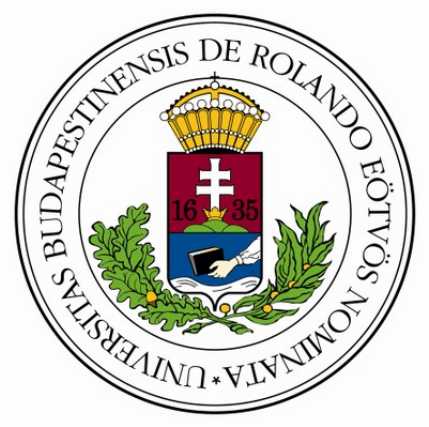
Introduction to the mathematics of Virtual Analog modeling

BSc Thesis in Applied Mathematics

Ték Róbert Máté

Supervisor: Dr. Gergó Lajos
Department of Numerical Analysis



Eötvös Loránd University
Faculty of Science
2025

to Anna

Contents

Introduction

Many different effects pedals and guitar amplifiers exist for electric guitar players to choose from. These devices originate from the mid-20th century, and have had great influence on decades of popular music. They traditionally employ analog circuitry to amplify and process the electric guitar's signal in various ways. Creating digital emulations of such electrical circuits is highly desirable. Analog circuits can be damaged by mechanical impacts, humidity or temperature changes. They degrade over time and require occasional maintenance. Guitar amplifiers are notoriously heavy, and thus their transportation can be a challenge. Some guitar amplifiers need to operate at very high volumes to sound best, which is not always feasible e.g. in an apartment or at night. A handful of digital devices can replace entire guitar rigs and require less maintenance while enabling features that would be impossible or very cumbersome to implement with traditional gear. Every analog circuit is unique and has a unique sound e.g. due to availability of electrical components, manufacturing tolerances or other imperfections. Vintage devices may no longer be in production. Therefore it is important to capture and preserve the tonality of these devices with careful attention to detail for future generations of musicians. The process of creating digital emulations of analog audio circuitry is often referred to as *Virtual Analog* modeling. The goal of this thesis is to offer a glimpse into the mathematical foundation of Virtual Analog modeling through a concrete example; deriving a real-time software emulation of the Ibanez TS808 Tube Screamer distortion pedal. The source codes for the emulation, and other related programs, are available on GitHub [1].

This work builds upon established results from relevant scientific disciplines such as physics or electrical engineering. While these results are rooted in deeper, more fundamental theories, their precise introduction is beyond the scope of this thesis. Instead, simplified yet mathematically coherent definitions will be used. For a more rigorous treatment of the underlying theories, the reader is referred to dedicated textbooks, such as [2, 3].

Though I tried my best to present relevant and up-to-date information on the topic, this thesis is not intended to be a guide to the best practices of digital signal processing.

All product names and trademarks are the property of their respective owners. Their use herein does not imply affiliation or endorsement.

Chapter 1

The Ibanez TS808 Circuit

The Ibanez TS808 is a so-called *overdrive pedal* designed to emulate the soft clipping of overdriven tube amplifiers. An amplifier is said to be overdriven when it is pushed beyond its ideal range of operation. For vacuum tube amplifiers, this produces a smooth, musical sounding distortion, which many listeners—and guitarists—find pleasing. Overdrive pedals, like the TS808, aim to recreate this tonal character at lower volumes, hence the name *Tube Screamer*.

The TS808 circuit can be thought of as the interconnection of several distinct electrical networks:

- input buffer,
- distortion stage,
- tone/volume stage,
- effects bypass switching,
- power supply, and
- output buffer.

Our primary concern will be toward the distortion and the tone/volume stages, which contribute most to the tonality of the pedal, while the other networks are of utilitarian nature and have little effect on the processed signal [4]. The schematic for these stages can be seen in Appendix A.

The clipping stage consists of an operational amplifier—or *op-amp* for short—in a non-inverting configuration, with some passive filtering and two antiparallel clipping diodes in the feedback loop. The resistor R_g and capacitor C_g form a simple high-pass filter, while the resistance R_f and capacitor C_f form a variable-frequency low-pass filter to "shape the amount of clipping and the frequency at which it occurs" [4]. This network is sometimes referred to as a *diode clipper with embedded low-pass filter*. In the real pedal, the resistance R_f actually consists of a 51kΩ resistor and a 0 to 500kΩ variable resistor—or *potentiometer*—in series, but, for the sake of simplicity, we can think of them as a single variable resistance.

The clipping stage feeds directly into the tone/volume stage, which consists of a fixed first-order low-pass filter, followed by an active tone control network centered around the op-amp labeled as 'OpTone'. Here, another potentiometer can be used to adjust the amount of high frequencies passed through to the output. Depending on the position of this potentiometer, the overall effect of the tone circuit can resemble:

- a second-order low-pass filter, attenuating high frequencies even more (potentiometer all the way to the 'bass' side), or
- a band-pass filter, which lets through more of the high frequencies (potentiometer all the way to the 'treble' side), or
- somewhere in-between [5].

The output level control at the end of this stage is just a simple voltage divider.

To create a real-time digital emulation of the TS808 circuit, we need to understand

a) the fundamentals of discrete-time signal processing,

b) how the components of the circuit interact to process an analog input signal, and

c) how to translate these interactions to discrete-time systems.

Circuit theory and *modified nodal analysis* can be used to understand *b*), while *a*) is answered by the theory of *discrete-time signals and systems*. Point *c*), however, is less straightforward. There are many different ways to approach this problem, all of them equally valid, and come with different strengths and weaknesses. The process described by *c*) is called *discretization*. In this thesis, we will look at several different discretization strategies as well as their mathematical background.

The clipping stage will be modeled using *white-box modeling*, that is, we have full knowledge of the reference circuit and its workings. The tone stage will be modeled using *black-box modeling*, where we will pretend that only the input-output relationship of the circuit is known to us.

Chapter 2

Understanding electrical circuits

To derive a software emulation of the Ibanez TS808 pedal, we need a mathematical framework for modeling electrical circuits. To make this thesis self-contained, this chapter outlines the basic theoretical elements of electrical circuits relevant to our digital modeling task. As such, readers not yet familiar with electrical systems shall be able to follow the material presented in this work.

2.1 Methodology

> *With thermodynamics, one can calculate almost everything crudely; with kinetic theory, one can calculate fewer things, but more accurately; and with statistical mechanics, one can calculate almost nothing exactly.*
>
> *Wigner Jenő Pál*

Electricity at a macroscopic scale is fully described by *classical electrodynamics*, however, direct application of its principles would be quite impractical for our purposes [2]. To bring forth more tractable mathematical models of electrical circuits, we must make certain simplifying assumptions, such as ideal operating conditions and perfectly manufactured eletrical components. Regardless, the resulting models have proven to be sufficiently accurate in predicting the behavior of many real-world circuits [2, 6].

2.2 Circuit Theory

Circuit theory is a special case of electromagnetic field theory with a focus on electrical networks. Circuit theory relies on the so-called *lumped circuit model*, which, together with the laws of electromagnetic field theory, result in a set of equations greatly reduced in complexity.

The following definition should not be considered ground thruth; it is merely my attempt at interpreting and formalizing the notion of lumped electrical networks.

Definition 1. The lumped circuit model assumes the following for an electrical network:

i) the components of the network are physically distinct and are connected only by wires,

ii) all conducting materials outside of components are ideal conductors, i.e. they possess no electrical resistance,

iii) electrical and magnetic energy is stored or converted to other forms of energy only within electrical components, i.e. no electric fields exist outside components and wires,

iv) there is no magnetic coupling between components of the network,

v) electrical effects happen instantaneously throughout the network,

vi) the net electric charge on every component of the network remains zero at all times.

A physical circuit may be considered lumped if said assumptions can be in any way *justified*, i.e. it can be theoretically or experimentally verified that the impact of the described phenomena on the circuit are *negligible*.

The above assumptions can be justified for analog audio circuits. Hence, from now on, we will consider electrical networks and components to be lumped. It follows that the lumped circuit model is concerned only with the *topology* of a network, and not its physical layout.

Definition 2. An *electrical component* is a device with two or more distinct conducting surfaces called *terminals*. Terminals act as connection points. Components may be connected via conducting *wires*. An *electrical network* is the interconnection of electrical components. A maximal set of connected terminals within a network is called a *node*.

The notion of *electrical circuits* is a bit more nuanced, and will be defined shortly. The schematic symbols of components of interest for this thesis can be seen in Appendix A.

It may be tempting to visualize the topology of an electrical network as a graph (V, E), with V corresponding to components, and E corresponding to connections or wires. Indeed, circuit schematics do resemble such a graph, with a single difference; the wires of a circuit node are united into a single hyperedge. Some examples can be found in Appendix A. However, it is more useful to define the topology of a network as the *dual* of said hypergraph, in which graph nodes correspond to circuit nodes and hyperedges correspond to components (or rather, the terminals of a component), as this view more closely resembles *flow networks* known from classical graph theory.

The lumped circuit model allows us to define *voltage* and *current* in the following manner:

Definition 3. We define voltage as the real-valued quantity that can be measured by an ideal voltmeter between two nodes or terminals. The voltage between measurement points a and b is denoted by $v_{a,b}$ and is measured in *Volts*. We define current as the real-valued quantity that can be measured by an ideal ammeter through a terminal. The current through terminal p is denoted by i_p and is measured in *Amperes*.

It is customary to appoint a node of an electrical network to be the so-called *ground node*, which acts as a universal basis for voltage measurements. This node is usually chosen to contain one terminal of a voltage source. A voltage source—e.g. electric guitar pickups—is a 2-terminal electrical device that generates a voltage across its terminals. Hence it is easy to see how the notion of a ground node is useful to us.

Definition 4. Let g denote the ground node within an electrical network. Let a denote a voltage measurement point. The voltage $v_{g,a}$ is simply referred to as the voltage in a.

These practical definitions suffice for the analysis of analog audio circuits without delving into the underlying electromagnetic theory. It should be mentioned that current measurements are *directional*, that is, one should always state the direction of measurement (*in* or *out* of the terminal). In practice, we will assign a measurement direction to all terminals when a network is introduced, thus the direction of measurement need not be indicated in subsequent mathematical formulae.

Axiom 1. *Let a, b, c be measurement points. The following equations hold for voltage measurements:*

$$\begin{aligned} v_{a,b} &= -v_{b,a}, \\ v_{a,a} &= 0, \\ v_{a,c} &= v_{a,b} + v_{b,c}. \end{aligned} \tag{2.1}$$

Axiom 2. *Let i denote the current flowing into terminal p. The current flowing out of terminal p is equal to $-i$.*

Definition 5. Let $p \neq q$ be terminals of an electrical component. If the current flowing into p is equal to the current flowing out of q at all times, then the unordered pair (p, q) is called a *branch*. Suppose that the measurement direction $\overrightarrow{pq}$ was affixed to the branch. The current flowing into terminal p (or equivalently, the current flowing out of terminal q) is called the *branch current*. The voltage $v_{q,p}$ is called the *branch voltage*.

In this work, we will only need to deal with two-terminal components (resistors, capacitors, diodes) that form a branch, and *operational amplifiers*, which we will discuss shortly.

Definition 6. An *electrical circuit* is an electrical network that contains at least one closed loop made of branches, through which electrical current may flow.

It is important to note that we often use the term *circuit* to refer to networks that contain no closed loops, but otherwise form a single *logical unit*. For example, the input network of a guitar effects pedal is an incomplete circuit that becomes complete only when e.g. an electric guitar's circuitry is connected.

In Circuit Theory, if a component has a branch between two of its terminals, then it is customary to define a so-called *branch constitutive equation* that relates the branch voltage to the branch current in the time domain. The graphs of such equations are sometimes referred to as *voltage to current curves* or *I-V curves*. The branch constitutive equations of components of interest can be found in Appendix A.

An ideal operational amplifier is characterized by an infinite input impedance, that is, the current flowing through its input terminals is zero. Furthermore, an ideal operational amplifier in a negative feedback configuration—like the 'OpClip' op-amp in the TS808 diode clipper circuit—will adjust its output voltage in such a way that the resulting voltage at its inverting input terminal is equal to the voltage seen by the operational amplifier at its non-inverting input terminal [2], given that this voltage falls into the operational range of the op-amp, and the feedback network is well-designed. This 'rule' is known as the *virtual short* principle. To achieve this, an ideal op-amp will draw as much current as necessary, which is equivalent to saying that the output impedance of an ideal op-amp is zero. The operational range of an op-amp is determined by the voltages at its power supply terminals. In our case, both op-amps operate in the 0V to 9V range. To be able to process an AC signal that falls within the -4.5V to 4.5V range, the signal is *biased* with 4.5V DC through a 10kΩ resistor. This DC offset will be 'removed' at the end of the tone/volume stage by a 1μF capacitor.

Voltages and currents in a lumped electrical network are governed by the following laws:

Axiom 3. (Kirchhoff's current law, KCL) *In a lumped network, the algebraic sum of branch currents through any node p is equal to* 0*. With mathematical notation:*

$$\sum_{r \in p} i_r = 0, \tag{2.2}$$

where the direction of current measurement for all $r \in p$ terminals are aligned, that is, into or out of node p.

Axiom 4. (Kirchhoff's voltage law, KVL) *In a lumped network, the algebraic sum of branch voltages around any closed loop C equals* 0*, that is, all branch voltage measurements are aligned in the same direction along loop C.*

2.3 Mathematical model of the clipping stage

The amount of distortion produced by the clipping stage can be adjusted using the 'OVERDRIVE' knob—or *drive knob* for short—on the physical pedal. The drive knob controls a variable resistrance that is embodied by the resistor R_f in our model.

The input signal enters the clipping stage at the node labeled 'IN' in Appendix A. The signal is then biased with a DC voltage of 4.5V.

Let us denote the input voltage as v_{in}, the voltages at the non-inverting and inverting input terminals of the op-amp as v_+ and v_- respectively, and the output voltage of the op-amp (and the clipping stage) as v_{out}. Using the virtual short principle, we have

$$v_+ = v_{in} + 4.5\text{V}, \tag{2.3}$$

$$v_- = v_+. \tag{2.4}$$

To formulate equations for the output voltage v_{out} we must first express the current through the feedback network using KCL:

$$i_{R_g} = i_{C_g} = i_{R_f} + i_{C_f} + i_{D_1} + i_{D_2} \tag{2.5}$$

Let $I_{D\parallel\uparrow}(v) : \mathbb{R} \to \mathbb{R}$ denote the instantaneous current through the antiparallel diode configuration as a function of voltage. With the shorthand notation $\Delta \coloneqq v_{out} - v_-$ and with v_Y denoting the voltage at node Y, we can rewrite (2.5) using the branch constitutive equations of the components as

$$\frac{v_Y}{r_g} = c_g(\dot{v}_- - \dot{v}_Y) = \frac{\Delta}{r_f} + c_f\dot{\Delta} + I_{D\parallel\uparrow}(\Delta) \tag{2.6}$$

Finally, (2.6) can be reformulated as an ordinary differential equation system with two unknown functions $v_Y(t)$ and $\Delta(t)$:

$$\begin{cases} \dot{v}_Y = \dot{v}_- - \dfrac{v_Y}{r_g c_g}, \\ \dot{\Delta} = \dfrac{v_Y}{r_g c_f} - \dfrac{\Delta}{r_f c_f} - \dfrac{I_{D\parallel\uparrow}(\Delta)}{c_f}, \end{cases} \tag{2.7}$$

or, to put it in terms of v_{out}:

$$\begin{cases} \dot{v}_Y = \dot{v}_{in} - \dfrac{v_Y}{r_g c_g}, \\ \dot{v}_{out} = \dot{v}_{in} + \dfrac{v_Y}{r_g c_f} - \dfrac{v_{out} - v_-}{r_f c_f} - \dfrac{I_{D\downarrow\uparrow}(v_{out} - v_-)}{c_f}. \end{cases} \tag{2.8}$$

Right now, these two formulations are functionally equivalent in the sense that one can be transformed into the other by a change of variables. However, we must be careful when designing our final solution, as the two forms produce different outcomes when discretized. This is demonstrated in Appendix C. Formulation (2.7) is preferred due to its conciseness, but (2.8) should be used in the final solution.

There are two things worth noting here. Firstly, even though a closed-form solution exists for v_Y, a much simpler formula can be derived in the discrete time domain by considering that the capacitor C_g and resistor R_g form a first-order high-pass filter network.

Second, we have a degree of freedom in defining $I_{D\downarrow\uparrow}$ for our mathematical model. A common approach is to use the *general diode equation* as seen in Appendix A, which is a reasonably accurate description of the behavior of diodes [7]. However, if we take the internal resistance of the diode into account, i.e. $r > 0$, the resulting equation is rendered implicit. Thus, often r is chosen to be 0, which results in a pure exponential relationship between the diode voltage and diode current. The $r = 0$ formulation of the general diode equation is also known as the *Shockley ideal diode equation* or the *explicit diode equation*. In either case, it is clear that the resulting ODE system (2.8) is nonlinear, and, as we will see in a later chapter, the system is *stiff*.

Using the explicit diode equation, $I_{D\downarrow\uparrow}$ may be defined as

$$I_{D\downarrow\uparrow}(v) \coloneqq I_S \left(e^{\frac{v}{nV_T}} - 1 \right) - I_S \left(e^{\frac{-v}{nV_T}} - 1 \right) = 2I_S \sinh\left(\frac{v}{nV_T} \right). \tag{2.9}$$

This equation may be further simplified by considering that the reverse current of one diode is dominated by the forward current of the other [8], i.e.

$$2I_S \sinh\left(\frac{v}{nV_T} \right) \approx \operatorname{sgn}(v) I_S \left(e^{\frac{|v|}{nV_T}} - 1 \right). \tag{2.10}$$

2.4 Mathematical model of the tone/volume stage

The purpose of this subcircuit is to modify the frequency content of the output signal. The 'LEVEL' knob adjusts the output level of the pedal, while the 'TONE' knob allows the user to adjust the tonal characteristics of their sound. This is a key feature of the TS808 circuit, and is often used in conjunction with the drive knob to shape the overall sound of the pedal.

The tone circuit can be considered a linear, time-invariant filter network. It is an active filter, as it contains a second operational amplifier ('OpTone'). This is a common configuration, as passive filters can introduce significant signal loss, especially at high frequencies. It is important to note that we will be considering the small-signal behavior of this subcircuit. Due to the clipping diodes in the previous stage, the signal entering the tone circuit is limited to a narrow range of voltages, thus the op-amp 'OpTone' is guaranteed to be operating in its linear region.

A filter modifies the frequency content of an input signal, i.e. it alters the amplitude and/or phase of the input signal as a function of frequency. Though the term filter implies filtering out certain frequencies, a filter may also amplify or boost certain frequencies. Linear and time-invariant filters can be fully described by their transfer function.

Lemma 1. Let $x(t) := e^{st}$ for some $s \in \mathbb{C}$. Let y denote the output signal of a linear, time-invariant system to the input signal x. Then, $\exists! \, b_s \in \mathbb{C}$ such that $y(t) = b_s \, e^{st}$.

Definition 7. Let us define the transfer function of a linear, time-invariant system as the $H : \mathbb{C} \to \mathbb{C}$ function $H(s) := b_s$, where $b_s \in \mathbb{C}$ is given by Lemma 1.

Definition 8. Let $f : \mathbb{R} \to \mathbb{C}$. The $F : \mathbb{C} \to \mathbb{C}$ function defined as

$$F(s) = \mathcal{L}\{f\}(s) := \int_{-1}^{+1} f(t) e^{-st} \, dt \tag{2.11}$$

is the bilateral Laplace transform of f.

Lemma 2. Let $x : \mathbb{R} \to \mathbb{C}$ be a continuous signal, such that $\mathcal{L}\{x\}(s)$ exists for all $s \in \mathbb{C}$. If H denotes the transfer function of a linear, time-invariant system, and y denotes the output signal of the system to the input signal x, then the following equation holds:

$$\mathcal{L}\{y\}(s) = H(s) \cdot \mathcal{L}\{x\}(s). \tag{2.12}$$

It is common for the transfer function to be expressed as a rational function of the form

$$H(s) = \frac{Y(s)}{X(s)}. \tag{2.13}$$

The transfer function of the TS808 tone stage can be derived using electrical engineering principles, and is given by

$$H_T(s) = \frac{\left[\frac{1}{R_{p2}} X(s) + \frac{1}{Z_f} \right]}{\frac{1}{R_{p1}} X(s) + \frac{1}{Z_S(s)} + \frac{1}{Z_{in}}}; \tag{2.14}$$

where

$$
\begin{aligned}
X(s) &= \frac{Z_p(s)R_p}{Z_p(s)R_p + R_{p1}R_{p2}};\\
Z_S(s) &= \frac{1}{s \;\; 220 \;\; 10^{\;9} + 10^{\;3}};\\
Z_p(s) &= 220 + \frac{1}{s \;\; 220 \;\; 10^{\;9}};\\
Z_{in} &= 10^3;\\
Z_f &= 10^3;\\
R_p &= 2 \;\; 10^4;\\
R_{p1} &= T \;\; R_p;\\
R_{p2} &= (1 \quad T) \;\; R_p;
\end{aligned}
$$

and $T \; 2 \; [0; 1]$ is the position of the tone knob, as calculated and experimentally veri ed by Paul Darlington [9].

Figure 2.1. Frequency response of the TS808 tone stage.

5.1.4 Solving the diode clipper equation

Using the previous result, we can focus on the diode clipper equation,

$$\dot{v}_{out} = \dot{v}_{in} + \frac{v_Y}{r_g c_f} - \frac{\Delta}{r_f c_f} - \frac{I_{D\Downarrow}(\Delta)}{c_f}, \tag{5.5}$$

where

$$\Delta(t) \coloneqq v_{out} - v_{-}. \tag{5.6}$$

We will use the 7-point stencil central finite difference method (4.1) to estimate $\dot{v}_{in}$. This requires 3 lookahead samples, which is a good tradeoff given the observed accuracy of the method [18].

As for the diode currents, we have two viable options. We can either use the explicit formulation (2.9), thus equation (5.5) becomes

$$\dot{v}_{out} = \dot{v}_{in} + \frac{v_Y}{r_g c_f} - \frac{\Delta}{r_f c_f} - \frac{2I_S \sinh\big(\Delta(nV_T)^{-1}\big)}{c_f}, \tag{5.7}$$

or we can opt for a lookup table, and 'worry about it later'.

Out of curiosity, I tried both options. First, as a naive approach, I implemented the fourth-order explicit Runge-Kutta method—also known as 'RK4'—in C++, to solve equation (5.7). For this experimentation, to be able to evaluate the right-hand side at 'half time steps', I oversampled the input signal beforehand, using the 768-point sinc interpolation method of REAPER [16], as a high-quality oversampling method. The solution quickly diverged to infinity. Could it be due to the hyperbolic sine function on the right-hand side? Possibly. The simulated diode I-V curve is very different to the explicit formulation, which describes a pure exponential relation between current and voltage.

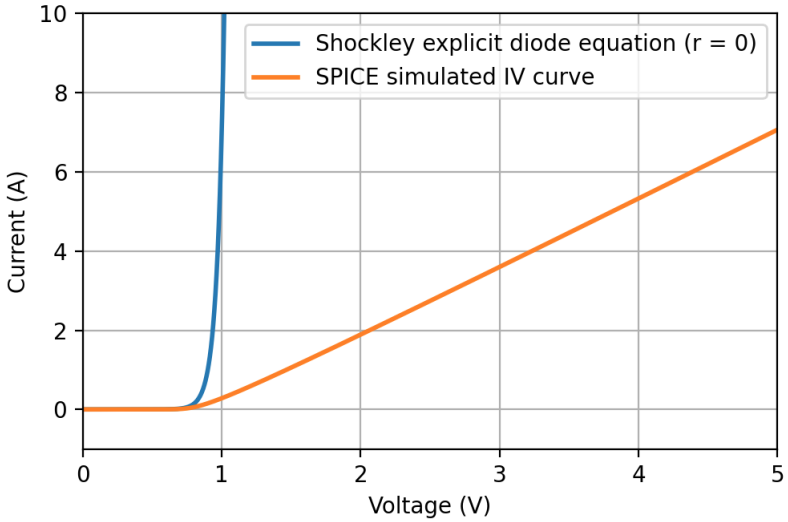


Figure 5.2. Shockley explicit diode equation with $V_T = 26.77\text{mV}$, $n = 1.92$ vs. SPICE simulated I-V curve.

I modified the C++ code to use the precomputed lookup table instead, but the results were the same. Therefore we can say that equation (5.5) is *stiff*.

Despite the indication that explicit methods may not be the best fit for this problem, I experimented with more capable explicit methods. [27] describes a new, fifth-order rational method with promising capabilities. Since this method is rather complicated, I set out to reproduce one of the experiments in the article to test my C++ implementation. Keeping the notations similar to the article, the method was reformulated as:

$$y_{n+1} = \frac{\omega y^2 + \big(2\beta + (\gamma + 480fc^2)h + \delta\big)y - 72h\big((f^3e - 5f^2bd - \frac{10}{3}f^2c^2 + 15fb^2c - \frac{15}{2}b^4)h - 5f^3d + 20f^2bc - 15fb^3\big)}{\omega y + (60bcd - 18b^2e - \alpha f - 40c^3)h^3 + \beta + (\gamma + 180fbd + 240fc^2)h + \delta}, \tag{5.8}$$

where

$$\begin{aligned}
\alpha &:= 15d^2 - 12ce,\\
\omega &:= \alpha h^2 + (36be - 60cd)h - 180bd + 240c^2,\\
\beta &:= \big(-90b^2d + (36fe + 120c^2)b - 60fcd\big)h^2,\\
\gamma &:= -72f^2e - 360b^2c,\\
\delta &:= 360f^2d - 1440fbc + 1080b^3.
\end{aligned}$$

Problem 3 describes the following nonlinear logistic growth model:

$$\begin{cases} \dot{y}(t) &= \frac{y(t)}{4}\left(1 - \frac{y(t)}{20}\right),\\ y(0) &= 1, \end{cases} \tag{5.9}$$

with true solution

$$y(t) = \frac{20\exp(t/4)}{19 + \exp(t/4)}. \tag{5.10}$$

I calculated the required formulae for the method as:

$$\begin{cases} f^{(0)} &:= \frac{y}{4}\left(1 - \frac{y}{20}\right),\\ f^{(1)} &:= \left(\frac{1}{4} - \frac{y}{40}\right)f^{(0)},\\ f^{(2)} &:= \left(\frac{1}{4} - \frac{y}{40}\right)f^{(1)} + \frac{(f^{(0)})^2}{40},\\ f^{(3)} &:= \left(\frac{1}{4} - \frac{y}{40}\right)f^{(2)} + \frac{f^{(1)}f^{(0)}}{40},\\ f^{(4)} &:= \left(\frac{1}{4} - \frac{y}{40}\right)f^{(3)} + \frac{(f^{(1)})^2}{40}. \end{cases} \tag{5.11}$$

I successfully reproduced the results for Problem 3, which can be seen in Figure 5.3.

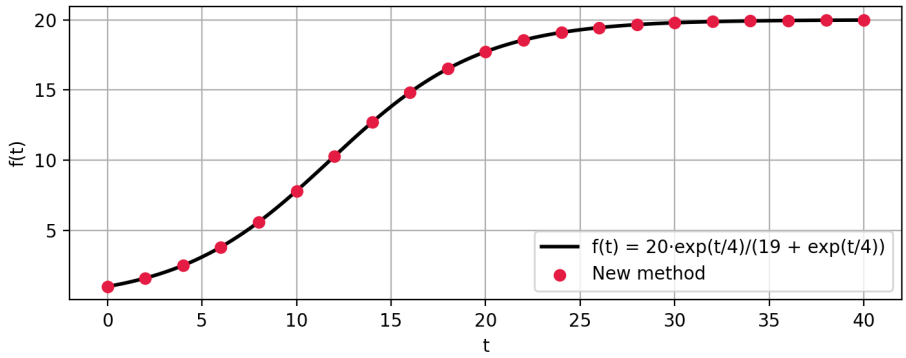


Figure 5.3. True solution vs. numerical solution of (5.9) using the new numerical method [27].

I then implemented this method for the diode clipper equation (5.7). Using the shorthand notations

$$\begin{aligned} y(t) &:= \frac{v_Y(t)}{r_g c_f}, \\ b &:= \frac{-1}{r_f c_f}, \\ d &:= \frac{1}{nV_T}, \\ S(t) &:= \frac{-2I_S \sinh(\Delta d)}{c_f}, \\ C(t) &:= \frac{-2I_S \cosh(\Delta d)}{c_f}, \end{aligned}$$

the derivatives of the right-hand side were calculated as follows.

$$\begin{cases} f^{(0)} &:= y + b\Delta + S, \\ f^{(1)} &:= y^{(1)} + (b + Cd) f^{(0)}, \\ f^{(2)} &:= y^{(2)} + (b + Cd) f^{(1)} + (f^{(0)} d)^2 S, \\ f^{(3)} &:= y^{(3)} + (b + Cd) f^{(2)} + 3 f^{(1)} f^{(0)} d^2 S + (f^{(0)} d)^3 C, \\ f^{(4)} &:= y^{(4)} + (b + Cd) f^{(3)} + 4 f^{(0)} f^{(2)} d^2 S + \\ &\qquad 6 (f^{(0)})^2 f^{(1)} d^3 C + 3 (f^{(1)} d)^2 S + (f^{(0)} d)^4 S. \end{cases} \tag{5.12}$$

The results were similar, the method diverges rather quickly.

Both explicit methods may have been convergent using shorter time steps, which I did not try. Though the tested signal may have been processed successfully, the stability of the resulting algorithms would be quite difficult to reason about.

This result seems to be not uncommon in Virtual Analog modeling, when dealing with nonlinear equations. Article [8] describes a possible solution to a similar diode clipper circuit equation using an implicit numerical method instead. In their work, the simplified explicit diode model (2.10) is used. First, the diode clipper equation is discretized by

the generalized 1-step linear implicit method. Then, to solve for the diode current, the resulting implicit equation is reformulated using the Lambert W function

$$W(z) \coloneqq f^{-1}(z), \tag{5.13}$$

where $f : \mathbb{C} \to \mathbb{C}$ is

$$f(z) \coloneqq ze^z. \tag{5.14}$$

Among many other applications of the W function [28], it can be used to solve equations of the form

$$e^{ax+b} = cx + d, \tag{5.15}$$

with the solution—if exists—being

$$x = \frac{W\left(\frac{-a}{c}e^{b-\frac{ad}{c}}\right)}{-a} - \frac{d}{c} \qquad (a, c \neq 0). \tag{5.16}$$

In their work, D'Angelo et al. proposed several methods to approximate the values of W efficiently using polynomial approximation and Newton-Raphson iteration.

However, their formulation of the diode clipper equation does not account for the internal resistance of the diodes. Trying to solve for the implicit diode equation analytically would add another layer of difficulty, and would possibly require the use of iterative methods as a subroutine for every time step. Instead, we carry on with our solution, keeping the function symbol $I_{D\downarrow\uparrow}$ inside the equations, and 'see where we end up'.

Let us discretize (5.5) using the implicit Euler method:

$$v_{out}[n] = v_{out}[n-1] + h\left(\dot{v}_{in}[n] + \frac{v_Y[n]}{r_g c_f} - \frac{\Delta[n]}{r_f c_f} - \frac{I_{D\downarrow\uparrow}(\Delta[n])}{c_f}\right), \tag{5.17}$$

where

$$\Delta[k] \coloneqq v_{out}[k] - v_{-}[k]. \tag{5.18}$$

Rearrange to get

$$I_{D\downarrow\uparrow}\big(\Delta[n]\big) = -\Delta[n]\left(\frac{c_f}{h} + \frac{1}{r_f}\right) + \frac{c_f}{h}\big(v_{out}[n-1] - v_{in}[n]\big) + c_f\dot{v}_{in}[n] + \frac{v_Y[n]}{r_g}, \tag{5.19}$$

or, in a more elegant form,

$$I_{D\downarrow\uparrow}\big(\Delta[n]\big) = -A\Delta[n] + C, \tag{5.20}$$

where

$$A \coloneqq \left(\frac{c_f}{h} + \frac{1}{r_f}\right),$$
$$C \coloneqq \frac{c_f}{h}\big(v_{out}[n-1] - v_{in}[n]\big) + c_f \dot{v}_{in}[n] + \frac{v_Y[n]}{r_g}.$$

Let us inspect equation (5.20) in more detail. The solution of this equation is exactly the x coordinate of the 2-dimensional point where the line $y = -Ax + C$ intersects the antiparallel diode I-V curve. This point exists and is unique, since $\frac{d}{dx} I_{D\downuparrows} \geq 0$, while $\frac{d}{dx}(-Ax + C) = -A < 0$. This is a very favorable outcome, as this equation can be solved easily and efficiently by using a precomputed lookup table for the antiparallel diode current. If $I_{D\downuparrows}$ is approximated piecewise with first-order polynomials, i.e. with a polyline P, the intersecting line segment of P can be found via binary search, and then the exact point of intersection can be calculated, with a single floating-point division being the 'most demanding' computation.

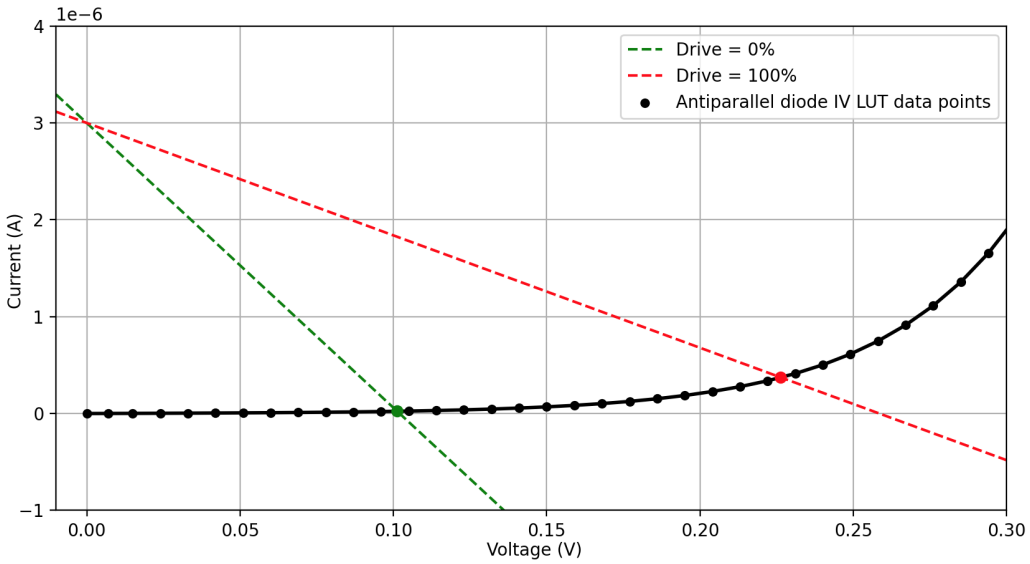


Figure 5.4. Typical example of an I-V intersection calculation during the execution of the diode clipper emulation.

Thus we have found the final solution for the diode clipper equation.

5.2 Stability investigation

Let us now inspect the stability of our solution. For the following calculations we will suppose that our algorithm runs at $T_{\mathrm{FS}} = 192000\mathrm{Hz}$—despite specifying earlier that the target sample rate is 48000Hz, this choice however will be justified shorty. Since the left-hand side of (5.20) is static, we focus on the right-hand side. We will suppose that $C > 0$.

If $\Delta[n]$ is a solution of (5.20), and $\Delta'[n]$ is a solution of

$$I_{D\Downarrow}\big(x\big) = C, \tag{5.21}$$

then

$$|\Delta[n]| \leq |\Delta'[n]|. \tag{5.22}$$

Let us suppose that the input signal v_{in} is bounded, and is in the range of -5V to 5V. Since the IIR system used to calculate v_Y is stable, and its magnitude response is not greater than 1, the following upper bound for C can be easily seen:

$$|C| \leq 4 \cdot 10^{-9}|\Delta[n-1]| + 28 \cdot 10^{-9} + \frac{1}{940}. \tag{5.23}$$

We can 'cheat' a little bit here, and suppose that, as the last operation in our implementation of the diode clipper simulation, we clamp the output signal to be in the range of -5V to 5V—which is actually quite sensible when considering the working principles of the operational amplifier in the clipping stage, and would obviously ensure that the algorithm is BIBO-stable—the final estimate becomes

$$|C| \leq 68 \cdot 10^{-9} + \frac{1}{940}. \tag{5.24}$$

From this, we can conclude that

$$|\Delta[n]| \leq I_{D\Downarrow}^{-1}(68 \cdot 10^{-9} + \frac{1}{940}) \approx 0.6. \tag{5.25}$$

This result tells us that, as long as we formulate our software so as to avoid a division by a near-zero value when working with the lookup table, i.e. maintain numerical stability, the output waveform should, in theory, follow the input waveform rather closely. This is indeed the case, and the algorithm shows no signs of instability when tested with various sensible input signals.

We can also see now that using the implicit diode equation brings little improvement compared to the explicit formulation, as $\Delta[n] \leq 0.6$ implies that only a very small amount of current is flowing through the diodes, and thus we can see from Figure 5.2 that the two formulations should perform similarly. This also explains why the implicit diode formulation did not help with the stability of the tested explicit methods.

5.3 Multirate signal processing to mitigate aliasing

During my experimentation, I realized that significant aliasing occurs when the diode clipper emulation runs at a rate of 48kHz. This is not surprising, as distortion algorithms

generate strong upper harmonics, thus increasing the bandwidth of the signal. The solution to this problem is to oversample the input signal on-the-fly, then feed the oversampled signal to the distortion algorithm, which runs at a higher rate, and then decimate the output signal. This technique is called *multirate signal processing*, and is very common in modern signal processing algorithms with nonlinear behavior.

For the sake of simplicity, we will only cover the case when our emulation runs in a 48kHz environment. We will aim for an emulation rate of 192kHz, therefore we need to oversample the input signal by a factor of 4, and then decimate the output by a factor of 4 as well.

One possible solution is to estimate the derivative of the unoversampled signal—we will need it for the diode clipper emulation anyway—and then incorporate this estimate into the oversampling polynomial. Let us construct a 7-th order Hermite-Fejér interpolating polynomial—named after French and Hungarian mathematicians Charles Hermite and Lipót Fejér—with 8 constraints, using four nearby points, so that

$$\begin{aligned} p(-h) &= x[n_0{-}1], & p'(-h) &= dx[n_0{-}1], \\ p(0) &= x[n_0], & p'(0) &= dx[n_0], \\ p(h) &= x[n_0{+}1], & p'(h) &= dx[n_0{+}1], \\ p(2h) &= x[n_0{+}2], & p'(2h) &= dx[n_0{+}2], \end{aligned} \tag{5.26}$$

where $x[k]$ denotes the unoversampled input signal, and $dx[k]$ denotes the estimated derivatives. We can use p to generate 3-3 new sample points at $t = \frac{1}{4}h$, $t = \frac{1}{2}h$ and $t = \frac{3}{4}h$ to achieve a 4x oversampling of both the input signal $x[k]$ and its derivative $dx[k]$. To do so, we must solve the following system of linear equations:

$$\begin{bmatrix} a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & u_1 & u_2 & u_3 & v_1 & v_2 & v_3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -h & (-h)^2 & (-h)^3 & (-h)^4 & (-h)^5 & (-h)^6 & (-h)^7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & h & h^2 & h^3 & h^4 & h^5 & h^6 & h^7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2h & (2h)^2 & (2h)^3 & (2h)^4 & (2h)^5 & (2h)^6 & (2h)^7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2(-h) & 3(-h)^2 & 4(-h)^3 & 5(-h)^4 & 6(-h)^5 & 7(-h)^6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2h & 3h^2 & 4h^3 & 5h^4 & 6h^5 & 7h^6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2(2h) & 3(2h)^2 & 4(2h)^3 & 5(2h)^4 & 6(2h)^5 & 7(2h)^6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & \frac{1}{4}h & (\frac{1}{4}h)^2 & (\frac{1}{4}h)^3 & (\frac{1}{4}h)^4 & (\frac{1}{4}h)^5 & (\frac{1}{4}h)^6 & (\frac{1}{4}h)^7 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & \frac{1}{2}h & (\frac{1}{2}h)^2 & (\frac{1}{2}h)^3 & (\frac{1}{2}h)^4 & (\frac{1}{2}h)^5 & (\frac{1}{2}h)^6 & (\frac{1}{2}h)^7 & 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & \frac{3}{4}h & (\frac{3}{4}h)^2 & (\frac{3}{4}h)^3 & (\frac{3}{4}h)^4 & (\frac{3}{4}h)^5 & (\frac{3}{4}h)^6 & (\frac{3}{4}h)^7 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 2(\frac{1}{4}h) & 3(\frac{1}{4}h)^2 & 4(\frac{1}{4}h)^3 & 5(\frac{1}{4}h)^4 & 6(\frac{1}{4}h)^5 & 7(\frac{1}{4}h)^6 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 2(\frac{1}{2}h) & 3(\frac{1}{2}h)^2 & 4(\frac{1}{2}h)^3 & 5(\frac{1}{2}h)^4 & 6(\frac{1}{2}h)^5 & 7(\frac{1}{2}h)^6 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 2(\frac{3}{4}h) & 3(\frac{3}{4}h)^2 & 4(\frac{3}{4}h)^3 & 5(\frac{3}{4}h)^4 & 6(\frac{3}{4}h)^5 & 7(\frac{3}{4}h)^6 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ d_0 \\ d_1 \\ d_2 \\ d_3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.27}$$

where

$$\begin{aligned} x_0 &\coloneqq x[n_0{-}1], & d_0 &\coloneqq dx[n_0{-}1], \\ x_1 &\coloneqq x[n_0], & d_1 &\coloneqq dx[n_0], \\ x_2 &\coloneqq x[n_0{+}1], & d_2 &\coloneqq dx[n_0{+}1], \\ x_3 &\coloneqq x[n_0{+}2], & d_3 &\coloneqq dx[n_0{+}2], \end{aligned} \tag{5.28}$$

and a_k denote the coefficients of p. The solution of (5.27) calculated symbolically is

$$\begin{aligned} u_1 &= \frac{3283x_0 + 165375x_1 + 25725x_2 + 2225x_3 + h(735d_0 + 33075d_1 - 11025d_2 - 525d_3)}{196608}, \\ u_2 &= \frac{13x_0 + 243x_1 + 243x_2 + 13x_3 + h(3d_0 + 81d_1 - 81d_2 - 3d_3)}{512}, \\ u_3 &= \frac{2225x_0 + 25725x_1 + 165375x_2 + 3283x_3 + h(525d_0 + 11025d_1 - 33075d_2 - 735d_3)}{196608}, \\ v_1 &= \frac{11935x_0 - 174825x_1 + 152145x_2 + 10745x_3 + h(2751d_0 + 44415d_1 - 58905d_2 - 2505d_3)}{147456h}, \\ v_2 &= \frac{-5x_0 - 405x_1 + 405x_2 + 5x_3 + h(-d_0 - 81d_1 - 81d_2 - d_3)}{256h}, \\ v_3 &= \frac{-10745x_0 - 152145x_1 + 174825x_2 - 11935x_3 + h(-2505d_0 - 58905d_1 + 44415d_2 + 2751d_3)}{147456h}. \end{aligned} \tag{5.29}$$

Not too surprisingly, this method is equivalent to convolving the 3-zero-stuffed input signal with the following 39-tap FIR kernel:

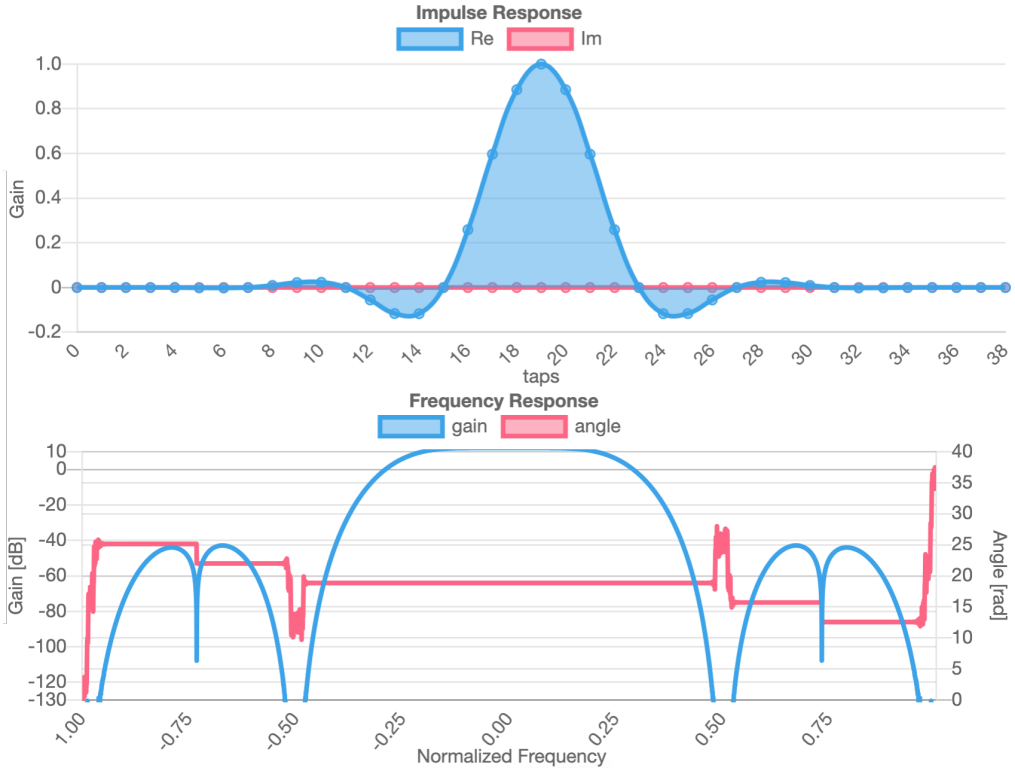


Figure 5.5. Created with `https://cho45.stfuawsc.com/fir-filter-vis/`.

The frequency response plot shows desirable qualities for the proposed method; a flat passband in the audible frequency range, and a strong attenuation near $\pm 0.5\pi$ and $\pm\pi$ to suppress the zero-stuffed spectral images. The passband gain is about 12dB, which describes a 4x amplification. This is logical if we consider that the equivalent FIR interpolator must 'work against' the 3-zero-stuffed input signal. To oversample a block of 32 input samples—resulting in 128-128 samples for the signal and its derivative—this method requires a total of 2240 multiplications and 1952 additions.

The decimation of the processed signal is more straightforward. For this, I used a FIR design tool [29] to design a 4x linear-phase decimation filter, which runs at 192kHz and has 105 taps, and therefore has a group delay of roughly 0.27 milliseconds. It has an excellent 0.13dB passband ripple and -75dB stopband attenuation. In my C++ code, I implemented it using the polyphase decimation technique described earlier in Chapter 3. I also created a small C++ benchmark to test the performance of the decimation algorithm. A 240 seconds long 192kHz audio track was decimated in about 315 milliseconds using the single-phase approach. This was reduced to about 270 milliseconds with the polyphase approach.

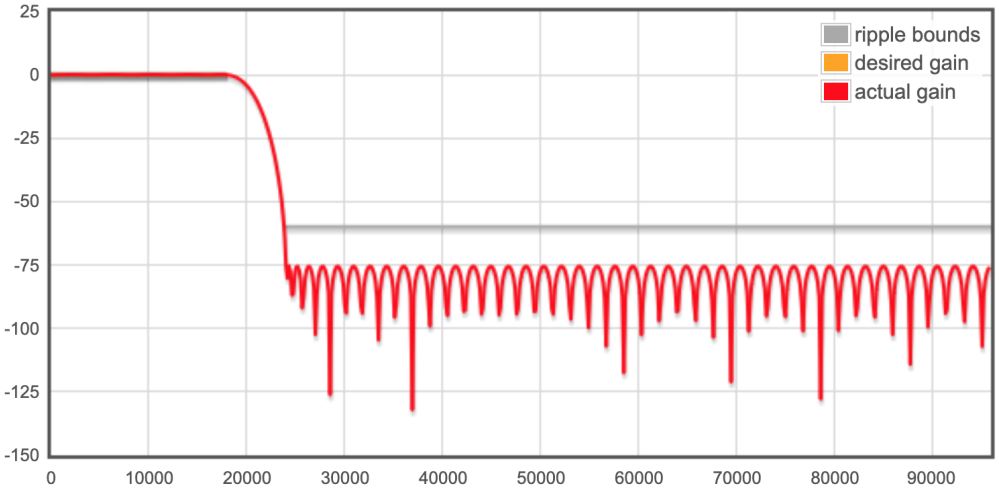


Figure 5.6. The designed 4x decimation filter.

5.4 Emulating the tone/volume section

So far, we have been doing white-box modeling. The same approach would work for the tone/volume section as well; we could use our knowledge of this subcircuit to decompose it into a cascade of LTI filters, perhaps a low-pass filter, followed by a high-frequency shelving filter. These filters may be derived using the 'more conventional' digital filter design approaches presented in [14]. However, I opted for a more general black-box modeling approach out of curiosity.

It should be mentioned that we have the choice of running the tone circuit emulation before or after the decimation process. Our first instict might say that we should run it

afterwards to cut down on the amount of required calculations. However, as we will see shortly, the tone circuit emulation is rather lightweight, and running it before decimation can in fact help with aliasing suppression.

Suppose that we only have knowledge about the frequency response of this subcircuit for a set of tone knob positions that span the $[0, 1]$ interval. Let $p_0, \ldots, p_N$ denote this set of positions in ascending order, with $p_0 = 0$ and $p_N = 1$. More specifically, we only have a finite set of samples of the frequency response in the 60Hz to $\frac{192000}{2} = 96000$Hz range, with 60Hz being the lowest frequency component that electric guitars typically produce. Let $f_0, \ldots, f_M$ denote this set of frequencies in ascending order. Let $\overset{\circ}{H}_k \in \mathbb{C}^{N+1}$ denote the samples of the target frequency response at the k-th tone knob position, where $k \in \{0, \ldots, N\}$. For all positions of the tone knob, the frequency response of the tone stage closely resembles that of a low-pass filter with some additional resonance (Figure 2.1). Therefore we will try to find a second-order IIR system of the form

$$y[n] = -a_{k,1}y[n-1] - a_{k,2}y[n-2] + b_{k,0}x[n] + b_{k,1}x[n-1] + b_{k,2}x[n-2], \tag{5.30}$$

whose frequency response $\overset{\circ}{H}'_k$ approximates $\overset{\circ}{H}_k$, and the unkown coefficients to be determined are real numbers. This may be formulated as an optimization problem, where the cost function to be minimized is the error of the IIR system's frequency response at frequencies $f_0, \ldots, f_M$, i.e. $\|\overset{\circ}{H} - \overset{\circ}{H}'\|_2^2$. This problem can be solved e.g. via gradient descent methods. For my experimentation, I used the *Broyden-Fletcher-Goldfarb-Shanno* (BFGS) algorithm from the SciPy Python library.

However, to make sure the resulting IIR systems are stable, we need to extend the cost function to include an error term that penalizes poles outside and on the complex unit circle. To keep the poles safely within the unit circle, I found the penalty function

$$(1.01 \cdot (|z| + 0.01))^{64} \tag{5.31}$$

to be very effective, where $z \in \mathbb{C}$ denotes a pole of the system. We can use the same formula to keep the zeros of the system within the unit circle, thus resulting in a minimum-phase IIR system, if desired. This optimization problem may be further refined by assigning different error weights to the $f_0, \ldots, f_M$ frequencies. In my experiment, I assigned a weight of 1 to audible frequencies in the range of 60Hz to 20kHz, and a weight of 0.2 to frequencies above 20kHz, thus resulting in a slightly better fit in the audible frequency range.

This approach gave me good results throughout the entire range of the tone knob parameter. Thus we have a way of precalculating optimal IIR coefficients for the set of measurements corresponding to the tone knob positions $p_0, \ldots, p_N$. However, to be able to incorporate a continuously variable parameter into the final emulation, we need

a way to 'interpolate between IIR filters'. At first I tried linear interpolation between the neighboring filter coefficients, which turned out to be subpar. The interpolated filters were not always stable or sounded badly. I needed a way to guarantee the stability of the interpolated filters. A much better approach is to interpolate between the poles and zeros of the system. It is easy to see that, given a 1-1 mapping between the poles and zeros of two IIR systems, and given that the poles of both systems lie within the unit circle, the interpolated filters will also be stable. Furthermore, we can see from (3.31) that varying the poles and zeros of an IIR system continuously results in a continuous 'morphing' of the frequency response, that is, the frequency response of the system will converge pointwise. I devised a new optimization problem, that includes all $5(N+1)$ coefficients, and besides the individual cost terms, also incorporated a new penalty term based on the distances of poles and zeros of neighboring IIR filters. The end result was a set of IIR filters corresponding to the tone knob positions $p_0, \ldots, p_N$ that I was able to interpolate between, and sounded good throughout the entire parameter range, with no signs of instability. The final set of IIR filters can be found in Appendix D.

5.5 Evaluating the final emulation

We can conduct various tests on the final software emulation to evaluate its performance. We can compare the output signal to that of other software emulations or a SPICE simulated output signal. I had access to two other software emulations, the Line 6 Helix Native [30] and the STL Tonality: Andy James [31] VST plugins. For a fair comparison, I set both the drive and tone knobs on all platforms to neutral, that is, to a 'noon' knob position. All output signals were phase-aligned and normalized to -18 LUFS (integrated). I found a significant discrepancy between the input level handling of the mentioned software, therefore I adjusted the level of the input signals individually to achieve a similar level of distortion from all software emulations.

The simplest method is to listen to the output signal and see if we can identify sonic differences compared to the reference signals. My personal experience was that our emulation sounds reasonably good, and captures the overall character of the real circuit. The amount of distortion and the frequency content of the output signal is not exactly spot-on, but this is to be expected. These aspects can and should be adjusted manually to fine-tune the final emulation and compensate for numerical or other inaccuracies. I am certain that the mentioned reference emulations are designed and engineered much more thorougly, therefore they sounded more faithful to the SPICE simulation. I did not identify any clear issues with our solution—e.g. popping or clicking artifacts, aliasing, instability—with sensible input signals, the sound was clear and consistent with any drive and tone control values, even when adjusting them continuously in real-time.

A slightly more scientific comparison is the so-called *null test*, where the two signals to be compared are phase- and level-matched and then played simultaneously, but with one signal phase-inverted. The frequency content and overall loudness of the resulting signal can be used to measure the similarity of the compared signals. The resulting 'difference signals' with respect to our own emulation had the following loudness values: SPICE simulation −37.6 LUFS (momentary), Helix Native −32.5 LUFS (momentary), and STL Tonality −43 LUFS (momentary). This result confirms my subjective experience, as I found our model to sound most similar to the STL Tonality plugin, both of which sounded slightly dissimilar to the SPICE simulation and Helix Native.

By inspecting the phase-aligned and normalized output waveforms visually, we can arrive at the same conclusions. We can also clearly see the effects of soft clipping, that is, the peaks of the output waveforms are rounded off in a smooth manner.

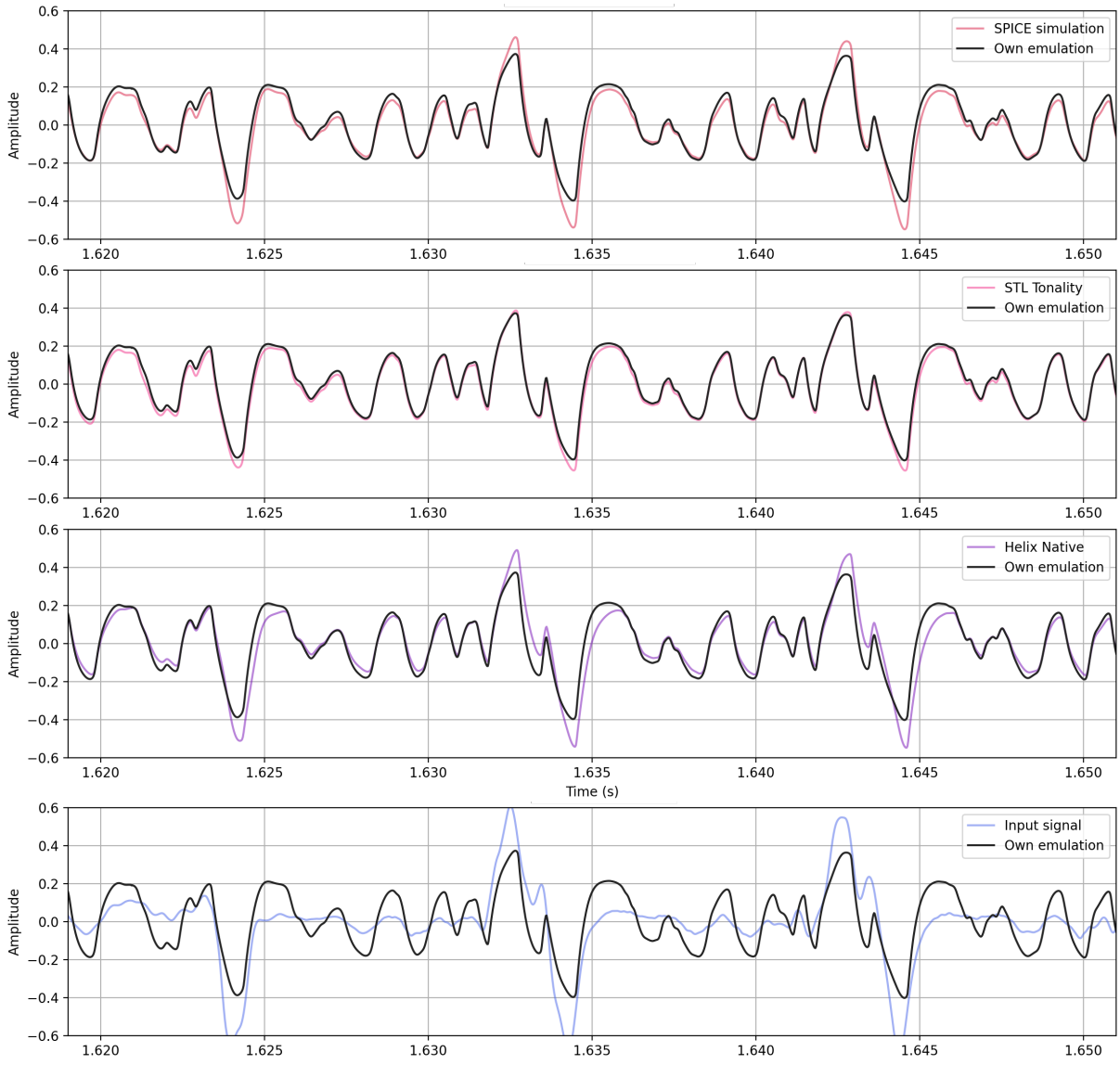
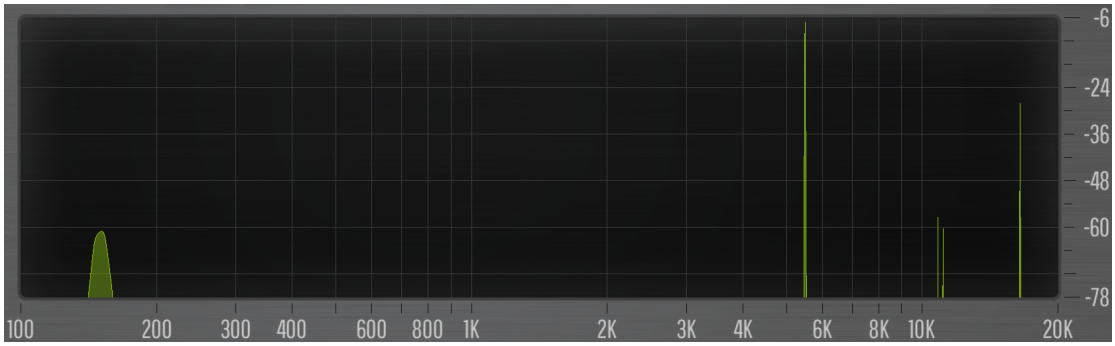


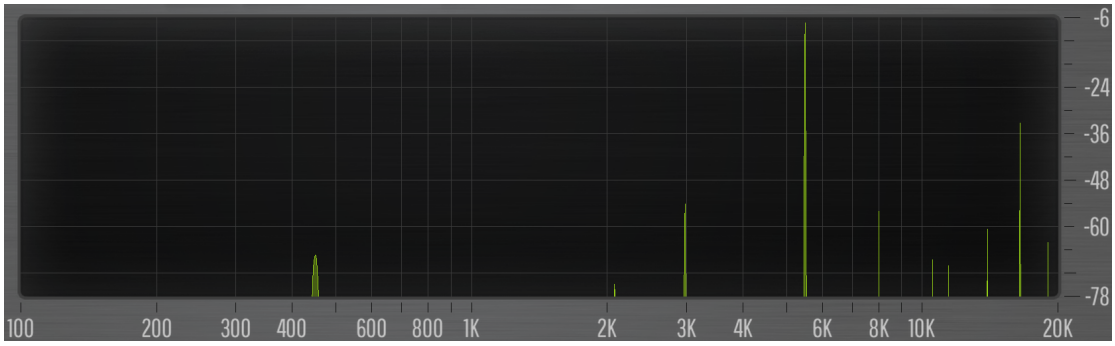
Figure 5.7. Comparison of output waveforms near a transient event (plucked guitar string) at ∼1.624s. The drive and tone controls were set to neutral (0.5) for all software.

To test aliasing suppression, a pure sinewave signal with a frequency of 5490Hz was processed using all software emulations. I did not encounter obvious signs of aliasing below about 5000Hz with any of the tested software. The output signal was then analyzed with a FFT spectrum analyzer, the result of which can be seen in Figure 5.8. Our own emulation showed comparable aliasing suppression to the reference emulations.



(a) Own emulation

(b) STL Tonality



(c) Helix Native

Figure 5.8. Comparison of output magnitude spectra to test aliasing suppression.

To reduce the size of the diode LUT, I used a primitive algorithm that greedily removed data points from the 10001-long LUT in such a way that the maximum relative error of the resulting LUT—when compared to the full table—was no more than 0.5%. Furthermore, since $I_{D\downarrow\uparrow}$ is odd, it is sufficient to store only the positive half of the table. This resulted in a 96-long LUT, thus a single lookup operation requires at most $\lceil \log_2 96 \rceil = 7$ floating-point comparisons. The introduced latency and amount of required calculations to process a block of 32 input samples are summarized in the following table.

	Required calculations	Required lookahead samples	Latency (ms)
Oversampling + derivative estimation	2240 multiplications 1952 additions	4 (48kHz)	0.083
Clipping circuit emulation	128 multiplications 129 divisions 513 additions 1024 comparisons 2049 std::fma invocations 128 std::lerp invocations	0	0
Tone circuit emulation	640 multiplications 512 additions	0	0
Decimation	3360 multiplications 3328 additions	52 (192kHz)	0.271

Table 5.1. Computational requirements of the final emulation, implemented in C++.

The total latency of the emulation is precisely 17 samples at 48kHz, which I could manually verify in REAPER. For comparison, Helix Native has a self-reported latency of 16 samples at 48kHz.

I used the built-in performance meter of REAPER to compare the CPU utilization of the three emulations. The single-core utilization for Helix Native fluctuated around 4.2%, STL Tonality around 3.8%, and our own emulation around 3%.

Based on the presented results, we can conclude that the final software is adequate for real-time use, and should be able to run on less powerful devices, such as portable audio DSP chips.

5.6 Final thoughts

We successfully created a real-time emulation of a guitar distortion pedal using various branches of mathematics. Yet there are numerous topics we didn't have time to explore, but can be found within the referenced works. For those interested, I highly recommend [2, 13, 14, 24] to guide them deeper into the world of digital signal processing.

I really enjoyed working on this project, and am very satisfied with the results. That said, the final emulation isn't perfect. There is plenty of room for improvement. The diode clipper emulation could be enhanced by using higher order numerical methods as discretization tools. The diode LUT could be optimized further based on the calculated upper bound for the range of output voltages, and so on. The current formulation works as-is, and makes for a good starting point, but wasn't designed to be optimal e.g. in terms of operation count or accuracy. Trying to improve upon the presented solutions to adhere to new, additional requirements would make for a good exercise, if not more.

References

[1] Accompanying GitHub repository for this thesis. [Online]. Available: https://github.com/Szapi/va-modeling

[2] J. W. Nilsson and S. A. Riedel, *Electric Circuits*, 10th ed. Pearson Education, 2015.

[3] J. D. Jackson, *Classical Electrodynamics*, 2nd ed. John Wiley & Sons, Inc., 1975.

[4] R. G. Keen, "The Technology of the Tube Screamer," 1998, last accessed 17 March 2025. [Online]. Available: http://www.geofex.com/article_folders/tstech/tsxtech.htm

[5] "ElectroSmash - Tube Screamer Analysis," last accessed 9 May 2025. [Online]. Available: https://www.electrosmash.com/tube-screamer-analysis

[6] D. E. Richards, *Basic Engineering Science - A Systems, Accounting and Modeling Approach.* Rose-Hulman Institute of Technology, 2001.

[7] P. Chakravorty, "A Modified General Diode Equation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.

[8] S. D'Angelo, L. Gabrielli, and L. Turchet, "Fast Approximation of the Lambert W Function for Virtual Analog Modelling," *Proceedings of the 22nd International Conference on Digital Audio Effects (DAFx-19)*, 2019.

[9] P. Darlington, "Analysis of the Tube Screamer," 2012, last accessed 14 April 2025. [Online]. Available: https://m0xpd.blogspot.com/2012/11/analysis-of-tube-screamer.html

[10] Voxengo SPAN real-time audio frequency spectrum analyzer. Last accessed 18 May 2025. [Online]. Available: https://www.voxengo.com/product/span/

[11] B. Boashash, *Time-Frequency Signal Analysis and Processing: A Comprehensive Reference*, 2003.

[12] P. Simon, "Fourier transform - University textbook," 2019, last accessed 27 April 2025. [Online]. Available: https://www.inf.elte.hu/dstore/document/300/Simon-Peter-Fourier-transzformacio.pdf

[13] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed. Pearson Education, 2010.

[14] T. W. Parks and C. S. Burrus, *Digital Filter Design.* John Wiley & Sons, Inc., 1987.

[15] M. Nagy, "Complex Function Theory (introduction) - Draft," p. 80, 2021, last accessed 27 April 2025. [Online]. Available: https://nmarci.web.elte.hu/dirs/jegyzetek/Complex.pdf

[16] REAPER Digital Audio Workstation. Last accessed 30 April 2025. [Online]. Available: https://www.reaper.fm/

[17] O. Niemitalo, "Polynomial Interpolators for High-Quality Resampling of Oversampled Audio," last accessed 30 April 2025. [Online]. Available: https://yehar.com/blog/wp-content/uploads/2009/08/deip.pdf

[18] M. Lewandowski, "Estimating the first and second derivatives of discrete audio data," *EURASIP Journal on Audio, Speech, and Music Processing*, 2024.

[19] A. Fettweis, "Wave Digital Filters: Theory and Practice," *Proceedings of the IEEE*, vol. 74, 1986.

[20] A. Sarti and G. D. Poli, "Toward Nonlinear Wave Digital Filters," *IEEE Transactions on Signal Processing*, vol. 47, 1999.

[21] K. J. Werner, J. O. S. III, and J. S. Abel, "Wave Digital Filter Adaptors for Arbitrary Topologies and Multiport Linear Elements," *Proceedings of the 18th International Conference on Digital Audio Effects (DAFx-15)*, 2015.

[22] S. Petrausch and R. Rabenstein, "Wave digital filters with multiple nonlinearities," *12th European Signal Processing Conference*, 2004.

[23] K. Werner, V. Nangia, A. Bernardini, J. Smith, and A. Sarti, "An Improved and Generalized Diode Clipper Model for Wave Digital Filters," 2015.

[24] K. J. Werner, "Virtual analog modeling of audio circuitry using wave digital filters," PhD thesis, Stanford University, Department of Music, 2016.

[25] 1N4148 diode SPICE model. Last accessed 2 May 2025. [Online]. Available: https://diotec.com/request/spice/1n4148.zip

[26] D. Horváth, Z. Červeňanská, and J. Kotianová, "Digital implementation of Butterworth first-order filter type IIR," *Research Papers Faculty of Materials Science and Technology Slovak University of Technology*, vol. 27, 2019.

[27] S. Qureshi, M. A. Akanbi, A. A. Shaikh, A. S. Wusu, O. M. Ogunlaran, W. Mahmoud, and M. Osman, "A new adaptive nonlinear numerical method for singular and stiff differential problems," *Alexandria Engineering Journal*, vol. 74, pp. 585–597, 2023.

[28] R. Corless, G. Gonnet, D. Hare, D. Jeffrey, and D. Knuth, "On the Lambert W Function," *Advances in Computational Mathematics*, vol. 5, pp. 329–359, 01 1996.

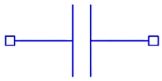
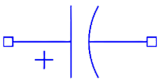
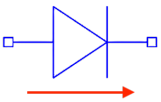
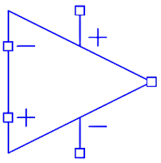
[29] TFilter filter design tool. Last accessed 7 May 2025. [Online]. Available: http://t-filter.engineerjs.com/

[30] Line 6 Helix Native Amp & Effects Plugin. Last accessed 23 May 2025. [Online]. Available: https://line6.com/helix/helixnative.html

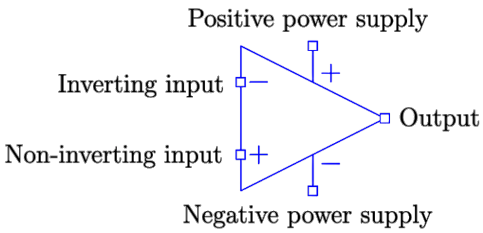
[31] STL Tonality: Andy James Guitar Plug-In Suite. Last accessed 23 May 2025. [Online]. Available: https://www.stltones.com/products/tonality-andy-james-guitar-plug-in-suite

Appendix A

Electrical components

Name	Symbol	Parameters	Branch Constitutive Equation
Resistor		$r > 0$ (resistance)	$i = \frac{v}{r}$
Capacitor		$c > 0$ (capacitance)	$i = c\dot{v}$
Polarized Capacitor		$c > 0$ (capacitance)	$i = c\dot{v}$
Diode		$I_S > 0$ (reverse saturation current) V_T (thermal voltage) $n \geq 1$ (ideality factor) $r \geq 0$ (internal resistance)	$i = I_S \left(e^{\frac{v-ir}{nV_T}} - 1 \right)$
Operational Amplifier			
Ground Node			

Operational amplifier nomenclature



TS808 schematic

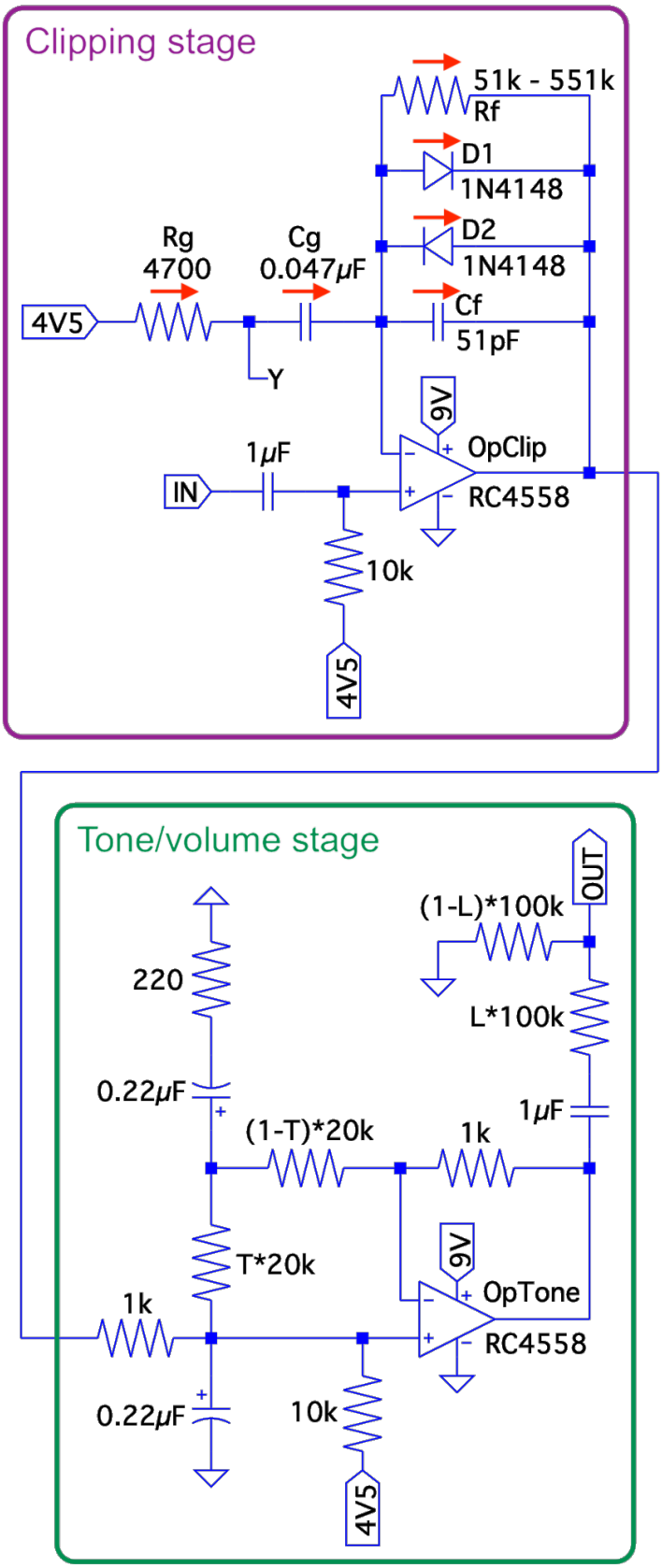


Figure 1. Schematics of the TS808 circuit, with indicated current measurements for the diode clipper network.

Appendix B

SPICE simulation of the TS808 circuit

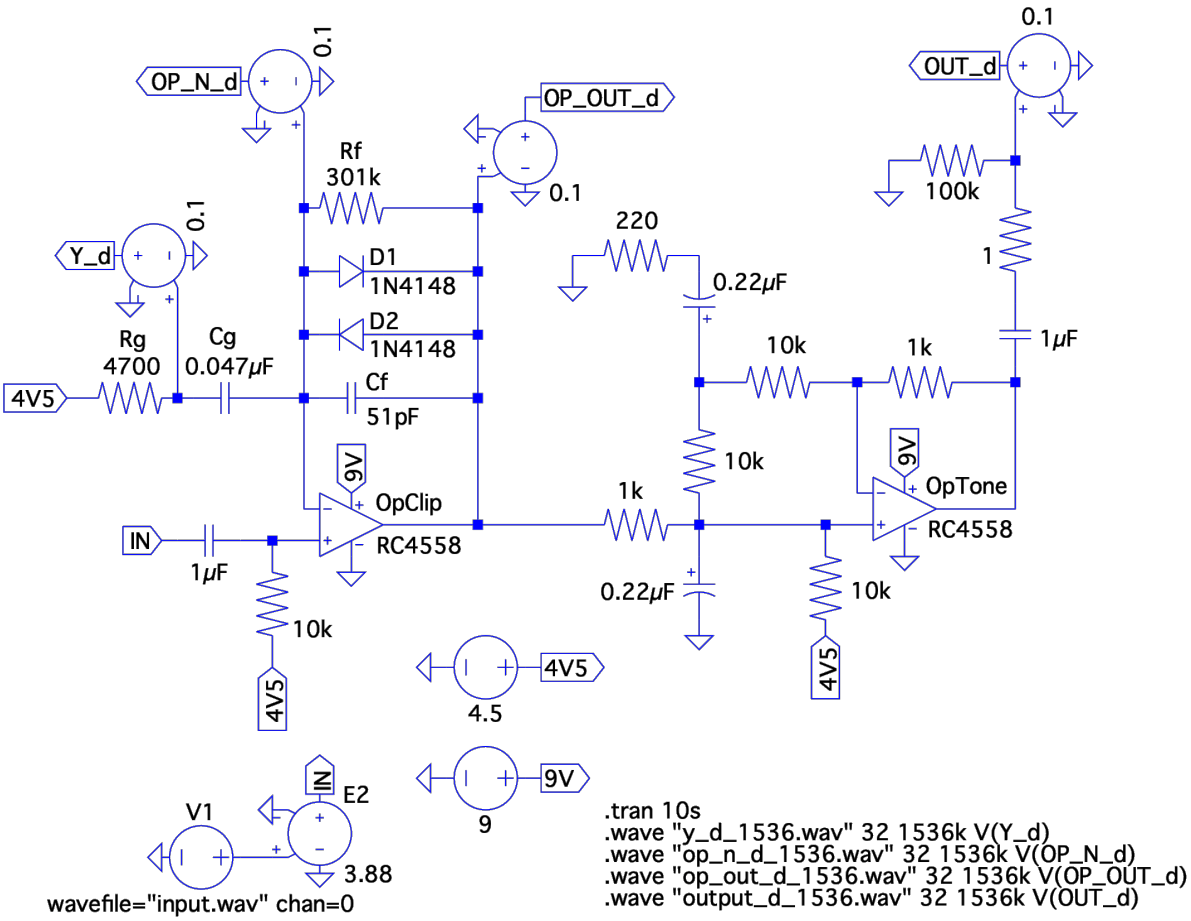


Figure 2. SPICE simulation of the TS808 circuit with both the drive and tone controls at 0.5.

Test program output

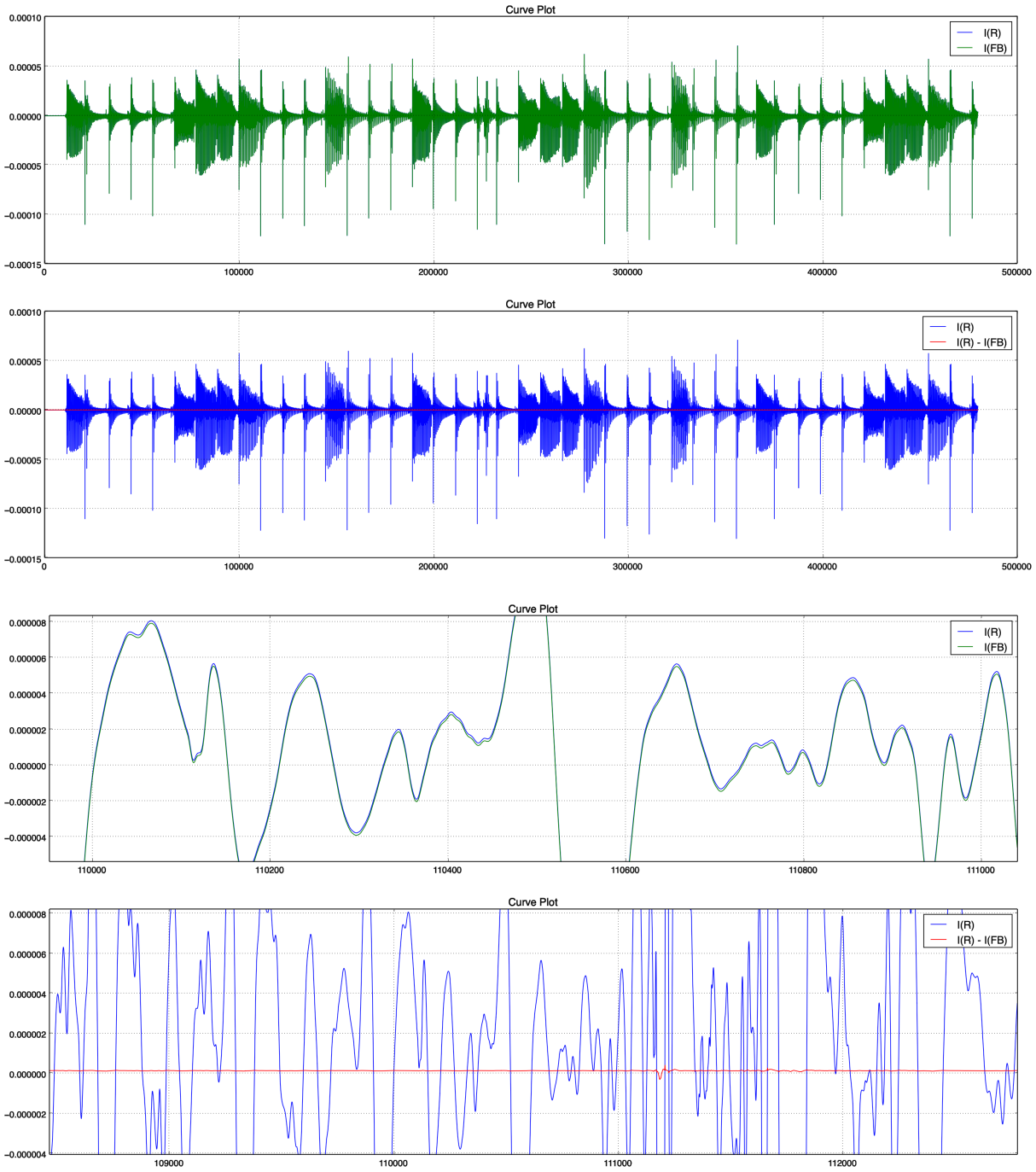


Figure 3. Output of a test program to verify equation (2.8). $I(R)$ denotes the current through resistor R_g. $I(FB)$ denotes the current through the feedback network (R_f, C_f, D_1, D_2).

Appendix C

Which diode clipper equation to discretize?

Let us discretize both formulations of the diode clipper equation and compare the results. Discretizing equation (2.7) yields

$$\Delta[n] = \Delta[n{-}1] + h\left(\frac{v_Y[n]}{r_g c_f} - \frac{\Delta[n]}{r_f c_f} - \frac{I_{D\Downarrow\Uparrow}(\Delta[n])}{c_f}\right), \tag{1}$$

where

$$\Delta[k] \coloneqq v_{out}[k] - v_{-}[k], \tag{2}$$

whereas discretizing (2.8) yields

$$v_{out}[n] = v_{out}[n{-}1] + h\left(\dot{v}_{in}[n] + \frac{v_Y[n]}{r_g c_f} - \frac{\Delta[n]}{r_f c_f} - \frac{I_{D\Downarrow\Uparrow}(\Delta[n])}{c_f}\right). \tag{3}$$

Let us subtract (1) from (3):

$$v_{-}[n] = v_{-}[n{-}1] + h\dot{v}_{in}[n], \tag{4}$$

which rearranges to

$$\frac{v_{-}[n] - v_{-}[n{-}1]}{h} = \dot{v}_{in}[n]. \tag{5}$$

This tells us that we can derive equation (1) from (3) by substituting the above approximation formula for $\dot{v}_{in}[n]$. We can get a much better approximation for the derivative of the input signal using the 7-point stencil formula, therefore equation (3) should result in a more accurate emulation.

Appendix D

Optimal tone circuit filters and the pole-zero-gain IIR parameterization

The optimal set of IIR filters for the tone/volume section turned out to have real-valued poles and zeros. This makes intuitive sense, as 2-2 real-valued poles and zeros allow for more flexible frequency response shaping than if they were complex conjugate pairs, as they can 'cover' more of the frequency spectrum. It would have been interesting to examine this phenomenon more closely, however, due to time constraints, I just accepted this result as the designed filters worked and sounded fine. These second-order IIR filters of the form (5.30) can be equivalently represented by 5 real numbers; the locations of the poles and zeros, and a 'gain' parameter. The following formula can be used to calculate the coefficients of such filters efficiently:

$$
\begin{aligned}
b_0 &= g, & a_1 &= -(p_1 + p_2), \\
b_1 &= -g \cdot (z_1 + z_2), & a_2 &= p_1 \cdot p_2, \\
b_2 &= g \cdot z_1 \cdot z_2,
\end{aligned}
$$

where $z_{1,2}$ and $p_{1,2}$ denote the zeros and poles respectively, and g denotes the gain parameter value.

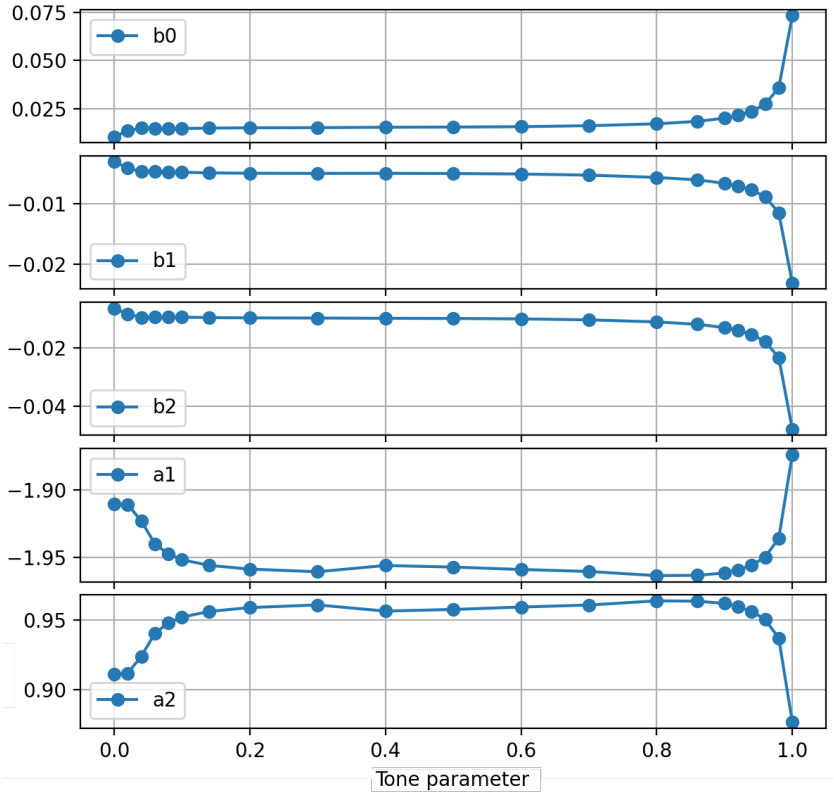


Figure 4. Optimal tone circuit IIR filter coefficients.

Tone param.	Pole 1	Pole 2	Zero 1	Zero 2	Gain
0.00	0.988573733	0.921770981	0.951611485	-0.665514294	0.010646047
0.02	0.989119823	0.921769510	0.965697699	-0.665495445	0.014008906
0.04	0.989765732	0.933399619	0.974874521	-0.665407386	0.015159218
0.06	0.990861110	0.949007324	0.982561684	-0.665383379	0.014826614
0.08	0.991688219	0.955808458	0.986273820	-0.665406549	0.014780225
0.1	0.992250912	0.959391745	0.988284028	-0.665421679	0.014831721
0.14	0.992628706	0.963154015	0.990052848	-0.665495656	0.014945623
0.2	0.992621601	0.965949156	0.990918706	-0.665387912	0.015088762
0.3	0.991903486	0.968582514	0.990890535	-0.665368746	0.015201411
0.4	0.986395586	0.969489563	0.985260394	-0.665473894	0.015432830
0.5	0.985550111	0.971522586	0.985362676	-0.665491359	0.015540709
0.6	0.985543241	0.973280182	0.986373992	-0.665484903	0.015809152
0.7	0.985634468	0.974657299	0.987416491	-0.665494284	0.016267243
0.8	0.988710323	0.974654237	0.990720330	-0.665514799	0.017239386
0.86	0.988603602	0.974529597	0.991276136	-0.665526396	0.018457541
0.9	0.987795197	0.973609324	0.991227246	-0.665517694	0.020203693
0.92	0.985548707	0.973621711	0.990256791	-0.665487693	0.021634968
0.94	0.982172595	0.973496962	0.989027403	-0.665325959	0.023827269
0.96	0.976187375	0.973494590	0.987349712	-0.665306245	0.027655391
0.98	0.973286563	0.962515944	0.984689069	-0.665302451	0.036337655
1.0	0.973297472	0.900786387	0.980303797	-0.665322887	0.074949153

Table 2. Optimal tone circuit IIR filters.

Alulírott Ték Róbert Máté nyilatkozom, hogy szakdolgozatom elkészítése során az alább felsorolt feladatok elvégzésére a megadott MI alapú eszközöket alkalmaztam:

Feladat	Felhasznált eszköz	Felhasználás helye	Megjegyzés
Szövegvázlat készítés	OpenAI GPT-4o	2. fejezet (bevezetés) 2.4 paragrafus 3.2 fejezet (bevezetés) 3.2.2 paragrafus (bevezetés) 4.2 paragrafus 4.3 paragrafus 4.5 paragrafus 5.4 paragrafus	Többmondatos vázlat készítése inspirációs céllal, egy-egy mondat/mondatrész/kifejezés szebb megfogalmazására javaslat.
Segítség új szakterület megismeréséhez	OpenAI GPT-4o	3. fejezet	Pl. legfontosabb szakkifejezések, definíciók, tételek, módszerek megismerése, amik alapján a nagyobb terjedelmű forrásokat (pl. [1], [2], [11], [12], [13]) már célzott módon tudtam feldolgozni.
Python kód generálás	OpenAI GPT-4o, GitHub Copilot	2. fejezet 5. fejezet	Ábrák készítéséhez program. Gyors prototipizálás. Numerikus és szimbolikus matematikai könyvtárak használatához segítség, pl. egyedi gradient descent IIR design program, 5.4 paragrafus.

A felsoroltakon túl más MI alapú eszközt nem használtam.

2025.05.25.