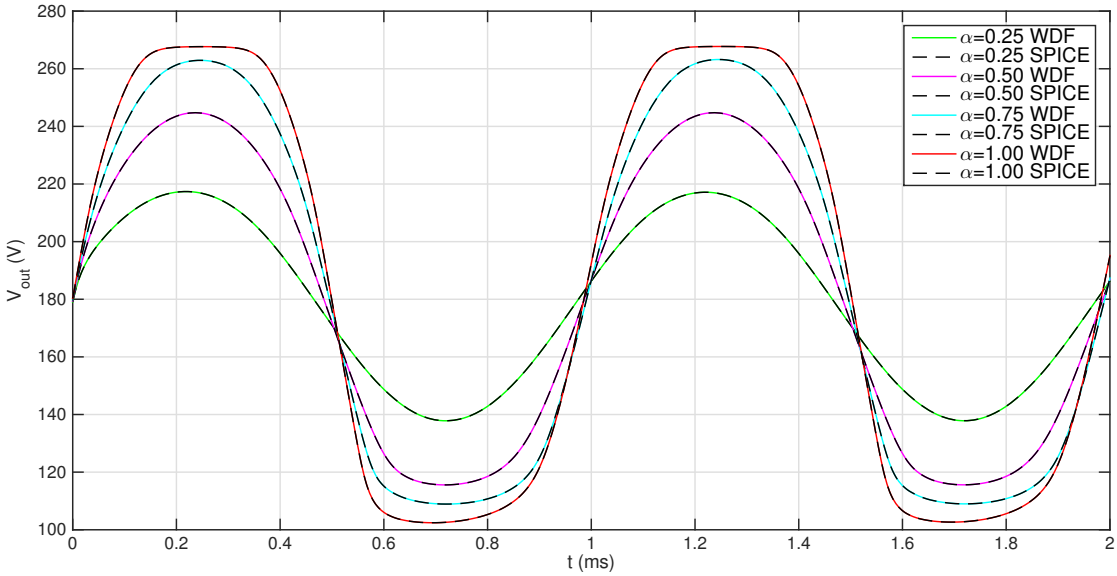


Figure 8: *WDF and SPICE simulation results for a* 1 kHz, 125 mV *peak-to-peak sinusoidal input.*

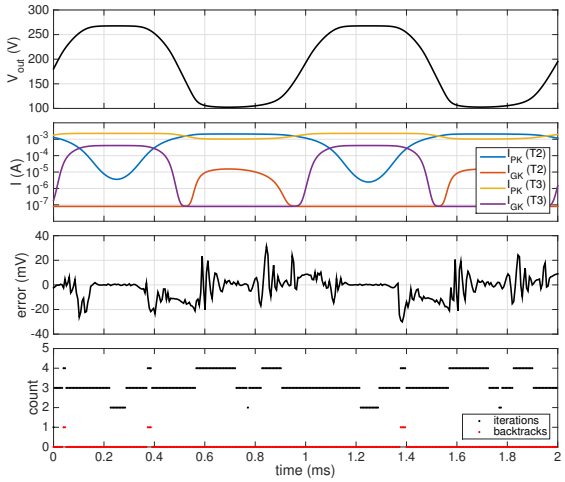


Figure 9: *WDF simulation results showing (top to bottom): output, triode T_2 and T_3 port currents, error (sample-by-sample difference between resampled SPICE and WDF results), and Newton iteration/backtrack count per audio sample (*1 kHz, 125 mV *peak-to-peak input, $\alpha = 1.0$).*

wise the peaks were more or less indistinguishable. The presence of strong even-order harmonics is characteristic of the asymmetric soft-clipping attributed to the inclusion of grid currents in the Dempwolf triode model. The relatively higher noise floor of the SPICE results is probably due to LTspice's tolerance settings (default values were used) and error introduced by resampling the variable-time-step SPICE data. No claims are made here as to the perceptual significance of this noise floor discrepancy, as psychoacoustic effects are not considered in this work.

QIFFT interpolation [26, 27] is applied to the spectral peaks revealing the first twenty harmonic peak frequencies to be identical to within less than 1 mHz. The interpolated dB peak magnitudes are listed in Table 2 and show remarkable agreement, differing by 0.1 dB or less up to the 17^{th} harmonic with a maximum deviation of 0.63 dB for the 18^{th} harmonic.

Harmonic	SPICE	WDF	Harmonic	SPICE	WDF
DC	143.31	143.31	11	73.61	73.54
1	130.72	130.72	12	79.08	79.09
2	114.76	114.76	13	70.99	70.99
3	113.57	113.57	14	73.52	73.55
4	104.53	104.52	15	67.74	67.70
5	91.24	91.23	16	64.81	64.81
6	86.96	86.94	17	64.32	64.22
7	89.42	89.42	18	53.80	53.17
8	81.37	81.37	19	57.29	57.55
9	80.28	80.28	20	55.93	56.21
10	73.41	73.46	–	–	–

Table 2: *SPICE and WDF frequency response interpolated peak values in dB (*1 kHz, 125 mV *peak-to-peak input, $\alpha = 1.0$).*

Finally, exponential sine sweeps [28] ranging from 20 Hz to 20 kHz over a 10 s time interval are used as inputs to the WDF simulation. Results for $\alpha = 0.25$ and $\alpha = 0.75$ are shown in the spectrograms of Figures 11 and 12 respectively. These results confirm that the model is well-behaved for input frequencies across the audible range and that the model's response features the expected increase in intensity of higher harmonics as volume is increased.

5. CONCLUSION AND FUTURE WORK

5.1. Future Work

As the current work models a single fixed jumper cable configuration, a future expanded preamplifier study will focus on including a fully general, real-time switchable jumper model. This will rely

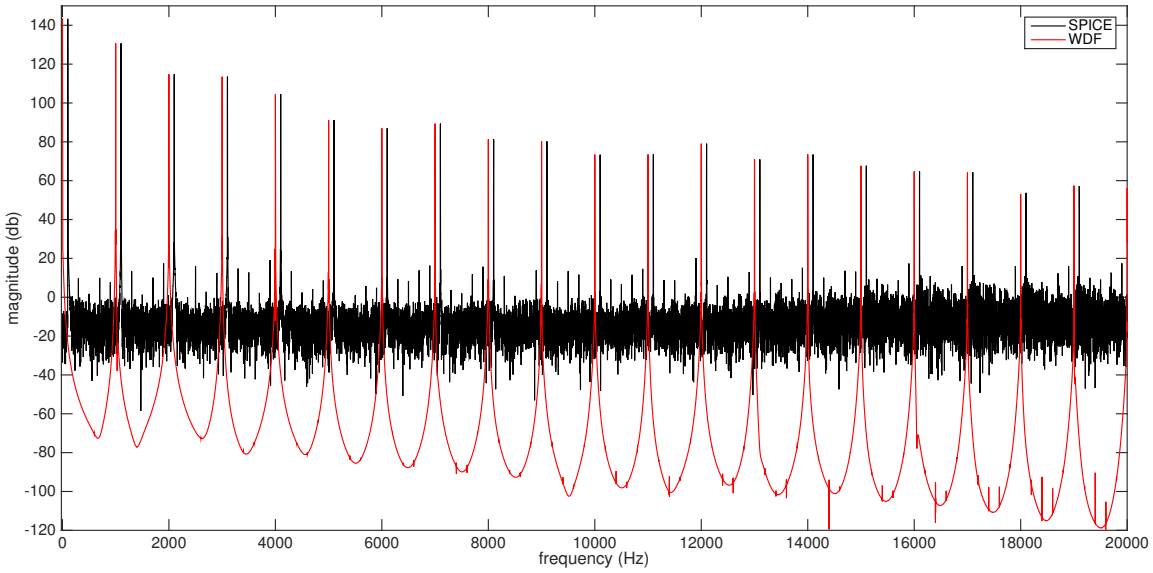


Figure 10: *WDF and SPICE frequency domain responses, SPICE offset by* 100 Hz *for clarity (*1 kHz*,* 125 mV *peak-to-peak input,* $\alpha = 1.0$*).*

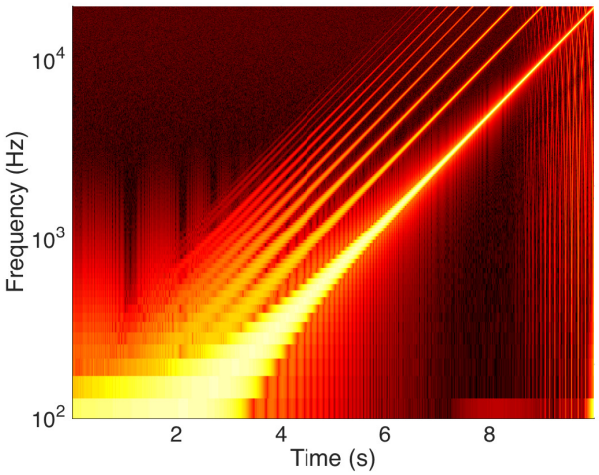


Figure 11: *Exponential sine sweep response spectrogram.* 125 mV *peak-to-peak input,* $\alpha = 0.25$*.*

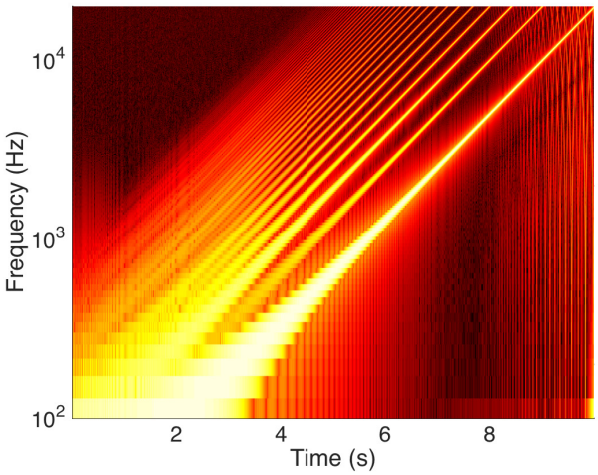


Figure 12: *Exponential sine sweep response spectrogram.* 125 mV *peak-to-peak input,* $\alpha = 0.75$*.*

on recent theoretical developments that enable WDFs to accommodate multiple non-adapted linear root elements [29]. The effects of stray pin capacitance will also be considered, most significantly the enhanced Miller equivalent capacitance [11] between plate and grid. It is our hope that Dempwolf model parameters for 12AY7 triodes can be obtained to facilitate comparison of the 5F6-A circuit's response to that of the JTM 45. Further triode models will also be evaluated and compared to the Dempwolf results presented here. To yield a complete, stand-alone preamplifier model the tone stack will also be incorporated into the WDF structure.

The current WDF simulation is in the process of being implemented and optimized in RT-WDF—a C++ real-time framework for WDF simulation described in [30]. Though the simulation is currently running slightly slower than real-time in an offline rendering utility, it is our hope and belief that further optimization will yield better-than-real-time performance.

Finally, it is worth noting that the most significant deviations of the JTM 45 design occur later in the amplifier's signal chain. For one thing, the Marshall utilizes significantly more negative feedback to drive the long-tail pair that feeds the push-pull power amp [1]. Also, while early JTM 45's utilized Radio Spares Deluxe [2] output transformers, the 5F6-A featured the Triad model 45249. Perhaps most significantly, the first Marshall cabinets made to accompany the JTM 45 were closed-back 4×12 designs loaded with Celestions G12 speakers whereas the Bassman 5F6-A was an open-backed 4×10 combo featuring Jensen P10Q's. There-

fore in addition to continued refinement of the preamplifier model it is our aim to keep moving down the signal path, producing WDF models of as many of these elements as possible.

5.2. Conclusion

The WDF simulation results for the JTM 45 preamplifier show excellent agreement with SPICE "ground truth" results for a range of input signals and volume settings. In contrast to other WDF tube amplifier studies no ad-hoc assumptions are made, obviating the necessity for domain knowledge in analyzing the circuit and setting up the simulation. The only simplification is the use of a simple load resistance R_L to represent the downstream circuit. When considered alongside a previous study of the Fender tone stack [6], the entire preamplifier circuit of Bassman-derived amplifiers has now been accurately modeled using WDFs. The use of Newton solver to calculate the nonlinear scattering at the root instead of tabulated methods leads to a much more compact memory footprint. It further allows the algorithm designer to trade simulation accuracy for computation time in a dynamic way by tuning the solver's tolerance parameter.

6. ACKNOWLEDGEMENTS

The author wishes to thank Jonathan Abel for enlightening conversations, his always unique perspective, and his generosity with baked goods.

7. REFERENCES

[1] Richard Kuehnel, *Circuit Analysis of a Legendary Tube Amplifier: The Fender Bassman 5F6-A*, Amp Books, Seattle, 2009.

[2] Michael Doyle, *The History of Marshall: The Illustrated Story of "The Sound of Rock"*, chapter The JTM 45 Series 1962–1966, pp. 17–22, Hal Leonard Corporation, Milwaukee, 1993.

[3] Jyri Pakarinen and David T. Yeh, "A review of digital techniques for modeling vacuum-tube guitar amplifiers," *Computer Music Journal*, vol. 33, no. 2, pp. 85–100, Summer 2009.

[4] Alfred Fettweis, "Wave digital filters: Theory and practice," *Proceedings of the IEEE*, vol. 74, no. 2, pp. 270–327, 1986.

[5] Giovanni De Sanctis and Augusto Sarti, "Virtual analog modeling in the wave-digital domain," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 715–727, May 2010.

[6] Kurt J. Werner, Julius O. Smith III, and Jonathan S. Abel, "Wave digital filter adaptors for arbitrary topologies and multiport linear elements," in *18th International Conference on Digital Audio Effects (DAFx-15)*, Trondheim, Norway, November 30 – December 3 2015.

[7] Kurt J. Werner, Vaibhav Nangia, Julius O. Smith III, and Jonathan S. Abel, "Resolving wave digital filters with multiple/multiport nonlinearities," in *18th International Conference on Digital Audio Effects (DAFx-15)*, Trondheim, Norway, November 30 – December 3 2015.

[8] Kurt J. Werner, Vaibhav Nangia, Julius O. Smith III, and Jonathan S. Abel, "A general and explicit formulation for wave digital filters with multiple/multiport nonlinearities and complicated topologies," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, October 18–21 2015.

[9] David T. Yeh and Julius O. Smith III, "Tutorial on wave digital filters," Available: `https://ccrma.stanford.edu/~dtyeh/papers/wdftutorial.pdf`, Jan. 25 2008.

[10] David T. Yeh and Julius O. Smith III, "Discretization of the '59 Fender Bassman tone stack," in *Proc. Int. Conf. Digital Audio Effects (DAFx-06)*, Montréal, Canada, September 18–20 2006.

[11] John M. Miller, *Dependence of the input impedance of a three-electrode vacuum tube upon the load in the plate circuit*, vol. 15, pp. 367–385, Government Printing Office, Washington, 1919.

[12] Rob Robinette, "How Fender normal/bright/hi/low input jacks work," Available: `https://robrobinette.com/How_Fender_Input_Jacks_Work.htm`.

[13] Matti Karjalainen and Jyri Pakarinen, "Wave digital simulation of a vacuum-tube amplifier," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France, May 14–19 2006, pp. 153–156.

[14] Peter Raffensperger, "Toward a wave digital filter model of the fairchild 670 limiter," in *15th International Conference on Digital Audio Effects (DAFx-12)*, York, UK, September 17–21 2012.

[15] Jyri Pakarinen and Matti Karjalainen, "Enhanced wave digital triode model for real-time tube amplifier emulation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 738–746, May 2010.

[16] Norman Koren, "Improved vacuum tube models for spice simulation," Available: `http://www.normankoren.com/Audio/Tubemodspice_article.html`.

[17] Jyri Pakarinen and Matti Karjalainen, "Wave digital modeling of the output chain of a vacuum-tube amplifier," in *12th International Conference on Digital Audio Effects (DAFx-09)*, Como, Italy, September 1–4 2009.

[18] Stefano D'Angelo, Jyri Pakarinen, and Vesa Välimäki, "New family of wave-digital triode models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 2, pp. 313–321, February 2013.

[19] Gian-Carlo Cardarilli, Marco Re, and Leonardo Di Carlo, "Improved large-signal model for vacuum triodes," in *International Symposium on Circuits and Systems (ICSCAS)*, Taipei, Taiwan, May 24–27 2009.

[20] Chung-Wen Ho, Albert Ruelhi, and Pierce Brennan, "The modified nodal approach to network analysis," *IEEE Transactions on Circuits and Systems*, vol. 22, no. 6, pp. 504–509, 1975.

[21] Dietrich Fränken, Jörg Ochs, and Karlheinz Ochs, "Generation of wave digital structures for networks containing multiport elements," *IEEE Transactions on Circuits and Systems–I: Regular Papers*, vol. 52, no. 3, pp. 586–596, 2005.

[22] W. Ross Dunkel, "Bassman/JTM 45 preamp WDF study supplemental materials for DAFx 2016," Available: `https://ccrma.stanford.edu/~chigi22/bassmanDAFx16/supplementalMaterials.html`.

[23] Michael Jørgen Olsen, Kurt James Werner, and Julius O. Smith III, "Resolving grouped nonlinearities in wave digital filters using iterative techniques," in *19th International Conference on Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, September 5–9 2016.

[24] Kristjan Dempwolf and Udo Zölzer, "A physically-motivated triode model for circuit simulations," in *14th International Conference on Digital Audio Effects (DAFx-11)*, Paris, France, September 19–23 2011.

[25] Thomas Jungmann, "Theoretical and practical studies on the behavior of electric guitar pickups," M.S. thesis, Department of Electrical Engineering, Acoustics Laboratory, Helsinki University of Technology, Espoo, Finland, 1994.

[26] Julius O. Smith III and Xavier Serra, "Parshl: An analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation," Available: `https://ccrma.stanford.edu/~jos/parshl/`.

[27] Mototsugu Abe and Julius O. Smith III, "Design criteria for simple sinusoidal parameter estimation based on quadratic interpolation of fft magnitude peaks," in *117th Convention of the Audio Engineering Society*, San Francisco, CA, October 28–31 2004.

[28] Angelo Farina, "Simultaneous measurement of impulse response and distortion with a swept-sine technique," in *108th Convention of the Audio Engineering Society*, Paris, France, February 19–22 2000.

[29] Kurt J. Werner and W. Ross Dunkel, "A computational model of the hammond organ vibrato/chorus using wave digital filters," in *19th International Conference on Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, September 5–9 2016.

[30] Maximilian Rest, W. Ross Dunkel, Kurt J. Werner, and Julius O. Smith III, "RT-WDF—a modular wave digital filter library with support for arbitrary topologies and multiple nonlinearities," in *19th International Conference on Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, September 5–9 2016.

A COMPUTATIONAL MODEL OF THE HAMMOND ORGAN VIBRATO/CHORUS USING WAVE DIGITAL FILTERS

Kurt James Werner, W. Ross Dunkel, and François G. Germain

Center for Computer Research in Music and Acoustics (CCRMA), Stanford University
660 Lomita Drive, Stanford, CA 94305, USA
[kwerner, chigi22, francois]@ccrma.stanford.edu

ABSTRACT

We present a computational model of the Hammond tonewheel organ vibrato/chorus, a musical audio effect comprising an LC ladder circuit and an electromechanical scanner. We model the LC ladder using the Wave Digital Filter (WDF) formalism, and introduce a new approach to resolving multiple nonadaptable linear elements at the root of a WDF tree. Additionally we formalize how to apply the well-known warped Bilinear Transform to WDF discretization of capacitors and inductors and review WDF polarity inverters. To model the scanner we propose a simplified and physically-informed approach. We discuss the time- and frequency-domain behavior of the model, emphasizing the spectral properties of interpolation between the taps of the LC ladder.

1. INTRODUCTION

The Hammond tonewheel organ's vibrato/chorus[1] (Fig. 1, Table 1) is a crucial ingredient of its unique sound. Its sonic character is highly valued by musicians, having even been made into a guitar effect [2]. The vibrato/chorus consists of an LC ladder circuit (Fig. 1) and an electromechanical "scanner" [3], with three user-selectable "vibrato" (V1, V2, V3) and "chorus" (C1, C2, C3) settings. In this paper, we introduce a model of the Hammond organ vibrato/chorus comprising a Wave Digital Filter (WDF) [4] model of the LC ladder circuit and a simplified model of the scanner.

WDF theory was originally developed to facilitate the design of digital filters based on analog ladder prototypes [5]. In that context, the low coefficient sensitivity of these prototypes leads to attractive numerical properties in the WDF. Recent years have seen an expansion of the use of WDFs into new fields including virtual analog circuit modeling [6]. Interestingly, ladder topologies also show up in electro-mechanical equivalent circuit models of the torsional modes of spring vibration relevant to spring reverb units [7], another effect common in Hammond organs.

Modeling the Hammond organ LC ladder as a WDF presents an issue that suggests an extension to WDF theory, and an opportunity to discuss finer points of polarity handling and reactance discretization. First, the ladder circuit has two non-adaptable linear elements (a voltage source and a switch), one more than classical WDF methods can handle. To address this, we extend the method of [8] to the case of multiple linear nonadaptable elements at the root of a WDF tree. Second, the circuit's 36 reactances create magnitude responses with numerous salient features. We apply the well-known frequency-warped bilinear transform to the wave-digital capacitor and inductor to help control magnitude response matching. Finally, polarity bookkeeping of port connections and the 19 outputs of the LC ladder is non-trivial. Since it is essential to get each port's polarity correct and to simplify the calculation of node voltages, we review the derivation of wave-digital polarity inverters and illustrate their systematic use.

Although the vibrato/chorus has not been studied in the virtual analog context, there exists extensive related work on modeling other aspects of the complex and pleasingly idiosyncratic sound of the Hammond organ. For the practicing musician, a series of five Sound on Sound articles (beginning with [9]) details how to mimic each sub-system of the Hammond from tonewheel to Leslie speaker using standard synthesis tools. [10] points out the difficulty of emulating the vibrato/chorus using a standard digital chorus. Numerous commercial emulations known as "clonewheel organs" have been released over the years. Academic papers have covered various aspects of the Hammond sound. Pekonen *et al.* [11] propose efficient models of the organ's basic apparatus including tonewheels draw-bars. More abstractly, a novel "Hammondizer" effect by Werner and Abel [12] imprints the sonic characteristics of the organ onto any input audio, extending effect processing [13] within a modal reverberator framework [14]. An important part of the organ's sound, the Leslie rotating speaker [15] has been the subject of the majority of Hammond-related academic work. Its simulation has been tackled using a perceptual approach [16], modulated and interpolated delay lines [17, 18][2], Doppler shift and amplitude modulation [19, 20], a measurement-based black box approach [21], and spectral delay filters [11].

The paper is structured as follows. Section 2 details the Hammond vibrato/chorus. Section 3 presents a simplified model of the scanner. Section 4 presents a WDF model of the LC ladder circuit. Section 5 characterizes these models.

2. REFERENCE SYSTEM DESCRIPTION

This section details the Hammond Organ vibrato/chorus, which includes a LC ladder circuit (Fig. 1, bottom, Section 2.1) and an electromechanical "scanner" apparatus (Fig. 1, top, Section 2.2). The gray box on Fig. 1 represents a bank of switches that connect the tap node voltages $v_1 \cdots v_{19}$ on the ladder to the terminals $t_1 \cdots t_9$ on the scanner. The setting (V1/V2/V3/C1/C2/C3) controls these switches according to Table 2.

In principle, the LC ladder serves the same purpose as the delay line in a standard digital chorus effect [22]. The LC ladder differs from a delay line in that the LC ladder is not strictly unidirectional and that it filters as it delays a signal. This filtering features pronounced non-uniform passband ripples and a lowpass cutoff that depends on the inductor and capacitor values.

On the other hand, the scanner serves the same purpose as interpolation in a standard digital modulated-delay effect [22]. Stan-

[1] We study the version used in late-model Hammond B-3s [1]

[2] https://ccrma.stanford.edu/ jos/pasp/Leslie.html

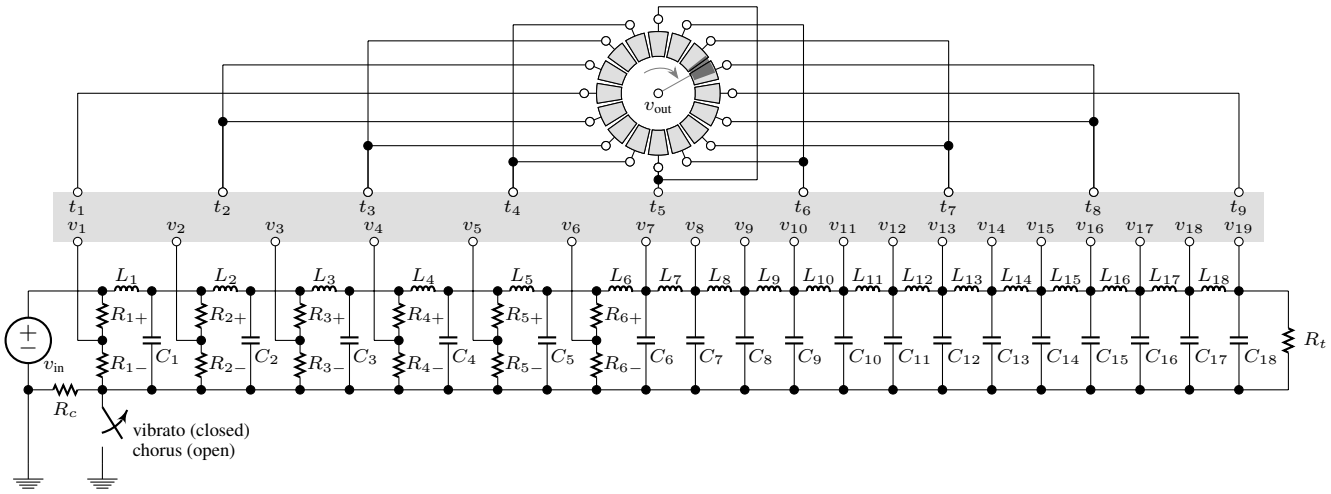


Figure 1: *Vibrato/Chorus Schematic.*

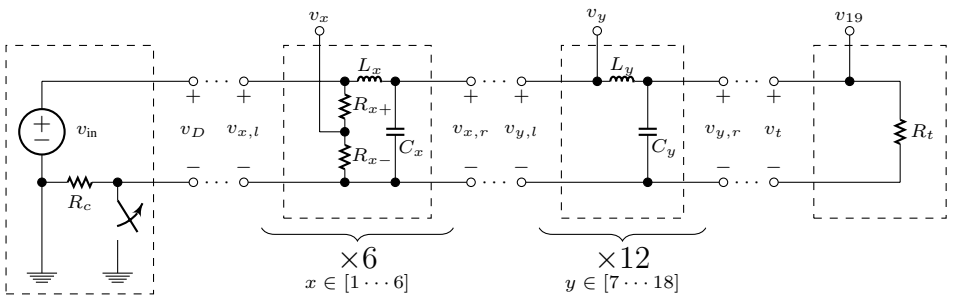


Figure 2: *Vibrato/Chorus Schematic Partitioned.*

Table 1: *Component values.*

Name	value	units
R_c	22	kΩ
R_{1+}	27	kΩ
R_{1-}	68	kΩ
R_{2+}	56	kΩ
R_{3+}	39	kΩ
R_{2-}, R_{3-}	0.15	MΩ
R_{4+}	33	kΩ
R_{5+}	18	kΩ
R_{6+}	12	kΩ
$R_{4-} \cdots R_{6-}$	0.18	MΩ
$L_1 \cdots L_{18}$	500	mH
$C_1 \cdots C_{17}$	0.004	μF
C_{18}	0.001	μF
R_t	15	kΩ

dard digital linear interpolation has a well-known lowpass characteristic [18][3] that digital audio effect designers often try to avoid by using, e.g., allpass interpolation [18][4]. Ironically, the scanner of the Hammond Organ vibrato/chorus essentially implements linear interpolation—meaning it does *not* have an allpass characteristic.

Table 2: *Taps for different depth settings.*

depth	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
V1/C1	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
V2/C2	v_1	v_2	v_3	v_5	v_7	v_9	v_{11}	v_{12}	v_{13}
V3/C3	v_1	v_2	v_4	v_7	v_{10}	v_{13}	v_{16}	v_{18}	v_{19}

2.1. LC Ladder Circuit

The input signal is represented as an ideal voltage source v_{in}. 19 LC ladder stages are composed of inductors $L_1 \cdots L_{19}$, capacitors $C_1 \cdots C_{19}$, and voltage divider pairs R_{k+} and R_{k-}, $k \in [1 \cdots 6]$. A termination resistor R_t ends the ladder. A switch controls whether R_c is shorted or not. Electrical component values for the circuit are given in Table 1 [1].

[3] https://ccrma.stanford.edu/~jos/pasp/Fractional_Delay_Filtering_Linear.html

[4] https://ccrma.stanford.edu/~jos/pasp/First_Order_Allpass_Interpolation.html

This highly structured circuit is partitioned into four subcircuits as shown in Fig. 2. The first subcircuit includes v_{in}, R_c, and the switch and presents a port "D" to the rest of the circuit.

The second subcircuit has 6 stages indexed by $x \in [1 \cdots 6]$: inductor L_x, capacitor C_x, and voltage divider pair R_{x+} and R_{x-}. The tap node voltage v_x is the output of each stage. Each stage presents a left-facing ("x, l") and right-facing ("x, r") port to the rest of the circuit. Ports "D" and "$1, l$" are connected and the 5 port pairs "$(k+1), l$" and "k, r", $k \in [2 \cdots 6]$ are connected.

The third subcircuit has 12 stages indexed by $y \in [7 \cdots 18]$: inductor L_y and capacitor C_y. The tap node voltage v_y is the output of each stage. Each stage presents a left-facing ("y, l") and right-facing ("y, r") port to the rest of the circuit. Ports "$6, r$" and "$7, l$" are connected and the 11 port pairs "$(k+1), l$" and "k, r", $k \in [7 \cdots 17]$ are connected.

The fourth subcircuit is simply the termination resistance R_t that presents port t to the rest of the circuit and has the tap node voltage v_{19} as an output. Ports "$18, r$" and "t" are connected.

2.2. Scanner Device

The vibrato scanner consists of a moving rotor with node voltage v_{out} that cyclically scans a stack of keystone-shaped output plates across 16 fixed stacks of identical plates arranged in a circle. At any given time, 2 of these 16 stacks partially overlap the rotor stack, forming two capacitors whose capacitances are pro-

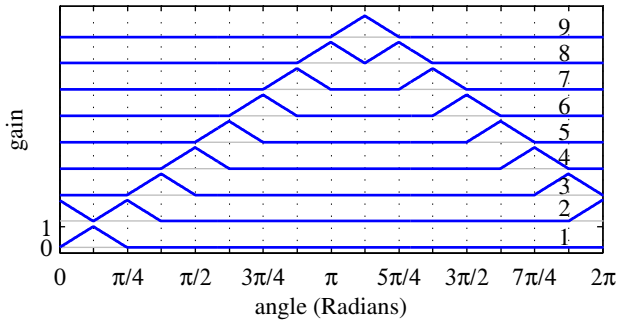


Figure 3: *Gain applied to each scanner terminal* $t_1 \cdots t_9$.

portional to the overlapping area between each fixed stack and the rotor stack. Conceptually, these two capacitances form a time-varying capacitive voltage divider, which crossfades between the node voltages of these 2 stacks. The 16 fixed plate stacks are connected to the 9 terminals $t_1 \cdots t_9$ which sets their respective node voltages to the node voltages of the corresponding terminals. As the rotor undergoes a complete rotation, it "scans" from t_1 to t_9 and back. The "there-and-back-again" form of the tap gains produces various cyclic vibrato effects.

3. SCANNER MODEL

In this section, we propose a simplified model of the scanner. Since the scanner capacitances are small compared to the ladder capacitors, it is reasonable to assume that they don't load the ladder circuit. Therefore, we are justified in modeling the LC ladder and scanner separately. The scanner takes the 19 node voltages $v_1 \cdots v_{19}$ as input and produces a single output voltage v_{out}.

At any given time, two plates overlap the rotor plate, creating two distinct capacitors. We assume that only these two capacitors have a non-negligible contribution to the output voltage. Under that assumption, those two form a capacitive voltage divider. To reflect this, we model the scanner output as a crossfade between two node voltages v_α and v_β according to $v_{\text{out}} = \eta\, v_\alpha + (1-\eta) v_\beta$. The mixing coefficient $\eta \in [0.0, 1.0]$ varies with the two capacitances C_α and C_β as $\eta = C_\alpha/(C_\alpha + C_\beta)$. Assuming circular symmetry of each plate and ignoring the small gap between stacks, $C_\alpha + C_\beta$ is a constant. Since each capacitance is theoretically proportional to the overlap area, the gain is modeled as a simple function of the rotor angle, according to Fig. 3. The variation of rotor angle over time follows the one in the physical organ, where the rotor is driven at a constant rate of ≈ 6 Hz by a synchronous motor [3].

4. WDF MODEL OF LC LADDER

In this section, we detail WDF simulation of the LC ladder circuit. The structure for subcircuits two, three, and four is outlined in Section 4.1. The first subcircuit contains multiple nonadaptable linear elements and cannot be handled with standard WDF techniques. We introduce a new approach to resolving this issue in Section 4.2 and apply it to the LC ladder in Section 4.3. In Section 4.4 we use the frequency-warped bilinear transform for WDF discretization of capacitors and inductors; in Section 4.5 we review WDF polarity inverters which are essential for proper bookkeeping.

4.1. WDF Tree (Subcircuits 2–4)

To model the LC ladder circuit, we derive the WDF structure of its circuit. Fig. 4 shows the partitioned schematic rearranged to highlight the underlying topology (with polarities labeled) and Fig. 5 shows the resulting WDF structure.

The 6 stages in the second subcircuit can be decomposed into standard WDF one-ports (R_{x-}, R_{x+}, L_x, and C_x) and adaptors ($\mathcal{S}_x$, $\mathcal{S}_{x\prime}$, $\mathcal{P}_x$, and $\mathcal{P}_{x\prime}$). The presence of the two inverters $\mathcal{I}_x$ and $\mathcal{I}_{x\prime}$ warrants explanation. We have already assigned polarities to the ports that connect stages, and series and parallel adaptors have inherent port polarities. Inverters $\mathcal{I}_x$ reconcile the discrepancy between the polarity of the right-facing port of each parallel adaptor $\mathcal{P}_x$ and the left-facing port of $\mathcal{P}_{(x+1)\prime}$ or $\mathcal{S}_7$. Inverters $\mathcal{I}_{x\prime}$ simplify the extraction of node voltages v_x, which are calculated by combining port voltages across resistors R_c and R_{x-} as

$$v_x = v_c + v_{x-} = \frac{1}{2}\left(a_c + b_c + a_{x-} + b_{x-}\right) , \tag{1}$$

where v_c, a_c, and b_c are the port voltage across, incident wave, and reflected wave at resistor R_c in the first subcircuit.

The 12 stages in the third subcircuit can be decomposed into standard WDF one-ports (L_y and C_y) and adaptors ($\mathcal{S}_y$ and $\mathcal{P}_y$). Again inverters $\mathcal{I}_y$ reconcile the discrepancy between the polarity of the right-facing port of each $\mathcal{P}_y$ and the left-facing port of $\mathcal{S}_{y+1}$ or R_t. Node voltages v_y in this subcircuit are extracted by combining the port voltages of resistor R_c and the left-facing port voltage of each stage $v_{y,l}$ as

$$v_y = v_c + v_{y,l} = \frac{1}{2}\left(a_c + b_c + a_{y,l} + b_{y,l}\right) . \tag{2}$$

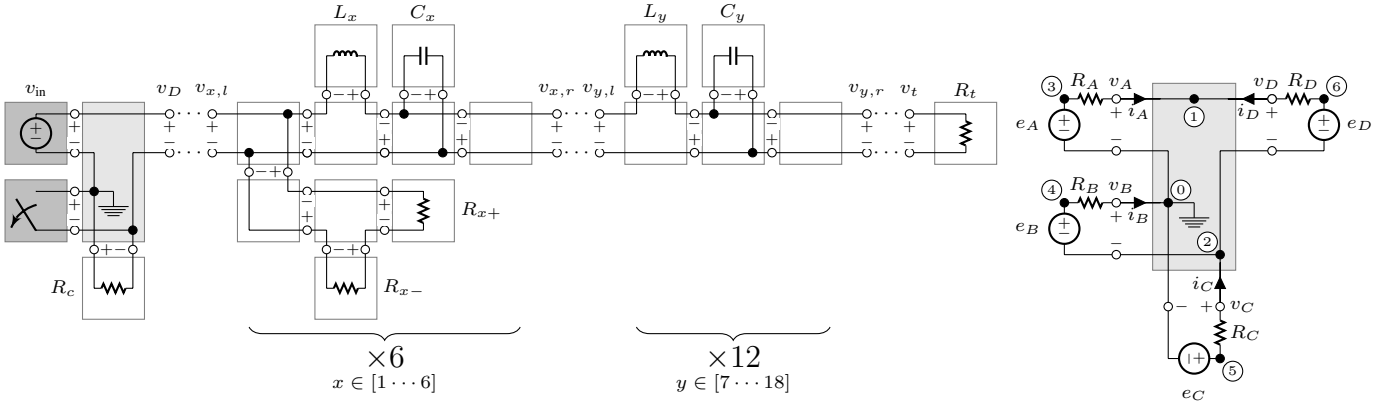
The fourth subcircuit is decomposed simply into a WDF resistor R_t. Node voltage v_{19} is extracted by

$$v_{19} = v_c + v_t = \frac{1}{2}\left(a_c + b_c + a_t + b_t\right) . \tag{3}$$

4.2. Root with Multiple Elements

Reference circuits such as the Hammond organ vibrato/chorus circuit commonly include multiple nonadaptable elements (linear and nonlinear). Trying to accomodate multiple nonadaptable elements in a standard WDF connection tree causes unavoidable delay-free loops which leads to computability problems.

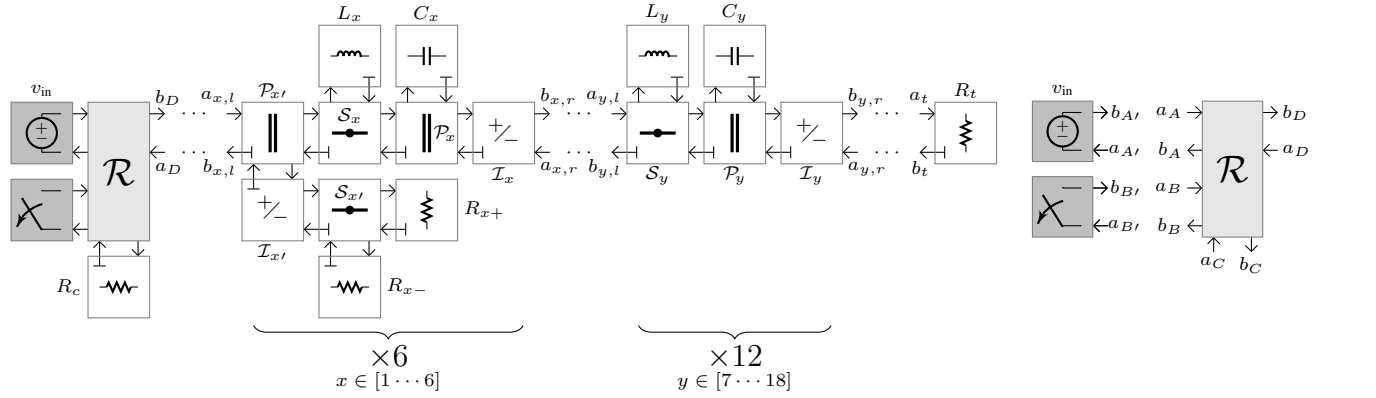
Historically, algorithm designers commonly use one of two tactics to ameliorate these issues. One tactic is to alter the reference circuit to make the structure computable. It is common to approximate ideal voltage sources as resistive voltage sources with small series resistances and to approximate ideal current sources as resistive current sources with large parallel resistances. The same principle can be used to approximate short circuits or the closed state of switches as small resistances and to approximate open circuits or the open state of switches as large resistances. Furthermore, certain nonlinear elements can be reasonably approximated by linearizing them with controlled sources and immitances [6,23]. A second tactic is to alter the WDF by introducing fictitious unit delays to resolve delay-free loops. Fettweis used this approach [5] before developing reflection-free ports [24], and it is still common in virtual analog [25, 26]. Of course, altering the reference circuit through these tactics introduces error (e.g. dissipation, dispersion) and can have adverse effects on stability.



(a) Highlight WDF structure

(b) Thévenin equivalents attached.

Figure 4: *LC ladder Schematic, rearranged towards WDF.*



(a) WDF structure

(b) Root Waves.

Figure 5: *WDF structure of LC ladder.*

In [8], Werner *et al.* propose a method for handling multiple nonlinearities that does not resort to these tactics. All of the nonlinearities are grouped as sub-elements of a WDF structure at the root of the WDF tree. Inside that structure, and after proper modification of the circuit graph, those elements end up being connected to each other through a complex $\mathcal{R}$-type adaptor that also interfaces those elements to the rest of the circuit. The method of [27] is used to solve for the scattering behavior of this $\mathcal{R}$-type adaptor. Because of the non-adaptable nature of the root elements, the response of the root adaptor structure from the perspective of the rest of the tree forms an implicit loop that we can resolve using either a tabulated solution [8] or an iterative solution [28, 29]. These approaches extends readily to nonadaptable linear elements, but is unnecessarily complex. Here we propose a novel more efficient approach for the case of multiple nonadaptable linear elements.

Consider a complex root topology with "external" incident waves $\boldsymbol{a}_e$ and reflected waves $\boldsymbol{b}_e$ facing the rest of the circuit and "internal" incident waves $\boldsymbol{a}_i$ and reflected waves $\boldsymbol{b}_i$ facing the nonadaptable linear elements, related by the scattering relationship

$$\begin{bmatrix} \boldsymbol{b}_i \\ \boldsymbol{b}_e \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{a}_i \\ \boldsymbol{a}_e \end{bmatrix} \tag{4}$$

The vector of nonadaptable linear elements relates the incident waves $\boldsymbol{a}_{\text{root}}$ and inputs $\boldsymbol{x}_{\text{root}}$ to reflected waves $\boldsymbol{b}_{\text{root}}$ by

$$\boldsymbol{b}_{\text{root}} = \boldsymbol{\Phi}\boldsymbol{a}_{\text{root}} + \boldsymbol{\Psi}\boldsymbol{x}_{\text{root}}\,, \tag{5}$$

where $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ embody the wave-domain behavior of the linear elements. $\boldsymbol{a}_{\text{root}}$ and $\boldsymbol{b}_{\text{root}}$ are related to the $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$ by

$$\boldsymbol{a}_{\text{root}} = \boldsymbol{b}_i \quad \text{and} \quad \boldsymbol{a}_i = \boldsymbol{b}_{\text{root}}\,. \tag{6}$$

Combining (4), (5), and (6) and solving for $\boldsymbol{b}_{\text{e}}$ yields

$$\begin{gathered} \boldsymbol{b}_{\text{e}} = \boldsymbol{\Gamma}\boldsymbol{a}_e + \boldsymbol{\Theta}\boldsymbol{x}_{\text{root}} \quad \text{with} \\ \boldsymbol{\Gamma} = \boldsymbol{\Theta}\mathbf{S}_{12} + \mathbf{S}_{22}\ ,\ \ \boldsymbol{\Theta} = \mathbf{S}_{21}\left(\mathbf{I} - \boldsymbol{\Phi}\mathbf{S}_{11}\right)^{-1}\boldsymbol{\Psi},\,. \end{gathered} \tag{7}$$

4.3. WDF Root (Subcircuit 1)

Here, we apply the theory developed in Section 4.2 to the first subcircuit of the Hammond vibrato/chorus. The first subcircuit contains two non-adaptable elements, a voltage source and a switch. As a result, those two elements need to be grouped at the root of the WDF structure following the method outlined in Section 4.2, connecting them through an $\mathcal{R}$-type adaptor [27] with incident and reflected waves

$$\boldsymbol{a} = \begin{bmatrix} \boldsymbol{a}_i^\top & \boldsymbol{a}_e^\top \end{bmatrix}^\top \quad \text{and} \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_i^\top & \boldsymbol{b}_e^\top \end{bmatrix}^\top\,, \tag{8}$$

$$
\begin{array}{c}
\\
⓪\\ ①\\ ②\\ ③\\ ④\\ ⑤\\ ⑥\\ A\\ B\\ C\\ D
\end{array}
\begin{array}{c}
\begin{array}{cccccccccccc}
⓪ & ① & ② & ③ & ④ & ⑤ & ⑥ & A & B & C & D
\end{array}\\
\left[\begin{array}{cccccccccccc}
G_B & 0 & 0 & 0 & -G_B & 0 & 0 & -1 & 0 & -1 & 0\\
0 & G_A+G_D & 0 & -G_A & 0 & 0 & -G_D & 0 & 0 & 0 & -1\\
0 & 0 & G_C & 0 & 0 & -G_C & 0 & 0 & -1 & 0 & 0\\
0 & -G_A & 0 & G_A & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
-G_B & 0 & 0 & 0 & G_B & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & -G_C & 0 & 0 & G_C & 0 & 0 & 0 & 1 & 0\\
0 & -G_D & 0 & 0 & 0 & 0 & G_D & 0 & 0 & 0 & 1\\
-1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\
-1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{array}\right]
\end{array}
\begin{bmatrix}
v_⓪\\ v_①\\ v_②\\ v_③\\ v_④\\ v_⑤\\ v_⑥\\ j_A\\ j_B\\ j_C\\ j_D
\end{bmatrix}
=
\begin{bmatrix}
0\\0\\0\\0\\0\\0\\0\\e_A\\e_B\\e_C\\e_D
\end{bmatrix}
$$

Figure 6: MNA matrix. Red and blue cells respectively show examples of resistor and voltage source "stamps" [27].

"external" wave vectors

$$\boldsymbol{a}_e = \begin{bmatrix} a_C & a_D \end{bmatrix}^\top \quad \text{and} \quad \boldsymbol{b}_e = \begin{bmatrix} b_C & b_D \end{bmatrix}^\top , \tag{9}$$

and "internal" wave vectors

$$\boldsymbol{a}_i = \begin{bmatrix} a_A & a_B \end{bmatrix}^\top \quad \text{and} \quad \boldsymbol{b}_i = \begin{bmatrix} b_A & b_B \end{bmatrix}^\top . \tag{10}$$

Using the method of [27], we can solve for the scattering matrix $\mathbf{S}$ that relates the incident waves $\boldsymbol{a}$ and reflected waves $\boldsymbol{b}$ as $\boldsymbol{b} = \mathbf{S}\boldsymbol{a}$. To do so, we attach instantaneous Thévenin port equivalents to each of the ports $A \cdots D$ (Fig. 4b) and confront Modified Nodal Analysis (the MNA system for Fig. 4b is shown in Fig. 6) with the standard voltage wave definition, yielding

$$\mathbf{S} = \mathbf{I} + 2 \begin{bmatrix} \mathbf{0} & \mathbf{R} \end{bmatrix} \mathbf{X}^{-1} \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix}^\top \tag{11}$$

where $\mathbf{R} = \operatorname{diag}\left([R_A \cdots R_D]\right)$ is the diagonal matrix of port resistances and $\mathbf{X}$ is the MNA system matrix.

The vector of nonadaptable linear elements includes the voltage source v_{in} and the switch, which relate "root" wave vectors

$$\boldsymbol{a}_{\text{root}} = \begin{bmatrix} a_{A'} & a_{B'} \end{bmatrix}^\top \quad \text{and} \quad \boldsymbol{b}_{\text{root}} = \begin{bmatrix} b_{A'} & b_{B'} \end{bmatrix}^\top . \tag{12}$$

As before, port connections enforce $\boldsymbol{a}_{\text{root}} = \boldsymbol{b}_i$ and $\boldsymbol{a}_i = \boldsymbol{b}_{\text{root}}$.

The ideal voltage source v_{in} has the wave-domain relationship

$$b_{A'}[n] = a_{A'}[n] + 2\, v_{\text{in}}[n]. \tag{13}$$

An ideal switch has the wave-domain relationship

$$b_{B'}[n] = w\, a_{B'}[n]\,, \quad w = \begin{cases} -1 & \text{open switch} \\ +1 & \text{closed switch} \end{cases} . \tag{14}$$

In the context of (5), (13)–(14) define

$$\boldsymbol{\Phi} = \begin{bmatrix} -1 & 0 \\ 0 & w \end{bmatrix} \quad , \quad \boldsymbol{\Psi} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \quad , \quad \boldsymbol{x}_{\text{root}} = \begin{bmatrix} v_{\text{in}} \\ 0 \end{bmatrix} . \tag{15}$$

Plugging (15) and (11) into (7) yields $\boldsymbol{b}_e$, solving the root topology with multiple nonadaptable linear elements.

4.4. Frequency-Warped One-Port Linear Reactances

Having solved the issue of realizing the WDF, we now turn our attention to discretization schemes for its reactances. The LC ladder's 36 reactances combine to create magnitude responses with numerous salient features, especially a sharp lowpass cutoff. To control the magnitude response's match to the reference domain, we apply the well-known frequency-warped bilinear transform to the wave-digital capacitor and inductor.

WDFs involve one-port ideal linear reactances: the capacitor (of capacitance C) and inductor (of inductance L). Their current–voltage relationships are:

$$C\dot{v}(t) = i(t) \quad \text{and} \quad v(t) = L\dot{i}(t) \tag{16}$$

where v is the port voltage, and i is the port current. Their corresponding Laplace transforms are:

$$Cs\mathcal{V}(s) = \mathcal{I}(s) \quad \text{and} \quad \mathcal{V}(s) = Ls\mathcal{I}(s)\,. \tag{17}$$

Plugging in the standard WDF voltage-wave definitions

$$a = v + Ri \quad \text{and} \quad b = v - Ri \tag{18}$$

parameterized by arbitrary port resistance R yields continuous-time transfer functions $\mathcal{H}(s) = \mathcal{B}(s)/\mathcal{A}(s)$:

$$\mathcal{H}_C(s) = \frac{1 - RCs}{1 + RCs} \quad \text{and} \quad \mathcal{H}_L(s) = \frac{R - Ls}{R + Ls} \tag{19}$$

To simulate the system, we discretize reactive elements to obtain $H(z^{-1}) = B(z^{-1})/A(z^{-1})$ for each. WDFs commonly use the bilinear transform (BLT) [4], which substitutes $\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}$ for s in $\mathcal{H}(s)$ to form $H(z^{-1})$ (T is the sampling period). The BLT's desirable numerical properties include transfer function order preservation, unconditional stability, and passivity in the WDF domain, but it suffers from a well-known frequency distortion [18].

A common extension to the BLT is the warped (or generalized) BLT which is identical except T is replaced by T' [30] so as to substitute $\frac{2}{T'}\frac{1-z^{-1}}{1+z^{-1}}$ for s. This degree of freedom is used to alter the BLT's frequency distortion and ensure that, by selecting T' properly, one continuous-time frequency Ω_0 is mapped correctly, i.e., $\mathcal{H}(j\Omega_0) = H(e^{-j\Omega_0 T})$. The coefficient T' that achieves the correct mapping is given by:

$$T' = 2\tan\left(\Omega_0 T/2\right)/\Omega_0\,. \tag{20}$$

The warped BLT has the same desirable numerical properties as the BLT. Since it is not common in the WDF context, we briefly develop warped BLT discretization of WDF one-port reactances.

One-port linear reactances have a first-order continuous-time transfer function, so the warped BLT yields a first-order transfer function in discrete time with z-transform

$$H(z^{-1}) = \frac{B(z^{-1})}{A(z^{-1})} = \frac{\beta_0 + \beta_1 z^{-1}}{\alpha_0 + \alpha_1 z^{-1}}\,. \tag{21}$$

For a capacitor C and inductor L, these coefficients are:

$$\text{C:} \quad \beta_0 = \alpha_1 = \frac{T'}{2C} - R \quad , \quad \beta_1 = \alpha_0 = \frac{T'}{2C} + R \tag{22}$$

$$\text{L:} \quad \beta_0 = -\alpha_1 = R + \frac{2L}{T'} \quad , \quad \beta_1 = -\alpha_0 = R - \frac{2L}{T'} \,. \tag{23}$$

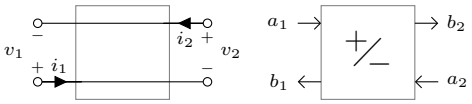
To eliminate delay-free loops, all one-port leaf elements of a WDF require *adaptation*: picking a value of R that satisfies $\beta_0 = 0$. The port impedances that adapt a capacitor and inductor are

$$R_C = T'/(2C) \quad \text{and} \quad R_L = 2L/T' \tag{24}$$

which yield discretized transfer functions

$$H_C(z^{-1}) = z^{-1} \quad \text{and} \quad H_L(z^{-1}) = -z^{-1} \,. \tag{25}$$

Interestingly, the discretized transfer functions of the capacitor and inductor do not depend on C, L, or T'. However, all of these *do* affect their adapted port resistance.



(a) Kirchhoff domain. (b) Wave domain.

Figure 7: *WDF 2-port series adaptor / inverter.*

4.5. Wave-Digital Inverter

We saw above that wave-digital polarity inverters must necessarily be employed for proper bookkeeping of port connection polarity and to simplify the calculation of node voltages. Here, we review the derivation of those inverters.

Consider two connected ports 1 and 2 with port voltages v_1 and v_2 and port currents i_1 and i_2; these ports can be connected in two ways. In the Kirchhoff domain, a two-port parallel connection is characterized by $v_1 = v_2$ and $i_1 = -i_2$ and a two-port series connection by $i_1 = i_2$ and $v_1 = -v_2$. Plugging in the standard WDF voltage wave definition (18) yields a scattering relationship

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \underbrace{\begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix}}_{\mathbf{S}} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} , \tag{26}$$

parameterized by the two port resistances R_1 and R_2. These two-port adaptors scatter according to

$$\mathbf{S} = \begin{bmatrix} -\frac{R_1 - R_2}{R_1 + R_2} & \frac{2\lambda R_1}{R_1 + R_2} \\ \frac{2\lambda R_2}{R_1 + R_2} & \frac{R_1 - R_2}{R_1 + R_2} \end{bmatrix} , \quad \lambda = \begin{cases} -1 & \text{series} \\ +1 & \text{parallel} \end{cases} \tag{27}$$

and are both rendered reflection-free by setting $R_1 = R_2$:

$$\begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix} = \begin{bmatrix} 0 & \lambda \\ \lambda & 0 \end{bmatrix} . \tag{28}$$

Notice that the reflection-free two-port parallel connection is simply a normal WDF port connection [4] with each incident wave equal to the opposite reflected wave. The two-port series connection inverts the reflected wave from each port to form the incident wave at the other port; it is in fact the wave-digital inverter (Fig. 7) [27, 31].

5. RESULTS

Here we discuss some results that characterize our model of the Hammond vibrato/chorus, including the impulse and magnitude responses of each tap in the LC ladder (Section 5.1), a study on the spectral aspects of scanner interpolation (Section 5.2), and the response to a single sinusoid (Section 5.3). These results reveal a variety of effects, including delay-length modulation, phaser-like effects, amplitude modulation, and modulated comb filter effects.

5.1. Impulse and Magnitude Responses of LC Ladder

Figs. 8 and 9 show the impulse and magnitude responses at each tap $v_1 \cdots v_{19}$ under two different WDF discretizations compared to a reference "ground truth" SPICE simulation.

In Fig. 8, we use a sampling rate of $f_s = 44100$ Hz, with the capacitors and inductors discretized using the standard BLT with no frequency warping, i.e., $T' = T = 1/f_s \approx 2.2676 \times 10^{-5}$. In Fig. 9, use use instead a warped BLT with T' chosen to match the frequency $\Omega_0 = 7075$ Hz, approximately the passband edge of the ladder, yielding $T' \approx 2.2724 \times 10^{-5}$ (20).

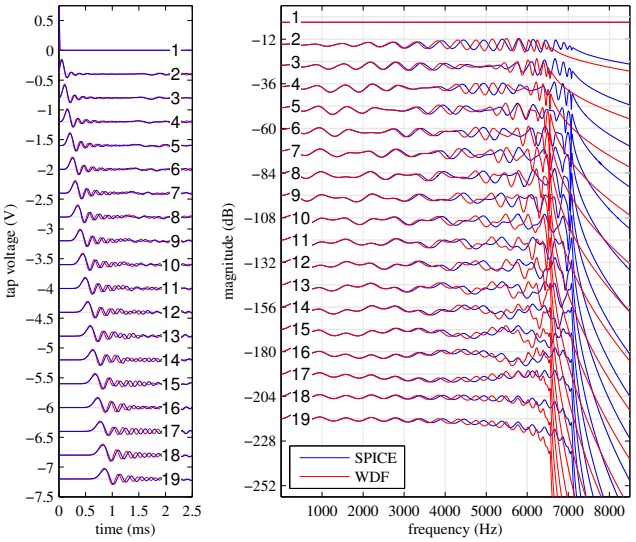
In the time domain plots, it can be seen that the LC ladder approximates a delay line. In theory, LC ladders have an idealized total delay time of $\sqrt{\sum L \times \sum C}$ [32], meaning ≈ 0.85 ms for the Hammond vibrato/chorus. It can be seen in the SPICE simulations that the impulse is delayed and "smeared" progressively as it travels down the line, and indeed experiences ≈ 0.85 ms of delay by tap 19. To understand the complex nature of this smearing, we turn to the magnitude response.

In the magnitude response, the lowpass characteristic of the LC ladder is apparent. In the SPICE simulations, the passband edge frequency is ≈ 7075 Hz. The amount of attenuation in the stopband depends on tap index: v_1 has no attenuation, and the slope increases as tap index increases. Notice that in the simulation using the unwarped BLT, dc is matched perfectly, while frequency distortion builds up as frequency increases. Specifically, the passband edge is depressed by almost 500 Hz compared to the SPICE simulation. Using the warped BLT, 7075 Hz is matched perfectly. While matching the passband edge may be preferable due to its perceptual salience, a mismatch remains for the rest of the magnitude response, most noticeably between dc and the passband edge. While the passband has *dozens* of features, the warped BLT can only match *one*. Notice that, back in the time domain, the frequency warpings of different discretizations manifests as different smearings. Alternatively, applying $4\times$ oversampling is an effective though expensive way to achieve good agreement from dc to the passband edge.

5.2. Magnitude Response of Scanner Model Interpolation

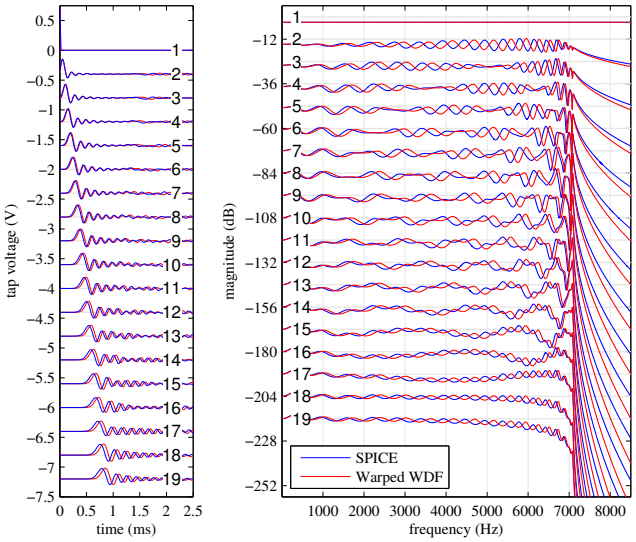
Fig. 10 shows the magnitude response of scanner model interpolation between terminals for the V1 (Fig. 10a), V2 (Fig. 10b), and V3 (Fig. 10c) settings (using the unwarped BLT). dB markings are shown on the color axis. The horizontal axis represents the scanner angle θ. At the vertical markings with tap indices labeled underneath, the scanner is exactly on one of the terminals. Between tap indices are interpolations between them.

In addition to providing a time-varying delay, the ladder circuit and scanner impart complex spectral coloration. First, the sharp passband edge is modulated slightly over the course of each vibrato cycle. The passband ripples also follow complex trajectories during each cycle. Since the ripples are relatively deep (many



(a) Impulse response. (b) Magnitude response.

Figure 8: *Responses of the LC ladder, using unwarped BLT.*



(a) Impulse response. (b) Magnitude response.

Figure 9: *Responses of the LC ladder, using warped BLT.*

around 6 dB and some larger), they create an audible phaser-like effect on broadband input signals. The voltage dividers R_{k-} and R_{k+}, $k \in 1 \cdots 6$, produce amplitude modulation during each cycle of θ [10]. Table 3 shows the gain of each stage's divider.

Table 3: *Voltage divider gains (in dB) at each tap.*

tap	1	2	3	4	5	6	$7 \cdots 19$
gain	-2.9	-2.8	-2.0	-1.5	-0.83	-0.56	0

Since the Hammond vibrato/chorus approximates a delay line it is not surprising that the magnitude response of the scanner interpolation exhibits comb-filter-like features. Assuming the idealized delay time discussed earlier, linear interpolation would produce notches halfway through each crossfade at frequencies dictated by the separation between taps, at the locations indicated by $\times$ symbols. The actual minima in the Hammond response are very close to these notches, as predicted.

5.3. Sinusoid Study

We study a single 1760 Hz (A6 in scientific pitch notation) sinusoidal input for the three vibrato depth settings (Fig. 11). Notice that the V1 setting produces the narrowest vibrato, the V2 setting produces a medium vibrato, and the V3 setting produces the widest vibrato. Notice also that each setting produces a differently shaped vibrato. The vibrato width and shape are a consequence of the different tap spacings of each setting; the time-varying phase shift for a given frequency, which manifests as frequency modulation, is proportional to the time derivative of its group delay [19]. The combination of amplitude modulation and frequency modulation is visible as 6×16 Hz spaced sidebands around the main signal.

6. CONCLUSION

In this study on modeling the Hammond organ vibrato/chorus, we introduced new theoretical tools enabling the inclusion of multiple linear nonadaptable elements at the root of a WDF tree, applied the well-known frequency-warped bilinear transform to the derivation of wave-digital capacitors and inductors, and illustrated the systematic use of wave-digital polarity inverters. Although beyond the scope of this paper, the complex spectral properties and frequency-dependent vibrato of the Hammond organ vibrato/chorus deserve further study (cf. the complexities of vocal vibrato, including "spectral modulation" [33]).

7. ACKNOWLEDGMENTS

Thanks to Jonathan Abel for helpful discussions on ladder circuits.

8. REFERENCES

[1] "Service manual—models: A, A-100, AB, BA, BC, BCV, BV, B2, B3, C, CV, C2, C-2G, C3, D, DV, D-100, E, G, GV, RT, RT-2, RT-3," Tech. Rep. HOOO-OOO495, Nov. 1987.

[2] S. Hammond, "Analog Outfitters scanner review," *Premier Guitar*, Dec. 18 2015.

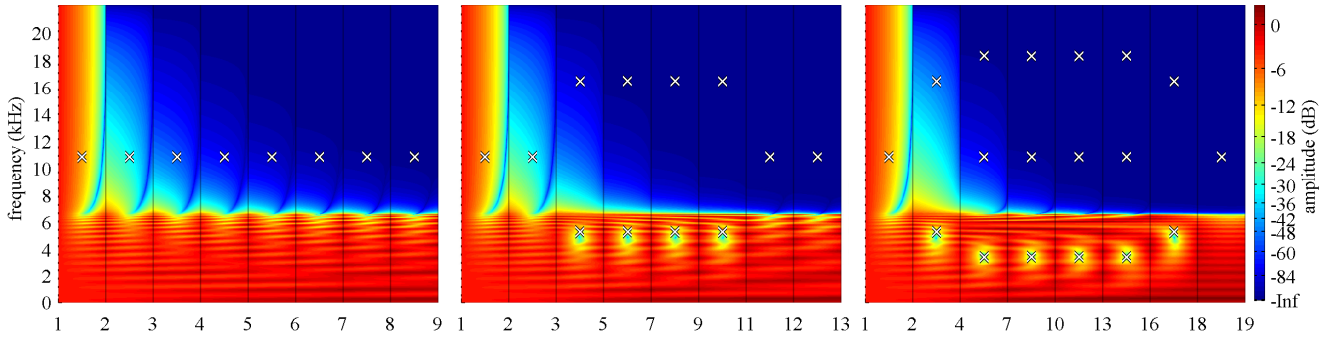
[3] J. M. Hanert, "Electrical musical apparatus," Aug. 14 1945, U.S. Patent No. 2,382,413.

[4] A. Fettweis, "Wave digital filters: Theory and practice," *Proc. IEEE*, vol. 74, no. 2, pp. 270–327, 1986.

[5] A. Fettweis, "Digital filters structures related to classical filter networks," *Archiv Elektronik Übertragungstechnik (AEÜ)*, vol. 25, pp. 79–89, 1971.

[6] G. De Sanctis and A. Sarti, "Virtual analog modeling in the wave-digital domain," *IEEE Trans. Audio, Speech, Language Process. (TASLP)*, vol. 18, no. 4, pp. 715–727, 2010.

[7] H. E. Meinema and W. C. Johnson, H. A. .and Laube Jr., "A new reverberation device for high fidelity systems," *J. Audio Eng. Soc. (JAES)*, vol. 9, no. 4, pp. 284–326, 1961.



(a) V1 Setting. (b) V2 Setting. (c) V3 Setting.

Figure 10: *Magnitude response at various interpolations. Recall Fig. 3 and Table 2.*

[8] K. J. Werner, V. Nangia, J. O. Smith III, and J. S. Abel, "Resolving wave digital filters with multiple/multiport nonlinearities," in *Proc. 18th Int. Conf. on Digital Audio Effects (DAFx)*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[9] G. Reid, "Synthesizing tonewheel organs," *Sound on Sound (SOS)*, Nov. 2003.

[10] G. Reid, "Synthesizing Hammond organ effects: Part 1," *SOS*, Jan. 2004.

[11] J. Pekonen, T. Pihlajamäki, and V. Välimäki, "Computationally efficient Hammond organ synthesis," in *Proc. 14th DAFx*, Paris, France, Sept. 19–23 2011.

[12] K. J. Werner and J. S. Abel, "Modal processor effects inspired by Hammond tonewheel organs," *Appl. Sci.*, vol. 6, no. 7 (Special Issue "Audio Signal Processing"), 2016.

[13] J. S. Abel and K. J. Werner, "Distortion and pitch processing using a modal reverb architecture," in *Proc. 18th DAFx*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[14] J. S. Abel, S. Coffin, and K. S. Spratt, "A modal architecture for artificial reverberation with application to room acoustics modeling," in *Proc. 137th AES*, Los Angeles, CA, Oct. 9–12 2014.

[15] D. J. Leslie, "Rotatable tremulant sound producer," Nov. 29 1949, U.S. Patent No. 2,489,653.

[16] R. Kronland-Martinet and T. Voinier, "Real-time perceptual simulation of moving sources: Application to the Leslie cabinet and 3D sound immersion," *EURASIP J. Audio, Speech, Music Process.*, 2008, Article ID 849696.

[17] J. Smith, S. Serafin, J. Abel, and D. Berners, "Doppler simulation and the Leslie," in *Proc. 5th DAFx*, Hamburg, Germany, Sept. 26–28 2002.

[18] J. O. Smith III, *Physical Audio Signal Processing for Virtual Musical Instruments and Audio Effects*, Online book, 2010 edition.

[19] S. Disch and U. Zölzer, "Modulation and delay line based digital audio effects," in *Proc. 2nd DAFx*, Trondheim, Norway, Dec. 9–11 1999.

[20] P. Dutilleux, M. Holters, S. Disch, and U. Zölzer, *Modulators and demodulators*, chapter 3, pp. 83–99, 2011, appears in [34].

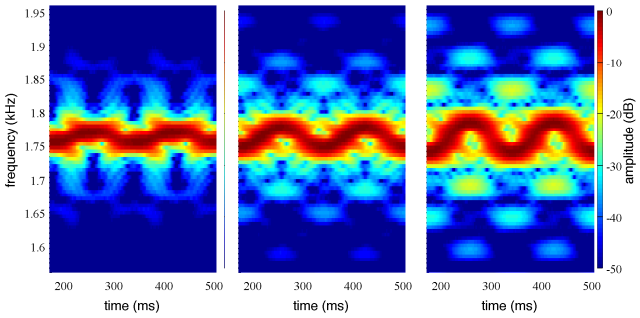
[21] J. Herrera, C. Hanson, and S. Abel, J. "Discrete time emulation of the Leslie speaker," in *Proc. 127th Conv. Audio Eng. Soc. (AES)*, New York, NY, USA, Oct. 9–12 2009, Conv. Paper 7925.

[22] J. Dattorro, "Effect design part 2: Delay-line modulation and chorus," *JAES*, vol. 45, no. 10, pp. 764–768, Oct. 1997.

[23] K. J. Werner, W. R. Dunkel, M. Rest, M. J. Olsen, and J. O. Smith III, "Wave digital filter modeling of circuits with operational amplifiers," in *Proc. European Signal Process. Conf. (EUSIPCO)*, Budapest, Hungary, Aug. 29 – Sept. 2 2016.

[24] A. Fettweis and K. Meerkötter, "On adaptors for wave digital filters," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 23, no. 6, 1975.

[25] R. C. D. de Paiva, J. Pakarinen, V. Välimäki, and M. Tikander, "Real-time audio transformer emulation for virtual tube amplifiers," *EURASIP J. Advances Signal Process.*, 2011.



(a) V1 setting. (b) V2 setting. (c) V3 setting.

Figure 11: *Studying a single* 1760 *Hz sinusoid.*

[26] J. Pakarinen and M. Karjalainen, "Enhanced wave digital triode model for real-time tube amplifier emulation," *IEEE TASLP*, vol. 18, no. 4, pp. 738–746, 2010.

[27] K. J. Werner, J. O. Smith III, and J. S. Abel, "Wave digital filter adaptors for arbitrary topologies and multiport linear elements," in *Proc. 18th DAFx*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[28] M. J. Olsen, K. J. Werner, and J. O. Smith III, "An iterative approach to resolving wave digital filters with grouped nonlinearities," in *Proc. 19th DAFx*, Brno, Czech Republic, Sept. 5–9 2016.

[29] W. R. Dunkel, M. Rest, K. J. Werner, M. J. Olsen, and J. O. Smith III, "The Fender Bassman 5F6-A family of preamplifier circuits—a wave digital filter case study," in *Proc. 19th DAFx*, Brno, Czech Republic, Sept. 5–9.

[30] F. G. Germain and K. J. Werner, "Design principles for lumped model discretization using Möbius transforms," in *Proc. 18th DAFx*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[31] S. D'Angelo and V. Välimäki, "Wave-digital polarity and current inverters and their application to virtual analog audio processing," in *IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Kyoto, Japan, Mar. 2012, pp. 469–472.

[32] Rhombus Industries Inc., "Passive delay line design considerations," Tech. Rep. APP1_PAS, Huntington Beach, CA, Jan. 1998.

[33] R. Maher and J. Beauchamp, "An investigation of vocal vibrato for synthesis," *Appl. Acoust.*, vol. 30, pp. 219–245, 1990.

[34] U. Zölzer, Ed., *DAFX: Digital Audio Effects*, John Wiley & Sons, second edition, 2011.

RESOLVING GROUPED NONLINEARITIES IN WAVE DIGITAL FILTERS USING ITERATIVE TECHNIQUES

Michael Jørgen Olsen, Kurt James Werner and Julius O. Smith III

Center for Computer Research in Music and Acoustics (CCRMA), Stanford University
660 Lomita Drive, Stanford, CA 94305, USA
[mjolsen|kwerner|jos]@ccrma.stanford.edu

ABSTRACT

In this paper, iterative zero-finding techniques are proposed to resolve groups of nonlinearities occurring in Wave Digital Filters. Two variants of Newton's method are proposed and their suitability towards solving the grouped nonlinearities is analyzed. The feasibility of the approach with implications for WDFs containing multiple nonlinearities is demonstrated via case studies investigating the mathematical properties and numerical performance of reference circuits containing diodes and transistors; asymmetric and symmetric diode clippers and a common emitter amplifier.

1. INTRODUCTION

The Wave Digital Filter concept is a method for digitizing analog reference circuits. WDFs were originally developed in the 1970s with the intention of realizing digital lattice and ladder filter topologies [1] and, as such, were developed to preserve modularity and properties analogous to passivity and losslessness inherent in the analog prototype circuits being modeled [2]. Currently, WDFs are used extensively in virtual analog and physical modeling [3–11] for those same reasons. In 2015, a more general approach has been developed that deals with both multiple/multiport nonlinearities and arbitrary network topologies [12, 13].

The purpose of this paper is to build upon this general approach by incorporating an iterative solver into the WDF structure to solve the system of multiple/multiport nonlinearities. Newton's Method with backtracking is employed because it is a classic and robust zero-finding technique. Common models of the diode and transistor are studied to learn about the convergence properties using Newton's method of the underlying mathematical functions. Lastly, a case study of three simple reference circuits is presented which demonstrates the generality and promise for WDF implementation of circuits with multiple nonlinearities.

Section 2 summarizes the previous work related to WDFs with nonlinearities and reviews the general approach which the iterative method will be incorporated into. In Section 3, the setup of the iterative solver within the general approach is presented and the iterative techniques are introduced. Sections 4–5 present case studies of circuits containing a single diode, pair of diodes and a bipolar junction transistor. Section 6 summarizes the results.

2. PREVIOUS WORK

2.1. Nonlinear WDFs and Iterative Techniques

Many interesting musical devices contain nonlinear circuit elements and topologies which cannot be decomposed into only serial and parallel connections. Early approaches to developing WDF models with nonlinearities focused on reference circuits with certain types of single nonlinearities [14, 15]. Other solutions use domain knowledge about the device or circuit to make simplifying assumptions to realize computability of the WDF structure. These include combining multiple nonlinearities into a single one-port nonlinearity [5, 6, 9] and simplifying multiport nonlinearities into cross-control models [4,7,10]. A comprehensive overview of early nonlinear WDF implementations can be found in [15].

The first example of including a nonlinear circuit element in a WDF [14] involved attaching the nonlinearity to an adapted port at the root of the WDF. This is necessary because the model of the nonlinearity is delay free, and a port reflection back to the nonlinearity would create a delay-free loop. In addition, the nonlinearity studied was modeled with an invertible function in the Kirchhoff domain from which it was possible to map to a corresponding invertible function in the wave domain which was then able to be solved explicitly. In modern WDF structures, child elements in the WDF tree are adapted so that their upward-facing port is reflection free. The root element cannot be adapted so if the reference circuit contains a non-adaptable nonlinear or linear element it is placed at the root of a binary connection tree [4].

In [5], the Lambert $\mathcal{W}$ function is used to solve a single diode equation or approximate the solution to an anti-parallel diode equation. In particular, a lookup table is used to determine an initial guess at the solution which is then refined through iteration on an approximation of the Lambert $\mathcal{W}$ function. The case of multiple diodes is generalized and improved in [6].

In [3], a WDF implementation of a triode tube amplifier using the Cardarilli triode model is introduced. This method involves solving one or two nonlinear equations depending on whether or not grid current is taken into account. In both cases, the authors use the secant method to solve the nonlinear equations and, in the case where grid current is taken into account, multidimensional zero finding is avoided by solving one of the nonlinear equations first and using that result to determine whether or not the second nonlinear equation needs to be iterated on. This model was found to be more computationally efficient than previous attempts [7, 10] and perform more realistically, especially when in saturation.

Another approach to WDFs with multiple nonlinearities is provided in [16] where the passivity property of WDFs is exploited to show that WDFs are contractive systems which are guaranteed to converge to a fixed point under global iteration. Using this, they introduce an iteration time dimension and at each sample iterate along this dimension until the steady-state solution for that sample has been reached for each nonlinearity with all other circuit values held constant. While this approach can be applied to complex topologies that are expressible as non-tree-like arrangements of series and parallel adapters and models with multiple nonlinearities, the convergence of the fixed-point iteration is only linear

and the speed of convergence is highly dependent on the choice of port resistance. Additionally, the contractivity property only holds if the nonlinear elements are also contractive which excludes non-contractive nonlinearities such as transistors.

This approach is expanded upon in [17] where the unrelaxed fixed point iteration scheme is replaced with the secant method. They additionally modify the secant method to control the step in the search direction in a way such that it always moves in the direction of the zero. For multidimensional nonlinearities, two iteration schemes are proposed. In the first, the iteration of all nonlinearities is performed simultaneously in a vector formulation whereas in the second each nonlinearity is iterated on individually.

The first method was found to be faster when it did converge and the second method was found to converge in situations where the first method did not converge. The primary benefits of the methods in these two papers are that they preserve the modularity of the WDF structure whilst the second paper's results improves the convergence of the iterations from linear to superlinear.

Another iterative approach based on a graph theoretic analysis is given in [18]. In this approach, the entire circuit topology is represented with a single scattering parameter matrix and power waves are propagated between the circuit element ports. From this representation a fixed point iteration can then be performed to resolve the delay-free loops introduced by nonlinear elements.

An example of iterative techniques being used in the implementation of circuits containing single nonlinearities as state space filters is given in [19, 20]. This approach involves using the K-method to linearize the nonlinearities into a system of equations which can then be iterated on until convergence is reached with the desired unknown quantity. Newton's method is used to solve single antiparallel diode nonlinearities whereas pretabulated tables are suggested for realization of triodes and bipolar junction transistors (BJTs) in amplifier circuits.

2.2. A General Approach for Multiple/Multiport Nonlinearities with Arbitrary Topologies

We review a general approach to set up a WDF structure with any number of multiport nonlinearities as well as with any general topology [12, 13].

First, the prototype circuit must be analyzed and decomposed into parallel, series and rigidly connected components. This can be accomplished using graph theoretic techniques [12, 21]. For circuits containing multiple/multiport nonlinearities, all nonlinearities are rigidly connected using the replacement graph technique in [21] so that they are kept together as a single entity. The results of this process is an *SPQR* tree where all nonlinear elements are grouped together at the root of the tree in an $\mathcal{R}$-type node.

To deal with the nonlinearities in the $\mathcal{R}$-type node, all of the nonlinearities are pulled out of the $\mathcal{R}$-type node. This results in a system of vector nonlinearities attached to a $\mathcal{R}$-type adapter represented mathematically by the following system:

$$\text{wave nonlinearity} = \{\mathbf{a}_I = F_w(\mathbf{b}_I) \tag{1a}$$

$$\text{scattering} = \begin{cases} \mathbf{b}_I = \mathbf{S}_{11}\mathbf{a}_I + \mathbf{S}_{12}\mathbf{a}_E \\ \mathbf{b}_E = \mathbf{S}_{21}\mathbf{a}_I + \mathbf{S}_{22}\mathbf{a}_E \end{cases} \tag{1b}$$

$$\text{S matrix} = \begin{bmatrix} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \mathbf{S}_{21} & \mathbf{S}_{22} \end{bmatrix}, \tag{1c}$$

where F_w represents the wave domain nonlinear equation(s), $\mathbf{a}_I$ and $\mathbf{b}_I$ represent the vectors of internal incident and reflected waves and $\mathbf{a}_E$ and $\mathbf{b}_E$ represent the external incident and reflected waves. More specifically, all incident and reflected waves are defined in terms of the $\mathcal{R}$-type adapter with external waves propagating from the WDF subtree and internal waves from the nonlinearities.

As described in [13], to calculate the $\mathbf{S}$ matrix, it is first necessary to form an $\mathbf{X}$ matrix. This is done by attaching an instantaneous Thévenin port equivalent to each port of the $\mathcal{R}$-type adapter and then using Modified Nodal Analysis (MNA) to determine the contents of the $\mathbf{X}$ matrix. It follows from the definitions of waves and Thévenin port equivalents that $\mathbf{S}$ is given by the following matrix equation:

$$\mathbf{S} = \mathbf{I} + 2\begin{bmatrix}\mathbf{0} & \mathbf{R}\end{bmatrix}\mathbf{X}^{-1}\begin{bmatrix}\mathbf{0} & \mathbf{I}\end{bmatrix}^T, \tag{2}$$

where $\mathbf{R}$ is a diagonal matrix of port resistances and $\mathbf{I}$ is the identity matrix.

Next, since most nonlinear circuit models are defined in the Kirchhoff domain and F_w may be hard to obtain, it may be easier to work with $\mathbf{i}_C = F_k(\mathbf{v}_C)$. In that case a w–K converter matrix $\mathbf{C}$ to convert incident and reflected waves $\mathbf{a}_I$ and $\mathbf{b}_I$ into voltage and currents $\mathbf{v}_C$ and $\mathbf{i}_C$ is typically used. The $\mathbf{C}$ matrix is given by:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} \\ \mathbf{C}_{21} & \mathbf{C}_{22} \end{bmatrix} = \begin{bmatrix} -\mathbf{R}_I & \mathbf{I} \\ -2\mathbf{R}_I & \mathbf{I} \end{bmatrix}, \tag{3}$$

where $\mathbf{R}_I$ is a diagonal matrix of internal port resistances. This process results in three vector delay-free loops which can then be reduced to one vector delay-free loop by combining the submatrices of $\mathbf{S}$ and $\mathbf{C}$ into matrices $\mathbf{E}$, $\mathbf{F}$, $\mathbf{M}$ and $\mathbf{N}$:

$$\mathbf{E} = \mathbf{C}_{12}(\mathbf{I} + \mathbf{S}_{11}\mathbf{H}\mathbf{C}_{22})\mathbf{S}_{12} \tag{4a}$$

$$\mathbf{F} = \mathbf{C}_{12}\mathbf{S}_{11}\mathbf{H}\mathbf{C}_{21} + \mathbf{C}_{11} \tag{4b}$$

$$\mathbf{M} = \mathbf{S}_{21}\mathbf{H}\mathbf{C}_{22}\mathbf{S}_{12} + \mathbf{S}_{22} \tag{4c}$$

$$\mathbf{N} = \mathbf{S}_{21}\mathbf{H}\mathbf{C}_{21}, \tag{4d}$$

with $\mathbf{H} = (\mathbf{I} - \mathbf{C}_{22}\mathbf{S}_{11})^{-1}$.

This formulation yields a structure in which all nonlinearities in the circuit can be isolated together at the top of the WDF tree above the $\mathcal{R}$-type adapter. This leads to the following system with the nonlinearities represented in the Kirchhoff domain:

$$\begin{cases} \mathbf{i}_C = F_k(\mathbf{v}_C) & \text{(5a)} \\ \mathbf{v}_C = \mathbf{E}\mathbf{a}_E + \mathbf{F}\mathbf{i}_C & \text{(5b)} \\ \mathbf{b}_E = \mathbf{M}\mathbf{a}_E + \mathbf{N}\mathbf{i}_C, & \text{(5c)} \end{cases}$$

where $\mathbf{E}$, $\mathbf{F}$, $\mathbf{M}$ and $\mathbf{N}$ are given in (4). If trying to work directly from (5), dealing with the nonlinearity in the Kirchhoff domain, it must be determined how to evaluate the nonlinearity as it still contains a delay-free loop.

In [12], this delay-free loop is eliminated by means of the K-method. With the K-method, the nonlinearity is solved either using iteration or table lookup with pretabulated values of the nonlinear function.

The general approach proposed in [12, 13] allows any prototype analog circuit to be turned into a computable WDF structure regardless of topology or number of nonlinearities as long as the nonlinearities possess a functional model representation. Use of the K-method, however, implies a shear transformation of the nonlinearity models from the Kirchhoff domain to a domain consisting of pseudo-wave variable $\mathbf{p}$ and current $\mathbf{i}_C$. Thus, if solutions are tabulated in either domain, the table of solutions must be shear transformed to the other domain. Finding the correct value in the

sheared space can require complex search and interpolation methods. Additionally, the storage and computational requirements of multidimensional tables quickly becomes challenging as the number of dimensions increases.

3. ITERATIVE TECHNIQUES

3.1. The General Approach with Nonlinear Solver

An iterative solution to the nonlinearities contained in (5a) is presented as an alternative to storing tables and to introduce generality and the abilityto obtain solutions of a desired numerical precision. The zero-finding formulation of the system in Section 2.2 is obtained by substituting (5a) into (5b):

$$\mathbf{v}_C = \mathbf{E}\mathbf{a}_E + \mathbf{F}F_k(\mathbf{v}_C)\,, \tag{6}$$

and then solving the equation for zero to obtain:

$$H(\mathbf{v}_C) = \mathbf{E}\mathbf{a}_E + \mathbf{F}F_k(\mathbf{v}_C) - \mathbf{v}_C\,. \tag{7}$$

This nonlinear function $H(\mathbf{v}_C)$ will give the values of $\mathbf{v}_C$ and $\mathbf{i}_C$ that solve the instantaneous relationship.

Consequently, using an iterative technique such as Newton's method to find the zero of (7) presents itself as a natural method for resolving the delay-free loop in the general framework presented in Section 2.2.

In using such a technique, having a good initial guess is crucial to the success of the zero-finding algorithm. In the context of (6) and (7), a typical choice would be:

$$\mathbf{v}_C^0(n) = \mathbf{E}\mathbf{a}_E(n-1) + \mathbf{F}F_k(\mathbf{v}_C(n-1))\,, \tag{8}$$

where $\mathbf{v}_C^0(n)$ is the initial guess at the value of $\mathbf{v}_C(n)$. However, considering that $\mathbf{v}_C(n-1)$ has already been propagated down and back up the tree structure, the most current value of $\mathbf{a}_E$ is already known. Thus, another possible initial guess would be:

$$\mathbf{v}_C^0(n) = \mathbf{E}\mathbf{a}_E(n) + \mathbf{F}F_k(\mathbf{v}_C(n-1))\,. \tag{9}$$

In the case studies in Sections 4 and 5 both initial guess choices will tested and investigated.

3.2. Newton's Method

The basis of Newton's method in a single dimension to find the zero x^* of a function f comes from the Taylor series expansion of f about x^* which leads to the recursive series of approximations of x^*:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}\,, \tag{10}$$

for which $x_n \to x^*$ as $n \to \infty$ given that f and the initial guess x_0 satisfy certain assumptions. The multivariate equivalent of Newton's method is:

$$\mathbf{X}^{k+1} = \mathbf{X}^k - J(\mathbf{X}^k)^{-1}F(\mathbf{X}^k)\,, \tag{11}$$

with $\mathbf{X} = (x_1, x_2, \ldots, x_n)^T$, $F = (f_1, f_2, \ldots, f_n)^T$ and where J is the Jacobian matrix of F and the superscript represents the current iterate.

In order for Newton's method to converge quadratically in the univariate case, it is necessary that f be twice continuously differentiable, that x^* is a simple zero of f (meaning that $f'(x^*) \neq 0$ and $f''(x^*) \neq 0$) and the initial guess x_0 is in a close enough neighborhood of the zero [22, p. 85].

The condition for having global convergence is given by f again being twice continuously differentiable as well as being an increasing, convex function that has a zero [22, p. 86]. For a function meeting these assumptions, the zero x^* is unique and can be reached with any initial guess.

In order to have superlinear convergence in the multivariate case, F must be continuously differentiable in a convex open set around a simple zero $\mathbf{X}^*$ as well as having a sufficiently close initial guess $\mathbf{X}^0$. Additionally, if F is Lipschitz continuously differentiable near $\mathbf{X}^*$ then, for sufficiently close $\mathbf{X}^0$, the convergence is quadratic [23, p. 276].

3.3. Newton's Method with Backtracking

Since the convergence of Newton's method depends on the proximity of the initial guess to a zero, attempts have been made to alter Newton's method to improve the convergence. One such algorithm is called Newton's method with backtracking or damped Newton's method.

The main idea behind the univariate version of this algorithm is to keep the linear approximation of the function from overshooting the zero. This is accomplished by performing a backtracking line search on the linear approximation of the function and ensuring that the norm of the function is being reduced at each iteration. Thus, rather than setting $x_{n+1} = x_n - f(x_n)/f'(x_n)$, the new iterate is set to $x_{n+1} = x_n - \alpha f(x_n)/f'(x_n)$ where $\alpha \in (0,\, 1]$. Starting with $\alpha = 1$, α is multiplied by 0.5 – or any value in $(0,\, 1)$ – until the following condition is met:

$$\left|f\left(x_n - \alpha f'(x_n)^{-1}f(x_n)\right)\right| \leq (1-\alpha\mu)|f(x_n)|\,, \tag{12}$$

where $\mu \in (0,\, 1)$. This condition is called the sufficient decrease condition and ensures that the next guess is moving closer to x^*.

In the multivariate case, the sufficient decrease criterion is given by

$$\left\|F\left(\mathbf{X}^K - \alpha^k J(\mathbf{X}^k)^{-1}F(\mathbf{X}^k)\right)\right\| \leq (1-\alpha\mu)\|F(\mathbf{X}^k)\|\,. \tag{13}$$

Newton's method with backtracking does achieve global convergence under certain assumptions about f [24]. Unfortunately, however, there are still a wide range of smooth functions for which global convergence is not guaranteed.

3.4. Improving the Initial Guess

As previously noted, Newton's method and Newton's method with backtracking depend on a good initial guess and being in a close neighborhood of the zero in order to achieve quadratic convergence.

In addition to the initial guess types discussed Section 3.1, an additional enhancement would be to use another more globally robust method to hone in on a better initial guess for Newton's method and then switch methods when the refined guess is sufficiently closer to the region of quadratic convergence. One way to accomplish this is by starting the iterations using a method called Steepest Descent.

The univariate Steepest Descent method [25, pp. 654–659] works by finding a value $\hat{x}$ that minimizes a merit function g. The merit function is chosen in such a way that $g(\hat{x}) = f(x^*) = 0$

Table 1: Diode Clipper Circuit Component Values

Circuit	Component R_1	C_1	C_2
Clipper 1	2.2 kΩ	0.01 µF	N/A
Clipper 2	2.2 kΩ	0.01 µF	0.47 µF

and, thus, is also the solution to the original problem at hand. The following merit function is typically chosen:

$$g(x) = \frac{1}{2} f(x)^2 \,. \tag{14}$$

Successive approximations of $\hat{x}$ are found by moving in the direction of greatest decrease which is $-g'(x)$, the negative of the derivative of g. Backtracking is also employed so that the newest estimate of the minimizer does not overshoot and move away from the minimizer.

In the multivariate case, the merit function is given by

$$G(\mathbf{X}) = \frac{1}{2} F(\mathbf{X})^T F(\mathbf{X}) \,, \tag{15}$$

and the direction of greatest decrease is given by the negative gradient $-\nabla G(\mathbf{X})$.

This method used by itself will eventually converge to the minimum of the merit function (which corresponds to the zero of the original function, if it exists), but the convergence is only linear. Even given that, it still approaches the zero quickly enough that a few iterations can be enough to generate an initial guess for Newton's method which will quickly converge to the zero.

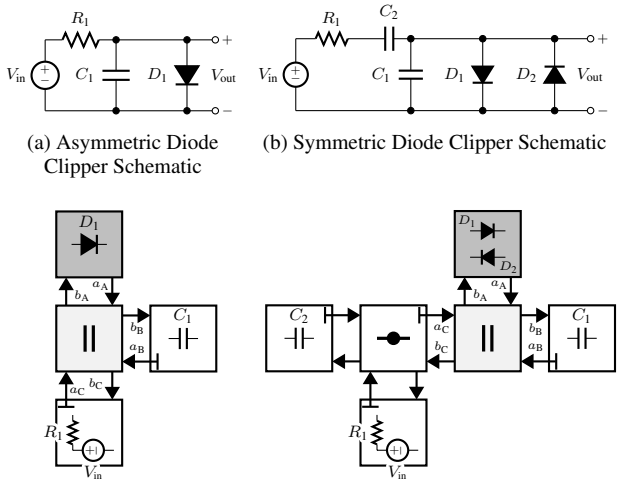
In the implementation of Steepest Descent used in this paper, a maximum number of iterations as well as a tolerance on the size of the norm of the merit function are given. Thus, the algorithm stops if the merit function has been sufficiently minimized or when the maximum number of iterations has been reached.

Other methods exist, such as the Secant method and Broyden's Method, which numerically approximate the derivative and Jacobian. Additionally, quasi-Newton methods exist which only evaluate the Jacobian once and then perform incremental numerical updates of it at each iteration. These methods can reduce the computational complexity of their corresponding algorithms but this may sometimes be at the expense of reduced convergence speed and/or loss of the roundoff error correction typically exhibited by Newton's method [25, ch. 10.3].

In the following case studies, the performance of methods from Sections 3.3 and 3.4 will be evaluated on a circuit containing a single diode, one containing antiparallel diodes and one containing a transistor. Additionally, the mathematical characteristics of the nonlinear models of the diode and transistor will be examined to determine whether any guarantees can be given to their convergence using the proposed iterative methods.

4. DIODE CLIPPER CIRCUITS

The wave-domain solution of the diode has been well-studied in literature [4–6, 8, 9, 12, 16, 17, 26–28] and an explicit solution exists using the Lambert $\mathcal{W}$ function [5, 6] (although the Lambert $\mathcal{W}$ function requires an iterative method to either precalculate for table lookup or solve at runtime). While the solution to the diode can be tabulated in the wave domain, as has been previously done, it will be informative to demonstrate the method of Section 3.1 on



(a) Asymmetric Diode Clipper Schematic

(b) Symmetric Diode Clipper Schematic

(c) Asymmetric Diode Clipper WDF

(d) Symmetric Diode Clipper WDF

Figure 1: Diode Clipper Schematics and WDF Structures
(Dark Grey: Nonlinearities; Light Grey: $\mathcal{R}$-type Adapter)

a circuit containing a single nonlinearity. Additionally, the mathematical properties of Shockley's diode model can be investigated.

An asymmetric diode clipper circuit consists of a resistive voltage source in parallel with a capacitor and a diode (Fig. 1a). We model the diode using the Shockley diode equation:

$$i_D = I_s \left(e^{v_D/\eta V_T} - 1 \right) \,, \tag{16}$$

where $I_s = 2.52 \times 10^{-14}$ represents the reverse bias saturation current, $V_T = 0.02585$ represents the thermal voltage, η is the ideality factor of the diode and v_D is the voltage across the diode.

The scattering behavior of a parallel three-port adapter, which is already known in explicit form [29], immediately gives us the S matrix from the formulation of Section 2.2:

$$\mathbf{S} = \left[\begin{array}{c|cc} \mathbf{S}_{11} & \mathbf{S}_{12} \\ \hline \mathbf{S}_{21} & \mathbf{S}_{22} \end{array}\right] = \left[\begin{array}{c|cc} \delta_A - 1 & \delta_B & \delta_C \\ \hline \delta_A & \delta_B - 1 & \delta_C \\ \delta_A & \delta_B & \delta_C - 1 \end{array}\right] . \tag{17}$$

where

$$\delta_m = \frac{2G_m}{G_A + G_B + G_C} \,. \tag{18}$$

From (3) it follows that the $\mathbf{C}$ matrix is:

$$\mathbf{C} = \begin{bmatrix} -R_A & 1 \\ -2R_A & 1 \end{bmatrix} . \tag{19}$$

The WDF (Fig. 1c) is formed by placing the diode at the unadapted port of the parallel adapter and then forming the system from (5).

From (7), the following equation representing the nonlinearity is determined:

$$\begin{aligned} h(\mathbf{v}_C(n)) &= \mathbf{E}\mathbf{a}_E(n) + \mathbf{F} f_k(\mathbf{v}_C(n)) - \mathbf{v}_C(n) \\ &= \mathbf{E}\mathbf{a}_E(n) + \mathbf{F} \left[I_s \left(e^{\mathbf{v}_C(n)/\eta V_T} - 1 \right) \right] - \mathbf{v}_C(n) \,. \end{aligned} \tag{20}$$

From (20) it follows that

$$h'(\mathbf{v}_C(n)) = \mathbf{F} \frac{I_s}{\eta V_T} e^{\mathbf{v}_C(n)/\eta V_T} - 1 \,, \tag{21}$$

$$h''(\mathbf{v}_C(n)) = \mathbf{F} \frac{I_s}{(\eta V_T)^2} e^{\mathbf{v}_C(n)/\eta V_T} \,, \tag{22}$$

Table 2: Newton's Method with Backtracking Diode Clipper

		Iterations		Backtracking		Output Error	
Circuit	Upsamp	Mean	Peak	Mean	Peak	RMSE	Peak
Clipper 1	$1\times f_s$	3.88	9	1.50	13	0.40	0.88
Clipper 1	$2\times f_s$	3.01	9	0.57	9	0.14	0.47
Clipper 1	$4\times f_s$	2.61	8	0.22	6	0.05	0.25
Clipper 1	$8\times f_s$	2.32	7	0.07	5	0.02	0.05
Clipper 2	$1\times f_s$	5.63	9	1.46	7	0.21	0.60
Clipper 2	$2\times f_s$	4.61	7	0.74	5	0.08	0.26
Clipper 2	$4\times f_s$	3.97	7	0.27	5	0.01	0.04
Clipper 2	$8\times f_s$	3.23	6	0.08	2	0.01	0.04

Input signal: 10 kHz, 4.5 peak amplitude sinusoid at 44.1 kHz sampling rate,

are the equations for the first and second derivatives, respectively. Since I_S, $(\eta V_T)^2$ and $e^{\mathbf{v}_C(n)/\eta V_T}$ are all positive, it is clear that as long as $\mathbf{F} \neq \begin{bmatrix}0\end{bmatrix}$, then the diode model is either strictly convex or strictly concave. In either case, from the results in Section 3.2 it follows that (20) is globally convergent if that condition holds for the $\mathbf{F}$ matrix. In particular, the only way for $\mathbf{F} = \begin{bmatrix}0\end{bmatrix}$ is if $\mathbf{S}_{11} = \begin{bmatrix}-1\end{bmatrix}$ which should never happen in practice. In the case of a parallel adapter, that condition would only be possible if G_A, the inverse of the port resistance R_A, of the diode's port, is equal to zero. The requirement that $R_A > 0$ prevents that from happening. In any arbitrary circuit topology, one should be able to explicitly set G_A to avoid the degenerate condition $\mathbf{F} = \begin{bmatrix}0\end{bmatrix}$.

A symmetric diode clipper circuit (Figs. 1b, 1d) contains two antiparallel diodes which (if identical) can be represented with the following model:

$$i_D = I_S \left(e^{v_D/\eta V_T} - e^{-v_D/\eta V_T} \right) . \tag{23}$$

While the derivative is nonnegative:

$$\frac{\mathrm{d}}{\mathrm{d}v_D} i_D = \frac{I_S}{\eta V_T} \left(e^{v_D/\eta V_T} + e^{-v_D/\eta V_T} \right) > 0 \tag{24}$$

$$\iff e^{v_D/\eta V_T} > -e^{-v_D/\eta V_T} , \tag{25}$$

from the equation for the second derivative:

$$\frac{\mathrm{d}^2}{\mathrm{d}v_D^2} i_D = \frac{I_S}{(\eta V_T)^2} \left(e^{v_D/\eta V_T} - e^{-v_D/\eta V_T} \right) , \tag{26}$$

it is easy to see that the second derivative takes on all values in $\mathbb{R}$ and so the function is not convex.

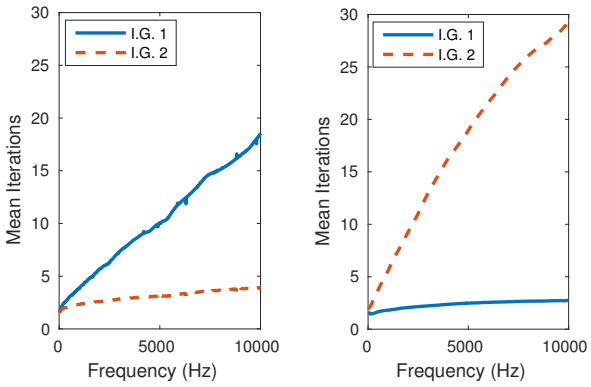
In the general setting of a circuit containing M_{fwd} parallel diodes and M_{rev} antiparallel diodes [6], the function for the nonlinearity is:

$$i_D = I_S [e^{v_D/M_{\text{fwd}}\eta V_T} - e^{-v_D/M_{\text{rev}}\eta V_T}] , \tag{27}$$

from which the same conclusions can be reached. Therefore, global convergence is not guaranteed for any combination of parallel and antiparallel diodes.

Numerical simulations were run for WDF implementations of both diode clipper circuits. The numerical values used in the simulations are given in Table 1 where Clipper 1 refers to the asymmetric diode clipper and Clipper 2 refers to the symmetric diode clipper. The device values and ideality factor of 1.75 are the same as were used in [9].

For both diode clipper circuits, an LTspice [30] simulation was ran for use as a baseline comparison to the WDF simulations. The



(a) Asymmetric Diode Clipper (b) Common Emitter Amplifier

Figure 2: Initial Guess Type Performance Comparison
(I.G. 1 refers to (9) and I.G. 2 refers to (8))

input voltage was a 10 kHz sinusoid with peak amplitude of 4.5 V which was used to test the performance of the iterative technique in response to a numerically challenging input signal. The LTspice simulations were generated using default LTspice configuration values with a maximum timestep frequency 176.4 kHz which reduces the interpolation error resulting from converting the adaptive timestep of the LTspice output to a fixed timestep.

Both Newton's method with backtracking and the hybrid Steepest Descent–Newton's method with backtracking algorithms were compared. For both algorithms, the tolerance (which is calculated as the L^2 norm of (7)) was set to 1.42×10^{-8} V [1] and maximum Newton iterations and backtracking steps were set to 200 and 50, respectively.

The results of the simulations for Newton's Method with backtracking are given in Table 2 where the error value is calculated as the difference in output voltage between the LTspice simulation and the WDF. RMSE is the root-mean-square error which is given by

$$E_{\text{RMSE}} = \sqrt{\frac{\sum_{n=0}^{N-1} (x_{\text{LT}}(n) - x_{\text{WDF}}(n))^2}{N}}, \tag{28}$$

where x_{LT} is the LTspice output voltage signal and x_{WDF} is the WDF output voltage. The peak error is calculated as the L^∞ norm of the difference in output voltages and is given by

$$E_{\text{PEAK}} = \max_n |x_{\text{LT}}(n) - x_{\text{WDF}}(n)|. \tag{29}$$

In both error formulas (28) and (29), x_{LT} refers to the LTspice output voltage, x_{WDF} refers to the WDF output voltage and N is the length of the signal in samples.

Even with this relatively high voltage and high frequency test signal, acceptable iteration counts are achieved after 4 to 8 times oversampling. The error values are the result of a combination of factors including linear interpolation and the fact that WDFs use the bilinear transform while LTspice does not. Results with lower amplitude and lower frequency test signals yielded extremely fast convergence with mean iterations typically between 1 or 2.

The results of the hybrid Steepest Descent–Newton's method (using the same 10 kHz sinusoid input signal) are given in Table 4.

[1] approximately the square root of machine epsilon in Matlab R2015b

Table 3: Newton's Method Common Emitter Amp Simulation Results

		Iterations		Backtracking		Error	
Freq.	Amp.	Avg	Peak	Avg	Peak	Avg	Peak
10 Hz	0.01V	1.00	2	0.00	0	0.0001	0.0004
10 Hz	0.1V	1.14	2	0.00	0	0.0015	0.0045
10 Hz	1V	1.69	2	0.00	0	0.0168	0.0630
100 Hz	0.01V	1.90	2	0.00	0	0.0025	0.0039
100 Hz	0.1V	1.98	2	0.00	0	0.0273	0.0588
100 Hz	1V	1.59	11	0.01	5	0.0835	1.1007
1 kHz	0.01V	2.00	2	0.00	0	0.0086	0.0137
1 kHz	0.1V	2.56	3	0.00	0	0.0889	0.4006
1 kHz	1V	2.31	30	0.45	105	0.1063	1.6206
10 kHz	0.01V	2.87	3	0.00	0	0.0093	0.0169
10 kHz	0.1V	3.48	5	0.00	0	0.1609	0.6769
10 kHz	1V	N/A	N/A	N/A	N/A	N/A	N/A

The number of maximum Steepest Descent algorithm iterations were set to 2, 4 and 8 to illustrate the impact that Steepest Descent has on reducing the mean number Newton iterations. The amount of backtracking required by the Steepest Descent portion of the algorithm did appear to increase somewhat rapidly with the increase in maximum allowable iterations. There are, however, a number of ways to perform the backtracking line search portion of the algorithm so it may be possible to reduce the amount of backtracking by implementing one of those different methods.

Overall, the hybrid algorithm achieves a significant reduction in the number of Newton's method iterations required, particularly when combined with oversampling and a modest amount of Steepest Descent iterations.

Regarding choice of initial guess, Fig. 2a shows that Newton's method required fewer iterations for the asymmetric diode clipper when using the incident wave value from the previous timestep in the initial(8). This result was observed over a wide range of frequencies for tests being run using half-second long sinusoidal signals with peak amplitude of 4.5 V and was also seen when the same tests were ran on the symmetric diode clipper WDF.

5. MULTIPORT NONLINEARITIES–BIPOLAR JUNCTION TRANSISTOR

In this section, a common emitter amplifier is simulated using the method developed in Section 3.1. The circuit contains an NPN BJT which has base, collector and emitter terminals (Fig. 3a) and can be viewed as a two-port network device with ports BC and BE (Fig. 3b). The BJT's behavior is completely described by the voltages across the two terminals: v_{BE} and v_{BC}, which are the voltage from base to emitter and base to collector, respectively. The nonlinear behavior of the BJT was simulated using the Ebers–Moll model:

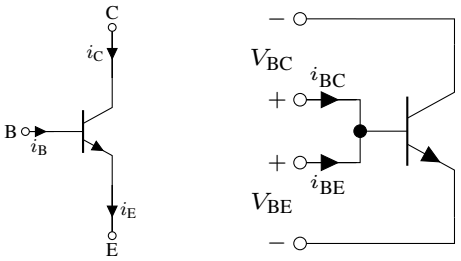
$$i_E = -I_s[e^{v_{\text{BC}}/V_T} - 1] + \frac{I_s}{\alpha_F}[e^{v_{\text{BE}}/V_T} - 1] \tag{30a}$$

$$i_C = -\frac{I_s}{\alpha_R}[e^{v_{\text{BC}}/V_T} - 1] + I_s[e^{v_{\text{BE}}/V_T} - 1] \tag{30b}$$

$$i_B = \frac{I_s}{\beta_R}[e^{v_{\text{BC}}/V_T} - 1] + \frac{I_s}{\beta_F}[e^{v_{\text{BE}}/V_T} - 1]\,, \tag{30c}$$

where

$$\beta_F = \frac{\alpha_F}{1-\alpha_F} \text{ and } \beta_R = \frac{\alpha_R}{1-\alpha_R}\,. \tag{31}$$



(a) BJT Symbol (b) BJT Port Definitions

Figure 3: Circuit element and network two-port of BJT

Since any one of the three equations in (30) can be derived from the other two, only two of them are needed to completely characterize the system. The values i_C and i_E are chosen in order to determine the current at the collector and emitter in terms of the current at the base.

The polarity of the currents given by the Ebers–Moll model (Fig. 3a) are not identical to the polarities of port currents when viewing the transistor as a two-port device (Fig. 3b). Port currents i_{textBC} and i_{BE} are found from the terminal currents i_C and i_E by

$$i_{\text{BC}} = -i_C \quad \text{and} \quad i_{\text{BE}} = i_E\,.$$

Following the method of Section 2.2, the reference circuit (Fig. 4a) was decomposed by isolating the two ports of the transistor above an $\mathcal{R}$-type adapter with series and parallel connections of linear components hanging below it (Fig. 4b).

Once the scattering behavior of the $\mathcal{R}$-type adapter was determined, the nonlinear equations were set up in the form of (7) leading to a complete WDF system; albeit one containing an implicit multidimensional delay-free loop.

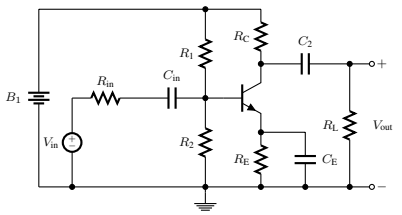
The zero-finding equation F_k for (7) is:

$$F_k(\mathbf{v}_C) = \begin{bmatrix} \frac{I_s}{\alpha_R}\left(e^{v_{\text{BC}}/V_T} - 1\right) - I_s\left(e^{v_{\text{BE}}/V_T} - 1\right) \\ -I_s\left(e^{v_{\text{BC}}/V_T} - 1\right) + \frac{I_s}{\alpha_F}\left(e^{v_{\text{BE}}/V_T} - 1\right) \end{bmatrix}\,, \tag{32}$$

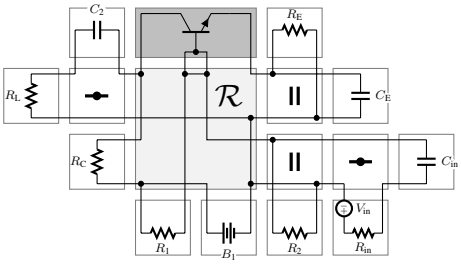
and $\mathbf{v}_C = (v_{\text{BC}},\, v_{\text{BE}})^T$.

The device parameters used for the simulations are given in Table 5 and component values for a 2N2222 transistor were used in the BJT model. Those values are $I_s = 1.0 \times 10^{-14}$, $\beta_F = 200$ and $\beta_R = 3$. The WDF digitization of the common emitter amplifier was analyzed by running a variety of sinusoidal input signals at different frequencies and peak amplitudes with a sampling rate of 44.1 kHz. The results are given in Table 3. Figure 2b shows that using the initial guess given by (9) results in fewer iterations with this circuit for an input sinusoid with peak amplitude of 0.5 V at a variety of frequencies. This is contrast to the result for both diode clipper circuits in which (8) performed better.

Newton's method with backtracking performed very well at frequencies up to 1 kHz with peak amplitudes up to 1 V. The method began to break down at higher frequency and amplitude combinations due to the procedure producing numbers which were unable to be represented by Matlab's double precision data type. Thus, the breakdown was not in Newton's method being unable to converge based on the provided initial guess but in a limitation of the numerics of the system being used for simulation. Using



(a) Common Emitter Amplifier Schematic



(b) Common Emitter Amplifier WDF
(Dark Grey: Nonlinearities; Light Grey: $\mathcal{R}$-type Adapter)

Figure 4: Common Emitter Amplifier Schematic and WDF Structure

Table 4: Hybrid Steepest Descent–Newton's Method Diode Clipper Iteration Results

		Steepest Descent				Newton's Method			
		Iterations		Backtracking		Iterations		Backtracking	
Circuit	Oversampling	Max	Mean	Mean	Peak	Mean	Peak	Mean	Peak
Clipper 1	$1 \times f_s$	2	2.00	2.39	25	2.61	9	1.42	13
Clipper 1	$1 \times f_s$	4	3.07	22.54	89	1.55	8	0.77	11
Clipper 1	$1 \times f_s$	8	3.99	43.65	135	0.35	5	0.00	0
Clipper 1	$8 \times f_s$	2	2.00	7.45	31	1.20	6	0.03	3
Clipper 1	$8 \times f_s$	4	2.60	36.32	95	0.18	3	0.00	0
Clipper 1	$8 \times f_s$	8	2.63	42.09	107	0.00	1	0.00	0
Clipper 2	$1 \times f_s$	2	2.00	5.37	25	3.79	7	0.76	4
Clipper 2	$1 \times f_s$	4	3.73	36.60	91	1.50	5	0.01	1
Clipper 2	$1 \times f_s$	8	4.44	78.80	121	0.01	1	0.00	0
Clipper 2	$8 \times f_s$	2	2.00	11.05	29	2.01	5	0.01	1
Clipper 2	$8 \times f_s$	4	2.92	57.84	81	0.15	2	0.00	0
Clipper 2	$8 \times f_s$	8	2.94	61.60	99	0.03	1	0.00	0

Table 5: Common Emitter Amp Component Values

Component	Value
B_1	18 V
R_{in}	1 kΩ
C_{in}	50 µF
R_1	27.35 kΩ
R_2	2.65 kΩ
R_{E}	220 Ω
C_{E}	100 µF
R_{C}	1.78 kΩ
C_2	10 µF
R_{L}	1 kΩ

Steepest Descent to improve the initial guess provided to Newton's method had no affect on the numerical limitation.

It should also be noted that the common emitter amplifier is designed to purely amplify small amplitude signals and is not designed to clip them. Since asymmetry can be seen in the peaks of a sinusoidal signal with amplitude of 0.2 V, it should be noted that a 1 V peak amplitude test signal would probably not be used in this circuit in practice and was used as a means of investigating the limits of the performance of Newton's method on a WDF containing an Ebers–Moll transistor model. A time domain output comparison of LTspice and the WDF simulation is given in Figure 5.

Common Emitter Amplifier Output

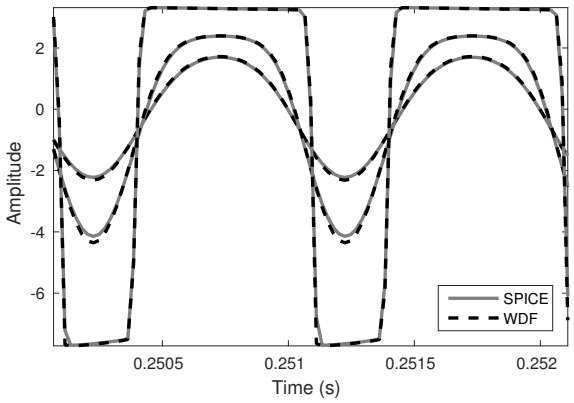


Figure 5: Two cycles of output voltage from 1 kHz input sinusoid

6. CONCLUSION

In this paper an iterative zero-finding technique was incorporated into a generalized WDF approach to digitizing analog reference circuits of arbitrary topology containing multiple/multiport nonlinearities. We elaborated on the work in [12, 13] by introducing two variations of Newton's method that show promise towards the realization of real-time WDFs with multiple nonlinearities. Newton's Method with backtracking was employed in addition to a variant where the Steepest Descent algorithm obtains better initial guesses and increases the speed of convergence.

Two different types of initial guesses were proposed for which (8) resulted in fewer iterations in both the asymmetric and symmetric diode clipper WDFS and (9) resulted in fewer iterations for the common emitter amplifier WDF. For the circuit simulated in [11], tests also indicated fewer iterations using (9) [2]. Both initial guess types should be tested to determine which one performs better for a particular WDF implementation.

We were able to show that the Shockley diode equation meets the requirements for global convergence with Newton's Method and that both algorithms employed in this paper yielded rapid convergence in an asymmetric diode clipper test circuit. Numerical results additionally indicated that a pair of identical antiparallel diodes treated as a singular nonlinearity exhibited good convergence characteristics even when tested with high frequency and amplitude test signals in a symmetric diode clipping circuit.

[2] Private Communication with W. Ross Dunkel, Jun. 10, 2016

Additionally, the numerical performance of the Ebers–Moll model of a BJT was studied via implementation of a common emitter amplifier circuit. Newton's Method with backtracking performs efficiently for signals that fall within and slightly outside the standard operating range of the amplifier circuit.

While the recent work of Schwerdtfeger and Kummert [17] preserves the modularity of the WDF structure, the convergence rate of the methods can be sensitive to the values chosen for the port resistances of the nonlinear elements.

The choice of port resistances for nonlinearities in the proposed approach does not exhibit that same sensitivity. Since the delay-free loop being resolved is restricted to the system of nonlinearities and isolated from the rest of the WDF, impedance mismatching does not occur in the presented approach. The only restriction on the nonlinear port resistances is that they must be chosen such the $\mathbf{F}$ matrix does not get set to zero. There is no inherent restriction on the number of nonlinearities that can be included.

While this paper focused on developing the theory and simple examples illustrating the proposed technique, higher dimensional nonlinearities have already been successfully tested. These include the first clipping stage of the Big Muff Pi distortion pedal [12] and the preamp of the Fender Bassman amplifier [11]. A real-time WDF software library using the presented approach has also been recently developed [31].

The only caveats of the presented method for handling nonlinearities in WDFs are that the properties of the multivariate nonlinear systems must allow them to be solved with iterative techniques and that the iterative techniques are computationally tractable.

7. ACKNOWLEDGMENTS

Michael Jørgen Olsen would like to acknowledge fruitful discussions with both W. Ross Dunkel and Maximilian Rest.

8. REFERENCES

[1] A. Fettweis, "Wave digital filters: Theory and practice," *Proc. IEEE*, vol. 74, no. 2, pp. 270–327, Feb. 1986.

[2] A. Fettweis, "Pseudo-passivity, sensitivity, and stability of wave digital filters," *IEEE Trans. Circuit Theory*, vol. 19, no. 6, pp. 668–673, Nov 1972.

[3] S. D'Angelo, J. Pakarinen, and V. Välimäki, "New family of wave-digital triode models," *IEEE Trans. Audio, Speech, Language Process.*, vol. 21, no. 2, pp. 313–321, Feb. 2013.

[4] G. De Sanctis and A. Sarti, "Virtual analog modeling in the wave-digital domain," *IEEE Trans. Audio, Speech, Language Process.*, vol. 18, no. 4, pp. 715–727, May 2010.

[5] R. C. D. Paiva, S. D'Angelo, J. Pakarinen, and V. Välimäki, "Emulation of operational amplifiers and diodes in audio distortion circuits," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 59, no. 10, pp. 688–692, Oct. 2012.

[6] K. J. Werner, V. Nangia, A. Bernardini, J. O. Smith III, and A. Sarti, "An improved and generalized diode clipper model for wave digital filters," in *Proc. 139 Int. Audio Eng. Soc. (AES)*, New York, NY, Oct. 29 – Nov. 1 2015.

[7] M. Karjalainen and J. Pakarinen, "Wave digital simulation of a vacuum-tube amplifier," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Toulouse, France, May 14–19 2006, vol. 5.

[8] D. T. Yeh, J. S. Abel, and J. O. Smith III, "Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinary differential equations," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-07)*, Bordeaux, France, Sept. 10–15 2007.

[9] D. T. Yeh and J. O. Smith III, "Simulating guitar distortion circuits using wave digital and nonlinear state-space formulation," Espoo, Finland, Sept. 1–4 2008.

[10] J. Pakarinen and M. Karjalainen, "Enhanced wave digital triode model for real-time tube amplifier emulation," *IEEE Trans. Audio, Speech, Language Process.*, vol. 18, no. 4, pp. 738–746, May 2010.

[11] W. R. Dunkel, M. Rest, K. J. Werner, M. J. Olsen, and J. O. Smith III, "The Fender Bassman 5F6-A family of preamplifier circuits—a wave digital filter case study," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, Sept. 5 – 9 2016.

[12] K. J. Werner, V. Nangia, J. O. Smith III, and J. S. Abel, "Resolving wave digital filters with multiple/multiport nonlinearities," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[13] K. J. Werner, J. O. Smith III, and J. S. Abel, "Wave digital filter adaptors for arbitrary topologies and multiport linear elements," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Nov. 30 – Dec. 3 2015.

[14] K. Meerkötter and R. Scholz, "Digital simulation of nonlinear circuits by wave digital filter principles," in *Proc. IEEE Int. Symp. Circuits Syst.*, Portland, OR, May 8–11 1989, vol. 1, pp. 720–723.

[15] A. Sarti and G. De Poli, "Toward nonlinear wave digital filters," *IEEE Trans. Signal Process.*, vol. 47, no. 6, pp. 1654–1668, June 1999.

[16] T. Schwerdtfeger and A. Kummert, "A multidimensional approach to wave digital filters with multiple nonlinearities," in *Proc. European Signal Process. Conf.*, Lisbon, Portugal, Sept. 1–5 2014, vol. 22.

[17] T. Schwerdtfeger and A. Kummert, "Newton's method for modularity-preserving multidimensional wave digital filters," in *IEEE 9th Int. Workshop Multidimensional (nD) Syst. (nDS)*, Vila Real, Portugal, Sept. 7–9 2015.

[18] C. Christoffersen, *Transient Analysis of Nonlinear Circuits Based on Waves*, pp. 159–166, Springer, Berlin, Germany, 2010.

[19] D. T. Yeh, J. S. Abel, and J. O Smith III, "Automated physical modeling of nonlinear audio circuits for real-time audio effects—part I: Theoretical development," *IEEE Trans. Audio, Speech, Language Process.*, vol. 18, no. 4, pp. 728–737, May 2010.

[20] D. T. Yeh, "Automated physical modeling of nonlinear audio circuits for real-time audio effects—part II: BJT and vacuum tube examples," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 4, pp. 1207–1216, May 2012.

[21] D. Fränken, J. Ochs, and K. Ochs, "Generation of wave digital structures for networks containing multiport elements," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 3, pp. 586–596, Mar. 2005.

[22] D. Kincaid and W. Cheney, *Numerical Analysis*, Mathematics of Scientific Computing. American Mathematical Society, Providence, RI, 3rd edition, 2002.

[23] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, New York, NY, 2nd edition, 2006.

[24] O. P. Burdakov, "Some globally convergent modifications of Newton's method for solving systems of nonlinear equations," *Dokl. Akad. Nauk SSSR*, vol. 254, no. 3, pp. 521–523, 1980.

[25] R. L. Burden and D. J. Faires, *Numerical Analysis*, Brooks/Cole, Cengage Learning, Boston, MA, 9th edition, 2010.

[26] A. Bernardini, K. J. Werner, A. Sarti, and J. O. Smith III, "Multi-port nonlinearities in wave digital structures," in *Proc. IEEE Int. Symp. Signals, Circuits, Syst. (ISSCS)*, Iaşi, Romania, July 9–10 2015.

[27] A. Bernardini, K. J. Werner, A. Sarti, and J. O. Smith III, "Modeling a class of multi-port nonlinearities in wave digital structures," in *Proc. European Signal Process. Conf. (EUSIPCO)*, Nice, Italy, Aug. 31 – Sept. 4 2015, pp. 664–668.

[28] K. J. Werner, V. Nangia, J. O. Smith III, and J. S. Abel, "A general and explicit formulation for wave digital filters with multiple/multiport nonlinearities and complicated topologies," in *Proc. IEEE Workshop Appl. Sig. Process. Audio Acoust.*, New Paltz, NY, Oct. 18–21 2015.

[29] A. Sarti and G. De Sanctis, "Systematic methods for the implementation of nonlinear wave-digital structures," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 2, pp. 460–472, Feb. 2009.

[30] A. Vladimirescu, *The SPICE Book*, John Wiley & Sons, NY, 1994.

[31] M. Rest, W. R. Dunkel, K. J. Werner, and J. O. Smith III, "RT-WDF—A modular wave digital filter library with support for arbitrary topologies and multiple nonlinearities," in *Proc. Int. Conf. on Digital Audio Effects*, Brno, Czech Republic, Sept. 5 – 9 2016.

RT-WDF—A MODULAR WAVE DIGITAL FILTER LIBRARY WITH SUPPORT FOR ARBITRARY TOPOLOGIES AND MULTIPLE NONLINEARITIES

Maximilian Rest [†*], *W. Ross Dunkel* [*], *Kurt James Werner* [*], *Julius O. Smith* [*]

[*]Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, California, USA
[†]Fakultät Elektrotechnik und Informatik, Technische Universität Berlin, Berlin, Germany
m.rest@e-rm.de, [chigi22,kwerner,jos]@ccrma.stanford.edu

ABSTRACT

Wave Digital Filters (WDF) [1] are a popular approach for virtual analog modeling [2]. They provide a computationally efficient way to simulate lumped physical systems with well-studied numerical properties. Recent work by Werner et al. [3, 4] enables the use of WDFs to model systems with complicated topologies and multiple/multiport nonlinearities, to a degree not previously known.

We present an efficient, portable, modular, and open-source *C++* library for real time Wave Digital Filter modeling: *RT-WDF* [5]. The library allows a WDF to be specified in an object-oriented tree with the same structure as a WDF tree and implements the most recent advances in the field. We give an architectural overview and introduce the main concepts of operation on three separate case studies: a switchable attenuator, the Bassman tone stack, and a common-cathode triode amplifier. It is further shown how to expand the existent set of non-linear models to encourage custom extensions.

Index Terms— wave digital filter, software, real time, virtual analog modeling, multiple nonlinearities

1. INTRODUCTION

There are numerous methods for virtual analog modeling of analog audio circuits on a digital system. While some of them operate in the Kirchhoff i–v domain with (non)-linear state space models [6, 7], the framework presented in this paper operates in the *wave* domain.

Though historically developed for the design of digital implementations of analog ladder/lattice filters, Wave Digital Filters (WDF) [1] have in recent years become a popular approach to virtual analog circuit modeling [2]. WDFs benefit from well-studied numerical properties and stability conditions. They have been used to successfully model lumped systems, including mechanical systems, electromechanical systems, and especially electronic circuits.

Among other benefits, they are attractive to algorithm developers due to their modularity, and desirable numerical behavior [1]. The efficiency of WDFs make real time simulation a possibility.

In this paper, we present the modular Real Time Wave Digital Filter *C++* software library: *RT-WDF* [5]. This library allows for more computationally efficient WDF simulation than existing frameworks and, most importantly, incorporates the field's recent theoretical advances. The goal of this paper is not to exhaustively document every feature of the library, but to introduce its main principles of operation and demonstrate its application to representative circuits. Full documentation accompanies the sourcecode (see Section 8).

The rest of the paper is structured as follows: Section 2 reviews recent theoretical advances and existing circuit simulation software packages. Section 3 gives an overview of the *RT-WDF* library. Section 4 details the use of the library to simulate representative circuits: a switchable attenuator, the Fender Bassman tone stack, and a common cathode triode amplifier. Section 5 compares the performance of the *RT-WDF* library with *SPICE* [8] and *Matlab* [9], Section 6 concludes and presents an outlook on future work.

2. PREVIOUS WORK

2.1. Recent Theoretical Advances

Recent work by Werner et al. [4] vastly expanded the class of circuits that could be systematically modeled with WDF to include those with complex topologies as well as multiport linear elements [3]. This approach has been successfully applied to model op-amps at various degrees of complexity [10] and has recently been extended to accommodate multiple non-adaptable linear elements [11].

These topological advances also yielded a general method for handling circuits with multiple nonlinearities [4] with WDFs, previously restricted to a special case. In this formulation, the nonlinearities are solved via table lookup [4] or iteration. Properties of Newton-based iterative approaches are studied in [12] and applied to a complex preamplifier circuit involving four nonlinear triode tubes in [13].

One of the main motivations for the creation of the *RT-WDF* library was to provide a reference implementation of these theoretical advances, which are not represented in existing software packages.

2.2. Existing Software Packages

Apart from generic signal processing environments like *Matlab*, platforms for systematically implementing real time virtual analog and physical modelling algorithms in the wave domain have existed for more than a decade now. A review of some of the packages mentioned here can be found in [14].

Even though it was not specifically designed for WDFs, *BlockCompiler* [15] was one of the first environments which was used for their implementation [16].

An approach more specifically tailored to WDFs was a program called *BCT* with its own GUI to graphically arrange and configure circuit elements in binary connection trees [17].

Both software packages are reviewed together with case studies in [16]. One advantage of *BlockCompiler* is its ability to generate optimized C-code of algorithms whereas an advantage of *BCT*

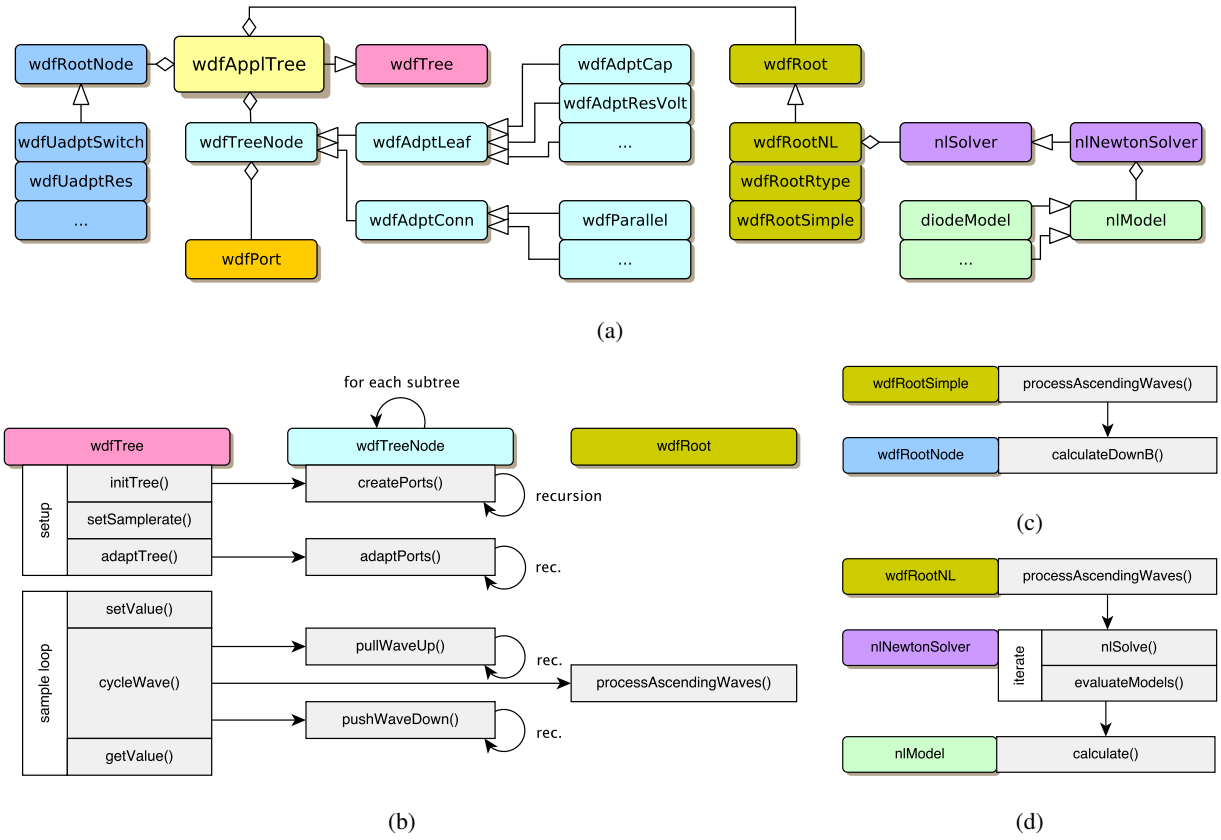


Figure 1: *General RT-WDF framework overview: (a) involved classes and their dependencies, (b) high level functions involved to initialize and run a WDF structure, (c) call graph of a simple WDF root implementation, (d) call graph of a nonlinear root with iterative solver.*

is its intuitive and user-friendly approach. None of them is fully modular in terms of their supported elements, portable to different computer architectures or supports arbitrary topologies and multiple/multiport nonlinearities.

The first public WDF programming library was published in [18] and features a modular, object-oriented class hierarchy written in *Matlab*. It acted as a blueprint for the release of a first public *C++* implementation in the audio developer community around the *JUCE* framework [19]. This library was later refined and published by the same author as *WDF++* [20]. It features architectural modularity and portable code but necessarily could only reflect the state of the art at the time of its development.

Other virtual analog modeling approaches operate in the Kirchhoff i–v domain. *LiveSPICE* [21] builds up on nonlinear state-space models [22] and has support for common nonlinear multiport electrical elements. Another program, *Signaldust 'Salt'* [23], is only available as a closed source preview within the audio DSP developer community and was never fully published.

3. FRAMEWORK OVERVIEW

The framework presented in this paper comes in the form of a publicly available library written in *C++*. This ensures compatibility with popular frameworks for audio applications like *JUCE* [24] and audio plugin APIs such as *LV2*, *VST*, *AU* and *AAX*. It also allows for a structure of hierarchical classes that clearly reflect WDF tree topologies and thus also serves an educational purpose to access the field. Linear algebra functionality is supplied by the third-party *Armadillo* library [25] and all example circuits (see Section 8) are implemented in a standalone host written in *JUCE*.

An overview of the class structure is given in Figure 1a. Within *RT-WDF*, the two main elements in all WDF structures are `wdfTreeNode` and `wdfRoot`. Every WDF *tree* consists of one or more *subtrees* that are formed by *adaptors* and *leafs* as extensions of `wdfTreeNode`. These parallel ($\mathcal{P}$), series ($\mathcal{S}$) and rigid ($\mathcal{R}$) adaptors scatter the waves correctly between the root and the leafs. The leafs represent linear electrical components such as resistors, reactances and non-ideal sources which are the endpoints of each *branch*. The other end of such a subtree is connected to a `wdfRoot` embodying the unadapted circuit components. If these unadapted elements have a closed-form wave domain description they can be implemented as a `wdfRootNode`. Nonlinear elements without closed-form representations are implemented as an `nlModel` which is solved iteratively using an `nlSolver`.

A particular WDF implementation of a circuit with all its adaptors, components and root elements is contained in a user-implemented extension of the `wdfTree` class, the application specific `wdfApplTree`. This class contains all elements and circuit values and provides the API necessary to operate the WDF towards the host application. The required sequence and call graph of these standard methods can be seen in Figure 1b. All functions are initi-

ated from the host by calling methods on such a `wdfApplTree` object as shown in Listing 1.

These methods clearly divide into setup and processing tasks. The first step after instantiation of an `wdfApplTree` is initialization, accomplished by calling `initTree()`. This method itself calls the recursive `createPorts()` function on the entry nodes of the subtrees extending from the root node. This assigns each tree node its up- and down-facing `wdfPort` object which keeps track of wave values and port resistances. Setting the sample rate is necessary for the next step, `adaptPorts()`, as the adaptation of reactive elements depends on the sample rate $f_s = 1/T$.

```
1  //create wdf
2  wdfApplTree myWdfTree( );
3
4  //set up wdf
5  myWdfTree->initTree( );
6  myWdfTree->setSamplerate( Fs );
7  myWdfTree->adaptTree( );
8
9  //process samples
10 for ( int n=0; n<numSamples; n++ ){
11   myWdfTree->setInValue( inSample[n] );
12   myWdfTree->cycleWave( );
13   double outSample[n] = myWdfTree->getOutValue( );
14 }
```

Listing 1: *High level usage of a user-implemented WDF structure from a host.*

Listing 2 illustrates the pattern of recursive function calls that traverse a subtree from the root to the leafs by considering the example of the `adaptPorts()` function.

The adaptation is carried out by first traversing down to the leafs, calculating their up-facing port resistances and then successively passing them on to the parent nodes while keeping also these parent nodes always adapted towards the root. Similar recursive schemes are implemented for example in `pullWaveUp()` and `pushWaveDown()` too.

```
1  double wdfTreeNode::adaptPorts( double T ){
2    for ( wdfPort* dport : downPorts ){
3      dport->Rp = dport->connectedNode->adaptPorts( T );
4    }
5
6    upPort->Rp = calculateUpRes( T );
7    return upPort->Rp;
8  }
```

Listing 2: *Recursive adaptation of the tree.*

After initialization, processing of each audio sample in the WDF is initiated by three function calls: `setInValue()`, `cycleWave()` and `getOutValue()`. The first and the latter have to be overwritten by the user in `wdfApplTree` to correctly assign the input value to the desired source component and collect the output value correctly. Cycling the wave is readily implemented in the `wdfTree` base class as shown in Listing 3.

This listing illustrates the concept of subtrees that hang off the root and again utilizes recursive methods to push and pull wave components to and from all of the leafs of the tree. The tree nodes which are connected to the root are handled as *subtree entry nodes* and act as the starting point of recursive traversals.

Between pulling and pushing, ascending wave components are processed in the root as specified in `wdfApplTree` and the result is returned as descending waves. The different root configurations are explained in detail with examples in Sections 4.1–4.3. The object and method dependencies of a root with a single unadapted one-port (`wdfRootSimple`) and one with multiple nonlinearities (`wdfRootNL`) are shown in Figure 1c/1d respectively. Of course it is worth noting that all these function calls and their dependencies are hidden from the host application and the user-implemented application tree by using a strong hierarchical approach and exposing the internal behaviour of the library only via a few generic high level functions and constructors.

```
1  void wdfTree::cycleWave( ){
2    int treeNo = 0;
3    for ( wdfTreeNode* subtree : subtreeEntryNodes ){
4      (*ascWaves)[treeNo++] = subtree->pullWaveUp( );
5    }
6
7    root->processAscendingWaves( ascWaves, descWaves );
8
9    treeNo = 0;
10   for ( wdfTreeNode* subtree : subtreeEntryNodes ){
11     subtree->pushWaveDown( (*descWaves)[treeNo++] );
12   }
13 }
```

Listing 3: *WDF cycle wave function.*

4. EXAMPLES

This section contains three example circuits that have been modeled using *RT-WDF* to create real time audio algorithms. The host application is based on *JUCE*, which readily provides all audio input and output functionality in a callback function by default and allows the simple creation of graphical user interfaces.

The examples are chosen to highlight several modular concepts of the framework and introduce the three available root types in detail.

4.1. Switchable Attenuator

The first example illustrates the usage of the `wdfRootSimple` object, which supports a single unadapted one-port element at the root. The circuit under examination is a switchable attenuator (Figure 2a) that consists of a voltage source V_{in}, a resistive voltage divider formed by R_1 & R_2 and a switch SW_1 to short the upper resistor R_1. This circuit could obviously also be modeled with less sophisticated approaches than WDF but we chose it here to introduce the library's main concepts on a simple example.

To turn this circuit into its WDF representation, all independent nodes and elements are first labeled with a letter and a digit respectively. These nodes, elements and their interconnections are transformed into a graph representing the circuit (Figure 2b). The graph separation techniques of [26] are applied, after which the graph is transformed into an SPQR tree (Figure 2c). This tree directly yields the WDF representation of the circuit. It consists of adaptors $\mathcal{P}_1$ and $\mathcal{S}_1$ that were introduced by the replacement graphs and the circuit elements SW_1, V_{in}, R_1 and R_2 (Figure 2d). Listing 4 shows the extensions of the `wdfTree` class necessary to model this particular circuit in *RT-WDF*. Such a class always begins with the declaration of pointers for all adapted tree nodes involved, in this case for the resistors R_1, R_2, voltage source V_{in}[1] and adaptors $\mathcal{S}_1$ and $\mathcal{P}_1$. The switch SW_1 is a non-adaptable but linear

[1]Please note that an arbitrary 1Ω resistor in series with the voltage source was necessarily introduced to yield a non-ideal, adaptable voltage source that can serve as a leaf component of the WDF tree. It would alternatively be possible to combine R_2 and the voltage source V_{in} into a non-ideal source and omit $\mathcal{S}_1$.

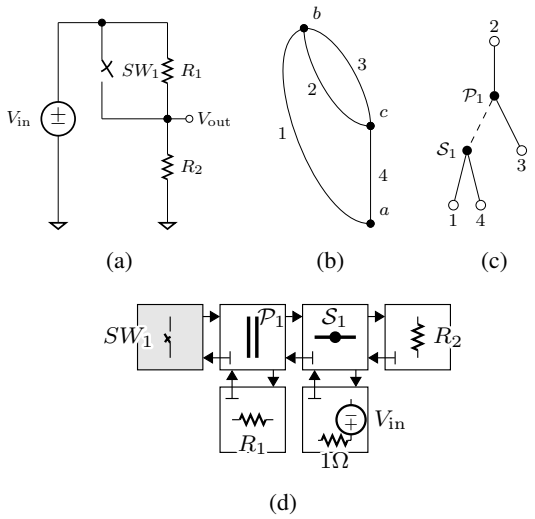


Figure 2: *Deriving a WDF adaptor structure for the switchable attenuator: (a) circuit, (b) graph, (c) SPQR tree, (d) WDF adaptor structure.*

WDF element, which has a closed-form reflection coefficient. It is thus treated as a non-adaptable `rootNode` in this framework and can be implemented using a `wdfRootSimple`

```
class wdfAttenTree : public wdfTree
{
private:
    wdfAdaptedRes*          R1;
    wdfAdaptedRes*          R2;
    wdfAdaptedResVSource*   Vres;
    wdfAdaptedSeries*       S1;
    wdfAdaptedParallel*     P1;
    wdfUnadaptedSwitch*     SW1;
public:
  wdfAttenTree( ){
    //treeNodes
    Vres = new wdfAdaptedResVSource( 0,1 );
    R1 = new wdfAdaptedRes( 250e3 );
    R2 = new wdfAdaptedRes( 250e3 );
    S1   = new wdfAdaptedSeries( Vres, R2 );
    P1   = new wdfAdaptedParallel( S1, R1 );
    //rootNodes
    SW1  = new wdfUnadaptedSwitch( 0 );

    subtreeEntryNodes.push_back( P1 );
    root = new wdfRootSimple( SW1 );
  }
  void setInValue( double voltageIn ){
    Vres->Vs = voltageIn;
  }
  double getOutValue( ){
    return Res2->upPort->getPortVoltage( );
  }
  void setParams( std::vector<double> params ){
    SW1->setSwitch( (int)params[0] );
  }
};
```

Listing 4: *Switchable attenuator tree.*

type root. A pointer for the switch is declared as a private member of the class.

The constructor of the class begins with the creation of the tree and root nodes. Adapted elements are created and initialized according to their physical parameters and the unadapted switch is initialized to be 'open'. The next step is conceptually important: the pointer to the single subtree entry node needs to be stored in a `wdfTree` base class member, `subtreeEntryNodes`. It is used to initiate recursive calls that traverse the subtrees as described in Section 3 and Listing 3. The pointer to the root node `SW1` is handed over to the root's constructor to register it as the root element. The pointer to this root is stored in another member of the base class, the `root` pointer.

The initialization of the WDF elements is followed by the required definitions of the functions `setInValue()` and `getOutValue()`, virtual methods of the `wdfTree` base class. They are used to set and get the input and output samples. Input samples can usually be directly set as voltages or currents of sources. Output samples are retrieved as (a combination of) port voltages or currents for which the port object holds a `getPortVoltage()` and `getPortCurrent()` function. In this case the voltage from the up-facing port of `Res2` is collected, which is the voltage across resistor R_2. The last method demonstrates the ability to further extend the base class to manipulate individual circuit elements: `setParams()` implements the switching functionality of our circuit. It can be called at runtime between samples to effectively configure the reflection coefficient of the root element on the fly. This functionality is necessary to model the circuit in Figure 2a in *RT-WDF*. The resulting `wdfAttenTree` class can now be operated as shown in Listing 1 as a real time algorithm.

4.2. Bassman Tone Stack

The second example makes use of the `wdfRootRtype` class, which allows multiport adaptors with arbitrary topologies in the form of an $\mathcal{R}$-type adaptor at the root. The Fender Bassman tone stack circuit is taken as an example as it is well studied [27] and has a rigid topology that has only recently been supported in WDFs [3]. The circuit's schematic as well as graph-, tree- and WDF-representations are shown in Figure 3. The same process as in the first example is carried out to transform the circuit into an SPQR tree[2]. The additional step here is to capture the rigid connections between the subtree ports of the $\mathcal{R}$-type adaptor in a scattering matrix $\mathbf{S}$ using instantaneous Thévenin port equivalents [3] and modified nodal analysis (MNA) [28].

Listing 5 shows an excerpt of the implementation of the circuit. The setup of the tree nodes is similar to the previous example and not repeated here. Three extra steps must be carried out to set up the $\mathcal{R}$-type root: in contrast to the former example, this time six subtrees need to be registered with their respective entry nodes. These are pointers to the six elements that are directly connected to the $\mathcal{R}$-type adapter at the root, namely `Vin_R3m`, `S2`, `S3`, `C2`, `R4` and `C3`. Secondly, the root object must be created with the number of subtrees as a parameter. This ensures sufficient memory allocation for the scattering matrix $\mathbf{S}$ within the root. In the last step the `setRootMatrData()` function is overwritten. This is an empty function in the `wdfTree` base class and needs to be implemented for specific root class types, including the one used here. Within this function, a `matData` struct is configured to hold the correct values for all required matrices in the root. `setRootMatrData()` is called by `adaptTree()` if the current root requires it. For the $\mathcal{R}$-type root, the `Smat` member of this

[2]The $\mathcal{R}$-type adaptor is chosen here as the root of the tree. It could also reside further down as a tree node adaptor, but this requires an inherent adaptation rule for the up-facing port that depends on the topology and the down-facing port resistances. See [3] for an example.

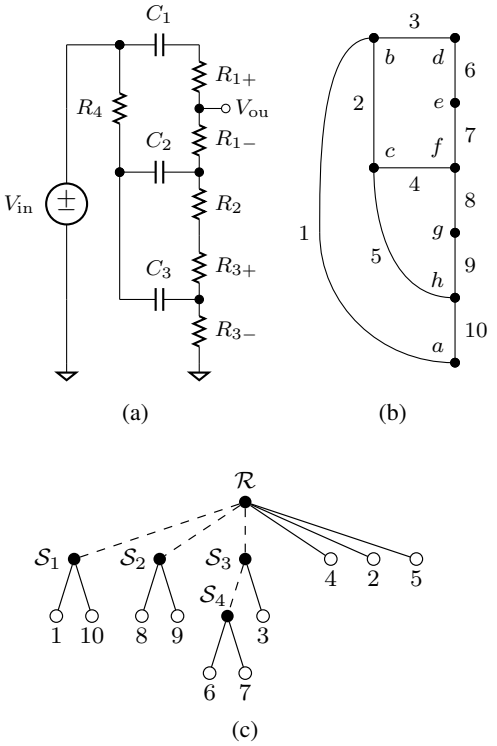


Figure 3: *Deriving a WDF adaptor structure for the Fender Bassman tone stack: a) circuit, b) graph, c) SPQR tree, d) WDF adaptor structure. Modified from [3]*

struct needs to be correctly initialized to embody the scattering behaviour. Dynamic coefficients that depend on the $\mathcal{R}$-type adapter port resistances are supported and enable the user to vary component values in the subtrees (thus affecting the circuit's behaviour) in real time during operation. This functionality is utilized in the code resources of this example.

At the end of the listing, the composition of the output voltage is shown in detail again to demonstrate the flexibility to collect several voltages across circuit elements and adaptors.

```
class wdfTonestackTree : public wdfTree
{
private:
  // pointers for tree elements
  ...
public:
  wdfTonestackTree( ){
    // create tree elements
    ...
    // collect subtree entry points
    subtreeEntryNodes.push_back( Vin_R3m );
    subtreeEntryNodes.push_back( S2 );
    subtreeEntryNodes.push_back( S3 );
    subtreeEntryNodes.push_back( C2 );
    subtreeEntryNodes.push_back( R4 );
    subtreeEntryNodes.push_back( C3 );
    // create new root
    root = new wdfRootRtype( numSubtrees );
  }
  int setRootMatrData( matData* rootMats,
                       double* Rp[] ){

    // populate rootMats->Smat according to
    // R-type scattering behaviour and subtree
    // port resistances Rp[]
    ...
  }
  double getOutValue( ){
    return R1m->upPort->getPortVoltage( ) +
           S2->upPort->getPortVoltage( ) +
           R3m->upPort->getPortVoltage( );
  }
  ...
```

Listing 5: *(partial) Bassman Tone Stack tree class with multiple subtrees and root matrix data update function.*

4.3. Common Cathode Triode Amplifier

The final example highlights the ability of the *RT-WDF* library to handle multiple/multiport Kirchhoff-nonlinearities in circuits via the `wdfRootNL` and `nlModel` classes. Here we model the common cathode triode tube amplifier shown in Figure 4a which has been studied for example in [29] as well. The results of recent WDF research [3, 4] to support arbitrary topologies enable us to extend the model from [29] to include the parasitic capacitances C_{gk}, C_{gp} and C_{pk} as well as continuously evaluated triode grid current I_{g}. Deriving the WDF adaptor structure is again accomplished as in the previous two examples, the result of which is shown in Figure 4b. The extension of the `wdfTree` class for this circuit again implements all elements and their topology in the form of tree nodes and registers all subtree entry nodes (not shown).

```
wdfCCTATree( ){
...
root = new wdfRootNL( numSubtrees,
                      {12AX7_DW},
                      NEWTON );
}
int setRootMatrData( matData* rootMats,
                     double* Rp[] ){

  // populate rootMats->{Emat,Fmat,Mmat,Nmat}
  // according to R-type scattering behaviour
  // and subtree port resistances Rp[]
  ...
}
...
```

Listing 6: *(partial) Common Cathode Triode Amplifier tree class with nonlinear root element and appropriate root matrix data update function.*

The root is created as a `wdfRootNL` object with the number of subtrees, a vector to specify the nonlinear models and the desired

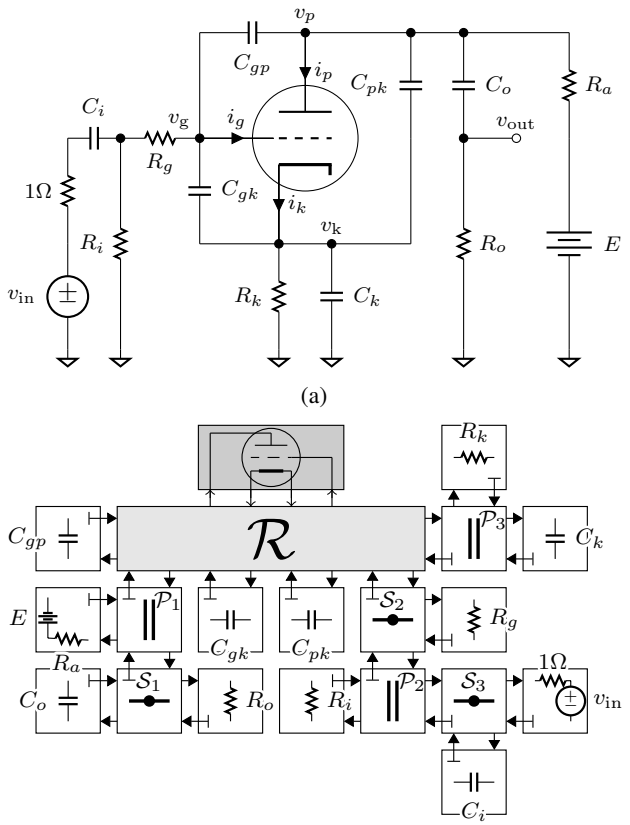


Figure 4: *Common Cathode Triode Amplifier: a) circuit, b) WDF adaptor structure with Kirchhoff nonlinearity. Grey:* `wdfRootNL`*; Dark Grey:* `12AX7DwModel`*.*

solver (Listing 6). In its current state the library supports a multi-dimensional Newton Solver as described in [12] in detail. For this type of root, the `setRootMatrixData()` method must configure the root's system matrices correctly [4]. These matrices implicitly contain a w–K converter to transform the wave variables into the Kirchhoff domain and back. All iterative nonlinear models are currently evaluated in the i–v domain. For this circuit the actual nonlinearity is specified from a user expandable list of models as `12AX7_DW`, a triode model after Dempwolf et al. [30].

$$i_\mathrm{k} = G \cdot \left(\log\Big(1 + \exp\big(C \cdot (\frac{1}{\mu} \cdot v_\mathrm{pk} + v_\mathrm{gk})\big)\Big) \cdot \frac{1}{C} \right)^{\gamma} \tag{1a}$$

$$i_\mathrm{g} = G_\mathrm{g} \cdot \left(\log\Big(1 + \exp\big(C_\mathrm{g} \cdot v_\mathrm{gk}\big)\Big) \cdot \frac{1}{C_\mathrm{g}} \right)^{\xi} + i_{\mathrm{g}0} \tag{1b}$$

$$i_\mathrm{p} = i_\mathrm{k} - i_\mathrm{g} \tag{1c}$$

The model is described by Equations (1a)–(1c) with perveances G, G_g, adaption factors C, C_g and positive exponents γ, ξ. The nonlinear two port model is defined in terms of the port voltages $v_{pk} = v_p - v_k$, $v_{gk} = v_g - v_k$ and port currents i_p, i_g. To be used with the Newton solver in this library, it is desirable that the modeling equations are continuously differentiable with respect to their port voltages in a region around the solution and the Jacobian must be invertible [12].

In general, these nonlinear model objects are managed by the `nlNewtonSolver` as shown in Figure 1a / 1d. They always consist of a `calculate()` function that reflects the physical behavior and a `getNumPorts()` method in the base class for housekeeping.

```
1  12AX7DwModel::12AX7DwModel()
2      : nlModel (2){
3
4  }
5  void 12AX7DwModel::calculate( vec* fNL, mat* JNL,
6                                vec* x,   int* port ){
7
8    double Vpk = x->at( *port);
9    double Vgk = x->at((*port)+1);
10
11   // calculate triode currents & their derivatives
12   // and assign them to the vector / matrix entries
13   ...
14
15   fNL->at(   *port )                  = Ip;
16   JNL->at(  (*port),     (*port) )    = dIp_dVpk;
17   JNL->at(  (*port),    ((*port)+1) ) = dIp_dVgk;
18
19   fNL->at(  (*port)+1 )               = Ig;
20   JNL->at( ((*port)+1),  (*port) )    = dIg_dVpk;
21   JNL->at( ((*port)+1), ((*port)+1) ) = dIg_dVgk;
22
23   (*port) = (*port)+getNumPorts( );
24 }
```

Listing 7: *Implementation of the nonlinear 12AX7 triode model.*

$$\boldsymbol{f}_\mathrm{NL}(\boldsymbol{v}) = \begin{bmatrix} i_\mathrm{p} \\ i_\mathrm{g} \end{bmatrix} \text{ with } \boldsymbol{v} = \begin{bmatrix} v_\mathrm{pk} \\ v_\mathrm{gk} \end{bmatrix} \tag{2}$$

and its Jacobian matrix

$$\mathbf{J}_\mathrm{NL} = \begin{bmatrix} \frac{\partial i_\mathrm{p}}{\partial v_\mathrm{pk}} & \frac{\partial i_\mathrm{g}}{\partial v_\mathrm{pk}} \\ \frac{\partial i_\mathrm{p}}{\partial v_\mathrm{gk}} & \frac{\partial i_\mathrm{g}}{\partial v_\mathrm{gk}} \end{bmatrix}. \tag{3}$$

The Newton Solver iteratively evaluates its specified models for each sample, converging towards a solution of the nonlinear system within a certain tolerance. This solution is then transformed back into the wave domain and returned as descending waves from the root down into the subtrees.

It must be noted that modeling of any nonlinear part in a circuit may introduce drastic aliasing and sufficient oversampling might be necessary to achieve the desired spectral results in the output signal [13]. Also, care must be taken that the selected physical models meet certain criteria in the operating range of the circuit or (fast) convergence of the Newton Solver is not guaranteed [12].

5. PERFORMANCE

All three example circuits from Sections 4.1–4.3 were also implemented in the *SPICE* distribution *LTSpice* and a CCRMA-internal object-oriented Matlab WDF framework. *RT-WDF* and *Matlab* WDF simulations were performed at double precision and a sample rate of $f_s = 44\,100\,\mathrm{Hz}$. *LTSpice* simulations were carried out as a transient analysis with waveform compression disabled. All parameters were left at their default values except `.OPTIONS numdgt=10` to enable internal double precision too. The maximum time step was set to $t_\mathrm{max} = 20\,\mu\mathrm{s} \approx 1/f_s$.

All processing times were captured on a laptop computer from 2013 with Intel i7 2,4 GHz CPU (4 cores) and 8GB RAM running OS X 10.11.4. The *RT-WDF* binaries were built with Apple

quantity	input	RT-WDF	SPICE	Matlab	
duration	8.717 s	0.042 s	15.498 s	252.120 s	
norm. dur.	1	0.005	1.778	28.923	(1)
ratio	–	1	369	6003	
duration	8.717 s	0.149 s	31.172 s	670.107 s	
norm. dur.	1	0.017	3.576	76.874	(2)
ratio	–	1	209	4522	
duration	2.961 s	0.272 s	16.411 s	324.652 s	
norm. dur.	1	0.092	5.543	109.643	(3a)
ratio	–	1	60	1192	
duration	2.961 s	1.035 s	50.014 s	1279.034 s	
norm. dur.	1	0.350	16.893	431.960	(3b)
ratio	–	1	48	1235	

(1) switchable attenuator (2) fender tone stack
(3a) triode amplifier (3b) triode amplifier @ $4\times f_s$

Table 1: *Comparison of processing times of WDF and SPICE circuit simulations for all case studies.*

LLVM 7.1 at optimization level `-O3` and ran with a single processing thread without any other considerable applications in the background.

The results of the benchmarking are shown in Table 1. The first row of each simulation holds the absolute processing times in seconds for all three approaches. For *RT-WDF*, the value describes the subsumed processing times of a block-wise operation, for *SPICE*, the value is taken from the logfile. In *Matlab*, the time to finish the sample processing loop is measured with `tic` and `toc`. The second row shows the normalized times with respect to the length of the input signal. This can be seen as an estimate of CPU load for a real-time operation, as it describes the relative amount of time needed by the simulation to process a certain amount of input samples. A normalized duration > 1 could not catch up with the input signal at full CPU load and the algorithm is thus not real-time capable. The last row correlates the performance of all three approaches in terms of "$\times$ times slower than" with respect to the least demanding candidate.

It is clear that for all case studies the *RT-WDF* simulation is by far the most performant approach to model this circuit as a modular, physically informed algorithm. As indicated by the normalized processing times, all of them could run in real-time applications, even with $\times 4$ oversampling enabled for the common cathode triode amplifier[3].

6. CONCLUSIONS

We presented the modular WDF *C++* library *RT-WDF* [5] which implements recent research advances in the field. It provides great opportunities for both researchers and audio algorithm developers to approach WDFs for analog modeling of lumped systems.

Due to its custom tailored codebase it greatly decreases computational demands compared to other implementations and is portable to many hardware platforms. Many classic WDF elements (such as resistors, capacitors, parallel and series adapters, etc.) are already implemented in the library and future extensions can be easily added due to its strictly modular, hierarchical approach.

The authors encourage and highly appreciate contributions to the codebase to keep up with current research and further improve the performance of the library.

[3]For the *SPICE* comparison of the $\times 4$ simulation, the maximum time step was set to $t_{\max} = 5\,\mu\text{s} \approx 1/(4 \times f_s)$.

7. ACKNOWLEDGMENTS

We would like to thank Michael Jørgen Olsen for his investigation of the application of Newton's Method to WDFs. Maximilian Rest likes to thank Christoph Hohnerlein for inspiring discussions and CCRMA for supporting the development of *RT-WDF*.

8. RESOURCES

The GNU GPL licensed version of the *RT-WDF* library as well as a reference documentation and examples can be found on GitHub at

`www.github.com/m-rest/rt-wdf`

9. REFERENCES

[1] Alfred Fettweis, "Wave digital filters: Theory and practice," *Proc. of the IEEE*, vol. 74, no. 2, pp. 270–327, 1986.

[2] Giovanni De Sanctis and Augusto Sarti, "Virtual analog modeling in the wave-digital domain," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 18, no. 4, pp. 715–727, 2010.

[3] Kurt J. Werner, Julius O. Smith III, and Jonathan S. Abel, "Wave digital filter adaptors for arbitrary topologies and multiport linear elements," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-15)*, Trondheim, NO, Nov. 30 – Dec. 3 2015.

[4] Kurt J. Werner, Vaibhav Nangia, Julius O Smith III, and Jonathan S. Abel, "Resolving wave digital filters with multiple/multiport nonlinearities," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-15)*, Trondheim, NO, Nov. 30 – Dec. 3 2015.

[5] Maximilian Rest, W. Ross Dunkel, and Kurt J. Werner, "RT-WDF–a modular wave digital filter library," Available at www.github.com/m-rest/rt-wdf, 2016.

[6] David T Yeh, Jonathan S Abel, and Julius O Smith III, "Automated physical modeling of nonlinear audio circuits for real-time audio effects—part i: theoretical development," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 728–737, 2010.

[7] Kristjan Dempwolf, Martin Holters, and Udo Zölzer, "Discretization of parametric analog circuits for real-time simulations," in *Proc. of the International Conference on Digital Audio Effects (DAFx-10)*, Graz, AU, Sep. 6 – Sep. 10 2010.

[8] Laurence William Nagel and Donald O. Pederson, *SPICE: Simulation program with integrated circuit emphasis*, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, 1973.

[9] MATLAB, *version 7.10.0 (R2010a)*, The MathWorks Inc., Natick, Massachusetts, 2010.

[10] Kurt J. Werner, W. Ross Dunkel, Maximilian Rest, Michael J. Olsen, and Julius O. Smith III, "Wave digital filter modeling of circuits with operational amplifiers," in *Proc. of the 24th European Signal Processing Conference. (EUSIPCO-24)*, Budapest, HU, Aug. 29 – Sep. 2 2016.

[11] Kurt J. Werner, W. Ross Dunkel, and François G. Germain, "A computational model of the hammond organ vibrato/chorus using wave digital filters," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-16)*, Brno, CZ, Sep. 5 – Sep. 9 2016.

[12] Michael J. Olsen, Kurt J. Werner, and Julius O. Smith III, "Resolving grouped nonlinearities in wave digital filters using iterative techniques," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-16)*, Brno, CZ, Sep. 5 – Sep. 9 2016.

[13] W. Ross Dunkel, Maximilian Rest, Kurt J. Werner, Michael J. Olsen, and Julius O. Smith III, "The Fender Bassmann 5F6-A family of preamplifier circuits—a wave digital filter case study," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-16)*, Brno, CZ, Sep. 5 – Sep. 9 2016.

[14] Julius O Smith III, "Keynote 2: Recent progress in wave digital audio," Nov. 30 – Dec. 3 2015, Available at https://youtu.be/kUk35_WwTEQ.

[15] Matti Karjalainen, "BlockCompiler: A research tool for physical modeling and DSP," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-03)*, London, UK, Sep. 8 – Sep. 11 2003, pp. 264–269.

[16] Rudolf Rabenstein, Stefan Petrausch, Augusto Sarti, Giovanni De Sanctis, Cumhur Erkut, and Matti Karjalainen, "Block-based physical modeling for digital sound synthesis," *IEEE Signal Processing Magazine*, vol. 24, no. 2, pp. 42–54, March 2007.

[17] Giovanni De Sanctis, Augusto Sarti, Gabriele Scarparo, and Stefano Tubaro, "An integrated system for the automatic block-wise synthesis of sounds," in *Proc. of the 5th European Signal Processing Conference. (EUSIPCO-05)*, Antalya, TR, Sep. 4 – Sep. 8 2005.

[18] Udo Zölzer, Xavier Amatriain, and Daniel Arfib, *DAFX: digital audio effects, Ed. 2*, John Wiley & Sons, 2011.

[19] Maxime Coorevits, "Wave Digital Filter (WDF) with Juce," Available at http://www.juce.com/forum/topic/wave-digital-filter-wdf-juce, Feb. 2013, accessed Feb. 17, 2016.

[20] Maxime Coorevits, "WDF++ - new restructuration & project (audio processor)," Available at http://www.juce.com/forum/topic/wdf-new-restructuration-project-audio-processor, Jul. 2014, accessed Feb. 17, 2016.

[21] Dillon Sharlet, "LiveSPICE: a real time SPICE simulator for audio signals," Available at http://www.livespice.org, Nov. 2013, accessed Feb. 18, 2016.

[22] Dillon Sharlet, "How LiveSPICE works: numerically solving circuit equations," Available at http://www.dsharlet.com/2014/03/28/livespice-numerically-solving-differential-algebraic-equations-circuits, Mar. 2014, accessed Feb. 18, 2016.

[23] Teemu Voipio, "Signaldust 'Salt'," Available at http://www.kvraudio.com/forum/viewtopic.php?f=246&t=398728&hilit=vst+circuit+design, Dec. 2013, accessed Feb. 18, 2016.

[24] "JUCE framework," Available at http://www.juce.com.

[25] Conrad Sanderson, "Armadillo: An open source C++ linear algebra library for fast prototyping and computationally intensive experiments," 2010.

[26] Dietrich Fränken, Jörg Ochs, and Karlheinz Ochs, "Generation of wave digital structures for networks containing multiport elements," *IEEE Trans. on Circuits Systems I: Regular Papers*, vol. 52, no. 3, pp. 586–596, 2005.

[27] David T. Yeh and Julius O. Smith III, "Discretization of the '59 Fender Bassman tone stack," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-06)*, Montreal, CA, Sep. 18 – Sep. 20 2006, pp. 18–20.

[28] Chung-Weng Ho, Albert E. Ruehli, and Pierce A. Brennan, "The modified nodal approach to network analysis," *IEEE Trans. on Circuits and Systems*, vol. 22, no. 6, pp. 504–509, 1975.

[29] Stefano D'Angelo, Jyri Pakarinen, and Vesa Välimäki, "New Family of Wave-Digital Triode Models," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 21, no. 2, pp. 313–321, Feb. 2013.

[30] Kristjan Dempwolf and Udo Zölzer, "A physically-motivated triode model for circuit simulations," in *Proc. of the International Conference on Digital Audio Effects. (DAFx-11)*, Paris, FR, Sep. 19 – Sep. 23 2011, vol. 11.

DIRECTIVITY PATTERNS CONTROLLING THE AUDITORY SOURCE DISTANCE

Florian Wendt, Matthias Frank, Franz Zotter, Robert Höldrich

Institute of Electronic Music and Acoustics,
University of Music and Performing Arts, Graz, Austria
{ wendt , frank , zotter , hoeldrich }@iem.at

ABSTRACT

What influence does the directivity of a sound source have on the perceived distance impression in a room? We propose different directivity pattern designs able to modify the auditory source distance. The idea is accompanied with a comprehensive experimental study investigating the audio effect and its behavior by auralization of directional sound source and room using a 24-channel loudspeaker ring inside an anechoic chamber. In addition to the proposed directivity designs, the study covers influence of auralized room, source-to-receiver distance, signal, and single-channel reverberation. Moreover, simple room acoustical measures perform well in predicting the new effect.

1. INTRODUCTION

Our ability to localize sound sources with regard to distance is generally much less accurate than it is with direction. Literature suggests that humans underestimate distant sources while overestimating sources closer than 1 m [1]. Nevertheless, auditory source distance is a decisive feature when shaping auditory scenes with audio effects, reverberation, or new variable-directivity sound sources such as the icosahedral loudspeaker [2].

In audio technology and electro-acoustic music, the distance impression is often controlled by the amplitude and the direct-to-reverberant energy ratio (D/R-ratio). While listener are exquisitely sensitive to small amplitude changes in fine distance discrimination, recent studies suggest that the D/R-ratio provides coarse but absolute distance information [5].

As well as modifying the D/R-ratio, in extension of what has been presented by Laitinen [6], our contribution proposes directivity pattern designs able to control the room response in a greater variety. In doing so, the proposed designs are considered as an audio effect altering the auditory source distance.

This paper is arranged as follows: It briefly introduces directivities to affect the perceived auditory distance in section 2, subsequently outlines an exhaustive listening test design based on an auralized rooms and directivities in section 3, presents detailed results in section 4, and discusses influence of room and signal in sections 5, 6. Section 7 discusses the influence of additional single-channel reverberation, and the last section presents models of the experimental results.

2. DIRECTIVITY-CONTROLLED AUDITORY DISTANCE

An elegant solution to control auditory source distance has been proposed by Laitinen [6] and employs a variable-directivity source. Such a source generally influences the spatial structure of energy arriving at the listening position. In particular, this also affects the D/R-ratio, as a temporal structure.

We obtain a great variety of control by employing a directional source of various higher-order Ambisonics directivity patterns. Frequency-independent beampatterns up to the 3^{rd} order are obtained by a combination of Legendre polynomials $P_n(\cos\vartheta)$

$$g_i(\vartheta) = \frac{\sum_{n=0}^{i}(2n+1)\,P_n(\cos\frac{137.9^\circ}{i+1.51})P_n(\cos\vartheta)}{\sqrt{\sum_{n=0}^{i}(2n+1)\,[P_n(\cos\frac{137.9^\circ}{i+1.51})]^2}} \qquad (1)$$

using the so-called max-$\boldsymbol{r}_{\mathrm{E}}$ weights, cf. [7, 8], which yield a relatively narrow main lobe and sufficiently suppressed side lobes for any beam order i.

Fig. 1 shows the proposed beampattern designs that modify

- A the beam order i from three to zero for $g_i(\vartheta)$ and $g_i(\pi-\vartheta)$,
- B the ratio a/b of two opposing beams: $a\,g_3(\vartheta)+b\,g_3(\pi-\vartheta)$,
- C the angle α of a beam pair: $g_3(\vartheta-\alpha/2)+g_3(\vartheta+\alpha/2)$.

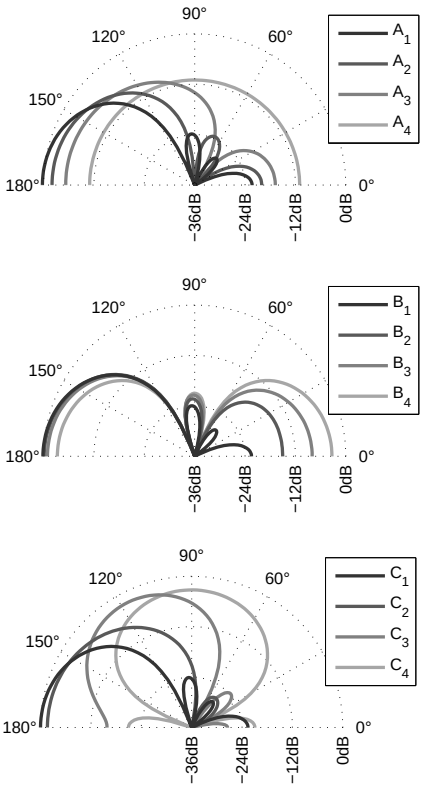


Figure 1: *Directivity designs A, B, C controlling the D/R-ratio.*

Table 1: *Properties of tested directivity designs A, B, and C.*

A	$A_{1/7}$	3^{rd}-order $max\ \mathbf{r}_{\text{E}}$ beam to/off listener
	$A_{2/6}$	2^{nd}-order $max\ \mathbf{r}_{\text{E}}$ beam to/off listener
	$A_{3/5}$	1^{st}-order $max\ \mathbf{r}_{\text{E}}$ beam to/off listener
	A_4	omnidirectional beampattern
B	$B_{1\ldots7}$	3^{rd}-order $max\ \mathbf{r}_{\text{E}}$ beams to and off listener linearly blended at $[\infty, 6, 3, 0, -3, -6, -\infty]$dB
C	$C_{1\ldots7}$	two 3^{rd}-order $max\ \mathbf{r}_{\text{E}}$ beams horizontally arranged at $\pm30° \cdot [0, 1, \ldots 6]$ wrt. the listener

Table 1 lists all tested directivity designs in particular, which differently modify the amount of diffuse, lateral, and direct energy, thus the D/R-ratio. Each directivity indicated by the index 1 and 7 corresponds to a 3^{rd}-order beam facing towards and away from the listening position ($A_1 = B_1 = C_1$, $A_7 = B_7 = C_7$). Furthermore, directivity pairs indicated by indices $1/7$, $2/6$, and $3/5$ of each design are identical in their shape but horizontally rotated by $180°$. Figure 1 shows the directivity patterns $A_{1\ldots4}$, $B_{1\ldots4}$, and $C_{1\ldots4}$ normalized to constant energy.

3. EXPERIMENTAL SETUP

The effect is evaluated in a listening experiment, in which the variable-directivity source in a room is auralized using the image source method. The room is shoebox shaped with a frequency-independent absorption coefficient $\bar{a}$. Specular reflections up to 3^{rd} order are considered [9] and diffuse reflections are simulated as spherical harmonics using the software tool MCRoomSim [10]. For simplicity, diffuse reverberation of an omni-directional excitation is considered.

Playback employed a ring of 24 equally-distributed Genelec 8020 loudspeakers with a radius of $r = 1.5$ m placed in an anechoic laboratory. Each listener was sitting in the center of the arrangement with ear height adjusted to the loudspeaker ring (see Figure 3).

On the circular setup, specular reflections are auralized by the loudspeaker with nearest azimuth angle. This avoids timbral effects of amplitude panning [11]. Elevated specular reflections are attenuated in the auralization by the cosine of their elevation. The impulse response $h_l(t)$ of the l^{th} loudspeaker is obtained after superimposing specular and diffuse reflections using MATLAB. Obviously, a two-dimensional representation of a three-dimensional sound field is not optimal, but findings in [12] indicate that reflections from floor and ceiling do not have a significant influence on the auditory source distance.

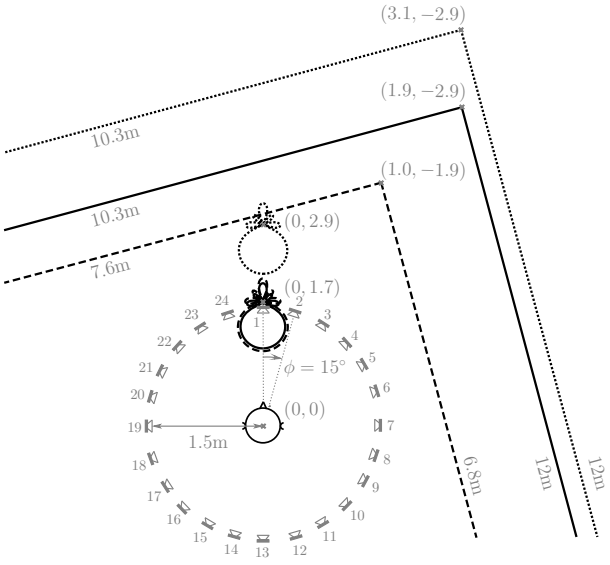


Figure 2: *Room and source constellation for R_1 (—), R_2 (⋯) and R_3 (- -) together with loudspeaker ring used for auralization.*

Table 2: *Properties of tested rooms R and signals S.*

room	R_1	IEM CUBE,	$T_{60} = 700$ ms, $d_1 = 1.7$ m
	R_2	IEM CUBE,	$T_{60} = 700$ ms, $d_2 = 2.9$ m
	R_3	IEM Lecture Room,	$T_{60} = 570$ ms, $d_3 = 1.7$ m
signal	S_1	female speech, taken from CD *B&O* 101, 1992	
	S_2	sequence of irregular artificial bursts	
	S_3	speech-spectrum noise w/ increased kurtosis	

Each impulse response was convolved with the sounds $S_{1\ldots3}$, yielding a 24-channel audio file for each condition. Audio playback was controlled by the open source software Pure Data on a standard PC with RME MADI audio interface and DirectOut D/A converters.

To monitor the influence of room acoustics, three different layouts were tested, including two rooms and two source-listener distances, see $R_{1\ldots3}$ in Tab. 2.
Geometry and reverberation time of the auralized rooms are based on the IEM CUBE, a $10.3\,\text{m} \times 12\,\text{m} \times 4.8\,\text{m}$ large room with $T_{60} = 700$ ms and the IEM Lecture Room, $7.6\,\text{m} \times 6.8\,\text{m} \times 3\,\text{m}$ with $T_{60} = 570$ ms.
The simulated sound source was placed near the corners of the room at a distance of 2 m and 3 m (IEM CUBE) and 1 m and 2 m (IEM Lecture Room). The listening position was chosen at a virtual distance of $d = 1.7$ m to the sound source, which already lies outside of the loudspeaker ring. Additionally, for the IEM CUBE an increased source-listener distance of $d = 2.9$ m was tested.
The listener was facing the sound source and the constellation, simulated at height of 1.8 m above the floor, had an angular offset of $\Delta\phi = 15°$ with regard to the sidewalls. Figure 2 shows the setup of the auralized room using the 24-channel loudspeaker ring and Table 2 lists rooms and source-listener distances tested.

The sounds fed into auralization were female speech (S_1), a sequence of irregular bursts (S_2), and Gaussian white noise shaped to speech spectrum (S_3) as listed in Table 2. For S_3, envelope fluctuations were slightly accentuated by multiplying the noise with its Hilbert envelope and by restriction to its original bandwidth, cf. [13]. By this procedure, S_1 and S_3 have similar spectra and kur-

Figure 3: *Experimental setup in the anechoic laboratory.*

Table 3: *Composition of tested sets, consisting of 7 and 9 samples.*

set no.	design	index	sound	room	reverb. level
1	A	$1 \dots 7$	S_1	R_1	0
2	A	$1 \dots 7$	S_2	R_1	0
3	A	$1 \dots 7$	S_3	R_1	0
4	B	$1 \dots 7$	S_1	R_1	0
5	B	$1 \dots 7$	S_2	R_1	0
6	B	$1 \dots 7$	S_3	R_1	0
7	C	$1 \dots 7$	S_1	R_1	0
8	C	$1 \dots 7$	S_2	R_1	0
9	C	$1 \dots 7$	S_3	R_1	0
10	A	$1 \dots 7$	S_1	R_2	0
11	A	$1 \dots 7$	S_1	R_3	0
12	A	$1 \dots 7$	S_1	R_1	1
13	A	$1, 4, 7$	$S_{1\dots3}$	R_1	0
14	A	$1, 4, 7$	S_1	$R_{1\dots3}$	0
15	A	$1, 4, 7$	S_1	R_1	$0, 1, 2$

tosis, which measures the envelope fluctuation, whereas S_2 is more transient with more energy at higher frequency ($f > 1kHz$). All sounds were normalized to their RMS value for level equalization.

The above sounds are anechoic. To monitor potential influence of additional reverberation for some conditions, sound samples were reverberated before auralization. Two levels of reverberation were tested, of which level 1 corresponds to a room impulse response with a reverberation time of $T_{60} = 0.5$ s, level 2 to one of $T_{60} = 1$ s, and level 0 to the anechoic signal.

The listening test was carried out as a multi-stimulus test where listeners had to comparatively rate multiple samples, denoted as sets. Tested sets comprise 7 samples, each representing a directivity pattern, room, sound, and reverberation level (set 1 to 12, see Table 3).

To keep the testing time decent the influence of room, signal, and reverberation level was examined with the directivity design A only. In order to allow comparability and due to the absence of common reference, the data need to be normalized. Therefore additional sets comparing multiple rooms, signals and reverberation levels with directivity patterns $A_{1,4,7}$ were included. Each of these sets consists of 9 samples and is listed in Table 3 (set 13 to 15).

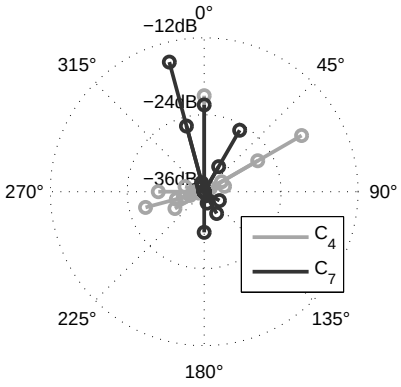


Figure 4: *Direct sound and specular reflections arriving at the listening position for C_4 and C_7, normalized wrt. C_1.*

The subjects' task was to indicate the perceived distance on a graphical user interface displaying a continuous slider for each sample of a set to permit comparative rating along the ordinal scale *very close* (vc), *close* (c), *moderate* (m), *distant* (d), and *very distant* (vd). The subjects were allowed to repeat each sample at will, and the sound files were played back in loop.

During the listening session, the subject was requested to face loudspeaker 1 ($\phi = 0°$), which corresponds to the direction of the auralized sound source.

At the beginning of the experiment, each subject was given a short training to familiarize with the evaluation scale. The training set included expected extreme values with regard to the perceived distance. Subjects were asked to rate along the whole scale and use extremes as an internal reference for further evaluations.

After the training phase, multi-stimulus tasks were presented. Each time a multi-stimulus set was displayed, the arrangement of its stimuli was an individual random permutation. The user could have the stimuli sorted by own ratings to facilitate comparative rating. The first part of the experiment consisted of the sets with 7 stimuli (set no. 1 to 12) in an individual random permutation, and the second part of the sets consisting of 9 samples (set no. 13 to 15) in an individual random permutation.

Fifteen subjects participated in the test. All of them were experienced listeners with normal hearing.

4. INFLUENCE OF DIRECTIVITY DESIGN

Fig. 5 shows a detailed analysis of the auditory source distance for the directivity designs $A_{1\dots7}$, $B_{1\dots7}$, and $C_{1\dots7}$ according to Table 1 and Fig. 1, based on the responses to the sets $1 \dots 3, 4 \dots 6$, and $7 \dots 9$ of Table 3, using all signals $S_{1\dots3}$ and the room R_1. The direct comparability of all curves in Fig. 5 is feasible as all designs were determined to include reference patterns corresponding to a 3^{rd}-order beam facing to ($A_1 = B_1 = C_1$) and off ($A_7 = B_7 = C_7$) the listening position, respectively. This allowed to linearly re-map the responses gathered in the sets $1 \dots 9$ to fill out the entire interval $[0; 1]$ for each subject. Fig. 5 shows the medians and corresponding 95% confidence intervals.

Both designs A and B yield monotonic curves. A pairwise analysis of variance (ANOVA) of the data pooled over all sounds reveals the directivity to be significant factor ($p \ll 0.01$) for $A_{1\dots5}$.

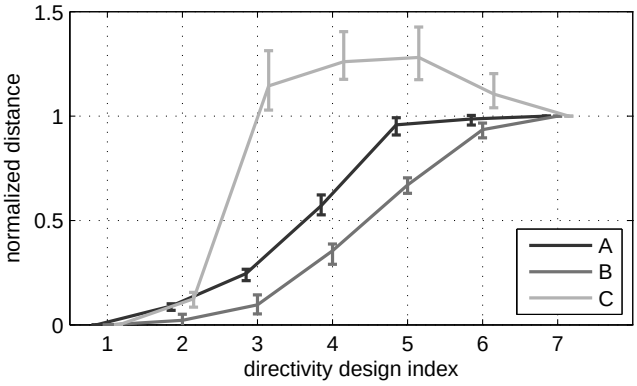


Figure 5: *Median and corresponding 95% confidence intervals for all directivity designs A, B, and C, pooled over all sounds and normalized individually on directivities indicated by 1 and 7.*

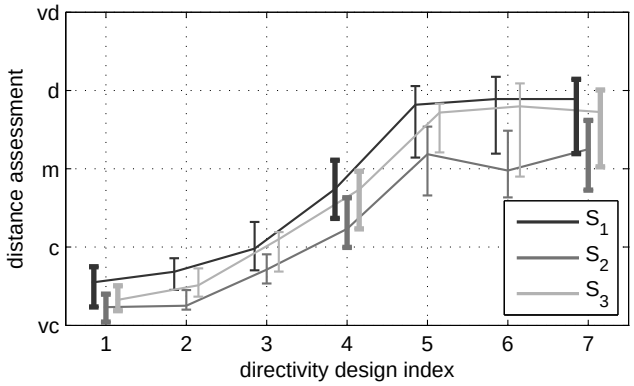


Figure 6: *Median and 95% confidence intervals for tested sounds $S_{1\ldots3}$ in R_1 with directivity design A.*

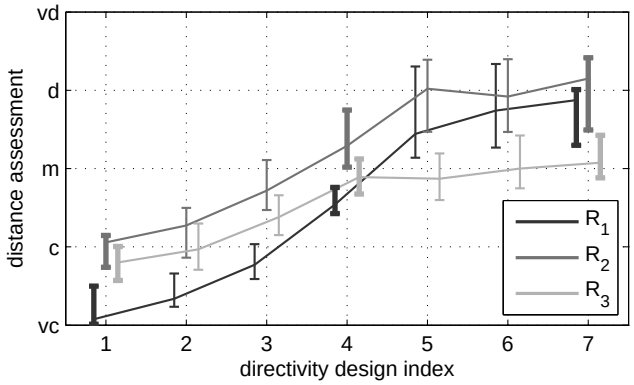


Figure 7: *Median and corresponding 95% confidence intervals for tested rooms $R_{1\ldots3}$ with directivity design A and signal S_1.*

For the design B, all directivities are significant ($B_{1\ldots7}$, $p < 0.08$). By contrast, the curve obtained for $C_{1\ldots7}$ is not monotonic in the proposed sequence. If we compare strength and angle of direct sound and specular reflections arriving at the listener for directivities C_4 and C_7, cf. Fig. 4, we see more energy coming from lateral directions for C_4. The more diffuse sound field explains the significant difference ($p \leq 0.04$) for $C_{2\ldots6}$ compared to C_7.

5. INFLUENCE OF THE SIGNAL

The influence of the signal $S_{1\ldots3}$ on the auditory source distance of the design A in R_1 is evaluated by the stimulus set 13 in Tab. 3. As the directivity indices $1, 4, 7$ of set 13 appear in the more detailed stimulus sets 1 to 3, a more detailed statistical analysis can be given in Fig. 6. Responses for indices $1, 4$ and 7 (set 13) are supplemented by the linearly re-mapped responses for $2, 3, 5, 6$ (set 1 to 3) for each subject, to fill out the ranges between the median values for the indices 1, 4 or 4, 7, respectively.

Fig. 6 shows the median values and corresponding 95% confidence intervals of the auditory source distance for the room R_1 and directivity designs $A_{1\ldots7}$. Along the indices, the distance impression exhibits a monotonic increase for all sounds until condition A_5. The ANOVA of neighboring values reveals conditions A_2 to A_5 as a significant factor ($p < 0.03$). By contrast, conditions $A_{5\ldots7}$ do not yield a significant change ($p \geq 0.45$), despite continuously reducing the D/R-ratio. This seems to comply with a general tendency to auditorily underestimate the physical distance [1].

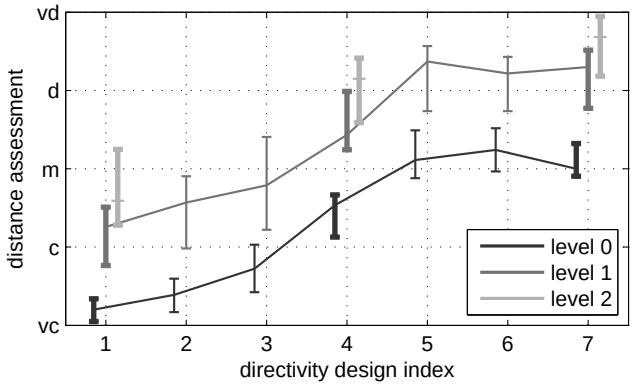


Figure 8: *Median and corresponding 95% confidence intervals for reverberation levels 0, 1, 2 in R_1 with S_1 and directivity design A.*

A sound-wise comparison of the obtained data reveals the significantly smaller auditory source distance for S_2 than for S_1 or S_3 ($p_{S_2/S_1} \ll 0.01$, $p_{S_2/S_3} = 0.02$). This seems to comply with the finding in [14, 15] that the auditory source distance of broadband signals decreases with the relative amount of high-frequency energy.

6. INFLUENCE OF THE ROOM

The influence of the room and the source-to-listener distance ($R_{1\ldots3}$) is evaluated by the data of the set 14. Figure 7 shows the median values and corresponding 95% confidence intervals, regarding signal S_1 and directivity design A, supplemented by the linearly and individually re-mapped responses of the sets 1, 10, and 11.

A smaller room with shorter T_{60} and sound source closer to adjacent walls but with the same source-to-listener distance (R_3) leads to a flatter curve. Similar flattening accompanied by an additional offset to bigger auditory source distances is achieved by extending the source-to-listener distance (R_2). Interestingly, for all tested rooms R the directivity is a significant factor ($p_{R_1} < 0.09, p_{R_2} < 0.03, p_{R_3} < 0.04$) in the range of $A_{1\ldots5}$. This significance is similar to the values obtained with pooled sounds $S_{1\ldots3}$ ($p \ll 0.01$, see Fig. 6).

7. INFLUENCE OF SINGLE-CHANNEL REVERBERATION

In electro-acoustics reverberation effects are used to control depth of sounds. To get an idea how this affects the perceived distance, artificial reverberation is added to signal S_1 and tested with directivity patterns $A_{1,4,7}$ in room R_1. Fig. 8 shows respective median values together with corresponding 95% confidence intervals. According to the ANOVA, the influence of reverberation on the auditory source distance is significant ($p < 0.05$).

Individually and linearly re-mapped responses from the sets 1 and 12 were used supplementing the responses from set 15 to provide a more detailed analysis for the reverberation levels $0, 1$ in terms progression over the 7 design indices. Both reverberation levels yield a similar progression with the known saturation for $A_{>5}$. Directivity is a significant factor ($p < 0.09$) for the dry signal (rev. level 0) in the range of $A_{1\ldots5}$, and by the addition of reverberation (rev. level 1), differences between the neighboring conditions $A_{1,2}$ and $A_{2,3}$ are no longer significant ($p \geq 0.16$).

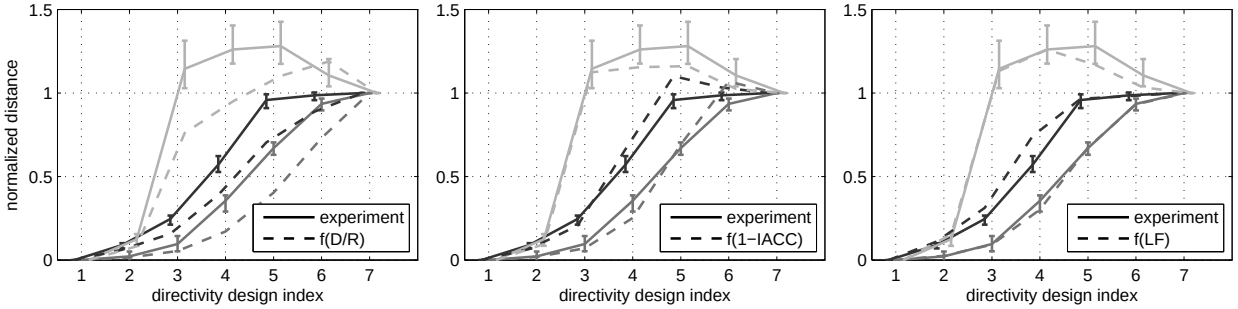


Figure 9: *Comparison of Median and 95% confidence intervals for all conditions with predictors: D/R, LF, and* $1 - \mathrm{IACC}$.

8. MODELING THE AUDITORY SOURCE DISTANCE

This section discusses linear auditory source distance models for the presented effect, based on characteristic metrics of the spatial sound field and their regression to the experimental data.

8.1. Direct-to-reverberant energy ratio

The most obvious predictor in this context is the D/R-ratio. It is widely accepted for prediction of auditory source distance [1] and is defined as

$$\mathrm{D/R} = 10 \log_{10} \frac{\int_{0\mathrm{ms}}^{T} s^2(t)dt}{\int_{T}^{\infty} s^2(t)dt}. \tag{2}$$

By using $s(t) = \sum_l h_l(t)$, the D/R-ratio can be calculated based on the loudspeaker impulse responses, with a time constant T regarding only direct sound.
Regression analysis fits a linear regression function $f(\mathrm{D/R}) = k\mathrm{D/R} + d$ depending on the D/R-ratio to the normalized experimental data and yields $k = -0.049$ and $d = 0.11$. Figure 9 shows the pooled data compared with $f(\mathrm{D/R})$. Although the D/R-ratio and the median values of the pooled data are highly correlated ($R^2 = 0.93$) their progression along the directivity indices is qualitatively different.

8.2. Inter-aural cross correlation coefficient

As reverberation caused by the room simulation introduces binaural cues by altering the sound attributes at the two ears differentially, the inter-aural cross correlation coefficient (IACC) is used as an additional measure for auditory source distance. The IACC is based on the inter-aural cross correlation function (IACF):

$$\mathrm{IACF}(\tau) = \frac{\int_{0\mathrm{ms}}^{80\mathrm{ms}} s_{\mathrm{left}}(t) s_{\mathrm{right}}(t+\tau)dt}{\sqrt{[\int_{0\mathrm{ms}}^{80\mathrm{ms}} s_{\mathrm{left}}^2(t)dt][\int_{0\mathrm{ms}}^{80\mathrm{ms}} s_{\mathrm{right}}^2(t)dt]}}, \tag{3}$$

with $s_{\mathrm{left}}(t) = h_{\mathrm{left}}(t) * s(t)$ and $s_{\mathrm{right}}(t) = h_{\mathrm{right}}(t) * s(t)$. The binaural impulse response $h(t)$ corresponds to responses for left and right ear at $\phi = 0^\circ$.
The IACC is defined as the maximum absolute value within $\tau = \pm 1$ ms:

$$\mathrm{IACC} = \max_{\forall \tau \in [-1\mathrm{ms};1\mathrm{ms}]} |\mathrm{IAFC}(\tau)|. \tag{4}$$

Simply stated, IACC is a powerful binaural cue of the similarity between ear signals [3].

It is widely accepted that a lower IACC value leads to a bigger spatial impression, and therefore $1 - \mathrm{IACC}$ is positively correlated with the magnitude of perceived spatial impression.
With the IACC binaurally measured in the experimental setup, linear regression yields $f(1 - \mathrm{IACC}) = 2.23(1 - \mathrm{IACC}) - 0.87$ to model the experimental data ($R^2 = 0.98$, cf. Fig. 9).

8.3. Lateral energy fraction

The lateral energy fraction (LF) is another acoustic measure quantifying the spatial impression. More specifically, it has been accepted as a measure of the effect of source broadening [16, 17]. Simply stated, LF is the ratio of the sum of the early lateral energy to the sum of the early total energy:

$$\mathrm{LF} = \frac{\int_{5\mathrm{ms}}^{80\mathrm{ms}} s_{\mathrm{lat}}^2(t)dt}{\int_{0\mathrm{ms}}^{80\mathrm{ms}} s^2(t)dt}, \tag{5}$$

with $s_{\mathrm{lat}}(t) = \sum_l h_l(t) \sin(\phi_l)$ and ϕ_l as azimuthal angle of the l^{th} loudspeaker.
Linear regression yields $f(\mathrm{LF}) = 7.3\mathrm{LF} - 0.54$, cf. Fig. 9. This LF-based linear model delivers the best matching results. This is underlined by its sublime correlation of $R^2 = 0.99$.

9. CONCLUSIONS

In this contribution, an investigation was carried out into the influence of various directivity patterns on the perceived auditory distance. Two-dimensional simulation of a variable-directivity sound source was shown to provide control of the perceived auditory distance from a single point in the room. Different beampattern designs were proposed that cause pronounced and graduated distance impressions. Additionally, the influence of auralized room, source-to-receiver distance, signal, and single-channel reverberation was studied.

The mapping of the directivity designs $A_{1\ldots7}$ and $B_{1\ldots7}$ to perceived distance curves is sigmoid-shaped. It resembles the compressive power functions described in [5] characterizing the relation between physical and perceived distance. Moreover, agreeing with [14, 15], signals with an increased relative amount of high-frequency energy appeared to be closer in the study.
Both decreasing the auralized room and increasing the source-to-receiver distance yield a more compressed curve, which is slight offset in case of the increased source-to-receiver distance. Despite this, the range of discriminability is persistent.

The use of single-channel reverberation is also effective at increasing the perceived auditory distance, however, it narrows the directivity-controllable range of distinguishable distance impressions.

Finally, successful modeling of the experimental results was presented. All models are highly correlated with the experimental data. Interestingly, spatial measures used to quantify the apparent source width provide very accurate predictions.
In a room, the physical distance to a source typically increases the amount of reflected sound in relation to the direct sound. Consequently, this affects the measures $1-\mathrm{IACC}$ and LF for the apparent source width, as the measurements in [18] showed. Our listening experiments only asked for distance ratings. Further research is required to determine to what extent the auditory distance and width are separable.

10. ACKNOWLEDGMENTS

The authors thank all subjects for their participation in the listening experiment. This work was funded by the Austrian Science Fund (FWF) project nr. AR 328-G21, Orchestrating Space by Icosahedral Loudspeaker.

11. REFERENCES

[1] P. Zahorik, D. S. Brungart, and A. W. Bronkhorst, "Auditory distance perception in humans: A summary of past and present research," *Acta Acustica united with Acustica*, vol. 91, no. 3, pp. 409–420, 2005.

[2] G. K. Sharma, F. Zotter, and M. Frank, "Orchestrating wall reflections in space by icosahedral loudspeaker: findings from first artistic research exploration," in *Proc. ICMC/SMC, Athens*, pp. 830–835, 2014.

[3] J. Blauert, *Spatial hearing - the psychophysics of human sound source localization*. The MIT Press, 1983.

[4] D. S. Brungart, N. I. Durlach, and W. M. Rabinowitz, "Auditory localization of nearby sources. II. Localization of a broadband source.," *The Journal of the Acoustical Society of America*, vol. 106, no. October, pp. 1956–1968, 1999.

[5] P. Zahorik, "Assessing auditory distance perception using virtual acoustics.," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1832–1846, 2002.

[6] M.-V. Laitinen, A. Politis, I. Huhtakallio, and V. Pulkki, "Controlling the perceived distance of an auditory object by manipulation of loudspeaker directivity," *The Journal of the Acoustical Society of America*, vol. 137, no. 6, pp. EL462–EL468, 2015.

[7] J. Daniel, *Représentation de champs acoustiques, application à la transmission et à la reproduction de scènes sonores complexes dans un contexte multimédia*. PhD thesis, Université Paris 6, 2001.

[8] F. Zotter and M. Frank, "All-round ambisonic panning and decoding," *AES: Journal of the Audio Engineering Society*, vol. 60, no. 10, pp. 807–820, 2012.

[9] J. B. Allen and D. A. Berkley, "Image Method for Efficiently Simulating Small-room Acoustics," 1979.

[10] A. Wabnitz, N. Epain, C. T. Jin, and A. Van Schaik, "Room acoustics simulation for multichannel microphone arrays," *International Symposium on Room Acoustics*, no. August, pp. CDROM: 1–6, 2010.

[11] S. Tervo, J. Pätynen, A. Kuusinen, and T. Lokki, "Spatial decomposition method for room impulse responses," *Journal of the Audio Engineering Society*, vol. 61, no. 1/2, pp. 17–28, 2013.

[12] R. Guski, "Auditory localization: effects of reflecting surfaces.," *Perception*, vol. 19, no. 6, pp. 819–830, 1990.

[13] A. Kohlrausch, R. Kortekaas, M. van der Heijden, S. van de Par, A. J. Oxenham, and D. Püschel, "Detection of Tones in Low-noise Noise: Further Evidence for the Role of Envelope Fluctuations," *Acta Acustica united with Acustica*, vol. 83, pp. 659–669, 1997.

[14] P. Coleman, "Dual Role of Frequency Spectrum in Determination of Auditory Distance," *Journal of the Acoustical Society of America*, vol. 44, no. 2, pp. 631–632, 1968.

[15] A. D. Little, D. H. Mershon, and P. H. Cox, "Spectral content as a cue to perceived auditory distance.," *Perception*, vol. 21, no. 3, pp. 405–416, 1992.

[16] A. H. Marshall, "A note on the importance of room cross-section in concert halls," *Journal of Sound and Vibration*, vol. 5(1), pp. 100–112, 1967.

[17] M. Barron and A. H. Marshall, "Spatial impression due to early lateral reflections in concert halls: The derivation of a physical measure," *Journal of Sound and Vibration*, vol. 77, no. 2, pp. 211–232, 1981.

[18] H. Lee, "Apparent Source Width and Listener Envelopment in Relation to Source-Listener Distance," *Audio Engineering Society Conference*, pp. 1–6, 2013.

AUDITORY PERCEPTION OF SPATIAL EXTENT IN THE HORIZONTAL AND VERTICAL PLANE

Marian Weger, Georgios Marentakis, Robert Höldrich *

Institute of Electronic Music and Acoustics
University of Music and Performing Arts, Graz, Austria
{weger,marentakis,hoeldrich}@iem.at

ABSTRACT

This article investigates the accuracy with which listeners can identify the spatial extent of distributed sound sources. Either the complementary frequency bands comprising a source signal or the individual grains of a granular synthesis-based stimulus were distributed directly on discrete loudspeakers. Loudspeakers were arranged either on the horizontal or the vertical axis. The algorithms were applied on white noise, an impulse train, and a rain drops stimulus. Absolute judgments of spatial extent were obtained separately for each orientation, algorithm, and stimulus using three different magnitudes of horizontal or vertical extent.

Horizontal spatial extent judgments varied systematically with physical extent for all conditions in the experiment. The correspondence between perceived and actual vertical extent was poor. The time-based synthesis algorithm resulted in significantly larger judgments of spatial extent irrespective of orientation and stimulus compared to the frequency-based algorithm.

1. INTRODUCTION

Perceived spatial extent is a measure of the perceived spatial volume that may be occupied by an auditory event. The term was proposed by [1] and may refer independently to width, height, and potentially also depth. It is important to note here that although most often in the literature the term Auditory or Apparent Source Width (ASW) has been used to refer to the perceived horizontal spatial extent of a sound, we use the term spatial extent here because it allows us to differentiate between spatial extent perception along each of the three axes of the Cartesian coordinate system. Furthermore, in the following, by spatial extent we refer exclusively to the spatial extent of the perceived auditory object and not of the sound producing object.

This study is motivated by the revived interest in algorithms for the representation of auditory spatial extent in the last years. This interest is justifiable if one considers that reliable representation of auditory spatial extent could be useful for both scientific and artistic purposes. Concerning music for example, reliable representations of spatial extent could provide an extra design parameter for composers, sound engineers, and music producers. Concerning interactive systems, such algorithms could improve and augment auditory representations in virtual and mixed reality systems. Importantly, successful representation of horizontal and vertical extent may pave the way for representing more complex shapes with sound, which would be vital for assistive technologies for example. In this article, we focus on the synthesis of relatively small spatial extents, keeping an eye on applications for which the space to deploy loudspeakers is limited. We proceed by first reviewing the literature and then presenting the experiment and their results.

Perceived spatial extent is influenced by both spatial and non-spatial acoustical features. Non-spatial features that affect the perception of spatial extent include loudness, duration, and base frequency of a sound. Increased sound pressure level and duration and lower base frequency are generally associated with larger spatial extent of sound generating sources [2, 3, 4]. Furthermore, decisions about the shape and size of sounding objects can be reached on the basis of spectral cues such as the (sometimes direct) relationship between the modal frequencies of vibrating objects and their geometric shapes and size. In experiments, above chance identification of auditory source shape solely based on spectral cues has been observed [5, 6, 7].

Concerning spatial factors, studies have focused on the perception of horizontal spatial extent (or ASW). This increases in reverse proportion to the interaural cross-correlation coefficient (IACC) [8, 2, 9].

A significant number of studies investigated the horizontal spatial extent of the auditory event that emeges when simultaneous uncorrelated noise sources are distributed directly to individual loudspeakers. Linear or circular loudspeaker arrangements were tested [10, 11, 12, 13]. It was shown that the perceived horizontal extent of such stimuli varies in proportion to the actual spatial extent occupied by the noise sources. The perceived horizontal spatial extent is, however, narrower than the actual spatial extent [12, 11]. Increasing the noise bandwidth or center frequency [11, 14], or the signal duration [13] results in a wider perceived spatial extent. Furthermore, small gaps in the loudspeaker spatial distribution are not easily noticed while the size of large gaps is often exaggerated [12].

In practice, decorrelation techniques for arbitrary monophonic signals are used to recreate a similar effect. A very promising approach works by splitting an auditory signal into a number of unique frequency bands which are then spatialized directly on loudspeakers or as virtual sources [10, 15, 16, 17]. Most often, signals are decomposed in bands whose bandwidth and center frequency correspond to the Equivalent Rectangular Bandwidth (ERB) scale [18, 10]. The way frequency bands are mapped to spatial positions is important as it influences both the center and the spatial extent of the perceived auditory event [10, 19]. Convincing synthesis of horizontal spatial extent has been achieved by using a Halton sequence [20] to map frequency bands to fixed locations [16]. In evaluation studies, this method resulted in perceived horizontal

* Contributions: Marian Weger and Georgios Marentakis designed the experiment, performed the statistical analysis, and authored the article. Marian Weger implemented and executed the evaluation study. Robert Höldrich contributed essential knowledge on acoustic measurements and predictors for apparent source width and provided useful comments and corrections to the article. This work was supported by the Zukunftsfonds Steiermark Klangräume Project (PN:6067) led by Georgios Marentakis.

spatial extent proportional to the physical extent of the distributed sound source. Impressions ranging from a narrow focused source to sounds completely surrounding the listener were obtained using a circular loudspeaker array. Sound quality was however strongly signal dependent [16]; this is a common problem in decorrelation techniques [21, 16, 17].

Another approach, originating in electroacoustic music, is to create spatially extended sound sources using spatialized granular synthesis [22, 23, 24]. Granular synthesis generates sounds by combining short signals (grains) [25, 24]. It can result in a great variety of sounds, including those of everyday events, such as rain, applause, etc. As grains are in general short and may be designed to have steep attacks they can be localized well. Their potential for spatial extent synthesis is therefore high. This hypothesis has however not been tested experimentally.

The two aforementioned algorithms, which from here on will be called the frequency-based and the time-based algorithm, are in a sense complementary to each other. While in the frequency-based algorithm, the frequency bands of a monophonic input signal are spatialized independently to yield a coherent sound, in the time-based algorithm this is achieved by using individually spatialized temporal grains. In both cases, it is envisaged that the spatial extent of the auditory event will relate to the spatial distribution of the grains or frequency bands comprising the source. However, both algorithms might be sensitive to the way grains or frequencies are spatialized in addition to the size and geometry of the area within which the signal content is distributed. It is reasonable to hypothesize that the allocation of the individual grains/frequencies to a single auditory event may be infeasible above a certain spatial dispersion.

The above observations motivated us to design and implement an experiment that compares the aforementioned time- and frequency-based spatial extent synthesis algorithms on the basis of their ability to create the impression of spatially distributed sound sources. Our experiment investigates the synthesis of both horizontally and vertically extended sound sources, with extents that are smaller compared to the ones used in the literature. The aim was to understand the relevance of the aforementioned synthesis techniques for fields other than surround music production, e.g., for Human Computer Interaction (HCI) applications.

2. EXPERIMENT

In the experiment, participants performed absolute judgments of perceived spatial extent in conditions that manipulated the spatial extent synthesis algorithm, the type of stimulus used, and the orientation and length of the spatial distribution of the loudspeakers that were used to distribute the stimuli. An overview of the variables is provided in Table 1 and a photo of the experiment setting is provided in Figure 2.

With reference to Figure 2, small, medium, and large spatial distributions were simulated by distributing signals on 3, 7, or 11 adjacent loudspeakers respectively using either the frequency- or the time-based algorithm. Distribution orientation was either horizontal or vertical. As the experiment targeted also the perception of vertical spatial extent, we opted to use discrete loudspeakers instead of phantom sources. This was done specifically because panning algorithms are known to provide weak and inaccurate perception of the vertical location of elevated phantom sources [26, 27, 28, 29].

Table 1: *The independent variables in the experiment.*

Factor	*Levels*
Spatial Distribution	small medium large
Algorithm	frequency-based (FB) time-based (TB)
Stimulus	white noise impulse train rain drops
Orientation	horizontal vertical

2.1. Stimuli

Three different stimuli were used in the experiment. The first two were white noise and an impulse train. These were chosen because they represent optimal scenarios for the frequency- and time-based algorithms, respectively. The third stimulus was designed to create the impression of strong rain and represented a more realistic scenario.

To create this rain drops stimulus 48 different rain drop samples were used. These were extracted from a recording of rain and normalized to the same amplitude. Average duration was 46 ms (standard deviation SD=18 ms) with an approximate attack time[1] of 2.2 ms (SD=1.8 ms). They were combined using a typical granular synthesis algorithm that selected grains by drawing samples from a uniform distribution. The onset of the next event relative to the onset of the current one was sampled from a normal distribution with mean M=10 ms (100 Hz) and SD=3 ms. Occasional negative delays were mirrored around zero to positive ones. Randomizing delay helped to avoid the impression of a pitched sound. The impulse train stimulus was implemented similar to the rain drops stimulus, but with a Dirac impulse as a grain.

While both white noise and impulse train stimuli have a flat frequency spectrum up to half the sampling frequency, the averaged spectral energy of the rain drops stimulus was concentrated primarily in the region between 2 kHz and 7 kHz.

2.2. Algorithms

In case of the frequency-based algorithm, the condition-dependent monophonic input signal was decomposed into frequency-bands whose center frequency and bandwidth corresponded to the Equivalent Rectangular Bandwidth (ERB) scale [30], according to the algorithm proposed in [10] and [19]. 38 ERB-bands with center frequencies from 142.5 Hz to 19.7 kHz were chosen. The complementary ERB-filters were implemented as rectangular windows in the spectral domain using the short-time Fourier transform (STFT) with an FFT size of 1024 samples. Hann-window and 75% overlap were chosen to yield perfect reconstruction[31, p. 113].

To make sure that the ERB-bands are evenly distributed to all active loudspeakers, and to ensure a reproducible distribution, the output channel to which each individual ERB-band was mapped, was chosen by using a Halton sequence [20], as proposed by [16]. In particular, a long (1000 elements) Halton sequence of base 2

[1] In this context the attack time was defined as the time to reach the lowest maximum of all grain's envelopes. The envelope of a signal was computed as the absolute value of its discrete-time analytic signal (Hilbert transform).

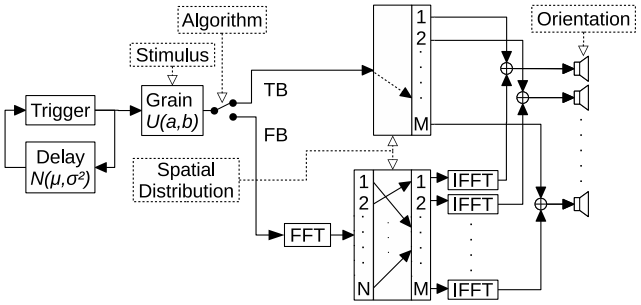


Figure 1: *Simplified block diagram of the signal chain for the granular synthesis-based stimuli (impulse train and rain drops). The independent variables of the experiment are framed in dotted lines. N: number of ERB-bands. M: number of output channels.*

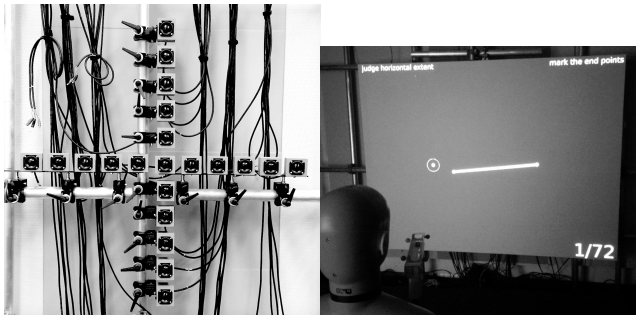


Figure 2: *The planar loudspeaker array (left) and the experiment setting (right).*

(without offset) was pre-computed. First, the resulting Halton sequence elements (that range between 0 and 1) were mapped to the 11 loudspeakers of the linear loudspeaker array (counted from left to right / top to bottom). This sequence was used to generate individual sequences for each condition. For each of those sequences, numbers corresponding to loudspeakers inactive for the corresponding condition were removed. In a second step, direct repetitions (neighboring bands in the same loudspeaker) were removed from the resulting sequence and it was made sure that no loudspeaker received more than one band than the rest to avoid concentrating energy on one loudspeaker. The first 38 sequence numbers of the remaining sequence were used to indicate the channels (from 1 to 11) in which the corresponding ERB-bands would be rendered[2]. The signal to each loudspeaker was calculated using 3, 7, or 11 complementary filters containing the relevant frequency bands. In all conditions, the simulated signal power difference between loudspeakers was less than 2 dBA.

In the case of the time-based algorithm, the individual grains (Dirac impulses or rain drop samples) were simply routed to discrete loudspeakers depending on the spatial distribution in each condition. This was done during sound synthesis which provided access to the individual grains. Each time a grain or impulse was played from a given loudspeaker location, the location of the next grain was selected again using a pre-defined sequence similar to the one in the frequency-based algorithm. However, now the sequence lengths corresponded to the number of active loudspeakers and each channel number was included once.[3]

A simplified overview of the complete signal chain for the granular synthesis-based stimuli (impulse train and rain drops) is illustrated in Figure 1.

When being processed by the frequency-based algorithm both granular stimuli (rain drops and impulse train) were synthesized as monophonic signals and a single monophonic white noise signal was used. The algorithm then did the filtering and the playback from the relevant channels in the same way for all stimuli. When being processed by the time-based algorithm, each grain (rain drop or click) of the two granular stimuli was allocated to its corresponding channel as soon as it was generated. The white noise stimulus was not processed by the time-based algorithm. Instead, each active loudspeaker depending on the condition played statistically independent white noise. This exception served as a control condition in the experiment. It was assigned to the time-based algorithm to simplify the condition names in the experiment.

2.3. Apparatus

Twenty-one custom 2-inch broadband speakers and class-D amplifiers were used [32], connected to Behringer ADA8000 DA converters running at 44.1 kHz / 24 bit. The loudspeakers were arranged on two intersecting lines of 11 loudspeakers each, which were aligned on the horizontal and vertical axis. The center speaker was shared and placed at a height of 120 cm, roughly aligned with the nose of the listeners. There was 10 cm distance between neighboring membrane centers leading to a maximum distance of 1 m between the two outmost speakers. Loudspeakers were aligned to the direction of the listener and hidden behind a 2 by 1.5 m acoustically transparent projection screen[4] installed 10 cm in front of them (see Figure 2). The three different spatial distributions led to physical extents of 25, 65, and 105 cm, or 7.2, 18.5, and 29.4 degrees.

Differences in distance to the listener were compensated by loudspeaker-specific gain and delay corrections, and to protect the loudspeakers all channels in all conditions were high-pass filtered at 200 Hz. Equivalent Continuous Sound Level (Leq) was 55 dBA at the listening position for all stimuli. The experiment took place in an acoustically treated room of 4.3(w) × 6.2(l) × 3.4(h) m size, with reverberation times between 0.15 and 0.22 s in the relevant frequency range above 200 Hz. The direct-to-reverberant ratio (DRR) [33] at the listening position was between 7 and 11 dB for the individual loudspeakers.

Participants performed the experimental task using a toy gun to indicate perceived horizontal or vertical extent. To achieve this, infrared reflective markers were mounted on the gun and the projection screen and tracked using a NaturalPoint OptiTrack optical motion capture system. The experiment and graphics were implemented in Pure Data using the Extended View Toolkit [34] for projection mapping.

[2]Small: 6,5,7,5,6,7,5,6,7,5,7,6,7,5,6,5,7,6,7,5,6,7,5,6,7,5,6,5 7,6,7,5,6,5,7,6,7,5; Medium: 6,4,8,3,7,5,9,3,6,5,8,4,7,9,3,6,4,8,7,5,9,3,7,5,8,4,6,9,3,6,4,8,7,5,9,3,7,5; Large: 6,3,9,2,7,5,10,1,4,8,11,1,6,4,9,2,8,5,10,7,3,11,1,6,3,9,2,8,5,10,7,4,11,1,7,4,9,2

[3]Small: 6,7,5; Medium: 6,3,9,7,5,4,8; Large: 6,3,9,2,7,5,10,1,4,8,11

[4]Gerriets OPERA® white perforated (PVC, 390 g/m^2, 7 percent perforation area)

Table 2: *The results of a four-way (Stimulus × Algorithm × Spatial Distribution × Orientation) repeated measures ANOVA on perceived spatial extent. Non-significant main effects and interactions (p>0.05) were omitted.*

Stimulus	F(2,34) = 20.749	p < 0.001
Algorithm	F(1,17) = 47.679	p < 0.001
Spatial Distribution	F(2,34) = 19.550	p < 0.001
Orientation	F(1,17) = 16.852	p = 0.001
Algorithm × Spatial Distribution	F(2,34) = 24.634	p < 0.001
Algorithm × Orientation	F(1,17) = 44.511	p < 0.001
Spatial Distribution × Orientation	F(2,34) = 15.171	p < 0.001
Algorithm × Spatial Distribution × Orient.	F(2,34) = 18.897	p < 0.001

2.4. Procedure and Participants

Participants sat on a chair at a distance of 2 m from the loudspeaker array. They went through the trials in a randomized order; there was a 500 ms silence between trials, enough to reset short-term echoic memory [35]. Each stimulus was presented continuously until the trial was over, with a 5 ms linear fade in and fade out. Horizontal and vertical orientation were tested in two separate trial groups presented in a counterbalanced order. There were four repetitions for each combination of algorithm, stimulus, and spatial distribution, leading to a total of 72 stimuli for both orientations.

Participants were instructed to use the toy gun to draw a straight line on the projection screen and to match its horizontal (or vertical) spatial extent to the perceived auditory horizontal (or vertical) spatial extent. They triggered the gun to indicate and adjust the end points of the line and were able to perform corrections before pressing a button on the gun to proceed to the next trial. 18 participants (5 female, M=26.3 years, SD=5.4 years), participated and received a small financial compensation. None of them had prior knowledge or training in the specific task. The task was not restricted in time.

2.5. Results

Perceived vs. physical spatial extent in the different conditions in the experiment are illustrated in Figure 3. It is evident that in most cases perceived extent was narrower in vertical compared to horizontal orientation. In addition, it appears that the time-based algorithm results in broader spatial extent perceptions compared to the frequency-based algorithm. Finally, although results are similar for the white noise and impulse train stimuli, the rain drops stimulus appears to be perceived narrower than the other signals.

The judgments in the different conditions in the experiment were verified to follow the normal distribution using the Lilliefors test [36]. No outliers were detected by Grubbs' test [37]. A four-way (Stimulus × Algorithm × Spatial Distribution × Orientation) repeated measures ANOVA on perceived spatial extent, as indicated by the horizontal or vertical distance between selection endpoints, was used to analyze the results (see Table 2). No violations of sphericity were observed.

2.5.1. Main effects

The main effects of Stimulus, Algorithm, Spatial Distribution, and Orientation were significant. Pairwise t-tests showed that white noise and impulse trains resulted in significantly broader perceived extent in comparison to the rain drops (p<0.001). Pairwise t-tests

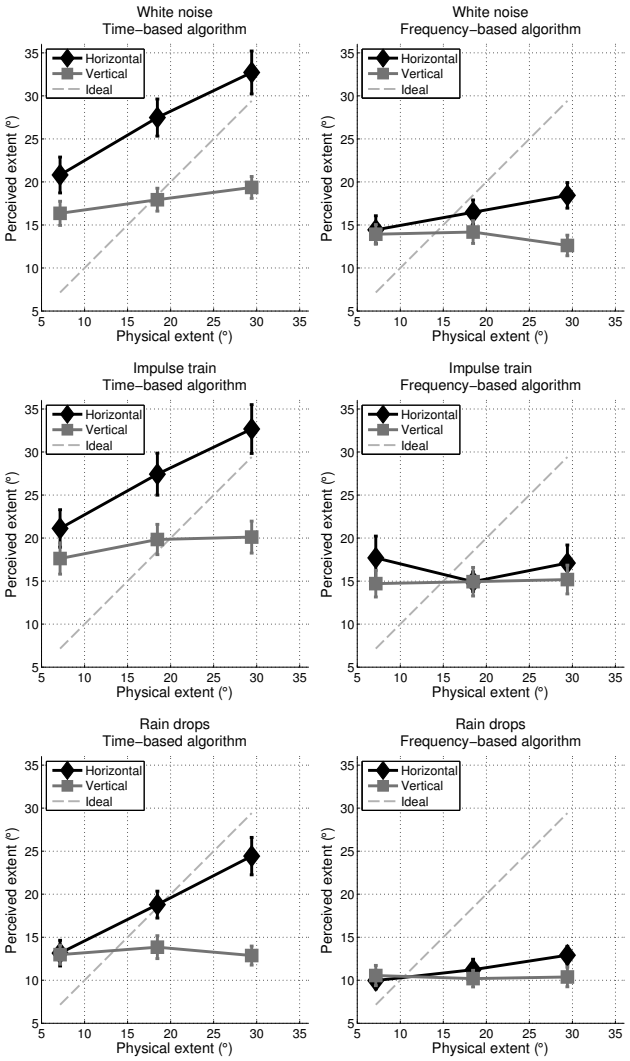


Figure 3: *Perceived vs. physical spatial extent in the different conditions in the experiment. Error bars indicate standard error of the mean.*

showed that the small spatial distribution (3 loudspeakers) was perceived to be significantly narrower than both the medium (7 loudspeakers) and the large (11 loudspeakers) distributions and the medium distribution narrower than the large (p<0.01). Finally, judgments of vertical extent were significantly narrower than those of horizontal extent, and the time-based algorithm resulted in significantly broader judgments.

2.5.2. Algorithm and Spatial Distribution

The interaction between Algorithm and Spatial Distribution was significant. This was because in the case of the time-based algorithm, averaged over orientation and stimulus, perceived extent was significantly different for the three spatial distributions tested in the experiment, e.g., small was perceived as significantly narrower than medium and large spatial distributions, and medium significantly narrower than the large spatial distribution (p<0.001).

This was not the case for the frequency-based algorithm, in which case averaged over stimuli and orientation no statistically significant differences in the perceived extent of different spatial distributions were observed.

2.5.3. Algorithm and Orientation

The interaction between Algorithm and Orientation was significant because in the case of the time-based algorithm, averaged over Stimulus and Spatial Distribution, judgments of the horizontal extent were significantly wider compared to those of vertical extent ($t(17)=5.855$, $p<0.001$), while for the frequency-based algorithm there was no significant difference between the extent of the horizontal and vertical judgments, arguably because they were narrow in both cases.

2.5.4. Spatial Distribution and Orientation

The interaction between Spatial Distribution and Orientation was also significant. This was because of two reasons. The first was that when loudspeakers were arranged horizontally, perceived horizontal extent was significantly influenced from the actual spatial extent, i.e., variations in the spatial distributions resulted in significantly different perceived spatial extent (pairwise-comparisons, at least $p<0.01$). When loudspeakers were aligned vertically, however, the perceived vertical extent varied only little in response to changes in the actual physical extent. Perceived spatial extents for the different spatial distributions were only marginally significantly different to each other, small smaller than medium ($t(17)=-1.839$, $p=0.083$), small smaller than large ($t(17)=-1.865$, $p=0.080$), medium vs. large not significant. The second reason is that for the smallest actual spatial distribution, the perceived horizontal and vertical extent judgments were not different to each other. However, perceived horizontal and vertical extent judgments in the other two spatial distributions were significantly different to each other in the horizontal but not in the vertical orientation.

2.5.5. Algorithm, Spatial Distribution, and Orientation

The three-way interaction between Algorithm, Spatial Distribution, and Orientation was also significant. This was because irrespective of stimulus and for spatial distributions other than the small, the perceived spatial extent was significantly larger in the case of the time-based spatialization for horizontally aligned stimuli in comparison to vertically aligned ones. This interpretation is supported by the observation that the two-way interaction between Orientation and Spatial Distribution was significant for the time-based spatialization algorithm but not for the frequency-based when analyzing the data averaged over stimuli.

3. PREDICTORS FOR APPARENT SOURCE WIDTH

For precise control of perceived spatial extent, predictors which can be calculated on the basis of measurable signal properties are sought. For vertical spatial extent no reliable predictors are known. However, in the case of horizontal spatial extent, i.e., apparent source width, interaural cross-correlation (IACC) and the lateral energy fraction (LF) may provide acceptable results. LF is related to IACC in the sense that a decorrelation between the two ear signals, yielding a low IACC (or high LF), could emerge on the one hand from a spatial distribution of uncorrelated individual sound sources, or on the other hand from room reflections [38]. On the other hand, changes in IACC may not only result from changes in the lateral energy arriving in the ears and may be the result of decorrelation operations in the signals.

3.1. Lateral energy fraction (LF)

Although traditionally used to quantify spaciousness in concert hall acoustics [8, p. 351], it was shown that the lateral energy fraction could also serve as a predictor for auditory source width of loudspeaker signals in short reverberation time environments [39]. The LF describes the ratio of the lateral energy to the total energy, and is computed by contrasting the impulse responses measured by an omni-directional microphone $h_\circ$ with that of a figure-eight microphone h_∞ [40, 41]. Usually, in case of the omni-directional microphone the first 5 ms of the impulse response are omitted [42]. However, for phantom sources in the horizontal plane it was shown that an adapted version, where both impulse responses start from zero (see Equation 1), is a better predictor of auditory source width, at least for pink noise signals [39]. Although this predictor may appear to have limited potential for application in our data, we include it here for completeness.

$$LF = \frac{\int_{0\ ms}^{80\ ms} h_\infty^2 \,\mathrm{d}t}{\int_{0\ ms}^{80\ ms} h_\circ^2 \,\mathrm{d}t} \tag{1}$$

Measurements of the adapted LF for the individual loudspeakers were performed with an NTi M2210 omni-directional microphone and a Schoeps type CMC 5 with MK 8 capsule as a figure-eight microphone. Both microphones were calibrated to compensate sensitivity-differences. The center loudspeaker led to an LF of 0.07, while the LF for the outmost left/right loudspeakers was 0.13. LF for the rest of the loudspeakers were obtained by linear interpolation. As a result, the 3, 7, or 11 simultaneous loudspeakers for the small, medium, or large spatial distribution led to an overall LF of 0.08, 0.09, and 0.10, respectively. These values may be interpreted to show a monotonically increasing LF for increasing physical extent. However, their range is small compared to the literature, e.g., [43] (0.025 compared to 0.15), and even less than one just-noticeable difference (JND) [44]. As already suggested by [43], it therefore appears that the LF is not a suitable predictor for the apparent source width in our experiments, especially as it is computed from impulse responses, ignoring the effects of algorithm and stimulus type.

3.2. Interaural cross-correlation coefficient (IACC)

In previous studies, the IACC, which is the maximum of the cross-correlation between the left and right channel of a binaural recording [8], was shown to be a good predictor for perceived spatial extent in the horizontal plane [2]. To verify this claim, binaural measurements were performed with a head and torso simulator (HATS, B&K type 4128C), which was placed at the listening position. Subsequently the IACC was calculated using the recordings for all combinations of the independent variables Stimulus, Algorithm, Spatial Distribution, and Orientation. While the IACC for vertically extended sound sources was always constantly above 0.8, in horizontal orientation it varied systematically with the apparent source width (see Figure 4).

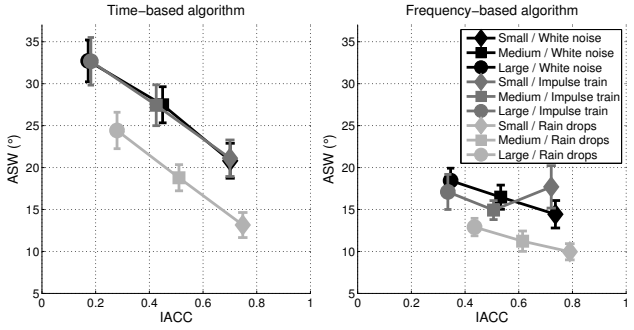


Figure 4: *Apparent source width (ASW) as a function of the IACC for the different conditions of the experiment. Error bars indicate standard error of the mean.*

It is evident in Figure 4 that the majority of the findings reported earlier can be explained on the basis of the IACC algorithms. In particular, larger spatial distributions led to a lower IACC than smaller ones, and the time-based algorithm resulted in lower IACC values and a smaller range of the values compared to the frequency-based algorithm. Furthermore, the rain drops led to always higher IACC than both white noise and impulse train stimuli which produced similar values, which explains why this was always judged to be narrower than the other two stimuli. The correlation coefficient between IACC values for each spatial distribution and the resulting perceived horizontal extent (ordinate and abscissa in Figure 4) was calculated. In the case of the time-based algorithm, a value of at least -0.996 was obtained when considering all three stimuli. In the case of the frequency based algorithm, a value of at least -0.997 was obtained for the noise and the rain stimuli, however the correlation for the impulse train was poor (0.27). On a closer inspection, this relates to an unexpected trend in the ASW value in the small spatial distribution for this condition and stimulus combination (see also the behavioral data in Figure 3).

Concluding, the IACC was found to be a good predictor for perceived spatial extent in the horizontal plane, as it is highly correlated with the absolute judgments of perceived spatial extent.

4. DISCUSSION

The results of the experiments and the acoustic measurements allow certain conclusions to be made with respect to the possibility of eliciting the perception of either vertically or horizontally extended sounds. In summary, even within the relatively small spatial distributions used in this experiment, it was possible to create the impression of horizontally extended sounds. This was not the case for vertical extent. The algorithms used here can only partially create the impression of vertical extent within the range of spatial extents used in the experiments. Finally, irrespective of orientation or stimulus type, the time-based algorithms resulted in significantly larger perceptions of both horizontal and vertical extent.

4.1. Time-based algorithm

Perceived horizontal extent created by the time-based algorithm varied systematically with the actual extent of the spatial distribution irrespective of stimulus type as evidenced by the fact that the different horizontal extents used in the study resulted in significantly different distributions of perceived spatial extent. Interestingly, actual horizontal extent was overestimated in the perceptual judgments, especially at the smaller actual spatial extents. In the vertical direction, however, perceived spatial extent varied less systematically with the actual one. Although a significant increase in the perceived vertical spatial extent with increased actual vertical spatial extent appeared for the white noise and the Dirac impulses, the difference was consistently significant only when comparing the smallest with largest displacement ($t(17)=3.46$, $p=0.003$ for white noise and $t(17)=2.43$, $p=0.047$ for impulses) and no significant differences in perceived vertical extent when using the rain drops stimulus were observed. In addition, judgments of perceived vertical extent underestimated actual extent by far pointing to limited applicability in real-world applications.

4.2. Frequency-based algorithm

Concerning the frequency-based algorithm, although in general perceived horizontal extent increased in proportion to the actual horizontal extent, as a rule judgments underestimated the actual horizontal extent of the spatial distribution and were significantly narrower than the ones obtained by the time-based algorithm. In addition, the algorithm failed to represent vertically extended sound sources. This could be attributed to the different mechanisms that operate and determine azimuth and elevation perception. While azimuth perception operates on the basis of interaural time differences, spectral cues and familiarity with source spectrum are mainly responsible for elevation perception [45, 8]. It appears therefore that while the combination of information from frequencies at different azimuths to yield the impression of coherent spatially extended auditory sources provides a functional basis for the creation of horizontally extended sources, this mechanism fails for vertically extended sources. This may be explained by the fact that presenting signal frequencies at different elevations destroys the consistency with which the signal spectrum is filtered by the outer ear to result in the perception of elevation. This is a fundamental problem when it comes to representing vertical extent by distributing signal frequencies in elevation that might be difficult to overcome.

4.3. Stimuli

Performance for the white noise and the impulse train stimulus was similar, both for horizontally and for vertically extended sources. The rain drops were perceived to be consistently narrower. In addition, although differences in spatial extent represented with this stimulus were well identified in the horizontal orientation, this was not the case in the vertical one. The aforementioned difficulties could arguably relate to the bandwidth of the rain drops stimulus, which was smaller compared to other two. The difficulties in vertical extent perception might relate to sound design issues that need to be investigated further, such as optimization of the grains to yield as good localization as possible.

4.4. Sound design

An aspect worth considering further is the overestimation of the actual spatial extent that occurred for all stimuli in the horizontal orientation in the case of the time-based and to a lesser extent in the case of the frequency-based algorithm. This may be attributed to

non-spatial factors pertaining to source-size perception, that may confound spatial extent judgments. It appears that the creation of predetermined spatial extent impressions requires the simultaneous calibration of both spatial and non-spatial factors. A general solution to provide a specific spatial extent that is applicable to all signals may therefore be difficult to achieve and perceptual calibration might be necessary in order to improve the match between actual and perceived spatial extent.

4.5. Predictors for apparent source width

The perceived spatial extent judgments in the horizontal orientation in the experiments could be explained on the basis of the IACC. LF values did not correlate with spatial extent measurements. This could be expected as room acoustics were the same throughout the experiment and the distance between active loudspeakers in the experiment was small. Listeners responded therefore on the basis of IACC and not LF. LF may be interpreted to indicate the contribution of the interaction between loudspeaker positioning and room on the judgments of spatial extent. The observed values show that this contribution is negligible.

4.6. Loudspeaker array design

In the experiment, adjacent loudspeakers were used to create the impression of spatially extended sources. This may appear uneconomical as an auditory source width of at least 10 degrees for a single loudspeaker emitting noise, depending on room acoustics and loudspeaker model has been observed [39]. Furthermore, gaps of up to 15 degrees in noise emitting loudspeaker arrays were found to be difficult to notice [10, 13]. The loudspeakers we have used, however, were smaller than the ones used in the aforementioned studies. We opted out from introducing gaps in the distribution in order to exclude the possibility of perceptual discontinuities in the perceived auditory event. It may however well be that the results of this experiment could be replicated with even less loudspeakers than used here, given appropriate calibration.

It may be worth noting that the difficulties with vertical perception in the experiment may originate in the small range of spatial extents used. It would therefore be interesting to replicate this study using larger spatial distributions in vertical orientation in order to understand whether the limitations observed here reflect a limitation in the algorithms used or a limitation in the perception of vertical extent in the auditory system. Larger spatial distributions and listener training may be interesting factors to vary in future experiments targeting this aspect.

4.7. Applications

Concerning the auditory representation of horizontal extent the results are very promising. Participants could differentiate well even in response to the small spatial distributions tested here. Designers may therefore start to integrate horizontally extended sounds in virtual and mixed reality applications. It also appears that musical compositions in which the spatial extent of sounds is explicitly manipulated will become commonplace in the future. The results of this study show that when extents of small magnitude need to be used, time-based extent synthesis algorithms are preferable, as they yield larger impressions of horizontal extent. The use of granular synthesis in this context appears to be a far reaching solution for sound and interface designers.

5. CONCLUSION

We presented a study that investigated the perception of auditory spatial extent using two spatial extent synthesis algorithms. The algorithms aimed to create the impression of spatially extended objects by either distributing the frequencies or the grains comprising a sound source in space. In a controlled experiment, the ability of the algorithms to create spatially extended sound sources as a function of Spatial Distribution, Orientation, and Stimulus type was tested. It was found that while both algorithms were successful in generating the impression of horizontally extended sound sources, the time-based algorithm resulted in broader perceptions of spatial extent irrespective of Stimulus, Orientation, or actual Spatial Distribution. Furthermore, for similar spatial distributions, judgments of horizontal extent were significantly larger than these of vertical extent. Finally, judgments of horizontal extent overestimated the physical extent, while judgments of vertical extent underestimated the physical extent. Results could be explained on the basis of measurements of the interaural cross-correlation in the different conditions in the experiment.

6. REFERENCES

[1] Jens Ahrens and Sascha Spors, "Two physical models for spatially extended virtual sound sources," in *Audio Engineering Society Convention 131*, Oct 2011.

[2] R. Mason, T. Brookes, and F. Rumsey, "Frequency dependency of the relationship between perceived auditory source width and the interaural cross-correlation coefficient for time-invariant stimuli," *J. Acoust. Soc. Am.*, vol. 117, no. 3 Pt 1, pp. 1337–1350, Mar 2005.

[3] D. R. Perrott and T. N. Buell, "Judgments of sound volume: effects of signal duration, level, and interaural characteristics on the perceived extensity of broadband noise," *J. Acoust. Soc. Am.*, vol. 72, no. 5, pp. 1413–1417, Nov 1982.

[4] Densil Cabrera and Steven Tilley, "Parameters for auditory display of height and size," in *Proceedings of the 9th International Conference on Auditory Display (ICAD)*, Eoin Brazil and Barbara Shinn-Cunningham, Eds., Boston, MA, July 2003, International Community for Auditory Display, Georgia Institute of Technology.

[5] A. J. Kunkler-Peck and M. T. Turvey, "Hearing shape," *J Exp Psychol Hum Percept Perform*, vol. 26, no. 1, pp. 279–294, Feb 2000.

[6] Claudia Carello, Krista L. Anderson, and Andrew J. Kunkler-Peck, "Perception of object length by sound," *Psychological Science*, vol. 9, no. 3, pp. 211–214, May 1998.

[7] Mark Kac, "Can one hear the shape of a drum?," *American Mathematical Monthly*, vol. 73, no. 4/2, pp. 1–23, Apr. 1966.

[8] Jens Blauert, *Spatial Hearing: The Psychophysics of Human Sound Localization*, MIT Press, revised edition, 1997.

[9] J. Blauert and W. Lindemann, "Spatial mapping of intracranial auditory events for various degrees of interaural coherence," *The Journal of the Acoustical Society of America*, vol. 79, no. 3, pp. 806–813, 1986.

[10] Toni Hirvonen and Ville Pulkki, "Center and spatial extent of auditory events as caused by multiple sound sources in frequency-dependent directions," *Acta Acoustica united with Acoustica*, vol. 92, pp. 320–330, 2006.

[11] Olli Santala and Ville Pulkki, "Directional perception of distributed sound sources," *The Journal of the Acoustical Society of America*, vol. 129, no. 3, pp. 1522–1530, 2011.

[12] Olli Santala and Ville Pulkki, "Resolution of spatial distribution perception with distributed sound source in anechoic conditions," in *Audio Engineering Society Convention 126*, May 2009.

[13] Toni Hirvonen and Ville Pulkki, "Perceived spatial distribution and width of horizontal ensemble of independent noise signals as function of waveform and sample length," in *Audio Engineering Society Convention 124*, May 2008.

[14] Koichiro Hiyama, Setsu Komiyama, and Kimio Hamasaki, "The minimum number of loudspeakers and its arrangement for reproducing the spatial impression of diffuse sound field," in *Audio Engineering Society Convention 113*, Oct 2002.

[15] Mikko-Ville Laitinen, Tapani Pihlajamäki, Cumhur Erkut, and Ville Pulkki, "Parametric time-frequency representation of spatial sound in virtual worlds," *ACM Trans. Appl. Percept.*, vol. 9, no. 2, pp. 8:1–8:20, June 2012.

[16] Tapani Pihlajamaki, Olli Santala, and Ville Pulkki, "Synthesis of spatially extended virtual source with time-frequency decomposition of mono signals," *J. Audio Eng. Soc*, vol. 62, no. 7/8, pp. 467–484, 2014.

[17] Franz Zotter, Matthias Frank, Georgios Marentakis, and Alois Sontacchi, "Phantom source widening with deterministic frequency dependent time delays," in *Proc. of the 14th International Conference on Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011, pp. 307–312.

[18] B. C. J. Moore and B. R. Glasberg, "A revision of Zwicker's loudness model," *Acustica United with Acta Acustica*, vol. 82, no. 2, pp. 335–345, 1996.

[19] Toni Hirvonen and Ville Pulkki, "Perception and analysis of selected auditory events with frequency-dependent directions," *J. Audio Eng. Soc*, vol. 54, no. 9, pp. 803–814, 2006.

[20] J. H. Halton, "Algorithm 247: Radical-inverse quasi-random point sequence," *Commun. ACM*, vol. 7, no. 12, pp. 701–702, Dec. 1964.

[21] Michael A. Gerzon, "Signal processing for simulating realistic stereo images," in *Audio Engineering Society Convention 93*, Oct 1992.

[22] Barry Truax, "Composition and diffusion: space in sound in space," *Organised Sound*, vol. 3, pp. 141–146, 8 1998.

[23] Natasha Barrett, "Spatio-musical composition strategies," *Organised Sound*, vol. 7, pp. 313–323, 12 2002.

[24] Etienne Deleflie and Greg Schiemer, "Spatial grains: Imbuing granular particles with spatial-domain information," in *Proceedings of the Australasian Computer Music Conference ACMC09*, July 2009.

[25] Curtis Roads, *Microsound*, The MIT Press, 2004.

[26] Rory Wallis and Hyunkook Lee, "The effect of interchannel time difference on localization in vertical stereophony," *J. Audio Eng. Soc.*, vol. 63, no. 10, pp. 767–776, October 2015.

[27] Hyunkook Lee, "Investigation on the phantom image elevation effect," in *139th Audio Engineering Society Convention*, October 2015, This is Author Accepted Manuscript for Green Open Access.

[28] Ville Pulkki, "Localization of amplitude-panned virtual sources ii: Two- and three-dimensional panning," *J. Audio Eng. Soc*, vol. 49, no. 9, pp. 753–767, 2001.

[29] James L. Barbour, "Elevation perception: Phantom images in the vertical hemi-sphere," in *Audio Engineering Society Conference: 24th International Conference: Multichannel Audio, The New Reality*, Jun 2003.

[30] B. R. Glasberg and B. C. J. Moore, "Derivation of auditory filter shapes from notched-noise data," *Hearing Research*, vol. 47, no. 1-2, pp. 103–138, 1990.

[31] D. Rocchesso, *Introduction to Sound Processing*, Mondo estremo, 2003.

[32] Sebastian Blamberger, "Development of a modular system for speaker array prototyping," Tech. Rep., Institute of Electronic Music and Acoustics, Graz University of Music and Performing Arts, 2011.

[33] P. Zahorik, "Direct-to-reverberant energy ratio sensitivity," *J. Acoust. Soc. Am.*, vol. 112, no. 5 Pt 1, pp. 2110–2117, Nov 2002.

[34] Peter Venus, Marian Weger, Cyrille Henry, and Winfried Ritsch, "Extended view toolkit," in *Proceedings of the 4th Pure Data Convention*, 2011, pp. 161–167.

[35] Nelson Cowan, "On short and long auditory stores," *Psychological Bulletin*, vol. 96, no. 2, pp. 341–370, Sep 1984.

[36] Hubert W. Lilliefors, "On the kolmogorov-smirnov test for normality with mean and variance unknown," *Journal of the American Statistical Association*, vol. 62, no. 318, pp. 399–402, 1967.

[37] Frank E. Grubbs, "Sample criteria for testing outlying observations," *Ann. Math. Statist.*, vol. 21, no. 1, pp. 27–58, 03 1950.

[38] Johannes Käsbach, Marton Marschall, Bastian Epp, and Torsten Dau, "The relation between perceived apparent source width and interaural cross-correlation in sound reproduction spaces with low reverberation," in *Proceedings of DAGA 2013*, 2013.

[39] Matthias Frank, "Source width of frontal phantom sources: Perception, measurement, and modeling," *Archives of Acoustics*, vol. 38, no. 3, pp. 311–319, 10 2013.

[40] Barron M and Marshall AH, "Spatial impression due to early lateral reflections in concert halls: the derivation of a physical measure," *Journal of Sound & Vibration*, vol. 77, no. 2, pp. 211–232, 1981.

[41] Trevor J. Cox, W. J. Davies, and Yiu W. Lam, "The Sensitivity of Listeners to Early Sound Field Changes in Auditoria," *Acustica*, vol. 79, no. 1, pp. 27–41, 1993.

[42] ISO, "3382-1:2009: Acoustics - measurement of room acoustic parameters - part 1: Performance spaces.," 2009.

[43] Matthias Blau, "Correlation of apparent source width with objective measures in synthetic sound fields," *Acta Acustica united with Acustica*, vol. 90, no. 4, pp. 720–730, 2004-07-01T00:00:00.

[44] Matthias Blau, "Difference limens for measures of apparent source width," in *Forum Acusticum*, Sevilla, Spain, 2002.

[45] F. Wightman and D. Kistler, "Factors affecting the relative salience of sound localization cues," *Binaural and spatial hearing in real and virtual environments*, vol. 1, pp. 1–23, 1997.

MODEL-BASED OBSTACLE SONIFICATION FOR THE NAVIGATION OF VISUALLY IMPAIRED PERSONS

Simone Spagnol, Omar I. Johannesson, Arni Kristjansson, Runar Unnthorsson

University of Iceland,
Reykjavík, Iceland
{spagnols,omarjo,ak,runson}@hi.is

Charalampos Saitis, Kyriaki Kalimeri

ISI Foundation,
Turin, Italy
{charalampos.saitis,kyriaki.kalimeri}@isi.it

Michał Bujacz

Lodz University of Technology,
Łódź, Poland
michal.bujacz@p.lodz.pl

Alin Moldoveanu

University Politehnica of Bucharest
Bucharest, Romania
alin.moldoveanu@cs.pub.ro

ABSTRACT

This paper proposes a sonification model for encoding visual 3D information into sounds, inspired by the impact properties of the objects encountered during blind navigation. The proposed model is compared against two sonification models developed for orientation and mobility, chosen based on their common technical requirements. An extensive validation of the proposed model is reported; five legally blind and five normally sighted participants evaluated the proposed model as compared to the two competitive models on a simplified experimental navigation scenario. The evaluation addressed not only the accuracy of the responses in terms of psychophysical measurements but also the cognitive load and emotional stress of the participants by means of biophysiological signals and evaluation questionnaires. Results show that the proposed impact sound model adequately conveys the relevant information to the participants with low cognitive load, following a short training session.

1. INTRODUCTION

In audio-based software applications, such as auditory displays and audio games, sonification is used to represent various actions, objects or situations in order to virtually describe scenes. *Sonification* can be defined as "a mapping of numerically represented relations in some domain under study to relations in an acoustic domain for the purposes of interpreting, understanding, or communicating relations in the domain under study" [1].

Sonification is also used in health care, for instance in motor rehabilitation systems [2], electronic travel aids (ETAs, i.e., devices which aid in independent mobility through obstacle detection or help in orientation and navigation) [3], and other assistive technologies for visually impaired persons (VIPs). Most of these systems are still in their infancy and mostly still at a prototype stage. Furthermore, available commercial products have limited functionalities, small scientific/technological value and high cost [3].

Available ETAs for VIPs provide various information that ranges from simple obstacle detection with a single range-finding sensor, to more advanced feedback employing data generated from visual representations of the scenes, acquired through camera technologies. The auditory outputs of such systems range from simple binary alerts indicating the presence of an obstacle in the range of a sensor, to complex sound patterns carrying almost as much information as a graphical image [4]. Finding the most suitable accuracy/simplicity trade-off in order to provide valuable information about the environment surrounding the user through sound is therefore a pivotal and challenging task.

This study aims to explore a novel scheme for translating 3D representations of a scene or an environment, represented as a list of objects with properties, into auditory feedback. The remainder of the paper is organized as follows. Section 2 introduces the sound model as well as two alternative models inspired by previous literature. Section 3 describes an experiment targeted at comparing the performance of the three models in a navigation task, through both psychophysical and psychophysiological measurements. Section 4 reports the results of the experiment, and Section 5 concludes the paper.

2. MODEL-BASED OBSTACLE SONIFICATION

Different sonification approaches for representing visual scenes to blind users have previously been studied. The most common natural mappings between object and sound properties are related to the spatial position of the object; the most recurring are

- azimuth → stereo panning / Head-Related Transfer Function (HRTF) filtering [5, 6, 7];
- elevation → HRTF filtering [6] / pitch [8];
- distance → amplitude [5, 9] / pitch [5, 9].

This Section provides details on a sonification model designed by the authors with the help of blind volunteers and specialists in training and rehabilitation of VIPs. Mappings within the model were both inspired by the previous literature shown above and original design. Parameter tuning was refined following a preliminary investigation using psychophysical evaluation methods only [10].

2.1. Sonification through impact sounds

The model we propose treats each object in the frontal hemisphere of the user as an independent virtual sound source that continuously emits impact sounds, as if the VIP was hitting it with a white cane. The pitch and timbre of the sound resulting from the impact are considered dependent on the object's width and category. The distance between object and user is coded into loudness: the closer the object, the higher the sound level. Furthermore, each

sound is spatialized in accordance with the direction of the object with respect to the user.

Single impact sounds are generated through a physical model of non-linear impact between two modal objects. This model is part of a number of sound models included in the Sound Design Toolkit (SDT),[1] an open-source (GPLv2) software package suitable for research and education in Sonic Interaction Design [11]. The SDT consists of a library of physics-based sound synthesis algorithms, available as externals and patches for Max and Pure Data.[2] The Pure Data version was used in the development of this model.

The physical model receives as input parameters related to the striking object (modal object 1) and the struck object (modal object 2), as well as the interaction between the two. The most relevant fixed parameters are strike velocity, set to 1.85 m/s, and striker mass, set to 0.6 kg. These were considered as reasonable parameters for a long white cane and the act of striking with it. Parameters of the struck object, i.e., the object that needs to be sonically represented, change with respect to the width and category of the object itself.

In particular, width is directly mapped to the frequency f of the single mode of the struck object. In order to maximize the available frequency range, this was chosen to vary from values as low as 50 Hz (very wide objects such as walls) to 4 kHz (20-cm narrow objects) according to the following mapping,

$$f = \frac{840}{w} \quad [Hz] \tag{1}$$

where w is the actual width of the object in meters.

Different categories of objects are on the other hand represented by different decay times of the frequency mode. Categorization of objects may follow different rules, e.g. be based on object material (with rubber, wood, glass and steel having increasing decay times [12]) or object type (simple objects, walls, poles or trees, holes or ponds, and so on). Having defined category $C = 1, 2, 3, \ldots$, the mapping to the corresponding acoustic parameter, i.e. decay time t_d, is

$$t_d = 0.02C \quad [s] \tag{2}$$

heuristically set in order to enable association to impacts on different materials [12]. Default parameters were used for the test reported in this paper, with category 1 assigned to wall sounds and category 5 assigned to wall edges (replacing two adjacent wall sounds).

The absolute distance r between the subject and the object is also considered as a parameter. Assuming all obstacles to be sonified lying further than 1 m (closer objects are in the reach of the white cane), thus in the subject's acoustic far field [13], this is directly mapped into the amplitude of the sound by following the classic $1/r$ pressure attenuation law [14]. The overall number n of objects present in the scene influences the repetition rate of the impact sound instead: the period T between two consecutive impacts on the same object is set to

$$T = 0.2(n - 1) \quad [s]. \tag{3}$$

The point associated to the object is either the estimated barycenter in the case of small objects, or the intersection between the closest surface and its normal vector crossing the observer in the case of bigger objects, such as walls.

Last but not least, the direction of the object with respect to the observer taken in angular coordinates (azimuth, elevation) is directly mapped to the corresponding parameters of a generic HRTF filter provided through the `earplug~` Pure Data binaural synthesis external. In particular, the filter renders the angular position of the sound source relative to the subject by convolving the incoming signal with left and right HRTFs from the MIT KEMAR database [15].[3] This way, the sound is spatialized along the actual direction of the object. It has to be highlighted that spatialization is non-individual; however, models for HRTF individualization [16, 17] or individual HRTFs themselves can be integrated (at an additional measurement cost) if higher spatial accuracy is needed [18].

There were two reasons for choosing impact sounds to convey information about objects. First, the ecological validity of physics-based sounds, whose nature allows a direct association to the virtual act of detecting the object by striking it with a cane. Second, the peculiar pattern of impact sounds, whose rich frequency content and short duration of the attack phase allow for improved sound localization on the horizontal plane [19]. Furthermore, choices about the mappings between object and sound properties were either adopted from previous literature (distance and direction) or based on the nature of the impact model. Actually, the association of higher pitches to smaller objects and different decay times to different categories, e.g. materials, has physical ground [12].

The model was implemented as three Pure Data patches. Both static scenes and simple dynamic scenes with a fixed number of objects are supported. The main Pure Data patch receives as input a text file containing one row per object present in the scene. Each row includes information about the object ID, azimuth angle (between -90 and 90 degrees [15]), elevation angle (between -40 and 90 degrees [15]), distance (above 1 m), width (above 20 cm), object category ($1, 2, 3, \ldots$), and mode (static $= 0$, dynamic $= 1$), separated by spaces.

At the beginning, sources are ordered by increasing azimuth, left to right. In order to avoid simultaneous impacts, the first impact on a given object is played 200 ms after the impact on the object on its immediate left. In the case of a dynamic scene, impacts corresponding to an object stop as soon as the object is behind the listener (i.e., outside the $[-90, 90]$ degree azimuth range).

2.2. Alternative sonification approaches

In the round of testing reported in this paper, the proposed model is compared against two other competing models developed in previous literature in order to solve the same problem. These two alternative approaches are now briefly described.

2.2.1. Depth scanning

The depth scanning model is a sonification method used in Bujacz *et al.* [20]. The main inspiration for the model was the fact that blind persons, especially those blind from birth, have a path-based perception of their environment [21]. The core concept of the method is a virtual scanning plane, i.e., a surface parallel to the observer's frontal plane that moves away from him/her through the scene. As the surface intersects scene elements, sounds originating

[1] `http://soundobject.org/SDT/`

[2] `https://puredata.info/`

[3] `http://sound.media.mit.edu/resources/KEMAR.html`

from the points of intersection are released. The scanning surface moves for 5 m in 1.5 s, then after a 0.5 s pause it restarts from the observer. This was the default speed chosen by the majority of the blind participants in previous prototype trials [20]. However, in the experiment reported in this paper the scanning surface was slightly sped up to fit 3 cycles into 5-second test samples. Furthermore, reference "tick" sounds are played each time the scanning plane moves 1 m away.

Sounds are designed to naturally correspond to object parameters. Distance, as the most important parameter, is encoded redundantly into the temporal delay inside each cycle as well as into the loudness and pitch of the sound. For instance, if a distant object appears later in a scanning cycle, its sound will be less loud and have lower pitch. The location of an object is encoded via HRTFs and its size through sound duration. The sound coder uses audio files pre-generated with a Microsoft General MIDI calliope synthesizer (no. 83) modulated with 5% noise (14 dB SNR), as previous trials showed that the addition of noise improves spatial localization of a sound [22]. The sounds were stored in collections of 5-s wave files of full tones from the diatonic scale (octaves 2 to 4). Sounds were spatially filtered using the MIT KEMAR generic HRTFs and modulated with a simple ADSR envelope.

2.2.2. Horizontal sweep

The horizontal sweep approach was used in previous sonification studies (e.g. Navbelt [5]), and is sometimes referred to as the "piano scan". It basically translates the distance to pitch in several directions from the observer. The sonification approach is very similar to the one previously described for depth scanning with the main difference being that of the scanning plane; instead of moving away from the frontal plane, it swings left to right around a vertical axis passing through the observer. The scan sweeps from -45 to 45 degrees in 1.5 s. Reference "tick" sounds are played each time the scanning plane moves by 15 degrees.

This model generates sounds from scratch using a simple Moog synthesizer and an ADSR envelope. Pitches are selected from the middle three octaves of the pentatonic scale. A difference with respect to the depth scan approach is that instead of smoothly moving sound sources along the intersection of the sweeping plane and walls, the scene was divided into discrete regions 15 degrees wide (according to the previously set reference ticks), and for each region a sound was produced corresponding to the nearest object.

3. MATERIALS AND METHODS

An experiment was designed where the above described sonification approaches were compared using methods from the fields of behavioural psychology and psychophysiology, namely response time and accuracy, electroencephalography (EEG), and monitoring of electrodermal activity (EDA). The goal was to explore various alternatives in rendering basic 3D visual scenes through sound signals to be delivered to VIPs, through assessing both functionality (psychophysics) and cognitive performance (psychophysiology). This study was accepted by the National Bioethical Committee of Iceland, with reference number VSN-15-107.

3.1. Participants

Five VIPs and five sighted students from the University of Iceland participated in the experiment (6 female; average age = 34 yrs, range = $21 - 52$ yrs) on voluntary basis. One VIP was fully blind, two had vision less than 5%, and two had vision between 5% and 10%. Three of them were congenitally or early blind (first $2-3$ yrs of life) and two had become blind later in life (generally after the age of 3). All participants spoke English fluently and reported having no hearing impairment as well as no general health issues. Two VIPs mentioned having some experience, one reported substantial experience, and two said that they are very experienced with IT technology. All participants gave free and informed consent.

3.2. Psychophysiological approach

Electrodermal activity is a well-known indicator of physiological arousal and stress activation [23]. EDA is more sensitive to emotion related variations in arousal as opposed to physical stressors, which can be better reflected in measurements of cardiovascular activity such as heart rate. Electroencephalography, on the other hand, can provide neurophysiological markers of cognitive and emotional processes induced by stress and indicated by changes in brain rhythmic activity [24]. Taking advantage of their inherent and complementary properties, EEG and EDA signals were collected and analysed concurrently with the more traditional behavioural measures of response time and accuracy.

A measurement of EDA is characterized by two types of behaviour: short-lasting phasic responses (which can be thought of as rapidly changing peaks) and a long-term tonic level (which can be thought of as the underlying slow-changing level in the absence of phasic activity) [23]. Phasic responses are primarily elicited by specific external stimuli, and are typically observed superposed in states of high arousal or short interstimulus interval paradigms such as those employed in cognitive research.

EDA was registered with the Empatica E4 wristband [25], which measures skin conductance through two ventral (inner) wrist electrodes ($f_s = 4$ Hz). Signals were analysed with Ledalab, a Matlab-based toolbox.[4] Ledalab implements a signal decomposition method based on standard deconvolution, which results in one single continuous measure of phasic activity. Time-integration over a specified window after the stimulus onset yields a simple and unbiased (i.e., avoiding biases due to superposing peaks) indicator of phasic EDA, namely integrated skin conductance response (ISCR) [26]. ISCR can be thought of as the cumulative phasic activity within the specified response time period. Our hypothesis was that a pleasant, easy to understand, and less stressful sonification mapping will generally elicit lower phasic activity as indexed by ISCR.

Brain activity is characterized by rhythmic patterns (waves) across distinct frequency bands, the definition of which can vary among studies. Here we analysed EEG in five bands, namely theta (4–7 Hz), alpha-1 (7.5–10 Hz), alpha-2 (10–12.5 Hz), beta (13–30 Hz), and gamma (30–60 Hz). Beta activity is associated with psychological and physical stress, whereas theta and alpha-1 frequencies reflect response inhibition and attentional demands such as phasic alertness [27]. Alpha-2 is related to task performance in terms of speed, relevance, and difficulty [24]. Gamma waves are involved in more complex cognitive functions such as multimodal processing or object representation [28].

EEG was recorded using the Emotiv EPOC+, a wireless headset with 14 passive electrodes (channels) registering over the 10-20 system locations AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, and FC4 (sampling rate $f_s = 128$ Hz) [29]. For each

[4] `http://www.ledalab.de`

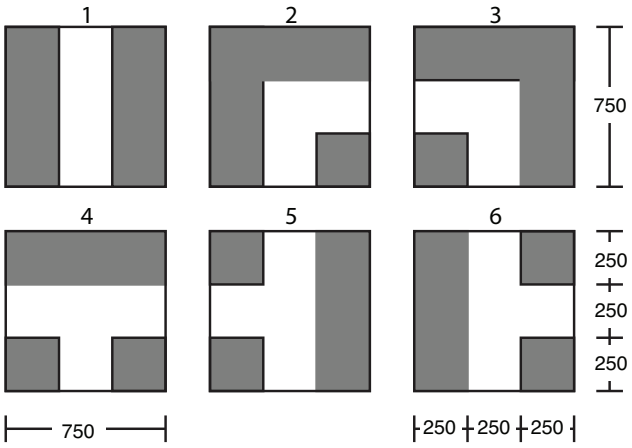


Figure 1: The six modules used in the experiment. The participant always entered each model from the bottom. White areas represent the virtual path and grey areas represent obstacles (delimited by walls). All measures are in cm.

channel we studied the relative spectral power in each of the bands described above over a specified time window following the stimulus onset using the Fourier transform. The total power across all bands (4–60 Hz) was also examined. Analyses were focused on a subset of 6 channels, namely AF3, F7, T7, T8, F8, and FC4, which are often considered suitable enough to monitor brain activity under emotional stress [30]. Our hypothesis was that a less mentally demanding and stressful sonification model will generally involve lower power across the whole EEG spectrum, smaller theta activity and larger alpha-2 power.

3.3. Setup and procedure

Participants of the virtual navigation task were instructed to use their dominant hand to respond using the arrow keys of a keyboard: left arrow for turning left, up arrow for walking straight and right arrow for turning right. The down arrow was removed from the keyboard during the experiment to help avoid potential confusion. All participants were blindfolded, and sound stimuli were delivered through a pair of in-ear headphones.[5] Information and feedback concerning the progress of the experiment were also presented to the participant through prerecorded audio files.

A virtual walk was designed, comprising 6 different scenes (modules) each representing a different configuration of a free path between walls. The 6 configurations can be seen in Fig. 1: the gray blocks depict walls while the walkable path is in white. Auditory representations of each configuration were created from each of the tested sonification models (Model 1: impact model; Model 2: horizontal sweep; Model 3: depth scanning).

Initially there was a training phase, where in order for the subject to comprehend the rationale of each sound model, a 3D representation of the 6 modules was created using Lego blocks. While touching each block, the participant listened to the corresponding stimulus along with prerecorded explanations of the task. This procedure was repeated two times for each of the sound models.

Subsequently, the physiological sensors were placed. Participants were asked to find a comfortable position and to avoid any unnecessary movement. EDA was recorded from the non-dominant hand (wrist) of the participants to minimize motion artifacts largely due to pressing response buttons [23]. EEG was recorded continuously from the 14 scalp electrodes of the EPOC headset. Signals were transmitted from the headset via a USB wireless receiver to proprietary Emotiv software running on a laptop.

The next three phases were repeated once for each of the three tested sound models, whose order during the test was randomized in order to avoid any bias.

3.3.1. Training

Upon setting up the sensors, a short training session started wherein each module was presented to the participant four times in random order. Responses were recorded but only used to provide feedback to the participant after each response and to calculate their accuracy. If less than 75% of the given responses were correct, the training session was repeated but never more than two times. Right after the training session, the participant was asked to relax completely for 300 s in order to record spontaneous resting state physiological activity.

3.3.2. Testing

During the virtual walk, the participant always entered each module from the bottom (see Fig. 1). The virtual walking speed was chosen to be 1 m/sec. For each module the participant had to make a decision no later than 5 m (5 s) after entering the module: whether to turn (and into which direction) or to continue straight ahead. Participants were instructed to respond as fast and as accurately as possible. Each module was presented 15 times in random order. One full virtual walk lasted approximately 6 to 10 minutes.

If the participant did not respond within the time limit (i.e., after 5 s), or if his decision was incorrect, the virtual walk was stopped and a short sound indicating an error was played. After 0.5 seconds the virtual walk would start again. In the case of registering a correct response, the model stimulus was stopped and the participant instantly moved to the next module where the same procedure was repeated.

Upon completion of testing a model, participants were asked to relax completely for 120 s while their spontaneous physiological activity was being registered.

3.3.3. Questionnaire

As soon as the resting period ended, the participant was asked to evaluate the model on five 5-point Likert scales:

- Q1 - I found the sounds pleasant to the ear.
- Q2 - I could imagine the sounds originating from the environment (as opposed to originating inside my head).
- Q3 - I found it easy to understand what each sound represents.
- Q4 - I found the task stressful (the sounds were too fast to understand).
- Q5 - I think that, with sufficient training, I would understand what each sound represents at even faster rates.

Subjects were also asked to freely comment on the functionality and pleasantness of the sound stimuli. Verbal responses were recorded.

[5] `https://earhero.com`

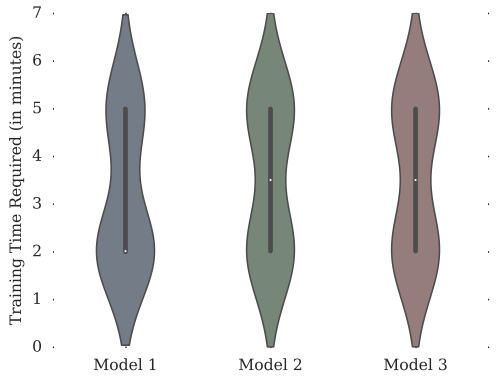


Figure 2: Violin plots of the probability density distribution of learning time required by subjects. The white dot represents the median value.

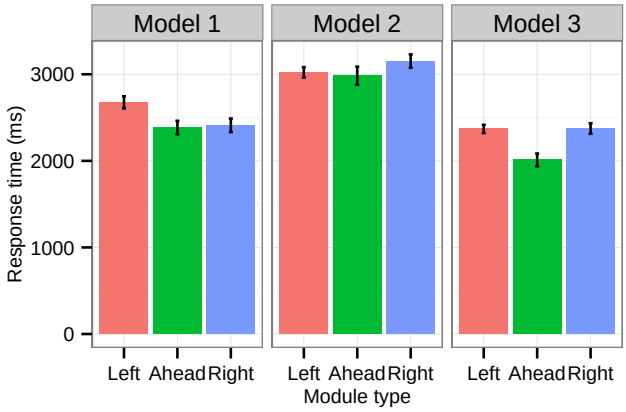


Figure 3: Average RTs by model and response type. Error bars represent the within subjects standard error.

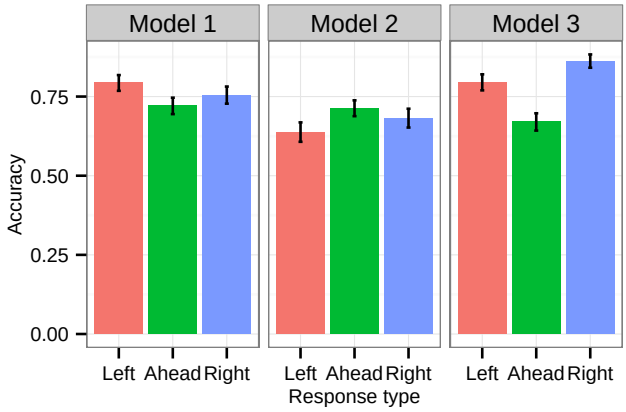


Figure 4: Average accuracy by model and response type. Error bars represent the within subjects standard error.

Once all models were tested, the participant was asked to freely respond to the following question: "Which of the three sonification approaches did you prefer? Can you explain why?". Verbal responses were again recorded. Finally, the EPOC and E4 sensors were detached from the participant. During the experimental session, the experimenter sat outside of the room and monitored the stimulus presentation and the recorded physiological data. The experiment lasted approximately 90 minutes in total.

4. RESULTS

4.1. Model learnability

As described in the previous Section, the experiment involved a short training session. Using response times from the training sessions, we looked at the time required from all participants to "understand" each model (i.e., how the respective sounds mapped to the different modules of the virtual walk task).

Figure 2 depicts the probability density of the learning interval required for each model. From this representation, Model 1 outperforms the other two, since it is the only one that required a median learning interval equal to two minutes. Note that since the training section was not repeated more than two times, the difference between Model 1 and Models 2 and 3 is relevant, even if small.

4.2. Response time and accuracy

The average response time and accuracy were computed for each model and compared using repeated measures ANOVA. Notice from Fig. 1 that modules 1 to 3 had only one correct response out of three (left, ahead, right), while modules 4 to 6 had two correct responses out of three. This means that random responses would lead to an average accuracy of 0.33 and 0.67 in the long run, respectively. Therefore, the average random accuracy in the experiment is 0.5. A response is considered correct only when given within the maximum response time of 5 s.

The average response time (RT) was 2441 ms (SD = 991 ms) for the visually impaired group and 2780 ms (SD = 1059 ms) for the sighted group. The difference between these groups was not significant [$t(7.57) = 1.11$, $p = 0.303$]. The average response time divided by model and response type (left, ahead, right) is reported in Fig. 3. A two-way repeated measures ANOVA with model and response type as factors revealed a significant main effect of response type [$F(2, 18) = 6.08$, $p = 0.010$] but not of model [$F(2, 18) = 2.65$, $p = 0.098$] and significant interaction between models and response type [$F(4, 36) = 2.96$, $p = 0.033$].

Average accuracy was 0.73 (SD = 0.44) and is significantly different from random [paired-t(9) = 6.44, $p < 0.001$]. The average accuracy was 0.73 (SD = 0.45) for the visually impaired group and 0.74 (SD = 0.44) for the sighted group. This very small difference was not significant [$t(4) = 0.51$, $p = 0.638$]. The average accuracy divided by model and response type is reported in Fig. 4. A two-way repeated measures ANOVA with model and response type as factors revealed a significant main effect of response type [$F(2, 18) = 6.71$, $p = 0.007$] but not of model [$F(2, 18) = 2.24$, $p = 0.135$], and the interaction was not significant [$F(4, 36) = 0.87$, $p = 0.491$]. Accuracy was therefore comparable across models.

4.3. Phasic electrodermal response

Prior to analysis, skin conductance data obtained from the E4 EDA sensor were filtered with a first-order Butterworth low-pass filter

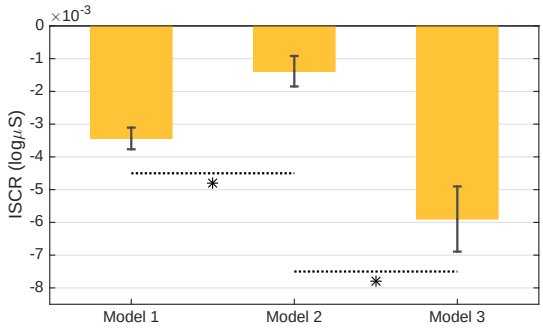


Figure 5: Average cumulative phasic activity for each of the tested models. Error bars represent the standard error of the mean. Star signs (*) indicate statistically significant differences.

using a cutoff frequency of 1 Hz to remove steep peaks stemming from artifacts such as pressure exerted on the electrodes [23]. The filtered time series were subsequently analyzed with Ledalab using the continuous decomposition method (see Section 3.2). As sound stimuli were presented as soon as the response to the previous module was registered (i.e., variable interstimulus intervals), a variable response window was considered, starting at 1 s after stimulus onset and ending at 4 s after stimulus offset. Integrals of the continuous phasic driver within a specified response time period were normalized by means of dividing by the duration of the respective window. To reduce inter-individual variation prior to averaging, means were subtracted from the trial-by-trial ISCR values. The resulted data were further transformed using the formula $y = log(1 + x)$ to improve distributional characteristics.

Figure 5 depicts the average ISCR for each model computed across all participants and all stimuli. It can be immediately observed that phasic electrodermal activity was substantially higher in Model 2 and lower in Model 3. To test whether these differences were statistically significant, a repeated measures ANOVA with models as factor was run. This analysis revealed a significant difference in physiological arousal between the three sonification alternatives [$F(2) = 10.6$, $p = 0.02$, using the Greenhouse-Geisser correction for sphericity]. Pairwise comparison of the means using Bonferroni post hoc tests showed that Model 2 is significantly different from Models 1 and 3 [$p = 0.04$ and $p = 0.05$, respectively], whereas the observed difference between Models 1 and 3 is marginally not significant [$p = 0.06$].

4.4. EEG power spectra

The Emotiv EPOC+ system involves a number of internal signal conditioning steps. Analogue signals are first high-pass filtered with a 0.16 Hz cut-off, pre-amplified, low-pass filtered with a 83 Hz cut-off, and sampled at 2048 Hz. Digital signals are then notch-filtered at 50/60 Hz and down-sampled to 128 Hz prior to transmission. Prior to analysis, the EEG data obtained from the headset was baseline-normalized by subtracting for each participant and for each channel the mean of the resting state registrations.

As described in the previous section, sound stimuli were presented as soon as the response to the previous module was registered. Considering the shortest interstimulus interval (3.4 s), EEG epochs lasting 3 s after stimulus onset were extracted for each model-module condition, resulting in a total of 2700 epochs per

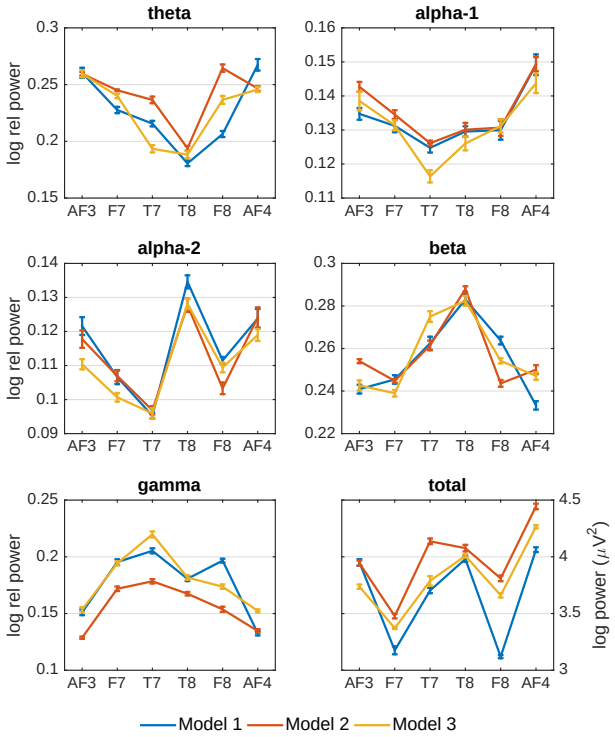


Figure 6: Mean relative and total band power for each of the tested models in 6 frontal electrode positions. Error bars represent the standard error of the mean.

EEG channel. A Hann window was applied to each epoch to minimize spectral leakage, and then the Fourier transform of the windowed data was used to calculate power spectral density estimates in the theta (4–7 Hz), alpha-1 (7.5–10 Hz), alpha-2 (10–12.5 Hz), beta (13–30 Hz), and gamma (30–60 Hz) bands as well as across the total 4–60 Hz range. Individual band power estimates were normalized by means of dividing by the across-band power. Before averaging, a logarithmic transformation $y = log(1 + x)$ of single-trial values was applied to improve their distributional characteristics.

Figure 6 shows the average band power in the AF3, F7, T7, T8, F8, and FC4 channels for each model, calculated across all participants and modules. A first look at the different plots suggests that Model 1 resulted in better cognitive performance during the experimental task than Models 2 and 3. Gamma activity, related to information representation and processing, was particularly low for Model 2, which had the largest total power. To test whether model differences were statistically significant, a repeated measures ANOVA with model as the between-subjects factor and electrode location as the within-subjects factor was run for each frequency range. Where appropriate, p-values were corrected by means of the Greenhouse-Geisser method. Bonferroni post-hoc tests were used for pairwise comparison of means.

There was a significant effect of model on total power [$F(10) = 61.14, p \ll .001$] and on relative power in each band [theta, alpha-2, beta, gamma: $F(10) \geq 6.39, p \ll .001$; alpha-1: $F(10) = 3.31, p = .0013$]. Total power for Model 1 was significantly

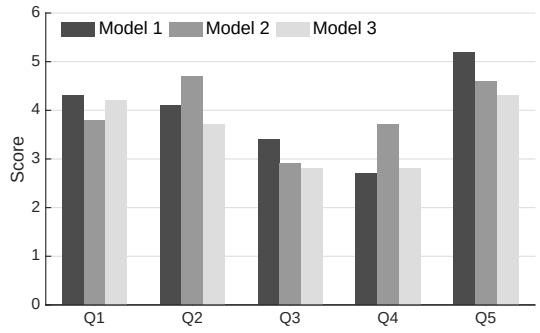


Figure 7: Average scores across the five usability scales for each of the tested models.

lower than for Models 2 and 3, and for Model 2 it was significantly higher than for Model 3 [$p \ll .001$]. This suggests that participants were more cognitively engaged when responding to Model 2 stimuli. However, gamma activity was significantly lower for Model 2 than for the other two models [$p \ll .001$] and no significant difference between Models 1 and 3 was revealed [$p = 1$]. This would imply that the larger total power for Model 2 reflected its reduced ability to convey the relevant information [28]. Model 2 further resulted in higher theta [significant effect, $p < .001$], alpha-1 [not significant, $p \geq .21$], and beta [not significant, $p \geq .62$] power than the other two models, suggesting higher response inhibition, attentional demands, and stress. Model 1 had the smallest theta [significantly so than Model 2 but not Model 3] and largest alpha-2 power [no significant differences between the three models, $p \geq .1$], suggesting better cognitive performance [24].

Analyses were repeated for all 14 channels of the Emotiv headset. The same effects and differences were observed, thus confirming that the selected 6 frontal electrode locations were suitable and sufficient to assess stress-related cognitive performance [30].

4.5. User experience questionnaire

The majority of the participants (8 out of 10) preferred Model 1 over the other two models referring to it as the easiest to use. The sounds in Model 1 were mainly described as pleasant and none of the subjects seemed to have strong negative opinions about their unpleasantness. Model 2 appeared to be the least favored one as participants reported having trouble understanding what the sounds were intended to convey. A few subjects thought the sounds were too similar to each other and had difficulty telling them apart, with some even describing them as confusing. Despite this, the majority of subjects found the model's sounds to be pleasant. A few participants had issues understanding some of the sounds in Model 3 while others described it as functional and easy to use. The sounds in this model were reported as the most annoying and/or irritating although some found them pleasant.

Complementary to the analysis of the verbal comments, the scores of each model in each of the 5 Likert-scales previously described in Section 3.3.3 were computed. Figure 7 shows that Model 1 was perceived as slightly more pleasant (Q1), easier to understand (Q3), less stressful (Q4), and easier to learn (Q5). Model 2, on the other hand, was characterized as the most natural sounding (Q2).

5. DISCUSSION AND CONCLUSIONS

Psychophysical results show overall that the proposed impact sound model (Model 1) adequately conveys the relevant information to the participants, who are able to use this information to guide their virtual walk. Although no significant differences in RTs and accuracy were found in the comparison against two other models, Model 2 was found to perform slightly worse, and Model 3 slightly better than Model 1. These results are consistent with the event-related analysis of phasic EDA: Model 2 appears to elicit the highest stress-related physiological arousal compared to Models 1 and 3, whereas Model 3 is shown to be marginally less stressful than Model 1.

This last finding, however, appears to disagree with the perceptions emerging from the verbal comments of the participants, where Model 1 came out as the most preferred and was ranked slightly higher than Model 3 in terms of functionality and pleasantness. Further analysis is necessary to examine the origins of this discrepancy. Furthermore, Model 1 had the best cognitive performance compared to the other two models. Finally, Model 1 was the easiest to learn among the three models.

The above results suggest that the adopted sonification approach may lead to improved results if adequate modifications, either to the mapping schemes or to the chosen sound stimuli, are performed. Ongoing work in this direction involves attempting to improve the model by combining impact sounds with the depth scan paradigm. Future work related to the model presented in this paper will therefore explore variations of the basic sound components used for encoding (impact sounds) and the scanning paradigm, as well as combining discrete encodings with continuous encodings for various object categories (e.g. walls, stairs, doors). We further plan to test the functionality and cognitive performance of the model in indoor and outdoor navigation scenarios using similar biosignal monitoring and analysis methods [31].

6. ACKNOWLEDGMENTS

The authors wish to thank the participants for their collaboration as well as the administration and O&M instructors at the National Institute for the Blind, Visually Impaired, and Deafblind in Iceland for their valuable input and generous assistance. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 643636.[6]

7. REFERENCES

[1] C. Scaletti, "Sound synthesis algorithms for auditory data representations," in *Auditory Display: Sonification, Audification, and Auditory Interfaces*, G. Kramer, Ed., vol. 1, pp. 223–251. Adison-Wesley, Reading, MA, USA, 1994.

[2] F. Avanzini, S. Spagnol, A. Rodá, and A. De Götzen, "Designing interactive sound for motor rehabilitation tasks," in *Sonic Interaction Design*, K. Franinovic and S. Serafin, Eds., chapter 12, pp. 273–283. MIT Press, Cambridge, MA, USA, March 2013.

[3] D. Dakopoulos and N. G. Bourbakis, "Wearable obstacle avoidance electronic travel aids for blind: A survey," *IEEE*

[6] http://www.soundofvision.net/

Trans. Syst. Man Cybern., vol. 40, no. 1, pp. 25–35, January 2010.

[4] Á. Csapó, G. Wersényi, H. Nagy, and T. Stockman, "A survey of assistive technologies and applications for blind users on mobile platforms: A review and foundation for research," *J. Multimod. User Interf.*, vol. 9, no. 4, pp. 275–286, December 2015.

[5] S. Shoval, J. Borenstein, and Y. Koren, "Auditory guidance with the Navbelt - A computerized travel aid for the blind," *IEEE Trans. Syst. Man Cybern.*, vol. 28, no. 3, pp. 459–467, August 1998.

[6] J. L. González-Mora, A. Rodríguez-Hernández, L. F. Rodríguez-Ramos, L. Díaz-Saco, and N. Sosa, "Development of a new space perception system for blind people, based on the creation of a virtual acoustic space," in *Engineering Applications of Bio-Inspired Artificial Neural Networks*, vol. 1607 of *Lecture Notes in Computer Science*, pp. 321–330. Springer Berlin Heidelberg, 1999.

[7] F. Fontana, A. Fusiello, M. Gobbi, V. Murino, D. Rocchesso, L. Sartor, and A. Panuccio, "A cross-modal electronic travel aid device," in *Human Computer Interaction with Mobile Devices*, vol. 2411 of *Lecture Notes in Computer Science*, pp. 393–397. Springer Berlin Heidelberg, 2002.

[8] P. B. L. Meijer, "An experimental system for auditory image representations," *IEEE Trans. Biomed. Eng.*, vol. 39, no. 2, pp. 112–121, February 1992.

[9] E. Milios, B. Kapralos, A. Kopinska, and S. Stergiopoulos, "Sonification of range information for 3-D space perception," *IEEE Trans. Neural Syst. Rehab. Eng.*, vol. 11, no. 4, pp. 416–421, December 2003.

[10] M. Bujacz, K. Kropidlowski, G. Ivanica, A. Moldoveanu, C. Saitis, A. Csapó, G. Wersényi, S. Spagnol, O. I. Jóhannesson, R. Unnthórsson, M. Rotnicki, and P. Witek, "Sound of Vision - Spatial audio output and sonification approaches," in *Computers Helping People with Special Needs - 15th International Conference (ICCHP 2016)*. Springer-Verlag, Berlin, Germany, July 2016.

[11] S. Delle Monache, P. Polotti, and D. Rocchesso, "A toolkit for explorations in sonic interaction design," in *Proc. 5th Audio Mostly Conference (AM '10)*, New York, NY, USA, September 2010, number 1, ACM.

[12] F. Avanzini and D. Rocchesso, "Controlling material properties in physical models of sounding objects," in *Proc. Int. Computer Music Conf. (ICMC'01)*, La Habana, Cuba, September 2001.

[13] S. Spagnol, "On distance dependence of pinna spectral patterns in head-related transfer functions," *J. Acoust. Soc. Am.*, vol. 137, no. 1, pp. EL58–EL64, January 2015.

[14] D. H. Ashmead, D. LeRoy, and R. D. Odom, "Perception of the relative distances of nearby sound sources," *Percept. Psychophys.*, vol. 47, no. 4, pp. 326–331, April 1990.

[15] W. G. Gardner and K. D. Martin, "HRTF measurements of a KEMAR," *J. Acoust. Soc. Am.*, vol. 97, no. 6, pp. 3907–3908, June 1995.

[16] S. Spagnol, M. Geronazzo, D. Rocchesso, and F. Avanzini, "Synthetic individual binaural audio delivery by pinna image processing," *Int. J. Pervasive Comput. Comm.*, vol. 10, no. 3, pp. 239–254, July 2014.

[17] S. Spagnol and F. Avanzini, "Frequency estimation of the first pinna notch in head-related transfer functions with a linear anthropometric model," in *Proc. 18th Int. Conf. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, December 2015, pp. 231–236.

[18] H. Møller, M. F. Sørensen, C. B. Jensen, and D. Hammershøi, "Binaural technique: Do we need individual recordings?," *J. Audio Eng. Soc.*, vol. 44, no. 6, pp. 451–469, June 1996.

[19] J. Blauert, *Spatial Hearing: The Psychophysics of Human Sound Localization*, MIT Press, Cambridge, MA, USA, 2nd edition, October 1996.

[20] M. Bujacz, P. Skulimowski, and P. Strumiłło, "Naviton - A prototype mobility aid for auditory presentation of three-dimensional scenes to the visually impaired," *J. Audio Eng. Soc.*, vol. 60, no. 9, pp. 696–708, September 2012.

[21] M. Brambring, "Language and geographic orientation for the blind," in *Speech, Place, and Action: Studies in Deixis and Related Topics*, R. J. Jarvella and W. Klein, Eds., pp. 203–218. John Wiley & Sons, Chichester, UK, 1982.

[22] F. L. Wightman and D. J. Kistler, "Factors affecting the relative salience of sound localization cues," in *Binaural and Spatial Hearing in Real and Virtual Environments*, pp. 1–24. Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1997.

[23] W. Boucsein, *Electrodermal Activity*, Springer, New York, NY, USA, 2nd edition, 2012.

[24] W. Klimesch, "EEG alpha and theta oscillations reflect cognitive and memory performance: A review and analysis," *Brain Res. Rev.*, vol. 29, no. 2–3, pp. 169–195, April 1999.

[25] M. Garbarino, M. Lai, D. Bender, R. W. Picard, and S. Tognetti, "Empatica E3 - A wearable wireless multisensor device for real-time computerized biofeedback and data acquisition," in *EAI 4th Int. Conf. Wirel. Mob. Commun. Healthcare (Mobihealth)*, Athens, Greece, November 2014, pp. 39–42.

[26] M. Benedek and C. Kaernbach, "A continuous measure of phasic electrodermal activity," *J. Neurosci. Methods*, vol. 190, no. 1, pp. 80–91, June 2010.

[27] W. J. Ray and H. W. Cole, "EEG alpha activity reflects attentional demands, and beta activity reflects emotional and cognitive processes," *Science*, vol. 228, no. 4700, pp. 750–752, May 1985.

[28] A. Keil, M. M. Müller, W. J. Ray, T. Gruber, and T. Elbert, "Human gamma band activity and perception of a gestalt," *J. Neurosci.*, vol. 19, no. 16, pp. 7152–7161, August 1999.

[29] N. A. Badcock, P. Mousikou, Y. Mahajan, P. de Lissa, J. Thie, and G. McArthur, "Validation of the Emotiv EPOC EEG gaming system for measuring research quality auditory ERPs," *PeerJ*, vol. 19, 2013.

[30] W. L. Zheng and B. L. Lu, "Investigating critical frequency bands and channels for EEG-based emotion recognition with deep neural networks," *IEEE Trans. Auton. Mental Develop.*, vol. 7, no. 3, pp. 162–175, September 2015.

[31] C. Saitis and K. Kalimeri, "Identifying urban mobility challenges for the visually impaired with mobile monitoring of multimodal biosignals," in *Universal Access in Human-Computer Interaction - 10th International Conference*, M. Antona and C. Stephanidis, Eds. Springer-Verlag, Berlin, Germany, July 2016, In press.

Author Index