Recent CCRMA Research in Digital Audio Synthesis, Processing, and Effects

Julius O. Smith III
CCRMA, Stanford University

DAFx-09, Keynote III, Como, Italy

September 4, 2009

Overview

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Overview
- Previously at DAFx06
- Research Update
- CCRMA

Spring Reverb Models

Virtual Analog Circuits

Acoustic Guitar Models

Haptic Instruments

New Oscillators

Spectral Delay Filters

Audio FFT Filter Banks

Microphone Array

Faust to Flash Plugins

Summary

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)

  What can I talk about now?

  New research results in the past three years?

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Three Years Ago

In my DAFx06 review, we looked at

- Early digital audio effects (delay lines, scanner vibrato, . . .)
- Acoustic propagation models
- Digital waveguide models (voice, strings, woodwinds, . . .)
- Commuted synthesis (acoustic guitar, harpsichord, piano, . . .)
- *All my best accumulated sound examples!*
- That presentation is available online:
  `http://ccrma.stanford.edu/~jos/pdf/DAFx06KeynoteII.pdf`
  (sound examples via HTTP)
  What can I talk about now?
  New research results in the past three years?

Since Three Years Ago

Problem:

I haven't done that much in the past three years!

Solution:

Summarize recent DAFx-related research at CCRMA as a whole

- **Talk Design:**
  - New results in the past year (three years is too much)
  - Developed at CCRMA
  - JOS involved as collaborator or adviser (*i.e.*, I know something about it!)

Overview
- Previously at DAFx06
- Research Update
- CCRMA

Spring Reverb Models

Virtual Analog Circuits

Acoustic Guitar Models

Haptic Instruments

New Oscillators

Spectral Delay Filters

Audio FFT Filter Banks

Microphone Array

Faust to Flash Plugins

Summary

Since Three Years Ago

Problem:

I haven’t done that much in the past three years!

Solution:

Summarize recent DAFx-related research at CCRMA as a whole

- **Talk Design:**
  - New results in the past year (three years is too much)
  - Developed at CCRMA
  - JOS involved as collaborator or adviser (*i.e.*, I know something about it!)

Overview
- Previously at DAFx06
- Research Update
- CCRMA

Spring Reverb Models

Virtual Analog Circuits

Acoustic Guitar Models

Haptic Instruments

New Oscillators

Spectral Delay Filters

Audio FFT Filter Banks

Microphone Array

Faust to Flash Plugins

Summary

Since Three Years Ago

Problem:

I haven’t done that much in the past three years!

Solution:

Summarize recent DAFx-related research at CCRMA as a whole

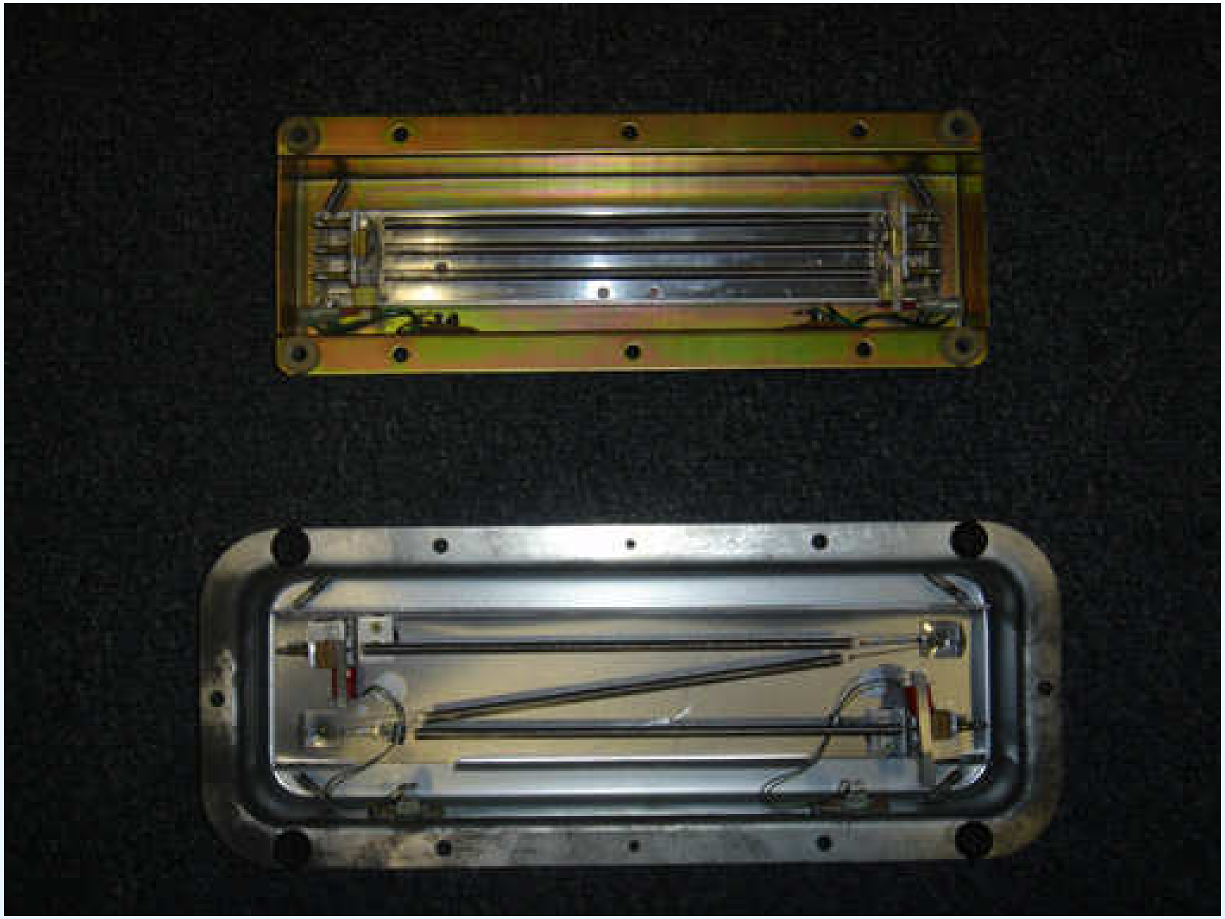
- **Talk Design:**
  - New results in the past year (three years is too much)
  - Developed at CCRMA
  - JOS involved as collaborator or adviser (*i.e.*, I know something about it!)

DAFx-Related Research involving JOS at CCRMA, 2008-2009

CCRMA building: The Knoll, Stanford University

Multimodal Spring Reverb Modeling

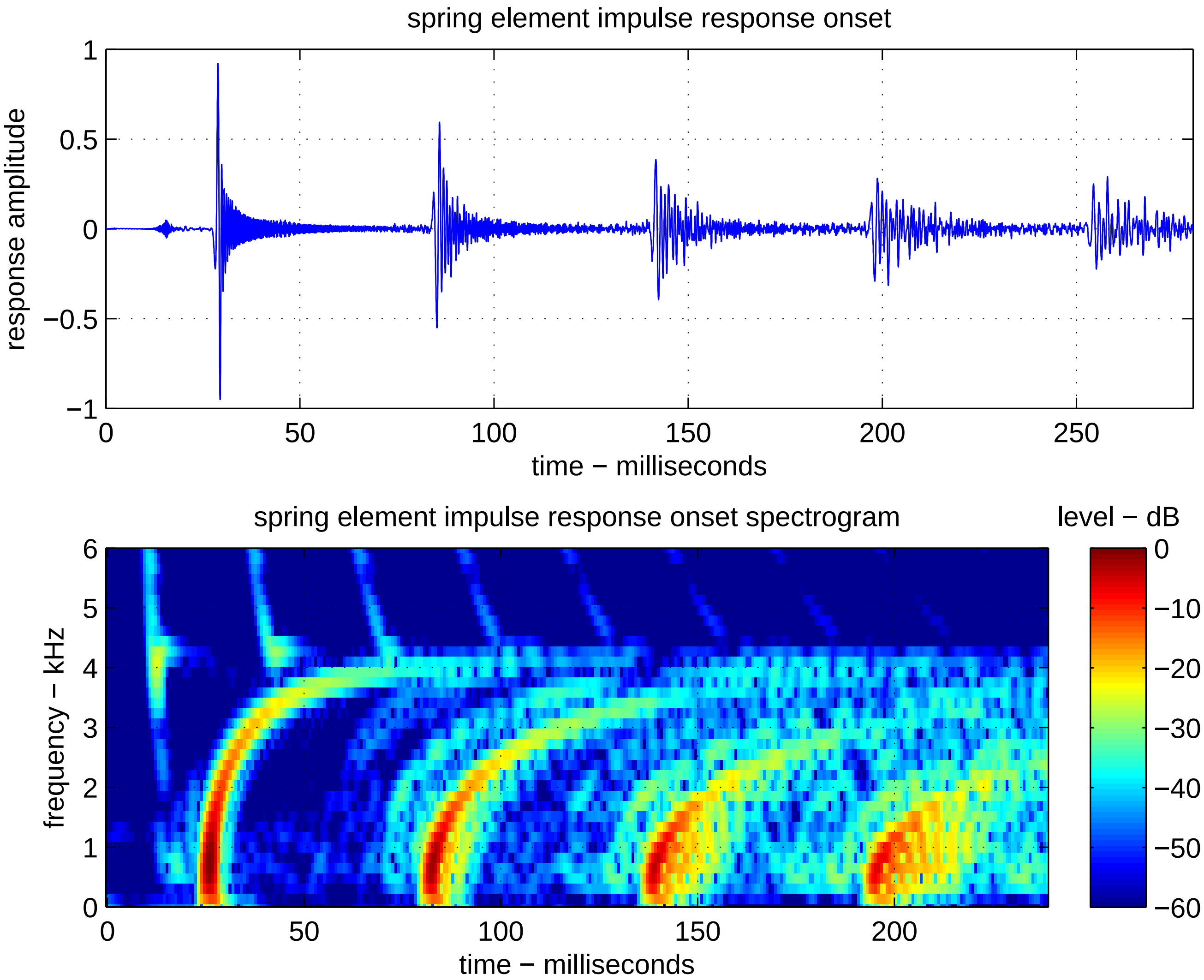
Accutronics Type 8 Spring Tank



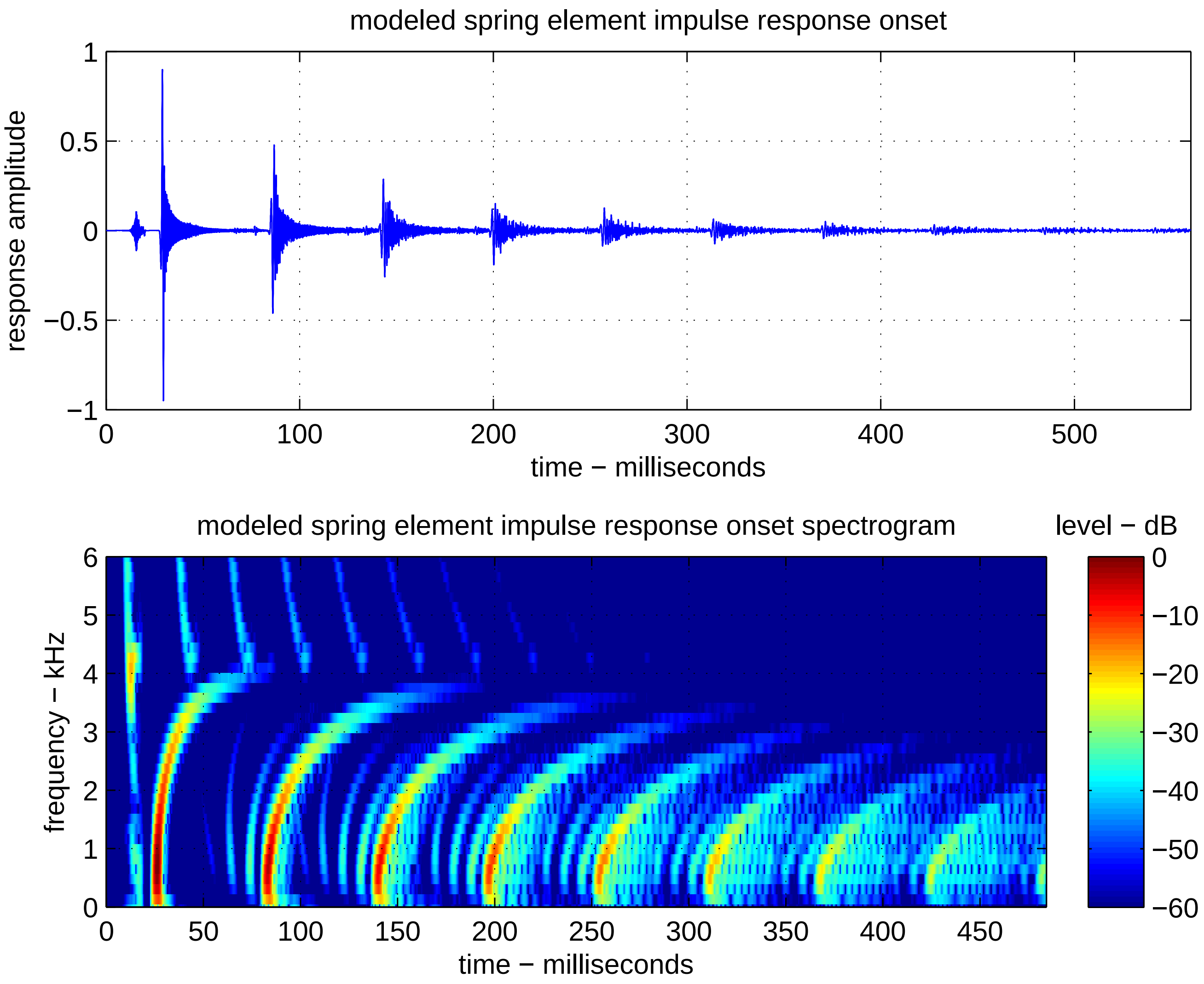
Top: Accutronics Type 8 Spring Tank

Bot: Single Spring "Folded" into a Compact Space

Measured Single-Spring Impulse Response



Model Impulse Response



Model Impulse Response in Time and Frequency Domains

Spring Reverb Sound Examples

- Dry Source Signal: (WAV) (MP3)
- Measured Spring-Reverb Response: (WAV) (MP3)
- Spring-Reverb Model Response: (WAV) (MP3)

Submitted Paper: *“A Spring Reverb Model Employing Coupled Torsional and Longitudinal Modes”*

Jonathan Abel, Dave Berners, Kyle Spratt, and Julius Smith

(in review)

Virtual Analog Circuits

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

Digital Implementation of Musical Distortion Circuits by Analysis and Simulation — June 2009

- Analog Audio Circuits → Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the "K Method" to
  - Nonlinear circuits, with automated application to "netlists"
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are "precomputed" as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

Digital Implementation of Musical Distortion Circuits by Analysis and Simulation — June 2009

- Analog Audio Circuits → Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the “K Method” to
  - Nonlinear circuits, with automated application to “netlists”
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are “precomputed” as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

> *Digital Implementation of Musical Distortion Circuits by Analysis and Simulation* — June 2009

- Analog Audio Circuits → Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the “K Method” to
  - Nonlinear circuits, with automated application to “netlists”
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are “precomputed” as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

> *Digital Implementation of Musical Distortion Circuits by Analysis and Simulation* — June 2009

- Analog Audio Circuits → Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the "K Method" to
  - Nonlinear circuits, with automated application to "netlists"
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are "precomputed" as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

> *Digital Implementation of Musical Distortion Circuits by Analysis and Simulation* — June 2009

- Analog Audio Circuits → Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the “K Method” to
    - Nonlinear circuits, with automated application to “netlists”
    - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are “precomputed” as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

Digital Implementation of Musical Distortion Circuits by Analysis and Simulation — June 2009

- Analog Audio Circuits $\longrightarrow$ Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the "K Method" to
  - Nonlinear circuits, with automated application to "netlists"
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are "precomputed" as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

Digital Implementation of Musical Distortion Circuits by Analysis and Simulation — June 2009

- Analog Audio Circuits $\longrightarrow$ Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the “K Method” to
  - Nonlinear circuits, with automated application to “netlists”
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are “precomputed” as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

Digital Implementation of Musical Distortion Circuits by Analysis and Simulation — June 2009

- Analog Audio Circuits → Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the "K Method" to
  - Nonlinear circuits, with automated application to "netlists"
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are "precomputed" as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

Digitizing Circuits in Real Time

Recent CCRMA/EE thesis by **David Yeh**:

Digital Implementation of Musical Distortion Circuits by Analysis and Simulation — June 2009

- Analog Audio Circuits → Real-Time Digital Audio Effects
- Includes work of four past DAFx papers
- Linear and nonlinear methods for digitizing circuits
- Nonlinear methods similar to SPICE (implicit) but modified for *real time* circuit-solving (semi-implicit) in discrete time
- One method extends the “K Method” to
  - Nonlinear circuits, with automated application to “netlists”
  - Discrete-time nonlinear state-space formulation (resolves issues with certain circuit types)
- Instantaneous nonlinearities are “precomputed” as in K Method
- Can be applied to transistor and vacuum-tube circuits
- Accuracy limited primarily by the underlying device models

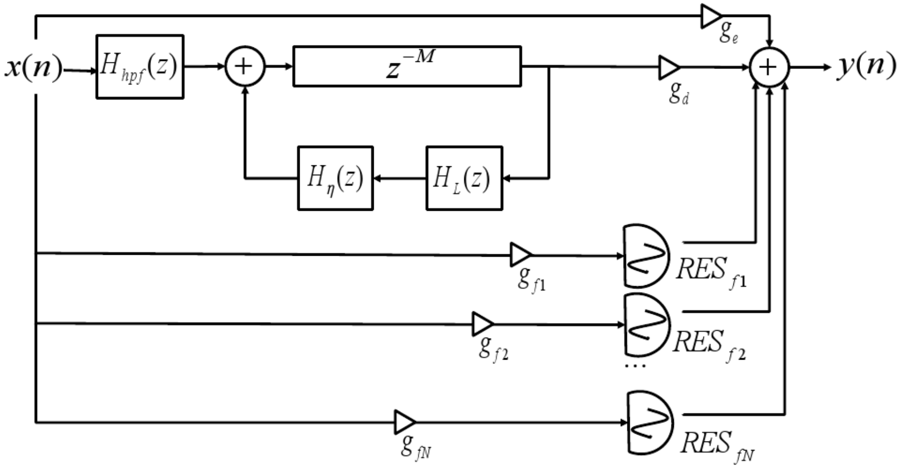
Virtual Acoustic Guitar Models

Coupled Strings Analysis and Synthesis

Submitted paper (from pending CCRMA/CS thesis) by **Nelson Lee**:

“Analysis and Synthesis of Coupled Vibrating Strings
Using a Hybrid Modal-Waveguide Synthesis Model”

Nelson Lee, Julius Smith, and Vesa Välimäki (in review)



Similar to Balázs Bank formulation, but replacing low-frequency partials by fourth-order resonators (instead of adding second-order resonators to existing partials)

Virtual Acoustic Guitar Sound Examples of Individual Effects

From Nelson Lee's thesis defense:

- Original waveform: (WAV) (MP3)
- Simple lossless, reflectively terminated digital waveguide (DWG): (WAV) (MP3)
- Add loop filter: (WAV) (MP3)
- Add interpolation filter: (WAV) (MP3)
- Add excitation (ICMC07): (WAV) (MP3)
- Add body response: (WAV) (MP3)
- Add hybrid modal/waveguide model: (WAV) (MP3)
- Exaggerate pitch glide due to tension modulation: (WAV) (MP3)

Virtual Acoustic Guitar Sound Examples

More Nelson Lee examples:

- Original 1: (WAV) (MP3)
- Synthesized 1: (WAV) (MP3)
- Original 2: (WAV) (MP3)
- Synthesized 2: (WAV) (MP3)
- Original 3: (WAV) (MP3)
- Synthesized 3: (WAV) (MP3)
- Original 4: (WAV) (MP3)
- Synthesized 4: (WAV) (MP3)
- Original 5: (WAV) (MP3)
- Synthesized 5: (WAV) (MP3)
- Original 6: (WAV) (MP3)
- Synthesized 6: (WAV) (MP3)
- Synthesized Chord Demo: (WAV) (MP3)

Haptic Feedback Control for Virtual Instruments

Haptic Virtual Musical Instruments

CCRMA/EE PhD student **Ed Berdahl** is working on

Haptic Feedback Control for Virtual Instruments

Dissertation expected by the end of the year

Goals:

- Assist and/or augment *gestures*
- Assist with *accurate playing*
- Recent projects:
    - Haptically plucked virtual string
    - Active drumhead (one-handed rolls, etc.): `http://ccrma.stanford.edu/~eberdahl/Projects/HapticDrum/`

Haptic Virtual Musical Instruments

CCRMA/EE PhD student **Ed Berdahl** is working on

Haptic Feedback Control for Virtual Instruments

Dissertation expected by the end of the year

Goals:

- Assist and/or augment *gestures*
- Assist with *accurate playing*
- Recent projects:
  - Haptically plucked virtual string
  - Active drumhead (one-handed rolls, etc.): `http://ccrma.stanford.edu/~eberdahl/Projects/HapticDrum/`

Haptic Virtual Musical Instruments

CCRMA/EE PhD student **Ed Berdahl** is working on

Haptic Feedback Control for Virtual Instruments

Dissertation expected by the end of the year

Goals:

- Assist and/or augment *gestures*
- Assist with *accurate playing*
- Recent projects:
  - Haptically plucked virtual string
  - Active drumhead (one-handed rolls, etc.): `http://ccrma.stanford.edu/~eberdahl/Projects/HapticDrum/`

Haptic Virtual Musical Instruments

CCRMA/EE PhD student **Ed Berdahl** is working on

Haptic Feedback Control for Virtual Instruments

Dissertation expected by the end of the year

Goals:

- Assist and/or augment *gestures*
- Assist with *accurate playing*
- Recent projects:
    - Haptically plucked virtual string
    - Active drumhead (one-handed rolls, etc.): `http://ccrma.stanford.edu/~eberdahl/Projects/HapticDrum/`

Haptic Virtual Musical Instruments

CCRMA/EE PhD student **Ed Berdahl** is working on

Haptic Feedback Control for Virtual Instruments

Dissertation expected by the end of the year

Goals:

- Assist and/or augment *gestures*
- Assist with *accurate playing*
- Recent projects:
  - Haptically plucked virtual string
  - Active drumhead (one-handed rolls, etc.): `http://ccrma.stanford.edu/~eberdahl/Projects/HapticDrum/`

Haptic Virtual Musical Instruments

CCRMA/EE PhD student **Ed Berdahl** is working on

Haptic Feedback Control for Virtual Instruments

Dissertation expected by the end of the year

Goals:

- Assist and/or augment *gestures*
- Assist with *accurate playing*
- Recent projects:
  - Haptically plucked virtual string
  - Active drumhead (one-handed rolls, etc.): `http://ccrma.stanford.edu/~eberdahl/Projects/HapticDrum/`

New Digital Oscillator Algorithms

Sawtooth Waveforms via Differentiated Polynomials

Given

$$f(x) = x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0$$

then differentiating $n-1$ times gives

$$f^{(n)}(x) = n!\, x + (n-1)!\, a_{n-1}.$$

This first-order polynomial (a line segment) generates a *sawtooth waveform* as x periodically traverses $[-1, 1]$.

Differentiated Polynomial Wave (DPW) Sawtooth Synthesis

1. Digitally synthesize

$$x(n) = f\left[2\left(\frac{nT}{P}\ \text{mod}\ 1\right) - 1\right],$$

where

$$\begin{aligned} f(x) &= x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0 \\ T &= \text{sampling interval (sec)} \\ P &= \text{desired period (sec)} \\ n &= \text{sample number (integer)} \end{aligned}$$

2. Apply $n-1$ *first-order finite differences* $x_{k+1}(n) = [x_k(n) - x_k(n-1)]/(2T/P)$ to get

$$X_{n-1}(z) = \left(\frac{1 - z^{-1}}{2T/P}\right)^{n-1} X(z)$$

Minimizing Aliasing

Initial waveform is given by sampling

$$f(x) = x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0, \quad x \in [-1, 1).$$

After $n-1$ derivatives, we get

$$f^{(n)}(x) = n!\,x + (n-1)!\,a_{n-1}.$$

- For zero mean, set $a_{n-1} = 0$
- The $n-1$ remaining degrees of freedom in $f(x)$ can be used to *maximize flatness* at the transition from $x = 1$ to $x = -1$
- This smoothness *minimizes aliasing* in the synthesized sawtooth
- We can set $a_0 = 0$ because it has no effect on smoothness
- This leaves $n-2$ coefficients to optimize

Overview
Spring Reverb Models
Virtual Analog Circuits
Acoustic Guitar Models
Haptic Instruments
New Oscillators
- Sawtooth Synth
- Diff'd Polynomials
- Aliasing Suppression
- More Examples
- Comparisons
- Comparisons LogF
- Aliasing Masked
- Sound Example

Spectral Delay Filters
Audio FFT Filter Banks
Microphone Array
Faust to Flash Plugins
Summary

First Several Maximum-Smoothness Examples

Results for polynomial orders up to $n = 6$:

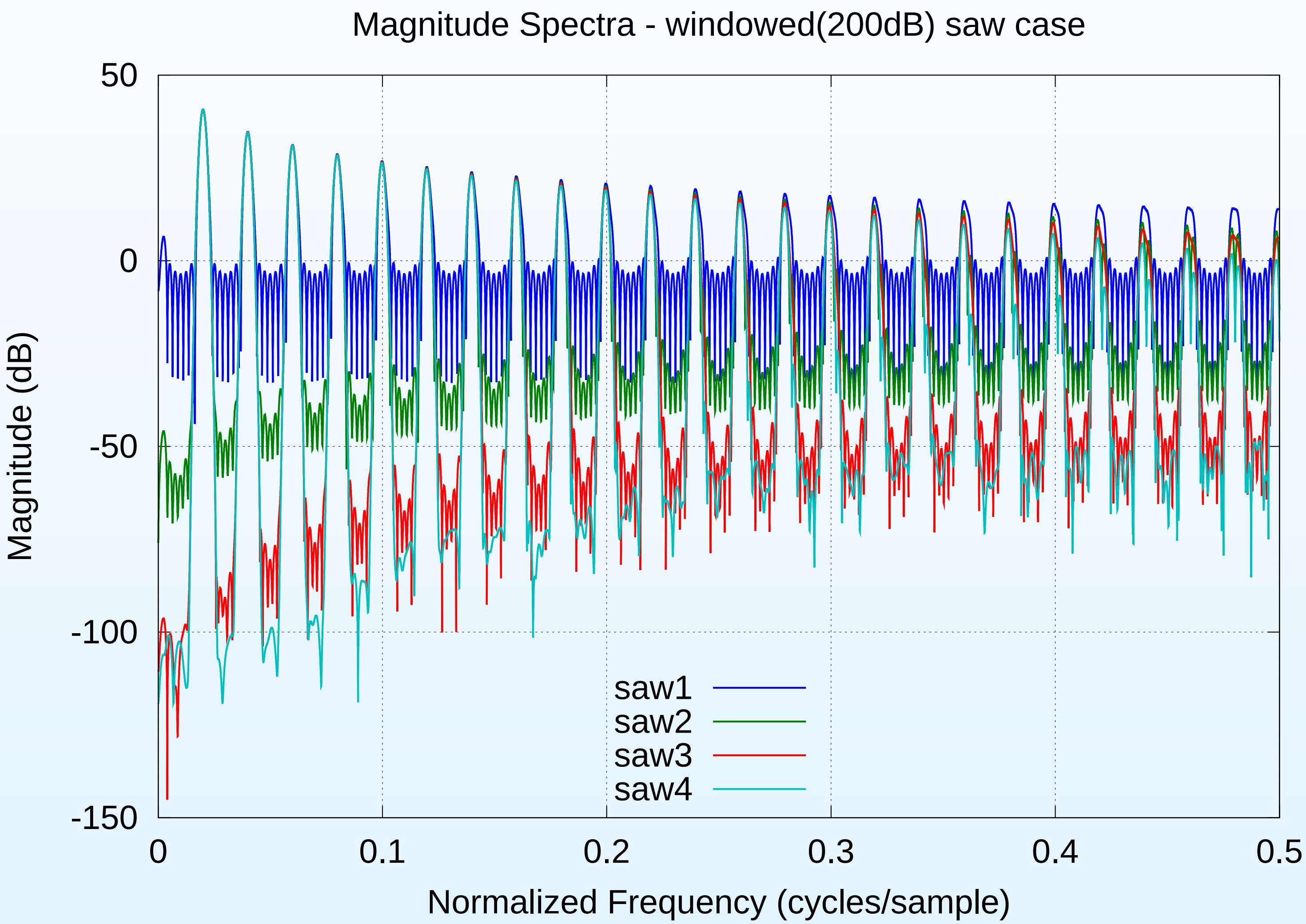
$$\begin{aligned}
f_2(x) &= x^2 \\
f_3(x) &= x^3 - x \\
f_4(x) &= x^4 - 2x^2 \\
f_5(x) &= x^5 - \frac{10}{3}x^3 + \frac{7}{3} \\
f_6(x) &= x^6 - 5x^4 + 7x^2
\end{aligned}$$

Submitted Paper: *"Alias-Suppressed Oscillators based on Differentiated Polynomial Waveforms"*

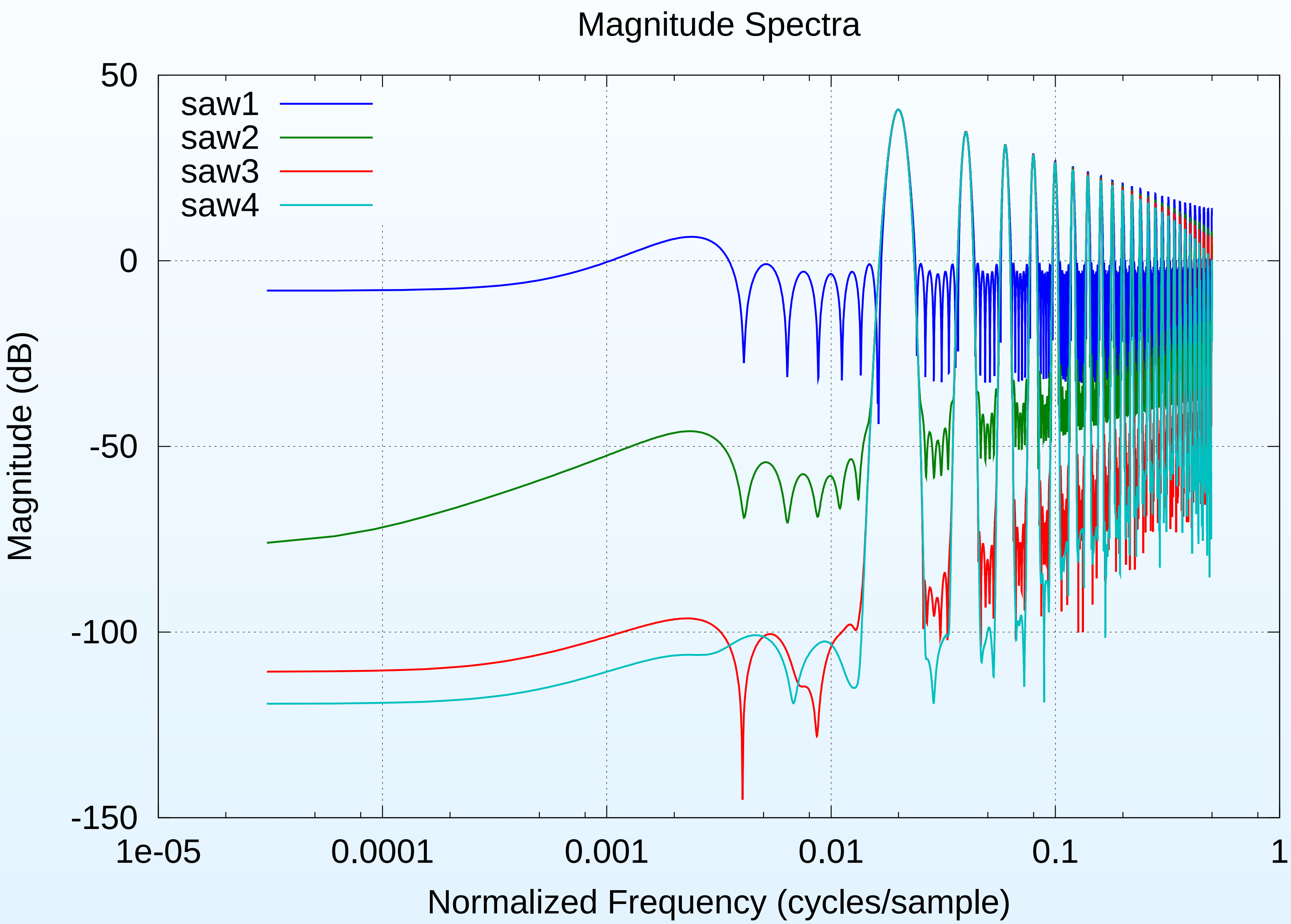
Vesa Välimäki, Juhan Nam, Julius Smith, and Jonathan Abel

IEEE Transactions on Acoustics, Speech, and Language Processing March 2010 (accepted for publication)

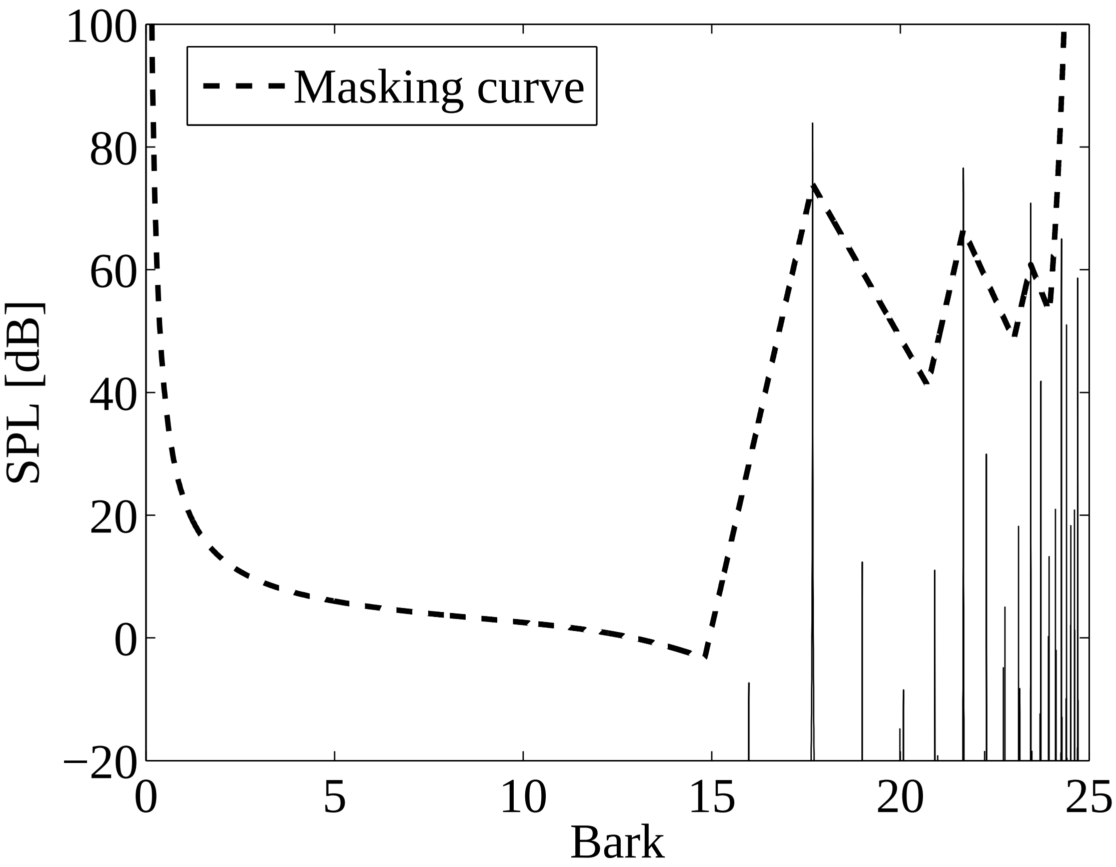
Comparison of First Four Polynomial Orders



Same Comparison over Log Frequency



Relation of Aliased Components to Masking Threshold



Spectrum of a sawtooth waveform over masking threshold

- F0 = 4.3 kHz, Fs = 44.1 kHz
- 3rd-order B-spline interpolation = 4th-order DPW

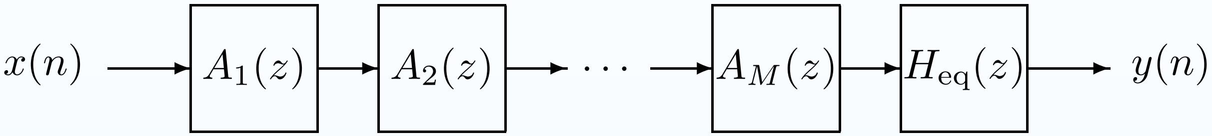
[Juhan Nam]

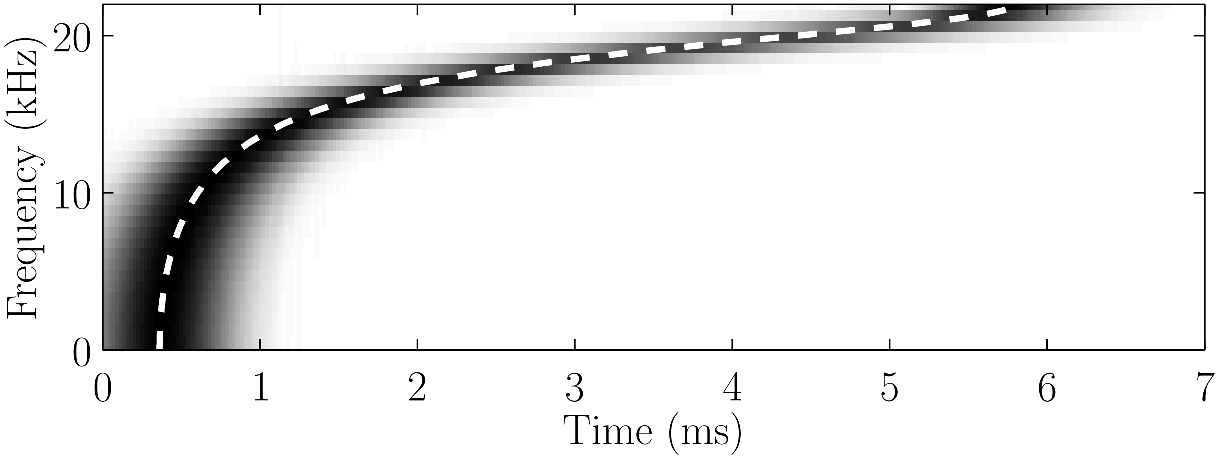
Sound Examples

- Plain Digital Sawtooth
- Differentiated-Parabolic-Wave Sawtooth
- Doubly Differentiated Cubic-Wave Sawtooth

Spectral Delay Filters

Spectral Delay Filters





See presentation later this morning (11:20 am):

“Spectral Delay Filters with Feedback Delay and Time-Varying Coefficients”

Jussi Pekonen, Vesa Välimäki, Jonathan Abel, and Julius Smith

Overview

Spring Reverb Models

Virtual Analog Circuits

Acoustic Guitar Models

Haptic Instruments

New Oscillators

Spectral Delay Filters
- Spectral Delay Filters
- Spectral Delay Filters

Audio FFT Filter Banks

Microphone Array

Faust to Flash Plugins

Summary

Spectral Delay Filters

Recent Paper (published online):

“Spectral Delay Filters”

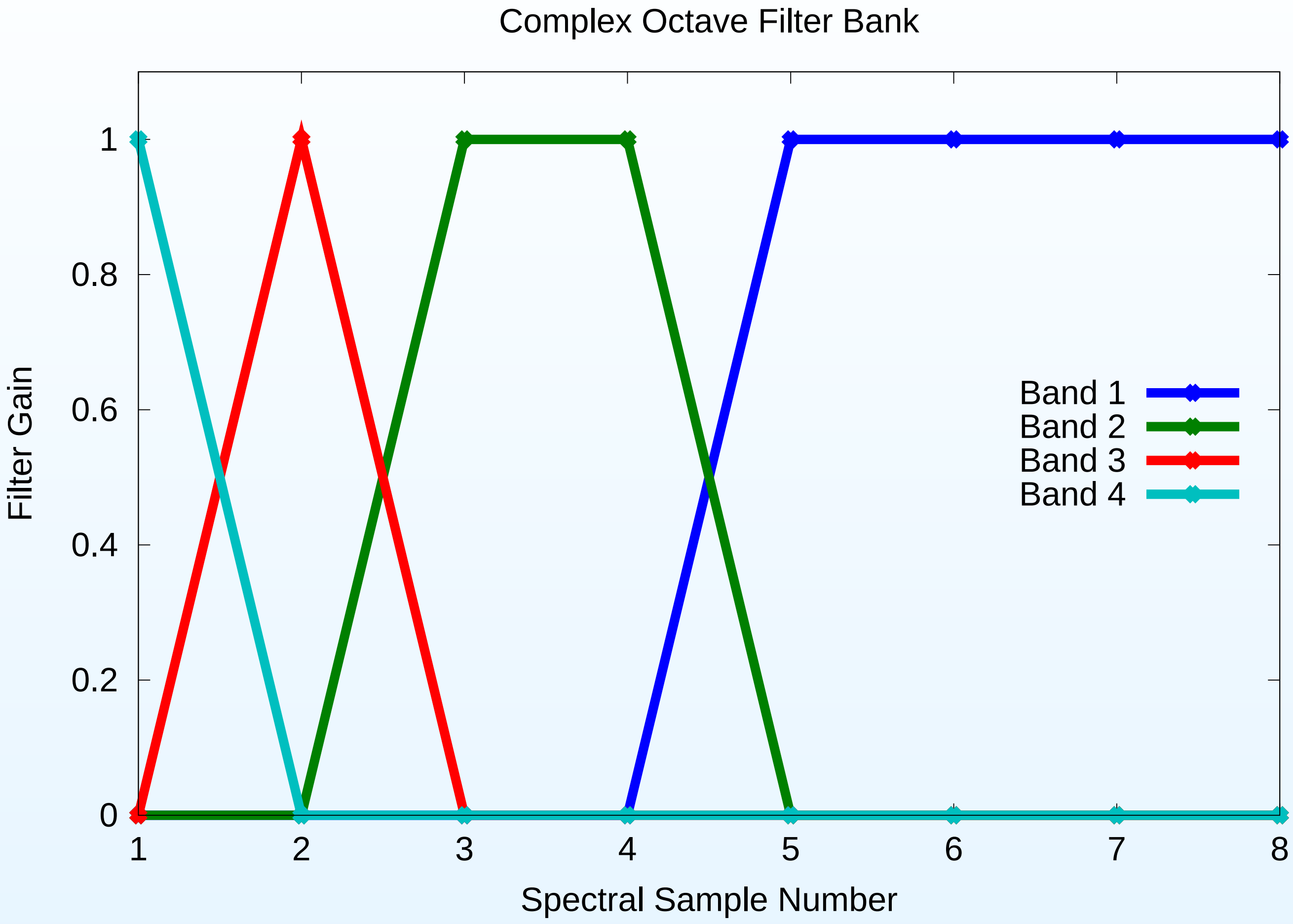
Vesa Välimäki, Jonathan Abel, and Julius Smith

Journal of the Audio Engineering Society

July/August 2009

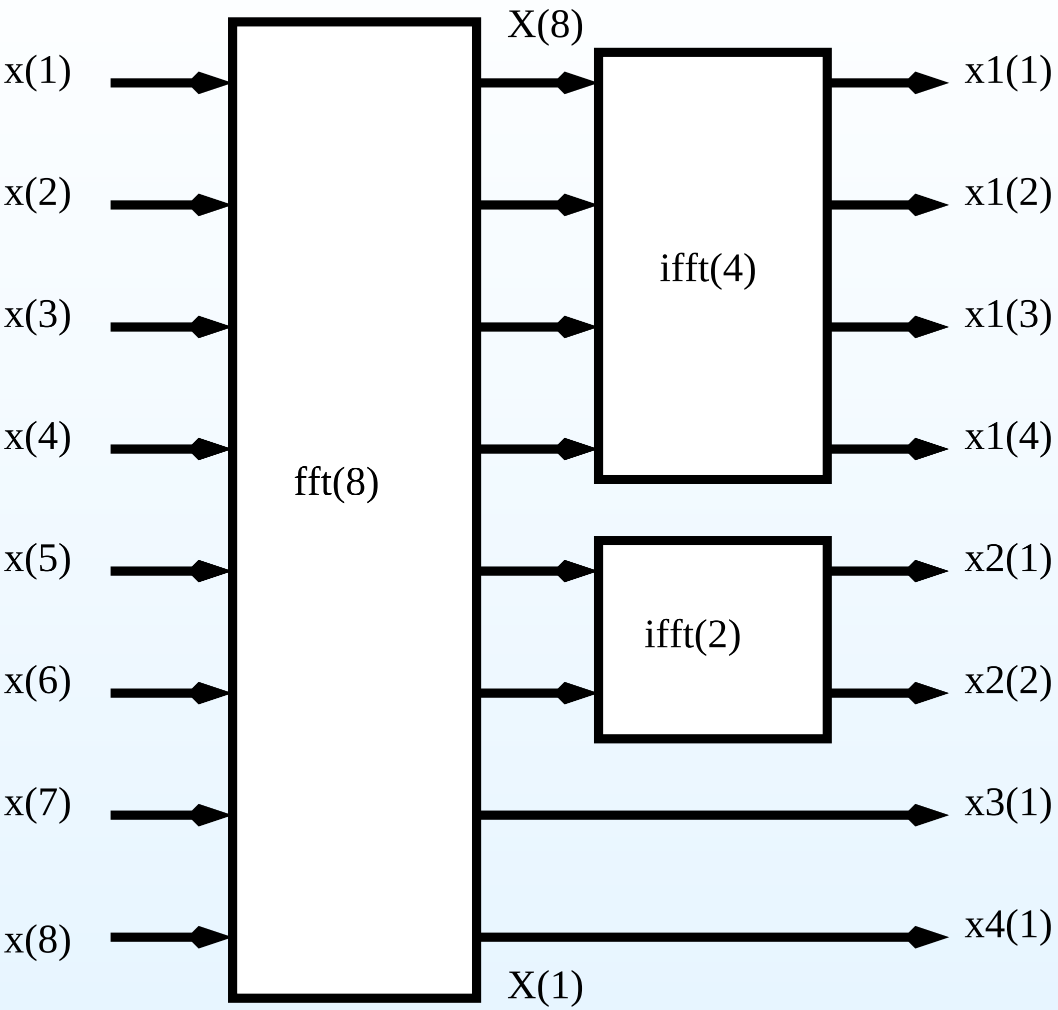
Audio FFT Filter Banks

Octave Filter Bank Schematic



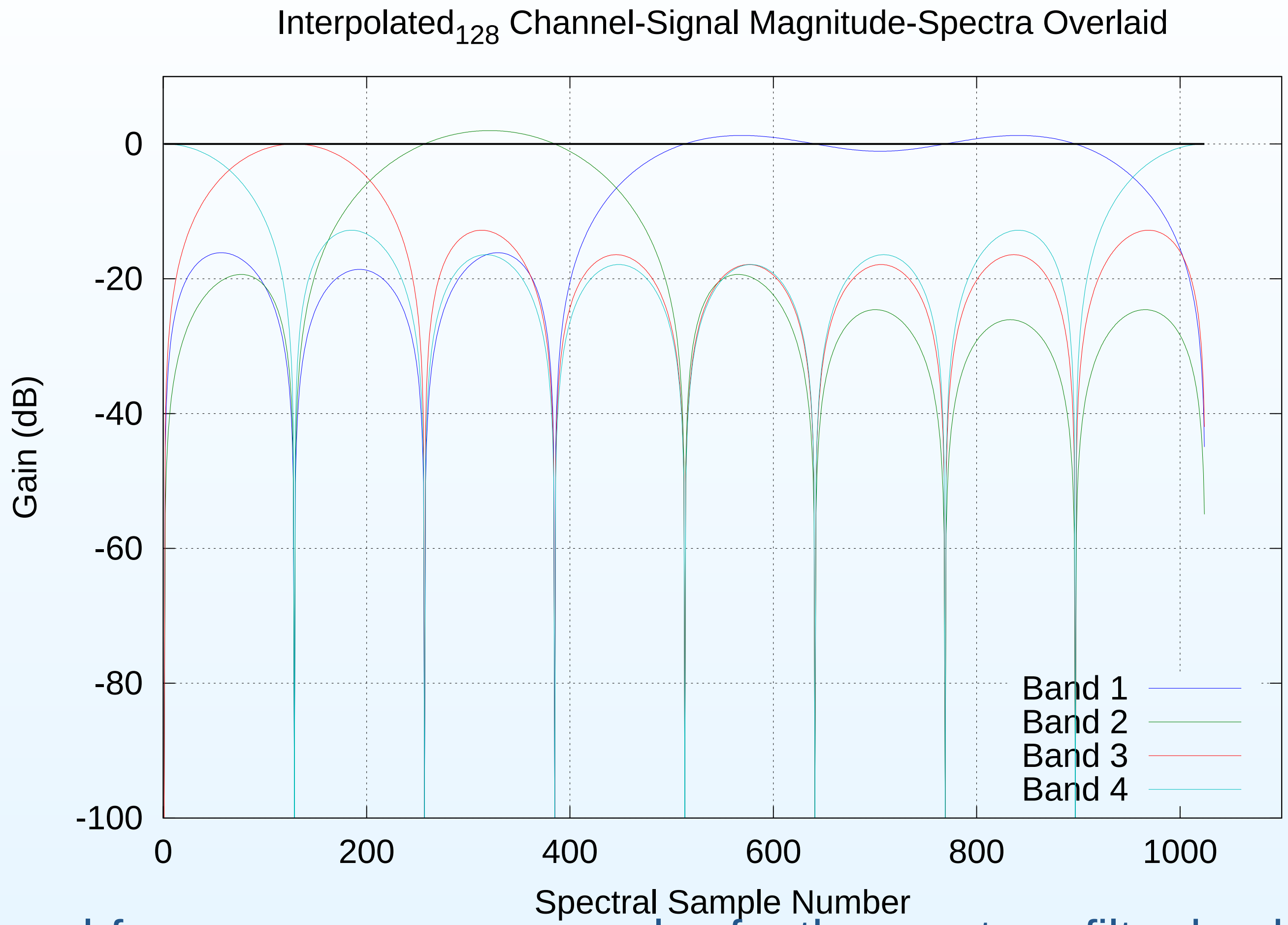
Simple octave filter bank for *complex* signals.

Basic Idea



- FFT implementation of one frame of simple octave filter bank
- Successive frames *non-overlapping* (rectangular window)

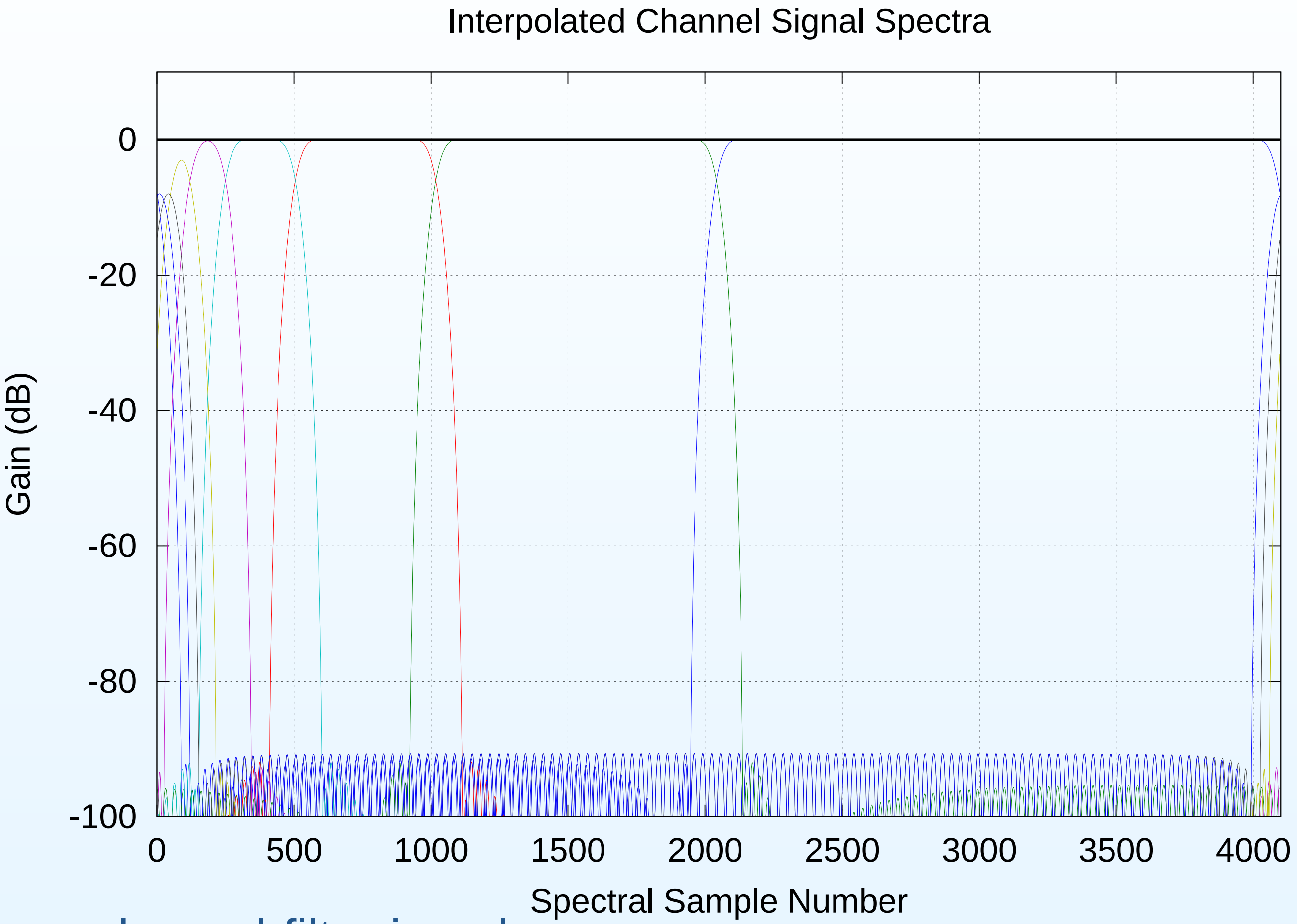
Interpolated Frequency Response



Channel-frequency-response overlay for three-octave filter bank

- Filter-bank driven by an impulse
- Zero-padded FFT taken for each channel signal
- Magnitude responses overlaid

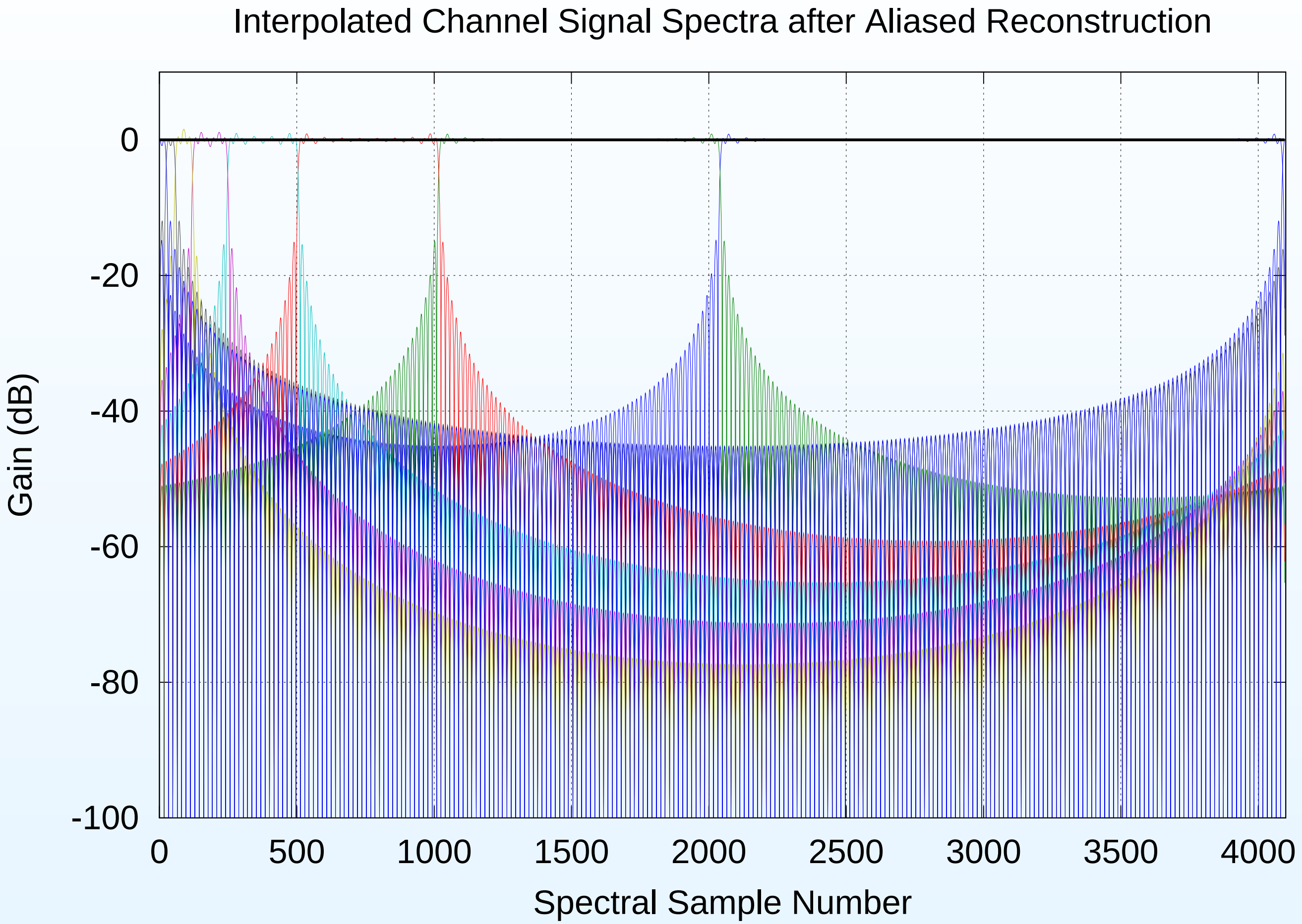
Practical Octave Filter Bank



Improve channel-filter impulse responses:

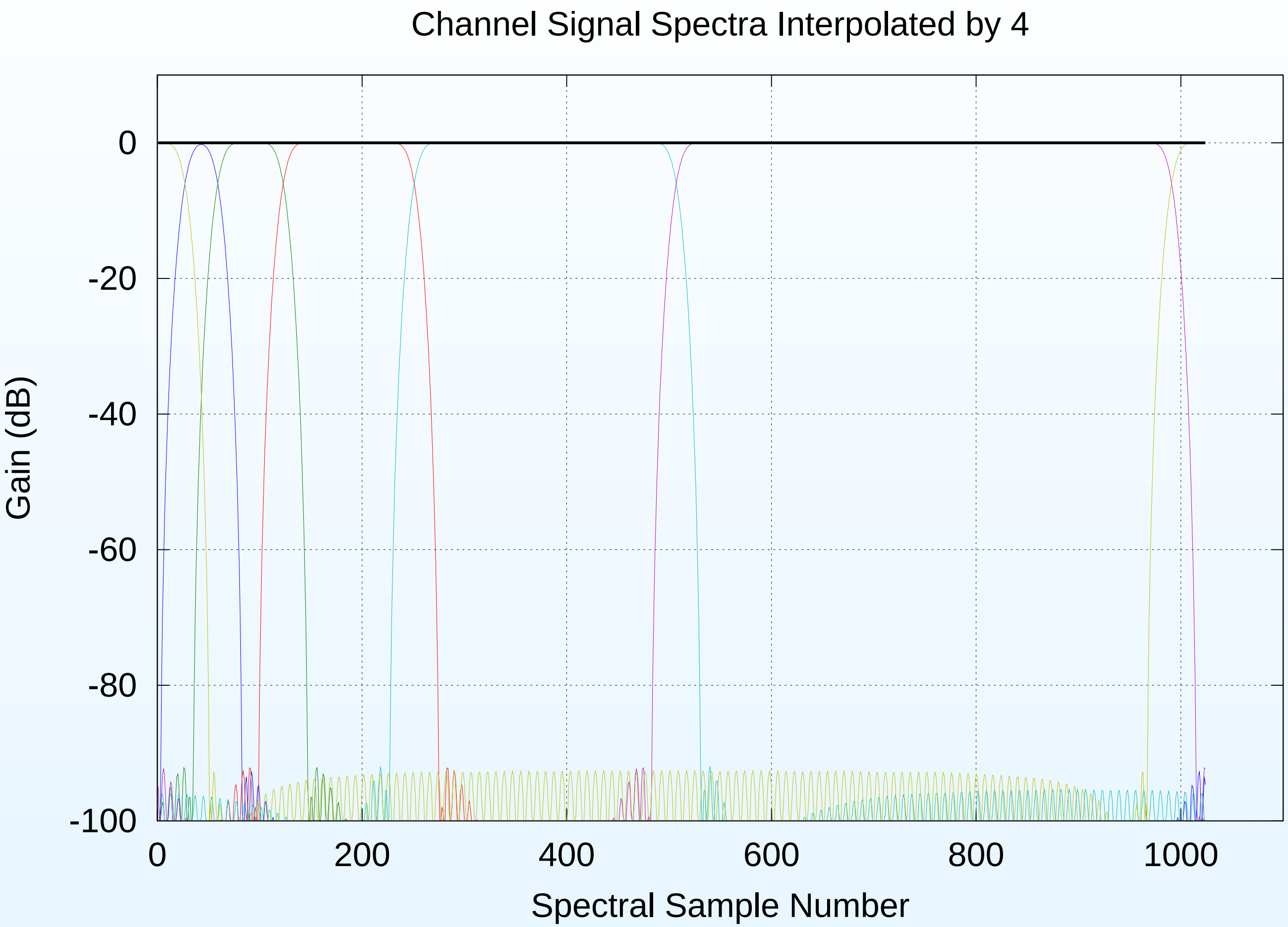
Rectangularly windowed sinusoids
⟶ Chebyshev-windowed sinusoids

Superposition of Channel Spectra after Critical Sampling



- Each channel maximally downsampled
- Transition bands alias heavily
- Aliasing cancels in filter-bank sum

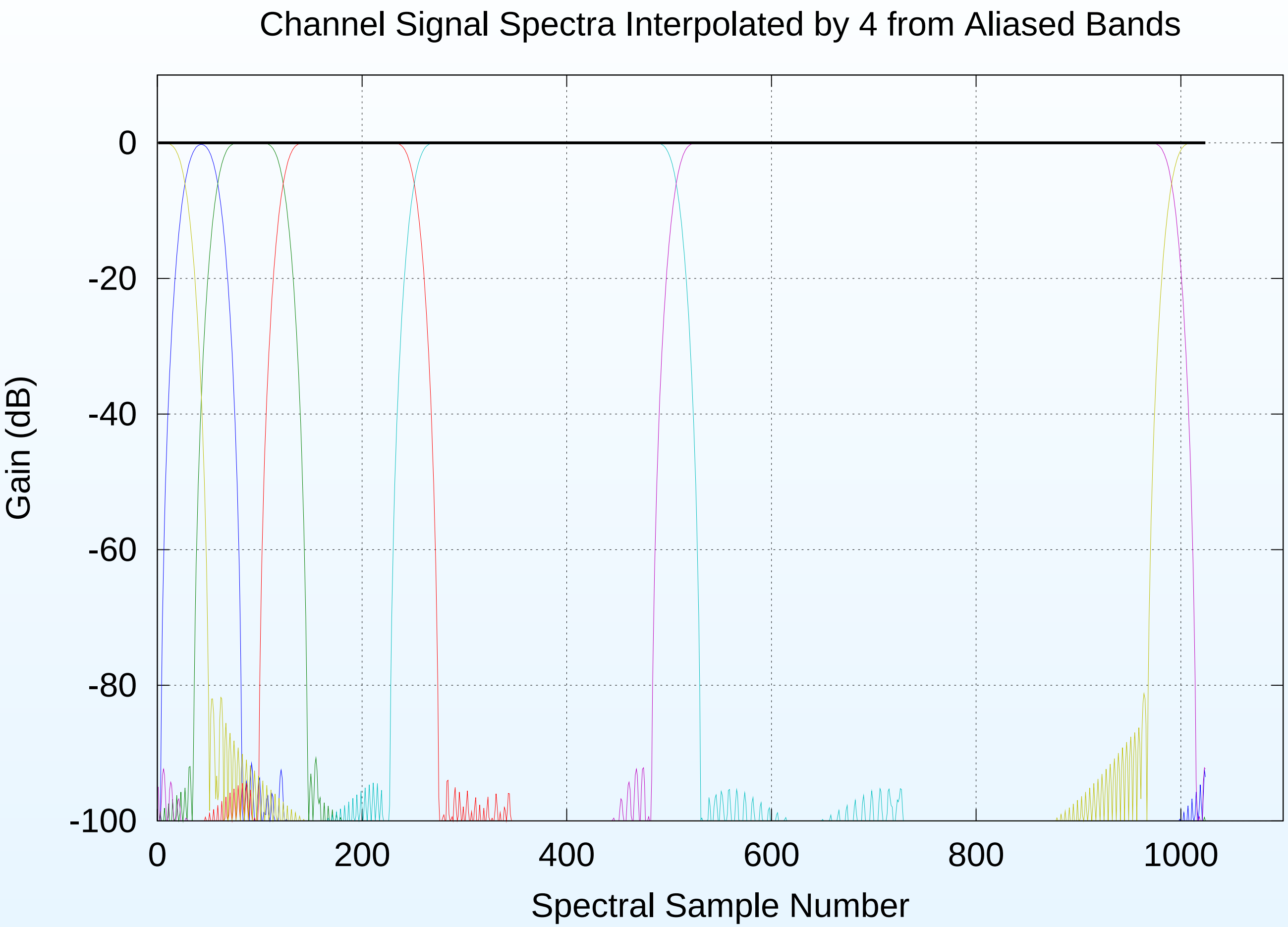
IFFT Band Allocation Including Transition Bands


Channel Signal Spectra Interpolated by 4

As before but allocating wider overlapping IFFT bands:

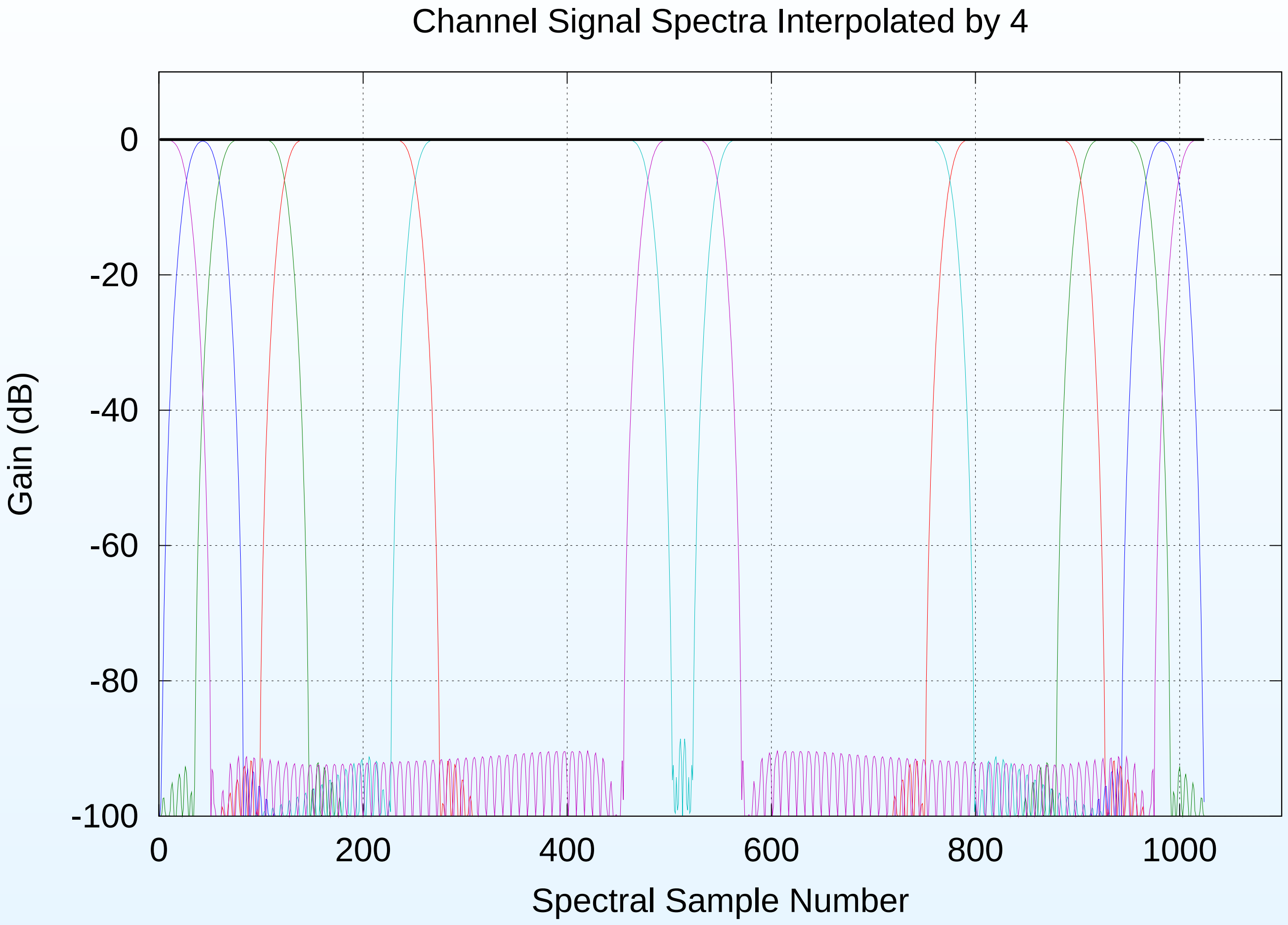
- Passband *plus* transitions fit inside IFFT
- Passbands contiguous as before, but IFFTs overlap more

Aliasing Suppression



- Reconstruction after critical downsampling
- Aliasing is reduced because now there is no aliasing of transition bands

Real Octave Filter Bank



Approximate octave filter bank for real signals

Notes on Audio FFT Filter Banks

- Arbitrary nonuniform spectral partitions and/or overlap-add decompositions are easily implemented, while preserving the FFT speed advantage

- Extension to time-varying nonuniform filter banks is straightforward

- Come see the poster this afternoon!

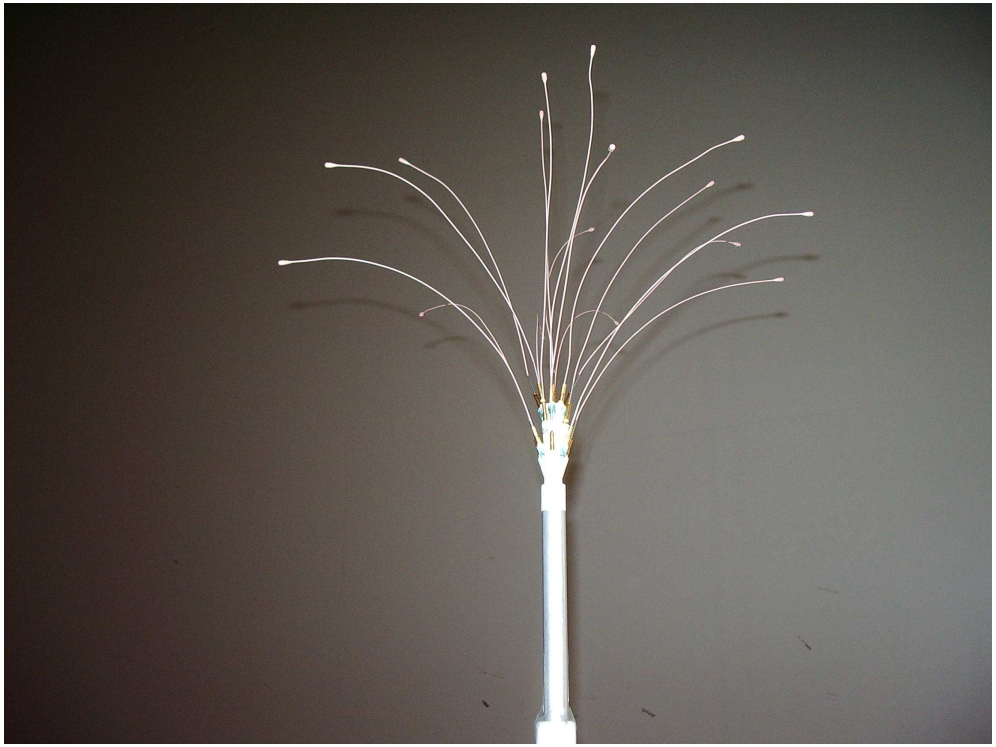
Spectrally Matched Click Synthesis

While outside the 1-year limit, Matt Wright’s poster this afternoon (4:30 pm) Spectrally Matched Click Synthesis is another work performed at least partially at CCRMA with JOS input.

- FIR filter design to achieve a minimum-duration “click” having a desired magnitude spectrum
- Applications:
  - Incremental attack strength modification
  - Continuous gradual “morphing” between an input sound and successively more impulsive/ percussive sounds

Acoustically Transparent and Configurable Microphone Array

Microphone Array



- Adustable geometry (software calibrated)
- Sixteen microphones (Countryman B6 Omni Lavalier):
  - 2 mm diameter capsules
  - 1 mm diameter flexible mounting wire
  - Acoustically transparent over most of the audio band

Upcoming Paper

“A Configurable Microphone Array with Acoustically Transparent Omnidirectional Elements”

Jonathan Abel, Nicholas Bryan, Travis Skare, Patty Huang, Darius Mostowfi, Miriam Kolar, and Julius Smith

AES-2009, New York

Current Application:

Recording and modeling acoustic properties of underground galleries at pre-Inca archeological site Chavín de Huántar in Peru

Flash Audio Plugins and Faust to ActionScript Conversion

Faust to ActionScript Conversion

CCRMA/EE graduate student **Travis Skare** developed a *Faust architecture file* for Flash browser plugins:
`http://ccrma.stanford.edu/~travissk/faustflash/`

- Faust generates C++ as usual
- Alchemy (by Adobe Labs) translates C++ to ActionScript
- Several Faust examples successfully compiled: `pitch-shifter`, `freeverb`, `karplus`, `osc`, `multibandfilter`
- Interesting points to note:
  - Flash version 10 needed for run-time sound processing
  - Delay from plugin controls to sound is about half a second
- Thanks to *Google* for allowing Travis to release his code as free software

Faust to ActionScript Conversion

CCRMA/EE graduate student **Travis Skare** developed a *Faust architecture file* for Flash browser plugins: `http://ccrma.stanford.edu/~travissk/faustflash/`

- Faust generates C++ as usual
- Alchemy (by Adobe Labs) translates C++ to ActionScript
- Several Faust examples successfully compiled: `pitch-shifter`, `freeverb`, `karplus`, `osc`, `multibandfilter`
- Interesting points to note:
  - Flash version 10 needed for run-time sound processing
  - Delay from plugin controls to sound is about half a second
- Thanks to *Google* for allowing Travis to release his code as free software

Faust to ActionScript Conversion

CCRMA/EE graduate student **Travis Skare** developed a *Faust architecture file* for Flash browser plugins: `http://ccrma.stanford.edu/~travissk/faustflash/`

- Faust generates C++ as usual
- Alchemy (by Adobe Labs) translates C++ to ActionScript
- Several Faust examples successfully compiled: `pitch-shifter`, `freeverb`, `karplus`, `osc`, `multibandfilter`
- Interesting points to note:
  - Flash version 10 needed for run-time sound processing
  - Delay from plugin controls to sound is about half a second
- Thanks to *Google* for allowing Travis to release his code as free software

Overview
Spring Reverb Models
Virtual Analog Circuits
Acoustic Guitar Models
Haptic Instruments
New Oscillators
Spectral Delay Filters
Audio FFT Filter Banks
Microphone Array
Faust to Flash Plugins
● Faust to Flash
Summary

Faust to ActionScript Conversion

CCRMA/EE graduate student **Travis Skare** developed a *Faust architecture file* for Flash browser plugins:
`http://ccrma.stanford.edu/~travissk/faustflash/`

- Faust generates C++ as usual
- Alchemy (by Adobe Labs) translates C++ to ActionScript
- Several Faust examples successfully compiled: `pitch-shifter`, `freeverb`, `karplus`, `osc`, `multibandfilter`
- Interesting points to note:
  - Flash version 10 needed for run-time sound processing
  - Delay from plugin controls to sound is about half a second
- Thanks to *Google* for allowing Travis to release his code as free software

Overview
Spring Reverb Models
Virtual Analog Circuits
Acoustic Guitar Models
Haptic Instruments
New Oscillators
Spectral Delay Filters
Audio FFT Filter Banks
Microphone Array
Faust to Flash Plugins
● Faust to Flash
Summary

Faust to ActionScript Conversion

CCRMA/EE graduate student **Travis Skare** developed a *Faust architecture file* for Flash browser plugins:
`http://ccrma.stanford.edu/~travissk/faustflash/`

- Faust generates C++ as usual
- Alchemy (by Adobe Labs) translates C++ to ActionScript
- Several Faust examples successfully compiled: `pitch-shifter`, `freeverb`, `karplus`, `osc`, `multibandfilter`
- Interesting points to note:
  - Flash version 10 needed for run-time sound processing
  - Delay from plugin controls to sound is about half a second
- Thanks to *Google* for allowing Travis to release his code as free software

Overview

Spring Reverb Models

Virtual Analog Circuits

Acoustic Guitar Models

Haptic Instruments

New Oscillators

Spectral Delay Filters

Audio FFT Filter Banks

Microphone Array

Faust to Flash Plugins
- Faust to Flash

Summary

Faust to ActionScript Conversion

CCRMA/EE graduate student **Travis Skare** developed a *Faust architecture file* for Flash browser plugins: `http://ccrma.stanford.edu/~travissk/faustflash/`

- Faust generates C++ as usual
- Alchemy (by Adobe Labs) translates C++ to ActionScript
- Several Faust examples successfully compiled: `pitch-shifter`, `freeverb`, `karplus`, `osc`, `multibandfilter`
- Interesting points to note:
  - Flash version 10 needed for run-time sound processing
  - Delay from plugin controls to sound is about half a second
- Thanks to *Google* for allowing Travis to release his code as free software

Faust to ActionScript Conversion

CCRMA/EE graduate student **Travis Skare** developed a *Faust architecture file* for Flash browser plugins:
`http://ccrma.stanford.edu/~travissk/faustflash/`

- Faust generates C++ as usual
- Alchemy (by Adobe Labs) translates C++ to ActionScript
- Several Faust examples successfully compiled: `pitch-shifter`, `freeverb`, `karplus`, `osc`, `multibandfilter`
- Interesting points to note:
  - Flash version 10 needed for run-time sound processing
  - Delay from plugin controls to sound is about half a second
- Thanks to *Google* for allowing Travis to release his code as free software

Summary

Summary

In summary, we took a quick look at some DAFx-Related Research involving JOS at CCRMA in 2008-2009:

- Spring Reverb Modeling — Jonathan Abel et al. — new propagation modes and calibration methods
- Digitizing Analog Circuits in Real Time — David Yeh — Automated K-Method for nonlinear analog circuits
- Coupled Strings Analysis and Synthesis — Nelson Lee — Fourth-order modes for low partials, wavuide model for upper partials; new analysis techniques
- Haptic Virtual Instruments — Ed Berdahl — Real controllers (with force feedback) for virtual instruments

Summary, continued

- Alias-Suppressed Virtual Analog — Vesa Välimäki et al. — differentiated higher-order polynomials suppress aliasing further than in the parabolic case (Välimäki 2005)
- Spectral Delay Filters — Vesa Välimäki et al. — “impulse response synthesis”
- Audio FFT Filter Banks — JOS — arbitrary nonuniform filter banks (spectral overlap-add decompositions) using overlapping IFFTs for each subband that include transition bands
- Microphone Array — Jonathan Abel et al. — Acoustically transparent, configurable, software-calibrated microphone array for sampling the 3D sound field
- Faust to Flash Plugins — Travis Skare — Tools for making Flash plugins from Faust source

Further Details on Differentiated Polynomial Synthesis

Maximum Smoothness Problem Formulation

For maximum wraparound smoothness, compute a_n, $n \in [1, n-2]$, such that

$$f^{(k)}(-1) = f^{(k)}(1)$$

for $k = 0, 1, \ldots, n-1$.

- These equations yield an *upper triangular system*
- Triangular matrix equations are easily “back-solved”
- Solution gives the *maximally flat coefficients* for $f(x)$

Even and Odd Polynomials

In general, $f(x)$ is a sum of its *even and odd parts*:

$$f(x) = f_e(x) + f_o(x)$$

where

$$\begin{aligned} f_e(-x) &= f_e(x) \triangleq \frac{f(x)+f(-x)}{2} \\ -f_o(-x) &= f_o(x) \triangleq \frac{f(x)-f(-x)}{2} \end{aligned}$$

- Even part $f_e(x)$ contains all *even powers* of x:

$$f_e(x) = \cdots + a_4x^4 + a_2x^2 + a_0$$

- Odd part $f_o(x)$ contains all *odd powers* of x:

$$f_o(x) = \cdots + a_5x^5 + a_3x^3 + a_1x$$

Solution Properties of Even and Odd Polynomials

We have

$$f(x) = f_e(x) + f_o(x)$$

where

$$\begin{aligned} f_e(x) &= \frac{f(x)+f(-x)}{2} \\ f_o(x) &= \frac{f(x)-f(-x)}{2} \end{aligned}$$

- Note that $f_e(-1) = f_e(1) \;\Rightarrow\;$ *smoothness constraint satisfied spontaneously* by even part
- Since $f_o(-1) = -f_o(1)$, we must have $f_o(1) = 0$
  $\Rightarrow$ *sum of f_o coefficients must be zero* in odd part

Simplified Maximum Smoothness Solution

- If $f(x)$ is an odd-order polynomial satisfying $f^{(k)}(-1) = f^{(k)}(1)$ for $k = 0, 1, \ldots, n-1$, then it *continues to satisfy those constraints when its even part is replaced by zero*
- Similarly, the odd part of an even-order polynomial $f(x)$ may be set to zero without affecting its wraparound smoothness
- Thus, without loss of generality, the starting polynomial $f(x)$ may be taken as even or odd, according its order
- The derivative of an even polynomial is odd, and vice versa
- Every other polynomial derivative has the sum-to-zero constraint
- Upper triangular system is reduced by about half