UNIVERSITY OF CALIFORNIA SAN DIEGO

Physically-informed Percussion Synthesis with Nonlinearities for Real-time Applications

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Music

by

Jennifer Hsu

Committee in charge:

Professor Tamara Smyth, Chair
Professor Thomas Erbe
Professor Sarah Hankins
Professor Scott Klemmer
Professor Miller Puckette

2019

The dissertation of Jennifer Hsu is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2019

DEDICATION

To my family and friends, thank you for your love, support, and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CS	commuted synthesis
DWM	digital waveguide mesh
FDM	finite difference methods
FM	frequency modulation
FTM	functional transformation method
MS	modal synthesis
PDE	partial differential equation
PM	phase modulation

LIST OF SYMBOLS

A_0 and A_1	initial and final values for exponential pitch and harmonic number variation
$A_{0,i}$ and $A_{60,i}$	initial and final amplitudes for amplitude envelopes
$\mathrm{AP}(n)$ and $\mathrm{AP}_i(n)$	time-varying allpass filters
B and B_i	loopback FM feedback coefficients
$B(n)$ and $B_i(n)$	loopback FM time-varying feedback coefficient functions
b_0 and $b_{0,i}$	loopback FM parameter that controls the number of harmonics
$b_0(n)$ and $b_{0,i}(n)$	loopback FM time-varying parameter that controls the number of harmonics
b_{x} and b_{y}	beginning and ending values for $b_0(n)$ for harmonic number variation
c and c_i	time-varying allpass filter intermediate parameter
d, $d(n)$, and $d_i(n)$	time-varying allpass filter intermediate parameters
$e(n)$	excitation function
f_π and $f_{\pi,i}$	time-varying allpass filter carrier frequencies
$\tilde{f}_\pi(n)$	time-varying allpass filter time-varying carrier frequency function
f_0 and $f_{0,i}$	sounding frequencies in Hz
$f_0(n)$ and $f_{0,i}(n)$	time-varying sounding frequency functions in Hz
f_{b} and $f_{\mathrm{b},i}$	time-varying allpass filter timbre parameters
f_{c} and f_{c_i}	carrier frequencies in Hz
f_{low} and f_{high}	low and high frequencies for filtered noise burst bandpass filters
f_{m} and $f_{\mathrm{m},i}$	modulator frequencies in Hz
$f_{\mathrm{r},i}$	resonant modal frequencies
f_{x} and f_{y}	pitch glide starting and ending frequencies in Hz
f_{xy}	square root pitch glide coefficient
$h(n)$	exponential function for pitch or harmonic number variation
I	FM modulation index
L	raised cosine excitation length
l_0 and l_1	coefficients for linear pitch or harmonic number variation

M and M_i	time-varying allpass filter modulation parameters
$m(n)$	loopback FM modal synthesis output
$m_s(n)$	traditional modal synthesis output
N_f	number of resonant modes
n	time sample
n_d	pitch or harmonic number variation time in samples
$n_{60,i}$	$\mathrm{T}_{60,i}$ time in samples
$p(n)$	excitation position function
$r(n)$	acoustic resonator impulse response
$s_i(n)$	traditional modal synthesis sinusoidal components
T	sampling period
T_{60} and $\mathrm{T}_{60,i}$	time to decay by 60 dB in seconds
t_d	pitch or harmonic variation time in seconds
$w_g(n)$	global amplitude envelope
$w_i(n)$	amplitude envelope functions
$x_{\mathrm{fb}}(n)$	feedback FM in discrete form
$x_{\mathrm{fm}}(n)$	FM in phase modulation form
$y(n)$	percussion synthesis output signal
$z_0(n)$ and $z_{0,i}(n)$	loopback FM closed-form representation
$z_{\mathrm{c}}(n)$ and $z_{\mathrm{c},i}(n)$	loopback FM sample-by-sample rotation formulation
$z_{\mathrm{fb}}(n)$	feedback FM in complex exponential form
$z_{\mathrm{fm,c}}(n)$	FM carrier oscillator in complex exponential form
$z_{\mathrm{fm,m}}(n)$	FM modulator oscillator in complex exponential form
τ_h	exponential decay constant for pitch or harmonic number variation
$\tau_{w,i}$	decay rates for amplitude envelopes
$\theta_0(n)$ and $\theta_0(t)$	instantaneous phase, integral of ω_0
ω_0 and $\omega_{0,i}$	sounding frequencies in rad/s
$\omega_0(n)$ and $\omega_{0,i}(n)$	time-varying sounding frequency functions in rad/s
ω_{c} and $\omega_{\mathrm{c},i}$	carrier frequencies in rad/s
$\omega_{\mathrm{c}}(n)$ and $\omega_{\mathrm{c},i}(n)$	time-varying carrier frequency functions in rad/s
ω_{m}	modulator frequency in rad/s

LIST OF SUPPLEMENTARY AUDIO EXAMPLES

24. hsu_circularPlate_loopbackFM.wav

25. hsu_circularPlate_traditionalMSCommutedSynthesis.wav

26. hsu_circularPlate_loopbackFMCommutedSynthesis.wav

27. hsu_circularPlate_traditionalMSTimeVaryingAPCommutedSynthesis.wav

28. hsu_circularPlate_loopbackFMTimeVaryingAPCommutedSynthesis.wav

29. hsu_reverseCircularPlate_loopbackFMTimeVaryingAPCommutedSynthesis.wav

ACKNOWLEDGEMENTS

I would like to thank the members of my committee for their insightful guidance over the past six years. Professor Tamara Smyth, my committee chair has always enthusiastically encouraged me to research any subject that I brought up to her, and working on research problems together has been an invaluable experience. Professor Miller Puckette sent me a loopback FM Pure Data (Pd) patch that inspired the entire subject of this dissertation. Professor Tom Erbe has been so helpful in guiding me through software implementations and encouraging me to improve my sound synthesis results ever since the first year that I began my program here at UCSD. I am grateful for my conversations with Professor Sarah Hankins, in which we conceptualized a possible percussion software synthesizer. Professor Scott Klemmer's advice has been instrumental in guiding me from my initial research interests during my qualifying exams to my current dissertation topic. Professor Shlomo Dubnov has also been an immensely important figure as he guided me through my first few years of research here at UCSD in music information retrieval and rhythmic transformations. I would like to extend my thanks to Professor Katharina Rosenberger who has also taken the time to advise me to look at the musical implications of my work.

I would like to acknowledge my UCSD colleagues, Chris Donahue, Cheng-i Wang, Colin Zyskowski, Kevin Haywood, and Issac Muñoz for their generous insight and support throughout my years in this program. I would like to thank my parents, Eddie and Grace Hsu and my siblings Irene, Caroline, and William, for their support throughout my entire life, for guiding me to pursue a technical degree in college and cultivating in me, a love for music and living creatively. I am immensely thankful for my friends from UC Berkeley and Stanford for reminding me to trust myself and my own abilities. And last, I want to thank my Tribe 20:4 music collective members, for encouraging me to step outside of my comfort zone, to challenge myself, and to honestly express my appreciation for everyone around me. The endurance required to complete this work was only possible because of the support and encouragement gifted to me by my advisors, family, and friends.

Portions of Chapters 3, 4, and 5 are coauthored with my advisor, Professor Tamara Smyth, as it appears in

Jennifer Hsu and Tamara Smyth. Percussion synthesis using loopback frequency modulation oscillators. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

The dissertation author was the primary researcher and author of this paper.

VITA

2011	Bachelor of Arts - Double Major in Cognitive Science and Computer Science, University of California Berkeley
2013	Master of Arts in Music - Music, Science, and Technology, Stanford University
2019	Doctor of Philosophy in Music, University of California San Diego

PUBLICATIONS

Jennifer Hsu and Tamara Smyth. Percussion synthesis using loopback frequency modulation oscillators. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

Tamara Smyth and Jennifer Hsu. Representations of self-coupled oscillators with time-varying frequency. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

Tamara Smyth and Jennifer Hsu. On phase and pitch in loopback frequency modulation with a time-varying feedback coefficient. In *the 26th International Congress on Sound and Vibration*, Montreal, Canada, July 2019.

Tamara Smyth, Jennifer Hsu, and Ryan Done. Toward a real-time waveguide mesh implementation. In *the International Symposium on Musical Acoustics*, pages 54–57, Montreal, Canada, June 2017.

ABSTRACT OF THE DISSERTATION

Physically-informed Percussion Synthesis with Nonlinearities for Real-time Applications

by

Jennifer Hsu

Doctor of Philosophy in Music

University of California, San Diego, 2019

Professor Tamara Smyth, Chair

This dissertation introduces a physically-informed, abstract synthesis method that applies loopback frequency modulation (FM) to real-time parametric synthesis of percussive sounds. Loopback FM is a variant of FM whereby the output "loops back" to modulate its frequency by an amount determined by a feedback coefficient which, when made time-varying, results in pitch glides. Here, loopback FM is used to parametrically synthesize this effect in struck percussion instruments, known to exhibit this characteristic due to nonlinearities in modal coupling. Inspired by the sonic potential of physics-based nonlinear percussion synthesis models, the loopback FM synthesis method uses an abstract synthesis approach in order to create a wide variety of percussive sounds in real-time. A linear, modal synthesis percussion model is modified to use loopback FM oscillators, which allows the model to create unique, eclectic, and experimental sounding percussive hits in real-time. Musically intuitive parameters are emphasized resulting in a usable percussion sound synthesis method.

Chapter 1

Introduction

1.1 Introduction

This work documents my journey to create a real-time percussion synthesis method that is capable of producing a wide variety of sounds with musically intuitive controls. As a beat-based music creator, my dissatisfaction with available tools and my interest in studying synthesis systems brought me to study many different types of percussion synthesis models and to figure out how to incorporate the parts of those sounds that I enjoy into my own percussion synthesis method. This document situates my percussion synthesis technique, largely based on the work described in [31], in the context of the current state of percussion synthesis methods, and how my focus shifted from a physical modeling-based synthesis algorithm to an abstract synthesis domain. Synthesis examples and applications are shown following an explanation of the novel percussion synthesis method.

1.2 Research Goals and Objectives

As a music producer, much of the time, I tend to work with drum samples. In my music, I like to use slightly different versions of the same percussion sample to break up the monotony. I frequently turn to modify the volume and pitch or filtering the sound through some audio effect. The issue with drum samples is that time-warping, pitch shifting, and adding audio effects can change the sound in undesirable ways. Sometimes these procedures create artifacts, and other times, the essence of the sound is lost. Through this process, I started to feel unsatisfied because, with samples, I simply do not have enough control of the sound.

I began synthesizing drums using additive and frequency modulation (FM) synthesis instruments but found it difficult to know which oscillator frequencies to use and how to modify the parameters to create the sound that I would like. These sounds were also a bit too artificial-sounding for my style of music. Physical modeling-based percussive synthesizers, like Ableton's Collision and Applied Acoustic System's Chromaphone, create high-quality percussive sounds. My issue with these instruments is that, although some people may prefer physical measurement-based parameters, such as mallet hardness and hit position, I want to control parameters that directly describe changes to the *sound* rather than *physical measurements*. These parameters might be described with words such as brightness, overall decay, sounding frequency, etc.

When I studied percussion synthesis models in the sound synthesis literature, I found that with the exception of the impact synthesizer presented in [4], most percussion models focus on accurately synthesizing the sound of a specific plate or membrane rather than synthesizing a wide variety of percussive sounds. Instead of striving for realistic simulation, as previous research has done, I am interested in assembling a percussion synthesizer that creates a variety of novel sounds. One of the sample libraries that strongly motivated my idea to work on a percussion sound synthesizer is Soundiron's Antidrum collection that features eclectic and experimental percussive samples including metallic hits, zips, and noisy, machine-like sounds. It is a goal of this research to create those types of unique percussive timbres.

The work presented in this document is inspired by the desire to have a percussion synthesizer that can create a wide variety of sounds with intuitive musical parameters. The sound palette of this synthesizer is inspired by the dynamic and nonlinear phenomena in percussion instruments, which give rise to eclectic and experimental percussive sounds. To summarize, I would like the percussion synthesis method to

1. have the ability to synthesize a variety of percussive sounds including those that sound unique and experimental
2. give the user parametric control over musical dimensions including note duration, sounding frequency, and timbre
3. synthesize sounds in real-time.

In the next section, we discuss the behaviors and sound effects observed in nonlinear percussion instruments. I am interested in nonlinear percussion instruments because complex effects result when such instruments are hit with a strong excitation. When a theoretically linear instrument is struck strongly, the sound that emerges is simply a louder version of the same instrument struck softly. However, when a nonlinear instrument is strongly hit, the sound that emerges is louder but also includes extra spectral components and pitch glides that are not usually present in the softly hit version. Because a single nonlinear plate, membrane, or instrument can express such a wide array of sonic timbres based on input excitation amplitude, a study of nonlinear percussive instruments can point us towards ideas for how to create a variety of percussive sounds. Following the summary of nonlinear effects observed in percussion instruments, we briefly review various percussion synthesis models and discuss parametric control, real-time synthesis capabilities, and whether it is possible to create a diverse set of sounds with each model.

1.3 Nonlinear Percussion Analysis

With linear systems, the only difference between striking an instrument softly or strongly is the amplitude of the frequency components. This is not common in real physical systems, which exhibit amplitude-dependent nonlinearities. For example, when a Chinese opera gong is hit lightly, it generates a sinusoidal-like tone, but when the gong is struck very strongly, pitch glides can be perceived [24]. The following section discusses the amplitude-dependent nonlinearities that are most frequently mentioned in percussion analysis research: pitch glides, the cascade of low to high frequency components, and chaotic behavior (wave turbulence). Some of these nonlinear effects have been implemented in models described in Section 1.4 and Chapter 2.

1.3.1 Pitch Glides

In some gongs, like those of the Chinese opera, and membrane-based instruments, a pronounced pitch glide can be observed where the frequency either increases or decreases over time when the instrument is struck with a large enough excitation. The pitch glide behavior of gongs is explored in [50] where it is explained that the geometry of the gong greatly influences the type of pitch glide. Flat plates have an increasing pitch glide, conical and spherical shells have a decreasing pitch glide, and curved plates can produce increasing or decreasing pitch glides depending on their shape. The frequency can change by as much as 20%.

Pitch glides also occur in membrane-based instruments due to tension modulation. In [23], a frequency analysis of bass drum recordings was performed. When the bass drum is struck harder, the starting frequency increases and then drops over time as amplitude decreases, creating a decreasing pitch glide. It was reported that the pitch glide rates did not change due to hit position or strength. The decrease in pitch glide may be a prominent feature of bass drums, and something that listeners have grown to expect, as it is discussed in Werner's circuit analysis and simulation of the popular Roland TR-808 drum machine bass synthesizer as a "pitch sigh" [62]. Another membrane-based instrument that experiences a pitch glide due to tension modulation is the tom tom. In [58], the researchers observe that at low amplitude strikes, the frequency components of the tom tom only decrease in amplitude. At moderate and high velocity hits, the frequency components begin at a higher frequency and amplitude and then decrease in both dimensions over time.

1.3.2 Low Frequency to High Frequency Energy Cascade

One of the most prominent nonlinearities observed in both cymbals and gongs is that when they are excited with a high amplitude, low frequency energy is converted into high frequency energy shortly after the cymbal is struck [49, 25, 50]. About a second after a cymbal is hit, there is a characteristic "shimmer" that can be observed in the 3 to 5 kHz range [49]. Without the "shimmer" nonlinearity, the sound is more reminiscent of a bell than a cymbal. For gongs, the transfer of energy from low frequencies to high frequencies usually occurs at a much slower scale than for cymbals [50]. This nonlinear phenomenon can be seen in Figure 2.1, which displays spectrograms of modeled plates under linear and nonlinear conditions.

1.3.3 Chaotic Behavior

Another nonlinear effect in percussion instruments is that when a cymbal is excited with a high amplitude sinusoid, chaotic behavior can be observed. For a low amplitude sinusoidal excitation, a few harmonics are generated. Then, as the amplitude of the excitation increases, subharmonics begin to appear. As the amplitude increases further, chaotic behavior is observed where the harmonics that are generated can be double, triple, or more times the frequency of the driving sinusoid [49, 50, 24]. Figure 1.1 displays a plot of harmonic generation at different amplitude levels. It has been noted that cymbals display chaotic behavior even at normal operating amplitudes [25].

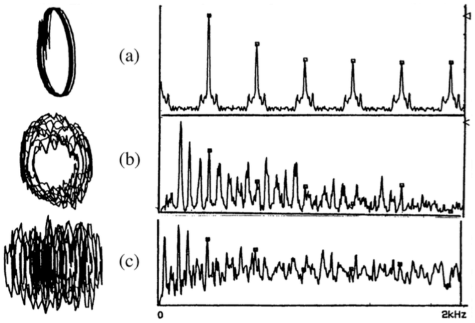


Figure 1.1: Phase plots of velocity vs. displacement from [49]. Right: phase plots of velocity vs. displacement. Left: the associated velocity spectra for a cymbal driven by a sinusoid at 320 Hz. (a) harmonics generated from a low amplitude excitation; (b) subharmonics generated from a medium amplitude excitation; (c) chaotic behavior from a large amplitude excitation.

1.4 Overview of Percussion Synthesis Methods

Synthesis of percussion instruments and the parts of which they are composed, such as membranes and plates, have been realized using several different techniques including FM Synthesis [15], modal synthesis (MS) [18], the Functional Transformation Method (FTM) [5], finite difference methods (FDM) [9], and the digital waveguide mesh (DWM) [20]. In this section, we examine physics-based methods for percussion synthesis and evaluate these models based on the three desired characteristics for my percussion synthesis system. We look at the types of sounds that are possible using each model, the musical parameters available from each model, and the real-time synthesis capabilities.

1.4.1 FM Synthesis

A single FM oscillator, in which the frequency of an oscillator is affected by another sinusoid, is capable of creating a percussive sound, as discussed in Chowning's paper on FM synthesis [15]. In this work, Chowning describes a number of sounds that can be synthesized with FM and introduces FM parameters to produce bell-like percussive sounds and wooden drums. He mentions that the amplitude envelope used contributes greatly to the difference in timbre for the percussive sounds. Although FM synthesis is suitable for real-time, synthesis, the modulating frequency and index parameters are not intuitive and it is difficult to tune a sound to one's liking. FM synthesis is used to create extra spectral information to emulate the sounds of nonlinear percussion instruments in [6] and [4], the latter of which will be further explained in Chapter 2. FM-like effects are also used in the percussion synthesis system presented in this dissertation to create sounds similar to those made by nonlinear percussion instruments.

1.4.2 Digital Waveguide Mesh (DWM)

The 2D DWM is a physics-based model that simulates wave propagation in two dimensions [20]. The basic 2D DWM is rectilinear in shape, although other topologies (e.g. triangular, hexagonal) have been explored. Triangular DWMs have been used extensively for their computational efficiency and accuracy [26, 27, 28, 41].

One of the issues with DWMs is dispersion. In an ideal membrane, the velocity of the propagating waves is equal for all spatial frequencies, but in a sampled DWM, dispersion error exists because the velocity of the waves is dependent on both spatial frequency and direction of travel. The resonant modes of a DWM are different from the expected modes of an ideal membrane. Many DWM methods make topology decisions and calculations with dispersion error in mind [26, 27, 28, 41].

Synthesizing a Variety of Sounds with the DWM

The 2D DWM has been used to model plates [20], membranes [20, 26, 27, 28], and even entire drum instruments [3]. To change the output of the 2D DWM from a membrane to a plate, one can simply switch the sign of the reflection at the boundaries of the 2D DWM [20]. Though the DWM is capable of creating many different types of sounds, synthesizing a wide variety of percussive timbres has not been the main focus of the work. Rather, most research seeks to accurately simulate a single circular membrane or plate [26, 27, 28, 41].

Parametric Control Over Musical Dimensions with the DWM

The decay time of a signal simulated with a DWM depends on the filters at the boundaries. In [3], IIR lowpass filters are used at the boundaries to model energy loss. In [28], a method is shared for specifying the decay time of the signal by accounting for a loss at every junction of the mesh.

In [3], one of the parameters is drum radius, which is related to the sounding frequency. Perhaps because the output of the 2D DWM, and possibly those of higher dimensions, is inharmonic, there has not been much work to specify the sounding frequency of the 2D DWM and its relationship to mesh size. In a general percussion synthesis instrument, though, being able to specify the sounding frequency (or closest sounding frequency) is desirable.

A relatively large amount of research has been given to transforming the timbre of the 2D DWM. By using frequency-dependent filters at the boundary of the DWM, the timbre can be modified. Van Duyne and Smith use 1st-order allpass filters at the boundaries of a 2D DWM to retune the modes of a plate model so that the mode frequencies are “stretched.”[21]. The higher partials are more spread apart than lower partials, as shown in Figure 1.2, modeling a stiff plate and adding inharmonicity to the timbre. Similarly, [28] implement mode shifting by inserting filters within the 2D DWM.

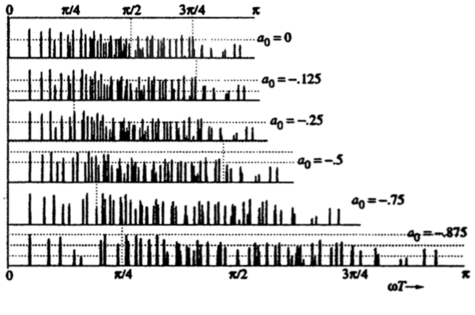


Figure 1.2: Magnitude response of a stretched 2D DWM stiff plate from [21]. The mode stretching changes as the allpass coefficient a_0 is changed from 0 to -0.875. The plots have been scaled to vertically align the fundamental mode.

Striking the 2D DWM at different positions and listening to the output from different positions also affect the timbre as the level of different spectral components changes as strike and listening position are modified. Striking the DWM in the corner excites all the modes [36]. Any input excitation, such as that from the hammer model in [28], can be moved around to strike the DWM in various positions.

Timbre can also be changed by modeling different materials for the excitation model, as was done for the hammer model in [28], or extending the DWM with material parameters such as density and membrane tension as described in [2].

The DWM and Real-time Synthesis

The 2D DWM is an extension of the 1D digital waveguide, which models a traveling wave efficiently as a digital delay line [33]. When extended to two dimensions though, as done so for percussive instruments here, the digital waveguide loses the efficiency gains from the one-dimensional model and is unable to run in real-time [32]. When DWMs are small, they can be run in real-time, as shown in [46], however, as they grow larger, computational times increase [53]. Explorations to represent the digital waveguide mesh as a transfer function, as discussed in [53], and expressing the transfer function as a set of 2$^{\text{nd}}$-order filters so that it can be run in real-time is explained in Chapter 2.

1.4.3 Finite Difference Methods (FDMs)

Finite difference methods (FDMs) are numerical methods that solve partial differential equations (PDE) which describe a physical system. With finite difference schemes, the derivatives in PDEs are replaced by forward, backward, and/or center differences, essentially discretizing the time and space dimensions. Finite difference models for percussion synthesis commonly use PDEs for plates and membranes, which include the von Kárman and Berger equations. Each PDE typically describes the transverse displacement of a point on a membrane or plate. The transverse displacement over time for a single point is the signal that is used for audio synthesis.

Synthesizing a Variety of Sounds with FDMs

Linear FDMs have been used to synthesis rectangular plates. An undamped numerical simulation of a linear plate based on the Kirchhoff model is presented in [52]. This model was then extended to a damped plate model in [14, 42]. Another FDM based on the Kirchhoff plate model was used in [12] to synthesize the sound of a vibrating linear plate. In this work, it is noted that the relationship between the stiffness parameter and a tension parameter determines whether a plate or membrane is simulated.

A nonlinear von Kárman plate was modeled in [7] and extended in [9] and [11]. This method is capable of creating the sounds of a linear and nonlinear plate and greatly inspired my research with percussion synthesis. Further details about this model are covered in Chapter 2. A more realistic model of a cymbal or gong based on a nonlinear curved shell model is presented in [10]. Nonlinear shell models are capable of creating pitch glides and crashes.

Parametric Control Over Musical Dimensions with FDMs

Decay time parameters are seen in a few of the FDMs. The linear plate model in [12] includes two damping parameters to allow for decay control. An even more musically intuitive parameter is the decay mechanism in [10], which is presented as a T_{60} parameter for each frequency component, specifying the time at which the frequency component should decrease by 60 dB. The mechanism allows for increased damping of high frequencies. This frequency-dependent decay is crucial for nonlinear effects as the pitch glide trajectory is influenced by the damping, and the cascade of high frequency energy that occurs in crashes must be attenuated in a natural way.

The FDM parameters that affect timbre are typically physical measurements such as plate size, thickness, and Poisson's ratio [14, 42, 12, 10]. Many of these physical measurements have to do with the material of the plate and determine the output timbre. In [14, 42], a number of materials were created by changing the physical parameters of this model: aluminum, glass, carbon, carbon-epoxy, and wood. The shell model in [10] includes a parameter that accounts for curvature, which modifies the timbre by adding more modes to the mid-range frequency spectrum, resulting in a more dissonant sound that is characteristic of a cymbal.

FDMs and Real-time Synthesis

Because plates and membranes are 2D systems, and the transverse displacement of each point depends on the displacement of every other point on the 2D system, large models can become computationally complex. Furthermore, for accurate simulations, the time and step sizes must be small, so increases in accuracy equate to increased computational expenses. Depending on the model, linear models, such as the one presented in [12], may have computational costs similar to those of modal synthesis and operate in real-time as described in Section 1.4.4. Incorporating nonlinear and computationally heavy calculations can further increase the complexity and computational expense of the entire system [7, 9, 10, 11].

1.4.4 Modal Synthesis (MS)

Modal Synthesis (MS) is a physical modeling technique that simulates the sound of an acoustic object using its acoustic modes or vibrational patterns. The sinusoidal motion of an acoustic object's modes gives rise to the resonant frequencies of the object. A resonating object's vibrational motions can be decomposed into a set of modes–each oscillating at a complex frequency. In traditional MS, each mode is characterized by a frequency, damping factor, and modal shape [1].

With modal synthesis, an input force signal is applied to a bank of damped harmonic oscillators. The oscillator outputs are added up together to form the sounding output. Modal information can be recorded from objects, analytically derived from theoretical equations or extracted from recorded audio signals.

Synthesizing a Variety of Sounds with MS

MS is useful for synthesizing a variety of sounds because unlike other physical-modeling methods, there are no underlying functions that need to be solved. In order to synthesize a different type of sound, all that needs to change are the modal frequencies, decays, and initial amplitudes. In [17], the MS system can create the sounds of bars, membranes, and plates. The work in [19] can synthesize struck wood and metallic timbres. In [61], the sounds of a rigid bar, rectangular membrane, circular membrane, circular plate, and L shaped membrane are simulated. The models used in [17] and [19] as well as available synthesis parameters are elaborated upon in Chapter 2.

Some MS systems are linear, but model nonlinearities through interactions or coupled modes. Modalys [22], previously known as MOSAIC [45], is a modal synthesis environment with built-in functions that can analytically compute modal data for simple objects. Methods for simulating the vibrations of rectangular and circular membranes and plates are included. Though the resonating structures themselves are linear, the structures may be connected in nonlinear ways, including interactions such as bowing and plucking. [13] present an efficient method for synthesizing the sound of struck thin shells experiencing small, nonlinear vibrations. The method is capable of producing “crashing” and “rumbling” sounds for objects such as trash cans, water bottles, recycling bins, and bell-like cymbals. This method uses a linear modal model where the modes are nonlinearly coupled. Similarly, the work in [18] uses a modal approach with nonlinearly coupled oscillators to synthesize vibrations in a plate and apply this to the sound synthesis of gongs and cymbals. The aim of this research was to resynthesize pitch glides and the low to high frequency energy cascade that contributes to the bright and shimmery timbre that is associated with gongs and cymbals.

Parametric Control Over Musical Dimensions with MS

Most MS models use parameters that control physical measurements as opposed to musical dimensions. For example, the parameters such as impact location, impact force, and object material are used in the MS system from [61]. The nonlinear cymbal and gong synthesis system from [18] uses physical parameters such as plate size, plate height, material density, input/output location, and input force pressure. The Modalys system also uses parameters for physical measurements such as radius, density, and strike position for a circular plate [22]. These physical measurements tend to affect many musical dimensions at the same time. An exception to these physical measurement parameters is the Coupled Mode Synthesis model introduced in [19], in which a single parameter affects the decay of the individual components and is capable of changing the timbre from wood to metal.

MS and Real-time Synthesis

MS systems are typically capable of real-time synthesis when the number of modes is small. Many of the models that have been described in this section are computable in real-time including [17] and [19]. Modalys [22] was later adapted to run in real-time [37].

For systems with a large number of modes, stored samples may be used instead of run-time synthesis as described in [61]. For the complex, struck thin shell model in [13], multiple optimizations along with large memory storage capabilities are necessary in order for the system to run in real-time. The sound simulations of the nonlinear cymbal and gong presented in [18], although complex, compelling, and rich in timbre, do not run in real-time.

1.4.5 Functional Transformation Method (FTM)

The Functional Transformation Method (FTM) is related to modal techniques in that it describes a linear system as a set of modes. Beginning with a partial differential equation (PDE), initial conditions, and boundary conditions, the FTM uses the Laplace transform to remove the temporal partial derivative and the Sturm-Liouville transform to remove the spatial partial derivative. This results in a multidimensional transfer function which can be implemented with digital filters. Because this technique does not discretize the spatial dimension, it results in accurate modal frequencies. One of the issues with the FTM is that it is mostly used on simple shapes because irregular and complex shapes cause the mathematics to become unwieldy [59].

Synthesizing a Variety of Sounds with the FTM

For linear percussion instruments, the FTM has been used to simulate a rectangular reverberation plate and a circular, kettledrum membrane [59]. A nonlinear, rectangular membrane, capable of simulating membrane tension modulation as a pitch glide, is presented in [48]. This nonlinear membrane is based on a 2D extension of a nonlinear, tension-modulated string model. A nonlinear, circular membrane that is also capable of pitch glides is presented in [5]. This model is reviewed in detail in Chapter 2. While the FTM is capable of creating many different types of percussive sounds, a single model is only able to create variations of its own sound, depending on the physical parameters available for each model. Changing plate or membrane shapes would require changing (and solving) the underlying PDE.

Parametric Control Over Musical Dimensions with the FTM

Because FTM models are directly derived from physical equations, parameters in FTM models are usually physical measurements such as plate or membrane thickness, radius, and density. These physical measurements affect multiple dimensions at the same time. For example, timbre and decay could both be affected by a plate density parameter. An exception to this is shown in [59], where a pitch frequency value can be modified for a plate and membrane model.

The FTM and Real-time Synthesis

Though most linear models are capable of real-time synthesis, the linear plate and circular membrane FTM models in [59] were unsuitable for real-time use for the full set of modes at the time of publication. It is possible that if those models were recreated today, the models could be run in real-time.

Although many nonlinear percussion synthesis models are unsuitable for real-time use due to computational complexity and stability issues, an exception to this rule is presented in [48]. By including the nonlinear tension modulation as a force input to the system and approximating computationally heavy calculations, the nonlinear membrane model is able to simulate up to 1000 modes in real-time at a sampling rate of 44.1 kHz. The nonlinear, circular membrane model from [5] requires a computationally heavy nonlinear calculation that prevents the model from synthesizing sounds in real-time. In [44], that calculation is replaced by an approximation that allows for efficient sound synthesis.

1.5 Dissertation Organization

In Chapter 2, we discuss the most relevant percussion synthesis systems and the motivation for moving from the physical modeling domain to an abstract synthesis domain. Chapter 3 introduces loopback FM, a nonlinear FM-based oscillator in which the output “loops back” to modulate its own frequency. In Chapter 4, loopback FM is applied to percussion synthesis to create sounds similar to the ones exhibited by nonlinear percussive systems. Synthesis examples and a step-by-step guide of how to synthesize a desired percussive sound are covered in Chapter 5. Chapter 6 presents an ongoing software implementation, musical implications, and future directions for the loopback FM percussion synthesis method.

Chapter 2

Percussion Synthesis Models

2.1 Introduction

In this chapter, the most relevant percussion synthesis models are covered in detail. We first look at three MS models that show how MS is capable of creating a wide variety of struck sounds in real-time. Then we review two physics-based percussion synthesis systems, [7] and [5], and discuss how incorporating nonlinearities is computationally expensive in both systems. We also examine the 2D DWM and its modal representation as a bank of parallel, resonating filters. This inspired the change in direction from physics-based synthesis to abstract sound synthesis for a percussion synthesis model.

2.2 Timbre Variety with the Physically Informed Control of MS (PhISM) System

In [17], Perry Cook introduces his Physically Informed Control of Modal Synthesis (PhISM) system that is capable of synthesizing the sound of objects with a few decaying resonating modes. The system is general in that the modes can be retrieved from solving theoretical equations, sinusoidal analysis of recorded sound, or any other means. An excitation can be found by inverse filtering a recording of an impact hit. Different excitations and collections of modal frequencies can be gathered to create a wide variety of sounds with this method.

With PhISM, one can develop simple rules to create a strike position parameter that modifies the level of different modes, thereby changing the timbre for each strike. Additionally, the user of the system is able to control performance style, which takes into account previous notes plated, previous strike position, and performer experience when making parametric choices for the current note.

2.3 Decay Time and Timbre Variety with Coupled Mode Synthesis

Coupled Mode Synthesis, as presented in [19] is capable of creating percussive sounds that are similar to those of struck wood and metal. Coupled Mode Synthesis reformulates traditional MS to allow for simple decay control and mode coupling effects [19]. In traditional MS, each mode is represented by a decaying, 2$^{\text{nd}}$-order resonating filter, and the output of these filters are summed and added for the synthesis output. These filters are realized according to

$$H_{\mathrm{ms},i}(z) = \frac{1}{1 - 2r_i \cos(2\pi f_{\mathrm{r},i} T) z^{-1} + r_i^2 z^{-2}}, \tag{2.1}$$

where i indicates the mode number/index, $f_{\mathrm{r},i}$ is the frequency of mode i, r_i is the decay associated with mode i, T is the sampling period, and z is a complex number in the form $z = e^{j\omega}$. In Coupled Mode Synthesis, these filters are modified to

$$H_{\mathrm{cms},i}(z) = \frac{z^{-1} H_{\mathrm{AP1}}(z)}{1 + z^{-1} H_{\mathrm{AP1}}(z)} \tag{2.2}$$

where

$$H_{\mathrm{AP1}}(z) = \frac{a_i + z^{-1}}{1 + a_i z^{-1}} \tag{2.3}$$

which is a 1$^{\mathrm{st}}$-order allpass filter with $a_i = -\cos(2\pi f_{\mathrm{r},i}T)$. This is a non-decaying resonating filter. To couple these resonators, all $H_{\mathrm{cms}}(z)$ resonators are arranged in a parallel filter bank, the outputs are summed together, the sum is filtered by a coupling filter, and finally fed back to sum with the resonator inputs. The coupling filter controls the decay and setting it to a lowpass filter creates natural sounding decays. This model is capable of creating percussive sounds that are similar to those of struck wood and metal.

2.4 A MS Impact Synthesizer

Aramaki et al. present a real-time synthesizer that emulates the sound of struck objects made of different materials in [4]. Using a human participant listening experiment, the researchers found that normalized decay time distinguishes wood from metal and glass sounds and that roughness distinguishes between metal and glass timbres. Additive synthesis was used to implement the synthesizer and modeled as

$$s(t) = \Theta(t) \sum_{m=1}^{M} A_m \sin(\omega_m t + \Phi_m) e^{-\alpha_m t} \tag{2.4}$$

where $\Theta(t)$ is the Heaviside function and m specifies the sinusoidal component out of a total of M components. For each m^{th} component, A_m is the amplitude, ω_m is the frequency, Φ_m is the phase, and α_m is the damping coefficient.

For damping, a frequency-dependent "damping law" of the form

$$\alpha(\omega) = e^{(\alpha_G + \alpha_R \omega)} \tag{2.5}$$

was developed where ω represents the frequency, α_G is a global damping coefficient, and α_R is a relative damping coefficient, tuned for a material such as metal, glass or wood. The damping coefficient per mode in (2.4) is set according to $\alpha_m = \alpha(\omega_m)$. α_G gives the user control over the global decay time while α_R gives control over the frequency-dependent decay based on material.

The model allows for spectral dilation, a timbre-based parameter that allows one to stretch (or contract) the distance between spectral components using a parameter for global S_G or relative S_R dilation. The spectral dilation operation moves ω_m in (2.4) to another frequency location $\bar{\omega_m}$. Spectral dilation was used to transition from metal to glass as that was found to decrease the roughness in the synthesized signal.

Another timbral control introduces more frequency components into the signal using amplitude and frequency modulation. This produces more perceptual roughness and is perceived as a transition from glass to metal. The researchers found that setting the modulation frequencies to the same value for each component sounded unnatural, so they designed the modulation frequency to be represented as a percent of the critical bandwidth associated with each spectral component. This percentage of critical bandwidth is a user-friendly parameter that could be used to greatly change the timbre of the resulting synthesis. The resulting synthesizer is a sum of frequency- (or amplitude-) modulated oscillators. While this is called “additive synthesis” in the text, here, we refer to it as MS because the oscillators are not pure sinusoids, as one would expect for additive synthesis.

The strategy of adding together the output of FM oscillators, as shown with the amplitude and frequency modulation in [4] is a technique used in the current work. In [6], frequency- and amplitude- modulated oscillators were summer to synthesize the sound of a Javanese gong. Similarly, in [38], the pitch glide capabilities of a Duffing oscillator are explored for application to the sound synthesis of a Chinese opera gong. This method is attractive for the purposes of this research because it is capable of creating a variety of percussive sounds, including those that are similar to nonlinear percussion instruments, and runs in real-time.

2.5 A FDM for the Sound Synthesis of Nonlinear Plates

In [7], Bilbao uses a finite difference scheme to synthesize the sound of a nonlinear plate based on the moderately nonlinear von Kárman thin plate equation

$$\ddot{w} = -\kappa^2\nabla^2\nabla^2 w + c^2\nabla^2 w - 2\sigma\dot{w} + b_1\nabla^2\dot{w} + \frac{1}{\rho}L[F,w] \tag{2.6}$$

where $w(x,y,t)$ is the displacement at position (x,y) and time t, ∇^2 is the Laplacian, $\nabla^2\nabla^2$ is the biharmonic operator, κ is the stiffness, ρ is Poisson's ratio, c is a parameter related to tension, and F is related to the Airy stress function. σ controls the overall decay rate of the signal and b_1 defines the decay rate by allowing the higher frequencies to die out more quickly. $L[\cdot,\cdot]$ is a nonlinear operator. (2.6) without the $\frac{1}{\rho}L[F,w]$ term on the left, is a linear Kirchhoff plate model with damping, similar to the plate models used in [52, 14, 42, 12] and described in Chapter 1.

A grid function $w^n_{i,j}$ is defined to approximate the continuous displacement variable $w(x,y,t)$ at spatial coordinates $x = i\Delta$ and $y = i\Delta$ at time step n for grid spacing Δ. By using centered and backward differences, Bilbao develops a nonlinear finite difference scheme that updates $w_{i,j}$ for the next time step $n+1$ as a function of previous values of $w_{i,j}$ and $L[F,w]^n$:

$$\begin{aligned} w^{n+1}_{i,j} =& \eta \sum_{|k|+|l|\leq 2} \beta_{|k|,|l|} w^n_{i+k,j+l} \\ &+ \eta \sum_{|k|+|l|\leq 1} \gamma_{|k|,|l|} w^{n-1}_{i+k,j+l} + \frac{T^2}{\rho}L[F,w]^n. \end{aligned} \tag{2.7}$$

The values for β and γ are given by the von Kárman equation parameters and the grid spacing Δ, and $|k|$ and $|l|$ vary between 0 and 2. The final term in (2.7), $\frac{T^2}{\rho}L[F,w]^n$, is the nonlinear portion. Calculating $F^b_{i,j}$ requires a matrix inversion, which is the most computationally heavy calculation in the model. Without the nonlinear calculating, the linear part of the model is explicit and can be calculated efficiently.

A comparison of the audio output from a linear and nonlinear model reveals the reason one would like to include nonlinear calculations even though they may be computationally expensive. One of the biggest differences between a linear plate and a nonlinear plate is the generation of high frequency components in response to an excitation. The spectral results of Bilbao's synthesis can be observed in Figure 2.1. This figure compares the spectral response for a linear plate, a nonlinear plate with a low amplitude excitation, and a nonlinear plate with a higher amplitude excitation. The generation of high frequency components can be clearly seen in the bottom graph. Note that the linear plate is not shown under low and high amplitude excitations as the spectral response would be similar; there would not be a generation of high frequency components no matter how large the amplitude of the excitation.

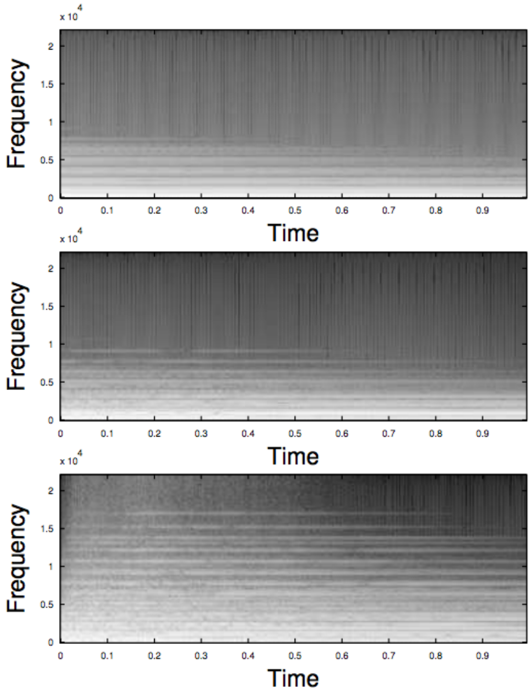


Figure 2.1: Linear and nonlinear thin plate synthesis spectrograms from [7]. Top: linear conditions; middle: nonlinear plate excited with a low amplitude; bottom: nonlinear plate excited with a high amplitude.

Perceptually, the linear model produces a sound similar to a lightly struck bell, while the nonlinear model produces a sound more like a gong with a slight crash and pitch glide. Though nonlinearities may involve more computation, the sonic results are interesting and add a dynamic dimension to the sound. An extension of this research is presented in [9], which focuses on using an energy-conservation method to preserve numerical stability in the nonlinear plate model. The inclusion of longitudinal inertia effects in the plate is covered in [11].

2.6 A Circular, Tension-modulated Nonlinear Membrane using the FTM

The sound synthesis of a circular, tension-modulated nonlinear membrane using the FTM is presented in [5]. The PDE used to model a vibrating circular membrane is

$$D\nabla^4 z(r,\varphi,t) + \sigma \frac{\partial^2 z(r,\varphi,t)}{\partial t^2} - T_0 \nabla^2 z(r,\varphi,t) + d_1 \frac{\partial z(r,\varphi,t)}{\partial t} + d_3 \frac{\partial \nabla^2 z(r,\varphi,t)}{\partial t} = f^{(\mathrm{ext})}(r,\varphi,t). \tag{2.8}$$

$z(r,\varphi,t)$ denotes the transverse displacement of polar coordinate point (r,φ) at time t. T_0 is the surface tension of the membrane, σ is the surface density, d_1 is a frequency-independent dissipation term, d_3 is a frequency-dependent dissipation term, ∇^2 is the Laplacian operator, ∇^4 is the biharmonic operator, and $f^{(\mathrm{ext})}(r,\varphi,t)$ is the driving force density that models a hammer hit.

In order to simulate the pitch glides that occur due to nonlinear tension modulation, (2.8) is modified to use the Berger approximation. The Berger model for nonlinear plates is a simplified version of the nonlinear von Kárman model used in [7, 9, 11]. The Berger model can create pitch glides for membranes but is unable to capture the highly nonlinear behaviors of cymbals and gongs. (2.8) is modified to include a nonlinear tension $T_{\mathrm{NL}}(z)$:

$$D\nabla^4 z + \sigma \frac{\partial^2 z}{\partial t^2} - [T_0 + T_{\mathrm{NL}}(z)]\nabla^2 z + d_1 \frac{\partial z}{\partial t} + d_3 \frac{\partial \nabla^2 z}{\partial t} = f^{(\mathrm{ext})} \tag{2.9}$$

where the spatial and temporal terms have been left out for brevity. The nonlinear tension term has the form

$$T_{\mathrm{NL}}(z) = \frac{Eh}{2\pi R^2(1-\nu^2)} \cdot \int_0^R \int_0^{2\pi} \left[\left(\frac{\partial z}{\partial r} \right)^2 + \frac{1}{r^2} \left(\frac{\partial z}{\partial \varphi} \right)^2 \right] d\varphi \, dr \tag{2.10}$$

where h is the height of the membrane, E is Young's modulus, ν is Poisson's ratio, and R is the radius of the circular membrane. The nonlinear tension term is an integral over the state of the membrane. The double integral can be interpreted as surface area, which means that, in a way, this Berger model is the 2D extension of a nonlinear tension-modulated string model. The real-time membrane model in [48] is also an extension of the same nonlinear tension-modulated string model but rectangular, not circular, in shape.

The nonlinear tension term in (2.9) is moved to the right-hand side of the equation and grouped with the driving force $f^{(\mathrm{ext})}(z)$. Both $f^{(\mathrm{ext})}(z)$ and $T_{\mathrm{NL}}(z)$ are nonlinear and require feedback from the output of the system. In this view, the system is comprised of a linear membrane model under constant tension with two driving forces that require feedback from the state of the membrane, with the nonlinearity captured by the $T_{\mathrm{NL}}(z)$ driving force. From this modified formulation of (2.9), the modal frequencies, decays, and shapes of the membrane are found using the FTM and implemented as a parallel bank of resonating filters.

The modal filters are second-order resonating filters of the form

$$H_{n,m}(s) = \frac{\sigma^{-1}}{s^2 + 2\alpha_{n,m}s + \omega_{n,m}^2} \tag{2.11}$$

where, for normal mode (n,m), $\alpha_{n,m}$ is a loss factor and $\omega_{n,m}$ is the center frequency of the filter. The values for $\alpha_{n,m}$ and $\omega_{n,m}$ can be found using material properties of the membrane. The driving force $f^{(\mathrm{ext})}$ is used as input for these modal filters. In this state, the membrane model is essentially a traditional MS model where the sound can be computed using the modal frequencies, decays, and some type of force input. However, in this example, both $f^{(\mathrm{ext})}(z)$ and $T_{\mathrm{NL}}(z)$ are used as input to the modal filters $H_{n,m}(s)$ and they require feedback from the output of $H_{n,m}(s)$, which represents the membrane displacement z. An example of the signal output by this model can be seen in Figure 2.2, where the downward pitch glide can be clearly seen in the first 0.2 seconds of the signal. Similar to the nonlinear plate FDM described in Section 2.5, the linear portion of this model can run efficiently and the nonlinear portion is more computationally expensive.

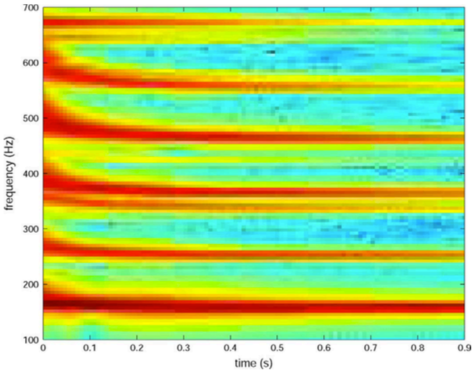


Figure 2.2: Spectrogram showing the synthesized pitch glide of a nonlinear, tension-modulated membrane from [5]. The membrane was struck with a high velocity excitation, and the pitch glides can be clearly seen immediately following the initial impact.

In a following paper [44], the nonlinear feedback calculation is replaced with an efficient approximation that can be calculated with a computational expense similar to that of the linear membrane model. To arrive at that approximation, we begin with the modal formulation of the nonlinear tension (2.10):

$$T_{\mathrm{NL}}(z) = \frac{Eh}{2\pi R^4(1-\nu^2)} \sum_{n,m} \frac{\mu_{n,m}^2 \bar{z}_{n,m}}{||K_{n,m}||_2^2} \tag{2.12}$$

where $\mu_{n,m}$ is a zero of a Bessel function of the first kind and relates to $K_{n,m}$ which are the spatial eigenfunctions of the circular tension. $\bar{z}_{n,m}$ are the modal amplitudes that are related to the membrane displacement $z(r,\varphi,t)$ according to

$$\bar{z}_{n,m} = \int_r \int_\varphi z(r,\varphi,t) K_{n,m}(r,\varphi)\, r\, dr\, d\varphi. \tag{2.13}$$

The nonlinear tension dependence on $\bar{z}_{n,m}$ for every audio sample greatly slows down the model.

In [44], an energy-based approximation for $T_{\mathrm{NL}}(z)$ is found and takes the form

$$T_{\mathrm{qs}}(z) = 2\frac{Eh}{2\pi R^4(1-\nu^2)}e^{-2t/\tau_{m,n}}\sum_{n,m}\frac{\mu_{n,m}^2 A_{n,m}^2}{||K_{n,m}||_2^2}. \tag{2.14}$$

Here, $A_{n,m}^2$ replaces $\bar{z}_{n,m}$ in (2.14) and an exponentially decaying function $e^{-2t/\tau_{m,n}}$ multiplied with the signal. $A_{n,m}^2$ is the initial amplitude of the decaying modal filters of the system and $\tau_{m,n}$ is the decay of the modes. Both $A_{n,m}^2$ and $\tau_{m,n}$ can be found using initial conditions, and $T_{\mathrm{qs}}(z)$ eliminates the need for feedback from the membrane displacement for every audio sample. The researchers report that, with this approximation, the nonlinear model can be computed with computational complexity similar to that of the linear model.

2.7 Experiments with using a DWM for Percussion Synthesis

In this section, we review the DWM in detail and then cover research that moved towards finding an analytical solution for the 2D DWM in order to reduce computational expenses that arise when moving from a 1D digital waveguide to a multidimensional mesh. Through this research, it was discovered that moving from a physics-based representation to an abstract one was more appropriate for my percussion synthesis goals.

2.7.1 DWM for Percussion Synthesis

The basic DWM is composed of bidirectional delay lines interconnected by 4-port scattering junctions. Figure 2.3 is an image of the basic rectilinear 2D DWM. Each scattering junction, labeled with S, has 4 input and 4 output ports. The scattering junctions determine the way in which waves reflect and transmit across the membrane. At the edges of the mesh are reflection filters that dictate how a wave is reflected at the boundary. To apply an excitation, one selects a junction and divides the excitation signal among the junction's input ports. The mesh output signal is the velocity (or pressure) at a specific junction and is calculated as a function of the neighboring junctions' output ports for each time step.

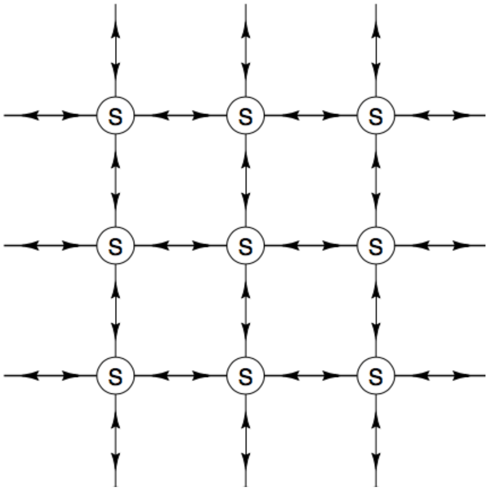


Figure 2.3: 3x3 2D DWM structure from [20]. Each encircled *S* indicates a 4-port scattering junction and the arrows between them are bidirectional delay lines.

With the most basic 2D DWM present in [20], the sounding frequency is related to the size of the DWM, although not much research has been made to map mesh size to a sounding frequency in Hz. The input and output junction points affect the timbre of the DWM output by changing the amplitude levels of the resonant frequencies. Striking the rectangular mesh along the edges, and especially the corners, excites the most resonant frequencies and creates the noisiest, harshest sounds. Decay time and timbre are affected by reflection coefficients or filters at the boundaries of the mesh. Setting all reflection coefficients to -1 creates a perfect, inverting reflection, simulating a membrane, and setting them to 1 creates a perfect, non-inverting reflection, simulating a plate. When the reflection coefficients are set to 1 or -1, there is no decay, and values in between cause the output signal to decay with values closer to 0 creating a quicker decay. The reflection coefficients can be replaced with frequency-dependent loss filters to create more natural decays as the sound dies away. The loss filters are typically implemented using lowpass filters in which higher frequencies decay more quickly.

As noted in Chapter 1, the 2D DWM loses the efficiency gains of the 1D digital waveguide and is computationally inefficient. In efforts to build a real-time percussion synthesis instrument, I, along with my advisor Tamara Smyth and colleague, Ryan Done, set out to find a transfer function that describes the behavior of the 2D DWM. The goal was an efficient implementation of the mesh that allowed it to run in real-time. What we found, however, is that the transfer function that describes the 2D mesh could be decomposed into a sum of resonating filters, which greatly resembles MS. This sparked the decision to move from physics-based percussion synthesis to an abstract modeling synthesis system.

2.7.2 Towards a Real-Time 2D DWM

In an effort to create a real-time digital waveguide mesh implementation, a method for analytically expressing a square, 2D DWM in the frequency domain as a parametric transfer function is presented in [53]. In this work, the input and output points are the center junction, and although this creates a fraction of the total number of modes that the square membrane is capable of creating, it allows us to take advantage of the symmetry and reduces the computational complexity of the system. Instead of computing the transfer function for the entire DWM, one only needs to compute for a half quadrant, requiring $\frac{1}{8}$ of the computation required for the entire DWM.

The transfer function expresses the output of the system in terms of its input and has the following form

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + \cdots + b_M z^{-M}}{a_0 + a_1 z^{-1} + \cdots + a_N z^{-N}} \tag{2.15}$$

for input $X(z)$, output $Y(z)$, and filter coefficients b_i and a_i. By expressing the 2D DWM in this form, not only are the computation times greatly improved, as shown in [53], but the transfer function can be broken down into a bank of 1$^{\text{st}}$- or 2$^{\text{nd}}$-order filters connected in either a parallel or series configuration. When the transfer function is implemented with these lower order filters, it is easier to control time-varying parameters, such as the reflection coefficients.

2.7.3 The 2D DWM as a Parallel Bank of 2$^{\text{nd}}$-Order Sections

The 2D DWM transfer function can be decomposed into 2$^{\text{nd}}$ order filters. These filters can be arranged in a series or parallel configuration, and for completeness, we cover both configurations in this section. A parallel bank of filters is similar to MS and sparked the move from a physics-based percussion synthesis model to an abstract-based model.

Filters in Series

The numerator and denominator of the filter transform function (2.15) can each be factored into a product of 1$^{\text{st}}$-order filters. This is shown in (2.16) where q_i are the roots of the numerator, which we call the zeros and p_i are poles or the roots of the denominator [35].

$$H(z) = \frac{(1-q_1z^{-1})(1-q_2z^{-1})+\cdots+(1-q_Mz^{-1})}{(1-p_1z^{-1})(1-p_2z^{-1})+\cdots+(1-p_Nz^{-1})} \tag{2.16}$$

(2.16) can be rearranged as a product of 1$^{\text{st}}$-order sections of the form

$$H_{s,i}(z) = \frac{1-q_iz^{-1}}{1-p_iz^{-1}}, \tag{2.17}$$

which would be multiplied together as

$$H(z) = \prod_{i=0}^{N-1} H_{s,i}(z) = H_{s,0}(z)H_{s,1}(z)\cdots H_{s,N}(z). \tag{2.18}$$

If the M and N differ, then the filter section $H_{s,i}(z)$ would be a one-pole or one-zero filter. Furthermore, these 1$^{\text{st}}$-order sections could be combined into the commonly used 2$^{\text{nd}}$-order filters, sometimes called a biquad. These would be multiplied together as shown in (2.18). If the filter coefficients are real, then the poles appear in complex conjugate pairs, providing a simple way to combine the 1$^{\text{st}}$-order sections.

Parallel Bank of Filters

The transfer function of the 2D DWM can also be decomposed into a sum of 1$^{\text{st}}$- or 2$^{\text{nd}}$-order filters through a partial fraction expansion [35]. (2.15) can be represented as a sum of one-pole filters

$$H(z) = \sum_{i=1}^{N} H_{p,i}(z) = \sum_{i=1}^{N} \frac{r_i}{1 - p_i z^{-1}} \tag{2.19}$$

where the poles of the transfer function are represented as p_i and the residues of each pole are noted by r_i. This equivalence is true if $M < N$, which is the case for the square 2D DWM discussed in [35]. The poles can be found by factoring the denominator, $A(z)$ and the residues can be found using

$$r_i = (1 - p_i z^{-1}) H_{p,i}(z)|_{z=p_i}. \tag{2.20}$$

If the coefficients for the filter are real, as in the case for the 2D DWM transfer function, then the one-pole filters appear as complex conjugate pairs where every (r, p) for any one-pole filter in (2.19) can be paired with a $(\bar{r}, \bar{p})$. The complex conjugate pairs can be combined into second-order sections

$$H'_{p,i}(z) = \frac{(r + \bar{r}) - (r\bar{p} - \bar{r}p)z^{-1}}{1 - (p + \bar{p})z^{-1} + p\bar{p}z^{-2}}. \tag{2.21}$$

Note that in some cases, like the 3 by 3 mesh example in the next section, there will be a few real poles which do not have complex conjugates. These can be combined as pairs into their own 2$^{\text{nd}}$-order filters or left as 1$^{\text{st}}$-order filters. This implies that the transfer function that describes the output of a single point on the 2D DWM can be expressed as a parallel bank of second-order resonating filters, which is equivalent to a MS representation of the membrane or plate.

Example

For a square, 3 by 3 mesh, the transfer function, using the method given in [35], is

$$H(z) = \frac{2z^{-1} + az^{-3} - a^2z^{-7}}{1 + (1+a)/2z^{-2} - (a+a^2)/2z^{-6} - a^2z^{-8}} \tag{2.22}$$

where a is the reflection coefficient at the boundaries of the mesh and is a value between $[-1,1]$. 8 poles p_i and residues r_i can be found through a partial fraction expansion by factoring the denominator and finding the residues using (2.20). There are three complex conjugate pairs that can be combined and two real poles with angles 0 and π that can be combined together to form 4 second-order sections. The poles and their pairings for $a = -0.999$ are shown in Figure 2.4. The corresponding impulse response for the filters formed by the paired poles and residues of that same mesh is presented in Figure 2.6.

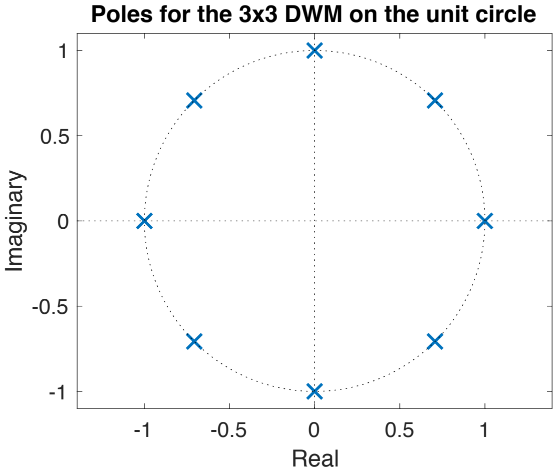


Figure 2.4: Poles plotted for the 3 by 3 DWM on the unit circle. The poles occur in conjugate pairs as indicated by the red outlines and correspond to the resonant frequencies in Figure 2.5.

Moving from the Physical to the Abstract Realm

This section showed that it is possible to express the 2D DWM as a transfer function and that the transfer function can be represented as a bank of parallel second-order sections. This

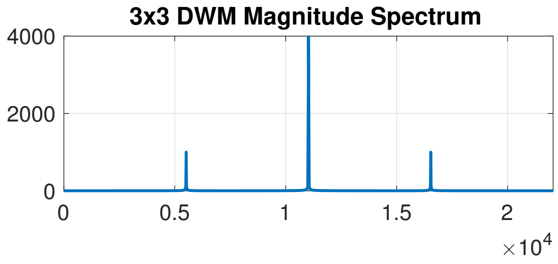


Figure 2.5: Magnitude spectrum for the 3 by 3 DWM. The reflection coefficient is set according to $a = -0.999$. The resonant frequencies correspond to the poles in Figure 2.4.

parallel bank of second-order filters looks quite similar to the filter configuration used in MS. When comparing 2D DWMs with MS, it becomes obvious that for the purposes of this work, MS is much more suitable. First of all the DWM suffers from dispersion error–the modes in the output do not match that of an ideal membrane or plate. With MS, one can take the theoretical modes of a plate or membrane and easily simulate the sound. Furthermore, even if we do not want to create the sound of an ideal plate or membrane, with MS, there is more control over every single spectral component. Access to individual spectral components is not available in a 2D DWM. With MS, we can synthesize a variety of percussive sounds as is shown in [17, 19, 4]. MS is also capable of real-time synthesis, provided that the number of modes is not too large (on the order of thousands), whereas the DWM can only run in real-time on personal computers up to a size of 51 by 51.

A caveat, however, is that traditional MS is not capable of creating the types of sounds that nonlinear percussion instruments create. In order to add spectral richness and pitch glides, we introduce loopback FM oscillators in Chapter 3. Loopback FM oscillators are used in the percussion synthesis MS method described in Chapter 4.

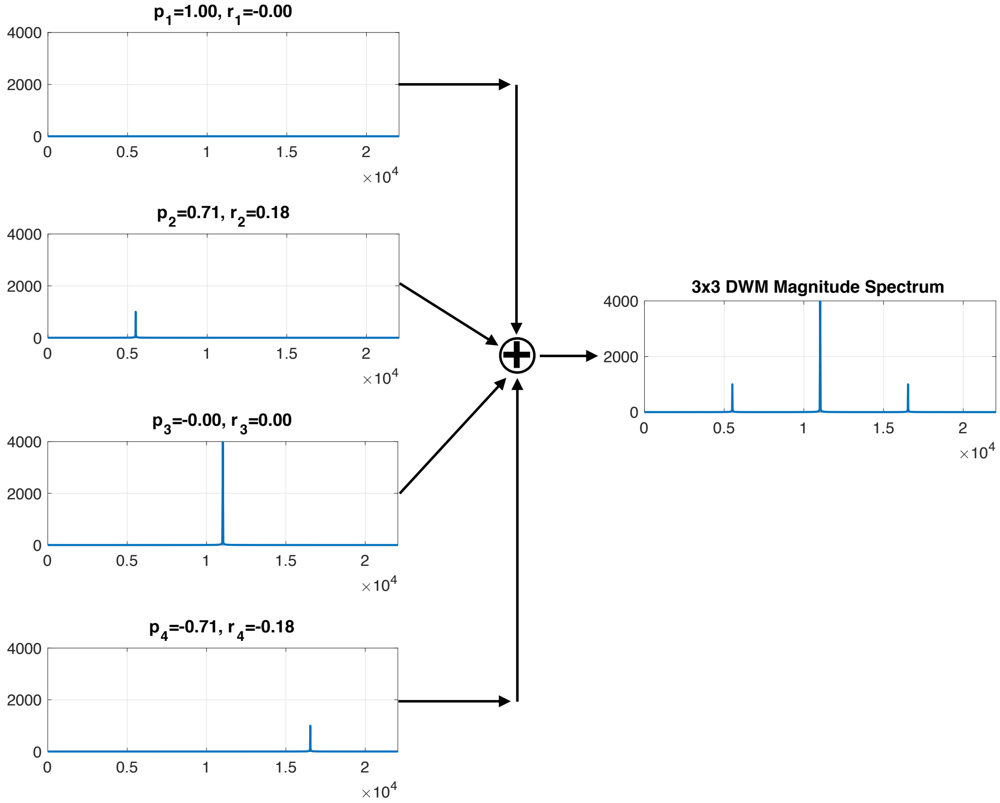


Figure 2.6: Magnitude spectrums for second-order sections add up to 3 by 3 DWM magnitude spectrum. The reflection coefficient is set according to $a = -0.999$. Each second-order section magnitude spectrum figure on the left corresponds to a pair of poles in Figure 2.4.

Chapter 3

Loopback FM

3.1 Introduction

Loopback FM is a self-modulated form of FM where the oscillator loops back and modulates its own carrier frequency according to a feedback coefficient. This differs from Feedback FM [57], in which the output is used to modulate its own initial phase. Loopback FM with a static feedback coefficient, *B*, and feedback FM both create peaks in the spectrum at integer multiples of a sounding frequency. As described in [54], the difference between the two synthesis methods is that with loopback FM, the feedback coefficient *B* can be varied over time to create both predictable pitch and spectral changes. Conversely, feedback FM preserves pitch (in some contexts a desirable feature) and only introduces spectral changes. Loopback FM is also similar to feedback AM as described in [39] and coefficient-modulated allpass filter synthesis in [40].

As shown in [55], loopback FM and its closed-form infinite impulse response (IIR) approximation, an expression that resembles the transfer function of a “stretched” allpass filter [60] but for which only the real part is used as a time-domain signal, can be used to create complex frequency spectra and pitch contours. Here, we begin by presenting the sample-by-sample rotation representation of an oscillator with constant frequency. This same representation is modified to express an oscillator with time-varying frequency, more commonly known as an FM synthesis oscillator. This is extended to feedback FM and then finally, we show how the FM expression can be adjusted to form the loopback FM sample-by-sample rotation formulation. The closed-form loopback FM representation is then introduced. We cover loopback FM representations with static pitch and number of harmonics followed by their time-varying formulations, which can be used to modulate the number of harmonics in the spectrum and sounding frequency.

3.2 An Oscillator with Constant Frequency

An oscillator with constant frequency can be thought of as a point that rotates about the complex plane. Consider a complex point with magnitude A and initial phase ϕ

$$Ae^{j\phi} = \cos(\phi) + j\sin(\phi). \tag{3.1}$$

We can multiply this point by $e^{j\omega T}$ to rotate the point about the complex plane by an angle ωT. Repeatedly multiplying $Ae^{j\phi}$ by $e^{j\omega T}$ n times produces a complex oscillator

$$z(n) = (e^{j\omega T})^n Ae^{j\phi} = Ae^{j\omega nT+\phi}. \tag{3.2}$$

$z(n)$ has angular frequency ω and initial phase ϕ. n is the time sample and $T = 1/f_\mathrm{s}$ is the sampling period for sampling rate f_s. $z(n)$ can be equivalently expressed by its sample-by-sample rotation representation

$$z(n) = e^{j\omega T} z(n-1) \tag{3.3}$$

where the value of the oscillator at the previous time step, $z(n-1)$, is rotated by $e^{j\omega T}$.

3.3 FM Formulation and Frequency Component Analysis

For an oscillator with time-varying frequency, (3.3) is modified so that the instantaneous frequency, ω, is now a function of time sample n:

$$z(n) = e^{j\omega(n)T} z(n-1). \tag{3.4}$$

With frequency modulation synthesis, as introduced in [15], $\omega(n)$ is expressed as

$$\omega(n) = \omega_{\mathrm{c}} + d\cos(\omega_{\mathrm{m}} nT) \tag{3.5}$$

where $\omega_{\mathrm{c}} = 2\pi f_{\mathrm{c}}$ is the carrier frequency, d is the peak frequency deviation, and $\omega_{\mathrm{m}} = 2\pi f_{\mathrm{m}}$ is the modulating frequency. ω_{c} and ω_{m} are both angular frequencies, and f_{c} and f_{m} are those same frequencies in Hz. FM can then be expressed in its sample-by-sample rotation form as

$$z_{\mathrm{fm}}(n) = e^{j(\omega_{\mathrm{c}} + d\cos(\omega_{\mathrm{m}} nT))T} z_{\mathrm{fm}}(n-1). \tag{3.6}$$

For implementation purposes, the initial value of $z_{\mathrm{fm}}(n)$ should be set to 1.

Note that FM synthesis is typically implemented in its phase modulation (PM) form

$$x_{\mathrm{fm}}(n) = \cos(\omega_{\mathrm{c}} nT + I\sin(\omega_{\mathrm{m}} nT)), \tag{3.7}$$

where I is the index of modulation, and (3.5) is merely a different representation of this familiar FM synthesis equation. The instantaneous phase of $x_{\mathrm{fm}}(n)$ is the expression within the $\cos(\cdot)$ function:

$$\theta(n) = \omega_{\mathrm{c}} nT + I\sin(\omega_{\mathrm{m}} nT). \tag{3.8}$$

The instantaneous frequency is the time derivative of $\theta(n)$ and computes to (3.5) with $d = \omega_{\mathrm{m}} I$.

The frequency components for a signal created using FM synthesis appear at $f_c \pm i f_m$ for integer values of i. The amplitude level of the frequency components can be expressed using Bessel functions of the first kind. As I, the index of modulation increases, the bandwidth of the sidebands increases, and aliased frequency components will appear in the spectrum. A plot of the magnitude spectrum for an FM signal created using $f_c = 200$, $f_m = 150$, and $I = 1$ is shown in Figure 3.1. A large peak can be found along the dashed red line, which indicates the carrier frequency $f_c = 200$. The other peaks appear at $f_c \pm i f_m$ which, for the given variable values, are 50, 350, 500, and 650.

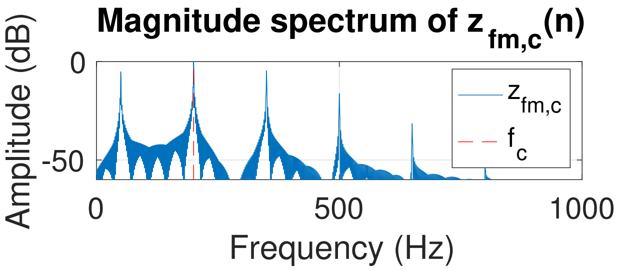


Figure 3.1: Magnitude spectrum of FM synthesis. $f_c = 200$, $f_m = 150$, and $I = 1$. Frequency components appear at $f_c \pm i f_m$.

3.4 Feedback FM Formulation and Frequency Component Analysis

Feedback FM was presented in [57] as an extension of FM where the output of the oscillator is used to modulate the phase. Like FM, feedback FM uses a carrier frequency ω_c but unlike FM, feedback FM does not make use of a modulating frequency, as the modulation occurs through the feedback. In [30], the discrete form of feedback FM is given as

$$x_{fb}(n) = \sin(\omega_c nT + B x_{fb}(n-1)) \tag{3.9}$$

where B specifies the amount of feedback. B is typically kept between values of -1 and 1 so that the spectral components decay monotonically as a function of frequency.

In complex exponential form, feedback FM can be expressed as

$$z_{\text{fb}}(n) = e^{j(\omega_c nT + B\Im\{z_{\text{fb}}(n-1)\})} \tag{3.10}$$

where $x_{\text{fb}}(n) = \Im\{z_{\text{fb}}(n)\}$. The phase of (3.10) is

$$\theta(n) = \omega_c nT + Bx_{\text{fb}}(n-1). \tag{3.11}$$

The time derivative of this is the instantaneous frequency and can be expressed in discrete form as

$$\omega(n) = \omega_c + \frac{B}{T}\Im\{z_{\text{fb}}(n-1) - z_{\text{fb}}(n-2)\} \tag{3.12}$$

where within the $\Im\{\cdot\}$ part of the equation, we use a difference to approximate the derivative. (3.12) can be used in (3.4) to form the sample-by-sample rotation representation for feedback FM to obtain

$$z_{\text{fb}}(n) = e^{j(\omega_c + \frac{B}{T}\Im\{z_{\text{fb}}(n-1) - z_{\text{fb}}(n-2)\})T} z_{\text{fb}}(n-1). \tag{3.13}$$

When implementing this, if the first sample begins at $n = 0$, then $z_{\text{fb}}(-1)$ should be set to 1 and $z_{\text{fb}}(-2)$ should be initialized to 0.

There are strong resemblances between the FM (3.6) and Feedback FM (3.13) equations. With feedback FM in (3.13), however, the second term in the complex exponential is a function of the oscillator itself, which contributes to the feedback. Contrarily, with the FM form in (3.6), the second term in the complex exponential is the modulating oscillator and there is no feedback.

The spectral components of feedback FM are centered at integer multiples of the carrier frequency, if_c for integer i, and, similar to FM, the amplitude of those components are proportional to Bessel functions of the first kind [57]. Figure 3.2 graphs an example of feedback FM synthesis output. The red line indicates the carrier frequency $f_c = 200$. Frequency components can be seen at $f_c = 200$ and at integer multiples of f_c. The amplitude level of the spectral components decreases monotonically after the tallest peak at f_c.

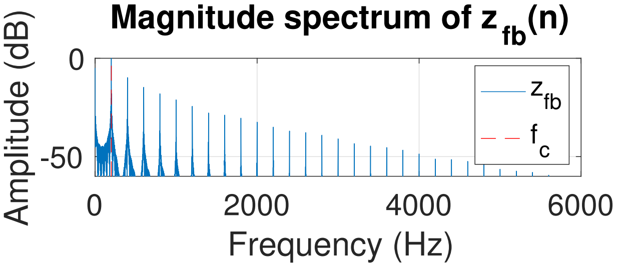


Figure 3.2: Magnitude spectrum of feedback FM synthesis. $f_c = 200$ and $B = 0.9$. Frequency components appear at integer multiples of the carrier frequency: if_c.

3.5 Loopback FM Formulation

In this section, we examine loopback FM, in which the output is fed back to modulate the instantaneous frequency of an oscillator. The equations take on a similar form as the sample-by-sample rotation equations for FM (3.6) and feedback FM (3.13). The loopback FM equation involves carrier frequency $\omega_c = 2\pi f_c$ and a feedback parameter B, which controls the fundamental frequency and the number of harmonics in the spectrum of the output, the latter of which greatly influences the timbre of the output.

The sample-by-sample loopback FM equation for time sample n with static B is

$$z_c(n) = e^{j(\omega_c + \omega_c B \Re\{z_c(n-1)\})T} z_c(n-1), \tag{3.14}$$

For implementation, the initial condition $z_c(0) = 1$ causes oscillation. The output that we listen to is the real part of $z_c(n)$. The fundamental frequency of this oscillator is not ω_c but rather $\omega_0 = 2\pi f_0$, where f_0 is the sounding frequency in Hz. The relationship between ω_0 and ω_c is described by

$$\omega_0 = \omega_c\sqrt{1-B^2}, \quad -1 \leq B \leq 1, \tag{3.15}$$

which shows that for ω_0 to remain real, the value of B must be within the interval $(-1,1)$.

The equation for loopback FM (3.14) is closely related to that of FM (3.6) and feedback FM (3.13), but there are key differences in the instantaneous frequencies. The instantaneous frequency for loopback FM is

$$\omega(n) = \omega_c + \omega_c B\Re\{z_c(n-1)\}. \tag{3.16}$$

When compared to FM (3.5), loopback FM replaces the modulating oscillator with the real part of the output at the previous time sample. This is the "loopback" portion of loopback FM. Comparing feedback FM with loopback FM, the second term of (3.12) involves a difference of the oscillator value at previous time samples while the second term of (3.16) only considers the last output sample. These differences make it easy to control pitch glides when the loopback FM equation is made time-varying as will be shown in Sections 3.8 and 3.9.

(3.6), (3.13), and (3.14) are the sample-by-sample rotation formulations for FM, feedback FM, and loopback FM, respectively. They are numerical approximations and suffer from numerical error due to the sample-by-sample rotation. For this reason, FM is typically implemented using its PM form (3.7). The loopback FM sample-by-sample rotation expression tends to experience even more error due to the unit delay in the feedback loop. In the next section, a more numerically accurate representation of loopback FM is presented.

3.6 The Closed-Form Representation of the Loopback FM Oscillator

The loopback FM oscillator $z_c(n)$ given in (3.14) with static pitch and number of harmonics may also be represented by the closed-form representation

$$z_0(n) = \frac{b_0 + e^{j\omega_0 nT}}{1 + b_0 e^{j\omega_0 nT}}, \tag{3.17}$$

which is similar to the transfer function of a "stretched" allpass filter used in [60]. In this synthesis context, (3.17) is used as a time-domain signal that is a function of time sample n, where b_0 influences spectrum, ω_0 specifies the sounding frequency, and the output sound is the real signal given by $\Re\{z_0(n)\}$. Parameters b_0 and ω_0 are related to the loopback FM feedback coefficient B. With (3.17), the number of harmonics and pitch can be independently controlled, but this is not possible with parameters ω_c and B given in (3.14). Coefficient b_0 in $z_0(n)$ is related to B through

$$b_0 = \frac{\sqrt{1-B^2}-1}{B}. \tag{3.18}$$

Note the singularity in (3.18) for $B = 0$. The relationship between ω_0 and ω_c is shown in (3.15).

3.7 Parametric Control of Loopback FM with Static Pitch and Number of Harmonics

The magnitude spectrum of a loopback FM signal generated using (3.14) is shown in Figure 3.3. Here, the parameters are set to the same values as they are for the feedback FM example in Figure 3.2 with $f_c = 200$ and $B = 0.9$. The biggest difference between the two is that the peaks do not appear at integer multiples of f_c, as they do for feedback FM, rather, they occur at integer multiples of sounding frequency $f_0 = f_c\sqrt{1-B^2}$. The sidebands also decay logarithmically.

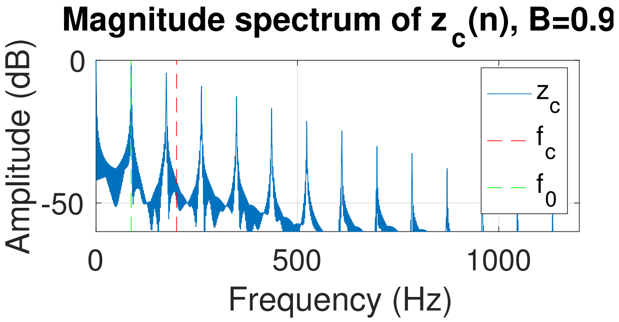


Figure 3.3: Magnitude spectrum of loopback FM synthesis with $B = 0.9$. $f_c = 200$ and frequency components appear at integer multiples of a sounding frequency: $f_0 = f_c\sqrt{1-B^2}$.

As B decreases toward 0 in Figures 3.4 to 3.5, two things begin to occur: 1) the number of sidebands decreases and 2) the sounding frequency f_0 approaches the carrier frequency f_c. By time-varying the feedback coefficient B, we are able to create pitch contours and spectral effects.

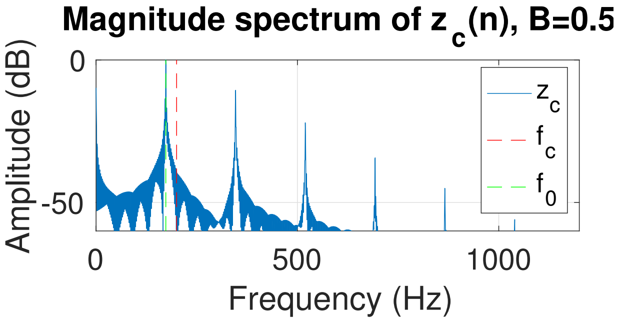


Figure 3.4: Magnitude spectrum of loopback FM synthesis with $B = 0.5$. $f_c = 200$ and compared to Figure 3.3, there are fewer sidebands and f_0 is closer to f_c

With the closed-form loopback FM equation, sounding frequency and the number of harmonics can be controlled independently of one another. $\omega_0 = 2\pi f_0$ directly controls the sounding frequency and b_0 affects the number of harmonics. B and b_0 affect the number of harmonics in the same way: when they are close to 1 or -1, there are more high frequency components and the timbre is brighter, and when they are close to 0, there are fewer frequency components and the sound is darker and more muffled. When B and b_0 are negated, there is no

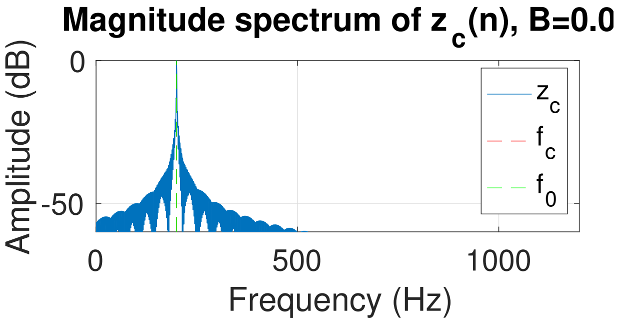


Figure 3.5: Magnitude spectrum of loopback FM synthesis with $B = 0.0$. $f_c = 200$ and compared to Figures 3.3 and 3.4, there is only a single frequency component centered at $f_0 = f_c$.

effect on the sounding frequency or spectral sidebands, so for a signal formed using B or b_0 will be equivalent to a signal formed using $-B$ or $-b_0$. The relationship between B and b_0 is shown in Figure 3.6 and given in (3.18).

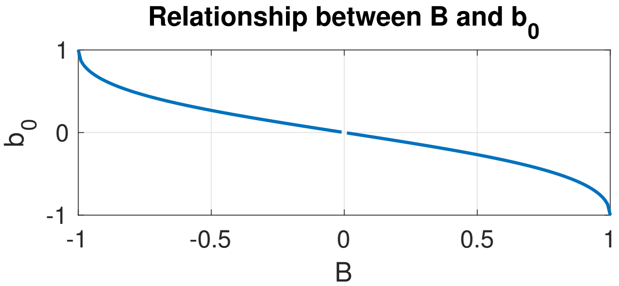


Figure 3.6: Relationship between B and b_0. B and b_0 both affect the number of harmonics in similar ways and their relationship is given by (3.18).

When comparing the two loopback FM formulations, z_0 and z_c, results are identical when ω_c is small and B is farther from 1 (or -1). When B is closer to 1 and ω_c is large, z_c begins to alias and experience spectral component foldover. This effect is shown in Figure 3.7 where the graph of z_c shows lower peaks in between taller peaks, where the taller peaks match those in the graph of z_0. If ω_c is set to extremely large values, such as 12 kHz with a sampling rate of 44.1 kHz, z_c begins to create noisy signals. If the sampling rate is increased, the output of z_c approaches that of z_0 even for high values of ω_c and B near 1. Like FM, both z_c and z_0 produce signals that are not bandlimited. With loopback FM, a large ω_c means a large feedback amount, which can result in increased bandwidth and aliasing, similar to how a large index of modulation corresponds to a wider bandwidth in traditional FM.

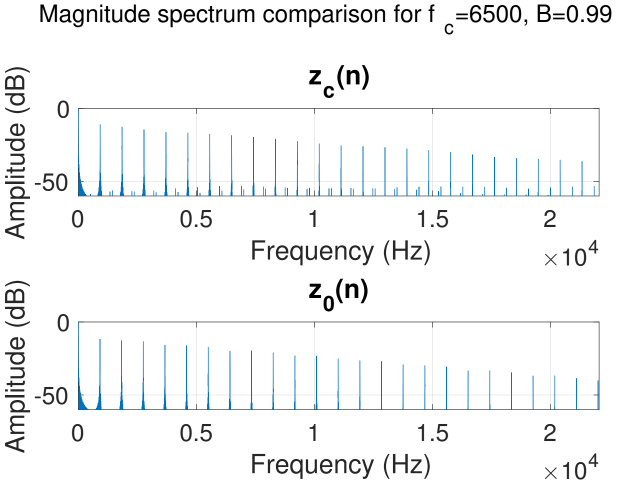


Figure 3.7: Magnitude spectrum of loopback FM: z_c vs. z_0 for large ω_c and B. The effects of aliasing shows up on the z_c graph as shorter peaks in between taller peaks.

3.8 Time-varying B: Modulation of Pitch and Number of Harmonics with $z_c(n)$

In (3.14), the feedback coefficient B can be varied over time between $(-1,1)$ to create pitch glides and changes to the number of harmonics over the length of the output signal. From (3.14), B is replaced by $B(n)$ to form

$$z_c(n) = e^{j\omega_c T(1+B(n)\Re\{z_c(n-1)\})} z_c(n-1) \tag{3.19}$$

(3.15) reveals that when B is made to vary over time, ω_0 also becomes time-varying. This creates a pitch trajectory where the sounding frequency follows the equation

$$\omega_0(n) = \omega_c\sqrt{1-B^2(n)}. \tag{3.20}$$

3.9 Time-varying b_0 and ω_0: Modulation of Pitch and Number of Harmonics with $z_0(n)$

Like (3.19), the parameters of z_0 in (3.17) can be made to vary over time to create pitch glides and spectral changes. Parameter b_0 can be mapped to $B(n)$ by

$$b_0(n) = \frac{\sqrt{1 - B^2(n)} - 1}{B(n)} \tag{3.21}$$

and used in (3.22) to create spectral variations.

A desired pitch contour can be created by setting $\omega_0(n)$ to a pitch trajectory in the form of (3.20). Directly using $\omega_0(n)$ in place of ω_0 in (3.17) will not result in the desired pitch glide, and it is necessary to use a generalization of (3.17):

$$z_0(n) = \frac{b_0(n) + e^{j\theta_0(n)}}{1 + b_0(n)e^{j\theta_0(n)}}. \tag{3.22}$$

To understand $\theta_0(n)$, the instantaneous phase of the complex exponential terms in (3.22), let $\omega_0(t)$ be the continuous counterpart of $\omega_0(n)$ serving as the instantaneous frequency, and $\theta_0(t)$, its integral with respect to time:

$$\theta_0(t) = \int_0^t \omega_0(t)\mathrm{d}t. \tag{3.23}$$

Examples of the discrete-time form of (3.23) given by $\theta_0(n)$ as used in (3.22), are shown in Section 4.4.5.

If one is interested in varying the number of harmonics, but maintaining a static sounding frequency ω_0, an alternate form excluding $\theta_0(n)$ may be used:

$$z_0(n) = \frac{b_0(n) + e^{j\omega_0 nT}}{1 + b_0(n)e^{j\omega_0 nT}}. \tag{3.24}$$

3.10 Parametric Control of Loopback FM with Time-varying Pitch and Number of Harmonics

An example of a spectrogram for exponentially decreasing pitch and time-varying number of harmonics for $z_c(n)$ and $z_0(n)$ is shown in Figure 3.8. The graph compares an exponentially decreasing pitch glide from 200 Hz to 140 Hz for $z_c(n)$ and $z_0(n)$ where the pitch glide is indicated by the red line on each plot. Because the pitch glide is specified using $B(n)$ for $z_c(n)$, and the number of harmonics is directly affected by $B(n)$, this pitch glide will always be paired with that spectral pattern for the top graph of $z_c(n)$. In the bottom graph, the number of harmonics is controlled by setting $b_0(0)$ to linearly increase from 0.001 to 0.999 over the course of the signal so that higher frequencies appear later on in the signal rather than right at the beginning. Control of the pitch glides and the number of harmonics are fundamental to the percussion synthesis method that is covered in Chapter 4.

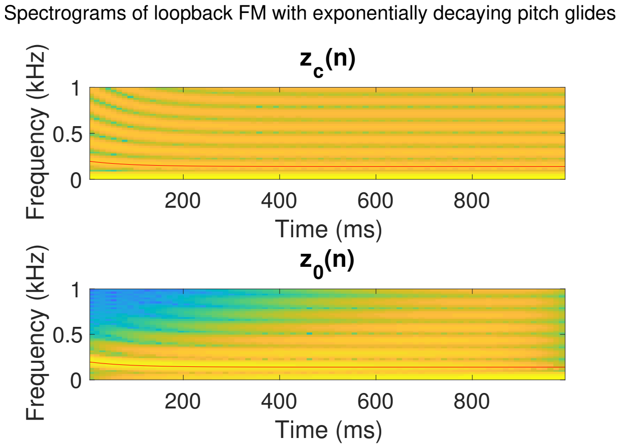


Figure 3.8: Spectrograms of z_c vs. z_0 with time-varying pitch and number of harmonics. The red line indicates an exponentially decreasing pitch glide from 200 Hz to 140 Hz. The number of harmonics in the spectrum can be controlled independently of pitch with b_0 in the bottom plot of z_0, while the number of harmonics and pitch are both tied to B in the top plot displaying z_c.

Pitch glides and time-varying harmonics created using $B(n)$ and $z_c(n)$ oscillators are constrained by $B(n)^2 \leq 1$. This can be seen in (3.20) which shows that in order for the sounding frequency $\omega_0(n)$ to remain real, $B(n)$ must abide by the given constraint. Pitch glides formed using $\omega_0(n)$ and $z_0(n)$ oscillators are not affected by the constraint. With the $z_{0,i}(n)$ oscillators, the only constraint is that $\omega_0(n)$ must be integrable so that $\theta_0(n)$ can be obtained and used in (3.22). Likewise, harmonic variations created with $b_0(n)$ and $z_0(n)$ oscillators can take on any form and do not need to follow the constraint on $B(n)$.

Here, we first introduce pitch glides formed using linear and exponential $B(n)$ and $z_{\mathrm{c}}(n)$ oscillators. We demonstrate how to create an identical pitch glide using $z_0(n)$ oscillators. Then, we discuss how linear, exponential, and square root pitch contours can be formed with $\omega_0(n)$ for $z_0(n)$ oscillators. The equivalent pitch glide (with constraints) is then described for $z_{\mathrm{c}}(n)$ oscillators. Last, controlling the number of harmonics over time is illustrated for $b_0(n)$ and $z_0(n)$ oscillators. The equivalent time-varying harmonic variation is shown for $B(n)$ with $z_{\mathrm{c}}(n)$ oscillators. Because $B(n)$ changes both pitch and the number of harmonics for $z_{\mathrm{c}}(n)$ oscillators, control of the number of harmonics for $z_{\mathrm{c}}(n)$ is not separately covered.

3.10.1 Pitch Glides with $B(n)$

Pitch Glides with Exponential $B(n)$

For the exponential case, $B(n) = g^n$ can be used directly in (3.19) to produce a pitch glide. g is typically set to a number close to 1. When using (3.22) for a pitch glide, $\theta_0(n)$ can be found using $B(n) = g^n$ along with (3.20) and (3.23) which, as shown in [55], is given by

$$\theta_0(n) = \frac{\omega_{\mathrm{c}}}{\log(g)}(\sqrt{1-g^{2n}} - \tanh^{-1}(\sqrt{1-g^{2n}})) + C \tag{3.25}$$

where C is the constant of integration.

Pitch Glides with Linear $B(n)$

For the linear case, $B(n) = l_0 + l_1 n$ produces a pitch glide when used with (3.19). (3.22) uses the instantaneous phase given by

$$\theta_0(n) = \frac{\omega_c T}{2l_1}((l_1 n + l_0)\sqrt{1-(l_1 n + l_0)^2} \\ + \sin^{-1}(l_1 n + l_0)) + C \tag{3.26}$$

3.10.2 Pitch Glides with $\omega_0(n)$

Rather than making pitch glides by setting $B(n)$, it may be more musically intuitive to specify pitch glides that change linearly, exponentially, or by a square root function. For each of these examples, the pitch trajectory, $f_0(n)$ will move from frequency f_x to f_y in Hz in t_d seconds, and b_0 is held constant at -0.3 so that the effects of varying the pitch without affecting the number of harmonics can be observed for $z_0(n)$. Pitch trajectories other than the ones shown here can be used with $z_0(n)$ as long as the continuous time pitch glide function $\omega_0(t)$ is integrable.

Linear Pitch Glides

For a linearly increasing or decreasing pitch glide, as shown in Figure 3.9, the sounding frequency will have the equation

$$f_0(n) = l_0 + l_1 nT \tag{3.27}$$

where l_1 is the slope and l_0 is the y-intercept. Let t_d be the time it takes for the pitch glide to move from f_x to f_y. Then l_1 can be calculated as

$$l_1 = \frac{f_\text{x} - f_\text{y}}{t_d}. \tag{3.28}$$

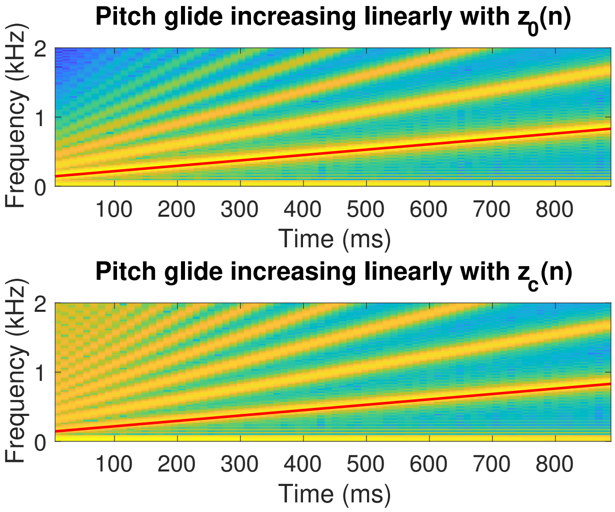


Figure 3.9: Linearly increasing pitch glide for $z_0(n)$ and $z_c(n)$. The upper spectrogram is formed using the time-varying closed-form loopback FM oscillator $z_0(n)$. The pitch trajectory, as shown by the overlain red line, begins at 140 Hz and increases linearly to 840 Hz. Although the pitch trajectory in the bottom plot for the $z_c(n)$ oscillator is the same as the pitch glide in the upper plot, the bottom plot shows more higher frequency content because with $z_c(n)$ oscillators, pitch and number of harmonics are both affected by B.

l_0 can be computed by rearranging (3.27) and solving for l_0 using $f_0(0) = f_x$ or $f_0(t_d) = f_y$ and the value for l_1 given by (3.28). Using the values computed for l_0 and l_1, $f_0(n)$ can be calculated for all values of n. Then, $\omega_0(n) = 2\pi f_0(n)$ can be used with either $z_c(n)$ or $z_0(n)$.

Linear Pitch Glides with $z_0(n)$: To set a pitch glide with $z_0(n)$, we must calculate $\theta_0(n)$ and use it in (3.22) as described in Section 3.9. In this case, $\theta_0(n)$ can be computed using

$$\theta_0(n) = 2\pi \left(\frac{l_1 (nT)^2}{2} + l_0 nT \right). \tag{3.29}$$

Linear Pitch Glides with $z_c(n)$: To set a pitch glide with $B(n)$ and $z_c(n)$ given $\omega_0(n)$, we must calculate ω_c. In order for $B(n)$ to remain real, there is a constraint that $\omega_0(n) \le \omega_c$ for all n. In many examples, it was found that setting ω_c equal to the largest value of $\omega_0(n)$ creates perceptually pleasing sonic results. After setting ω_c, $B(n)$ can be found using

$$B(n) = \sqrt{1 - \left(\frac{\omega_0(n)}{\omega_c}\right)^2} \tag{3.30}$$

Then, $B(n)$ and ω_c can be used in the time-varying, sample-by-sample loopback FM equation, (3.19), to create the desired pitch glide.

Exponentially Decreasing Pitch Glides

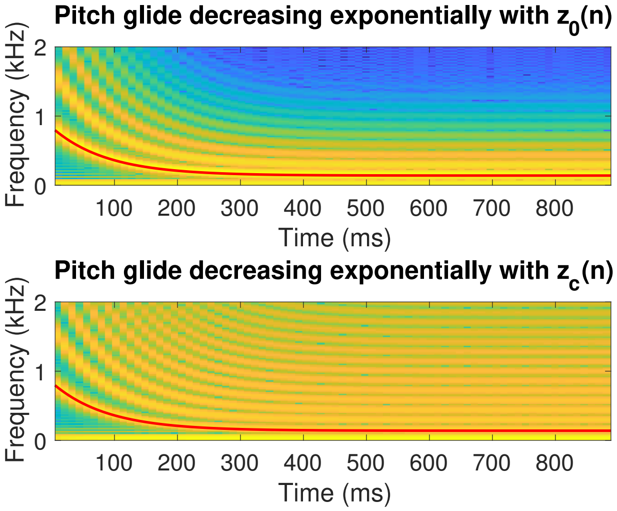


Figure 3.10: Exponentially decreasing pitch glide for $z_0(n)$ and $z_c(n)$. The upper spectrogram is formed using the $z_0(n)$ loopback FM formulation while the lower spectrogram is made using the $z_c(n)$ formulation. The pitch glide is marked by the red line, begins at 840Hz, and decreases exponentially to 140 Hz. The lower spectrogram has more energy in the higher frequencies because B affects both pitch and number of harmonics with the $z_c(n)$ formulation.

For an exponentially decreasing pitch glide, a function that decays exponentially from 1 to a value close to 0 is first created with a desired decay rate. That exponentially decaying function is then scaled to match the pitch glide specifications. An example is shown in Figure 3.10. The exponentially decaying function is set according to

$$h(n) = A_0 e^{-nT/\tau_h}. \tag{3.31}$$

A_0 is the initial value at time sample $n = 0$ and τ_h is the decay constant. The value of this function at $n = 0$ is $h(0) = A_0 = 1$. Let A_1 be the value that $h(n)$ will decrease towards and $n_d = t_d f_s$ be the time (in samples) that it takes for $h(n)$ to reach A_1 from its initial value A_0. A_1 should be set close to 0, and the examples in this document use 0.001, as this value is used when setting reverberation decay times. Using $h(0) = A_0 = 1$ and $h(n_d) = A_1 = 0.001$, (3.31) can be rearranged and τ_h can be found using

$$\tau_h = -\frac{n_d T}{\log(A_1/A_0)}. \tag{3.32}$$

$h(n)$ is computed using A_0 and τ_h. Then $h(n)$ can be scaled using f_x and f_y to obtain

$$f_0(n) = (f_\mathrm{x} - f_\mathrm{y})h(n) + f_\mathrm{x}. \tag{3.33}$$

$\omega_0(n) = 2\pi f_0(n)$ can be used with loopback formulations $z_\mathrm{c}(n)$ or $z_0(n)$ to create the desired exponential decay.

Exponentially Decreasing Pitch Glides with $z_0(n)$: $\theta_0(n)$ can be calculated according to

$$\theta_0(n) = -2\pi(f_x - f_y)\tau_h A_0 e^{-nT/\tau_h} + 2\pi f_y nT \tag{3.34}$$

$\theta_0(n)$ can then be used in (3.22) to produce the desired pitch glide.

Exponentially Decreasing Pitch Glides with $z_\mathbf{c}(n)$: Following the constraint that $\omega_0(n) \leq \omega_\mathrm{c}$ for all n, ω_c can be set to the largest value of $\omega_0(n)$ which is the first value, $\omega_0(0)$. After setting ω_c, $B(n)$ can be found using (3.30). Both of these parameters can be used in (3.19), to create the desired pitch glide.

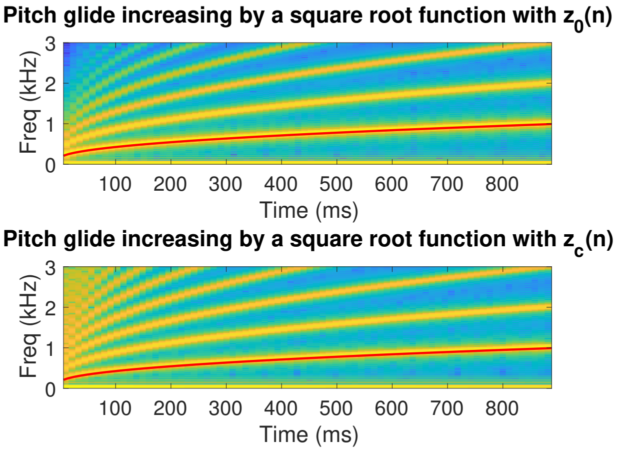


Figure 3.11: Increasing pitch glide using a square root function for $z_0(n)$ and $z_c(n)$. The upper spectrogram is formed using $z_0(n)$, and the lower spectrogram is made using $z_c(n)$. The pitch glide, indicated by the red line in both spectrograms, begins at 140 Hz and increases according to a square root function to 840Hz. As is the case with Figures 3.9 and 3.10, the bottom graph has more higher frequency energy because B affects both number of harmonics and pitch with the $z_c(n)$ oscillator.

Pitch Glides Increasing with a Square Root Function

An example of a pitch trajectory that increases according to a square root function is shown in Figure 3.11. To create a pitch glide that increases from f_x to f_y according to a square root function, the sounding frequency trajectory is modeled as

$$f_0(n) = f_{xy}\sqrt{nT} + f_x, \tag{3.35}$$

where

$$f_{xy} = \frac{f_y - f_x}{\sqrt{t_d}}. \tag{3.36}$$

As it is used in the exponential pitch glide equation, t_d is the time (in seconds) it will take for $f_0(n)$ move from f_x to f_y. Setting $\omega_0(n) = 2\pi f_0(n)$ sets us up to create the pitch glide with $z_c(n)$ or $z_0(n)$.

Square Root Increasing Pitch Glides with $z_0(n)$: $\theta_0(n)$ is calculated as

$$\theta_0(n) = 2\pi\left(\frac{2}{3}f_{\mathrm{xy}}(nT)^{3/2} + f_{\mathrm{x}}nT\right). \tag{3.37}$$

(3.22) paired with $\theta_0(n)$ can be used to create the specified pitch glide.

Square Root Increasing Pitch Glides with $z_{\mathbf{c}}(n)$: ω_{c} should first be set to the largest value of $\omega_0(n)$ or any value smaller than the largest value of $\omega_0(n)$. After setting ω_{c}, $B(n)$ can be found using ω_{c} and $\omega_0(n)$ along with (3.30). Both ω_{c} and $B(n)$ can then be used in (3.19) to create the specified pitch glide.

3.10.3 Varying the Number of Harmonics with $b_0(n)$

This section shows how varying $b_0(n)$ linearly and exponentially affects the number of harmonics in the spectrum of the output of the closed-form loopback FM oscillator $z_0(n)$. How to obtain the equivalent harmonic variation with a $z_{\mathrm{c}}(n)$ oscillator is also explained. In these examples b_{x} is the starting value for $b_0(n)$, b_{y} is the ending value, and f_0 is set to 1000Hz. As used above in the pitch glide examples, $n_{\mathrm{d}} = t_{\mathrm{d}}f_{\mathrm{s}}$ is the time it takes for $b_0(n)$ to change from b_{x} to b_{y}.

Varying the Number of Harmonics with Linear $b_0(n)$

A linear $b_0(n)$ can be modeled according to

$$b_0(n) = l_0 + l_1 n \tag{3.38}$$

with slope l_0 set as

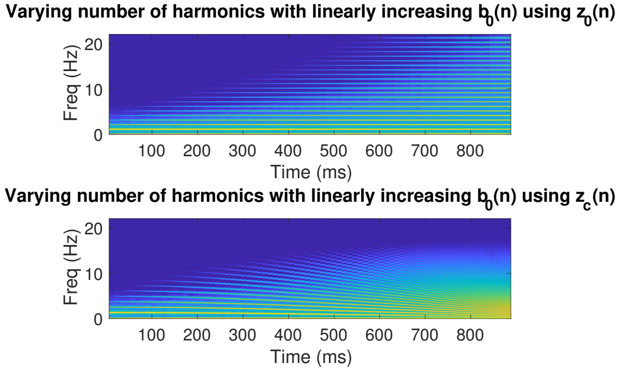
$$l_0 = \frac{b_{\mathrm{y}} - b_{\mathrm{x}}}{n_d}, \tag{3.39}$$

and y-intercept $l_0 = b_\mathrm{x}$. Because the sounding frequency f_0 is held constant, (3.24) can be used with $b_0(n)$ to create the closed-form loopback FM signal with time-varying number of harmonics.

The $B(n)$ function that corresponds to the linear $b_0(n)$ function can be found using

$$B(n) = -\frac{2b_0(n)}{b_0(n)^2 + 1}. \tag{3.40}$$

$B(n)$ can be directly used in (3.19) to create a loopback FM signal that varies in pitch and number of harmonics over time. An example of $z_0(n)$ with a linearly increasing $b_0(n)$ and the corresponding $z_\mathrm{c}(n)$ is plotted in Figure 3.12. The linear $b_0(n)$ function and corresponding $B(n)$ function used to create Figure 3.12 is shown in Figure 3.13.


Varying number of harmonics with linearly increasing b_0(n) using z_0(n)

Varying number of harmonics with linearly increasing b_0(n) using z_c(n)

Figure 3.12: Varying the number of harmonics with linearly increasing $b_0(n)$ for $z_0(n)$ and $z_\mathrm{c}(n)$. The upper plot is the output of a $z_0(n)$ oscillator while the lower plot is the output of a $z_\mathrm{c}(n)$ oscillator. The two plots show how the number of harmonics can be varied over time. As $b_0(n)$ increases, more frequency components appear in the upper spectrogram. Pitch is affected in the lower plot as B affects both number of harmonics and pitch with $z_\mathrm{c}(n)$.

Varying the Number of Harmonics with Exponential $b_0(n)$

Similar to the exponentially decaying pitch glide, an exponentially decreasing $b_0(n)$ can be modeled by scaling and shifting the exponentially decaying function (3.31), which ranges from a value close to 0 up to 1. As was explained in Section 3.10.2, $h(0) = A_0 = 1$ and $h(n_d) = A_1 = 0.001$. The decay constant τ_h can be found according to (3.32). $h(n)$ can then be

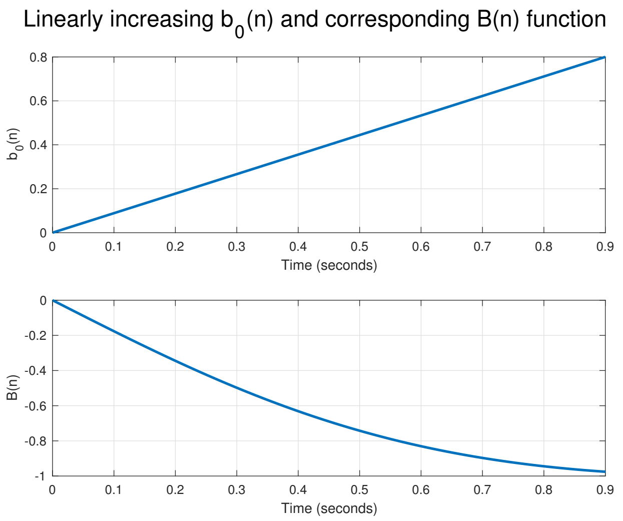


Figure 3.13: Linearly increasing $b_0(n)$ and corresponding $B(n)$. The $b_0(n)$ and $B(n)$ functions plotted here were used to synthesize the signals plotted in Figure 3.12. As $b_0(n)$ increases linearly, $B(n)$ decreases according to (3.40).

scaled using b_x and b_y to obtain

$$b_0(n) = (b_\mathrm{x} - b_\mathrm{y})h(n) + b_\mathrm{y}. \tag{3.41}$$

$b_0(n)$ can be used with (3.24) to modify the number of harmonics using the closed-form loopback FM oscillator.

The $B(n)$ function that corresponds to the exponentially decreasing $b_0(n)$ function can be found using (3.40). To create the time-varying loopback FM signal, $B(n)$ can be directly used with (3.19). This signal will vary in pitch and time. The varying of the number of harmonics with an exponentially decreasing $b_0(n)$ with $z_0(n)$ oscillator along with the corresponding $z_\mathrm{c}(n)$ is shown in Figure 3.14. The exponentially decreasing $b_0(n)$ and corresponding $B(n)$ functions used to plot (3.14) are presented in Figure 3.15.

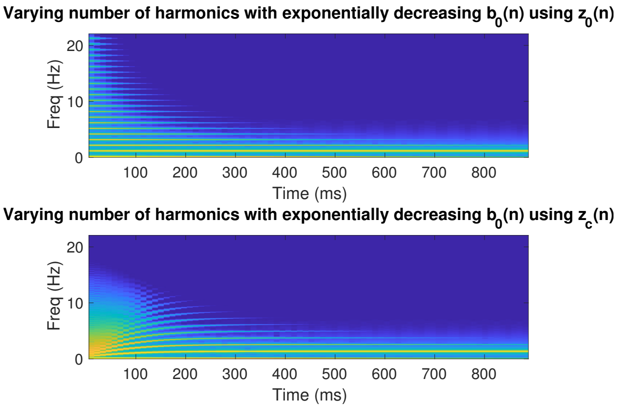


Figure 3.14: Varying the number of harmonics with exponentially decreasing $b_0(n)$ for $z_0(n)$ and $z_\mathrm{c}(n)$. A variation of the number of harmonics is shown in the upper plot for $z_0(n)$ and in the bottom plot for $z_\mathrm{c}(n)$. As $b_0(n)$ decreases, the number of frequency components also decreases in the spectrogram. A pitch glide can be seen in the lower plot as B changes both the number of harmonics and pitch when using $z_\mathrm{c}(n)$.

3.11 Loopback FM Application

This chapter includes material coauthored with Professor Tamara Smyth as it appears in

Jennifer Hsu and Tamara Smyth. Percussion synthesis using loopback frequency modulation oscillators. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

The dissertation author was the primary researcher and author of this paper. Now that we can specify pitch glides and vary the number of harmonics with loopback FM oscillators, we can use these oscillators to synthesize more complex sounds. In Chapter 4, we detail how these oscillators can be used to synthesize percussive sounds through a simple modification of traditional MS.

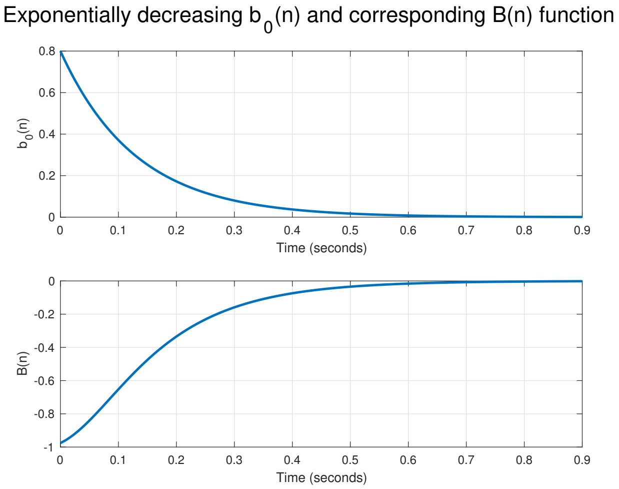


Figure 3.15: Exponentially decreasing $b_0(n)$ and corresponding $B(n)$. $b_0(n)$ and $B(n)$ as plotted in this figure were used to synthesize the signals with spectrograms plotted in Figure 3.14. As $b_0(n)$ decreases exponentially, $B(n)$ increases according to (3.40).

Chapter 4

Percussion Synthesis using Loopback FM Oscillators

4.1 Introduction

This chapter describes a real-time percussion synthesis method that is capable of creating sounds inspired by nonlinear percussion instruments. In this technique, traditional MS is modified to use loopback FM oscillators so that pitch glides and higher frequency components are created in the output sound. The basic configuration of this percussion method is shown in Figure 4.1. When sonic enhancements are included, the diagram is modified to Figure 4.2. We first formalize traditional MS and how it can be used to synthesize percussive sounds in Section 4.2. Section 4.3 explains how traditional MS can be modified with loopback FM oscillators to create a wider variety of percussion sounds. Sonic enhancements that add more nonlinear effects and a sense of space are presented in Sections 4.3.2 and 4.3.3.

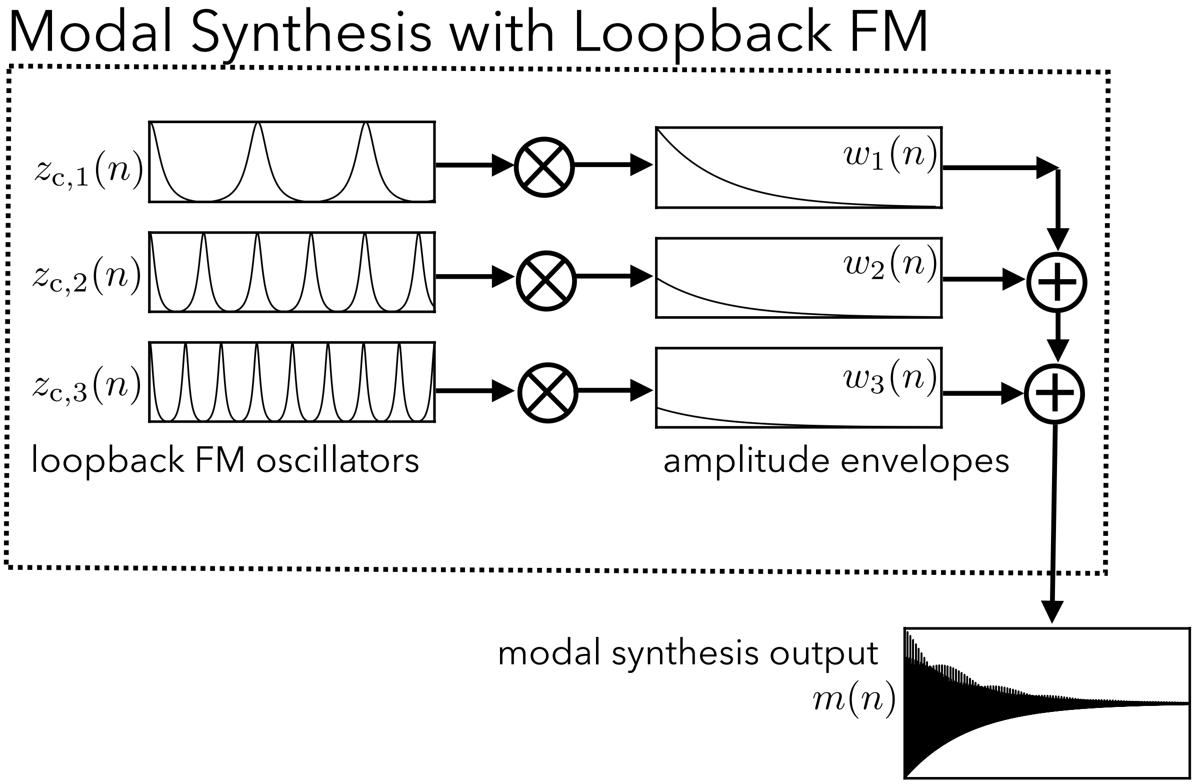


Figure 4.1: The loopback FM percussion synthesis method. Loopback FM oscillators are enveloped and summed together to form the synthesis output.

4.2 Percussion Synthesis using Traditional MS

As described in Chapter 1, MS is a technique that resynthesizes the sound of an acoustic object according to its acoustic modes or vibrational patterns. The resonant frequencies of an acoustic object arise through the sinusoidal motion of the object's modes. With traditional MS, each mode is synthesized using a 2$^{\text{nd}}$-order resonating filter with a corresponding frequency, initial amplitude, and decay [1].

To synthesize a percussive sound with MS, we begin with a list of N_{f} resonant, modal frequencies $f_{\text{r},i}$, the values of which can be obtained from acoustic experiments, spectral analysis of recorded or physically modeled sounds (e.g. DWMs, FDMs, etc), or calculated using theoretical equations. The i in $f_{\text{r},i}$ indicates the index of the modal frequency.

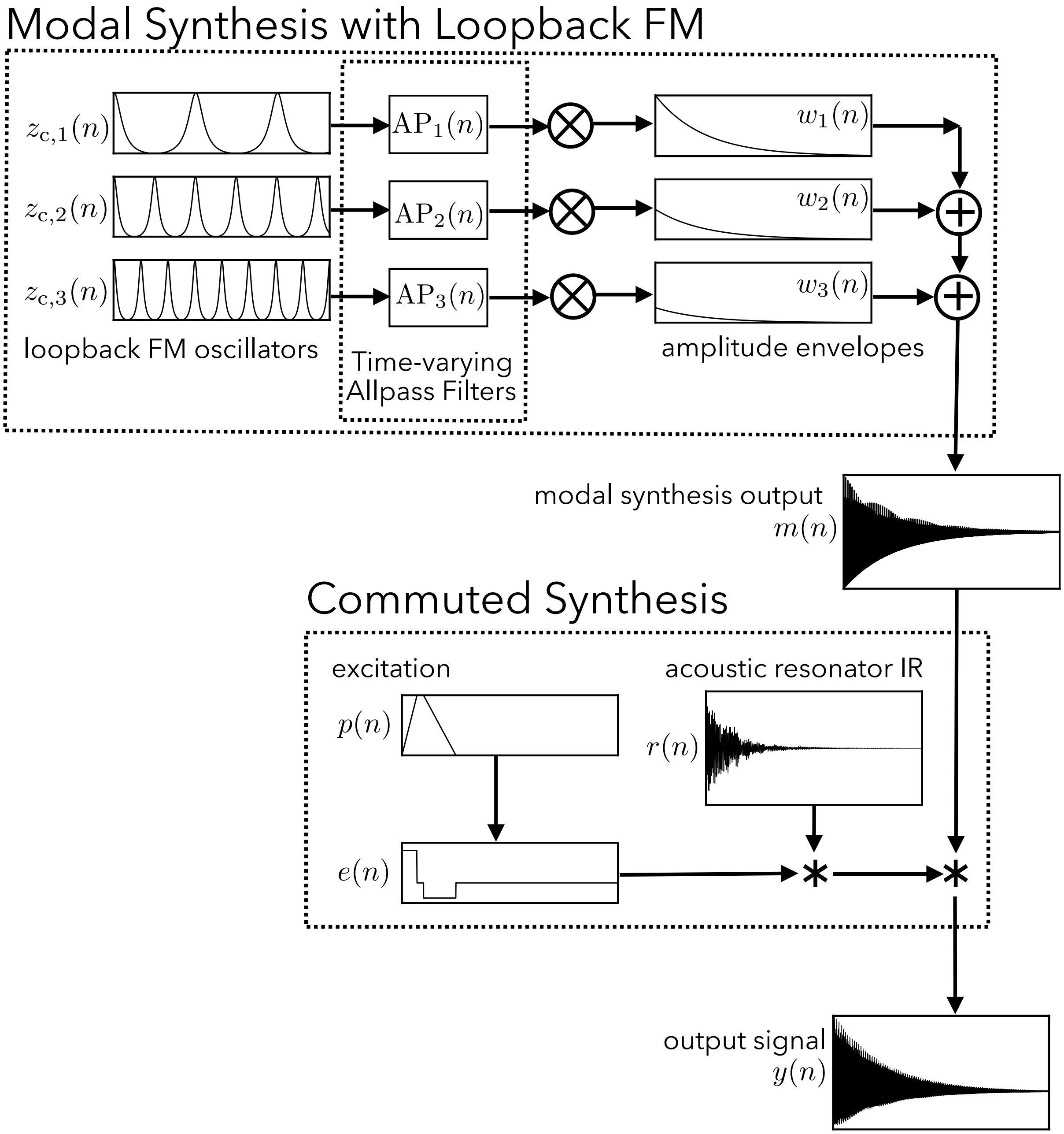


Figure 4.2: The loopback FM percussion synthesis method with sonic enhancements. The "Time-varying Allpass Filters" and "Commuted Synthesis" blocks have been added to the basic loopback FM percussion synthesis diagram from Figure 4.1.

Though MS traditionally models each frequency mode with a 2$^{\text{nd}}$-order bandpass resonant filter with center frequency $f_{\mathrm{r},i}$, in this work the filters are replaced by sinusoidal oscillators of frequency (or center frequency if frequency is time-varying) $f_{\mathrm{r},i}$. This allows for a straightforward comparison with the loopback FM version (also implemented here with oscillators) than if traditional MS bandpass filters had been used. A sinusoidal component with carrier frequency $\omega_{\mathrm{c},i} = 2\pi f_{\mathrm{r},i}$ is expressed as $s_i(n) = \sin(\omega_{\mathrm{c},i} nT)$ for time sample n and period $T = 1/f_{\mathrm{s}}$ for sampling rate f_{s}. Each sinusoidal component is multiplied with an amplitude envelope $w_i(n)$. For percussive sounds, $w_i(n)$ are typically exponentially decreasing envelopes with possibly different initial amplitudes and decay constants for different modes. For example, for natural sounding results, higher-frequency modes should be made to decay more rapidly. Enveloped sinusoidal components are added together to form the MS output given by

$$m_s(n) = \sum_{i=0}^{N_{\mathrm{f}}-1} w_i(n) s_i(n), \tag{4.1}$$

the spectrum of which is shown in Figure 4.3 for $N_{\mathrm{f}} = 3$.

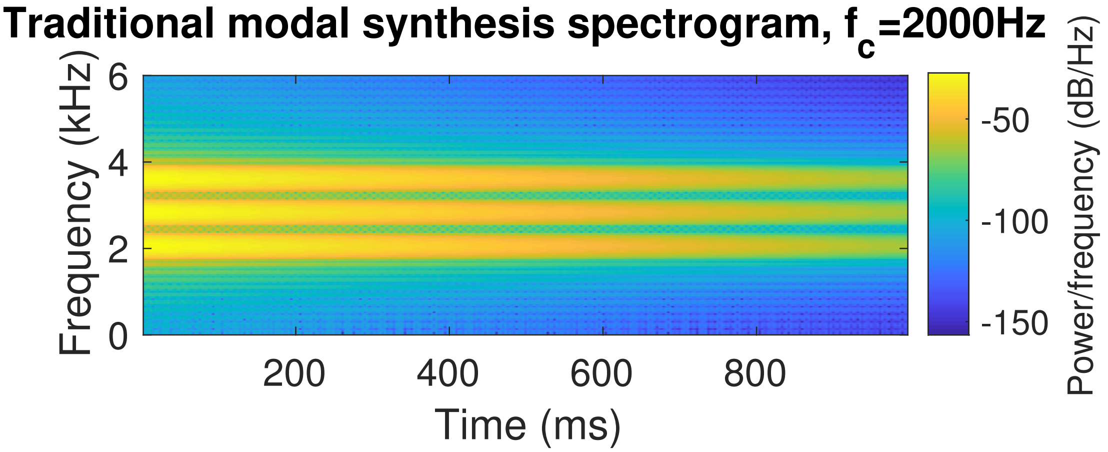


Figure 4.3: Traditional MS for three modal frequencies. The spectral components are sinusoids and the synthesis output is the sum of those spectral components.

MS is efficient and useful for recreating the sound of objects that consist of a small number of resonant frequency modes. However, MS is a linear method and (4.1) is incapable of capturing nonlinear effects. A simple modification to the sinusoidal components of the MS framework allows the method to create complex and dynamic sounds reminiscent of nonlinear vibrations in percussion instruments. This modification uses loopback FM, a synthesis technique described in Chapter 3, in which each sinusoidal component is looped back to modulate its own carrier frequency.

4.3 Percussion Synthesis with Loopback FM Oscillators

The loopback FM percussion synthesis method consists of MS with loopback FM oscillators and two optional sonic enhancements to add further nonlinearities and space. Figure 4.1 shows the basic loopback FM percussion synthesis method and Figure 4.2 shows the method with the enhancements. The "Modal Synthesis with Loopback FM" block in Figures 4.1 and 4.2 consists of synthesizing the vibrations of an abstract, nonlinear surface using MS and either of the two time-varying loopback FM equations ((3.19) or (3.22)) to produce output $m(n)$. The loopback FM oscillators can be processed with time-varying allpass filters to add more nonlinearities as shown in the "Time-varying Allpass Filters" block in Figure 4.2. The "Commuted Synthesis" block in Figure 4.2 completes the percussion model by convolving a parametric excitation function and acoustic resonator impulse response with $m(n)$, adding a sense of space to the resulting synthesis.

4.3.1 MS with Loopback FM

Like the percussion MS technique described in Section 4.2, the "Modal Synthesis with Loopback FM" block begins with a list of modal frequencies $f_{\mathrm{r},i}$ of length N_{f}. Instead of sinusoidal oscillators, N_{f} loopback FM oscillators are generated using the frequencies in $f_{\mathrm{r},i}$. As described in [55], the loopback FM oscillators can be expressed as resonating filters, though here, they are implemented as oscillators. This is similar to implementing MS with sinusoidal oscillators as opposed to resonating filters as described in Section 4.2. The loopback FM oscillator $z_{\mathrm{c},i}(n)$, given in (3.14) and (3.19), has been synthesized with carrier frequency $\omega_{\mathrm{c},i} = 2\pi f_{\mathrm{r},i}$, where subscript i means that $\omega_{\mathrm{c},i}$ is set using the i^{th} frequency in $f_{\mathrm{r},i}$. The other parameter used with $z_{\mathrm{c},i}(n)$ can also be indexed similarly as B_i.

The real part of each loopback FM oscillator is multiplied with an amplitude envelope $w_i(n)$ and the enveloped loopback FM oscillators are summed to create the MS output

$$m(n) = \sum_{i=0}^{N_{\mathrm{f}}-1} w_i(n)\Re\{z_{\mathrm{c},i}(n)\}. \tag{4.2}$$

Alternatively, the closed-form loopback FM equation $z_{0,i}(n)$, given in (3.17) and (3.22), can be used where subscript i means that $\omega_{0,i}$ is set using the i^{th} frequency in f_i, and $z_{\mathrm{c},i}(n)$ in (4.2) is replaced with $z_{0,i}(n)$ to form

$$m(n) = \sum_{i=0}^{N_{\mathrm{f}}-1} w_i(n)\Re\{z_{0,i}(n)\}. \tag{4.3}$$

Similarly, the number of harmonics parameter used with $z_{0,i}(n)$ can be notated as $b_{0,i}$.

4.3.2 Sonic Enhancement 1: Adding Further Nonlinearities with Time-Varying 2$^{\mathrm{nd}}$-Order Allpass Filter Processing

To create noisy and cymbal-like sounds, the loopback FM output can be filtered with time-varying allpass filters before enveloping and summing. Coefficient-modulated allpass filters have been used to distort the sound of electronic guitars [47] and to create rich spectra in [36, 40, 39, 43]. Because varying the coefficients of a filter can result in an unstable system, techniques to create stable, power-preserving, time-varying allpass filters have been presented in [8] and [56]. The filter presented in [56] is stable, can be used to create phase distortion, and can be parametrically controlled, making it especially appealing for our synthesis application here.

The work in [36] influenced the research described in this dissertation. In [36], Smith and Michon introduce a formulation of a coefficient-modulated allpass filter of arbitrary order that can be used to create complex, spectral components. In an application example, a 2D DWM is made nonlinear by attaching the time-varying allpass filters to the boundaries of the mesh. With this architecture, the mesh is able to produce sounds similar to nonlinear percussive instruments. Taking inspiration from [36], we apply the parametrically controllable 2$^{\text{nd}}$-order time-varying filters from [56] to our percussion synthesis model.

Time-Varying 2$^{\text{nd}}$-Order Allpass Filter Formulation

The static 2$^{\text{nd}}$-order allpass filter is formed using two parameters: f_π, which indicates the frequency at which the filter phase response is $-\pi$, and f_b, which controls the bandwidth of the phase transition region. f_π and f_b are used to set two intermediate parameters:

$$c = \frac{\tan(\pi f_b/f_\mathrm{s}) - 1}{\tan(\pi f_b/f_\mathrm{s}) + 1} \quad \text{and} \quad d = -\cos\left(\frac{2\pi f_\pi}{f_\mathrm{s}}\right). \tag{4.4}$$

For an input signal $x(n)$, the static 2$^{\text{nd}}$-order allpass filter difference equation is

$$\begin{aligned} \mathrm{AP}(n) = -cx(n) + d(1-c)x(n-1) + x(n-2) - \\ d(1-c)\mathrm{AP}(n-1) + c\mathrm{AP}(n-2). \end{aligned} \tag{4.5}$$

For the time-varying case, d becomes time-varying, and the 2$^{\text{nd}}$-order allpass filter difference equation is

$$\mathrm{AP}(n) = -cx(n) + d(n)(1-c)x(n-1) + x(n-2) - \tag{4.6}$$

$$d(n)(1-c)\mathrm{AP}(n-1) + c\mathrm{AP}(n-2), \tag{4.7}$$

where

$$d(n) = -\cos\left(\frac{2\pi \tilde{f}_\pi(n)}{f_\mathrm{s}}\right) \tag{4.8}$$

and $\tilde{f}_\pi(n)$ is a function that sinusoidally modulates f_π as

$$\tilde{f}_\pi(n) = f_\pi + M\cos\left(\frac{2\pi f_m n}{f_\mathrm{s}}\right). \tag{4.9}$$

In our context of percussion synthesis, the input to the time-varying 2nd-order allpass filter can be either a sinusoid or the output of a loopback FM oscillator. When the input is a pure sinusoid, we would have traditional MS with the addition of time-varying 2nd-order allpass filters. For ease of understanding, when the input to the 2nd-order allpass filter is a pure sinusoid, the output is similar to FM, and so the parameters are similar. f_π acts as the carrier frequency, and f_m is the modulation frequency. In the output signal, frequency components appear at $f_\pi \pm k f_m$ where $k = 0, 1, 2, \cdots$. M is the index of modulation that controls the bandwidth of the sidebands. f_b is an additional parameter that affects the levels of the sidebands in the magnitude spectrum.

Time-Varying 2nd-Order Allpass Filters in Percussion Synthesis

The time-varying 2nd-order allpass filters can be incorporated in the percussion synthesis model by filtering the real part of the loopback FM oscillators before they are windowed and summed. Each loopback FM oscillator is filtered by a time-varying allpass filter according to

$$\mathrm{AP}_i(n) = -c_i\Re\{z(n)\} + d_i(n)(1-c_i)\Re\{z(n-1)\} + \Re\{z(n-2)\} - \tag{4.10}$$

$$d_i(n)(1-c_i)\mathrm{AP}_i(n-1) + c_i\mathrm{AP}_i(n-2) \tag{4.11}$$

where the i in $\mathrm{AP}_i(n)$, c_i, and $d_i(n)$ indicates the time-varying allpass filter and its parameters associated with the i^{th} mode. Note that the parameters for the time-varying allpass filters can be different for each oscillator. $z(n)$ indicates any of the loopback FM forms given by (3.14), (3.17), (3.19), or (3.22). Next, (4.2) and (4.3) can be modified to include the allpass filtered loopback

FM oscillators

$$m(n) = \sum_{i=0}^{N_f - 1} w_i(n)\mathrm{AP}_i(n). \tag{4.12}$$

4.3.3 Sonic Enhancement 2: Sense of Space with Commuted Synthesis

In [34], Smith efficiently models stringed musical instruments using commuted synthesis. This technique is adapted here for percussion synthesis.

To complete the percussion instrument model, $m(n)$ must be excited by an excitation function, $e(n)$, and coupled to an acoustic resonator with impulse response $r(n)$. The equation to synthesize this relationship is

$$y(n) = e(n) * m(n) * r(n) \tag{4.13}$$

where $*$ indicates convolution. Because there is no dependence between $m(n)$, $e(n)$, and $r(n)$, $m(n)$ can be commuted with $r(n)$. The excitation and resonator impulse response can be convolved to form an aggregate excitation $a(n) = e(n) * r(n)$.

Aggregate excitations can be stored for several excitation and resonator combinations. During run-time, a low-latency convolution method, such as the one described in [29], can be used to convolve $a(n)$ with $m(n)$ to form the final percussion model output

$$y(n) = a(n) * m(n). \tag{4.14}$$

In our syntheses, we use a variety of resonator impulse responses as presented in Section 5.3 along with two different types of parametric excitations. Commuted synthesis can also be used to process audio recordings to enhance the sound of the original recording with the loopback FM percussion synthesis method. For this effect, the resonator impulse response is replaced by an audio recording, such as that of a drum hit. Examples of using commuted synthesis to enhance audio recordings are also shown in Section 5.3.

Excitations

The “Excitation” block in Figure 4.2 involves $p(n)$, a function that describes the vertical position of a drumstick/mallet hitting a surface at time n. The excitation signal $e(n) = p(n) - p(n-1)$, relates to the velocity of the drumstick/mallet and is convolved with the acoustic resonator impulse response to form $a(n)$. Here, we use raised cosine envelopes and filtered noise bursts for $p(n)$. These signals are parametric and affect the resulting output timbre. Examples of each excitation type are shown in Figure 4.4.

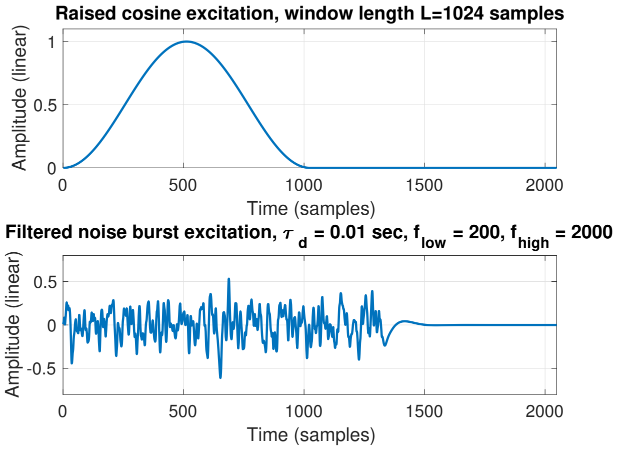


Figure 4.4: Excitation examples. Top: raised cosine envelope with a window length of 1024 samples. Bottom: noise burst with a length of 0.01 seconds and bandpass filtered from 200 Hz to 2000 Hz.

Raised Cosine Envelopes: The raised cosine envelope has a single parameter: the window length L. The equation for the excitation is

$$p(n) = \begin{cases} 0.5\left(1-\cos\left(\dfrac{2\pi n}{L-1}\right)\right), & \text{for } 0 \le n < L \\ 0, & \text{for } n \ge L \end{cases} \tag{4.15}$$

Filtered Noise Bursts: The parameters for a filtered noise burst are noise burst duration t_d and low and high frequency cutoffs for a bandpass filter f_{low} and f_{high}. Examples in this paper use white noise filtered by a 2^{nd}-order Butterworth bandpass filter.

Acoustic Resonator Impulse Response

The resonator impulse response $r(n)$ is not limited to drum shells or bodies. $r(n)$ can be the impulse response of any acoustic resonator, recorded or synthesized. Additionally, the resonator impulse response can be replaced by an audio recording so that the loopback FM percussion synthesis method effectively processes the audio recording, adding variations to the timbre. Examples of possible resonator impulse responses and audio recordings are presented in Section 5.3.

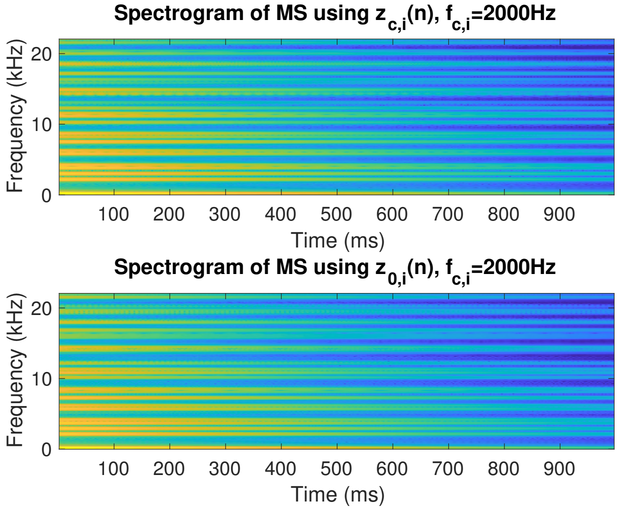
4.4 Musical Parameters for MS with Loopback FM

Musical parameters for the MS portion of the loopback FM percussion synthesis method are presented here along with their corresponding variables and equations.

4.4.1 Static Number of Harmonics: Oscillators Created with $z_{\mathbf{c},i}(n)$ or $z_{0,i}(n)$

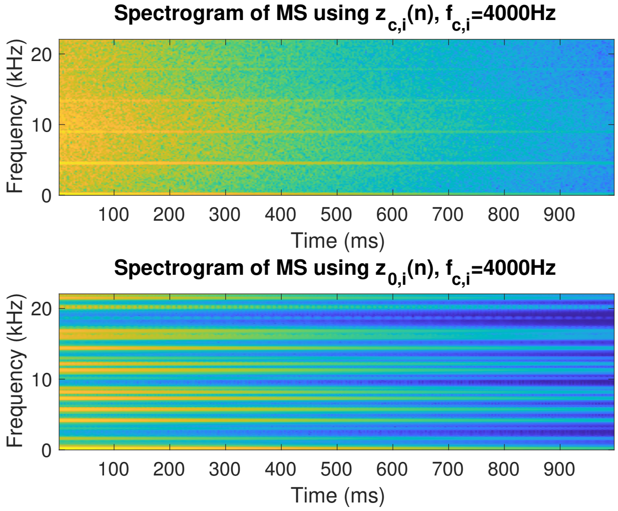
The MS oscillators can be synthesized using $z_{\mathrm{c},i}(n)$ or $z_{0,i}(n)$ as shown in (4.2) and (4.3). While both forms produce almost identical results from $f_{\mathrm{c},i} = 0$ Hz to around $f_{\mathrm{c,i}} = 2500$ Hz at a sampling rate of 44.1 kHz, when $f_{\mathrm{c},i} > 2500$ Hz, the version that uses $z_{\mathrm{c},i}(n)$ becomes much noisier, due to aliasing. As explained in Section 3.7, if the sampling rate is increased, the output is the same whether the oscillators are created using $z_{\mathrm{c},i}(n)$ or $z_{0,i}(n)$.

Figure 4.5 shows that the $z_{\mathrm{c},i}(n)$ and $z_{0,i}(n)$ MS oscillators produce similar spectrograms when the lowest of 3 modal frequencies is set to a low carrier frequency of $f_{\mathrm{c},0} = 2000$ Hz. Vastly different spectrograms are produced when the lowest of the 3 modal frequencies is set to a higher carrier frequency of $f_{\mathrm{c},0} = 4000$ Hz as shown in Figure 4.6. The MS using $z_{\mathrm{c},i}(n)$ synthesizes a noisier output and can be used to create cymbal- and crash-like sounds as shown in Section 5.3.5.


Spectrogram of MS using $z_{c,i}(n)$, $f_{c,i}$=2000Hz

Spectrogram of MS using $z_{0,i}(n)$, $f_{c,i}$=2000Hz

Figure 4.5: Loopback FM MS with $z_{\mathrm{c},i}$ and $z_{0,i}$ oscillators with low carrier frequencies create almost identical results.

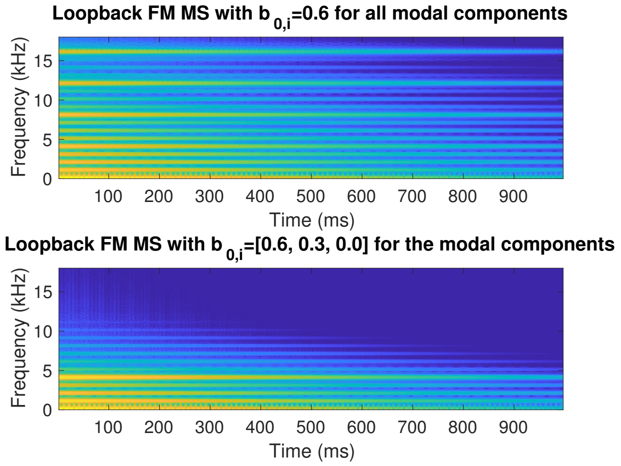

Spectrogram of MS using $z_{c,i}(n)$, $f_{c,i}$=4000Hz

Spectrogram of MS using $z_{0,i}(n)$, $f_{c,i}$=4000Hz

Figure 4.6: Loopback FM creates noisier output than stretched allpass filter MS for high carrier frequencies.

4.4.2 Static Number of Harmonics: B_i and $b_{0,i}$

As explained in Section 3.7, B_i controls the number of harmonics with $z_{c,i}(n)$ oscillators given by (3.14) while $b_{0,i}$ affects the number of harmonics with $z_{0,i}(n)$ oscillators given by (3.17). When B_i and $b_{0,i}$ are close to 1 or -1, more sidebands appear in the spectrum. With loopback FM MS, the B_i or $b_{0,i}$ values can be different for each modal component. Figure 4.7 shows two spectrograms of a loopback MS signal created using three $z_{0,i}(n)$ oscillators with sounding frequencies of 100, 1000, and 4000 Hz. In the upper graph, the $b_{0,i}$ values are all set to 0.6. The amount of sidebands about each of the three main modal frequency components is about the same. IN the lower graph, the $b_{0,i}$ values are set to 0.6, 0.3, and 0. The lowest frequency modal component at 100 Hz has the most sidebands, followed by the middle frequency component at 1000 Hz. The highest modal component does not have any sidebands at all because $b_{0,i}$ is set to 0. Different values B_i and $b_{0,i}$ values can be given for individual modal components, and this results in a change over the overall timbre of the loopback MS signal.



Loopback FM MS with $b_{0,i}$=0.6 for all modal components

Loopback FM MS with $b_{0,i}$=[0.6, 0.3, 0.0] for the modal components

Figure 4.7: The loopback FM magnitude spectrum. Frequency components occur at integer multiples of the sounding frequency, 300 Hz, and the amplitude of the components decreases logarithmically.

4.4.3 Time-varying Number of Harmonics: $B_i(n)$ and $b_{0,i}(n)$

With $z_{c,i}(n)$ oscillators given by (3.19), $B_i(n)$ affect the time-varying number of harmonics and sounding frequency. When using the closed-form loopback FM oscillator $z_{0,i}(n)$, given by (3.22), $b_{0,i}(n)$ controls the time-varying number of harmonics, independent of pitch. Methods for controlling the time-varying number of harmonics by means of $b_{0,i}(n)$ are discussed in Section 3.10.3.

In Figure 4.8, $B_i(n) = g^n$ where $g_i = 0.9999$, $b_{0,i}(n)$ is obtained according to (3.21), and amplitude envelopes are the same for all modal frequencies. The top and middle plots in Figure 4.8 compare spectrograms between the static and time-varying versions for the closed-form loopback FM oscillators, $z_{0,i}(n)$. The top plot shows a static number of harmonics with $b_{0,i} = -0.6312$ with oscillators formed using (3.17), and the middle plot shows the number of harmonics varying over time where $b_{0,i}(n)$ is used with oscillators formed from (3.22). The sidebands in the top plot are the same over the course of the signal, but the higher frequency sidebands die out over time in the middle plot as $b_{0,i}(n)$ increases from -1 to 0. The middle and bottom plots can be used to compare time-varying number of harmonics between $z_{c,i}$ oscillators using (3.19) and $z_{0,i}$ oscillators using (3.22). In the bottom plot, it is clear that $B_i(n)$ creates variations in both the number of harmonics and pitch as n increases. In the middle plot, $b_{0,i}(n)$ changes the number of harmonics without affecting the frequency trajectories.

Figure 4.9 shows how varying the number of harmonics can be used to delay the appearance of higher frequency components, an effect known to occur with cymbals. Both graphs use $z_{0,i}$ oscillators with loopback FM MS and linearly decreasing $b_{0,i}(n)$ functions to modulate the number of harmonics. A cascade of frequency components can be seen in the first 50ms of the bottom graph. To create this effect, the higher modal components have delayed amplitude envelopes $w_i(n)$ and $b_{0,i}(n)$ functions where the delay increases linearly as modal frequency increases. Although much work must be done in order to make this example sound like a cymbal, this is a first step in creating those types of timbres.

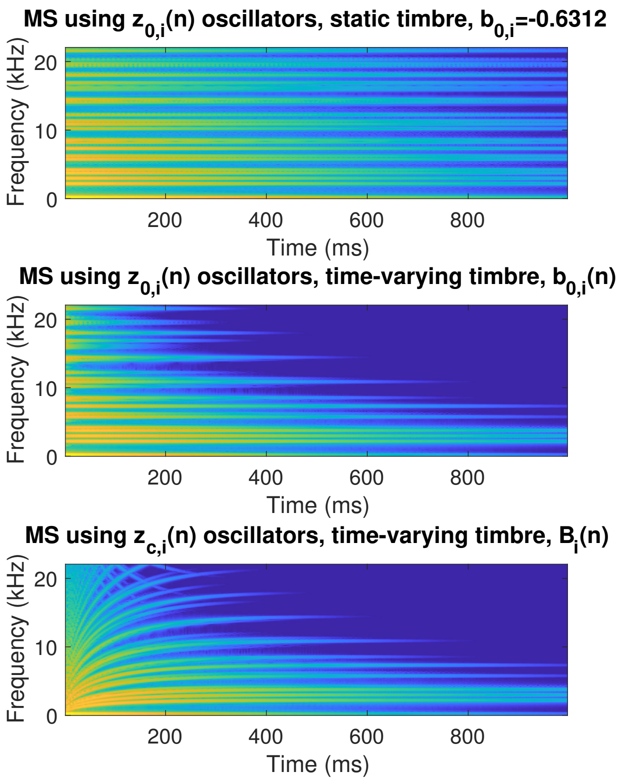
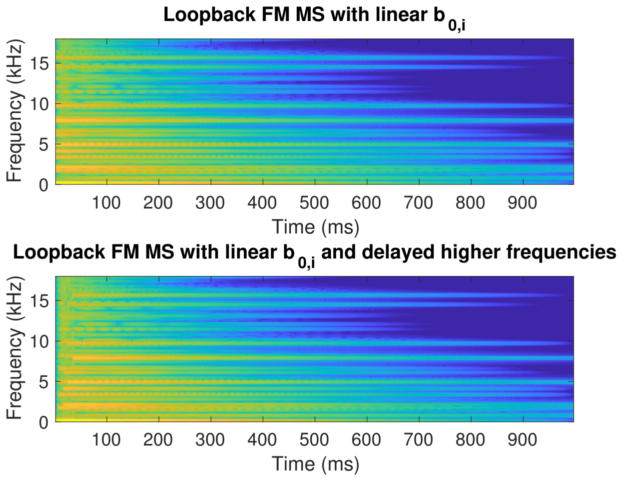


Figure 4.8: Static and time-varying number of harmonics for closed-form MS and time-varying number of harmonics for loopback FM MS. Top: sideband levels are constant for static $b_{0,i}$. Middle: $b_{0,i}(n)$ modulates the number of harmonics independently of pitch. Bottom: $B_i(n)$ affects both the number of harmonics and pitch.

4.4.4 Static Sounding Frequency: $\omega_{0,i}$

For the static pitch loopback FM forms $z_{\mathrm{c},i}(n)$ in (3.14) and $z_{0,i}(n)$ in (3.17), the sounding frequency can be controlled with $\omega_{0,i} = 2\pi f_{0,i}$. For a desired $\omega_{0,i}$ with $z_{\mathrm{c},i}$, one would use (3.15) and either 1) set B_i to a desired value and solve for $\omega_{\mathrm{c},i}$ or 2) set $\omega_{\mathrm{c},i}$ and solve for B_i.

Because the modal frequencies for percussive instruments are often inharmonic, the sounding frequency for percussion synthesis is not clearly defined. With MS using $z_{0,i}(n)$ oscillators, $\omega_{0,i} = 2\pi f_{\mathrm{r},i}$ is used to set the sounding frequencies of individual oscillators. For MS using $z_{\mathrm{c},i}(n)$ oscillators, the carrier frequencies can be set to the modal frequencies: $\omega_{\mathrm{c},i} = 2\pi f_{\mathrm{r},i}$ or the sounding frequencies can be set to the modal frequencies: $\omega_{0,i} = 2\pi f_{\mathrm{r},i}$. According to (3.15), when $B_i = 0$, $\omega_{0,i} = \omega_{\mathrm{c},i}$, and setting either $\omega_{\mathrm{c},i}$ or $\omega_{0,i}$ to the modal frequencies would


Loopback FM MS with linear $b_{0,i}$

Loopback FM MS with linear $b_{0,i}$ and delayed higher frequencies

Figure 4.9: Number of harmonics varying linearly with and without delayed appearance of higher frequencies. In both graphs, the number of harmonics is varying linearly for each of five modal components. In the bottom graph, an additional delay is added to higher frequency modal components to imitate effects that are frequently observed with cymbals.

create the same output. When B_i is large and close to 1 or -1, $\omega_{0,i}$ will be a lower frequency than $\omega_{c,i}$. This means that using $\omega_{c,i} = 2\pi f_{r,i}$ produces lower sounding frequencies while setting $\omega_{0,i} = 2\pi f_{r,i}$ produces higher sounding frequencies, which will most likely produce aliasing effects as described in Sections 3.7 and 4.4.1. Figure 4.10 demonstrates that when B is close to 1, $\omega_{c,i} = 2\pi f_{r,i}$ creates an output where the individual spectral components can be clearly seen while $\omega_{0,i} = 2\pi f_{r,i}$ creates a noisy output as higher frequencies contribute to extreme aliasing effects.

4.4.5 Pitch Glides: $B_i(n)$ and $\omega_{0,i}(n)$

With a time-varying loopback FM oscillator $z_{c,i}(n)$ given by (3.19), a pitch glide can be added by varying $B_i(n)$ over time. This also produces timbral changes. A pitch glide can be created with a closed form loopback FM oscillator, $z_{0,i}(n)$ in (3.22), by varying $\omega_{0,i}(n)$ over time as described in Section 3.9. To modify the pitch independently of the number of harmonics, $b_{0,i}(n)$ should be held constant for all values of n. Examples of possible pitch glides are elaborated in Sections 3.10.1 and 3.10.2. Note that each modal component is associated with its own pitch glide, so different pitch glide functions can be combined in a single MS signal to create interesting

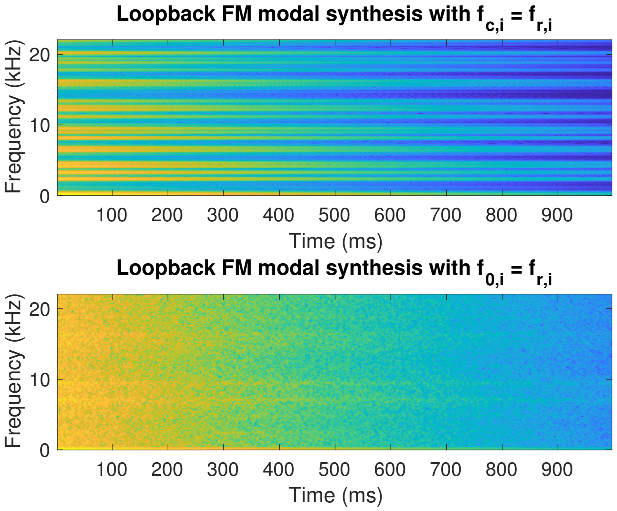


Figure 4.10: Loopback FM spectrograms for $\omega_{c,i} = 2\pi f_{\mathrm{r},i}$ (top) and $\omega_{0,i} = 2\pi f_{\mathrm{r},i}$ (bottom). The top and bottom signals are generated using the same three modal frequencies with $B = 0.9$.

timbres. The two spectrograms in Figure 4.11 are loopback FM MS signals with five modal components. In the upper graph, all pitch glides are linear as described in 3.10.2. In the bottom graph, the pitch glides alternate between linear and exponential. The combination of pitch glide types contributes to a more complex timbre.

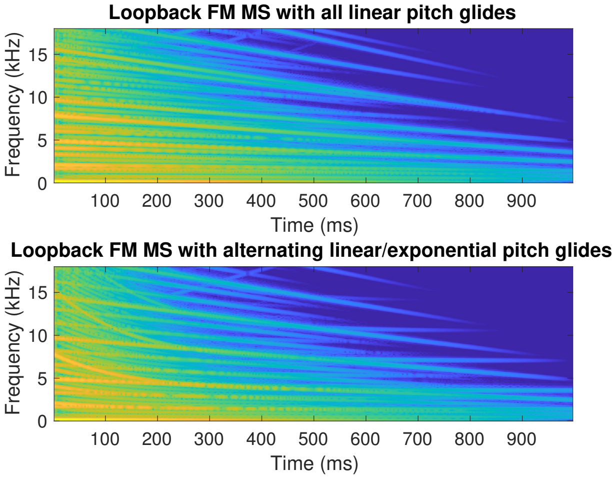


Figure 4.11: Loopback FM MS pitch glide with all linear pitch glides or alternating linear and exponential pitch glides. The upper graph uses all linear pitch glides while the lower graph alternates between linear and exponential pitch glides to create a more complex sound.

As described in Section 4.4.1, differences between using oscillators formed with $z_{\mathrm{c},i}(n)$ or

$z_{0,i}(n)$ can be observed when $\omega_{c,i}$ is high, and this effect occurs with pitch glides. Figure 4.12 shows the high carrier frequency difference for a pitch glide over three modal frequencies using MS with $z_{c,i}(n)$ and $z_{0,i}(n)$. The pitch glide is created with $B_i(n) = 0.9999^n$, so the number of harmonics also changes. At higher carrier frequencies, MS with $z_{c,i}(n)$ creates noise-like output for the first 100ms and more spectral components than MS with $z_{0,i}(n)$ from $100-250$ms. From 300ms through the remainder of the signal, the frequency components are more similar.

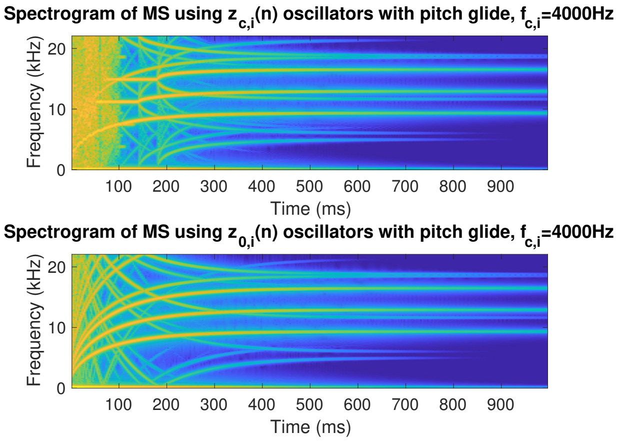


Figure 4.12: MS using $z_{c,i}(n)$ vs. $z_{0,i}(n)$ oscillators at high carrier frequencies with a pitch glide produce different spectrograms for the first 250ms. The lowest oscillator has a carrier frequency of 4000 Hz.

4.4.6 Decay Time: $w_i(n)$

The decay time for the percussion signal can be controlled through the amplitude envelopes $w_i(n)$. A natural sounding way to set these envelopes is to model them as exponentially decreasing envelopes over time:

$$w_i(n) = A_{0,i} e^{-nT/\tau_{w,i}}, \tag{4.16}$$

with different initial amplitude values $A_{0,i}$, as shown in Figure 5.10, and/or different decay constants $\tau_{w,i}$.

We can specify the decay time in seconds for each modal component using a measurement, frequently seen in reverberation studies, called the T_{60}. This is defined as the time, in seconds, for the signal to decay by 60dB. For each modal component, we choose an initial amplitude value $A_{0,i}$ so that $w_i(0) = A_{0,i}$ and a $\mathrm{T}_{60,i}$ value in seconds. Let $n_{60,i} = \mathrm{T}_{60,i} f_\mathrm{s}$ be the time in samples that it takes for the signal to decay by 60dB. Then, we calculate the value that is 60dB down from the initial amplitude as

$$A_{60,i} = \frac{A_{0,i}}{10^{60/20}}. \tag{4.17}$$

Using (4.16) and $n = n_{60,i}$, we have

$$w_i(n_{60,i}) = A_{60,i} = A_{0,i} e^{-n_{60,i} T/\tau_{w,i}}, \tag{4.18}$$

and $\tau_{w,i}$ can be found using

$$\tau_{w,i} = -\frac{n_{60,i} T}{\log(A_{60,i}/A_{0,i})}. \tag{4.19}$$

A set of amplitude envelopes $w_i(n)$ can now be calculated using (4.16). An example of a set of envelopes for five modal components is shown in Figure 4.13. Each of the amplitude envelopes is given a different $\mathrm{T}_{60,i}$ value and initial amplitude value $A_{0,i}$. For natural sounding results, the lower frequency modal components should have longer $\mathrm{T}_{60,i}$ times and larger initial amplitudes than the higher frequency modal components. This means that the lower frequency modal components should pair with the blue, red, and orange amplitude envelopes.

If one would like to specify an overall decay time, $w_g(n)$ can be used as the global amplitude envelope. After loopback FM MS has created a signal $m(n)$ and after any sonic enhancements, we are left an output signal $y(n)$. Multiplying $y(n)$ with $w_g(n)$ applies a global decay to the signal so that the signal lasts as long as the user desires. $w_g(n)$ can be calculated with a T_{60} parameter and (4.16) using the same process that was covered previously for $w_i(n)$.

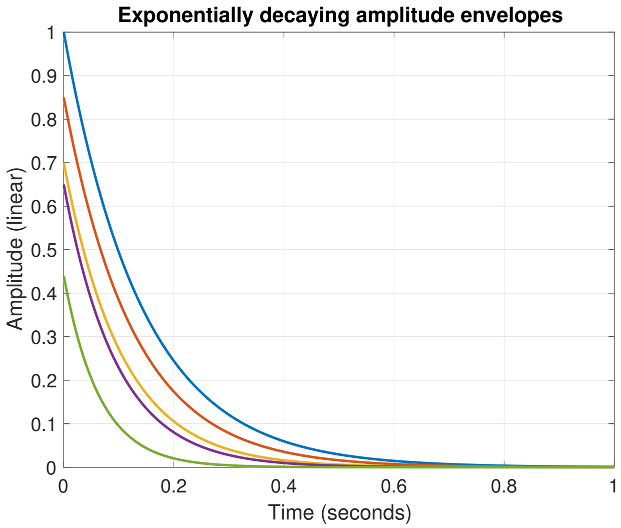


Figure 4.13: Exponentially decaying amplitude envelopes for five modal components. Each amplitude envelope has a different $T_{60,i}$ and initial amplitude value $A_{0,i}$.

4.5 Musical Parameters for the Time-Varying 2$^{\text{nd}}$-Order Allpass Filters

This section covers parameters for the 2$^{\text{nd}}$-order allpass filters and how they affect the resulting percussion synthesis output. Setting the parameters for these allpass filters can be tricky, so we begin by exploring how each of the parameters affects a single, sinusoidal oscillator input, followed by how each of the parameters affects a single, loopback FM oscillator input. We build up and explore how the parameters influence MS with sinusoidal oscillators and finally how the parameters change the sound when MS is combined with loopback FM oscillators.

4.5.1 Time-Varying 2$^{\text{nd}}$-Order Allpass Filter with a Single Sinusoidal Oscillator Input

For a sinusoidal input, the time-varying 2$^{\text{nd}}$-order allpass filter creates a magnitude spectrum as shown in Figure 4.14. In that figure, $f_\pi = 5500$ Hz $f_m = 500$ Hz, $M = 1000$, and $f_b = 100$. Peaks in the magnitude spectrum appear at $f_\pi \pm k f_m$ for $k = 0, 1, 2, \cdots$. The largest peak in the spectrum in this figure is at $f_\pi = 5500$ Hz. Increasing M increases the bandwidth of the sidebands, creating a brighter timbre. f_b affects the amplitude of the peak at f_π and the sideband amplitudes. How f_b affects the peak levels is dependent on f_π. As f_b increases from 0 to a value F, the amplitude of the peak at f_π decreases while the sidebands increase in amplitude. This creates a brighter and more nasal-sounding timbre. As f_b increases beyond F, the level of the peak at f_π increases while the amplitudes of the sidebands decrease. The timbre darkens and approaches a pure sinusoidal tone. F is dependent on f_π and the exact equation for this value is left for future research.

Like FM, the 2$^{\text{nd}}$-order allpass filter is not bandlimited, so aliasing will occur if f_π is set extremely high or low with large M and f_m values. This means that there will be additional frequency components in the spectrum other than $f_\pi \pm k f_m$ as shown in Figure 4.15. The parameters for this figure are all the same as those in Figure 4.14 except that $f_\pi = 300$ Hz.

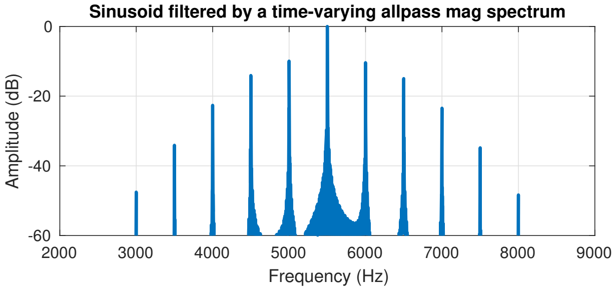


Figure 4.14: Time-varying allpass filter for a single oscillator. $f_\pi = 5500$ Hz $f_m = 500$ Hz, $M = 1000$, and $f_b = 100$. Peaks occur at $5500 \pm k500$ for $k = 0, 1, 2, \cdots$.

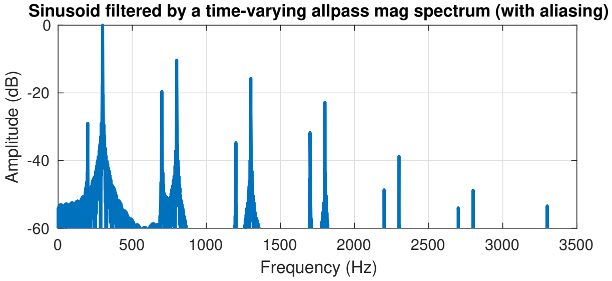


Figure 4.15: Time-varying allpass filter for a single oscillator with aliasing. $f_\pi = 300$ Hz $f_m = 500$ Hz, $M = 1000$, and $f_b = 100$. Peaks occur at $300 \pm k500$ for $k = 0, 1, 2, \cdots$. Peaks that do not follow that equation are due to aliasing/foldover across the x-axis.

4.5.2 Time-Varying 2^{nd}-Order Allpass Filter with a Single, Static Loopback FM Oscillator Input

The top of Figure 4.16 displays the magnitude spectrum of a static loopback FM oscillator, and the bottom shows that same static loopback FM oscillator output filtered by a time-varying 2^{nd}-order allpass filter. The allpass parameter f_π is set equal to f_0 so that sidebands are formed around the main peak of the static loopback FM oscillator. The rest of the allpass parameters are $f_m = 500$ Hz, $M = 1000$, and $f_b = 100$. $B = 0.9$ for the loopback FM oscillator so that a large number of sidebands are created. The peaks that are output are those expected from the static loopback FM input, indicated by the dotted red lines and occurring at frequencies $f_0 \pm kf_0$ for $k = 0, 1, 2, \cdots$, as well as sidebands that are spaced apart according to the modulation frequency f_m. The peaks that appear follow the equation $f_0 \pm kf_0 \pm qf_m$ for $q = 0, 1, 2, \cdots$. Considerable aliasing will most likely occur across the $x = 0$ and/or $x = f_s/2$ axis, depending on which axis f_0 is closest to and how large f_m is, and the spectrum will grow increasingly complex.

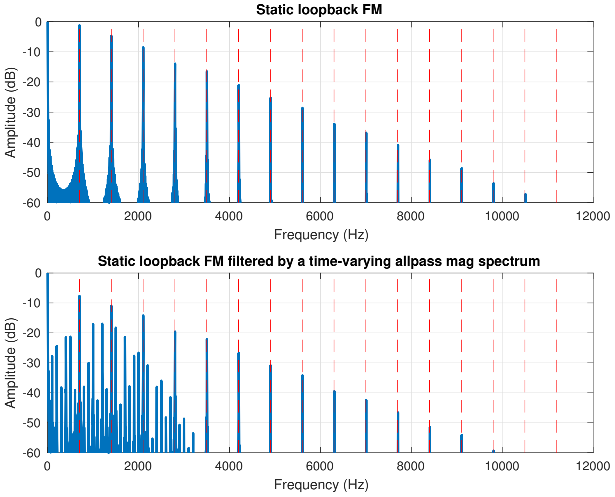


Figure 4.16: Time-varying allpass filter with a static loopback FM oscillator. Top: static loopback FM oscillator spectrogram with $f_0 = 700$, $B = 0.9$. Bottom: time-varying allpass filter with the static loopback FM oscillator from the top graph as input with $f_\pi = f_0 = 700$ Hz $f_m = 500$ Hz, $M = 1000$, and $f_b = 100$. Peaks occur at $700 \pm k700 \pm q500$ for $k = 0, 1, 2, \cdots$ and $q = 0, 1, 2, \cdots$. The dotted red lines indicate expected frequencies from the static loopback FM oscillator. All other unmarked peaks are a result of filtering the loopback FM output with the time-varying allpass filter.

4.5.3 Time-Varying 2^{nd}-Order Allpass Filter with a Single, Time-varying Pitch and Number of Harmonics Loopback FM Oscillator Input

Figure 4.17 compares a time-varying loopback FM oscillator spectrogram with a time-varying allpass filtered version of that same loopback FM oscillator. The pitch and the number of harmonics of the loopback FM oscillator are being varied over time with $B = 0.9999^n T$. The time-varying allpass filter used to filter the loopback FM oscillator has parameters $f_\pi = f_0 = 700$ Hz $f_m = 500$ Hz, $M = 1000$, and $f_b = 100$. The allpass filtered version is clearly noisier than the one that has not been filtered. The pitch glide also does not extend as high in frequency as the non-filtered version.

4.5.4 Time-Varying 2^{nd}-Order Allpass Filter with Multiple Sinusoidal Oscillator Inputs (MS)

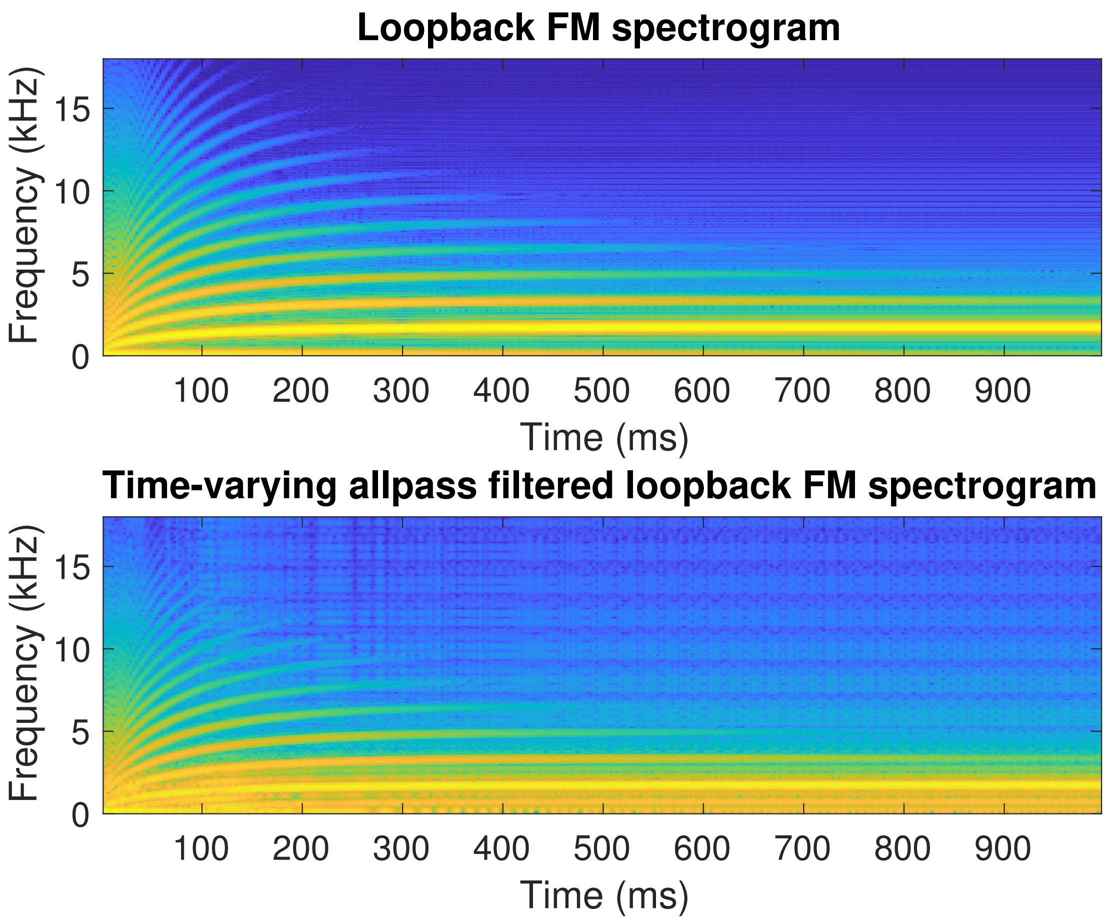
The nonlinear, time-varying allpass filters can be used with traditional MS to create richer sounds with more nonlinearities. Analyzing the parameters for this simpler case is helpful for understanding how the parameters affect more complex oscillators, like loopback FM oscillators. Each sinusoid $s_i(n)$ is filtered by a 2^{nd}-order time-varying allpass filter $\text{AP}_i(n)$ as

$$\text{AP}_i(n) = -c_i s_i(n) + d_i(n)(1-c_i)s_i(n-1) + s_i(n-2) - \tag{4.20}$$

$$d_i(n)(1-c_i)\text{AP}_i(n-1) + c_i\text{AP}_i(n-2) \tag{4.21}$$

To simplify parameter setting for the time-varying allpass filter, $f_{\pi,i}$ is set equal to each modal frequency $f_{\text{r},i}$. This allows us to create and control the sidebands around each individual modal frequency with the remaining parameters M_i, $f_{m,i}$, and $f_{b,i}$. The allpass filtered sinusoids are then windowed and summed for the final output as shown in (4.12).

Figure 4.18 shows the time-varying allpass filtered MS spectrogram for 3 modal frequen-


Loopback FM spectrogram

Time-varying allpass filtered loopback FM spectrogram

Figure 4.17: Time-varying allpass filter with a time-varying loopback FM oscillator. Top: time-varying loopback FM oscillator spectrogram with $f_0 = 700$, $B = 0.9999^n$. Bottom: time-varying allpass filter with the time-varying loopback FM oscillator from the top graph as input with $f_\pi = f_0 = 700$ Hz $f_m = 500$ Hz, $M = 1000$, and $f_b = 100$. The time-varying allpass filtered version is noisier and the pitch glide does not extend as high in frequency as the non-allpass filtered version.

cies. These modal frequencies are the same ones use to create Figure 4.3. Here, $M_i = 1000$, $f_{m,i} = 500$, and $f_{b,i} = 100$. Compared to Figure 4.3, the allpass filtered one shows many more sidebands around the modal frequencies. We can now examine how each of the allpass filter parameters, M_i, $f_{m,i}$, and $f_{b,i}$, affect the output signal.

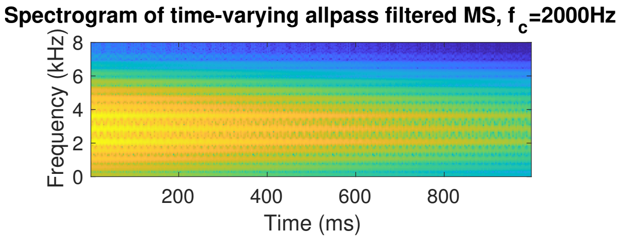


Figure 4.18: Time-varying allpass filter with traditional modal synthesis. This example uses 3 modal frequencies with the lowest frequency set to 2000 Hz. Compared to Figure 4.3, sidebands around the modal frequencies are evident in this graph as they result from the use of the nonlinear allpass filters.

Timbre using Sideband Bandwidth Parameter: M_i

Figure 4.19 shows the spectrogram of sinusoidal MS with three components, where each sinusoidal component has been filtered by its own time-varying allpass filter across different values of M_i while other parameters are held constant. M_i is set to the same value for all three modal components. As M_i increases, the bandwidth of the sidebands also increases, synthesizing noisier and more complex signals.

Timbre using Sideband Spacing Parameter: $f_{m,i}$

Figure 4.20 is a spectrogram of sinusoidal MS where each component is filtered with a time-varying allpass filter for varying values of $f_{m,i}$ while other parameters are kept constant. As $f_{m,i}$ increase in value for all three modal components, the spacing between the sidebands and modal frequency peaks increases. When $f_{m,i}$ is small, beating frequencies are visible, as shown in the top left figure, and the sound is buzzy and rough. As $f_{m,i}$ continues to increase from 100 to 3000, the sound becomes brighter and more piercing. As $f_{m,i}$ grow very large, the spacing between the modal frequencies and the sidebands becomes large enough to fill the entire auditory spectrum.

Timbre Brightness/Darkness using Secondary Sideband Bandwidth Parameter: $f_{b,i}$

Figure 4.21 shows how varying $f_{b,i}$, and holding other parameters constant for the time-varying allpass filters used with traditional sinusoidal MS affects the output. At first, when $f_{b,i}$ increases, the bandwidth of the sidebands increases and the timbre becomes brighter. For the modal frequencies used here, in between $f_{b,i} = 1000$ and $f_{b,i} = 10000$, the effect of increasing $f_{b,i}$ begins to decrease the bandwidth of the sidebands. The value at which the effect of increasing $f_{b,i}$ changes is different depending on the modal frequencies and is left as a topic for future research.

4.5.5 Time-Varying 2$^{\text{nd}}$-Order Allpass Filter with Loopback FM MS

The analysis of how the time-varying allpass filter parameters affect the synthesis output is extended here for loopback FM MS.

Loopback FM MS with Static Pitch and Number of Harmonics

The time-varying allpass filters can be applied to loopback FM MS. Figure 4.22 shows the effect of using the time-varying allpass filters on static loopback FM oscillators. The top graph is loopback FM MS without the time-varying allpass filters, and the bottom is that same loopback FM MS but with the addition of time-varying allpass filters. In this example, $f_{\pi,i} = f_{\mathrm{r},i} = f_{0,i}$, $f_{b,i} = 100$, $M_i = 2000$, and $f_{m,i} = 1000$. There is clearly more spectral information in the bottom graph around 8 kHz and below. As in many of the time-varying allpass filter examples, the allpass filtered loopback FM MS sounds noisier, more complex, and as if the sound has a fuller body.

Loopback FM MS with Pitch Glides and Time-Varying Number of Harmonics

In Figure 4.23, the time-varying allpass filters are applied to loopback FM MS when the loopback FM oscillators are varying over time. The top graph is the time-varying loopback FM MS without any filtering and the bottom is that same time-varying loopback FM MS with time-varying allpass filters applied to each oscillator. The parameters for the time-varying allpass filters are set as they are in Section 4.5.5. The effect of the time-varying allpass filters can be seen as more spectral energy below the 8 kHz mark, contributing to a noisier output.

4.6 Musical Parameters for Commuted Synthesis

4.6.1 Attack Sharpness: Raised Cosine Envelopes

With raised cosines envelopes, small values of L create sharper sounding attacks, while longer values of L increase the presence of low frequencies in the output and result in bass-heavy sounds. Intuitively, L is proportional to the mass of a hammer or mallet used to excite a drum head: a longer L means a hammer/mallet with greater mass.

4.6.2 Attack Noisiness: Filtered Noise Bursts

For filtered noise burst excitations, a longer noise burst t_{d} and higher bandpass frequency cutoff f_{high} will create a noisier attack. f_{low} and f_{high} should be tuned to filter out undesired frequencies. For example, for a high pitched percussion sound, the lower frequencies could be filtered out from the noise burst by setting f_{low} to a higher frequency.

4.6.3 Timbre: Acoustic Resonator Impulse Response $r(n)$

The acoustic resonator filters the synthesis output, so the timbre can be further shaped by the frequencies present in $r(n)$. For an expansive and large sound, a room impulse response with a long T_{60} may work well while for a shorter, tuned sound, the impulse response of a small, acoustic tube model could be used.

4.7 Conclusions

This chapter includes material coauthored with Professor Tamara Smyth as it appears in

Jennifer Hsu and Tamara Smyth. Percussion synthesis using loopback frequency modulation oscillators. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

The dissertation author was the primary researcher and author of this paper. This chapter presented a real-time method to synthesize novel, abstract percussion sounds using MS with loopback FM oscillators. Loopback FM creates complex spectra and pitch glides similar to the nonlinear effects observed in existing percussion instruments. The synthesis technique allows for parametric control of musical dimensions including sounding frequency, decay time, number of harmonics, and pitch glide. Two methods to enhance the sound with more nonlinearities by use of time-varying allpass filters and space through commuted synthesis were also examined. Next, in Chapter 5, we walk through an example of how to synthesize the sound of a kick drum using this percussion synthesis method and explore sound examples including those of a marimba, tom tom, and circular plate.

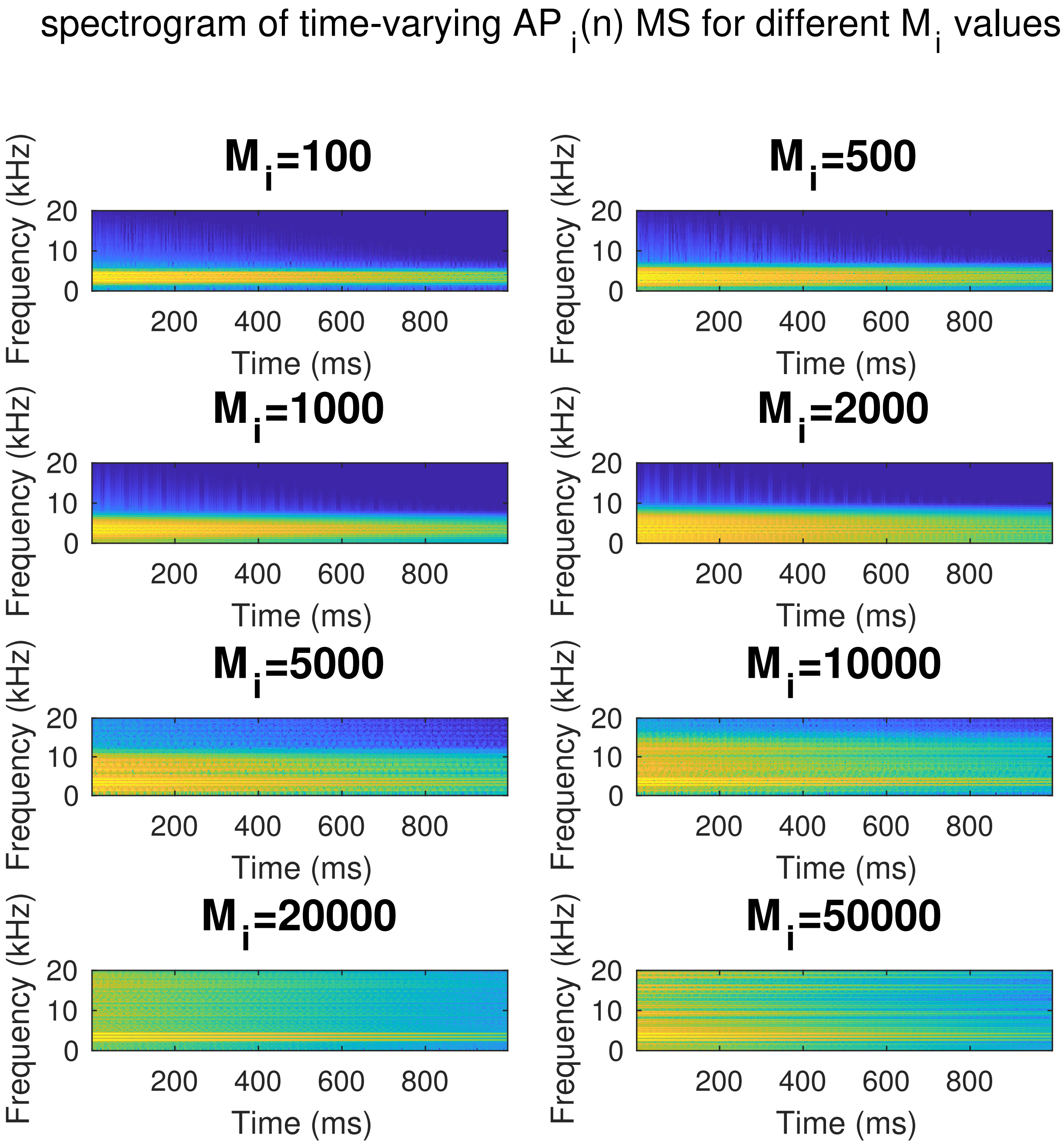


Figure 4.19: Spectrogram of various M_i values. When M_i is increased for the time-varying allpass filtered traditional MS, the bandwidth of the output signal increases, creating a brighter and more nasal-sounding output.

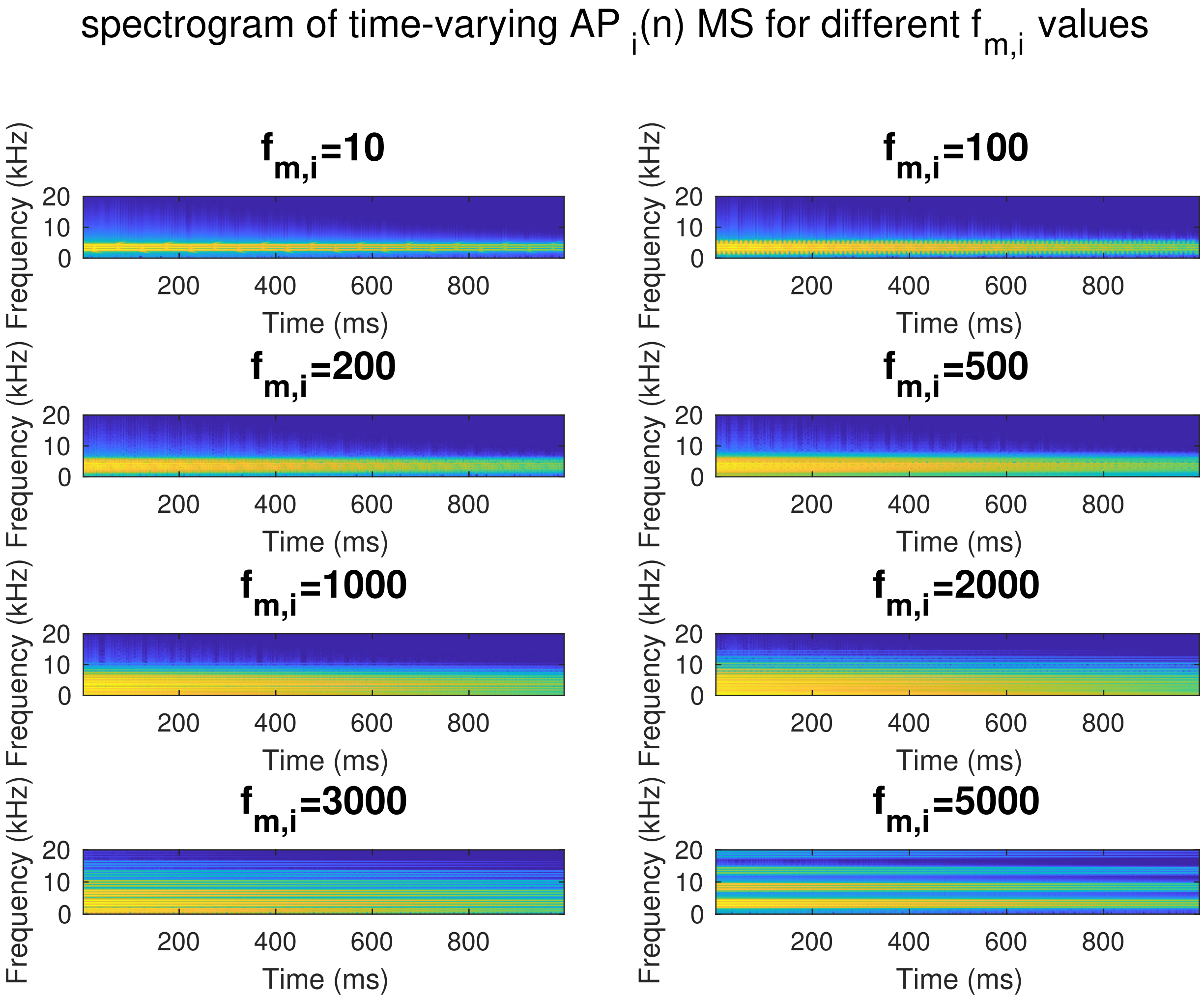


Figure 4.20: Spectrogram of various $f_{m,i}$ values. Increasing $f_{m,i}$ for the time-varying allpass filtered traditional MS increases the spacing between the sidebands and modal frequency peaks.

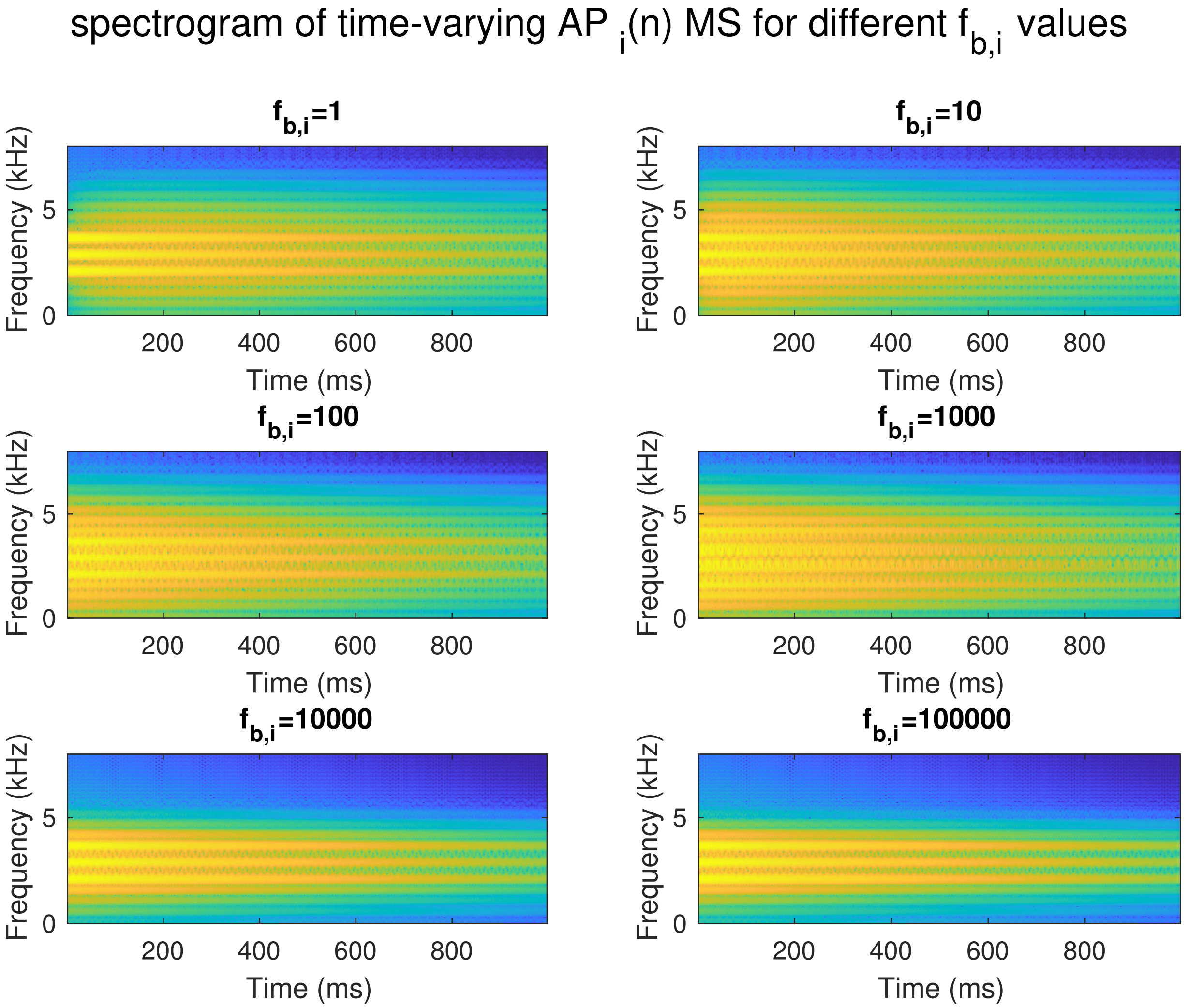
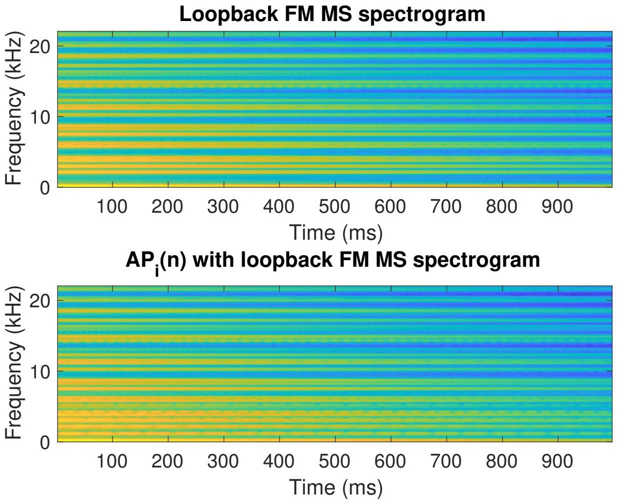
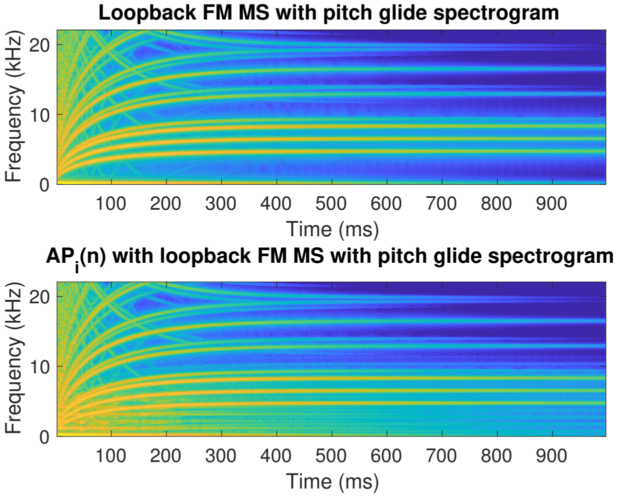


Figure 4.21: Spectrogram of various $f_{b,i}$ values. The effect of changing the value of $f_{b,i}$ for the time-varying allpass filtered traditional MS is shown through the spectrograms. Increasing $f_{b,i}$ increases the bandwidth of the signal up until somewhere between $f_{b,i} = 1000$ and $f_{b,i} = 10000$, at which point, the bandwidth begins to decrease as $f_{b,i}$ increases.


Loopback FM MS spectrogram

$AP_i(n)$ with loopback FM MS spectrogram

Figure 4.22: Time-varying allpass filter on static loopback FM MS. Top: Static loopback FM MS. Bottom: Time-varying allpass filters with static loopback FM MS. The difference between the two signals can be seen around 8 kHz and below where there is much more information in the bottom graph.


Loopback FM MS with pitch glide spectrogram

$AP_i(n)$ with loopback FM MS with pitch glide spectrogram

Figure 4.23: Time-varying allpass filter on time-varying loopback FM MS. Top: Time-varying loopback FM MS. Bottom: Time-varying allpass filter for time-varying loopback FM MS. Results are similar to the static example where the difference between the two output signals is around 8 kHz and below where there is much more information in the bottom graph.

Chapter 5

Loopback FM Percussion Synthesis Examples

5.1 Introduction

Using the percussion synthesis method described in Chapter 4, a wide variety of percussive sounds can be synthesized. In Section 5.2, we walk through an example of how to synthesize the sound of a kick drum using this percussion synthesis method. Section 5.3 covers synthesis examples of a snare drum, marimba, tom tom, and a circular plate. Audio examples are included as supplementary material.

5.2 Synthesis Walkthrough Example: the Kick Drum

This section demonstrates how to synthesize a simple kick drum sound using the loopback FM percussion synthesis method with a single $z_0(n)$ or $z_\mathrm{c}(n)$ oscillator. This section gives the reader an intuitive idea of how to set the parameters for the loopback FM MS percussion synthesis method. We cover how to create the same pitch glide with both loopback equations and ways to drastically change the timbre so that exceeds beyond our physical expectations of the sound of a kick drum.

5.2.1 A Simple Kick Drum with Static Pitch and Number of Harmonics

To synthesize a kick drum with static pitch and number of harmonics using loopback FM MS with $z_{0,i}(n)$ oscillators, we must specify values for three parameters:

- $f_{0,i} = f_{\mathrm{r},i}$: sounding frequency set to the modal/resonant frequency
- $b_{0,i}$: number of harmonics (timbre)
- $w_i(n)$: amplitude envelope

A kick drum can be synthesized using a single oscillator, so the number of modal components, $N_f = 1$, and we only need to give values for $f_{0,0}$ and $b_{0,0}$ and an amplitude envelope for $w_0(n)$. If we had more than one modal component, we would need to decide on more values and amplitude envelopes. For example, if $N_f = 3$, then we would have values for $f_{0,i}$ and $b_{0,i}$ and amplitude envelopes for $w_i(n)$ for $i = 0, 1$, and 2. Even though this example only consists of a single modal component, we use notation as if we have many modal components (i.e. $f_{0,0}$ instead of f_0) so that the concepts can be extended to synthesis using multiple modal components.

For the modal frequency, we choose $f_{0,0} = 100$ Hz, because intuitively, a kick drum should have a low frequency. The number of harmonics parameter $b_{0,0}$ can be kept small and close to 0 so that not too many sidebands appear in the spectrum. $b_{0,0} = 0.2$ works well. An exponentially decaying amplitude envelope is commonly used for percussive sounds and performs well with the kick drum example. As described in Section 4.4.6, an exponentially decaying amplitude envelope can be found with an initial amplitude $A_{0,i}$ and a $T_{60,i}$ value. Here, we let $A_{0,0} = 1$ and $\mathrm{T}_{60,0} = 0.8$ seconds, solve for $\tau_{w,0}$ and compute the exponentially decaying amplitude envelope $w_0(n)$. The loopback FM signal using a $z_{0,i}(n)$ oscillator can now be found using (3.17). To create the simple kick sound, $z_{0,0}(n)$ should be enveloped with $w_0(n)$ so that $m(n) = z_{0,0}(n)w_0(n)$.

To create a similar signal with a $z_{c,i}(n)$ oscillator, B_0 can be computed from $b_{0,0} = 0.2$ by rearranging (3.18) into

$$B_i = -\frac{2b_{0,i}}{b_{0,i}^2 + 1}. \tag{5.1}$$

$f_{c,0}$ can be found by rearranging (3.15) and using it with the newly calculated B_0 and angular frequencies, $\omega_{0,0} = 2\pi f_{0,0}$ and $\omega_{c,0} = 2\pi f_{c,0}$. $z_{c,0}(n)$ can be computed using (3.14) and multiplied with $w_0(n)$ to create the output signal.

Figure 5.1 graphs the time-domain signal and the spectrogram of the kick drum with static pitch and number of harmonics. An audio example is provided as the supplementary audio file *hsu_kick_staticPitchAndTimbre.wav*

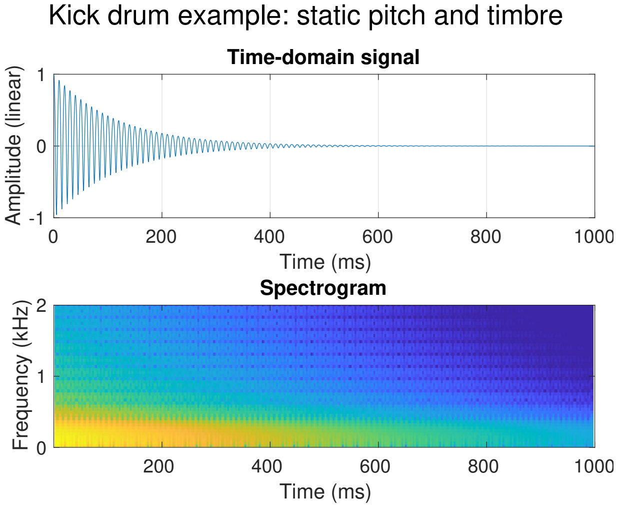


Figure 5.1: Kick drum example with static pitch and number of harmonics. The sound is synthesized using a single $z_{0,i}(n)$ oscillator with sounding frequency $f_{0,0} = 100$ Hz and number of harmonics parameter $b_{0,0} = 0.2$.

5.2.2 Adding a Pitch Glide

In this section, we create a downwards pitch glide, similar to the "pitch sigh" of the Roland TR-808 bass drum discussed in [62]. We begin by choosing our starting and ending frequencies and the type of pitch glide that we want. In the previous section, the sounding frequency is set according to $f_{0,0} = 100$ Hz. Let us create an exponentially decreasing pitch glide so that the ending frequency is 40 Hz. Using the variables and equations from Section 3.10.2, $f_{x,0} = 100$ and $f_{y,0} = 100$. The time at which we want to be close to $f_{y,0}$ is $t_{d,0} = 0.6$ seconds.

Using (3.31), we set $h_0(0) = 1$ and $h_0(n_d) = 0.001$ to find that $A_{0,0} = 1$. The decay constant can be found using (3.32), which, for $f_s = 44100$ calculates as $\tau_{h,0} = 0.0869$. The equation for $h_0(n)$ now looks like

$$h_0(n) = e^{-nT/0.0869} \tag{5.2}$$

From (3.33), $h_0(n)$ can be scaled between our starting and ending pitch glide frequencies, $f_{x,0}$ and $f_{y,0}$, so that $f_{0,0}(n)$ takes on the form

$$f_{0,0}(n) = (100-40) * h(n) + 40. \tag{5.3}$$

The pitch glide, $f_{0,0}(n)$, is plotted in Figure 5.2. To create the pitch glide with our $z_{0,0}(n)$ kick drum oscillator, we must calculate $\theta_{0,0}(n)$ with (3.34) and use it along with $b_{0,0}$ in (3.22). Linear and/or square root pitch trajectories can be made through a similar process with the equations and instructions given in Section 3.10.2.

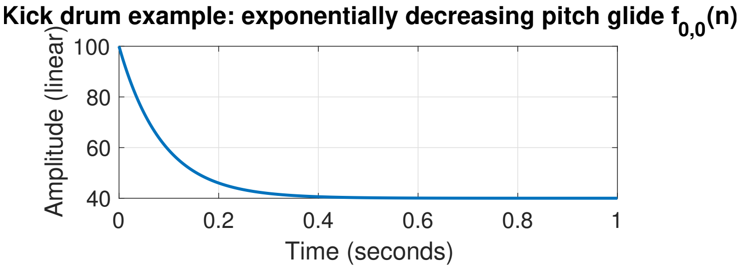


Figure 5.2: Kick drum example pitch glide $f_{0,0}(n)$. The pitch glide exponentially decreases from 100 Hz to 40 Hz and can be used with $z_{c,i}$ or $z_{0,i}$ oscillators.

To create this same pitch glide with the $z_{c,0}(n)$ oscillator, we first find $\omega_{0,i}(n) = 2\pi f_{0,0}(n)$, then we set $\omega_{c,0} = \omega_{0,0}(1) = 2\pi f_x = 2\pi 100$ so that $\omega_{0,0}(n) \leq \omega_{c,0}$ for all n. We then use (3.30) to solve for $B_0(n)$, which is shown in Figure 5.3.

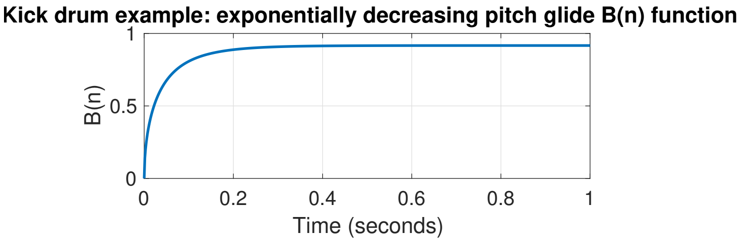


Figure 5.3: Kick drum example: $B(n)$ for exponentially decaying pitch glide. When using the $z_{\mathrm{c},i}(n)$ oscillator to synthesize the kick drum, $B(n)$ is used to create a pitch glide. This will create changes in both pitch and number of harmonics.

A graph of the synthesized kick drum with a pitch glide using both kinds of loopback FM oscillators is shown in Figure 5.4. With the $z_{\mathrm{c},i}(n)$ oscillator, $B_i(n)$ controls pitch and number of harmonics. The kick synthesized using the $z_{\mathrm{c},i}(n)$ oscillator sounds much brighter than the kick synthesized using the $z_{0,i}(n)$ oscillator and this is also reflected in the spectrograms in Figure 5.4. Audio examples of these two signals are provided as supplementary audio files *hsu_kick_pitchGlidez0.wav* and *hsu_kick_pitchGlidezc.wav*

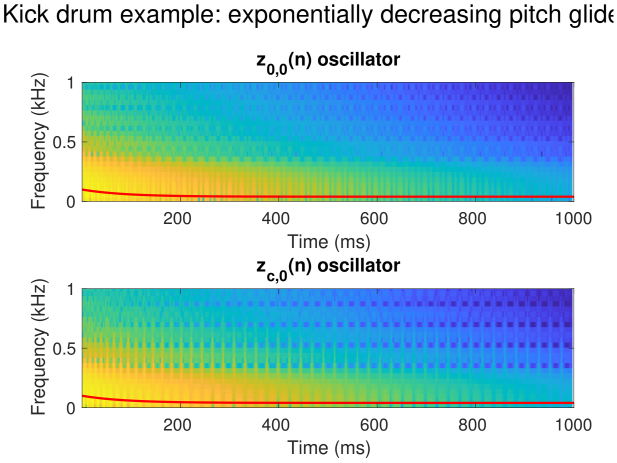


Figure 5.4: Kick drum example using $z_{0,i}(n)$ and $z_{\mathrm{c},i}(n)$ oscillators with a pitch glide. (Top) $z_{0,0}(n)$ synthesized with an exponentially decreasing pitch glide from 100 Hz to 40 Hz with $b_0 = 0.1$ and (Bottom) $z_{\mathrm{c},0}(n)$. The red line is the exponentially decreasing pitch glide $f_{0,0}(n)$. Because the pitch glide is controlled with B_0 in the bottom plot, the number of harmonics is affected and this can be seen as slightly more energy in the 500 Hz frequency range.

5.2.3 Varying the Number of Harmonics Over Time

For the kick drum created using the $z_{0,0}(n)$ oscillator, we can control how the number of harmonics varies over time by creating a function for $b_{0,0}(n)$. Section 3.10.3 shows how $b_{0,0}(n)$ can be set to a linear or exponential function to vary the number of harmonics over time. For the kick drum, we use a linearly decreasing function so that $b_{0,0}(n)$ decreases from 0.5 to 0 over the course of the signal. This creates a signal that has more higher frequency content right around the beginning of the signal, giving the attack a bit more impact. For a more realistic sounding signal, one may like to decrease $b_{0,0}(n)$ exponentially rather than linearly. An example of this signal is plotted in Figure 5.5 and an audio example is provided with the supplementary audio files as *hsu_kick_pitchAndTimbreVariation.wav*.

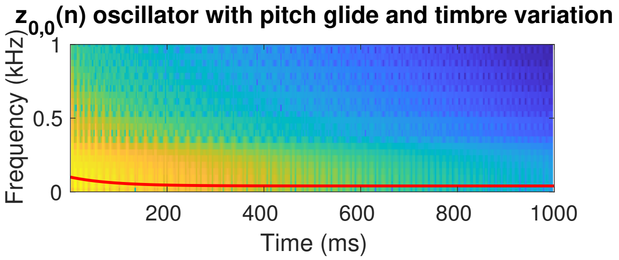


Figure 5.5: Kick drum example using a $z_{0,0}(n)$ oscillator with a pitch glide and linearly decreasing $b_0(n)$. Compared to the upper plot of Figure 5.4, there is more high frequency energy at the beginning of the signal. This is caused by the larger $b_0(n)$ value. That high frequency energy falls away over time as $b_0(n)$ decreases to 0.

Note that the kick drum created using the $z_{c,0}(n)$ oscillator already has a time-varying number of harmonics because the pitch and harmonics are both controlled by a single parameter, $B_0(n)$.

5.2.4 Applying 2$^{\text{nd}}$-Order Time-varying Allpass Filters for Further Non-linearities

If we were synthesizing a realistic kick drum, we would most likely end our synthesis at the previous step, but for the purposes of showing how the method fits together, in this section, we discuss how to further shape the timbre using the 2nd-order time-varying allpass filters described in 4.3.2. In order to drastically change the sound, we add extra sidebands to the signal by setting the time-varying allpass filter parameters according to $f_{m,0} = 100$, $M_0 = 120$, $f_{b,0} = 2500$, and $f_{\pi,0} = f_{0,0} = 100$. This changes the timbre to create an effect that sounds similar to a flam, adding dimension and making the sound a bit more hollow. A spectrogram of this synthesis is plotted in 5.6 and an audio example is provided with the supplementary files as *hsu_kick_timeVaryingAP.wav*. If M_i and $f_{m,i}$ are set to much larger values, such as 1000 or 2000, metallic timbres begin to emerge.

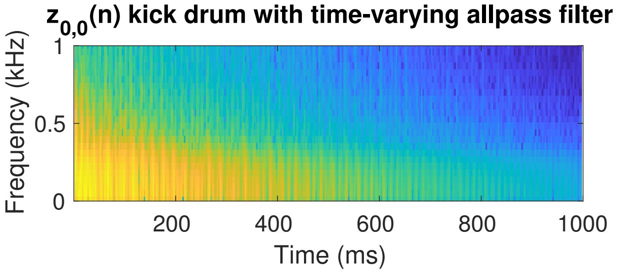


Figure 5.6: Kick drum example: a 2nd-order time-varying allpass filter applied to the kick drum synthesis using the $z_{0,0}(n)$ oscillator. Compared to Figure 5.5, there is more energy in this spectrogram from around 300 to 700 Hz in the beginning part of the signal.

5.2.5 Applying Commuted Synthesis for a Sense of Space

Commuted synthesis can be applied to the signal to give the sound a sense of space or to process an existing recording. Commuted synthesis can be applied whether or not the time-varying allpass filters have been used. Parameters for commuted synthesis are presented in Section 4.3.3. For the kick drum, any type of low frequency hit or thump could perform well as the acoustic resonator. Here, we use the sound of a taiko hit retrieved from freesound.org, and the taiko hit is processed using the loopback FM kick sound. We use a raised cosine excitation because it is slightly softer sounding than the noise burst and gives a sound that is more commonly associated with "membrane" timbres. To create a sharper attack, L is set to a small number of samples. Here, $L = 4$ samples. Figure 5.7 is a spectrogram of commuted synthesis applied to the kick signal after the time-varying allpass filter has been used. Compared to Figure 5.6, the spectrogram in Figure 5.7 looks more smeared and the signal takes longer to decay. Most signals tend to look more smeared after commuted synthesis due to the convolution operation. The time it will take for the signal to decay is the maximum over the length of the loopback FM MS signal and the acoustic resonator impulse response. An audio example of the commuted synthesis kick signal is included in the supplementary audio files as *hsu_kick_commutedSynthesis.wav*.

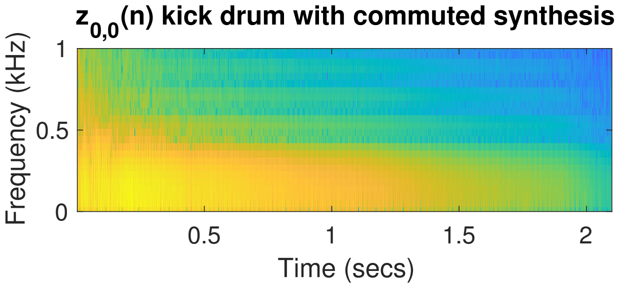


Figure 5.7: Kick drum example: commuted synthesis applied to the kick drum signal. The loopback FM kick drum signal is being used to process the sound of a taiko drum through commuted synthesis. The acoustic resonator impulse response is a recording of a struck taiko drum, and the excitation is a 4-sample long raised cosine function.

5.3 Synthesis Examples

While the loopback FM percussion synthesis method is capable of creating a variety of percussive sounds, this section covers five sound synthesis examples: the snare drum, marimba, wood block, tom tom, and circular plate. Although these modal frequencies are associated with real, physical instruments, the aim of this synthesis is not to recreate the naturally occurring sounds. Rather, we seek to synthesize many different types of sounds with nonlinearities similar to those that occur in percussion instruments. Although the modal frequencies for these examples are retrieved from other sources, the pitch trajectories have been carefully chosen by the author to illustrate the capabilities of this synthesis method. For these examples, differences between percussion synthesis using traditional and loopback FM MS are compared for the same modal frequencies, decaying amplitude envelopes, and commuted synthesis parameters. Time-varying allpass filters are also applied to these examples. Sound examples are included as supplementary material as noted within each synthesis example section.

5.3.1 Snare Drum

In this example, the sound of a snare is synthesized using two loopback FM oscillators $z_0(n)$, which are set to modal frequencies retrieved from [16]. The modal frequencies are $[185, 330]$, and they decrease exponentially to $[140, 280]$. $b_{0,i}$, which control the number of harmonics, are both set to 0.5 and do not vary over time. For the amplitude envelopes, the lowest modal frequency is given an initial amplitude of 1 and that decreases exponentially with a $T_{60,0} = 0.6$ seconds. The other modal frequency is given an initial amplitude of 0.2 with $T_{60,1} = 0.5$ seconds. The global amplitude envelope $w_g(n)$ is the same as the amplitude envelope for the lowest modal frequency. For commuted synthesis, the input excitation is a noise burst that lasts for 0.6 seconds and bandpass filtered between 120 and 8000 Hz, as snare drums are frequently synthesized using additive synthesis with noise bursts. The acoustic resonator is simply an impulse that has a value of 1 for the first sample and is 0 everywhere else. For the time-varying allpass filter versions of the snare, the parameters were set according to $f_{b,i} = [100, 200]$, $M_i = [1000, 250]$ and $f_m = [300, 2000]$.

Figure 5.8 shows the four variations for the snare drum synthesis and the sound examples are included as supplementary files called *hsu_snare_traditionalMS.wav*, *hsu_snare_loopbackFM.wav*, *hsu_snare_traditionalMSTimeVaryingAP.wav*, and *hsu_snare_loopbackFMTimeVaryingAP.wav*. In the graphs, the loopback FM MS version appears to have more energy than the traditional MS version from 0 to 1 kHz at the beginning of the signal. The time-varying allpass filter versions have much more energy around the 2 kHz region around the beginning of the signal. In comparing the sounds, the traditional MS without the pitch glide sounds quite similar to an electronic snare drum. With the pitch glide, the synthesis takes on the characteristics of an electronic tom tom drum. The time-varying allpass filters cause the timbre to sound much more metallic.

5.3.2 Marimba

Figure 5.9 compares the spectrograms of a marimba modeled as a bar with two free ends using traditional and loopback FM MS. This example sets $f_{0,i}$ to seven modal frequencies, retrieved from [51], and calculated as

$$f_{0,i} = \begin{cases} 440, & \text{for } i = 0 \\ 440\frac{(2i+3)^2}{3.011^2}, & \text{otherwise} \end{cases} \tag{5.4}$$

The marimba example is created using $z_{c,i}(n)$ oscillators and a pitch glide made by setting $B_i(n) = 0.9999^n$. The amplitude envelopes are decaying exponentials where initial amplitudes decrease exponentially from 1 for the first (lowest) modal frequency to 0.01 for the seventh (highest) modal frequency. Figure 5.10 is a plot of the amplitude envelopes used. For commuted synthesis, the excitation is an 8-sample long raised cosine excitation. The acoustic resonator is an ideal, open-closed tube synthesized using traditional MS. The marimba examples created using traditional MS and loopback FM MS with commuted synthesis are shown in the upper two graphs of Figure 5.9. The pitch glide and the delay in higher frequency components during the attack in the loopback FM MS version can be clearly seen. Examples of these syntheses are included in the supplementary audio files as *hsu_marimba_traditionalMS.wav* and *hsu_marimba_loopbackFM.wav*.

The two lower figures of Figure 5.9 show the marimba synthesized using traditional MS and loopback FM MS with time-varying allpass filters applied. These examples have not been processed by commuted synthesis. Sorted by lowest frequency modal component to highest frequency, the parameters for the allpass filters are: $M_i = [1000, 2000, 1500, 3000, 2300, 6000, 2500]$, $f_{m,i} = [78, 31, 42, 83, 100, 400, 300]$, and $f_{b,i} = 100[1, 2, \cdots, N_f]$. The time-varying allpass filters add modulation around the main frequency components in both the traditional MS and loopback FM MS graphs. Perceptually, the examples have a “buzzy” timbre. Sound examples are included as *hsu_marimba_traditionalMSTimeVaryingAP.wav* and *hsu_marimba_loopbackFMTimeVaryingAP.wav*.

5.3.3 Wood Block

Figure 5.11 shows the spectrograms of traditional MS and loopback FM MS, with and without commuted synthesis, for the sound synthesis of a wooden block. The modal frequencies for the wood block are found through a peak analysis of a wood block recording retrieved from freesound.org. The 10 most prominent frequencies from this analysis were used for synthesis:

$$\begin{aligned} f_{r,i} = [&321.7, 467.0, 1545, 589.5, 703.9, 846.5, \\ &1545, 1711.9, 1993.2, 2162.7]. \end{aligned} \tag{5.5}$$

The amplitude envelopes are quickly decaying exponentials. The initial amplitude of each of the amplitude envelopes decreases exponentially from 1 for the first modal frequency to 0.01 for the tenth modal frequency. The loopback FM MS example is created using $z_{0,i}(n)$ oscillators with linear pitch glides that decrease from $f_{0,i}$ to $\frac{f_{0,i}}{4}$. The synthesis versions without commuted synthesis can be seen in the upper two graphs of Figure 5.11. The linearly decreasing pitch glide and high frequency components can be observed in the loopback FM MS graph. Audio examples are provided as supplementary material and called *hsu_woodBlock_traditionalMS.wav* and *hsu_woodBlock_loopbackFM.wav*.

The lower two graphs of Figure 5.11 show how commuted synthesis affects the results. For commuted synthesis, the excitation is a 0.2-second long noise burst that has been bandpass filtered from 120 Hz to 12000 Hz. The acoustic resonator is a recording of a struck guitar body retrieved from freesound.org. The two commuted synthesis graphs look strikingly similar.

Sonically, they sound similar as well, though the traditional MS with commuted synthesis version seems to have a lower sounding frequency than that of the loopback FM MS with commuted synthesis version. Examples of these syntheses are included with the supplementary audio files as *hsu_woodBlock_traditionalMSCommutedSynthesis.wav* and *hsu_woodBlock_loopbackFMCommutedSynthesis.wav*.

5.3.4 Tom Tom

The spectrogram of a tom tom synthesized using traditional vs. loopback FM MS is shown in Figure 5.12 with and without commuted synthesis. The modal frequencies used to synthesize the tom tom are from [51] and are set to

$$f_{r,i} = 142 \cdot [1, 2.15, 3.17, 3.42, 4.09, 4.80, 4.94] \tag{5.6}$$

The amplitude envelopes are the same as those used for the marimba as shown in Figure 5.10. The loopback FM MS synthesis uses $z_{0,i}(n)$ oscillators, $b_{0,i} = -0.3$, and the pitch glide decreases linearly from $f_{0,i}$ to $\frac{f_{0,i}}{1.3}$. The traditional MS and loopback FM MS versions without commuted synthesis are plotted in the two upper graphs of Figure 5.12. There is more spectral energy up to about 5 kHz for the loopback FM MS version, and the linearly decreasing pitch glide can also be seen. For audio examples, see supplementary files *hsu_tomtom_traditionalMS.wav* and *hsu_tomtom_loopbackFM.wav*.

The two bottom graphs of Figure 5.12 show the effects of commuted synthesis. The excitation is a 0.02-second long noise burst filtered by a 2$^{\text{nd}}$-order Butterworth bandpass filter with frequency cutoffs at 120 Hz and 4000 Hz. The acoustic resonator is a recording of a taiko drum retrieved from freesound.org, so the loopback FM MS tom tom signal is being used to process the recording of the taiko drum. In Figure 5.12, there is more high frequency energy for the loopback FM MS than for the traditional MS, especially in the beginning of the signal. In the "Loopback FM MS with CS" graph has some more frequency spreading around the 1.5 to 3 kHz range. Perceptually, a strong pitch glide can be heard in the loopback FM MS with commuted synthesis version. Audio examples for these two syntheses are included with the supplementary audio files as *hsu_tomtom_traditionalMSCommutedSynthesis.wav* and *hsu_tomtom_loopbackFMCommutedSynthesis.wav*.

5.3.5 Circular Plate

In Figure 5.13, loopback FM MS of a simply-supported circular plate is compared to traditional MS of the same circular plate. Audio examples are provided as supplementary files *hsu_circularPlate_traditionalMS.wav* and *hsu_circularPlate_loopbackFM.wav*. The modal frequencies, retrieved from [51], are

$$f_{\mathrm{r},i} = f_0 \cdot [1, 2.80, 5.15, 5.98, 9.75, 14.09, \\ 14.91, 20.66, 26.99] \tag{5.7}$$

where $f_0 = 0.2287 c_L (h/a^2)$ for plate thickness $h = 0.005$m, plate radius $a = 0.09$m, and longitudinal wave speed $c_L = \sqrt{E/\rho(1-\nu^2)}$ with Young's modulus $E = 2 \cdot 10^{11}\mathrm{N/m}^2$, plate density $\rho = 7860\mathrm{kg/m}^3$, and Poisson ratio $\nu = 0.3$. The amplitude envelopes are decaying exponentials over time. The initial amplitude of these envelopes decreases exponentially as frequency increases from 1 for the lowest modal frequency to 0.5 for the highest modal frequency. By using $z_{\mathrm{c},i}(n)$ oscillators and setting $B_i = 0.99$ for all oscillators, this loopback FM MS synthesis creates drastic aliasing effects that result in an extremely "noisy" signal. Perceptually, the traditional MS output sounds like a clean bell sound, while the loopback FM MS sounds more like a noisy, struck cymbal. This difference can be clearly seen between the spectrograms plotted in Figure 5.13.

In Figure 5.14, commuted synthesis has been applied to the traditional MS and loopback FM MS signals to add a sense of space. The excitation is a 16-sample long raised cosine envelope and the acoustic resonator is a room impulse response retrieved from echothief.com. Sound examples are included as supplementary files *hsu_circularPlate_traditionalMSCommutedSynthesis.wav* and *hsu_circularPlate_loopbackFMCommutedSynthesis.wav*.

Figure 5.15 adds time-varying allpass filters to the signals from Figure 5.14. For all modal components, the allpass filter parameters are $M_i = 11500$, $f_{m,i} = 1000$, and $f_{b,i} = 2750$. This adds extra spectral components into the signal and gives both the traditional MS and loopback FM MS signals an inharmonic, dissonant sound. The loopback FM MS version also sounds like it is set a bit lower in frequency. The supplementary audio files include examples of these signals, and they are named *hsu_circularPlate_traditionalMSTimeVaryingAPCommutedSynthesis.wav* and *hsu_circularPlate_loopbackFMTimeVaryingAPCommutedSynthesis.wav*.

5.4 Conclusions

This chapter includes material coauthored with Professor Tamara Smyth as it appears in

Jennifer Hsu and Tamara Smyth. Percussion synthesis using loopback frequency modulation oscillators. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

The dissertation author was the primary researcher and author of this paper. This chapter has given a step-by-step explanation of how to synthesize the sound of a kick drum using the loopback FM percussion synthesis method. Synthesis examples using the modal frequencies of a snare, marimba, wood block, tom tom, and circular plate have been presented. In Chapter 6, we introduce a real-time software application of this synthesis method. We also compare this technique to previous percussion synthesis models, review the musical implications of such an instrument, and consider future research directions.

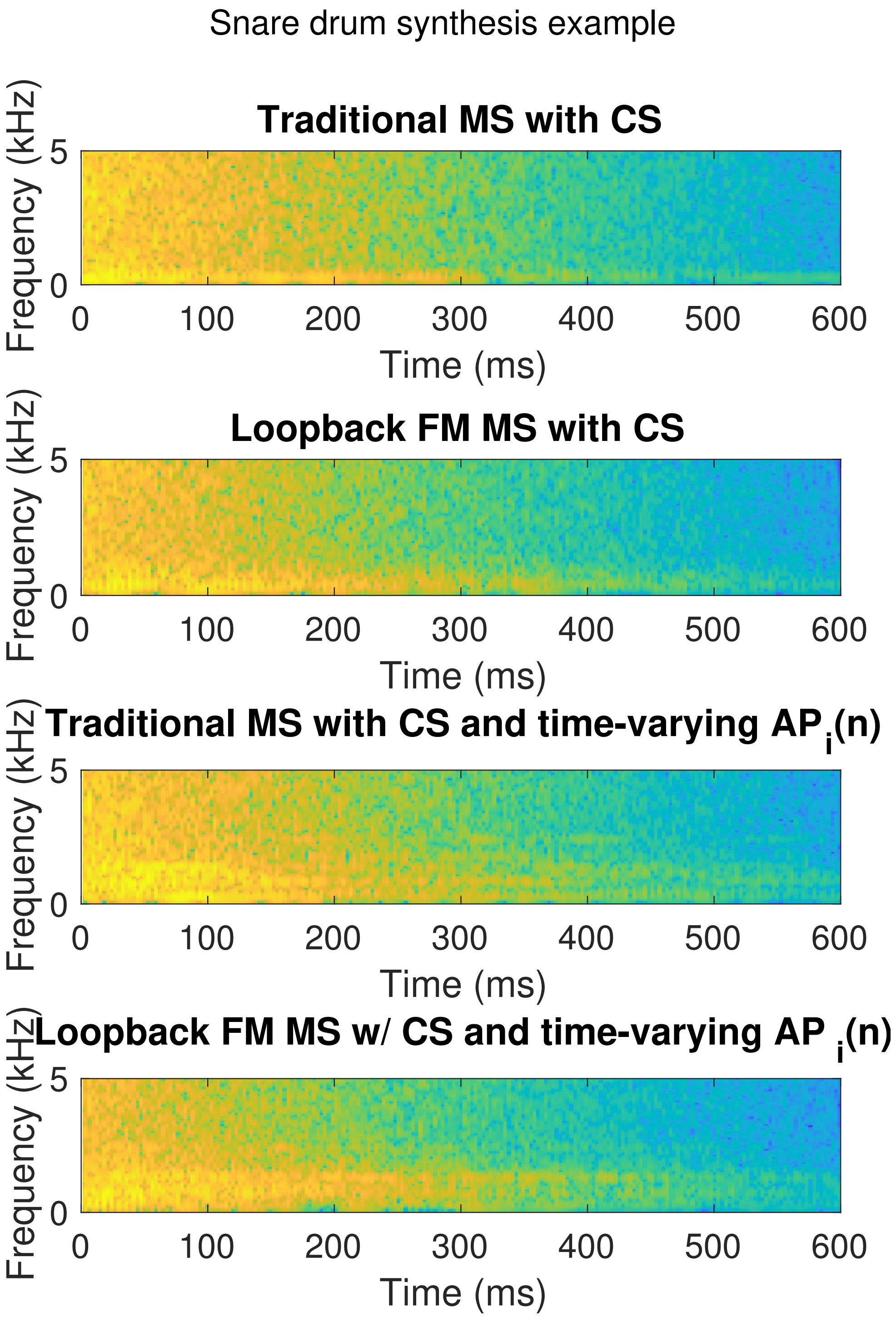


Figure 5.8: Traditional vs Loopback FM MS with and without time-varying allpass filters for the snare drum synthesis.

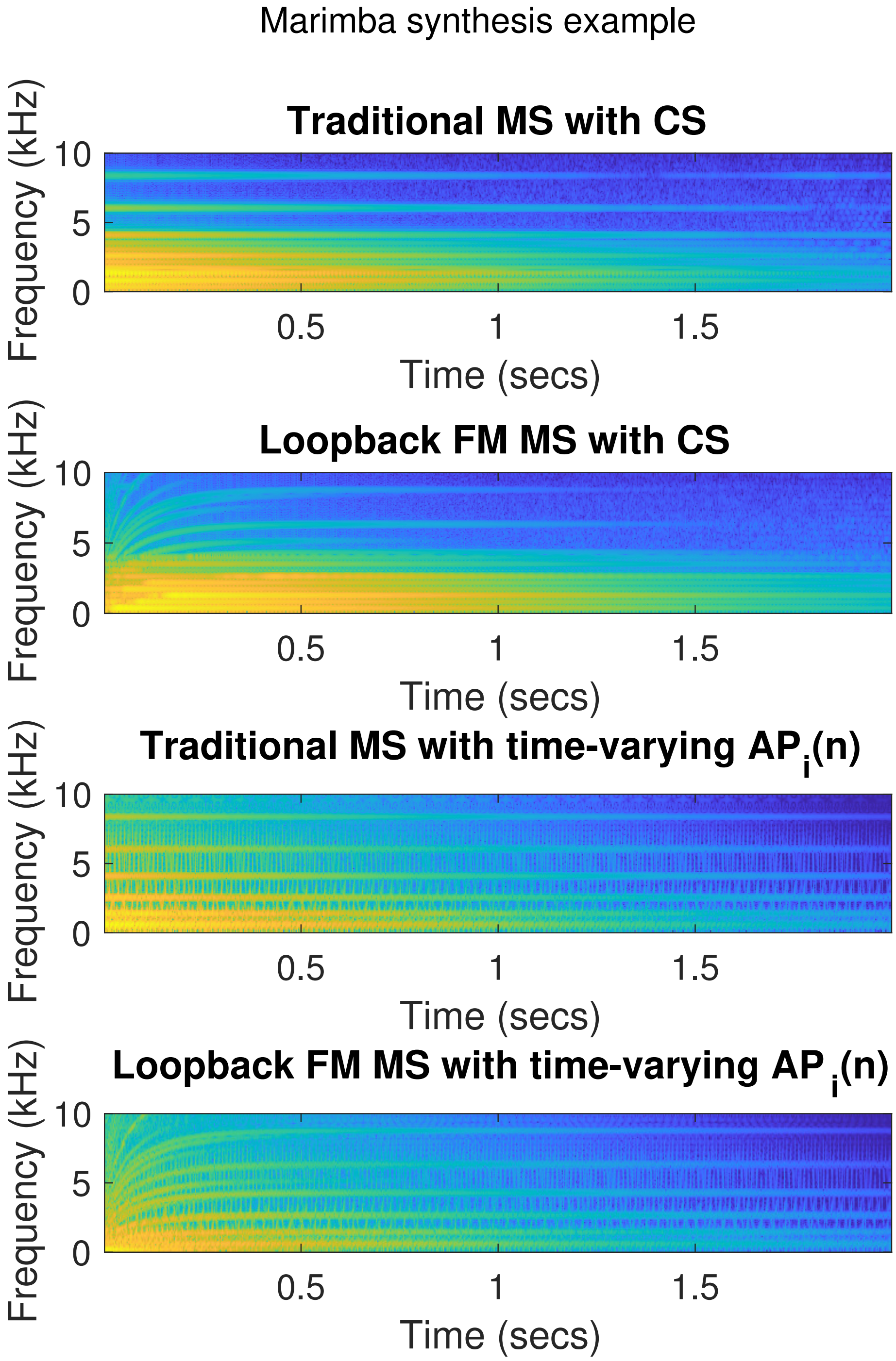


Figure 5.9: Traditional vs Loopback FM MS with and without time-varying allpass filters for the marimba synthesis. The marimba synthesis uses the modal frequencies of an ideal bar with two open ends. The excitation is a raised cosine and the acoustic resonator is an ideal tube synthesized using traditional MS. Commuted synthesis has been applied to the top two signals, and time-varying allpass filters have been applied to the bottom two graphs.

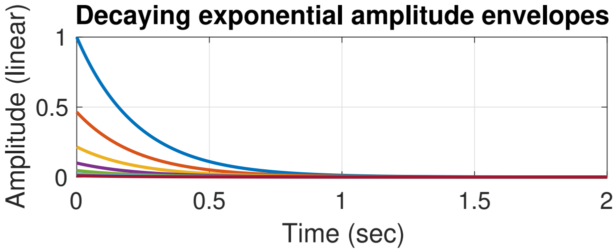


Figure 5.10: Amplitude envelopes for marimba synthesis. The marimba synthesis amplitude envelopes are decaying exponentials. The initial amplitude of the envelopes is inversely proportional to the modal frequency of the oscillator that is paired with the envelope.

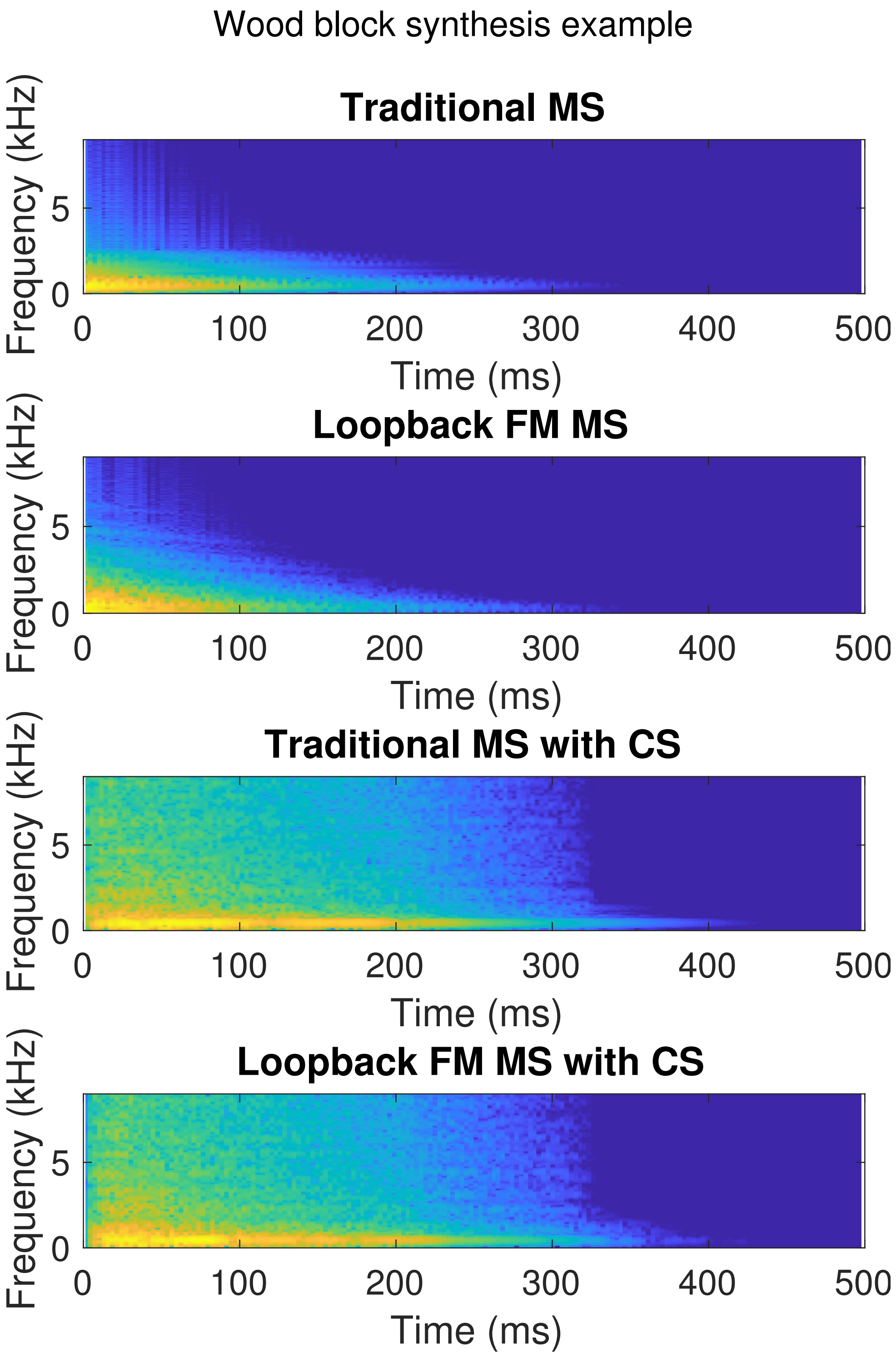


Figure 5.11: Traditional (top) vs. Loopback FM MS (bottom) using prominent frequencies picked out from a recording of a wood block. For the two commuted synthesis examples (the two lower graphs), the excitation is a filtered noise burst and the acoustic resonator impulse response a recording of a struck guitar body.

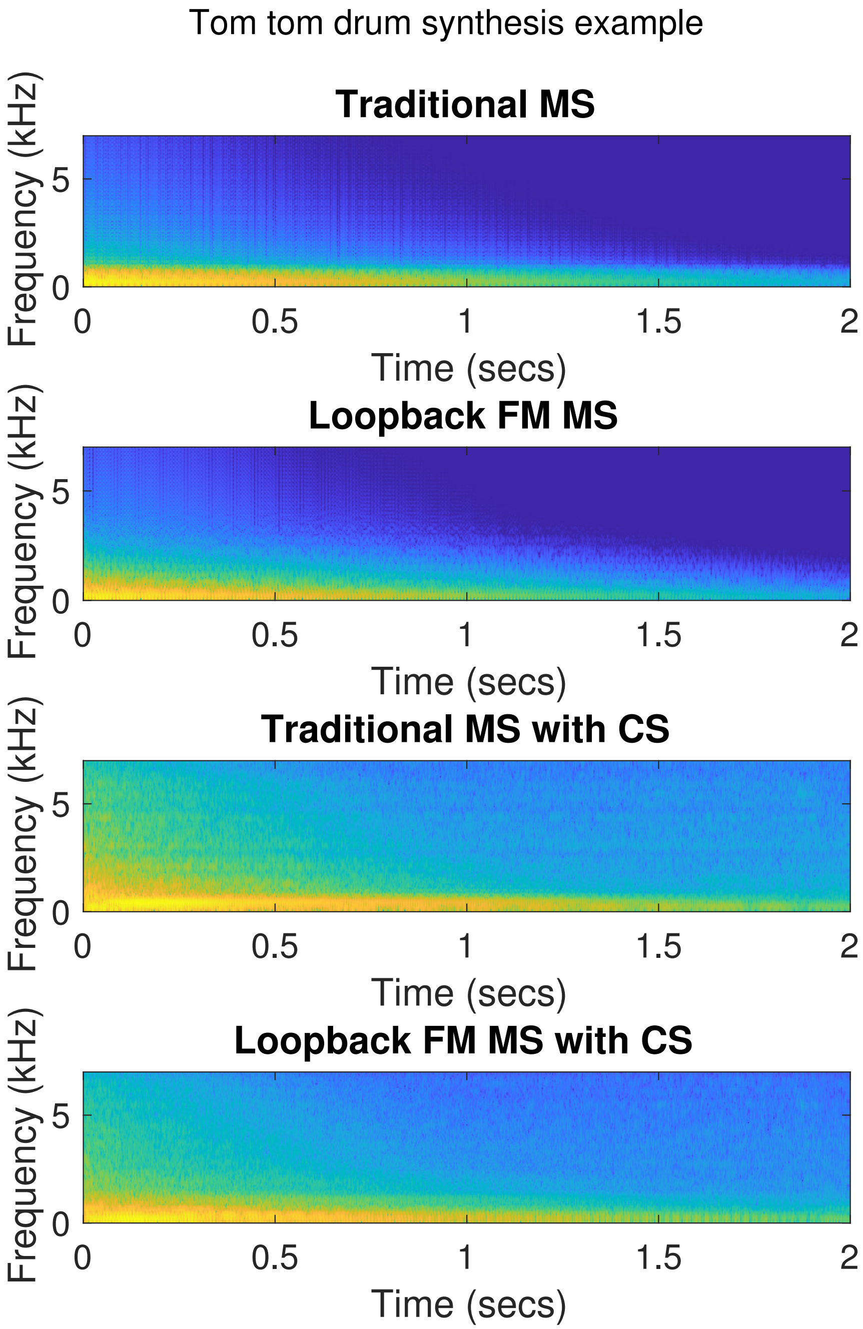


Figure 5.12: Traditional vs Loopback FM MS for a tom tom with and without commuted synthesis. Traditional (top) vs. Loopback FM MS (second from the top) using the modal frequencies of a tom tom. Commuted synthesis has been applied to the signals in the two bottom graphs so that the loopback FM MS signal is being used to process a recording of a taiko drum. The excitation is filtered noise.

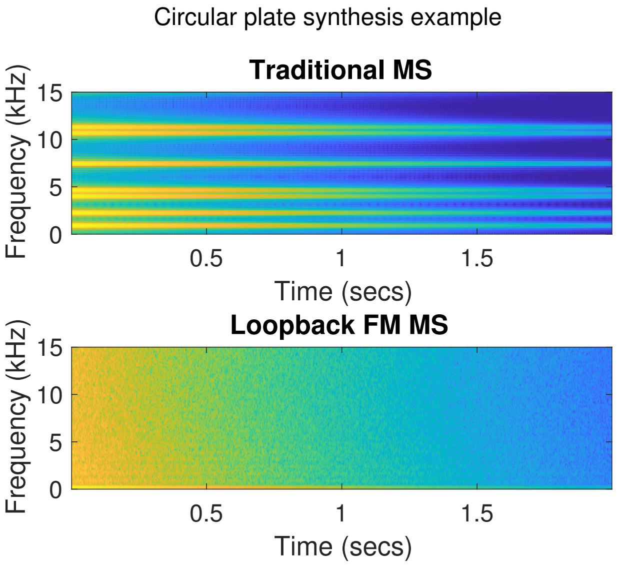


Figure 5.13: Traditional vs Loopback FM MS for a circular plate. Traditional (top) vs. Loopback FM MS (bottom) using the modal frequencies of a simply-supported circular plate. The loopback FM MS version sounds like a struck piece of sheet metal, possibly due to aliasing, while the traditional MS version sounds like a bell.

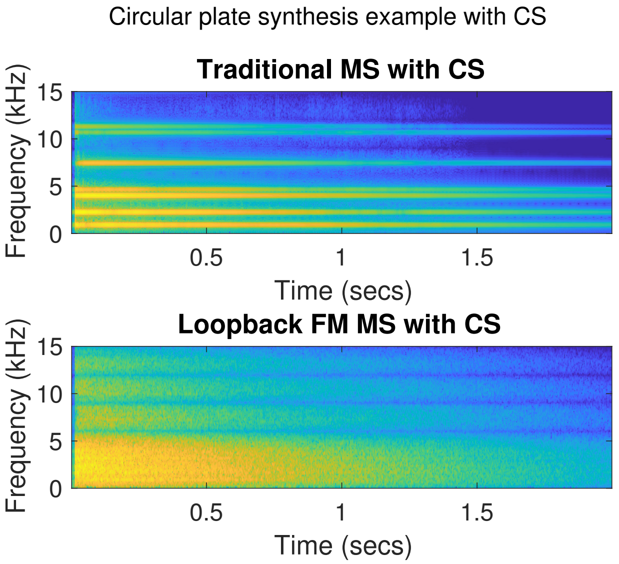


Figure 5.14: Traditional vs Loopback FM MS for a circular plate with commuted synthesis. Commuted synthesis is applied to the signals from Figure 5.13. The excitation is a raised cosine and the resonator is a room impulse response.

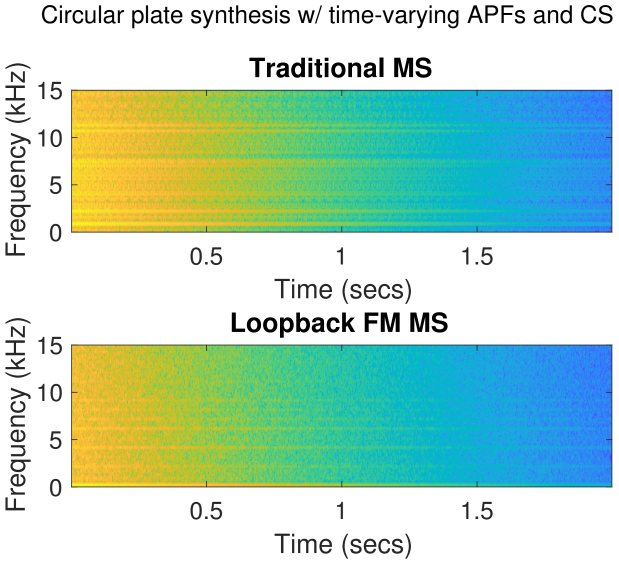


Figure 5.15: Traditional vs Loopback FM MS for a circular plate with commuted synthesis and time-varying allpass filters. Time-varying allpass filters have been applied to the signals from Figure 5.14. The result is a more inharmonic sounding signal from the traditional MS and a lower pitched signal from the loopback FM MS version.

Chapter 6

Applications and Conclusion

6.1 Introduction

This work began by briefly reviewing existing percussion synthesis techniques including MS, FDMs, and DWMs along with nonlinear effects that occur in percussion instruments. We then took an in-depth look at percussion synthesis models that inspired the research covered in this document. One of the key ideas is that incorporating nonlinearities in percussion synthesis models is more computationally expensive than a synthesis model without nonlinearities [7, 5]. Another important notion is how previous research that sought to derive an analytical solution to the 2D DWM paved the pathway for moving from a physics-based approach to an abstract synthesis method. We found that the transfer function for a 2D DWM with a single input and output point could be decomposed into a sum of resonating filters. This configuration reflects modal synthesis in both theory and implementation, and so we moved from the physical to the abstract domain.

By moving to a modal approach and replacing the resonating filters (or sinusoids) with nonlinear, loopback FM oscillators, we are able to synthesize novel, abstract percussion sounds using the loopback FM percussion synthesis method. Loopback FM creates complex spectra and pitch glides similar to the nonlinear effects observed in existing percussion instruments. Enhancements for creating noisy and metallic sounds and adding a sense of space to the syntheses were covered. The synthesis technique allows for parametric control of musical dimensions including sounding frequency, decay time, timbre, and pitch glide. Synthesis examples of a kick drum, snare, marimba, wood block, tom tom, and circular plate were examined.

In this chapter, we begin by qualitatively examining this percussion synthesis method. Then we introduce a real-time implementation of the loopback FM percussion synthesis technique and conceptualize a software synthesizer that uses this synthesis method. Finally, the musical implications and future research directions are considered.

6.2 Qualitative Evaluation

In reviewing the current state of the loopback FM percussion synthesis method, there are parts about it that I am pleased with and other parts that I feel could be improved. This method is great in that it is capable of making sounds that are eclectic, experimental, and unique, and it can synthesize signals in real-time. Changes to the musical parameters reflect straightforward changes to the sonic qualities in the resulting synthesis. Pitch glides are easy to specify, decay times directly affect the synthesis, and timbre controls darkness and brightness in an intuitive way.

When comparing this model to high-quality physics-based, nonlinear percussion models, such as the nonlinear plate model presented in [7], the loopback FM percussion synthesis technique does not create signals that sound as natural, because it is not based on physical calculations. From preliminary experimentation, it seems that although the loopback FM system is capable of synthesizing pitch glides and the cascade from low to high frequency energy as shown in Figure 4.9, the method cannot create the same type of gong-like sounds that the nonlinear plate model from [7] is capable of synthesizing. In my experiments, I used the modal components from the linear plate model and attempted to create the sound of the nonlinear plate model by setting values for the loopback FM percussion synthesis parameters. Setting all of the individual parameters by hand is difficult and time-consuming, and the resulting synthesis sounded more like a bell with a pitch glide than a nonlinear plate. It is possible that the nonlinear plate synthesis could be recreated with the loopback FM percussion synthesis technique if more time is spent tuning the parameters.

Of all the synthesis examples shown in the previous chapter, the circular plate example presented in Section 5.3.5 is the most similar in timbre to the sound created by the nonlinear plate model. That circular plate example uses $z_{\mathrm{c},i}(n)$ oscillators and exploits the aliasing that occurs due to numerical error in order to create the striking crash-like timbres. The nonlinear plate model has a clearly decreasing pitch glide, and to recreate this decreasing pitch glide with $z_{\mathrm{c},i}(n)$ oscillators, $B_i(n)$ would need to increase over time. B_i is already set to 0.99 in the circular plate example and when B_i are increased too far towards 1, the loopback FM output signal begins to break down due to numerical error.

Although it seems like it may make more sense to recreate the sound of the nonlinear plate using $z_{0,i}(n)$ oscillators since pitch and the number of harmonics can be independently controlled, we run into another issue. If the $z_{\mathrm{c},i}(n)$ oscillators are replaced with $z_{0,i}(n)$ oscillators, the resulting sound output is not similar at all. The $z_{0,i}(n)$ oscillator synthesis sounds like an inharmonic, buzzy bell sound and nothing like the crashing, noisy timbre created by the numerical errors of the $z_{\mathrm{c},i}(n)$ oscillators. Perhaps the next step is to take a look at how to recreate the aliasing of $z_{\mathrm{c}}(n)$ oscillators with $z_0(n)$ oscillators so that we can have full control over the pitch, number of harmonics, and perceptually pleasing aliasing effects.

6.3 A Real-time Loopback FM Percussion Synthesis Implementation

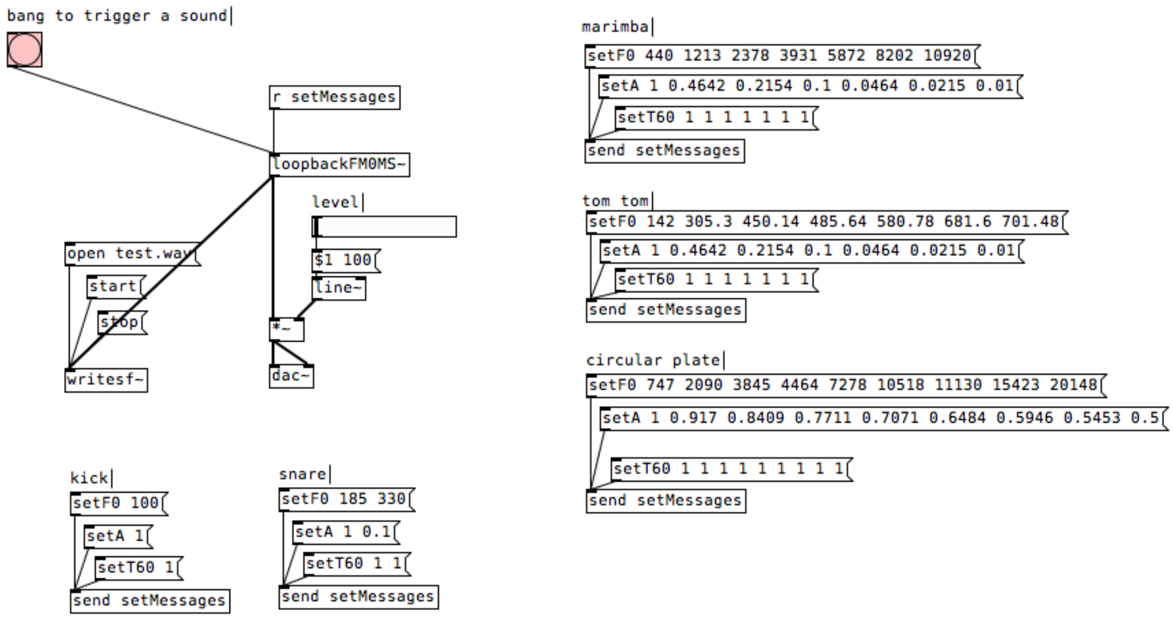
The loopback FM MS percussion synthesis technique has been implemented as a Pure Data (Pd) external in the C programming language. Efforts to include the time-varying allpass filters and commuted synthesis enhancements are in progress. A screenshot of a Pd patch using the external is shown in Figure 6.1. The inputs to the system are given in Table 6.1.

Table 6.1: A table of the inputs for the Max/MSP implementation of the loopback FM percussion synthesis method.

Input type	**Description**
bang	triggers a hit
list of floats	modal (sounding) frequencies
list of floats	ending sounding frequencies for pitch glide
list of floats	initial amplitudes for each mode
list of floats	amplitude envelope T_{60} decay times for each mode
string	'linear' or 'exp' : pitch glide function type
list of floats	initial b_0 values (if $z_0(n)$ oscillators are used)
list of floats	end b_0 values (if $z_0(n)$ oscillators are used)
string	'linear' or 'exp' : $b_0(n)$ trajectory function type
string	'z0' or 'zc' : indicates which loopback FM oscillator to use

6.4 Ideas for a Software Synthesizer

While the Pd loopback FM MS percussion synthesis implementation may be useful for musicians that are more proficient with graphical programming languages and the synthesis algorithm, I would like to create a software synthesizer in the form of an audio plugin for use in popular digital audio workstations (DAWs). This format would be more accessible for electronic musicians that may not have a background in computer music and engineering. The loopback FM MS percussion method has many parameters for each mode in addition to parameters for



```
bang to trigger a sound
r setMessages
loopbackFM0MS~
level
$1 100
line~
*~
dac~
open test.wav
start
stop
writesf~

marimba
setF0 440 1213 2378 3931 5872 8202 10920
setA 1 0.4642 0.2154 0.1 0.0464 0.0215 0.01
setT60 1 1 1 1 1 1 1
send setMessages

tom tom
setF0 142 305.3 450.14 485.64 580.78 681.6 701.48
setA 1 0.4642 0.2154 0.1 0.0464 0.0215 0.01
setT60 1 1 1 1 1 1 1
send setMessages

circular plate
setF0 747 2090 3845 4464 7278 10518 11130 15423 20148
setA 1 0.917 0.8409 0.7711 0.7071 0.6484 0.5946 0.5453 0.5
setT60 1 1 1 1 1 1 1 1 1
send setMessages

kick
setF0 100
setA 1
setT60 1
send setMessages

snare
setF0 185 330
setA 1 0.1
setT60 1 1
send setMessages
```

Figure 6.1: The loopback FM MS percussion synthesis implementation in Pd. Loopback FM MS is implemented as an external object in Pd called loopbackFMMS0 . Messages setting the synthesis parameters are first fed into the loopbackFMMS0 object, and the bang object at the upper left corner triggers the synthesis.

commuted synthesis and the time-varying allpass filters. Because it is a complicated system, there would be a basic layout that is more suitable for new users and a deeper, more detailed layout for experienced users. With the basic layout, parameters are grouped into macro parameters that control multiple, algorithm parameters at once. For the detailed layout, users would be able to access individual modal frequencies, decay times, loopback FM parameters, etc.

Such a system would have presets for major types of percussive sounds where all the synthesis parameters are already set to create a default sound. These presets would include many of the sounds presented in Chapter 5 such as kick, snare, marimba, wood block, and circular plate. Additional categories could be membrane or bell. The number of modes used for each preset is set. For the basic layout, the macro parameters for each of these sounds would include fundamental (or sounding) frequency, decay time, and a timbre knob that changes between a brighter or darker sound. These three parameters would change multiple modal frequencies, amplitude envelopes, and loopback FM parameters simultaneously. The detailed layout would allow a user to modify modal frequencies, decay envelopes, loopback FM feedback parameters, and the option to include more modes in the model.

For the basic layout, there would be an option to include time-varying allpass filters. One or two macro parameters that encompass multiple time-varying allpass filter parameters could be used to add nonlinearities and noisiness to the output. The detailed layer would allow access to the time-varying allpass filter parameters associated with each loopback FM oscillator. Commuted synthesis would also be an option in the basic layout, and the user could choose between different excitation types and their parameters as well as importing their own resonator impulse responses.

By separating the synthesizer into basic and detailed layouts, a user would be able to make sounds right away and have some parameters to change the sound to their liking. As the user becomes more familiar with the system and the types of sounds that are possible, he or she may feel more comfortable digging into the detailed layout and controlling finer aspects of the sound.

6.5 Musical Implications

As the creator of this method, I envision the experimental and eclectic sounds from the loopback FM percussion synthesis technique being used in popular and experimental electronic music. The more traditional percussive sounds, such as the kick and snare examples presented in Chapter 5 could definitely be used as the main percussive instrument of a rhythmic beat in pop and electronic music. The unique and unconventional percussive sounds could be used to accent a rhythmic beat pattern created using samples or other drum synthesizers/machines. The circular plate-type sounds are great for impactful sections in music, such as right before a moment of silence or right after a build. The sounds with whimsical pitch glides, such as the marimba example shown in Chapter 5, could fare well as accents to rhythmic beats, similar to the way bongos are used in contemporary electronic music. The system could also be used to create risers right before an impactful section within a song by reversing the amplitude envelopes in time. One could make even more dramatic risers by reversing the resonator impulse response for a reverse reverb effect. An example of a riser is included in the supplementary files as *hsu_reverseCircularPlate_loopbackFMTimeVaryingAPCommutedSynthesis.wav*.

Furthermore, although this method is presented for percussion synthesis, the technique can be used in other ways. If the modal frequencies used are programmed to be harmonically related, then the synthesis will have a clear fundamental, sounding frequency. This can be used to create melodic instruments with a percussive quality.

One of the motivating ideas for this research was a sample library of metallic hits. My biggest issue with using those percussion samples is that the sound is static and changing the sound can create undesirable artifacts. The advantage of this percussion synthesis method is that pitch glide, timbre, and note length can all be modified to create variations in the sound, resulting in dynamic timbres and textures. Because the technique uses abstract synthesis techniques, the loopback FM percussion synthesis method gives the user the ability to create sounds that do not exist in the physical world. This adds a unique sonic identity to syntheses made with this system and inspires music creators to look beyond their understanding of music, encouraging them to experiment with novel, unusual sounds.

6.6 Future Work and Conclusion

As explained in 6.2, a future research direction involves investigating the aliasing that occurs with large carrier frequencies and large B values for the sample-by-sample rotation loopback FM equation and recreating those aliasing effects using the closed form representation. This will be a step towards imitating the sounds that other physics-based nonlinear percussion synthesis systems are capable of making, but this method would be able to do so in real-time.

Another topic for future research is the coupling of oscillators and investigating how the results of coupled oscillators could improve percussion synthesis. Loopback FM is currently self-coupled. As a first step, we could take a look at the behavior of two loopback FM oscillators coupled to one another. The analysis would look at how the individual carrier/sounding frequency of each oscillator affects the resulting, coupled oscillator sounding frequency and what kind of timbral changes to expect. Then, we could examine coupled loopback FM oscillators in a MS method and explore the sound synthesis possibilities.

Other research interests involve the musical parameters. In the existing implementation, it is difficult to control the current parameters without knowing what each parameter actually does, and it is difficult to input exact numbers for various parameters. It would be much easier if intuitive parameter names were included. For example, a parameter name like "Brighter" could modify b_0 values. A more informed method to set the amplitude envelope for each modal component could be helpful. This would involve analyzing the amplitude envelopes of real percussion instrument recordings and applying them to the loopback FM MS synthesis. It is also necessary to look into ways to combine the synthesis parameters into macro parameters so that the method can be used in the proposed software synthesizer described in Section 6.4.

The loopback FM MS percussion synthesis method is a real-time method that allows for a music creator to synthesize unique, eclectic and experimental percussive sounds. With this technique, one can create a wide variety of percussive sounds using musical parameters that control sounding frequency, pitch glide, timbre, and note duration. Future work could improve on this percussion synthesis method by improving the parametric control, adding in coupled oscillators, and improving the sound synthesis so that it is capable of creating an even wider variety of percussive sounds.

Bibliography

[1] Jean Marie Adrien. The missing link: Modal synthesis. In *Representations of Musical Signals*, pages 269–295. The MIT Press, 1991.

[2] Marc Aird and Joel Laird. Extending digital waveguides to include material modelling. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-01)*, Limerick, Ireland, Dec. 2001.

[3] Marc Aird, Joel Laird, and John P ffitch. Modelling a drum by interfacing 2-d and 3-d waveguide meshes. In *Proceedings of the International Computer Music Conference*, pages 82–85, Berlin, Germany, Aug 2000.

[4] Mitsuko Aramaki, Mireille Besson, Richard Kronland-Martinet, and Sølvi Ystad. Controlling the perceived material in an impact sound synthesizer. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(2):301–314, Feb. 2011.

[5] Federico Avanzini and Riccardo Marogna. A modular physically based approach to the sound synthesis of membrane percussion instruments. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(4):891–902, May 2010.

[6] Lydia Ayers and Andrew Horner. Synthesizing a Javanese gong ageng. In *Proceedings of the International Computer Music Conference*, Barcelona, Spain, 2005.

[7] Stefan Bilbao. Sound synthesis for nonlinear plates. In *Proceedings of the 8th International Conference on Digital Audio Effects (DAFx'05)*, Madrid, Spain, Sept. 2005.

[8] Stefan Bilbao. Time-varying generalizations of all-pass filters. *IEEE Signal Processing Letters*, 12(5):376–379, May 2005.

[9] Stefan Bilbao. A family of conservative finite difference schemes for the dynamical von Karman plate equations. *Numerical Methods for Partial Differential Equations*, 24(1):193–216, Jan. 2008.

[10] Stefan Bilbao. Percussion synthesis based on models of nonlinear shell vibration. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(4):872–880, May 2010.

[11] Stefan Bilbao, Olivier Thomas, Cyril Touzé, and Michele Ducceschi. Conservative numerical methods for the full von kármán plate equations. *IEEE Transactions on Audio, Speech, and Language Processing*, 31(6):1948–1970, Nov. 2015.

[12] Stefan Bilbao and Maarten van Walstijn. A finite difference plate model. In *Proceedings of the International Computer Music Conference*, Barcelona, Spain, Sept. 2005.

[13] Jeffrey N. Chadwick, Steven S. An, and Doug L. James. Harmonic shells: A practical nonlinear sound model for near-rigid thin shells. *ACM Trans. Graph.*, 28(5):119:1–119:10, Dec 2009.

[14] Antoine Chaigne and Christophe Lambourg. Time-domain simulation of damped impacted plates. i. theory and experiments. *Journal of the Acoustical Society of America*, 109:1433–1447, March 2001.

[15] John M. Chowning. The synthesis of complex audio spectra by means of frequency modulation. *Journal of the Audio Engineering Society*, 21(7):526–534, Sept 1973.

[16] James J. Clark. Advanced programming techniques for modular synthesizers. http://www.cim.mcgill.ca/~clark/nordmodularbook/nordmodularbook.pdf. Accessed: 2019-04-10.

[17] Perry Cook. Physically informed sonic modeling (PhISM): Synthesis of percussive sounds. *Computer Music Journal*, 21(3):38–49, 1997.

[18] Michele Ducceschi and Cyril Touzé. Modal approach for nonlinear vibrations of damped impacted plates: Application to sound synthesis of gongs and cymbals. *Journal of Sound and Vibration*, 344:313–331, Jan. 2015.

[19] Scott A. Van Duyne. Coupled mode synthesis. In *Proceedings of the International Computer Music Conference*, pages 248–251, Thessaloniki, Greece, 1997.

[20] Scott A. Van Duyne and Julius O. Smith. Physical modeling with the 2-d digital waveguide mesh. In *Proceedings of the International Computer Music Conference*, pages 40–47, Tokyo, Japan, 1993.

[21] Scott A. Van Duyne and Julius O. Smith. A simplified approach to modeling dispersion caused by stiffness in strings and plates. In *Proceedings of the International Computer Music Conference*, pages 407–410, Aarhus, Denmark, Sept. 1994.

[22] Gerhard Eckel, Francisco Iovino, and René Caussé. Sound synthesis by physical modelling with Modalys. In *International Symposium on Musical Acoustics*, pages 479–482, Dourdan, France, July 1995.

[23] Harvey Fletcher and Irvin G. Bassett. Some experiments with the bass drum. *Journal of the Acoustical Society of America*, 64(6):1570–1576, Dec. 1978.

[24] Neville H. Fletcher. Nonlinear dynamics and chaos in musical instruments. In David G. Green and Terry Bossomaier, editors, *Complex Systems: from Biology to Computation*, pages 106–117. IOS Press, Amsterdam, 1993.

[25] Neville H. Fletcher. The nonlinear physics of musical instruments. *Reports on Progress in Physics*, 62:723–764, 1999.

[26] Federico Fontana and Davide Rocchesso. A new formulation of the 2d waveguide mesh for percussion instruments. In *Proceedings of XI Colloquium on Musical Informatics*, Bologna, Italy, Nov. 1995.

[27] Federico Fontana and Davide Rocchesso. Simulations of membrane-based percussion instruments. In *Proceedings Workshop: Sound Synthesis by Physical Modeling*, Firenze, Italy, Centro Tempo Reale, June 1996.

[28] Federico Fontana and Davide Rocchesso. Physical modeling of membranes for percussive instruments. *Acustica United with Acta Acustica*, 84(3):529–542, May/June 1998.

[29] William G. Gardner. Efficient convolution without input-output delay. *Journal of the Audio Engineering Society*, 43(3):127–136, March 1995.

[30] Andrew Horner. Nested modulator and feedback FM matching of instrument tones. *IEEE Transactions on Speech and Audio Processing*, 6(4):398–409, July 1998.

[31] Jennifer Hsu and Tamara Smyth. Percussion synthesis using loopback frequency modulation oscillators. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

[32] Julius O. Smith III. On the equivalence of the digital waveguide and finite difference time domain schemes. https://ccrma.stanford.edu/~jos/wgfdtd/. Accessed: 2019-04-10.

[33] Julius O. Smith III. Physical modeling using digital waveguides. *Computer Music Journal*, 16(4):74–91, 1992.

[34] Julius O. Smith III. Efficient synthesis of stringed musical instruments. In *Proceedings of the International Computer Music Conference*, Tokyo, Japan, Sept. 1993.

[35] Julius O. Smith III. *Introduction to Digital Filters: With Audio Applications*. W3K Publishing, 2007. Online book, accessed: 2019-04-10.

[36] Julius O. Smith III and Romain Michon. Nonlinear allpass ladder filters in Faust. In *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx-11)*, Paris, France, Sept. 2011.

[37] Francisco Iovino, René Caussé, and Richard Dudas. Recent work around modalys and modal synthesis. In *Proceedings of the International Computer Music Conference*, Thessaloniki, Greece, 1997.

[38] Marguerite Jossic, David Roze, and Thomas Hélie. Energy shaping of a softening duffing oscillator using the formalism of port-hamiltonian systems. In *Proceedings of the 20th International Conference on Digital Audio Effects (DAFx'17)*, Edinburgh, UK, Sept. 2017.

[39] Jari Kleimola, Victor Lazzarini, Vesa Vlimki, and Joseph Timoney. Feedback amplitude modulation synthesis. *EURASIP Journal on Advances in Signal Processing*, 2011(1), Dec. 2010.

[40] Jari Kleimola, Jussi Pekonen, Henri Penttinen, and Vesa Välimäki. Sound synthesis using an allpass filter chain with audio-rate coefficient modulation. In *Proceedings of the 12th International Conference on Digital Audio Effects (DAFx-09)*, Como, Italy, Sept. 2009.

[41] Joel Laird, P. Masri, and C. N. Canagarajah. Efficient and accurate synthesis of circular membranes using digital waveguides. In *IEE Colloquium, Audio and Music Technology: The Challenge of Creative DSP*, pages 12/1–12/6, London, UK, Nov. 1998.

[42] Christophe Lambourg, Antoine Chaigne, and Denis Matignon. Time-domain simulation of damped impacted plates. ii. numerical model and results. *Journal of Acoustical Science and Technology*, 109(4):1433–1447, April 2001.

[43] Victor Lazzarini, Joseph Timoney, Jussi Pekonen, and Vesa Välimäki. Adaptive phase distortion synthesis. In *Proceedings of the 12th International Conference on Digital Audio Effects (DAFx-09)*, Como, Italy, Sept. 2009.

[44] Riccardo Marogna, Federico Avanzini, and Bálazs Bank. Energy based synthesis of tension modulation in membranes. In *Proceedings of the 13th International Conference on Digital Audio Effects (DAFx-10)*, Graz, Austria, Sept. 2010.

[45] Joseph Derek Morrison and Jean-Marie Adrien. MOSIAC: A framework for modal synthesis. *Computer Music Journal*, 17(1):45–56, 1993.

[46] Damian T. Murphy, Chris J. C. Newton, and David M. Howard. Digital waveguide mesh modelling of room acoustics: Surround sound, boundaries, and plugin implementation. In *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-02)*, Limerick, Ireland, Dec. 2001.

[47] Jussi Pekonen. Coefficient-modulated first-order allpass filter as distortion effect. In *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx-08)*, Espoo, Finland, Sept. 2008.

[48] Stefan Petrausch and Rudolf Rabenstein. Tension modulated nonlinear 2d models for digital sound synthesis with the functional transformation method. In *Proceedings of EUSIPCO-05, Thirteenth European Signal Processing Conference*, Antalya, Turkey, Sept. 2005.

[49] Thomas D. Rossing. Acoustics of percussion instruments: Recent progress. *Journal of Acoustical Science and Technology*, 22(3):177–188, 2001.

[50] Thomas D. Rossing and Neville H. Fletcher. Acoustics of a tamtam. *The Bulletin of the Australian Acoustics Society*, 10(1):21–25, Apr. 1982.

[51] Tom Rossing. *Science of Percussion Instruments*, volume 3 of *Series in Popular Science*. World Scientific, Singapore, 2000.

[52] S. Schedin, C. Lambourg, and Antoine Chaigne. Asymmetric non-linear forced vibrations of free-edge circular plates. part ii: Experiments. *Journal of Sound and Vibration*, 221(3):471–490, 1999.

[53] Tamara Smith, Jennifer Hsu, and Ryan Done. Toward a real-time waveguide mesh implementation. In *International Symposium on Musical Acoustics*, pages 54–57, Montreal, Canada, June 2017.

[54] Tamara Smyth and Jennifer Hsu. On phase and pitch in loopback frequency modulation with a time-varying feedback coefficient. In *26th International Congress on Sound and Vibration*, Montreal, Canada, July 2019.

[55] Tamara Smyth and Jennifer Hsu. Representations of self-coupled oscillators with time-varying frequency. In *Proceedings of the Sound and Music Computing Conference*, Málaga, Spain, May 2019.

[56] Greg Surges, Tamara Smyth, and Miller Puckette. Generative feedback networks using time-varying allpass filters. In *Proceedings of the International Computer Music Conference*, pages 78–85, Denton, Texas, 2015.

[57] Norio Tomisawa. Tone production method for an electronic musical instrument. U.S. Patent 4 249 447A, Feb 1998.

[58] Alberto Torin and Michael J. Newton. Nonlinear effects in drum membranes. In *International Symposium on Musical Acoustics*, pages 107–112, Le Mans, France, July 2014.

[59] Lutz Trautmann and Rudolf Rabenstein. *Digital Sound Synthesis by Physical Modeling Using the Functional Transformation Method.* Springer, New York, 1 edition, 2003.

[60] Vesa Välimäki, Jonathan S. Abel, and Julius O. Smith. Spectral delay filters. *Journal of the Audio Engineering Society*, 57(7/8):521–531, July/Aug. 2009.

[61] Kees van den Doel and Dinesh K. Pai. The sounds of physical shapes. *Presence: Teleoperators and Virtual Environments*, 7(4):382–395, August 1998.

[62] Kurt James Werner, Jonathan S. Abel, and Julius O. Smith III. A physically-informed, circuit-bendable, digital model of the Roland TR-808 bass drum circuit. In *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx-14)*, York, UK, Sept. 2014.