ADC20
AUDIO
DEVELOPER
CONFERENCE

A comparison of virtual analog modelling techniques

Jatin Chowdhury

About me…

- Musician, Electrical Engineer, Mixing/Mastering Engineer
- Studied audio DSP at CCRMA
- 5+ years of making audio plugins, DAWs, etc.
- Not a great guitarist (but I'm learning…)

Klon Centaur

Guitar pedal made by Bill Finnegan (MIT) from 1994-2000

Virtual Analog Modelling

Creating a digital emulation of a classic analog audio effects.

- Provide access to effects that are old or rare.
- Lower cost.
- Convenience.
- Improved understanding.

“White-Box” Modelling

Modelling a circuit through mathematical simulations of the physical interactions of the component parts.

- Nodal Analysis
- Modified Nodal Analysis (MNA)
- State-Space Formulation
- Wave Digital Filters (WDF)
- Port-Hamiltonian Formulation

“White-Box” Modelling

Advantages:

- Accurate modelling of circuit behaviour (even in extreme situations)
- Accurate modelling of control parameters
- Improved understanding of the modelled effect

Disadvantages:

- Often computationally expensive (especially for real-time use)
- Requires knowledge of DSP, as well as physics, circuit theory, etc.

“Black-Box” Modelling

Modelling a circuit by taking measurements, and designing a system to give a perceptually equivalent output.

- Convolution with Impulse Response (for linear systems)
- Volterra Series
- Weiner-Hammerstein Method
- Neural Networks

“Black-Box” Modelling

Advantages:

- Better for capturing “unique” behaviour
- Computationally cheaper
- Only requires background knowledge of DSP

Disadvantages:

- Difficult to include control parameters
- Minimal understanding of the effect being modelled

Different Platforms

Desktop Audio Plugin:

- Consumer-grade CPU
- Plenty of memory
- Have to share resources with other plugins

Embedded Device:

- Depends on the device (pedal, Eurorack module, multi-effects processor)
- More powerful processors are more expensive
- Limited memory
- (Usually) don’t have to share resources

Research Goals

- Model sub-circuits from the Klon Centaur using different modelling methods:
    - Nodal Analysis
    - Wave Digital Filters
    - Neural Networks

- Create desktop and embedded implementations of the modelled effect

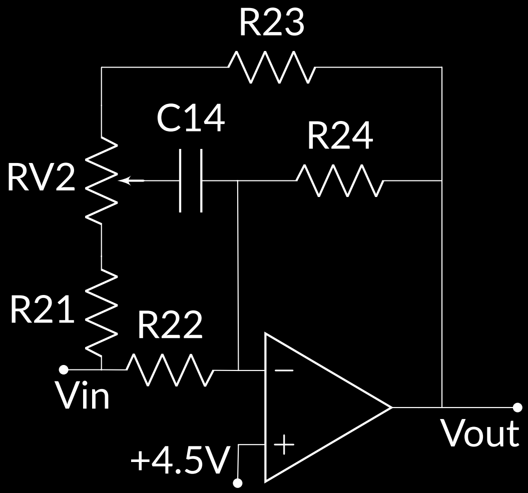
- Compare the advantages/disadvantages of each method

Outline

- Traditional Circuit Modelling
    - Nodal Analysis (Tone Stage sub-circuit)
    - Wave Digital Filters (FF-1 sub-circuit)
- Neural Network Circuit Modelling
    - Recurrent Neural Network (Gain Stage sub-circuit)
- Desktop and embedded implementations
- Comparisons and Results

Nodal Analysis

Example Circuit: Tone Stage



Nodal Analysis: Continuous Time

1. Convert the circuit to the Laplace Domain, using the Laplace variable $s = j\omega$. The complex impedance of each principal circuit component is defined as:

$$Z_R = R, \quad Z_C = \frac{1}{Cs}, \quad Z_L = Ls \tag{1}$$

Nodal Analysis: Continuous Time

2. Form the Laplace domain transfer function.[1]

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{C_{14}\left(\frac{1}{R22}+\frac{1}{R21+R_{v2b}}\right)s+\frac{1}{R22}\left(\frac{1}{R21+R_{v2b}}+\frac{1}{R23+R_{v2a}}\right)}{C_{14}\left(\frac{1}{R23+R_{v2a}}+\frac{1}{R24}\right)s+\frac{-1}{R24}\left(\frac{1}{R21+R_{v2b}}+\frac{1}{R23+R_{v2a}}\right)} \tag{2}$$

[1] Maby, *Solid State Electronic Circuits*.

Nodal Analysis: Discrete Time

3. Use a conformal map to map from the s-plane to z-plane (often the bilinear transform).[2]

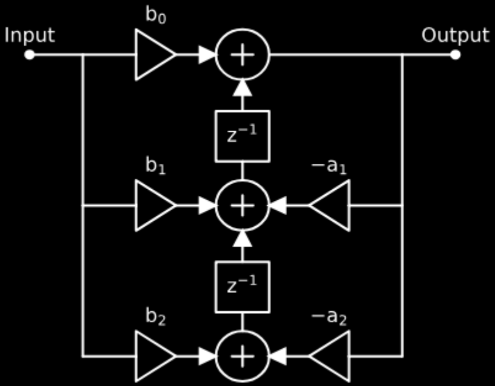
$$s \leftarrow \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}} \tag{3}$$

[2] Smith, *Physical Audio Signal Processing*.

Nodal Analysis: Discrete Time

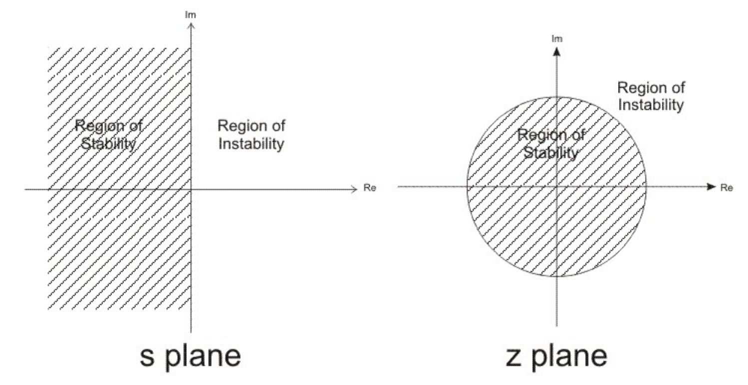
4. Implement the system as a digital filter.

$$y[n] = b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] - a_1 y[n-1] - a_2 y[n-2] \quad (4)$$

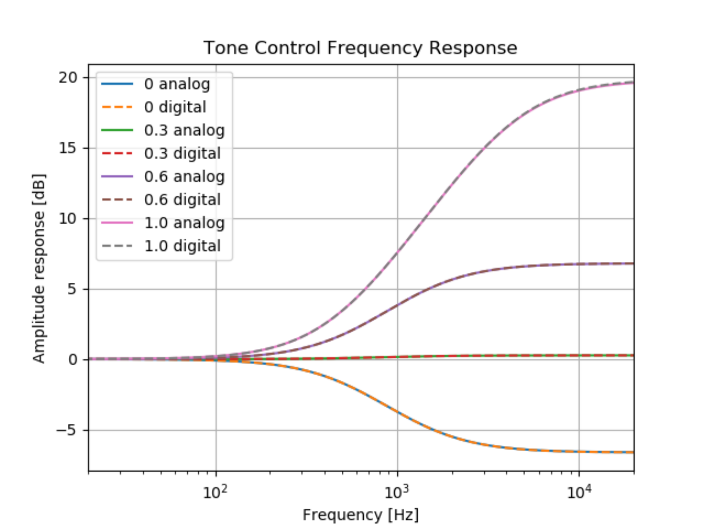


Discretization Considerations

- Frequency warping
- Stability



Tone Stage Frequency Response



Nodal Analysis

Advantages:

- Simple and computationally efficient circuit models

Disadvantages:

- Cannot be used to model nonlinear circuits (can be extended with Modified Nodal Analysis)
- Sometimes difficult to compute parameter changes

Wave Digital Filters

Kirchoff Domain Circuits

- Each circuit component has an impedance
- Each component has a voltage across its terminals and current between
- Components are connected in series/parallel configurations (usually)

Wave Domain Circuits

Circuits are made up of wave ports with incident and reflected waves.

Incident wave:

$$a = v + R_0 i \tag{5}$$

Reflected wave:

$$b = v - R_0 i \tag{6}$$

Wave Domain Circuits

- Each circuit component is a “1-port element” that inputs incident and outputs reflected wave variables
- Each series/parallel junction is an “N-port adaptor” that connects the 1-ports with a scattering junction
- Free parameter: port resistance

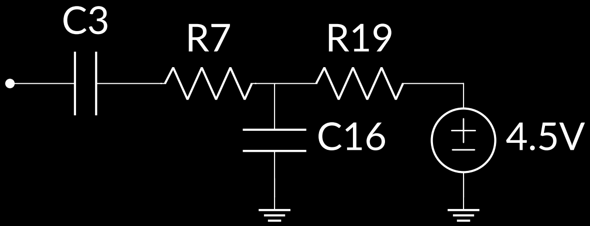
Wave Digital Filters

Wave Digital Filters (WDFs) were developed by Alfred Fettweis in the 1970’s and 80’s.[3]

- Digital simulation of circuits in the wave domain
- Discretize each circuit element independently
- Create binary connection tree (BCT) between circuit elements

[3] Fettweis, “Wave digital filters: Theory and practice”.

Example Circuit: Feed-Forward Network 1



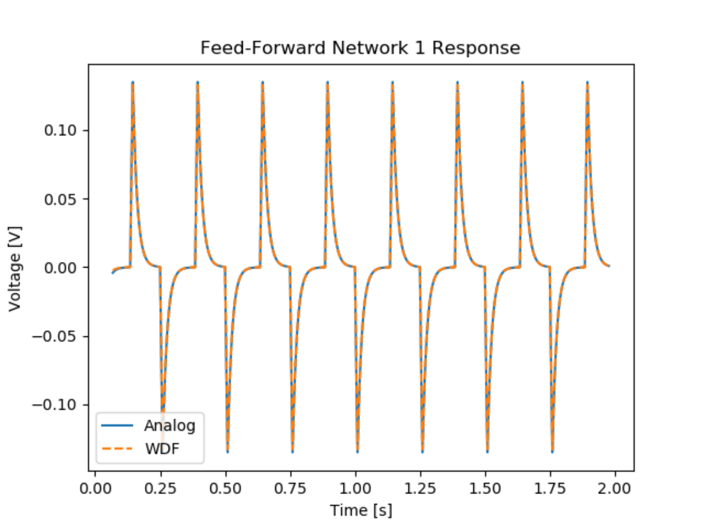
Klon Centaur Feed-Forward Network 1 Circuit

Example Circuit: Feed-Forward Network 1

$$\begin{tikzcd}
 & & & R_7 & & R_{19} \\
 & & \mathcal{S}_2 \arrow[ur] \arrow[dr] & & \mathcal{S}_3 \arrow[ur] \arrow[dr] & \\
V_{in} \arrow[r] & \mathcal{S}_1 \arrow[ur] \arrow[dr] & & \mathcal{P}_1 \arrow[ur] \arrow[dr] & & V_{4.5} \\
 & & C_3 & & C_{16} &
\end{tikzcd}$$

WDF tree for the Klon Centaur Feed-Forward Network 1 Circuit. $\mathcal{S}$ *and* $\mathcal{P}$ *nodes refer to series and parallel adaptors respectively.*

Time-Domain Response



Wave Digital Filters

Advantages:

- Modularity: circuit elements and topology can be altered on-the-fly
- Each element can be discretized with a different conformal map

Disadvantages:

- Cannot model circuits with multiple nonlinearities or $\mathcal{R}$-type topologies
- These types of circuits can be modelled using $\mathcal{R}$-adaptors, but with an increase in complexity

Wave Digital Filters

More information:

- Alfred Fettweis, "Wave Digital Filters: Theory and Practice", *Proceedings of the IEEE*, vol. 74, no. 2, 1986
  - Original reference for deriving WDF formalism
- Kurt Werner, *Virtual Analog Modeling of Audio Circuitry Using Wave Digital Filters*, PhD. Thesis, Stanford University, 2016
  - Great reference for deriving WDFs, including more recent advancements
  - Expands WDFs to handle $\mathcal{R}$-type topologies and multiple nonlinearities

Wave Digital Filters

More information:

- François Germain, *Non-oversampled physical modeling for virtual analog simulation*, PhD. Thesis, Stanford University, 2019
    - Example of independently discretizing circuit elements with Alpha Transform
- Jingjie Zhang and Julius Smith, “Real-time Wave Digital Simulation of Cascaded Vacuum Tube Amplifiers Using Modified Blockwise Method”, *Proc. of the 21st International Conference on Digital Audio Effects*, 2018
    - Real-time simulation of an impressively large circuit

Real-Time Neural Networks

Black Box Modelling with Neural Nets

Previous work: Damskägg et al., 2019[4]

- Uses a WaveNet-style, "Temporal Convolutional Network"
- Used to model distortion pedal circuits
- Also used to model tube amp distortion[5]
- Disadvantage: computationally expensive

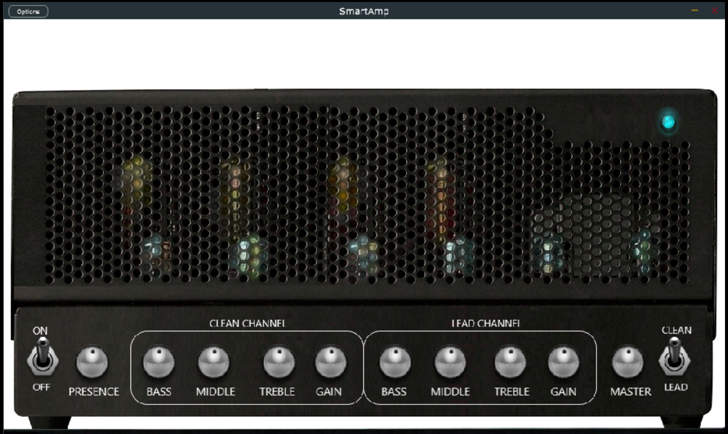
---

[4] Damskägg, Juvela, and Välimäki, "Real-Time Modeling of Audio Distortion Circuits with Deep Learning".

[5] Damskägg et al., *Deep Learning for Tube Amplifier Emulation*.

Temporal Convolutional Networks

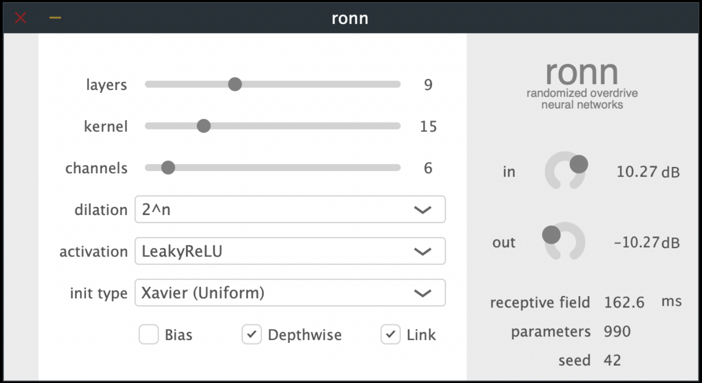
Keith Bloemer: Smart Guitar Amp[6]



[6] https://github.com/keyth72/SmartGuitarAmp

Temporal Convolutional Networks

Christian Steinmetz: Randomized Overdrive Neural Networks[7]



[7] `https://github.com/csteinmetz1/ronn`

Black Box Modelling with Neural Nets

Previous work: Parker et al., 2019[8]

- Uses a deep, fully-connected "State Transition Network"
- Approximates a state-space solution for nonlinear distortion and filter circuits
- Effectively a "grey-box" model

8 Parker, Esqueda, and Bergner, "Modelling of Nonlinear State-Space Systems Using a Deep Neural Network".

State Transition Networks

Native Instruments: Guitar Rig 6 Pro[9]

[9] https://blog.native-instruments.com/the-making-of-icm/

Black Box Modelling with Neural Nets

Previous work: Wright et al., 2019[10]

- Uses a single layer recurrent neural network
- Used to model guitar distortion circuits
- Can also be used to model time-varying circuits[11]

[10]Wright, Damskägg, and Välimäki, “Real-Time Black-Box Modelling with Recurrent Neural Networks”.

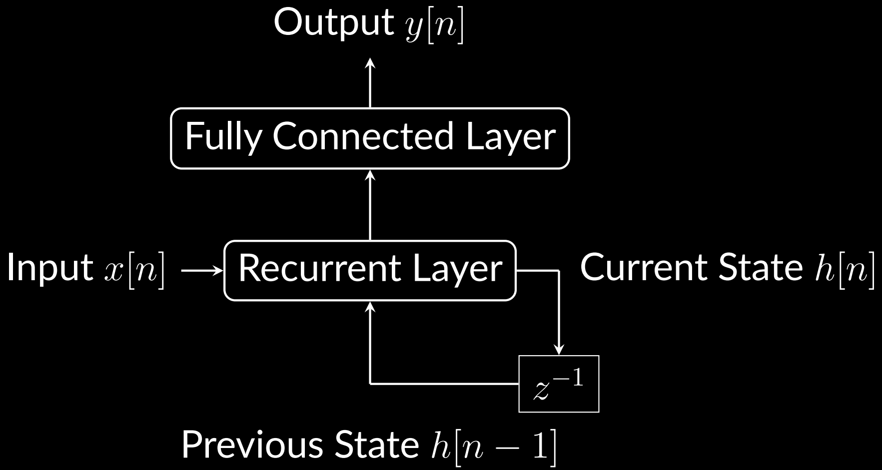
[11]Wright and Välimäki, “Neural Modelling of Time-Varying Effects”.

Recurrent Neural Network

Advantages of using RNNs to model distortion circuits:

- Makes sense (recurrent units can be distortion effects)
- Computationally efficient
- Can include circuit control parameters

Recurrent Neural Network



Recurrent Neural Network

Recurrent layer: Gated Recurrent Unit

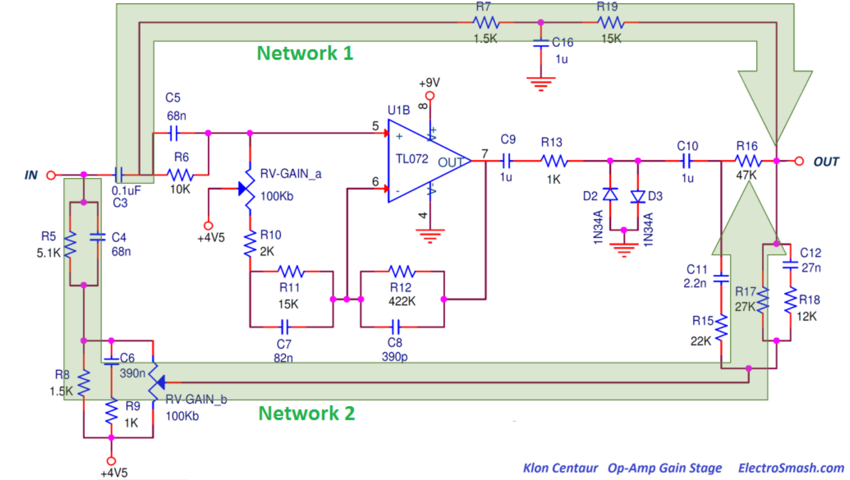
$$z[n] = \sigma(W_z x[n] + U_z h[n-1] + b_z) \tag{7}$$

$$r[n] = \sigma(W_r x[n] + U_r h[n-1] + b_r) \tag{8}$$

$$c[n] = \tanh(W_c x[n] + r[n] \circ U_c h[n-1] + b_c) \tag{9}$$

$$h[n] = z[n] \circ h[n-1] + (1 - z[n]) \circ c[n] \tag{10}$$

Example Circuit: Centaur Gain Stage



Recurrent Neural Network: Training

Training Data:

- ~ 4 minutes of guitar recordings (direct) at 44.1 kHz
- Split into 0.5 second segments
- 400 training samples, 25 validation samples
- Simulated Klon output using SPICE
- 5 positions of “Gain” parameter

Loss Function: Error-to-Signal Ratio

$$\mathcal{E}_{ESR} = \frac{\sum_{n=0}^{N-1} |y[n] - \hat{y}[n]|^2}{\sum_{n=0}^{N-1} |y[n]|^2} \tag{11}$$

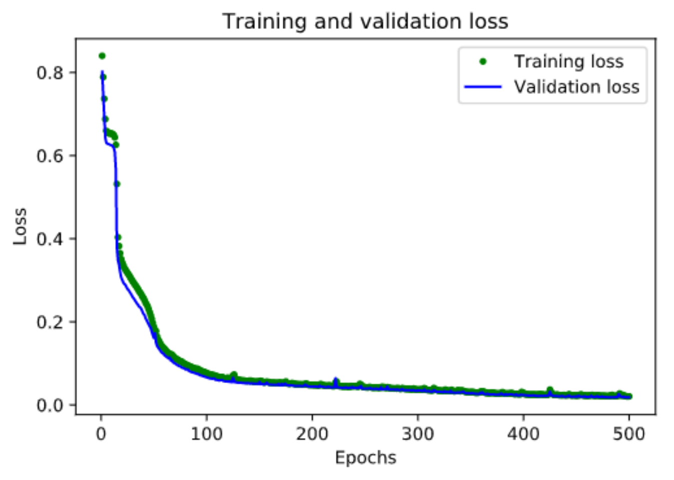
Recurrent Neural Network: Control Parameters

In training, we were unable to successfully train a network that included the “Gain” parameter.

Instead, we trained 5 independent networks, one for each “Gain” knob position. In the final implementation, we “fade” between the models in real-time.

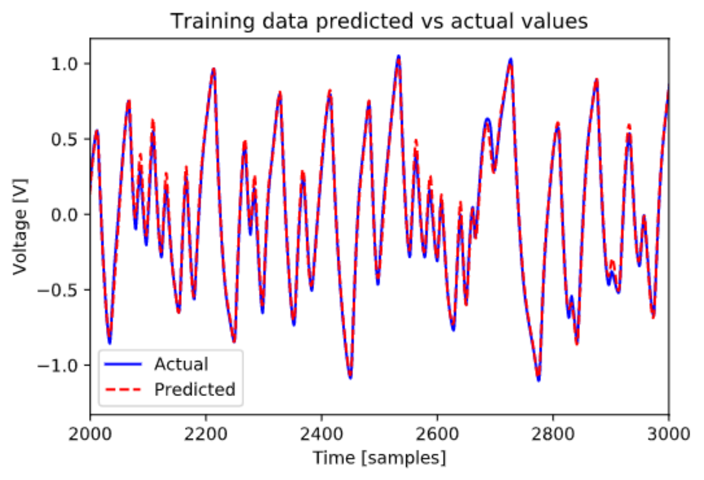
Recurrent Neural Network: Training

Training: 500 epochs, ~ 8 hours



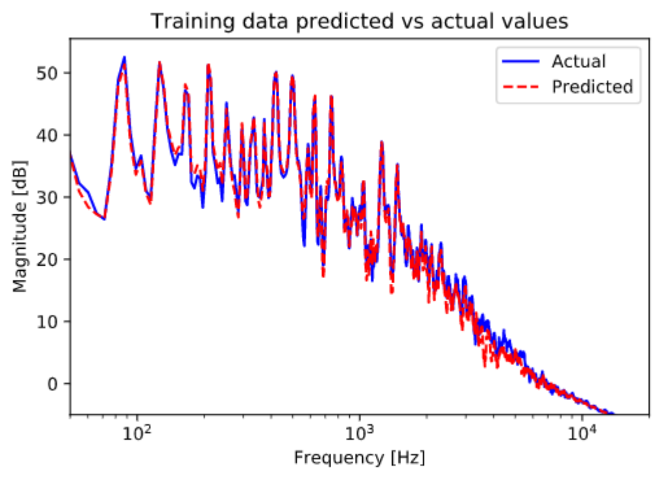
Recurrent Neural Network: Training

Training results (time domain)



Recurrent Neural Network: Training

Training results (frequency domain)



Recurrent Neural Networks

Advantages:

- Efficient black-box modelling technique for distortion circuits
- Can potentially include control parameters

Disadvantages:

- Large networks can be computationally expensive
- Must be used at the same sample rate as training data
- Can be difficult to train with control parameters

Neural Networks: Future Work

Computational Efficiency

- Dense, recurrent, and convolutional layers often require nonlinear activation functions, like $\tanh$
- In DSP, we often use fast approximations, or look-up tables
- Can we use function approximations in neural networks?
  - Is it better to train with approximations, or train with full precision, and use approximations for real-time implementation?
  - Similar to questions in TinyML about weight quantization

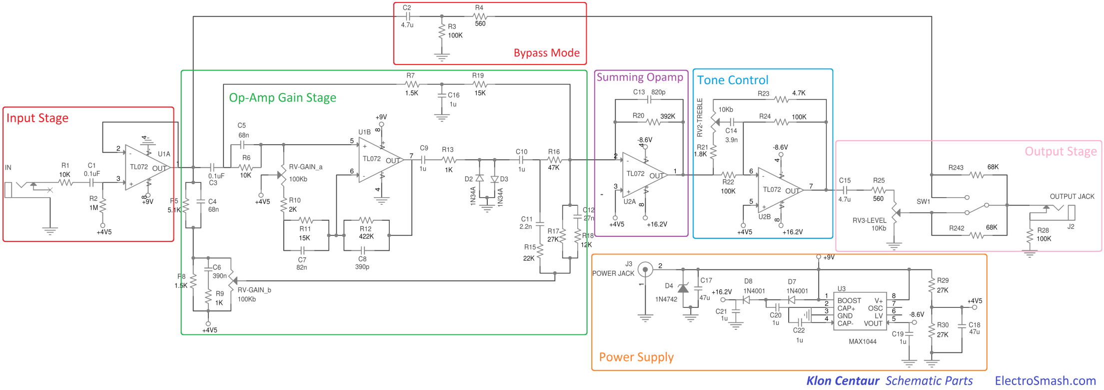
Neural Networks: Future Work

Sample Rate

- Currently most networks must be used at the same sample rate as the training data
- Can one network to be used for a range of sample rates?
  - Sample rate as input?
  - Transform network weights?
  - Fractional delay (RNN only)?
- What about aliasing?

Real-Time Implementation

Klon Centaur Circuit Schematic



Implementation

Non-ML Implementation

- Use a combination nodal analysis, WDFs
- Control parameters for Treble, Gain, Level

ML Implementation

- RNN model for Gain Stage, nodal analysis elsewhere
- Fade between models for variable Gain control
- Custom GRU and Dense layer implementations in C++

RNN Inferencing Engine

Tensorflow Lite

- Converts a Tensorflow model to a format that can be run on embedded devices

- Support for GRUs is still experimental

- Real-time audio concerns: no thread locks, no memory allocation on real-time audio thread

RNN Inferencing Engine

Custom engine: Eigen

- Eigen is a linear algebra C++ library with SIMD support for matrix/vector operations

- Custom implementations of GRU and fully connected layers, validated against Tensorflow

- Can be difficult to compile on embedded devices

RNN Inferencing Engine

Custom engine: C++ STL

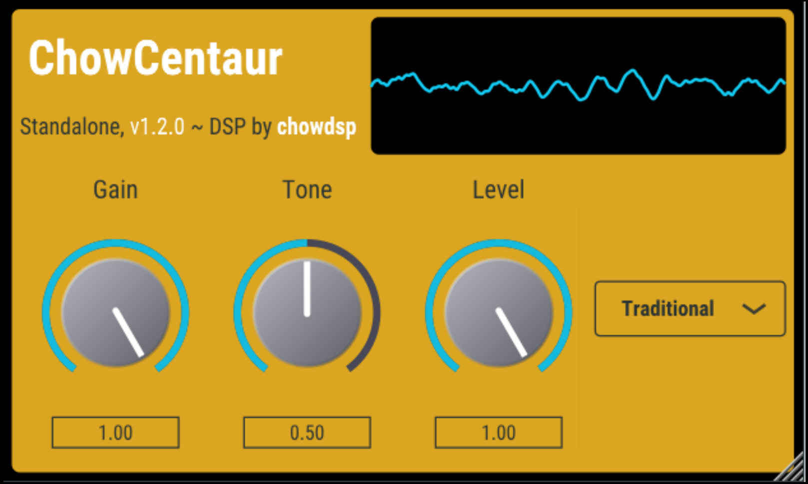
- Optimized algorithms for operations such as std :: inner_product

- Custom implementations of GRU and fully connected layers, validated against Tensorflow

- Can be compiled on most embedded devices

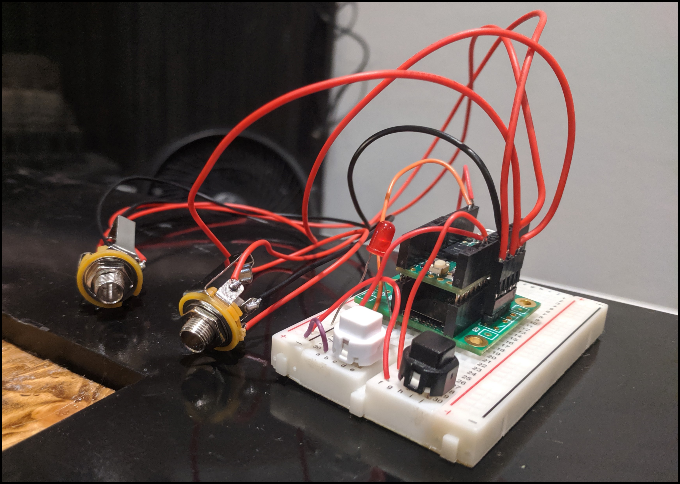
Implementation

Desktop Audio Plugin (JUCE/C++)



Implementation

Teensy 4.0, Teensy Audio Shield, Teensy Audio Library



Results: Performance

Compute time per second of audio.

Block Size	NonML Speed	ML Speed
8	0.0723437	0.0528792
16	0.0703079	0.0510437
32	0.0652856	0.0511147
64	0.0662835	0.0502434
128	0.0666593	0.0495194
256	0.0696844	0.0480298
512	0.0669037	0.0477946
1024	0.060816	0.0488841
2048	0.0695175	0.0488309
4096	0.0623839	0.0472191

Results: Summary

- Subjectively, non-ML and ML models sound very similar.
- ML model has slightly damped high frequency response, (not a big deal on guitar input; more noticeable on other audio).
- ML model is more efficient!

Takeaways

- 3 methods for modelling circuits:
  - Nodal Analysis (simplest)
  - Wave Digital Filters (modular)
  - Neural Networks (experimental)
- Modelling circuits with neural networks can be done, but more research/experimentation is needed
- Desktop vs. Embedded:
  - Memory management
  - Processing power (floating point processing, SIMD)
  - Price

Links

- Technical Paper
- Source Code (and plugin download)
- Video Demos

Acknowledgements

- Pete Warden and the EE 292D class, for insipiring this project
- Julius Smith, Kurt Werner, and Jingjie Zhang, for assistance with WDFs

Thank You!

ADC20

AUDIO DEVELOPER CONFERENCE