UNIVERSITY OF CALIFORNIA, SAN DIEGO

Extensions to Convolution for Generalized Cross-Synthesis

A Thesis submitted in partial satisfaction of the
requirements for the degree
Master of Arts

in

Music

by

Christopher Donahue

Committee in charge:

Miller Puckette, Chair
Tom Erbe
Tamara Smyth

2016

The Thesis of Christopher Donahue is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

Chair

University of California, San Diego

2016

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank the members of my committee for their generous availability and support during not just the process of writing this thesis but throughout my education at UCSD. Their wisdom was a constant source of inspiration especially at times when I felt the most stuck. Additional thanks to countless peers in the UCSD Department of Music graduate program for being a source of respite from a tough workload.

I feel indebted to those who set me on the path towards what I am doing today during my undergraduate education. Specifically, thanks to Dr. Russell Pinkston, Dr. Ethan Frederick Greene and Dr. Eli Fieldsteel at the UT Austin Butler School of Music. I am lucky to have been introduced to them and others at UT through a completely chance encounter.

Finally, I would like to thank my parents Jim and Linda Donahue for their love and guidance. They have always supported me in my decisions and have given me the freedom to do what I enjoy most.

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1053575.

Chapter 3, in part, has been submitted for publication of the material as it may appear in the proceedings of the International Computer Music Conference, 2016, Donahue, Chris; Erbe, Tom; Puckette, Miller. The thesis author was the primary investigator and author of this paper.

ABSTRACT OF THE THESIS

Extensions to Convolution for Generalized Cross-Synthesis

by

Christopher Donahue

Master of Arts in Music

University of California, San Diego, 2016

Miller Puckette, Chair

Cross-synthesis is the process of blending two or more audio signals to produce a hybrid signal with timbral characteristics of the originals. There are a number of methods to perform cross-synthesis of digital audio including but not limited to vocoding, phase vocoding and convolution. Convolution is a common approach to perform cross-synthesis on arbitrary sound files but the process suffers from inflexibility. Having no control over the process itself, a musician interested in convolutional cross-synthesis is left to modify the original sounds to influence the outcome.

There are a number of drawbacks with the process of convolutional cross-

synthesis that impede its musical usefulness. One such drawback is the lack of control over which sound the outcome more closely resembles. Another is a perceived absence of high frequency energy in the outcome. This thesis will introduce novel extensions to the process of discrete convolution for the purposes of offline cross-synthesis that attempt to remedy these concerns.

Discrete convolution and the proposed extensions will be analyzed for their effect on acoustic features. This black-box analysis consists of subjecting a novel, heterogeneous dataset of sounds dubbed RFS1k to convolutional cross-synthesis and examining the average effects on a set of features.

Chapter 1

Introduction

1.1 Introduction

Digital cross-synthesis is a signal processing technique that aims to combine timbral characteristics of input sounds into hybrid outputs. There are a family of techniques that can perform cross-synthesis such as vocoding, phase vocoding and discrete convolution. Discrete convolution (referred to hereafter as *convolution*) is perhaps the most generalized of these approaches; it makes no assumptions about input spectra and treats sounds as *generalized resonators* [Dol85]. This thesis will focus primarily on cross-synthesis via convolution which will be referred to as *convolutional cross-synthesis* (CCS).

CCS is usually employed for black-box cross-synthesis of two arbitrary sound files, such as combining a recording of a hand bell with a recording of a duck's quack. Other methods such as vocoding are more commonly used for cross-synthesis of the human voice with other wide band sources. The goal of this thesis is to present a fully generalized cross-synthesis technique. As such, other techniques for performing

cross-synthesis will be examined mostly for historical purposes.

While a reliable choice for the cross-synthesis of arbitrary sounds, CCS is not without its problems. Performing CCS on any number of sounds always produces exactly one result as the convolution operation is both commutative and associative. A musician interested in employing CCS on a limited number of sounds is faced with a rigid, finite process and forced to modify input sounds to influence the outcome.

Another problem with CCS is that the user has no control over which source sound the outcome more closely resembles. This makes it impossible to synthesize musically-meaningful gestures such as gently fading between two sounds. Additionally, the outcome of CCS is often unpredictable given the sources; it may bear characteristics of the sources but resemble neither [Roa96]. The process essentially interprets the input sounds as resonators and couples them leading to ambiguous results.

A final complaint about CCS concerns the timbral characteristics of the outcome. An arbitrary recording interpreted as a finite impulse response (FIR) filter is likely to have a spectrum with a decreasing amplitude/frequency slope [Ser97]. This informal observation will be quantified further in this thesis. Due to this issue, the outcome of CCS on arbitrary sounds is likely to be lacking in high frequency content as both sounds will attenuate the high frequencies of each other, resulting in a sound which is perceived as "dark."

1.2 Novel Contribution

This thesis introduces a few novel extensions to the process of offline CCS that seek to resolve all of the concerns listed in the introduction. The end result is a parametrized operation allowing musicians to influence the result of cross-synthesis in

a meaningful way.

In order to quantify the issue of high frequency attenuation during cross-synthesis, a large dataset of heterogeneous sound files is required. An additional component of this thesis is the curation of *RFS1k* (random Freesound [FRS13][1] dataset with $1,000$ entries) and its usage in comparing the proposed extensions to ordinary convolution.

1.3 Thesis Organization

This thesis will begin by discussing the history of cross-synthesis with an emphasis on the convolutional approach in Chapter 2. The proposed extensions to CCS will then be built from the ground up for the two-sound case in Chapter 3 with the final formulation appearing in Section 3.4. This formulation will be expanded to the N-sound case in Section 3.5. Drawbacks and artifacts of the algorithm will be discussed in Chapter 4. Chapter 5 will cover the curation strategy and analysis of the heterogeneous RFS1k dataset. The proposed extensions to CCS will then be analyzed objectively in Chapter 6, followed by a few concrete examples with subjective descriptions in Chapter 7. Some final thoughts and conclusions will appear in Chapter 8.

[1] http://freesound.org/

Chapter 2

Background and Related Work

The history of cross-synthesis dates back to early exploration of speech compression for telecommunications. In 1938, Homer Dudley and Bell Labs were granted a patent for the *vocoder*, a strategy of compressing recorded speech using a small filter bank representation allowing for more reliable transmission over telephone lines [Dud39]. While not an example of cross-synthesis itself, this early innovation is the foundation upon which many subsequent discoveries in cross-synthesis were based.

In 1961, Max Matthews and colleagues at Bell Labs studied the spectral envelopes of vowel sounds using digital computers [MMDJ61]. They devised a representation of speech that independently and efficiently encodes the contributions from both the glottal source and the vocal tract. In the mid to late 1970s, further refinements in linear predictive filter estimation techniques yielded the "talking orchestra" effect that would become synonymous with the term vocoder [Pet75, Moo79b]. This effect is perhaps the earliest example of musical cross-synthesis; a broadband signal is subjected to the time-varying spectral envelope of a speech signal yielding a hybrid output.

The phase vocoder [Moo78] has also seen use as a method of cross-synthesis. In particular, the IRCAM Super Phase Vocoder (SVP) offers musicians time-varying, parametrized control over the cross-synthesis of two arbitrary sounds using the formulation listed in Equation 2.1 [Ser97]. Here, $A_n(r,k)$ and $\bar{f}_n(r,k)$ represent the amplitude and instantaneous frequency spectra respectively for short-time Fourier transform (STFT) frame r, bin k of sound n. $E_n(r)$, $F_n(r)$ and $q(r)$ are constant or time-varying parameters which modify the spectral cross-synthesis procedure within the context of the phase vocoder.

$$A(r,k) = E_1(r)\,A_1(r,k) + E_2(r)\,A_2(r,k) + q(r)\,A_1(r,k)\,A_2(r,k) \tag{2.1}$$
$$\bar{f}(r,k) = F_1(r)\,\bar{f}_1(r,k) + F_2(r)\,\bar{f}_2(r,k)$$

This formulation allows musicians to mix the input sounds with a term akin to the windowed convolution of the sounds at a given STFT frame. While producing completely different results, this technique from SVP is remarkably similar in methodology to the novel contributions of this thesis in that it attempts to parametrize cross-synthesis of arbitrary sounds.

Discrete convolution has long been employed to apply the impulse response of real and simulated acoustic environments to arbitrary signals [Moo79a]. This technique, often referred to as *convolution reverb*, couples a signal to a finite impulse response (FIR) filter which captures a reverberant resonator. One could also consider treating any arbitrary sound file as a resonator by interpreting its samples as the coefficients of an FIR filter. This application is immediately apparent to those who

understand the significance of convolution, however it was not explored musically until the early 1980s by Mark Dolson [Dol85] and Richard Boulanger [Bou85].

Dolson and Boulanger referred to the technique of convolutional cross-synthesis as a marriage of *generalized resonators*. Interpreting any arbitrary digital sound as an FIR filter is effectively an attempt to characterize the sound's properties as a resonator. Referring to this phenomenon, Boulanger remarked that "the only difference between Carnegie Hall and the suspended cymbal (besides the seating) is that, acoustically, they are manifestations of two slightly different filters" [Bou85].

In the mid to late 1990s, Curtis Roads investigated and published some thoughts on convolution techniques for the purposes of sound transformation [Roa97]. He referred to the convolution of arbitrary sound files as *cross-filtering* and remarked that "unfortunately, there is no straightforward way to adjust the "balance" of the two sources or to lessen the convolution effect" [Roa97]. Roads suggested mixing in some of the original sounds with the convolved sound to deemphasize the effect.

In more recent history, a primary focus on research in convolution techniques has involved real-time methods. In the mid 1990s, William Gardner published an overlap-partition method for implementing real-time digital convolution with minimal delay (one audio blocksize) [Gar94]. This stood in contrast to previous overlap-partition methods that would incur a latency of at least one block of the partition size [SJ66]. Gardner's real-time convolution strategy, further refined for modern computers by Eric Battenberg in 2011 [BA11], allowed for tradeoffs in time, memory and latency to be configured on a per-application or even per-filter basis. This additional flexibility paved the way for numerous software products employing real-time convolution, particularly convolution reverb plugins. These low-latency, real-time convolution techniques are

considerably more difficult to implement than their high-latency counterparts, often requiring sophisticated scheduling strategies when FIR sizes become large.

The advent of low-latency, real-time convolution methods has also permitted general applications involving the convolution of large FIR filters to run in real-time. In one example from 2007, Tamara Smyth demonstrated an acoustic modeling system that replaced the traditional waveguide with two FIR filters approximating the transmission and reflection characteristics of wind instruments [SA07]. This substitution allowed a synthetic reed model to be coupled to measured FIRs of real instruments. The FIRs could also be parametrized, allowing for real-time control of synthesis with flexibility similar to that of a waveguide model. The technique, dubbed *convolutional synthesis*, could be modified for use in a number of applications pairing synthetic excitation functions with synthetic or measured FIRs yielding flexible, feedback-based convolution systems. This technique could also be considered a form of cross-synthesis using convolution although with significantly different goals than those of this thesis.

Chapter 3

Extended Convolutional Cross-Synthesis

This chapter will individually detail the proposed parameter extensions for the CCS of two sounds, building up to the full form. Convolution (represented by $*$) is the process by which discrete signal f is subjected to a finite impulse response filter g to produce a new signal $f * g$. If f has domain $[0, N)$ and is 0 otherwise and g has domain $[0, M)$, then $f * g$ has domain $[0, N + M - 1)$. Convolution is defined as Equation 3.1. This approach to computing a convolution will be referred to as *direct convolution*.

$$(f * g)[n] = \sum_{m=0}^{M-1} f[n-m]\, g[m] \tag{3.1}$$

The convolution theorem states that the Fourier transform of the result of convolution is equal to the point-wise multiplication of the Fourier transforms of the two sources. Let $\mathcal{F}$ denote the discrete Fourier transform (DFT) operator and $\cdot$

represent point-wise multiplication. An equivalent definition for convolution employing this theorem is stated in Equation 3.2 and is often referred to as *fast convolution*.

$$\begin{aligned} \mathcal{F}(f * g) &= \mathcal{F}(f) \cdot \mathcal{F}(g) \\ &= \|\mathcal{F}(f * g)\| \; e^{\mathrm{i}\angle\mathcal{F}(f*g)} \\ \text{where } \|\mathcal{F}(f * g)\| &= \|\mathcal{F}(f)\| \cdot \|\mathcal{F}(g)\|, \\ \angle\mathcal{F}(f * g) &= \angle\mathcal{F}(f) + \angle\mathcal{F}(g) \end{aligned} \tag{3.2}$$

Given a fast implementation of the DFT, fast convolution uses fewer operations than direct convolution to achieve the same result when filter sizes equal or exceed 64 samples [Dol85]. In the domain of offline cross-synthesis, one is usually operating well above this threshold and thus fast convolution is desirable. In addition to increased efficiency, fast convolution also provides a meaningful interpolation space and is crucial to all of the subsequent extensions proposed by this thesis.

This thesis will first expand on the CCS process for two sounds f and g of lengths N and M respectively. Their convolution $f * g$ has length $N + M - 1$. It is assumed that f and g are zero-padded to a window size greater than or equal to $N + M - 1$ before employing the DFT for fast convolution.

3.1 Extended Convolutional Cross-Synthesis

Given that convolution represents a spectral multiplication as outlined in Equation 3.2, one might attempt to balance the two spectra by various weighting schemata. For example, one could consider weighting the spectra by multiplicative

scalars as in Equation 3.3. The symbol $\hat{*}$ is used to represent convolution operations that are modified versions of the original operation.

$$\mathcal{F}(f \,\hat{*}\, g) = a\,\mathcal{F}(f) \cdot b\,\mathcal{F}(g) \tag{3.3}$$

One might quickly notice that this formulation is inherently flawed. Convolution is associative to scalar multipliers and as such a and b apply to the entire spectra resulting simply in an attenuation of the ordinary convolution. Since convolution represents a spectral multiplication, a natural choice to weight the inputs is to employ the weighted geometric mean as in Equation 3.4.

$$\mathcal{F}(f \,\hat{*}\, g) = (\mathcal{F}(f)^a \cdot \mathcal{F}(g)^b)^{\frac{1}{a+b}} \tag{3.4}$$

Equation 3.4 can be altered without loss of generality to an intuitive "mixing" form by setting $b = (1 - a)$. To not compromise the generality of the equation, the entire product should be raised to the c power, a new parameter. Finally, c is multiplied by a weighting coefficient of 2 so that ordinary convolution is achieved when $a = 1/2, c = 1$. A more "user-friendly" parametrization of weighted geometric mean convolution is presented in Equation 3.5.

$$\mathcal{F}(f \,\hat{*}\, g) = (\mathcal{F}(f)^a \cdot \mathcal{F}(g)^{(1-a)})^{2\,c} \tag{3.5}$$

When expanded, Equation 3.5 equates to the weighted geometric mean of the magnitude spectra and the weighted arithmetic mean of the phase spectra. This expansion is represented in Equation 3.6.

$$\mathcal{F}(f \hat{*} g) = \|\mathcal{F}(f \hat{*} g)\| \; e^{\mathrm{i}\angle\mathcal{F}(f \hat{*} g)} \tag{3.6}$$

$$\text{where } \|\mathcal{F}(f * g)\| = (\|\mathcal{F}(f)\|^{a} \cdot \|\mathcal{F}(g)\|^{(1-a)})^{2c},$$

$$\angle\mathcal{F}(f * g) = 2\,c\,(a\,\angle\mathcal{F}(f) + (1-a)\,\angle\mathcal{F}(g))$$

All of the subsequently proposed extensions to convolution will fit within the context of Equation 3.6. The extensions will be introduced as solutions to the high-level shortcomings of CCS as outlined in the introduction.

3.2 Brightness Control

Convolution of arbitrary sounds has a tendency to exaggerate low frequencies and understate high frequencies. One way to interpret the cause of this phenomenon is that the magnitude spectra of the two sounds constructively and destructively interfere with each other when multiplied during convolution. The interference of low-frequency peaks in natural sounds is likely to yield much higher resultant amplitudes than the interference of high-frequency peaks.

To resolve this issue, the geometric mean can be employed when combining the magnitude spectra of two sounds. The geometric mean mitigates both the constructive and destructive effects of interference resulting in a more flattened spectrum. The form of the convolved magnitude spectrum from Equation 3.2 is altered in Equation 3.7.

$$\|\mathcal{F}(f \hat{*} g)\| = \sqrt{\|\mathcal{F}(f)\| \cdot \|\mathcal{F}(g)\|} \tag{3.7}$$

More generally, a parameter q is introduced that controls the flatness of the hybrid magnitude spectrum in Equation 3.8.

$$\|\mathcal{F}(f \hat{*} g)\| = (\|\mathcal{F}(f)\| \cdot \|\mathcal{F}(g)\|)^q \tag{3.8}$$

Note that this formulation collapses to ordinary convolution as defined in Equation 3.2 when $q = 1$ and geometric mean convolution as defined in Equation 3.7 when $q = 1/2$. As q decreases towards 0, the magnitude spectrum flattens resulting in noise-like sounds. This effect is demonstrated in Figure 3.1. As q increases past 1, constructive interference between the frequency spectra of f and g is further emphasized, eventually resulting in tone-like sounds.

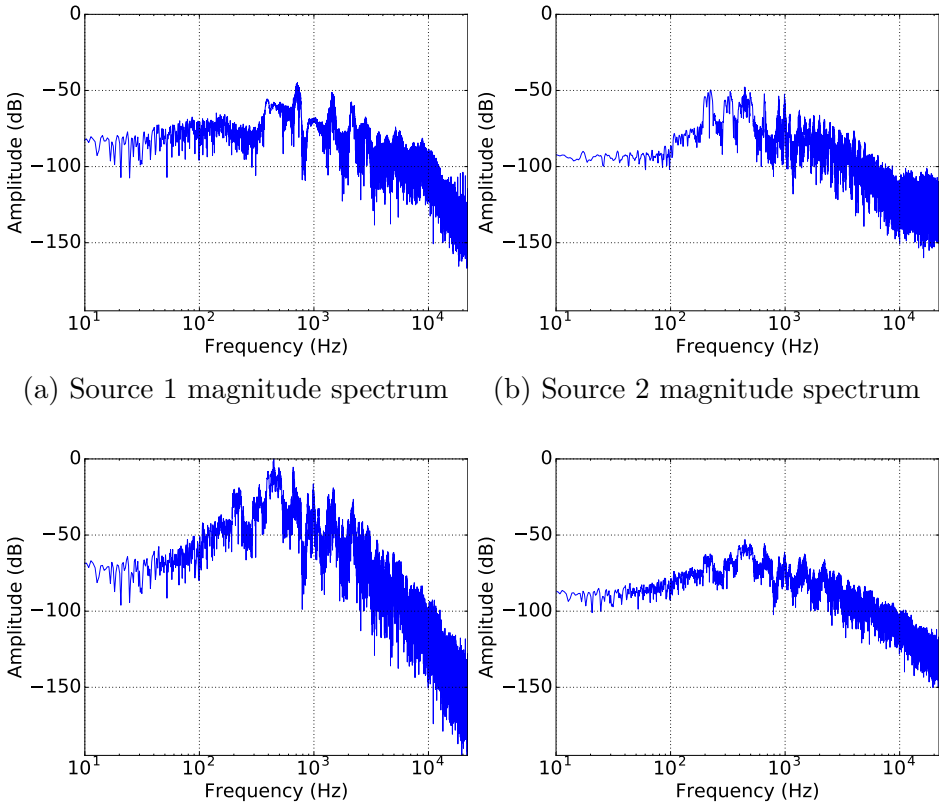
3.3 Source Emphasis

The inability to "skew" the influence from input sounds more towards one or another represents a large problem with using CCS as a musical process. The separation of sources into magnitude and phase spectra via fast convolution conveniently allows for the modification of the amount of influence each source has over the result.

3.3.1 Skewed Magnitude

Equation 3.8 is extended to Equation 3.9, adding a parameter p which allows the influence of source magnitude spectra $\|\mathcal{F}(f)\|$ and $\|\mathcal{F}(g)\|$ to be skewed in the outcome $\|\mathcal{F}(f \hat{*} g)\|$.

$$\|\mathcal{F}(f \hat{*} g)\| = (\|\mathcal{F}(f)\|^p \cdot \|\mathcal{F}(g)\|^{(1-p)})^{2q} \tag{3.9}$$



(a) Source 1 magnitude spectrum

(b) Source 2 magnitude spectrum

(c) Ordinary convolution magnitude spectrum ($q = 1$)

(d) Geometric mean convolution magnitude spectrum ($q = 1/2$)

Figure 3.1: Example of cross-synthesis of two sounds (a and b) using ordinary convolution (c) and geometric mean convolution (d)

With this form $p = 1$ fully emphasizes $\|\mathcal{F}(f)\|$, $p = 0$ fully emphasizes $\|\mathcal{F}(g)\|$, and $p = 1/2$ emphasizes neither. As p skews further towards 0 or 1, one of the source's magnitude spectrum is increasingly flattened and the result becomes akin to vocoding. The q parameter is multiplied by the coefficient 2 to maintain the same scale as in Equation 3.9 when $p = 1/2$.

Skewed Phase

A similar extension is be made for the phase of the outcome in Equation 3.10, adding a parameter r which allows the influence of source phase spectra $\angle\mathcal{F}(f)$ and

$\angle\mathcal{F}(g)$ to be skewed in the outcome $\angle\mathcal{F}(f \hat{*} g)$.

$$\angle\mathcal{F}(f \hat{*} g) = 2\,(r\,\angle\mathcal{F}(f) + (1-r)\,\angle\mathcal{F}(g)) \tag{3.10}$$

With this form $r = 1$ fully emphasizes $\angle\mathcal{F}(f)$, $r = 0$ fully emphasizes $\angle\mathcal{F}(g)$, and $r = 1/2$ emphasizes neither. As with q, s is multiplied by the coefficient 2 to maintain the analogy of parameter r to parameter p.

With the addition of parameter r, the original input sounds can be recovered in the extended convolution parameter space ($p = r = [0, 1], q = 1/2$).

3.3.2 Phase Scattering

One final extension to convolution for the purpose of cross-synthesis is suggested that does not directly address the two core issues of brightness and source influence. Analogous to the parameter q for manipulating the hybrid magnitude spectra, the parameter s is added to the definition of hybrid phase spectra in Equation 3.11.

$$\angle\mathcal{F}(f \hat{*} g) = 2\,s\,(r\,\angle\mathcal{F}(f) + (1-r)\,\angle\mathcal{F}(g)) \tag{3.11}$$

As s decreases towards 0, source phase is nullified resulting in significant amounts of time domain cancellation yielding impulse-like outcomes. As s increases past 1, source phase is increasingly scattered around the unit circle eventually converging to a uniform distribution. Randomizing phase in this manner is similar to the additive phase noise of [Erb11] and produces ambient-sounding results with little variation in time.

With the addition of parameter s, the autocorrelation of either input sound

can be obtained in the extended convolution parameter space ($p = r = [0, 1], q = 1, s = 0$).

3.4 Full Form for Two Sounds

The amalgamation of the extensions made in Section 3.2 and Section 3.3 is presented in Equation 3.12. This equation combines all of the proposed extensions to convolution for the purposes of cross-synthesis and will be henceforth be referred to as *extended convolutional cross-synthesis* (ECCS).

$$\mathcal{F}(f \hat{*} g) = \|\mathcal{F}(f \hat{*} g)\| \; e^{\mathrm{i} \angle \mathcal{F}(f \hat{*} g)} \tag{3.12}$$

$$\text{where } \|\mathcal{F}(f \hat{*} g)\| = (\|\mathcal{F}(f)\|^p \cdot \|\mathcal{F}(g)\|^{1-p})^{2q},$$

$$\angle \mathcal{F}(f \hat{*} g) = 2s\,(r \angle \mathcal{F}(f) + (1 - r) \angle \mathcal{F}(g))$$

Performing ECCS is relatively straightforward given a Fast Fourier transform (FFT) implementation and two sounds f and g with lengths N and M respectively.

1. Zero pad each sound to length $N + M - 1$
2. Perform FFT of the padded f and g to get $\mathcal{F}(f)$ and $\mathcal{F}(g)$
3. Combine magnitude and phase spectra according to provided p, q, r, s parameters and Equation 3.12 yielding $\mathcal{F}(f \hat{*} g)$
4. Use inverse FFT to recover $f \hat{*} g$
5. *(Optional)* Normalize to peak amplitude 1 for audition

If provided the common radix-2, decimation-in-time implementation of the FFT, one would zero pad f and g to the smallest power of two that is greater than or equal to $N + M - 1$ for maximum efficiency. This works fine for CCS as one can simply truncate the output to length $N + M - 1$. For ECCS however, this will arbitrarily lengthen the output; a further discussion of this phenomenon is presented in Chapter 4.

3.5 Generalization for N Sounds

Now that ECCS has been established within the context of two sounds, it can be easily generalized to any number of sounds. Given a set $\{H\}$ consisting of N sounds, one can find their extended convolution $h_1 \hat{*} h_2 \hat{*} \ldots \hat{*} h_N = \hat{*}\{H\}$ using Equation 3.13.

$$\mathcal{F}(\hat{*}\{H\}) = \|\mathcal{F}(\hat{*}\{H\})\| \; e^{\mathrm{i}\angle\mathcal{F}(\hat{*}\{H\})} \tag{3.13}$$

$$\text{where } \|\mathcal{F}(\hat{*}\{H\})\| = \Big(\prod_{i=1}^{N} \|\mathcal{F}(h_i)\|^{p_i}\Big)^{\frac{N\,q}{\sum_{i=1}^{N} p_i}}$$

$$\angle\mathcal{F}(\hat{*}\{H\}) = \frac{N\,s}{\sum_{i=1}^{N} r_i}\Big(\sum_{i=1}^{N} r_i\,\angle\mathcal{F}(h_i)\Big)$$

All previous equations have been examining the case where $N = 2$ so one parameter was used to determine the skew mix for both sounds. When $N > 2$, this simplifying assumption can no longer be made and all p and r values are normalized to maintain the same meaning for q and s.

If each sound h_n has length L_n, then their convolution will have length $L_{\{H\}} =$

$(\sum_{i=1}^{N} L_i) - N + 1$. In an offline cross-synthesis environment, one must window each input sound to at least $L_{\{H\}}$ before performing the DFT.

Chapter 3, in part, has been submitted for publication of the material as it may appear in the proceedings of the International Computer Music Conference, 2016, Donahue, Chris; Erbe, Tom; Puckette, Miller. The thesis author was the primary investigator and author of this paper.

Chapter 4

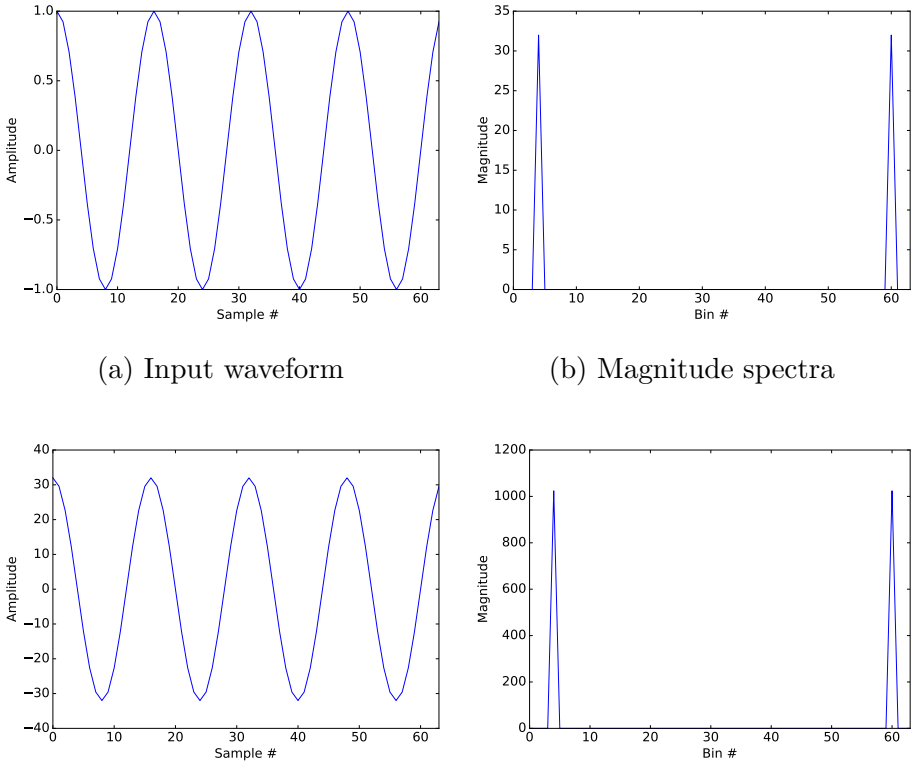
Artifacts

This chapter discusses time-domain artifacts that arise through the use of ECCS. The proposed extensions to convolution modify input spectra nonlinearly. These nonlinear modifications are inherent to the desirable properties of the ECCS algorithm (averaging/flattening spectra) but give rise to certain issues in the time domain of the result. The cross-synthesis aspect of the procedure is not itself responsible for the artifacts; they can arise using the techniques with only one sound. For simplicity, this chapter will examine these artifacts from the perspective of transformations applied to single waveforms. This simpler picture helps to give broader understanding of the type of artifacts that can be expected from ECCS results.

The artifacts presented in this chapter behave in interesting ways from a musical perspective. Fortunately, they are always circular and there are no discontinuities at the end points. This allows the result of ECCS to be looped to create interesting textures. If a looping sound is not desired, the musician may wish to alter the result to begin at a zero crossing near a perceptual onset.

The nonlinear spectral modifications of ECCS behave as well as the provided

input spectra. As a simple example, one might examine the result of squaring the magnitude spectra of an input and preserving the phase spectra. This is equivalent to performing ECCS Equation 3.13 with $N = 1$ and $q = 2$. The waveform and spectra for the input (four cycles of a sinusoid sampled at 64 points) and the transformed result are displayed in Figure 4.1. In this highly contrived example, the input spectra is as well behaved as possible and squaring the magnitude spectra produces a reasonable result: the same waveform with a higher gain.



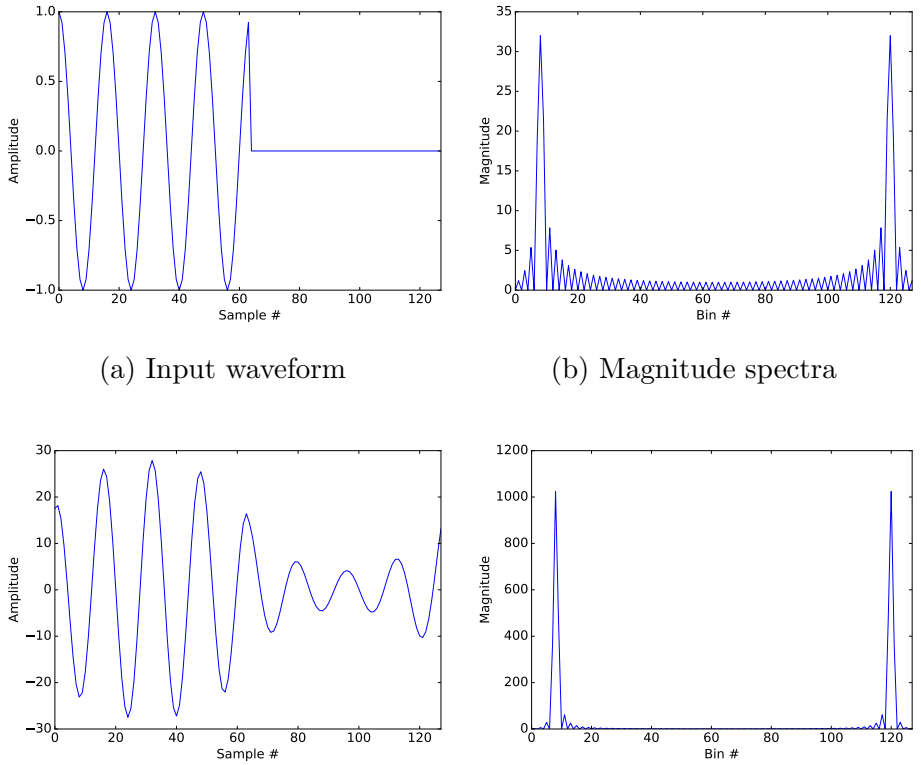
(a) Input waveform (b) Magnitude spectra

(c) Resultant waveform (d) Input magnitude spectra squared

Figure 4.1: A well-behaved sinusoidal input undergoes ECCS with $q = 2$

When performing CCS via fast convolution, one must zero-pad an input waveform to at least the sum of the inputs less one. To the agnostic discrete Fourier transform, this zero-padding operation looks equivalent to a cyclical function that has simply been multiplied by a rectangular window. A multiplication in the time

domain is equivalent to a convolution in the frequency domain through the convolution theorem. The same 64-sample sinusoid from Figure 4.1 has been zero-padded to a length of 128 in Figure 4.2 before ECCS. As expected, the input magnitude spectra appears to be a convolution of the true spectra with the magnitude spectra of the imposed rectangular window (a sinc function). This has a profound effect on the resultant waveform after subjecting the waveform to ECCS with $q = 2$; it is not the zero-padded sinusoid with a higher gain that one might expect to see.



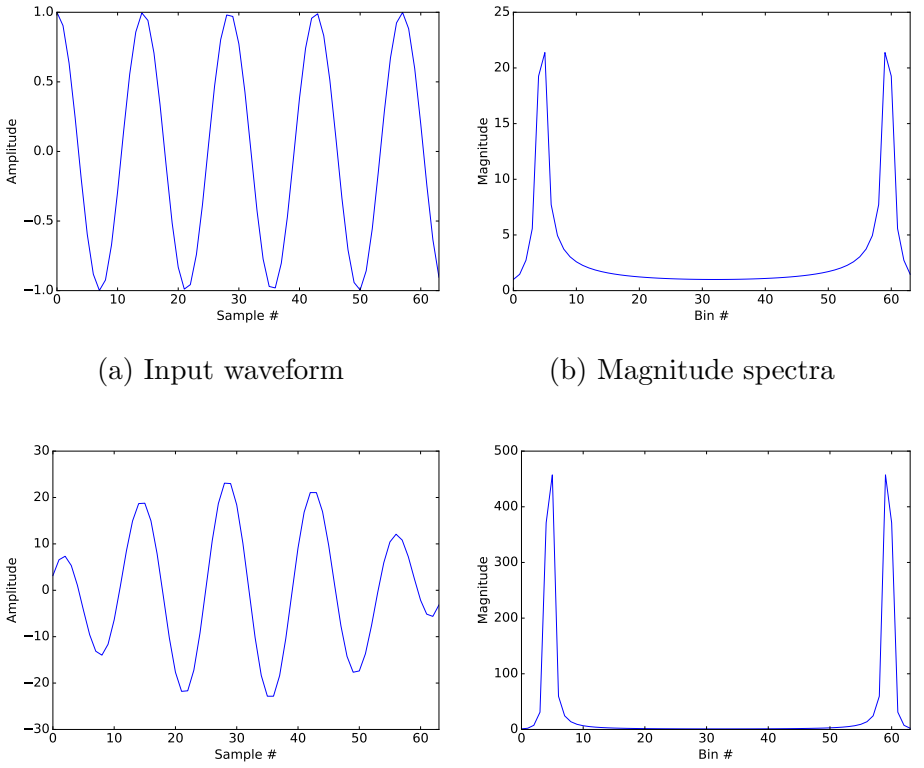
(a) Input waveform (b) Magnitude spectra

(c) Resultant waveform (d) Input magnitude spectra squared

Figure 4.2: A rectangular-windowed sinusoidal input undergoes ECCS with $q = 2$

If one were to continue to increase the value of q with the same rectangular-windowed input, the relative amplitudes of the sinc function sidelobes would continue to decrease compared to the peak at bin 8 eventually yielding a pure sinusoid.

Another issue that can produce ill-behaved spectra is spectral leakage. Figure 4.3 shows a 4.5 cycle sinusoid subjected to ECCS with $q = 2$. Here the result of squaring the spectra yields a modulation in the time-domain and a phase shift. As q is increased, the waveform will eventually converge to five cycles of a sinusoid with a $-\pi/2$ phase shift from the original.



(a) Input waveform (b) Magnitude spectra

(c) Resultant waveform (d) Input magnitude spectra squared

Figure 4.3: A poorly-behaved sinusoidal input undergoes ECCS with $q = 2$

From these figures, it is clear that the majority of the artifacts from applying nonlinear spectral transformations arise from the window functions inherent to analysis of signals via the DFT. It is therefore easier to observe the time-domain warping caused by the proposed nonlinear transformations by performing them on the spectra of the window function itself. In Figure 4.4, a 128-sample rectangular window is subjected to a variety of transformations to its magnitude and phase spectra using

$NFFT = 1024$. The values of q and s correspond to Equation 3.13 for $N = 1$; q is the power that the magnitude spectrum is raised to and s is a multiplier of the phase spectrum. These figures paint an accurate portrayal of an "artifact" envelope that a given sound will be subjected to with the specified q and s parameters. The integral of each row is the same; the s parameter does not affect the total area underneath the window given a particular q value.

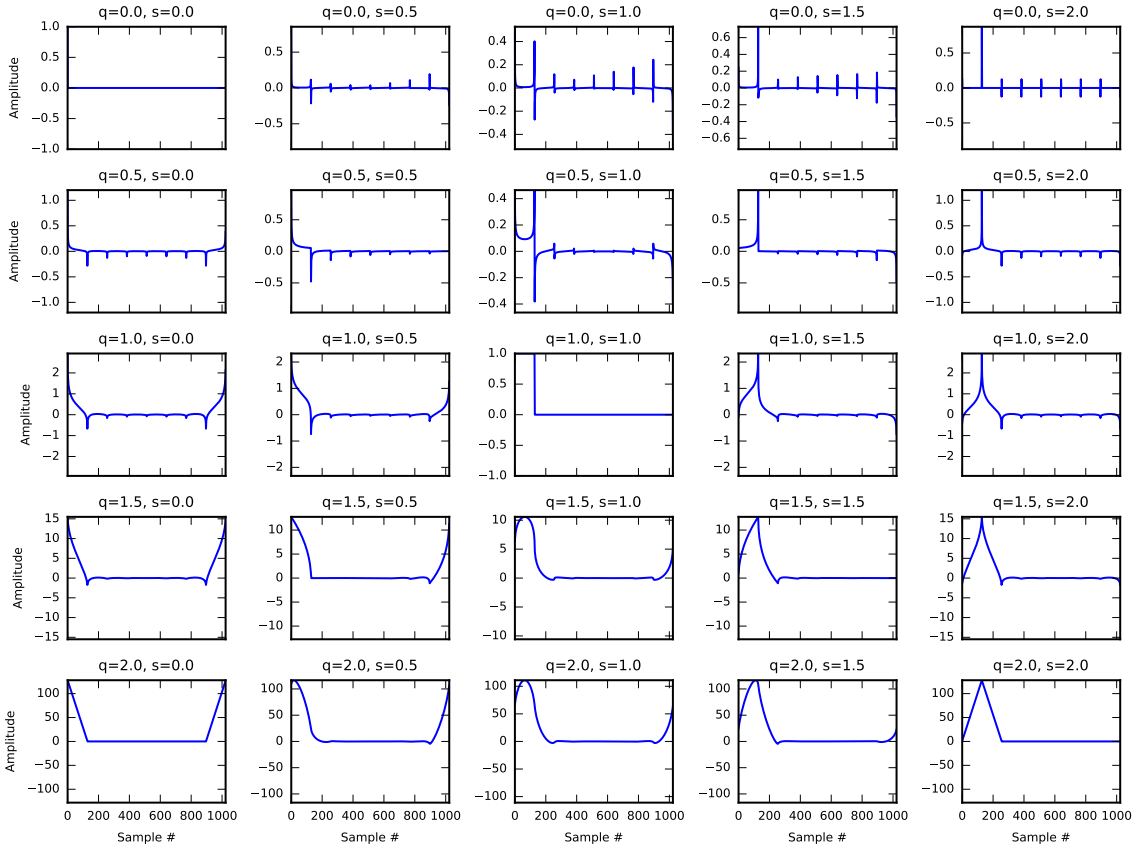


Figure 4.4: Subjecting a rectangular window to nonlinear transformations of its magnitude (q) and phase spectra (s)

For the purposes of offline cross-synthesis, the effects of the artifacts caused by the rectangular window can be mitigated by using a DFT size equal to the sum of the input lengths less one. If this length has a small number of prime factors (or is itself prime), then the efficiency of most FFT algorithms will be compromised.

In this case, one can use Bluestein's algorithm [Blu70] for computing the Chirp Z-transform [RSR69] to achieve the desired $O(n\,log(n))$ efficiency for arbitrary DFT lengths. This is a crucial detail for employing the DFT for long window sizes that are not powers of two. If one is not as concerned with the introduction of arbitrary circular artifacts, one can continue using sufficiently large powers of two as FFT sizes. With extravagant parameter configurations for ECCS (especially high s values), the FFT size can be reinterpreted as a parameter affecting the length of the output.

Chapter 5

RFS1k

A key goal of this thesis is to not just introduce extensions to convolution but also to empirically compare the extensions to ordinary convolution for the purposes of cross-synthesis. A heterogeneous dataset of sounds without bias is desirable for performing this black-box analysis. No dataset was found that met these criteria so one was instead curated from scratch. The Freesound [FRS13] [1] website was used to gather sound material for cross-synthesis experiments. Another goal of this dataset was for it to be useful to a variety of projects and to publish it for wider consumption by the music information retrieval/audio content analysis community.

5.1 Curation

Freesound is a service where anyone can upload sound material of all types under permissive Creative Commons licenses. This service was ideal for the curation of a heterogeneous dataset because of the service's wide variety of content

[1] http://freesound.org/

types, recording/encoding chains and excellent API. There were $345,136$ sounds with sequentially-ordered IDs on Freesound as of Saturday 21^{st} May, 2016.

Freesound as a search engine is meant to be queried by free-text. This approach is too biased for the purposes of heterogeneity. Luckily, the sounds on Freesound are all associated with a sequential integer ID and the API allows users to retrieve sounds by this ID. Sound material was acquired by querying Freesound for randomly-generated integer IDs and a few additional criteria.

The strategy for acquiring sounds was to sample one integer at a time without replacement from a set of integers on the interval $[0, 345135]$, query Freesound for associated metadata and repeat until 1000 sounds were acquired. A sound was downloaded and added to the dataset if it met the following criteria:

- Existed still; not deleted by uploader or administrator
- Uploaded by a user not already represented in RFS1k
- Was encoded as a $44.1kHz/16bit$ PCM WAV file
- Was either a mono or stereo recording
- Duration was in the range of $[50ms, 5s]$
- Was not licensed under the Creative Commons Sampling Plus 1.0 license [2]

Unique user representation in the dataset ensured that source material, recording chains, encoding chains, etc. were as varied as possible. This method of collection started off with a high rate of success per query which slowed over time due to a few

[2] http://creativecommons.org/licenses/sampling+/1.0/legalcode

Freesound users being significantly overrepresented in the uploaded content on the website.

The lower limit of $50ms$ was chosen to ensure that sounds contained at least two 1024-sample windows for spectral analysis at $44.1kHz$. The upper limit of $5s$ was chosen to limit the overall size of the dataset and the CPU time required for multiple convolutions. Sounds licensed under Creative Commons Sampling Plus 1.0 were excluded as it has been deprecated by Freesound.

5.2 Analysis

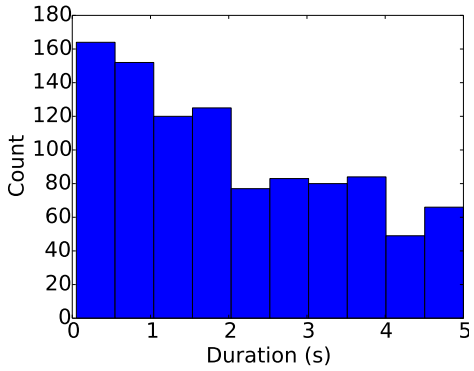
The curation script was run on Saturday 21st May, 2016. The dataset (containing 1,000 items) was named the Random Freesound dataset or *RFS1k* for short. A few high-level features of RFS1k will be analyzed in this section.

5.2.1 Metadata

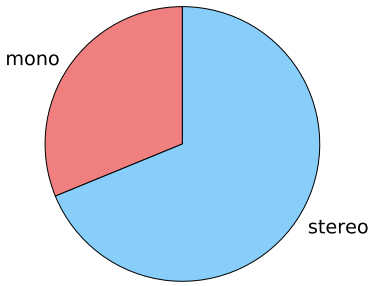
The average length of a sound in RFS1k is 2.07 seconds. In Figure 5.1a, a histogram is displayed illustrating the distribution of sound lengths in RFS1k with $500ms$ bins. Sounds on Freesound are frequently tight croppings of quick sound events, hence the trend seen in the histogram towards shorter sounds.

Every sound in RFS1k is a $44.1kHz/16bit$ PCM WAV file. Most of the sounds (68.8%) in RFS1k are stereo as shown in Figure 5.1b. The distribution of licenses for the sound files is shown in Figure 5.1c.

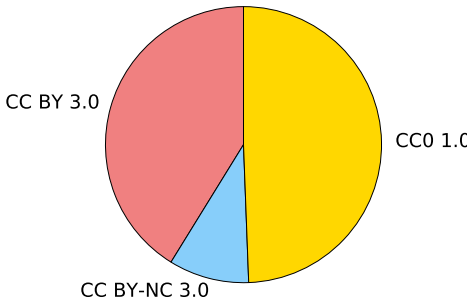
The upload year distribution of sound files in RFS1k corresponds to increasing usage of the Freesound service since its inception as well as decreasing prevelance of



(a) Durations in RFS1k



(b) Number of channels in RFS1k

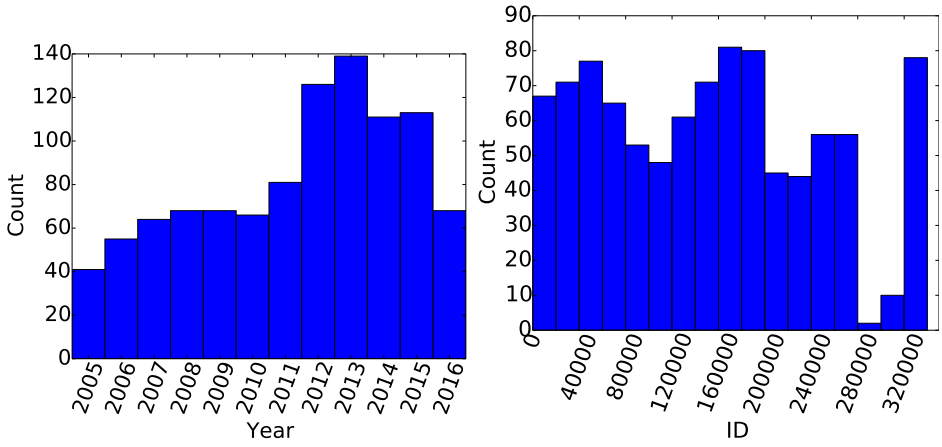


(c) Licenses in RFS1k

Figure 5.1: Metadata distributions for RFS1k

the Sampling Plus license. The number of sound files in the dataset by upload year is illustrated in Figure 5.2a.

In Figure 5.2b, a histogram of sound IDs in RFS1k separated into 16 equal length ranges is shown. While one may expect this chart to be fairly flat based on the curation criteria, there are additional factors to consider. Certain users are extremely overrepresented on Freesound.org. For example, the most active user on the website, *modularsamples*, has uploaded more than 10% of the total sound files on the service. Most of these sounds fall into one bin of sound IDs and as such that has very few members.



(a) Upload year representation in RFS1k

(b) Sound ID representation in RFS1k

Figure 5.2: Chronological histograms for RFS1k

5.2.2 Tags

Freesound allows users to upload free-text metadata associated with their sound files. For each sound file the name, description and tags are included among other information. Sound file names average 17.56 characters in length while descriptions average around 117.20 characters. Sound files have an average of 6.27 tags, though individual tags are user-determined and sparsely represented. The top 20 tags of the sound files in RFS1k are listed in Table 5.1.

Table 5.1: 20 most popular tags in RFS1k

Tag	Count	Percentage
drum	81	8.10%
percussion	62	6.20%
voice	55	5.50%
bass	55	5.50%
loop	54	5.40%
hit	50	5.00%
noise	48	4.80%
metal	47	4.70%
game	43	4.30%
field-recording	42	4.20%
synth	42	4.20%
kick	42	4.20%
electronic	39	3.90%
sound	36	3.60%
fx	31	3.10%
drums	30	3.00%
beat	30	3.00%
vocal	27	2.70%
guitar	26	2.60%
door	25	2.50%

Chapter 6

Analysis of Extended Convolutional Cross-Synthesis

This chapter details a black-box analysis of convolutional cross-synthesis using ordinary convolution and the proposed extensions. The RFS1k dataset gathered as outlined in Chapter 5 was used as audio material for the analysis.

6.1 Introduction

From the 1000 sounds in the test set, 500 random pairings were generated without replacement. Each of these pairs was subjected to CCS and ECCS with five different parameter configurations resulting in six sets of 500 hybrid sounds. Using the Essentia software package [BWG+13], feature extraction was performed on both the original sounds and the hybrid sets to analyze what changes convolution yields to acoustic features on average.

6.2 Acoustic Features

The following acoustic features were identified as useful for general analysis: *loudness*, *spectral flux*, *spectral centroid*, *spectral complexity* and *spectral flatness*. Each of these features was computed for the original sounds as well as the six hybrid sets listed below. All spectral features were computed using windows of size 1024 with 50% overlap and Hann windowing. Features were averaged across all windows per sound then across all sounds per set. In other words, the "Max." column of each table represents the mean across all tracks of the max across all windows in each track.

Note that these acoustic feature numbers will differ slightly from those listed in [DEP16]. The dataset from that earlier paper was later regathered as RFS1k in order to fully acquire the associated metadata with each sound, and thus the dataset contains different sounds (though the averages are almost identical). This was done for the purpose of broader dataset publication and to make sure sounds had acceptable licenses.

1. *RFS*: 1000 RFS1k sounds
2. *OC*: 500 RFS pairs subjected to Ordinary Convolution ($[p, q, r, s] = [1/2, 1, 1/2, 1]$)
3. *GMC*: 500 RFS pairs subjected to Geometric Mean Convolution ($[p, q, r, s] = [1/2, 1/2, 1/2, 1/2]$)
4. *GMMC*: 500 RFS pairs subjected to Geometric Mean Magnitude Convolution ($[p, q, r, s] = [1/2, 1/2, 1/2, 1]$)
5. *HPC*: 500 RFS pairs subjected to Half Phase Convolution ($[p, q, r, s] = [1/2, 1, 1/2, 1/2]$)

6. *SMC*: 500 RFS pairs subjected to Skewed Magnitude Convolution ($[p, q, r, s] = [1, 1, 1/2, 1]$)

7. *SPC*: 500 RFS pairs subject to Skewed Phase Convolution ($[p, q, r, s] = [1/2, 1, 1, 1]$)

6.2.1 Loudness

Raw gain in peak amplitude created by convolving two sources is difficult to predict. For the realm of offline cross-synthesis the issue can be ignored somewhat. Instead, perceptual loudness metrics can be examined assuming that all sounds have been scaled to the same peak amplitude.

Loudness, defined by Steven's power law as energy raised to the power of 0.67 [Ste75], is a psychoacoustic measure representing the perceived intensity of a signal. Loudness of two signals with the same peak amplitude can differ significantly. Loudness values for RFS and all of the hybrid sets appear in Table 6.1.

Table 6.1: Windowed loudness values for all sound sets

Set	Mean Rel.	Mean	Std. Dev.	Min.	Max.
RFS	1.0000	7.6333	7.8888	0.3095	34.152
OC	0.9865	7.5306	8.4834	0.0067	41.703
GMC	0.0563	0.4299	0.3977	0.1347	3.4691
GMMC	0.7533	5.7503	5.3752	0.3390	28.092
HPC	0.3001	2.2905	1.9530	0.4358	13.476
SMC	1.2633	9.6432	9.1378	0.7742	45.805
SPC	0.8406	6.4163	6.6037	0.6629	40.045

The mean loudness of all hybrid sets is skewed by the exaggerated tail created by convolution. It is more telling to examine the max loudness. OC produces higher average max loudness than RFS, while GMC, GMMC and HPC produce lower max loudness. GMC, GMMC and HPC have an averaging effect on the amplitude envelope

of the result which causes this reduction (as is indicated by their lower standard deviation). SMC and SPC both produce an increase in max loudness compared to RFS that is similar in magnitude to the increase produced by OC.

6.2.2 Spectral Flux

Spectral flux is any metric that describes the propensity of a signal's magnitude spectrum to fluctuate over time [TC99]. In this case, it is defined as the L2-norm of the change in magnitude spectrum from window to window. A higher value indicates more average fluctuation in the signal. Spectral flux is used to demonstrate that certain parameter combinations of ECCS are more likely to yield fluctuating timbres than others. Spectral flux values for RFS and all of the hybrid sets appear in Table 6.2.

Table 6.2: Spectral flux values for all sound sets

Set	Mean Rel.	Mean	Std. Dev.	Min.	Max.
RFS	1.0000	0.1084	0.1168	0.0082	0.6058
OC	0.8416	0.0912	0.0959	0.0004	0.5465
GMC	0.1588	0.0172	0.0165	0.0060	0.1813
GMMC	0.9448	0.1024	0.0835	0.0157	0.4800
HPC	0.3936	0.0427	0.0386	0.0095	0.3847
SMC	0.9189	0.0996	0.0844	0.0135	0.5236
SPC	0.7732	0.0838	0.0747	0.0147	0.5220

None of the parameter combinations of ECCS produce results that fluctuate more than RFS. Convolution tends to have an averaging effect on timbre as sounds get smeared in time. GMC produces by far the lowest spectral flux suggesting that this averaging effect is most pronounced when it applies to both the magnitude and phase spectrum. Given that GMC is a combination of the parameter changes provided by

GMMC and HPC, it is clear that "averaging" the phase produces a drastic reduction in spectral flux as HPC flux is also significantly lower than that of RFS.

6.2.3 Spectral Centroid

The spectral centroid is the barycenter of the magnitude spectrum using normalized amplitude [Pee04]. Listed in Table 6.3 in Hz, the spectral centroid represents a good approximation of the "brightness" of a sound. The higher the value, the brighter the perceived sound. The spectral centroid is used to quantify the informal observation of high frequency attenuation produced by CCS.

Table 6.3: Spectral centroid values for all sound sets

Set	Mean Rel.	Mean	Std. Dev.	Min.	Max.
RFS	1.0000	3333.3	1467.7	1212.0	7634.0
OC	0.3619	1206.4	677.18	341.14	4480.5
GMC	1.0791	3596.9	530.25	2530.9	7375.7
GMMC	1.0771	3590.2	745.49	2049.8	6083.7
HPC	0.3620	1206.6	433.52	803.56	5455.0
SMC	0.3146	1048.8	351.16	623.38	3842.9
SPC	0.3469	1156.2	343.68	677.39	3768.9

The average spectral centroid of the outcome of OC is approximately 64% lower than that of RFS. This confirms the previously-stated observation that the output of convolution is often perceptually darker than the inputs. GMMC brings the average spectral centroid to a similar level of the original sounds. Both SMC and SPC have a similar effect on the perceived brightness compared to OC, indicating that the source influence parameters (p, r) are relatively independent from the parameter controlling brightness (q).

6.2.4 Spectral Flatness

Spectral flatness is a measure of the noisiness of a signal and is defined as the ratio of the arithmetic mean to the geometric mean of spectral amplitudes [Pee04]. The measure approaches 1 for noisy signals and 0 for tonal signals. Spectral flatness values for all sets appear in Table 6.4. This measure is used to demonstrate the informal observation that ordinary convolution overemphasizes constructive spectral interference yielding results that are less flat than the inputs.

Table 6.4: Spectral flatness values for all sound sets

Set	Mean Rel.	Mean	Std. Dev.	Min.	Max.
RFS	1.0000	0.2158	0.1132	0.0628	0.5778
OC	0.0960	0.0207	0.0299	0.0015	0.2761
GMC	1.2637	0.2727	0.0626	0.1715	0.7971
GMMC	1.2276	0.2649	0.0670	0.1289	0.5192
HPC	0.0887	0.0191	0.0550	0.0040	0.6240
SMC	0.0338	0.0073	0.0320	0.0005	0.4047
SPC	0.0754	0.0163	0.0310	0.0026	0.3972

The difference between the spectral flatness for OC and GMMC is pronounced; the average spectra for the product of GMMC is roughly 13 times flatter than that of OC. This trend continues roughly exponentially as demonstrated in Figure 6.1. SMC further emphasizes constructive interference as it is a self-multiplication and produces even less flat results than OC. HPC and SPC have a less notable effect on spectral flatness when compared to OC as the measure does not consider phase.

6.2.5 Spectral entropy

Spectral entropy [MIBH04] computes the Shannon entropy of the magnitude spectrum. Entropy is often used in information theory to get a naive estimate of the

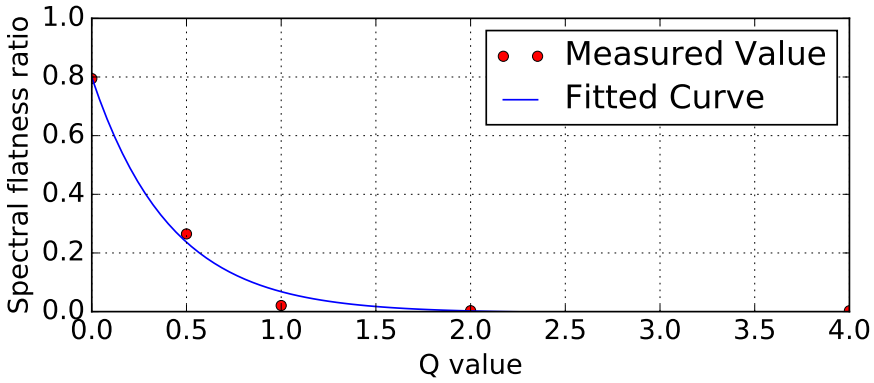


Figure 6.1: Analysis of spectral flatness of ECCS on RFS using increasing values of q

"compressability" of a given set of data. The higher the entropy, the less compressible it is and the more inherent information it contains.

Table 6.5: Spectral entropy values for all sound sets

Set	Mean Rel.	Mean	Std. Dev.	Min.	Max.
RFS	1.0000	5.8741	1.0361	3.5831	8.0125
OC	0.7753	4.5539	0.8281	2.6706	7.0413
GMC	1.2015	7.0579	0.2884	6.2037	8.5819
GMMC	1.1747	6.9001	0.4345	5.5805	7.9696
HPC	0.8273	4.8595	0.4724	3.9351	8.1126
SMC	0.7099	4.1698	0.5056	3.1776	7.2237
SPC	0.8108	4.7625	0.5166	3.5843	7.2501

Surprisingly, most of the hybrid sets have lower spectral entropy than the input set. OC reduces spectral energy by more than 20% when compared to RFS, suggesting that a number of peaks in the input spectrum are not preserved through the process. GMC and GMMC both have higher spectral entropy than RFS, implying that the geometric mean averaging process does a better job of preserving spectral information than OC. HPC and SPC mitigate spectral entropy to a level similar to OC.

Chapter 7

Examples of Extended Convolutional Cross-Synthesis

This chapter examines a few concrete examples of ECCS. Since a generalized analysis of ECCS was already covered in Chapter 6, this section mostly focuses on subjective descriptions of the results of ECCS with particular sound/parameter combinations.

7.0.1 Violin and Suspended Cymbal

This section explores the cross-synthesis of a recording of a violin excerpt[1] with a recording of a ride cymbal struck by a stick[2]. This is a fairly standard cross-synthesis operation that one might want to try to perform: coupling the melody from the violin to a cymbal resonator. The original waveforms and their standard convolution ($[p, q, r, s] = [0.5, 1.0, 0.5, 1.0]$) are displayed in Figure 7.1.

[1] https://www.freesound.org/people/FreqMan/sounds/25481/
[2] https://www.freesound.org/people/Veiler/sounds/209900/

The result of these two sounds subjected to ordinary CCS sounds convincingly like a violin causing a cymbal to vibrate sympathetically. It lacks the characteristic high frequencies that give the violin its shimmering timbre however. The standard approach with ECCS to resolve this issue is to compute the hybrid magnitude spectrum with the geometric mean using $[p, q, r, s] = [0.5, 0.5, 0.5, 1.0]$. The result, shown in Figure 7.2 has restored the characteristic shimmer of the violin while maintaining the "convolved" sound of the ordinary convolution. The effect is a little too prominent however as the result starts to sound noisy; the geometric mean has effectively raised the noise floor of both of the inputs. A more satisfying result can be obtained with $[p, q, r, s] = [0.5, 0.75, 0.5, 1.0]$. Reducing the intensity of the averaging from 0.5 to 0.75 also mitigates some of the undesirable circular artifacts that occur as a result of nonlinear spectral modification.

If q is returned to a value of 0.5 and s is set to this same value, the result of ECCS is the true mean of the input spectra (which equates to the geometric and arithmetic mean of the magnitude and phase spectra respectively). Here, a less acoustically-desirable result is produced. The circular range of the phase spectra has essentially been cut in half leading to significant amounts of time-domain cancellation. The result, shown in Figure 7.3, is impulse-like and the dynamic range has been increased dramatically; it sounds like a brief popped followed by an extremely quiet version of the convolution. A certain horizontal symmetry has been introduced however as halving the phase is producing something akin to autocorrelation. This effect can be exaggerated by further decreasing s and increasing q for desirable results as shown in Figure 7.4.

Returning all parameters to that of ordinary convolution and drastically in-

creasing s produces interesting effects as seen in Figure 7.5. Increasing the value of s to 4 yields an ambient texture that is still quite identifiable as originating from the source material. This result sounds almost like a sustained choir of tuned cymbals. The violin melody has been all but lost, though if one squints at the waveform one can conceivably make out the amplitude envelope of the original violin sound.

7.0.2 Voice with Voice

This section explores the ECCS of a recording of a female voice[3] and a recording of a male voice[4] speaking two similar but slightly different phrases. The female speaker is saying "I see five lamps" and the male speaker is saying "I have three books". These utterances were chosen as ECCS examples as they are clear, dry and were recorded on the same equipment. There is not much overlap in the phonemes used for the two utterances as well which eliminates ambiguity in the result. The intelligibility of speech subjected to cross-synthesis gives a good indication of whether or not the technique maintains semantic and acoustic meaning of the input. The original waveforms and their standard convolution ($[p, q, r, s] = [0.5, 1.0, 0.5, 1.0]$) are displayed in Figure 7.6. The p and r skew parameters will be manipulated for these sounds so it is important to establish which is which. With respect to Equation 3.12, the female speaker is sound f and the male speaker is sound g. In other words, higher values of p and r bias more strongly towards the female speaker and lower values towards the male.

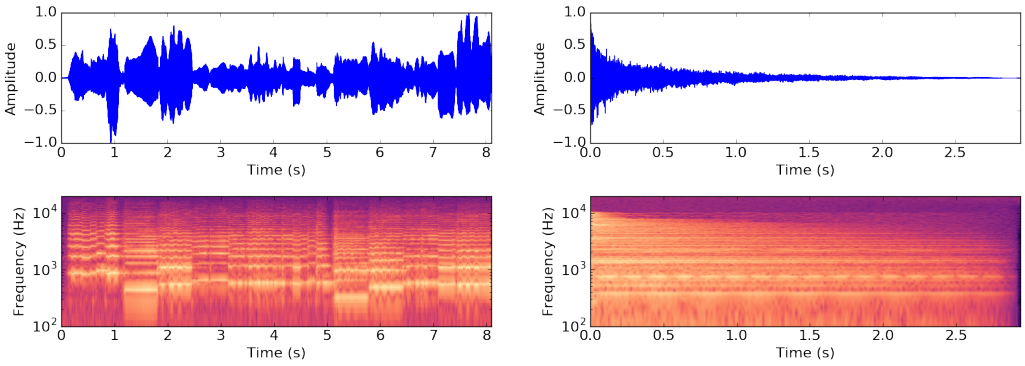
The result of ordinary CCS sound scrambled and mostly unintelligible. The semantic meaning of both sentences is lost and the result has timbral characteristics reminiscent of ring modulation. The original sounds are recoverable through

[3]https://www.freesound.org/people/margo_heston/sounds/196971/
[4]https://www.freesound.org/people/margo_heston/sounds/196966/

ECCS using the parameter combinations $[p, q, r, s] = [0.0, 0.5, 0.0, 0.5]$ and $[p, q, r, s] = [1.0, 0.5, 1.0, 0.5]$. More interesting results can be obtained somewhere in between the original sounds and their convolution. For example, one might want to understand the female speaker but also include characteristics of the male speaker. A natural parameter combination would be $[p, q, r, s] = [0.75, 1.0, 0.75, 1.0]$, shown in Figure 7.7. With this set of parameters, the female speaker is clearly audible over a background of echoes of her own speech colored by the average frequency spectra of the male speaker.

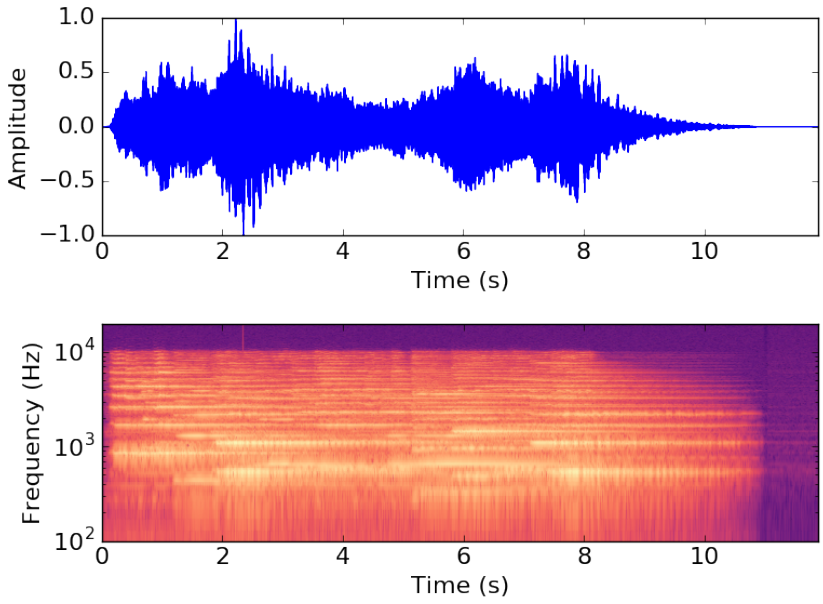
The p and r behaviors do not respond in a linear way to the human ear however; a more convincing mix of the two where the intelligibility of the male speaker is more strongly maintained is $[p, q, r, s] = [0.6, 1.0, 0.6, 1.0]$. Qualitative adjustments to the q and s parameters can be made to increase brightness and intelligibility respectively, yielding a satisfying mix at $[p, q, r, s] = [0.6, 0.75, 0.6, 0.75]$, shown in Figure 7.8. With these parameters one can clearly make out the female speaker and bits of the male speaker.

Interesting combinations can be made by using different values for p and r. For example, using the parameter combination $[p, q, r, s] = [0.0, 0.75, 1.0, 0.75]$, one can clearly understand the female speaker in the foreground while the background contains an unintelligible, ambient drone of the male speaker. This combination is depicted in Figure 7.9. This suggests that the phase skew parameter r is more important for preserving the time envelope of the input, whereas the magnitude skew parameter p is more important for preserving tonal imprint. Strangely, inverting p and r to $[p, q, r, s] = [1.0, 0.75, 0.0, 0.75]$ does not preserve the intelligibility of the male speaker as strongly. Using $[p, q, r, s] = [0.9, 0.75, 0.1, 0.75]$, shown in Figure 7.10, one can interpret the male speaker much more clearly.



(a) Short violin melody from Luigi Boccherini's *String Quintet in E major, Op. 11, No. 5, Mvt. 3*

(b) Recording of a ride cymbal struck with a stick



(c) Their convolution

Figure 7.1: Waveforms and spectra of a violin excerpt, a ride cymbal and their ordinary convolution

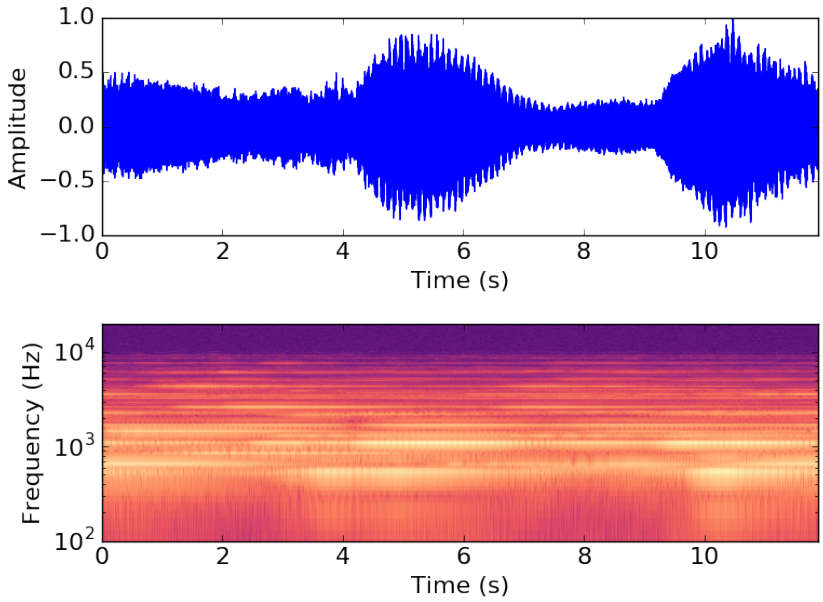


Figure 7.2: ECCS of violin and cymbal: geometric mean of magnitude spectrum ($[p, q, r, s] = [0.5, 0.5, 0.5, 1.0]$)

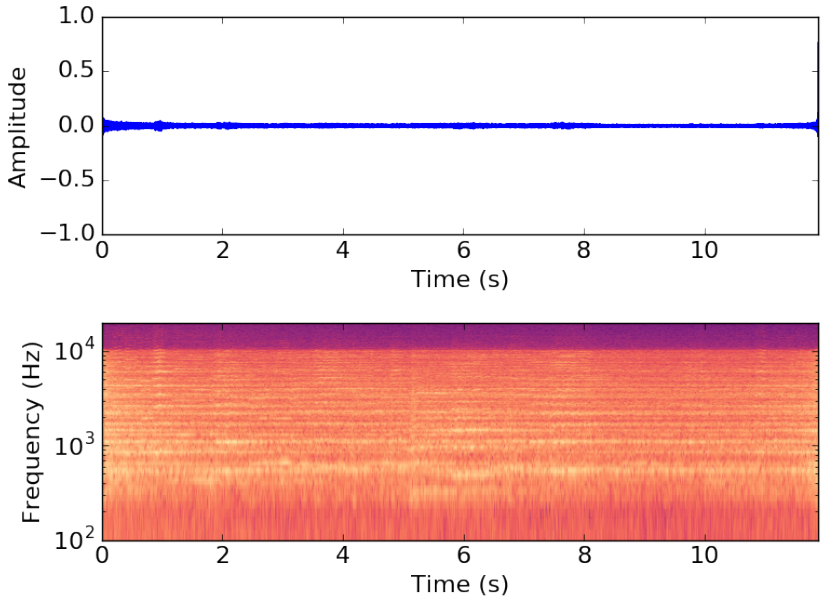


Figure 7.3: ECCS of a violin and cymbal: geometric mean of spectra ($[p, q, r, s] = [0.5, 0.5, 0.5, 0.5]$)

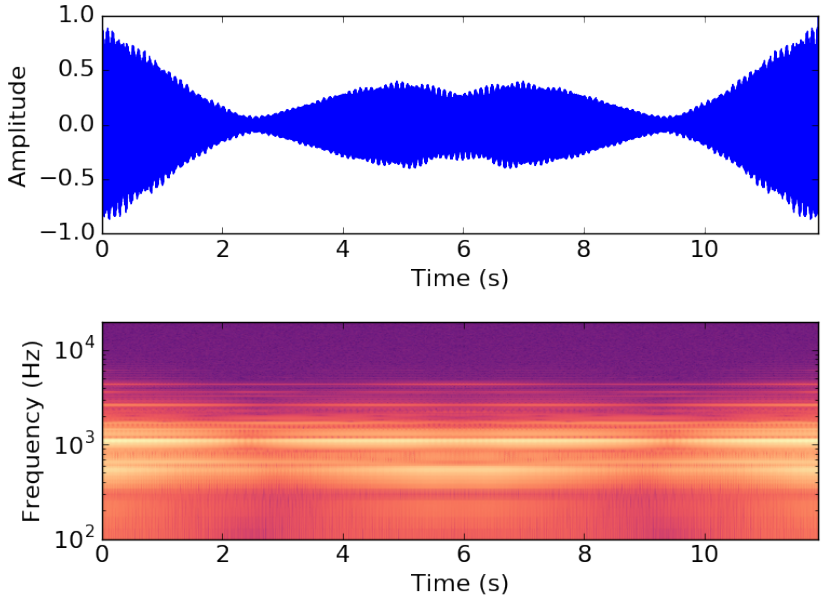


Figure 7.4: ECCS of a violin and cymbal: null phase yields symmetry $[p, q, r, s] = [0.5, 0.5, 4.0, 0.0]$

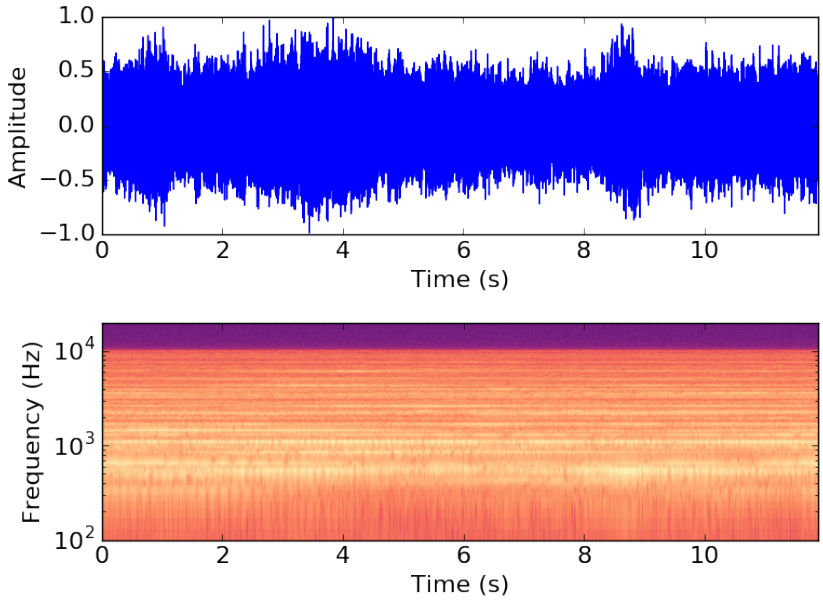
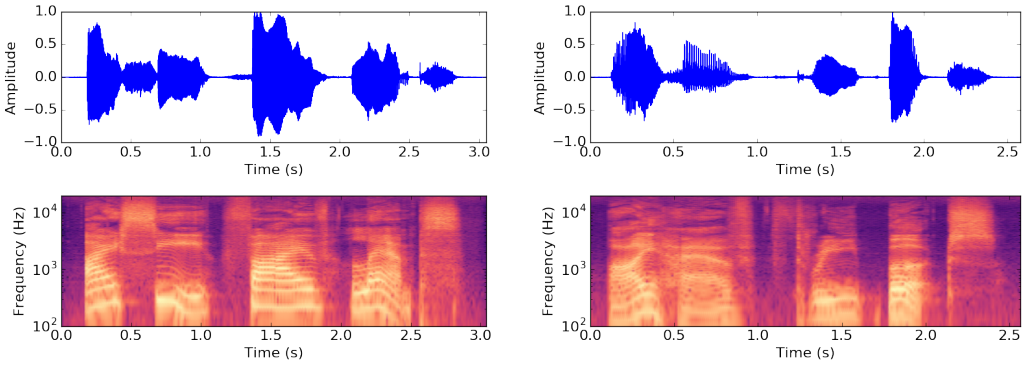
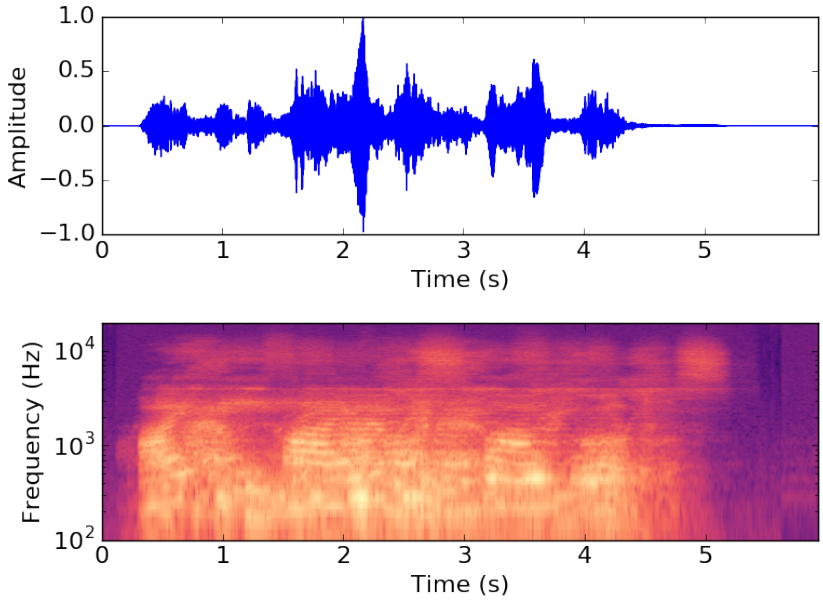


Figure 7.5: ECCS of a violin and cymbal: scattered phase yields ambience $[p, q, r, s] = [0.5, 0.5, 1.0, 4.0]$



(a) Recording of a female speaker saying "I see five lamps"

(b) Recording of a male speaker saying "I have three books"



(c) Their convolution

Figure 7.6: Waveforms and spectra of a female speaker, a male speaker and their ordinary convolution

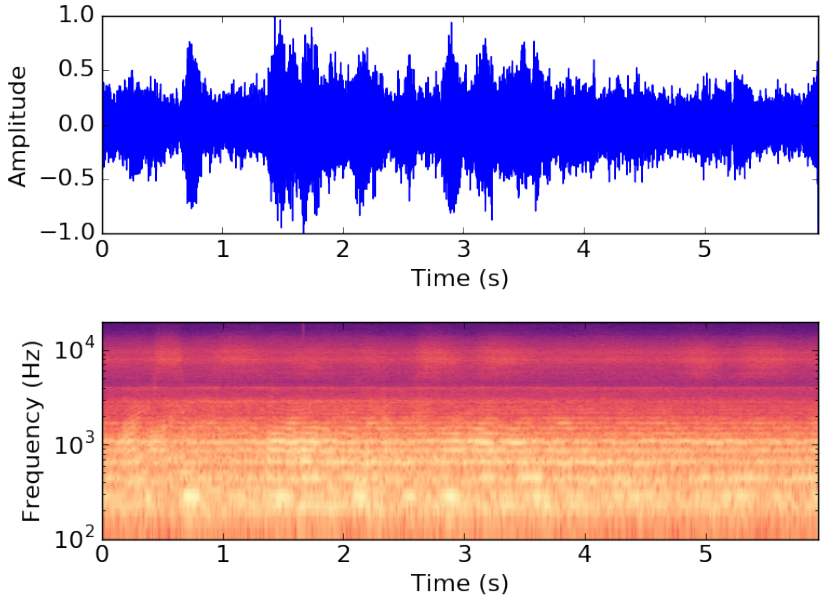


Figure 7.7: ECCS of a female and male speakers: mix with one speaker intelligible $[p, q, r, s] = [0.75, 1.0, 0.75, 1.0]$

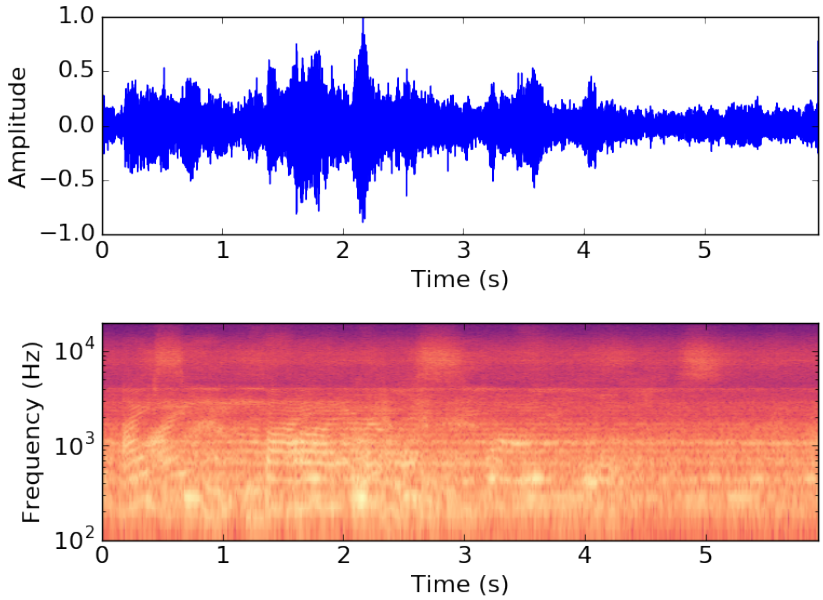


Figure 7.8: ECCS of a female and male speaker: mix of two speakers with more clarity $[p, q, r, s] = [0.6, 0.75, 0.6, 0.75]$

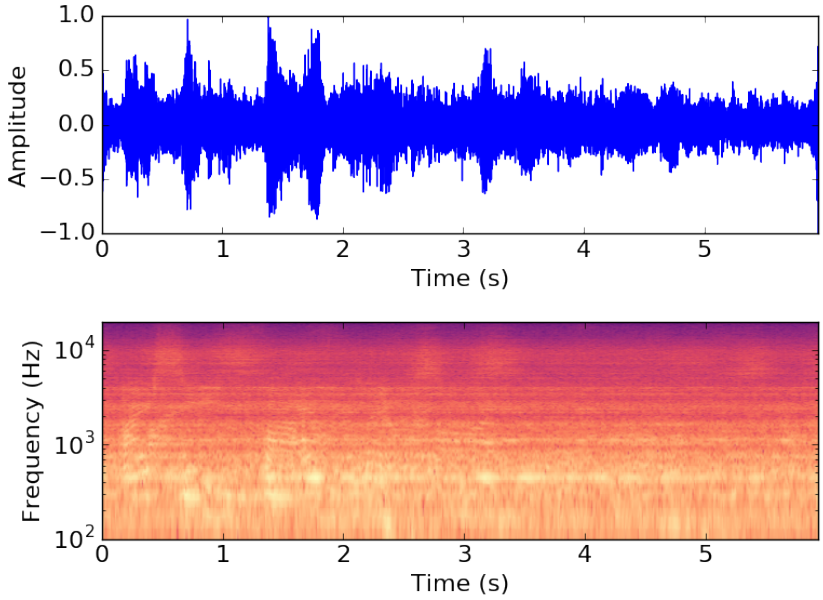


Figure 7.9: ECCS of a female and male speaker: inverted magnitude and phase bias $[p, q, r, s] = [0.0, 0.75, 1.0, 0.75]$

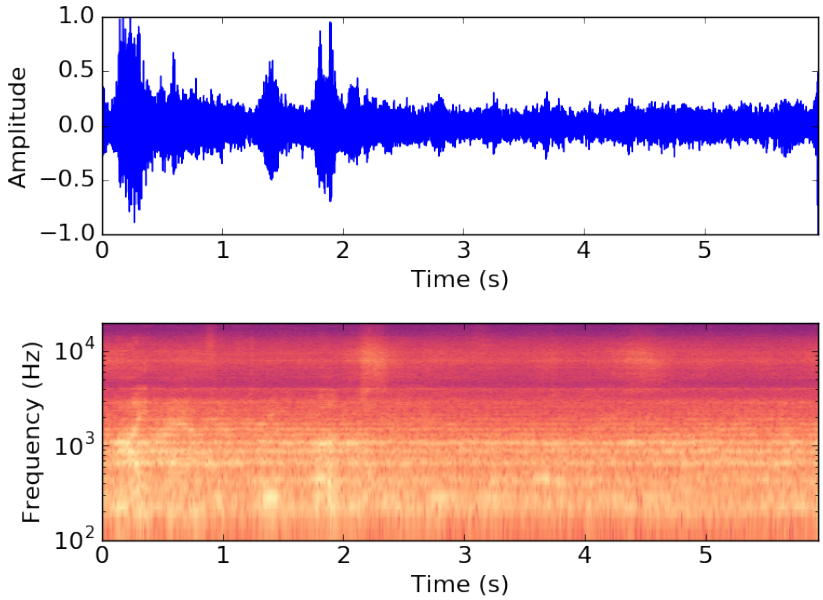


Figure 7.10: ECCS of a female and male speaker: opposing inversion of magnitude and phase bias $[p, q, r, s] = [0.9, 0.75, 0.1, 0.75]$

Chapter 8

Conclusions

This thesis has introduced novel extensions to fast convolution for the purposes of creating a generalized cross-synthesis technique for arbitrary sound input. The proposed extensions were analyzed objectively using RFS1k, a curated dataset of heterogeneous sounds. Random pairings of sounds in RFS1k were subjected to ECCS using a variety of parameter configurations to determine the effect that these configurations had on acoustic features on average. This black-box analysis incidentally established acoustic feature baselines not only for ECCS but for CCS and a heterogeneous sampling of sound files. The proposed extensions were also analyzed subjectively on a few examples as a method of explaining the intuitive functionality of each parameter.

Artifacts that arise through the process of ECCS were presented without much resolution. Future work for this topic will likely focus on methods to mitigate these effects such as experimenting with different window functions and alternate methods of spectral manipulation. Ideally, this further investigation would yield a real-time method that produces equivalent or similar results to the offline methods described in

this thesis.

A cross-platform, C++ software application called JECT (**J**UCE **E**xtended **C**onvolution **T**echniques) that implements the techniques described in this thesis can be obtained at http://chrisdonahue.github.io/ject/. The software allows users to perform ECCS on any number of sound files. A screenshot is shown in Figure 8.1.

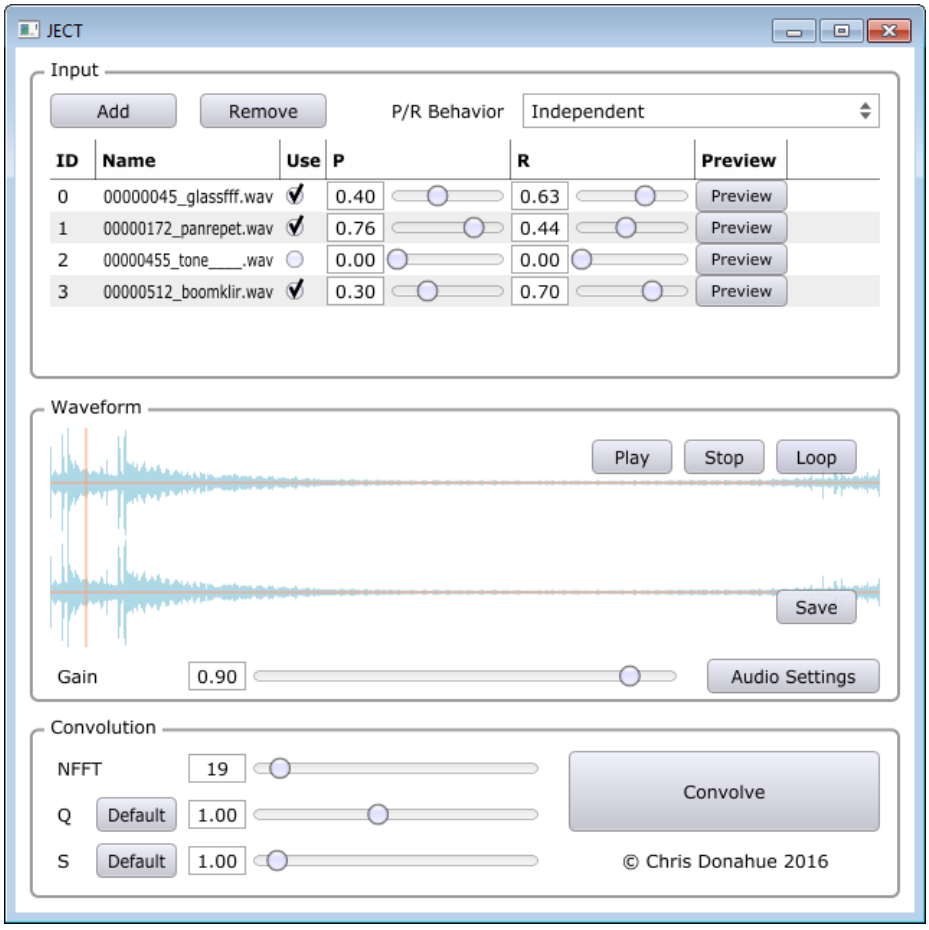


Figure 8.1: Screenshot of JECT performing ECCS on three sound files

The RFS1k dataset can be obtained from http://rfs1k.ucsd.edu/. For each sound RFS1k includes the original WAV audio files as uploaded by the user, two OGG preview encodings and a JSON file of metadata.

Bibliography

[BA11] Eric Battenberg and Rimas Avizienis. Implementing real-time partitioned convolution algorithms on conventional operating systems. In *Proceedings of the 14th International Conference on Digital Audio Effects. Paris, France*, 2011.

[Blu70] Leo I Bluestein. A linear filtering approach to the computation of discrete fourier transform. *Audio and Electroacoustics, IEEE Transactions on*, 18(4):451–455, 1970.

[Bou85] Richard Boulanger. *The transformation of speech into music: a musical exploration and interpretation of two recent digital techniques*. PhD thesis, University of California, San Diego, 1985.

[BWG+13] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José R Zapata, and Xavier Serra. Essentia: An audio analysis library for music information retrieval. In *ISMIR*, pages 493–498. Citeseer, 2013.

[DEP16] Chris Donahue, Tom Erbe, and Miller Puckette. Extended convolution techniques for cross-synthesis. 2016.

[Dol85] Mark Dolson. *Recent advances in musique concrete at CARL*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 1985.

[Dud39] Homer Dudley. *The vocoder*. publisher not identified, 1939.

[Erb11] Tom Erbe. *PVOC KIT: New Applications of the Phase Vocoder*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 2011.

[FRS13] Frederic Font, Gerard Roma, and Xavier Serra. Freesound technical demo. In *Proceedings of the 21st acm international conference on multimedia*, pages 411–412. ACM, 2013.

[Gar94] William G Gardner. Efficient convolution without input/output delay. In *Audio Engineering Society Convention 97*. Audio Engineering Society, 1994.

[MIBH04] Hemant Misra, Shajith Ikbal, Hervé Bourlard, and Hynek Hermansky. Spectral entropy based feature for robust asr. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 1, pages I–193. IEEE, 2004.

[MMDJ61] MV Mathews, Joan E Miller, and EE David Jr. Pitch synchronous analysis of voiced sounds. *The Journal of the Acoustical Society of America*, 33(2):179–186, 1961.

[Moo78] James A Moorer. The use of the phase vocoder in computer music applications. *Journal of the Audio Engineering Society*, 26(1/2):42–45, 1978.

[Moo79a] James A Moorer. About this reverberation business. *Computer music journal*, pages 13–28, 1979.

[Moo79b] James A Moorer. The use of linear prediction of speech in computer music applications. *Journal of the Audio Engineering Society*, 27(3):134–140, 1979.

[Pee04] Geoffroy Peeters. A large set of audio features for sound description (similarity and classification) in the cuidado project. Technical report, IRCAM, 2004.

[Pet75] Tracy Lind Petersen. *Vocal tract modulation of instrumental sounds by digital filtering*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 1975.

[Roa96] Curtis Roads. *The computer music tutorial*. MIT press, 1996.

[Roa97] Curtis Roads. *Musical signal processing*, chapter Sound transformation by convolution. Routledge, 1997.

[RSR69] Lawrence R Rabiner, Ronald W Schafer, and Charles M Rader. The chirp z-transform algorithm and its application. *Bell System Technical Journal*, 48(5):1249–1292, 1969.

[SA07] Tamara Smyth and Jonathan S Abel. Convolutional synthesis of wind instruments. In *Applications of Signal Processing to Audio and Acoustics, 2007 IEEE Workshop on*, pages 219–222. IEEE, 2007.

[Ser97] Marie-Helene Serra. *Musical signal processing*, chapter Introducing the phase vocoder. Routledge, 1997.

[SJ66] Thomas G Stockham Jr. High-speed convolution and correlation. In *Proceedings of the April 26-28, 1966, Spring joint computer conference*, pages 229–233. ACM, 1966.

[Ste75] Stanley Smith Stevens. *Psychophysics*. Transaction Publishers, 1975.

[TC99] George Tzanetakis and Perry Cook. Multifeature audio segmentation for browsing and annotation. In *Applications of Signal Processing to Audio and Acoustics, 1999 IEEE Workshop on*, pages 103–106. IEEE, 1999.