DIGITAL IMPLEMENTATION OF MUSICAL DISTORTION CIRCUITS BY ANALYSIS AND SIMULATION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

David Te-Mao Yeh
June 2009

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Julius O. Smith III) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Boris Murmann)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Jonathan Abel)

Approved for the University Committee on Graduate Studies.

Abstract

In the field of music technology, certain products stand out as having unique and desirable characteristics. Musicians will go to great lengths to find an instrument or piece of equipment that produces a particular sound or feel. As technology progresses, products that may have been popular in the past become obsolete, yet musicians still demand them for a specific artistic purpose. This research is concerned with preserving the sound of classic musical electronics, namely guitar amplifiers and distortion circuits, through modeling the circuits and emulating their sonic characteristics using efficient techniques to simulate audio circuits.

This work applies concepts from the field of physical modeling for musical synthesis, namely that of computing as much as possible beforehand to reduce the amount of work to be done in the time-critical runtime loop. This is done by an intricate understanding of the physics of the system, analyzing the system as much as possible beforehand, and simplifying the runtime computation to the bare minimum needed to recreate the behavior accurately.

Specifically, this work introduces the use of circuit analysis to derive and emulate digitally the signal paths of guitar distortion effects circuits. This work also explores in depth the use of numerical methods to simulate nonlinear circuits for real-time audio processing as a recursive nonlinear filter and develops a systematic method for deriving the nonlinear filter corresponding to a circuit. Finally, this approach will be applied to some basic building blocks of guitar distortion circuits.

Acknowledgement

First and foremost, I thank my family for providing the supportive environment that led me along this path. Throughout my formative years they have provided the resources for me to pursue my interests in music, making the sacrifices to purchase a beautiful grand piano, committing time to bring me to piano and violin lessons and show up to my orchestra concerts and recitals. I'm also thankful that they did not complain too much as I filled the house with guitars, synthesizers, guitar amplifiers and recording gear "for research purposes."

I thank Stanford Graduate Fellowship, National Defense Science and Engineering Graduate Fellowship, National Science Foundation Graduate Fellowship funding sources for making it possible for me to study something that would otherwise receive no financial support. I thank Professor Pierre Khuri-Yakub and his research group for mentoring my first few and difficult years at Stanford.

My reading committee spends a great deal of time deciphering the strange code of PhD theses, and I'm grateful for their effort. I thank my adviser Julius Smith for his constant support and enthusiasm and for allowing me the space to study and explore the topic at my own will. Many thanks also to my adviser Jonathan Abel who inspired me to choose this thesis topic, and for all those times spent at the cafe going over drafts of our paper submissions. Thanks also to Professor Boris Murmann who was always willing to take time out of his busy schedule to meet with students and give advice about research.

Over the course of many years at Stanford, I have formed close friendships with several individuals who have helped me get though the challenges of grad school. I thank my roommate for many years Paul for his patience in putting up with me and for all those stress-relieving late evening conversations about the important matters in life. Thanks also

to my other roommates throughout the years, Lawrence and Byron for their friendship and advice.

In the Khuri-Yakub group, I worked closely with Steve Zhuang, Ömer Oralkan, and Ira Wygant. We shared many struggles together, burning midnight oil to get awesome new results for the many conferences we attended. I've learned a great deal from my colleagues and have fond memories of our conference travels to Montreal, San Diego, and Europe.

From my time at the Helsinki University of Technology, I thank professors Matti Karjalainen and Vesa Välimäki for hosting me and inspiring me with their research. I also thank colleagues Jyri Pakarinen and Balázs Bank for their friendship and help in my work.

I have been at CCRMA for the majority of my time at Stanford now and I appreciate the environment we have here. I appreciate the staff who've kept things running smoothly, Carr, Nando, Tricia, Chrissie, Sasha. Thanks to Bill Verplank who's kindly let me use the Max lab for some electronics work. Many thanks to CCRMA director Chris Chafe for approving funding for conference travel support and to Jonathan Berger for his inspiration in doing audio work.

Finally I would like to acknowledge my office mates and DSP group buddies Tak, Tim, Kyogu, Woony, Aaron, Ryan, Gautham, Sook Young, Hiroko, Greg, Juhan, Nelson, and Ed for the special camaraderie we've had as researchers in music technology. CCRMA is a special place with unique challenges and benefits and I'm glad I've been able to share this significant part of my life with you all.

Finally I acknowledge my Lord Jesus Christ who is my inspiration and the purpose for all that I do.

Contents

Abstract **v**

Acknowledgement **vi**

1 Introduction **1**

1.1 Summary of the Thesis 2

1.2 Background 3

1.3 Digital Implementations of Guitar Distortion 4

1.3.1 Full effect models 4

1.3.2 Memoryless nonlinearity 5

1.3.3 Nonlinearity with memory 5

1.3.4 Volterra Series Expansion Representation 6

1.3.5 Simulation Methods 6

1.4 Issues in Digital Distortion Implementations 7

1.5 Distortion and Aliasing 8

1.5.1 Aliasing suppression by oversampling 8

1.5.2 Typical saturating nonlinearities in digital distortion 9

1.6 Summary 13

2 Circuit Analysis and Digital Filter Design **14**

2.1 Fundamental tools to model linear circuits 15

2.1.1 SPICE simulation 16

2.1.2 Continuous-time pole-zero analysis 16

2.1.3 Analysis of operational amplifier circuits 17

2.1.3.1 Ideal op amp approximation . . . 17
2.1.3.2 Non-inverting configuration . . . 17
2.1.4 Bilinear Transform of low-order transfer functions . . . 18
2.1.5 Other discretization methods . . . 20
2.1.6 Filter design techniques . . . 20
2.1.7 Prior work . . . 21
2.1.7.1 Black-box approach . . . 21
2.1.7.2 White-box approach . . . 22
2.2 Guitar Amplifier Tone Stack . . . 23
2.2.1 Properties of the tone stack . . . 23
2.2.2 Related work . . . 25
2.2.3 Discretization Procedure . . . 26
2.2.3.1 Symbolic Circuit Analysis . . . 26
2.2.4 Verification with SPICE circuit simulation . . . 28
2.2.5 Discretization by Bilinear Transform . . . 28
2.2.5.1 Implementation . . . 28
2.2.6 Analysis of Results . . . 29
2.2.6.1 Comparison of continuous- and discrete- time responses . 29
2.2.6.2 Implications of system poles and zeros . . . 33
2.2.7 Considerations for real-time implementation . . . 34
2.2.8 Approximate analysis with simplified circuits . . . 35
2.2.9 Physically informed digital filter architecture . . . 35
2.2.10 Conclusions . . . 36
2.3 Circuit Analysis of Distortion Pedal . . . 37
2.3.1 Emitter Follower buffers . . . 37
2.3.2 Single bipolar transistor transimpedance gain stage . . . 37
2.3.3 Op amp gain stage . . . 39
2.3.4 Diode clipper . . . 41
2.3.4.1 Diode clipper filter . . . 41
2.3.4.2 Implementation of diode clipper nonlinearity . . . 43
2.3.5 Tone stage . . . 43

2.3.6 Experimental Results . . . 45
2.4 Circuit Analysis of Overdrive Pedal . . . 48
2.4.1 High-pass filters . . . 48
2.4.2 Non-inverting op amp with diode limiter . . . 48
2.4.3 Tone stage . . . 50
2.4.4 Experimental Results . . . 51
2.4.5 Conclusions . . . 54

3 Solution of Nonlinear ODEs 55
3.1 Prior Work in Ordinary Differential Equation Solvers . . . 56
3.2 Numerical Methods . . . 56
3.2.1 Integration Formulas . . . 57
3.2.1.1 Forward Euler . . . 58
3.2.1.2 Backward Euler . . . 58
3.2.1.3 Implicit Trapezoidal Rule . . . 58
3.2.1.4 Backward Difference Formula Order 2 . . . 59
3.2.1.5 Explicit Runge-Kutta Order 4 . . . 59
3.2.2 Newton's Method for Solving Nonlinear Equations . . . 59
3.2.2.1 Homotopy . . . 61
3.2.2.2 Semi-Implicit Methods . . . 61
3.3 Accuracy of Numerical Integration Methods . . . 62
3.3.1 Local Truncation Error . . . 62
3.3.2 Stability . . . 63
3.3.2.1 Explicit Methods . . . 63
3.3.2.2 Implicit Methods . . . 65
3.3.2.3 Stiff Stability . . . 65
3.3.3 Considerations for Application to Audio Distortion Circuits . . . 65
3.3.3.1 Error . . . 65
3.3.3.2 Oversampling . . . 66
3.4 Case study: Diode clipper circuit . . . 66
3.4.1 Diode Clipper Equation . . . 67

3.4.2 Static Approximation of Diode Clipper . . . 70
3.4.3 Comparative Results . . . 70
3.4.3.1 Two-Tone Sine . . . 71
3.4.3.2 Explicit Methods: Stability Problems . . . 73
3.4.3.3 Single High-Frequency Sine . . . 74
3.4.3.4 Sine Sweep . . . 76
3.4.3.5 Comparison with Measurement . . . 77
3.4.4 Computational Cost . . . 78
3.4.5 Discussion . . . 82
3.4.5.1 Choice of Method . . . 82
3.4.5.2 Real-Time Considerations . . . 82
3.5 Explicit discretization of Moog . . . 83
3.6 Conclusions . . . 86

4 Numerical Simulation of General Lumped Systems 88
4.1 Wave Digital Filter . . . 89
4.1.1 Wave Digital Formulation . . . 90
4.1.1.1 Wave digital elements . . . 90
4.1.1.2 Wave digital adaptors . . . 91
4.1.1.3 Nonlinear wave digital elements . . . 93
4.1.1.4 Other wave digital filter considerations . . . 93
4.2 Modified Nodal Analysis / Transient Simulation . . . 94
4.2.1 Nodal Analysis . . . 94
4.2.2 Modified Nodal Analysis (MNA) . . . 95
4.2.2.1 Ideal voltage source . . . 96
4.2.2.2 An example . . . 96
4.2.3 Component-wise discretization . . . 97
4.2.3.1 Capacitors . . . 97
4.2.3.2 Inductors and mutual inductance . . . 98
4.2.4 Nonlinear devices . . . 99
4.2.5 Algorithmic construction of MNA . . . 99

4.3 State-space approaches with memoryless nonlinearity . . . 101
4.3.1 Iteration on currents (Classical K-method) . . . 101
4.3.1.1 Deriving the classical K-method . . . 102
4.3.1.2 Classical K-method summary . . . 103
4.3.1.3 Newton's method for classical K-method . . . 104
4.3.1.4 Homotopy . . . 105
4.3.2 Iteration on terminal voltages (VK-method) . . . 105
4.3.2.1 Newton's method . . . 106
4.3.2.2 Homotopy . . . 107
4.3.3 Discussion . . . 107
4.4 Nodal K-method . . . 108
4.4.1 Development of NK-method . . . 108
4.4.2 Element tables for NK-method . . . 110
4.5 DK-method - Discrete State-Space with Memoryless Nonlinearity . . . 112
4.5.1 Component-wise discretization . . . 112
4.5.2 Modified nodal analysis . . . 113
4.5.3 Solving the nonlinearity . . . 114
4.5.3.1 Newton's method . . . 114
4.5.3.2 Homotopy . . . 115
4.5.4 Summary . . . 116
4.5.5 DK-method Element Tables . . . 117

5 Applications of Selected Simulation Methods 118
5.1 Bright switch/filter . . . 118
5.2 Two-capacitor diode clipper . . . 119
5.2.1 WDF implementation . . . 121
5.2.2 K-Method implementation . . . 122
5.2.2.1 Simulation results . . . 123
5.2.2.2 Comparative discussion . . . 124
5.3 Common-emitter transistor amplifier with feedback . . . 124
5.3.1 Bipolar Junction Transistor (BJT) device model . . . 124

5.3.2 K-Method formulation . . . 125
5.3.3 Simulation results . . . 127
5.4 Common-cathode triode amplifier with supply bypass . . . 128
5.4.1 Triode device model . . . 129
5.4.2 K-method formulation . . . 129
5.4.3 Simulation results . . . 132
5.5 DK-method examples . . . 132
5.5.1 Single capacitor diode clipper derivation . . . 133
5.5.2 Common Emitter BJT amplifier . . . 135
5.5.2.1 Two-tone sinusoidal test . . . 136
5.5.2.2 High-frequency single-tone sinusoid . . . 136
5.6 Conclusions . . . 146

6 Conclusions **147**
6.1 Contributions . . . 148
6.2 Future work . . . 149
6.2.1 Nonlinear modeling . . . 149
6.2.2 Function approximation . . . 149
6.2.3 Further algorithm development . . . 150
6.2.4 Improved component modeling . . . 150
6.2.5 Extension to nonlinear musical acoustics simulation . . . 150
6.3 Final thoughts . . . 151

A Derivations of DK-Method Elements **152**
A.1 Discrete capacitors . . . 152
A.1.1 Trapezoidal Rule . . . 152
A.1.2 Backward Euler . . . 153
A.2 Discrete inductors . . . 153
A.2.1 Trapezoidal Rule . . . 153
A.2.2 Backward Euler . . . 155

Bibliography **157**

List of Tables

3.1 Cost comparison of methods for clipper ODE 79

4.1 Wave digital elements 91
4.2 Transient Modified Nodal Analysis construction table for two terminal devices 100
4.3 Transient Modified Nodal Analysis construction table for inductors and mutual inductance 100

List of Figures

1.1 Spectrum of hard clip aliasing 9
1.2 Distortion processor block diagram 10
1.3 Saturating nonlinearities 11
1.4 Log spectrograms of 20 Hz to 20 kHz sine sweeps of clipping functions. . . 12

2.1 Non-inverting op amp gain 18
2.2 '59 Bassman tone stack circuit 23
2.3 Vox top boost tone stack circuit 24
2.4 Fender blackface amp tone stack circuit 24
2.5 Comparison of tone stack magnitude response between analytical expression and SPICE 28
2.6 Comparison of tone stack phase response between analytical expression and SPICE 29
2.7 Comparison of filter magnitude response $f_s = 44.1$ kHz, $l = 0$ 30
2.8 Comparison of filter magnitude response $f_s = 44.1$ kHz, $l = 0.1$ 31
2.9 Comparison of filter magnitude response $f_s = 44.1$ kHz, $l = 1$ 32
2.10 Tone stack frequency response error 33
2.11 Analysis of the tone stack as two parallel sections 36
2.12 Block diagram of Distortion pedal. 37
2.13 Input buffer: Emitter follower circuit. 38
2.14 BJT transimpedance gain 38
2.15 Frequency response of BJT stage 39
2.16 Operational amplifier gain stage 40
2.17 Frequency response of op amp gain stage 41

2.18 RC low-pass filter with diode limiter 42
2.19 Static diode clipper nonlinear transfer curve 42
2.20 Tone circuit of Distortion pedal 43
2.21 Distortion pedal tone circuit frequency response 44
2.22 DS1 model verification: time response and output spectrum. 46
2.23 Sine sweep log spectrogram of DS-1 pedal. 47
2.24 Block diagram of Overdrive pedal. 48
2.25 Clipping stage of overdrive pedal. 49
2.26 Overdrive tone circuit. 50
2.27 Overdrive tone circuit frequency response 51
2.28 Model verification of overdrive pedal: time response and output spectrum. . 52
2.29 Sine sweep log spectrogram of overdrive pedal. 53

3.1 Stability regions of discretization methods 64
3.2 Partitioning scheme and block diagram for the Boss DS-1 circuit. 67
3.3 RC low-pass filter with diode limiter. 67
3.4 Linearized diode-clipper circuit. 68
3.5 Tabulated static nonlinearity of the diode clipper 69
3.6 Clipper ODE: Time-domain results for 110 Hz + 165 Hz input. 71
3.7 Clipper ODE: Peaks in spectra of responses to 110 Hz + 165 Hz input . . . 72
3.8 Clipper ODE: Forward Euler and fourth order Runge-Kutta 73
3.9 Clipper ODE: Time-domain waveforms for 15,001 Hz input 74
3.10 Clipper ODE: Magnitude responses to 15,001 Hz, 4.5 V input 75
3.11 Log spectrogram of diode-clipper response to sine sweep 76
3.12 Verification of clipper ODE: model time response to 220-Hz 77
3.13 Verification of clipper ODE model: spectrum of time response to 220-Hz . 77
3.14 Verification of clipper ODE model: log spectrograms of sine sweep 78
3.15 Clipper ODE: iteration count for exponential sine sweep 80
3.16 Clipper ODE: iteration count for power chord 81
3.17 Clipper ODE: iteration count for single note riff 81
3.18 Comparison of $\text{sech}^2(x)$ and $\cosh(x)$ 85

4.1 WDF two port 90
4.2 Three-port adaptors and corresponding circuit schematic 92
4.3 Example circuit to illustrate MNA 96
4.4 Discretized companion circuit for capacitor 98
4.5 Signal flow diagram of K-method filter 104

5.1 Schematic of the bright switch from guitar electronics. 119
5.2 WDF tree to implement the bright switch 119
5.3 Magnitude response of volume attenuator with bright switch 120
5.4 Schematic of the diode clipper with high-pass and low-pass capacitors. . . . 120
5.5 WDF tree of the two-capacitor diode clipper. 121
5.6 Simulated output of two-capacitor diode clipper 122
5.7 Schematic of the common-emitter amplifier with feedback. 126
5.8 Generic BJT device model. 126
5.9 Output of the common-emitter amplifier for sine input 127
5.10 BJT K-method nonlinearity $i_2 = f(\mathbf{g}(\mathbf{p}))$ 128
5.11 Schematic of the common-cathode triode amplifier. 130
5.12 Generic triode device model. 130
5.13 Plate voltage of common-cathode amplifier for sine input 132
5.14 Common-cathode triode amplifier K-method nonlinearity 133
5.15 Companion circuit of single capacitor diode clipper. 133
5.16 Explicit nonlinearity for single capacitor diode clipper. 135
5.17 BJT amplifier response to 110 and 165 Hz sinusoids 138
5.18 BJT amplifier response to 110 and 165 Hz sinusoids, harmonics 139
5.19 BJT amplifier response to 110 and 165 Hz sinusoids 140
5.20 BJT amplifier response to 110 and 165 Hz sinusoids, harmonics 141
5.21 BJT amplifier response to 1 kHz 142
5.22 BJT amplifier response to 1 kHz, spectrum 143
5.23 BJT amplifier response to 11 kHz 144
5.24 BJT amplifier response to 11 kHz, spectrum 145

Chapter 1

Introduction

In the field of music technology, certain products or implementations stand out as having unique and desirable characteristics. Musicians will go to great lengths to find an instrument or piece of equipment that produces a particular sound or feel. As technology progresses, components that may have been popular in the past become obsolete, yet musicians still demand them for a specific artistic purpose. Musical circuits may generate sound, or process sound, modifying its spectral features in a musically pleasing way. These circuits include oscillators, voltage controlled filters, dynamic range compressors, parametric equalizers, reverberation units, magnetic tape based echo, tube amplifiers, and guitar stompboxes. These electronics provide a spectral palette from which a musician can craft a musical portrait.

This thesis is concerned with the preservation of the sound of classic musical electronics, namely guitar amplifiers and distortion circuits. These circuits employ electronic devices that once were commonplace and low cost, yet are no longer manufactured due to the commodified nature of the electronics industry. Namely, these devices are vacuum tubes, carbon composition resistors, germanium bipolar junction transistors, diodes, and classic operational amplifiers.

Circuits are well characterized by their schematics and component models. If the devices are accurately modeled and the schematics are available, then a nearly exact simulation can be done digitally. If the simulation can be done efficiently with low latency, then

the unique sound produced by a specific circuit used in music can be preserved in digital form and remain in common use.

The rapid progress of digital computing has enabled real-time digital emulations of analog audio circuits. Digital implementations allow easy recall of presets during performance, enable automation of parameter settings during mixing, provide a variety of effects in a form factor with reduced size and weight relative to analog electronics, and reduce the dependence of the sonic performance on generational changes in analog technology. The application of digital signal processing technology to emulate vintage circuits have been termed in the literature *virtual analog*. Several commercial products emulating vintage guitar amplifiers and stompbox circuits form a category known as *amplifier modeling*.

1.1 Summary of the Thesis

This thesis expounds upon methods to simulate audio circuits efficiently. The techniques exploit particular characteristics of audio signals, especially their bandlimited nature. This work borrows concepts from the field of physical modeling for musical synthesis, namely that of computing as much as possible to reduce the amount of work to be done in the time-critical runtime loop. This is done by an intricate understanding of the physics of the system, analyzing the system as much as possible beforehand, and simplifying the runtime computation to the bare minimum needed to recreate the behavior accurately.

Specifically, this thesis covers basic device models and circuit analysis techniques, filter design based on continuous-time prototypes to simulate linear components, and numerical methods that can be used to simulate nonlinear dynamical systems. The signal path for these circuits can be decomposed into series or parallel stages, which may be linear or nonlinear.

In this introductory chapter we first survey the literature of digital effects to implement audio distortion. Then we analyze the oversampling requirements to minimize aliasing when strongly clipping with sampled systems.

Chapter 2 explores the decomposition of circuits into linear and nonlinear stages. Using examples such as the guitar tone stack, or operational amplifier based circuits, we propose that a stagewise analysis of a complete circuit can yield very accurate emulations with

little tuning required. This also offers a procedural approach to the design of amplifier modeling algorithms as an alternative to a purely parametric approach using a cascade of prefilter/memoryless-nonlinearity/postfilter blocks.

Chapter 3 describes a dedicated simulator for the diode clipper that has been developed to compare several numerical integration methods and their real-time feasibility. We found that implicit or semi-implicit solvers are preferred, although the prefilter / static-nonlinearity approximation for the diode clipper itself comes close to the actual solution, further validating the stagewise analysis approach.

Chapter 4 develops and compares two strategies in the literature for modeling nonlinear dynamic systems, in addition to traditional circuit simulation approaches. We also present a systematic method for deriving a nonlinear filter corresponding to any circuit, thus creating a special purpose circuit simulator for audio processing. Chapter 5 considers applications of this method to some canonical building blocks used in guitar distortion circuits.

1.2 Background

Analog guitar effects, whether based upon vacuum tubes or solid-state devices, consist of circuits that are accurately described in the audio frequency band by nonlinear ordinary differential equations (ODEs). A circuit simulator such as SPICE (Simulation Program with Integrated Circuit Emphasis) (Nagel, 1975) solves these systems of nonlinear ODEs to accurately predict their behaviors. However, SPICE simulation is computationally involved, so real-time effects processing requires a simplified approach. Often, the circuits can be approximately partitioned into stages, neglecting loading effects where possible (Yeh et al., 2007a), or even incorporating the loading effects as an equivalent circuit. Linear stages can be efficiently implemented by infinite impulse response (IIR) digital filters, although the remaining nonlinear ODEs may need to be solved by a numerical method or other approximation, usually employing a static nonlinearity.

Often guitar effects are digitized from a high level understanding of the function of the effect (Zölzer, 2002; Schimmel, 2003). Instead, we take a more detailed, physical approach to modeling guitar distortion. This approach has been adopted previously in the context of generating tube-like guitar distortion (Karjalainen et al., 2006). Furthermore,

our approach starts at a deeper level of detail with the equations that describe the physics of the circuit and is an alternative to obtaining the static transfer curves of a nonlinear system by measurement (Möller et al., 2002).

Stages are partitioned at points in the circuit where an active element with low source impedance drives a high impedance load. This approximation is also made with less accuracy where passive components feed into loads with higher impedance. Neglecting the interaction between the stages introduces magnitude error by a scalar factor and neglects higher order terms in the transfer function that are usually small in the audio band.

The nonlinearity may be evaluated as a nonlinear ordinary differential equation (ODE) using numerical techniques (Karjalainen and Pakarinen, 2006; Yeh et al., 2007b). However, often the solution of nonlinear ODEs is computationally intensive, and the differences can be subtle. Therefore in most implementations the nonlinearity is approximated by a static nonlinearity and tabulated. This can be justified on perceptual grounds. However, musicians tend to be very particular; therefore, we seek efficient ways solve nonlinear ODEs to improve the quality of digital implementations of guitar distortion.

1.3 Digital Implementations of Guitar Distortion

1.3.1 Full effect models

Practical digital models of full effect circuits are implemented either by emulating the effect with a simple algorithm tuned to give a similar sound, or by emulating the blocks along the signal path of the circuit.

Many commercial digital distortion pedals feature pre- and post-distortion filters surrounding a saturating nonlinearity. The filters are commonly multiband (three or four bands) parametric filters that are tuned to taste. This model simulates the distortion and filtering of a distortion circuit, by tuning parameters (filter frequencies, nonlinear characteristics) without emulating the physics and structure of the prototype circuit. The input / output characteristics of the circuit are modeled using a simple, efficient model.

An intuitive, more accurate method to approximate the behavior of a prototype distortion circuit involves using several nonlinearities and filters to imitate the signal flow of the

prototype (Kuroki and et al., 1998; Möller et al., 2002; Zölzer, 2002; Goetze, 2005; Yeh and Smith, 2006; Yeh et al., 2007a). Static nonlinearities are often approximated from the dynamic nonlinearities in the circuit by measuring input-output DC transfer characteristics of the nonlinear stages (Möller et al., 2002).

1.3.2 Memoryless nonlinearity

The simplest digital implementations of guitar distortion use a static nonlinearity, borrowing from classical waveshaping synthesis techniques (Arfib, 1979; Le Brun, 1979). The static nonlinearity is usually a lookup table, or a polynomial (e.g., spline fit) of an arbitrary function that saturates and clips. In the waveshaping technique, Chebyshev polynomials are used as a basis function for this nonlinearity, because they allow the control of individual harmonics when the input signal is a full-amplitude sinusoid (Le Brun, 1979). However, Chebyshev polynomials do not model intermodulation of multiple sinusoidal components.

Some methods have been proposed to use digital processing for greater control over distortion processing, including a perceptual map of distortion (Martens and Marui, 2003), processing different frequency bands with different distortions (Fernández-Cid et al., 1999), and an analysis of the spectral effect of piecewise-linear waveshaping curves (Schimmel and Misurec, 2007).

1.3.3 Nonlinearity with memory

Digital implementations of electric guitar distortion effects provide various ways to approximate the dynamic behavior of the nonlinear ordinary differential equations of a particular circuit block. A typical implementation employs a special case of the Volterra series expansion that uses a pre-filter, memoryless nonlinearity and a post-filter structure in various combinations (Zölzer, 2002; Doidic and et al., 1998; Abel and Berners, 2006). Designers then tune the parameters to simulate various kinds of distortion. The nonlinearity is assumed to be static (i.e., memoryless) for implementation efficiency. Although this assumption is false for most circuits, the approach often yields a perceptually satisfactory approximation as indicated by the market size for commercially available amplifier-modeling products.

Various approaches have also been attempted to incorporate memory into the nonlinearity. A sophisticated nonlinear system identification approach using a dynamic nonlinearity—one that depends on system state—has been patented (Gustafsson and et al., 2004). Another possibility is to use dynamic convolution, patented by Kemp (2006), approximating the nonlinear dynamic system by treating each sample level with a different transfer function or impulse response. This approach can only model a class of nonlinear system consisting of a static (memoryless) nonlinearity followed by linear filtering.

In many amplifier circuits, the bias point changes according to past input. Simplified approaches to imitate this effect include changing the offset into the static nonlinearity depending on a filtered signal of the input (Kuroki and et al., 1998) or the output, which is fed back (Schimmel, 2003; Karjalainen et al., 2006).

1.3.4 Volterra Series Expansion Representation

A nonlinear system with memory can be represented analytically as a Volterra series (Boyd and Chua, 1985). There has been work on forming finite-order Volterra series for simulating electronics (Schattschneider and Zölzer, 1999; Abel and Berners, 2006; Hèlie, 2006) . However, these are interesting only for low-order circuits, whereas for highly nonlinear systems, direct simulation by numerical methods is more computationally efficient. Even with many terms, Volterra series, which use polynomial models, do not converge sufficiently to represent accurately a clipping nonlinearity with large signal excursions.

1.3.5 Simulation Methods

The use of circuit simulation for real-time distortion processing was possibly first mentioned by Sapp et al. (1999). Santagata et al. (2007) also applied a numerical technique on a memoryless circuit, which amounts to an implementation of Newton's method with only one iteration per time step. Huovilainen (2004, 2005), (Välimäki and Huovilainen, 2006) effectively simulated the Moog filter and other effects circuits (e.g., phaser, flanger, and chorus effects) using the explicit Forward Euler method to generate a computable filter algorithm from the ODE of the circuit. Yeh et al. (2007a, 2008) investigated implicit ODE

methods for distortion circuits. Subsequently, research was published employing the explicit Runge-Kutta ODE solver to simulate a nonlinear voltage-controlled filter (Civolani and Fontana, 2008), and Forward Euler for the diode ring modulator (Hoffmann-Burchardi, 2008).

An alternative formulation to the ODE problem is to express the signals and states in terms of wave variables and apply component-wise, or local, discretization (Fettweis, 1986) at a uniform sample rate. This formulation is known as the Wave Digital Principle, and the resulting ODE solvers are called Wave Digital Filters (WDF). Karjalainen and Pakarinen (2006) simulated the ODE of a simplified vacuum tube preamplifier circuit for guitar distortion using a WDF.

1.4 Issues in Digital Distortion Implementations

Analysis of effects boxes shows that discrete analog circuits tend to use low-order filters. To keep costs low, circuits are designed with minimal component count, which limits filter order. This justifies the use of low-order IIR filter designs to emulate these analog filters.

Distortion causes harmonic and intermodulation products, which need to be simulated accurately. Modulation products from supersonic signal components are typically negligible owing to the natural spectral rolloff of input signals from guitars – assuming a triangle wave pluck excitation (Smith III, 2008), usually -40 dB/decade.

The nonlinearities cause an expansion of bandwidth through modulation that may lead to aliasing if the sampling rate is insufficiently high (Schattschneider and Zölzer, 1999; Zölzer, 2002). Consequently, typical digital implementations of distortion upsample by a factor of eight or ten, process the nonlinearities, and downsample back to typical audio rates (Zölzer, 2002; Doidic and et al., 1998). Frequency content tends to roll off with increasing frequency, and remaining aliases at oversampling factors of eight or above tend to be masked by the dense spectrum of guitar distortion. Additionally one can design approximations of the nonlinearity that mitigate the spectral expansion while being accurate in the audio band (Thornburg, 1999).

Because the filters in this work are derived from analog prototypes, upsampling also increases the audio band accuracy of the discretization by bilinear transform. An alternate

approach to upsampling would be to design low order filters so that the response at the Nyquist limit matches the continuous-time transfer function (Orfanidis, 1997; Berners and Abel, 2003).

For the purpose of distortion effect modeling, the frequency range of interest is from just above dc to 20 kHz. Features in the frequency domain above 20 kHz can be ignored, reducing the order of the filter required. Frequency features below 20 Hz must be retained, however, because intermodulation due to mixing of subsonic components with audio frequency components is noticeable in the audio band.

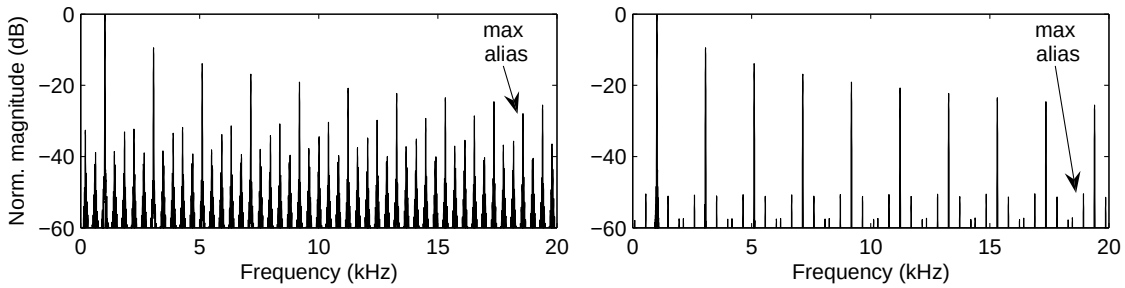
1.5 Distortion and Aliasing

1.5.1 Aliasing suppression by oversampling

Distortion causes mixing of the harmonic components of the input and expands the bandwidth of the output signal. To suppress aliasing, typically digital implementations of distortion process the nonlinearity after upsampling so that the bandwidth of the output remains below the Nyquist limit. This process of resampling to an N-times higher sampling rate, processing the nonlinearity, and resampling to the original rate is known as oversampling by $N\times$.

We give here an argument for why eight times ($8\times$) oversampling of the audio sampling rate $f_s = 44100\,\text{Hz}$ is typically sufficient for distortion effects. Guitar distortion circuits tend to have a monotonic input-output transfer characteristic. The strongest distortion that could occur would be an ideal comparator-like characteristic that essentially gives the sign of the input signal, and has in the extreme case infinite small-signal gain. The output would then look like a pulse-width modulated signal with the pulse widths varying according to the zero-crossing rate of the input signal. Assuming in the worst case the input to be a signal at the audio band limit, 20 kHz, the distortion transforms this into a 20 kHz square wave (Fig. 1.1), which has a Fourier series of

$$a_k = \begin{cases} \frac{\sin(k\omega_0 d)}{k\pi}, & \text{for } k \in \mathbb{Z}, k \neq 0 \\ 0, & \text{for } k = 0 \end{cases}$$



(a) No oversampling (b) Oversample by 8×

Figure 1.1: Spectrum of hard clip aliasing of 1021 Hz sinusoid. Largest alias is identified by arrow.

for the kth harmonic, $\omega_0 = 2\pi 20\,\text{kHz}$, $d = 12.5\mu\text{s}$. The envelope of this spectrum rolls off as $1/f$, or $-6\,\text{dB}$ per octave. The signal will fold over when it crosses into the audio band of the next spectral alias at $353 - 20 = 333\,\text{kHz}$, where a 20 kHz square wave would have been attenuated to $-24\,\text{dB}$ relative to the fundamental.

Guitar fundamental frequencies range from approximately 80 Hz to 1 kHz. Harmonic frequencies relative to this are greatly attenuated as well. For a 1 kHz fundamental frequency, the envelope of the aliasing signal will be $-50\,\text{dB}$ relative to the fundamental at the audio band edge. If no oversampling were used, $f_s = 44.1\,\text{kHz}$, the aliases could be as high as $-28\,\text{dB}$ relative to the fundamental. These are illustrated in Fig. 1.1, where a 1021 Hz sinusoid with amplitude 1.0 was amplified by 60 dB and clipped to 1.0, effectively generating square waves with a fundamental of 1021 Hz. The normalized output spectra are shown for no oversampling and $8\times$ with the first alias identified by an arrow.

1.5.2 Typical saturating nonlinearities in digital distortion

The preceding discussion was an extreme case with infinite small-signal gain. For practical purposes, clipping distortion is implemented as a finite preamplification gain followed by a saturating nonlinearity or clipping function. A typical distortion application would have parameters pre-gain a to adjust the intensity of the distortion effect, and post-gain g to normalize the perceived volume, leading to the structure in Fig. 1.2.

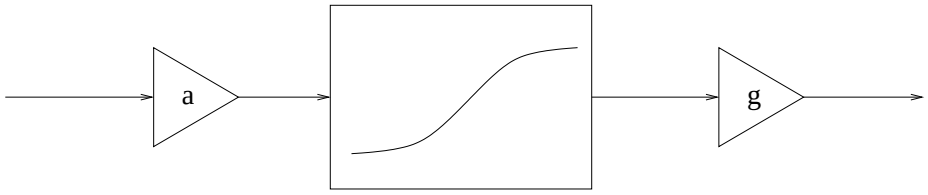


Figure 1.2: Typical structure of a memoryless nonlinearity in a distortion processor.

In this section, we evaluate different clipping functions shown in Fig. 1.3 experimentally at various oversampling factors by plotting their linear-frequency spectrograms for sine frequency sweep input from 20 Hz - 20 kHz in Fig. 1.4.

The static nonlinearities evaluated here are the sigmoid function or hyperbolic tangent

$$f(x) = \tanh(x)$$

inverse tangent, or arctangent

$$f(x) = \tan^{-1}(x)$$

a numerical form proposed by Abel (2006), which approximates hyperbolic tangent when $n = 2.5$

$$f(x) = \frac{x}{(1+|x|^n)^{1/n}}$$

the hard clip

$$f(x) = \begin{cases} x, & |x| \leq a \\ a, & otherwise \end{cases}$$

and the cubic soft clipper (Sullivan, 1990; Smith III, 2008)

$$f(x) = \begin{cases} -\frac{2}{3}, & x \leq -1 \\ x - \frac{x^3}{3}, & -1 \leq x \leq 1 \\ \frac{2}{3}, & x \geq 1 \end{cases}$$

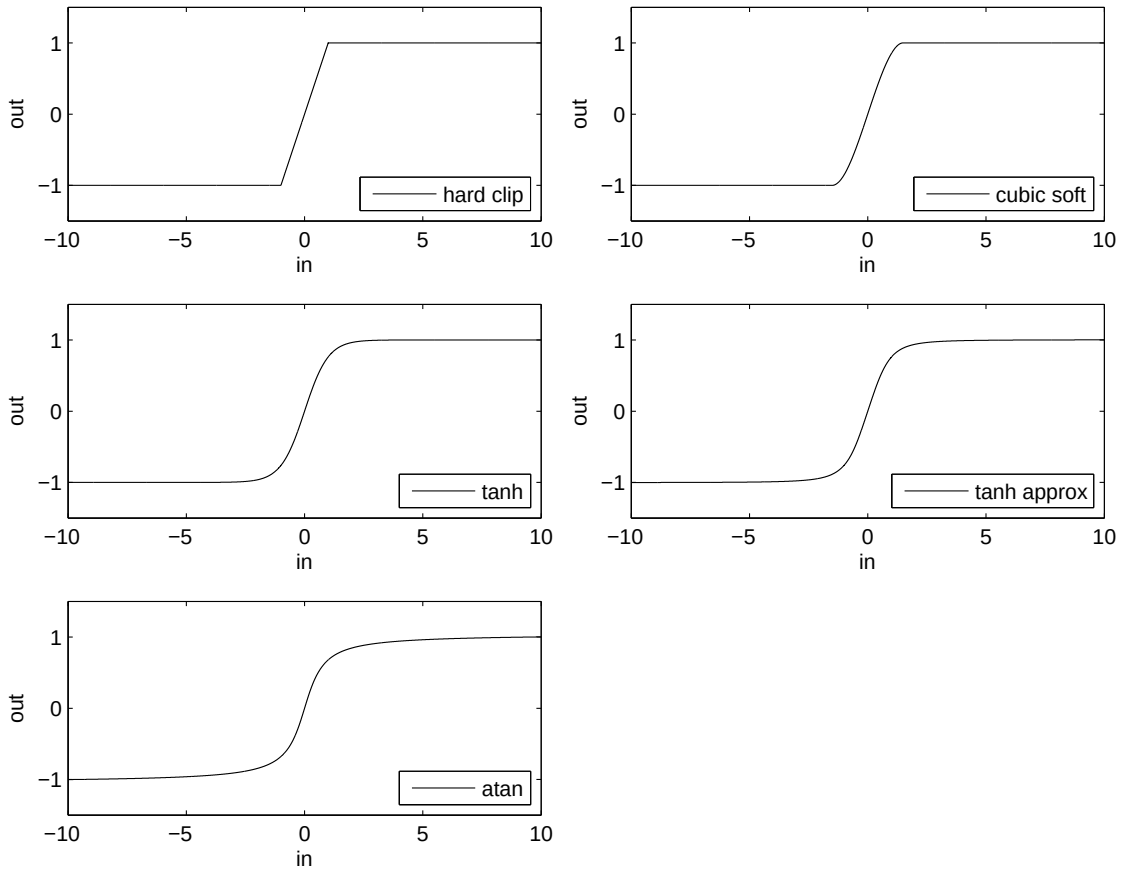


Figure 1.3: Saturating nonlinear functions for digital implementation of distortion. Hard clip, cubic soft clipper, tanh, Abel nonlinearity/tanh approximation, and atan. Clip to ± 1.0 with small signal gain of 10.0.

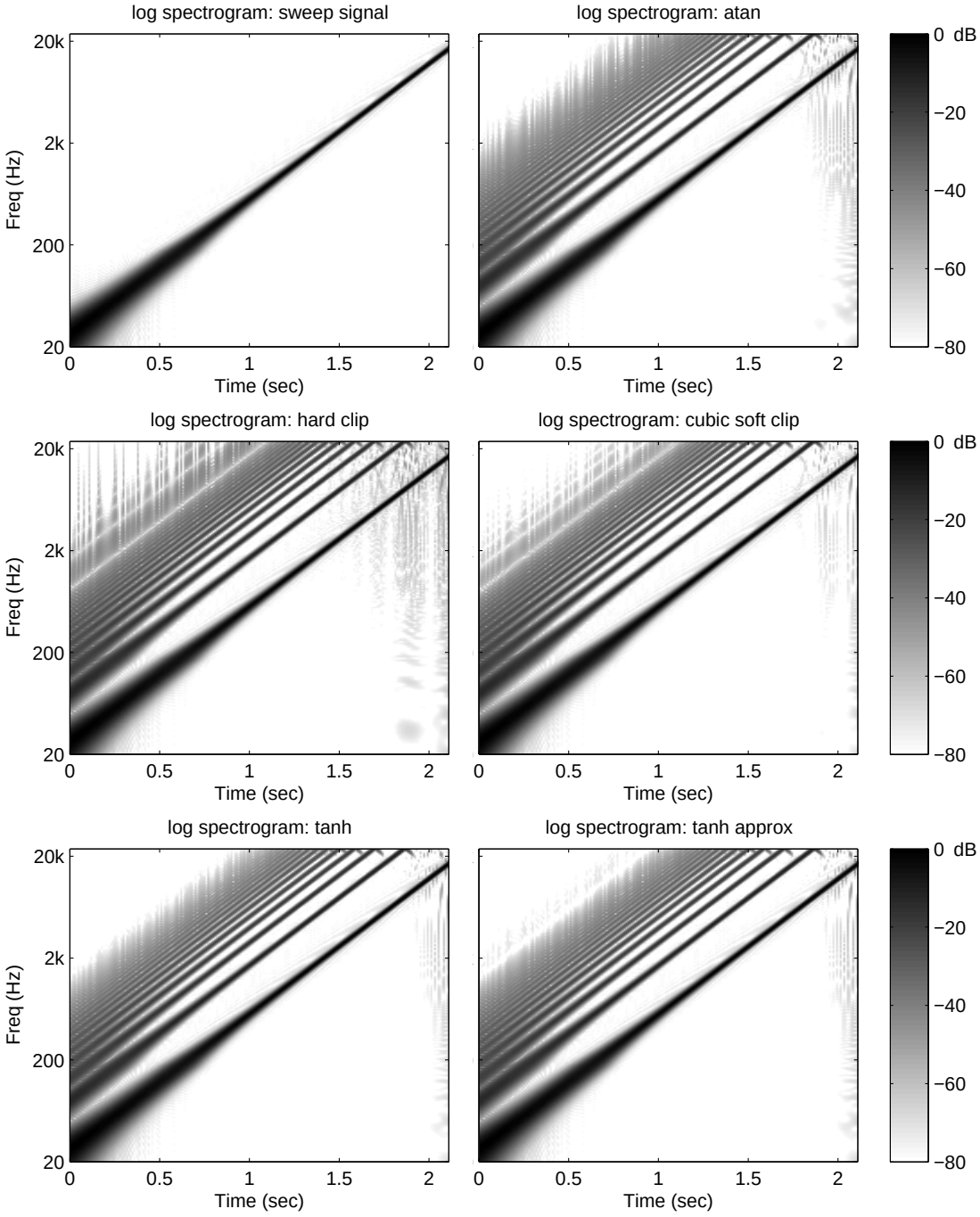


Figure 1.4: Log spectrograms of 20 Hz to 20 kHz sine sweeps of clipping functions.

With a limited sample rate, 20 to 40 dB gain on a full scale sinusoidal input can produce practically square-wave-like output, regardless of clipping function. All saturating nonlinearities approach a hard clip asymptotically as gain increases. Figure 1.4 shows the spectral differences between different distortion functions, which are subtle but perceptible at lower gain levels. For the high level of gain used here, the hard and soft cubic clippers are very similar in terms of output spectrum. Arctangent also saturates highly at large input levels, but the smooth transition produces a smoother spectral response. The hyperbolic tangent functions saturate more slowly at large input levels, thereby reducing high-order distortion and aliasing as compared to the arctangent function.

1.6 Summary

There exists a plethora of techniques to implement guitar distortion. This chapter has introduced some of the basic issues of guitar distortion. We find that for high quality distortion effects, an oversampling factor of $8\times$ is sufficient. In practice, lower oversampling factors may also be acceptable because the aliasing is masked by complex distorted guitar tones or noise.

The subsequent chapters take the approach of modeling the linear and nonlinear dynamical equations of guitar distortion circuits by discretization and numerical solution.

Chapter 2

Modeling of Circuits by Analysis and Digital Filter Design

This chapter considers the decomposition of the signal path of distortion circuits into linear and nonlinear blocks that can be implemented separately and recombined to form a digital emulation. Stages are partitioned at points in the circuit where an active element with low source impedance drives a high impedance load, and also where a high source impedance drives a low impedance load. This approximation is also made with less accuracy where passive components feed into loads with higher impedance. Neglecting the interaction between the stages introduces magnitude error by a scalar factor and neglects higher order terms in the transfer function that are usually small in the audio band.

In Sec. 2.1 we consider the discretization of linear circuits and filter design, which is a substantial area of research in itself because audio circuits can be highly parametric. Then we present several examples of this approach. First we delve into the analysis and implementation of the guitar amplifier tone stack. The results of this work were subsequently integrated into the LADSPA plugin suite of guitar effects, CAPS (Goetze, 2005). Next we evaluate the approach of design by analysis (Yeh et al., 2007a), deriving the stages of the Boss DS-1 distortion pedal and the Ibanez Tube Screamer TS-9 pedal. We find that the approach creates an algorithm that is very close to measured results from the original circuits with no tuning required.

2.1 Fundamental tools to model linear circuits

The characteristics of linear filtering greatly influence the tonal quality of electric guitar amplifiers. Often switches will be provided to allow a guitarist to choose between different component values in a circuit to vary its frequency response. Certain frequency responses are associated with particular genres or styles of music, and are often associated with specific guitar amplifier models.

For example, the tone stack, commonly found in many guitar amplifiers, especially those that derive from the Fender design, filters the signal of the guitar in a unique and non-ideal way. The user can adjust Treble, Middle, and Bass controls to modify the gain of the respective frequency bands. However, these controls are not orthogonal, and changing some controls affects the other bands in a complex way.

The digital filter simulates linear systems with high efficiency and accuracy. Because circuits of interest in digital audio effects are often linear, a linear transfer function well-characterizes these systems. Therefore, we seek to derive coefficients for digital filters that simulate these systems.These approaches fall into two main categories: digital filter design by discretization of an analog prototype transfer function, and digital filter design by optimization.

For the discretization approach, various means are possible to obtain an analog transfer function. One could do Laplace domain circuit analysis of the linear circuit to find an analytical expression for the transfer function, which can then be converted to digital form by one of various discretization techniques, including impulse invariance and the bilinear transform. The N-Extra Element Theorem (NEET) (Middlebrook et al., 1998) provides a method to derive an expression for the transfer function of a linear circuit. As an alternative to using brute force solution by computer algebra packages, the NEET potentially derives expressions in a more interpretable form. Audio circuits are often parametric, e.g., with volume and tone knobs, and the expressions derived in this approach provide a compact parametric digital implementation. Sometimes full analysis is too unwieldy and an alternative approach analyzing and taking into account the design of the analog circuit can yield efficient parametric, digital approximations.

For the optimization approach, impulse responses or transfers functions for the system are gathered by simulation or experiment. Various filter design approaches can minimize the error between the designed filter response and the target response, either in the time domain or the frequency domain, over the range of possible coefficient values. Often, the time domain problem is linear, accounts for phase effects in a simple way, and ultimately is easier to solve. This identification of the filter coefficients needs to be done for all combinations of the parameters, resulting in a large lookup table of coefficients that usually needs to be compressed by means of sparse sampling and interpolation. Typically from a user's perspective, highly approximate digital implementations sufficiently capture the essence of the original circuit.

The approaches described will be illustrated in detail for several cases studied over the course of this research.

2.1.1 SPICE simulation

For circuits that are difficult to analyze, SPICE simulation provides detailed numerical analysis. DC analysis in SPICE performs static sweeps of voltage or current sources to measure memoryless transfer curves. AC analysis finds the frequency response of a circuit linearized about an operating point. These responses can be imported into MATLAB and converted to digital filters as in Yeh and Smith (2006). SPICE also serves as a reference solver for numerical solutions of the time domain response for nonlinear ODEs.

2.1.2 Continuous-time pole-zero analysis

Linear circuits are described by rational transfer functions. For most low-cost audio circuits such as guitar effects, the transfer functions are typically low order. The location of real (not complex) poles and zeros can be identified on a log-frequency plot of magnitude in dB. In dB, it can be seen that the magnitude contributions of poles subtract and the magnitude contributions of zeros add. For the low-pass filter, at the pole frequency, the magnitude is 3 dB lower than at its low frequency asymptote. For the high-pass filter, the magnitude at the pole frequency is 3 dB lower that at its high frequency asymptote. Therefore, the

frequencies of well-separated real poles and zeros can be identified from the decibel magnitude response by looking for the 3-dB points. These frequencies can then be used to reconstruct the rational expression for the transfer function.

2.1.3 Analysis of operational amplifier circuits

Transfer functions can be easily found analytically for circuits with operational amplifiers (op amps).

2.1.3.1 Ideal op amp approximation

The ideal op amp approximation states that if negative feedback and infinite gain are present,

1. $V_+ = V_-$,
2. $I_+ = I_- = 0$

where V_+ is the voltage at the $+$ terminal of the op amp and V_-, the voltage at the $-$ terminal. I_+ and I_- are the currents flowing into the two terminals. These conditions do not hold if negative feedback is not present, for example, if V_o is not connected to V_- or if the op amp output is close to the supply voltages, causing it to clip.

2.1.3.2 Non-inverting configuration

An example of this analysis is done for the non-inverting op amp configuration shown in Fig. 2.1.

The ideal op amp rule gives $V_- = V_i$, so the current through Z_s is $I_s = V_i/Z_s$. Because $I_- = 0$, all the current flows across Z_f, so $V_0 = V_i + I_s Z_f = V_i + V_i/Z_s$. After algebraic manipulation, the transfer function is found to be $\frac{V_o}{V_i} = \frac{Z_s+Z_f}{Z_s}$. This results in a continuous-time transfer function if complex impedances are used for Z_f and Z_s:

$$A_v(s) = \frac{Z_f}{Z_s}\left(\frac{Z_s}{Z_f} + 1\right) \tag{2.1}$$

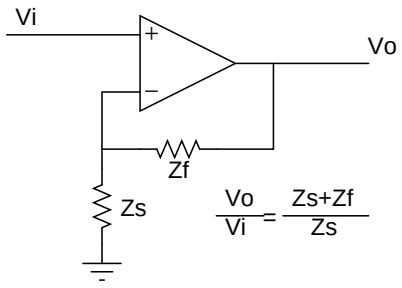


Figure 2.1: Non-inverting op amp gain

2.1.4 Bilinear Transform of low-order transfer functions

Once a continuous-time transfer function is obtained either by analysis or by inspection of the magnitude response, the bilinear transform

$$s = c\frac{1-z^{-1}}{1+z^{-1}} \tag{2.2}$$

can be used to digitize this filter. First- and second-order continuous-time systems are common, so their mappings are given below as directly implementable formulas used in this work.

The continuous-time system,

$$H(s) = \frac{b_n s^n + \ldots + b_1 s + b_0}{a_n s^n + \ldots + a_1 s + a_0}, \tag{2.3}$$

results in

$$H(z) = \frac{B_0 + B_1 z^{-1} + \ldots + B_n z^{-n}}{A_0 + A_1 z^{-1} + \ldots + A_n z^{-n}}, \tag{2.4}$$

where, for a first-order system, coefficients of $H(z)$ are

$$\begin{aligned} B_0 &= b_0 + b_1 c, \\ B_1 &= b_0 - b_1 c, \\ A_0 &= a_0 + a_1 c, \\ A_1 &= a_0 - a_1 c. \end{aligned}$$

For a second-order system, coefficients of $H(z)$ are

$$\begin{aligned}
B_0 &= b_0 + b_1 c + b_2 c^2, \\
B_1 &= 2b_0 - 2b_2 c^2, \\
B_2 &= b_0 - b_1 c + b_2 c^2, \\
A_0 &= a_0 + a_1 c + a_2 c^2, \\
A_1 &= 2a_0 - 2a_2 c^2, \\
A_2 &= a_0 - a_1 c + a_2 c^2,
\end{aligned} \tag{2.5}$$

For a third order system (e.g. a tone stack)

$$H(z) = \frac{B_0 + B_1 z^{-1} + B_2 z^{-2} + B_3 z^{-3}}{A_0 + A_1 z^{-1} + A_2 z^{-2} + A_3 z^{-3}} \tag{2.6}$$

the coefficients are

$$\begin{aligned}
B_0 &= -b_0 - b_1 c - b_2 c^2 - b_3 c^3, \\
B_1 &= -3b_0 - b_1 c + b_2 c^2 + 3b_3 c^3, \\
B_2 &= -3b_0 + b_1 c + b_2 c^2 - 3b_3 c^3, \\
B_3 &= -b_0 + b_1 c - b_2 c^2 + b_3 c^3, \\
A_0 &= -a_0 - a_1 c - a_2 c^2 - a_3 c^3, \\
A_1 &= -3a_0 - a_1 c + a_2 c^2 + 3a_3 c^3, \\
A_2 &= -3a_0 + a_1 c + a_2 c^2 - 3a_3 c^3, \\
A_3 &= -a_0 + a_1 c - a_2 c^2 + a_3 c^3 .
\end{aligned}$$

Often $c = 2/T$, where T is the sampling period, which is ideal for frequencies close to DC and corresponds to averaging the time derivatives of the current and future steps.

2.1.5 Other discretization methods

There are other ways to transform a continuous-time transfer function into discrete time. Similar to the bilinear transform are transforms based upon discretizations of the derivative as a finite difference. A low-order continuous to discrete mapping is the backward difference formula (Smith III, 2008), which provides good stability and frequency warping properties, while introducing numerical damping. Schneider et al. (1991) provide a comprehensive analysis of higher order s-to-z mappings based upon Adams-Moulton numerical integration methods. Orfanidis (1997) describes a method that applies constraints to the digital filter coefficients to meet prewarped analog specifications using the bilinear transform. Berners and Abel (2003) present an iterative method based on the bilinear transform to design digital shelf filters to meet specifications for gain at the Nyquist limit and resonant bandwidth and gain. One can also use methods based on sampling, such as impulse invariance, which transforms the exact sampled impulse response into a digital filter, or the Weighted-Sample method (Wan and Schneider, 2001), which samples the continuous-time response to a polynomial approximation of the input signal.

2.1.6 Filter design techniques

For circuits that are not parametric (i.e., no user controllable knobs) and with a preset sampling frequency, linear system identification or filter design techniques work well to model linear analog circuit responses.

In the black-box approach, the linear system is excited with a test signal that covers all the frequencies of interest. This is usually a frequency sweep of a low-amplitude sinusoidal input, or broadband white noise. A set of measurements is obtained for various settings of the parameters, which may be multivariate as for the low, mid, and high tone knobs of the guitar tone stack. Various techniques are well known for extracting a frequency response from these measurements (Abel and Berners, 2006; Rife and Vanderkooy, 1989; Smith III, 1983).

Once the impulse response is found, it can be used directly as a finite impulse response (FIR) filter to simulate the measured system. Because the original systems are typically low-order infinite impulse response (IIR) systems, it is computationally advantageous to

identify IIR filters corresponding to the measured response. The digital filter system identification process optimizes either the error in impulse response (time-domain identification) or frequency response (frequency-domain identification) over the set of digital filter coefficients, given a desired filter order.

The frequency-domain equation error method is implemented in MATLAB and Octave as invfreqs for continuous-time coefficients or invfreqz for direct discrete-time filter design, and allows perceptual weighting and smoothing to simplify high-order systems such as violin body resonances (Smith III, 1983). Alternatively, optimizing over the impulse response captures phase information and is a simpler, more robust formulation suitable for the low-order systems found in audio circuits. These methods include Prony's method (prony) and Steiglitz-McBride iteration (stmcb).

2.1.7 Prior work

2.1.7.1 Black-box approach

Because the parametrized filter coefficients are usually implemented as lookup tables, the patents covering linear modeling of amplifier components generally concern methods to reduce table size and storage costs in a practical implementation.

The Fender tone stack patent (Curtis et al., 2001) covers an active filter topology that replicates the range of frequency responses of a tone stack. Assuming this filter structure, system identification comprises obtaining coefficients for various knob settings by manual tuning to match the resulting frequency responses. The mapping from parameters to coefficients is compressed for implementation by sparse sampling (a suggested 5 points per knob) and 3-D linear interpolation of the coefficients.

The Softube patent (Gustafsson and et al., 2004) also describes multidimensional linear interpolation for the compression of mapping from parameters to filter coefficients. They improve the accuracy of classical linear interpolation and reduce the number of entries needed in the table by warping each parameter dimension using nonlinear mapping functions prior to interpolated table lookup. They also describe the decomposition of the resulting filter into a linear combination of Kautz basis filters (Ngia, 2001), a particular form of second-order digital filter, for stability in implementation. This is a special case of

the general technique in digital signal processing to ensure numerically stable filter implementations by decomposition into second-order sections.

A gray-box approach incorporating some insight into the structure of the circuit, described in a patent application (Gallien and Robertson, 2007), divides the tone stack into a parallel bank of two first order filters, one high pass and one low pass, which are weighted and summed. The filters are cleverly devised approximate equivalent circuits comprising resistors and capacitors that allow for implementation of the parameter mapping. The equivalent circuits are simulated and compared to a simulation of the full circuit to derive component values for the equivalent circuits and the filter weights so that the resulting response matches that of the actual circuit. The circuits, which are defined using capacitors and resistors, are taken into the discrete-time domain by the bilinear transform for digital implementation.

In summary, black-box approaches decide on a particular filter structure, and then decide on coefficients for that structure to match the response of the target system. Ad hoc mappings from parameter space to coefficient space parametrize the filter.

2.1.7.2 White-box approach

Yeh and Smith (2006) propose an analytical approach to the full tone stack circuit and suggest that the resulting parameter update equations are not prohibitively complicated. This approach derives the full third-order transfer function with no approximations for the filter by symbolic circuit analysis. Because the coefficients are described as algebraic functions of the parameters, this method is fully parametric. Yeh et al. (2007a) describe the same approach applied to filters based upon operational amplifiers. The tone stack for the Boss DS-1 distortion pedal was implemented by interpreting the analog filter as a weighted sum of a high-pass and low-pass functions and implementing the analogous structure digitally.

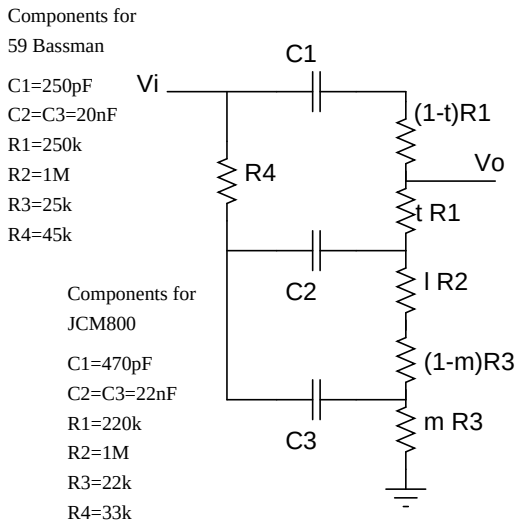


Figure 2.2: '59 Bassman tone stack circuit with component values.

2.2 Guitar Amplifier Tone Stack

2.2.1 Properties of the tone stack

As previously described, the tone stack is a passive filter composed of resistors and capacitors that allows the user to adjust bass, mid, and treble frequency bands. It is found in most guitar amplifiers, for example, the iconic Fender '59 Bassman, which, though originally intended for electric bass, soon became one of the most highly regarded amplifiers for electric guitar (Pittman, 2002). The full Bassman schematic can be found online (Fender Music Instruments Corp., 1999) and in guitar amplifier books, e.g., (Pittman, 2002). While other guitar amplifiers may vary slightly, in the Bassman type designs, the tone stack is found after the preamplifier stages and before the phase splitter. In good designs, the tone stack is preceded by a cathode follower to buffer the input and reduce variations in frequency response due to loading. Typically this presents a $1k\Omega$ load to the input and the phase splitter stage presents a $1M\Omega$ load to the output.

The Fender '59 Bassman tone stack circuit is shown in Fig. 2.2. The Treble, Middle, and Bass knobs are potentiometers, which have been modeled here as parametrized resistors. The Treble and Middle controls use linear potentiometers, while the Bass control uses a logarithmic taper potentiometer. In the following, t and m correspond to the Treble and

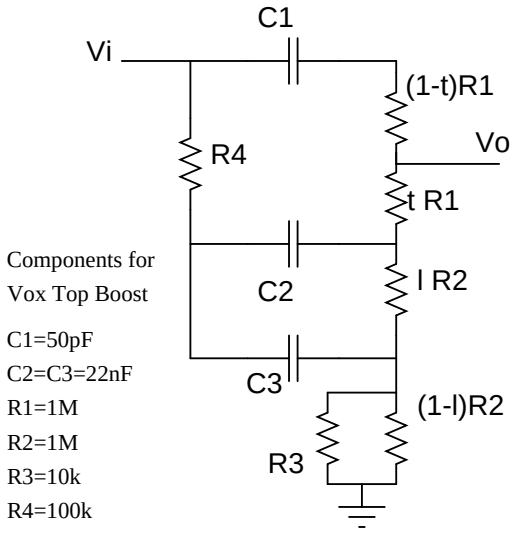


Figure 2.3: Vox top boost tone stack circuit with component values.

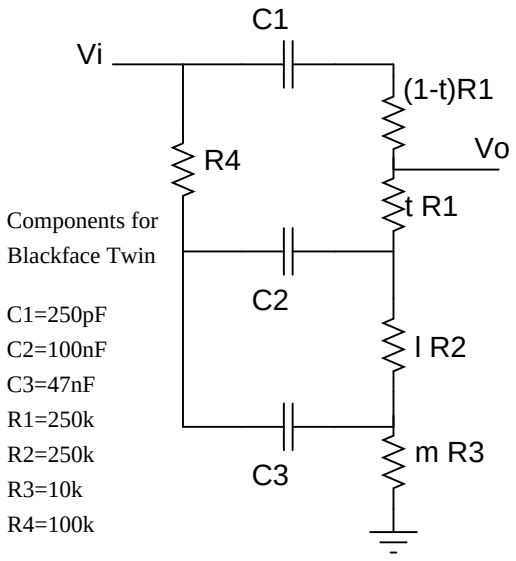


Figure 2.4: Fender blackface amp tone stack circuit with component values.

Middle controls and range in value from $[0, 1]$. The Bass control, l, also ranges from $[0, 1]$, but is swept logarithmically.

Typically the Bassman tone stack is also used in Marshall and other amps usually coupled to loudspeaker stacks. The Vox AC-30 variant of the tone stack is displayed in Fig. 2.3. There are only two knobs on this tone stack. The tone stack used in blackface Fenders and later amps is shown in Fig. 2.4. The potentiometers correspond to independent resistors in the schematic, making it the most flexible of the three tone stack styles, and allowing it to possibly emulate the other two variants.

2.2.2 Related work

Various methods to simulate the tone stack were described in Sec. 2.1.7. Line 6 models the behavior of the Bassman tone stack in their products as indicated in the BassPODxt manual. However, their implementation is proprietary knowledge. An open source guitar effects plug-in suite for Linux, CAPS (Goetze, 2005), uses shelving filters instead of the tone stack. An updated version of CAPS includes the tone stack described here.

Previous works have analyzed the tone stack using numerical circuit analysis techniques. This involves setting up the nodal equations as a matrix and inverting it or performing Gaussian elimination to find the solution. For example, the Tone Stack Calculator from Duncan Amps will plot the frequency response of various tone stacks given the control settings (Duncan Amps, 2006). Kuehnel (2005) in his book analyzed the mesh equations of the tone stack, using low frequency and high frequency circuit approximations. He also compares these simplified equations to the numerical solutions solved by inverting the matrix of the mesh equations. While the approximations make the circuit analysis more tractable, they do not reduce the order of the equations and do not make the discretization of the filter any easier.

Because the tone stack is a third-order passive network of resistors and capacitors (RC), its filter coefficients can be derived and modeled in the digital domain as shown later. The approach taken here is to find the continuous-time transfer function of the circuit analytically and to discretize this by the bilinear transformation. This provides a means of updating the digital filter coefficients based upon changes to the tone controls.

Passive filter circuits are typically suited to implementation as a wave digital filter (WDF) (Fettweis, 1986). This approach can easily model standard components such as inductors, capacitors, and resistors that are connected in series and in parallel. However, the tone stack is a bridge circuit, which falls into a category of connections that are neither parallel or series (Fränken et al., 2005). A bridge adapter with 6 ports can be derived (Fränken et al., 2005; Sarti and De Sanctis, 2009), but in general, for a 6-port linear system, there are 6×6 input/output relationships that must be computed. Efficient, parametric implementations for these circuits are not currently obvious.

2.2.3 Discretization Procedure

2.2.3.1 Symbolic Circuit Analysis

Because this is a relatively simple circuit, it is amenable to exact symbolic analysis by mathematical Computer Aided Design (CAD) software such as Mathematica (Wolfram Research, Inc., Champaign, IL). The filter coefficients can thus be found without any approximations. Performing symbolic nodal analysis on this circuit yields the following input/output transfer function $H(s) = V_o(s)/V_i(s)$, where V_o is the output and V_i is the input as in Fig. 2.2. The formulas for the coefficients are given below for direct implementation as a digital audio effect.

$$H(s) = \frac{b_1 s + b_2 s^2 + b_3 s^3}{a_0 + a_1 s + a_2 s^2 + a_3 s^3}, \tag{2.7}$$

where

$$
\begin{aligned}
b_1 &= tC_1R_1 + mC_3R_3 + l(C_1R_2 + C_2R_2) + (C_1R_3 + C_2R_3),\\
b_2 &= t(C_1C_2R_1R_4 + C_1C_3R_1R_4) - m^2(C_1C_3R_3^2 + C_2C_3R_3^2)\\
&\quad + m(C_1C_3R_1R_3 + C_1C_3R_3^2 + C_2C_3R_3^2)\\
&\quad + l(C_1C_2R_1R_2 + C_1C_2R_2R_4 + C_1C_3R_2R_4) + lm(C_1C_3R_2R_3 + C_2C_3R_2R_3)\\
&\quad + (C_1C_2R_1R_3 + C_1C_2R_3R_4 + C_1C_3R_3R_4),\\
b_3 &= lm(C_1C_2C_3R_1R_2R_3 + C_1C_2C_3R_2R_3R_4) - m^2(C_1C_2C_3R_1R_3^2 + C_1C_2C_3R_3^2R_4)\\
&\quad + m(C_1C_2C_3R_1R_3^2 + C_1C_2C_3R_3^2R_4) + tC_1C_2C_3R_1R_3R_4 - tmC_1C_2C_3R_1R_3R_4\\
&\quad + tlC_1C_2C_3R_1R_2R_4,
\end{aligned}
$$

$$
\begin{aligned}
a_0 &= 1,\\
a_1 &= (C_1R_1 + C_1R_3 + C_2R_3 + C_2R_4 + C_3R_4) + mC_3R_3 + l(C_1R_2 + C_2R_2),\\
a_2 &= m(C_1C_3R_1R_3 - C_2C_3R_3R_4 + C_1C_3R_3^2 + C_2C_3R_3^2)\\
&\quad + lm(C_1C_3R_2R_3 + C_2C_3R_2R_3) - m^2(C_1C_3R_3^2 + C_2C_3R_3^2)\\
&\quad + l(C_1C_2R_2R_4 + C_1C_2R_1R_2 + C_1C_3R_2R_4 + C_2C_3R_2R_4)\\
&\quad + (C_1C_2R_1R_4 + C_1C_3R_1R_4 + C_1C_2R_3R_4 + C_1C_2R_1R_3 + C_1C_3R_3R_4 + C_2C_3R_3R_4),\\
a_3 &= lm(C_1C_2C_3R_1R_2R_3 + C_1C_2C_3R_2R_3R_4)\\
&\quad - m^2(C_1C_2C_3R_1R_3^2 + C_1C_2C_3R_3^2R_4)\\
&\quad + m(C_1C_2C_3R_3^2R_4 + C_1C_2C_3R_1R_3^2 - C_1C_2C_3R_1R_3R_4)\\
&\quad + lC_1C_2C_3R_1R_2R_4 + C_1C_2C_3R_1R_3R_4,
\end{aligned}
$$

and where t is the Treble (or "top") control, l is the Bass (or "low") control, and m is the "middle" control. While these formulas appear complicated, the computational expense can be reduced by observing that only l, m, and t are variables; therefore, the coefficients to the terms containing these variables can be computed prior to runtime.

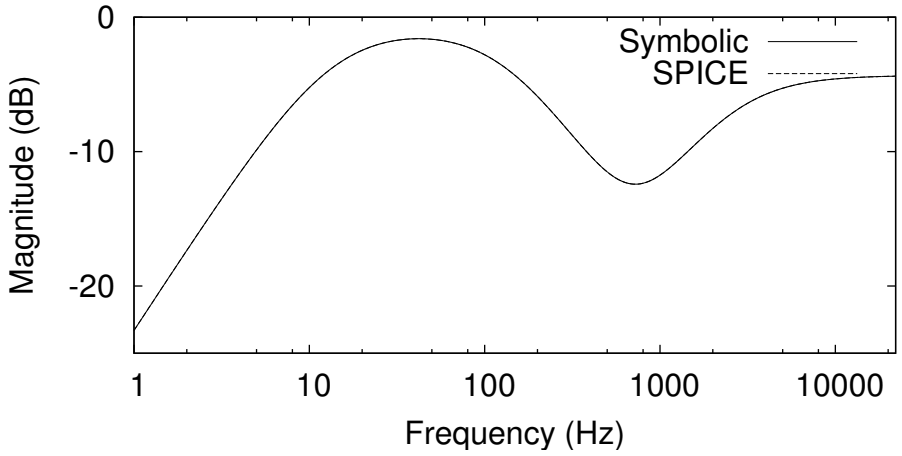


Figure 2.5: Comparison of magnitude response demonstrating no visible difference between analytical expression and SPICE for $t = l = m = 0.5$ (Treble, Low, and Mid parameters as in Fig. 2.2).

2.2.4 Verification with SPICE circuit simulation

To verify the correctness of this expression, Figs. 2.5 and 2.6 compare the frequency response with the result from the AC analysis of SPICE[1] simulation at the settings $t = l = m = 0.5$. The plots show an exact match, verifying that (2.7) is a complete and exact expression for the transfer function of the tone stack. SPICE simulation also determined that the frequency response was unaffected by the typical loading of $1\,k\Omega$ at the input and $1\,M\Omega$ at the output.

2.2.5 Discretization by Bilinear Transform

2.2.5.1 Implementation

Because this filter is relatively low order, it can be implemented as a transposed Direct Form II digital filter that runs in real time. This was coded in the C programming language as a LADSPA (Linux Audio Developer's Simple Plugin API)[2] plugin based upon the C* Audio Plugin Suite (CAPS).[3] The equations above were implemented directly for

[1] `http://bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/`
[2] `http://www.ladspa.org/`
[3] `http://quitte.de/dsp/caps.html`

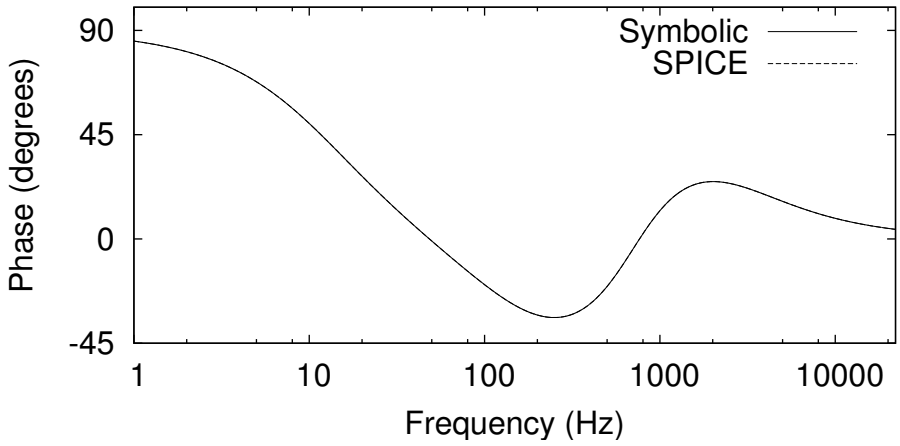


Figure 2.6: Comparison of phase response demonstrating no visible difference between analytical expression and SPICE for $t = l = m = 0.5$.

coefficient update. It was found that fading to new coefficients was not required to prevent artifacts in the output from changing the tone settings.

2.2.6 Analysis of Results

2.2.6.1 Comparison of continuous- and discrete- time responses

Figs. 2.7–2.9 show the discrete- and continuous-time transfer functions compared for various settings of t, m, and l. Each figure shows a different setting of l, and each sub-figure shows a different setting of m. In each plot, the treble control, t, was swept from 0.0001 to 0.5 to 0.9999 and can be distinguished by the corresponding increase in high frequency response.

The discretized filter used a sampling frequency of 44.1 kHz as typical for audio systems. The plots for $f_s = 44.1$ kHz show an excellent match through 10 kHz. The discrete and continuous plots are practically indistinguishable, with some deviations at the higher frequencies, as expected with the bilinear transform.

The errors, defined as the difference between the dB values of $H_a(s)$, the analog transfer function, and $H_d(z)$, the discretized transfer function, at each frequency, are plotted in Fig. 2.10 for $f_s = 20$ kHz and $f_s = 44.1$ kHz (abbreviated as 44k) for the settings of t, m,

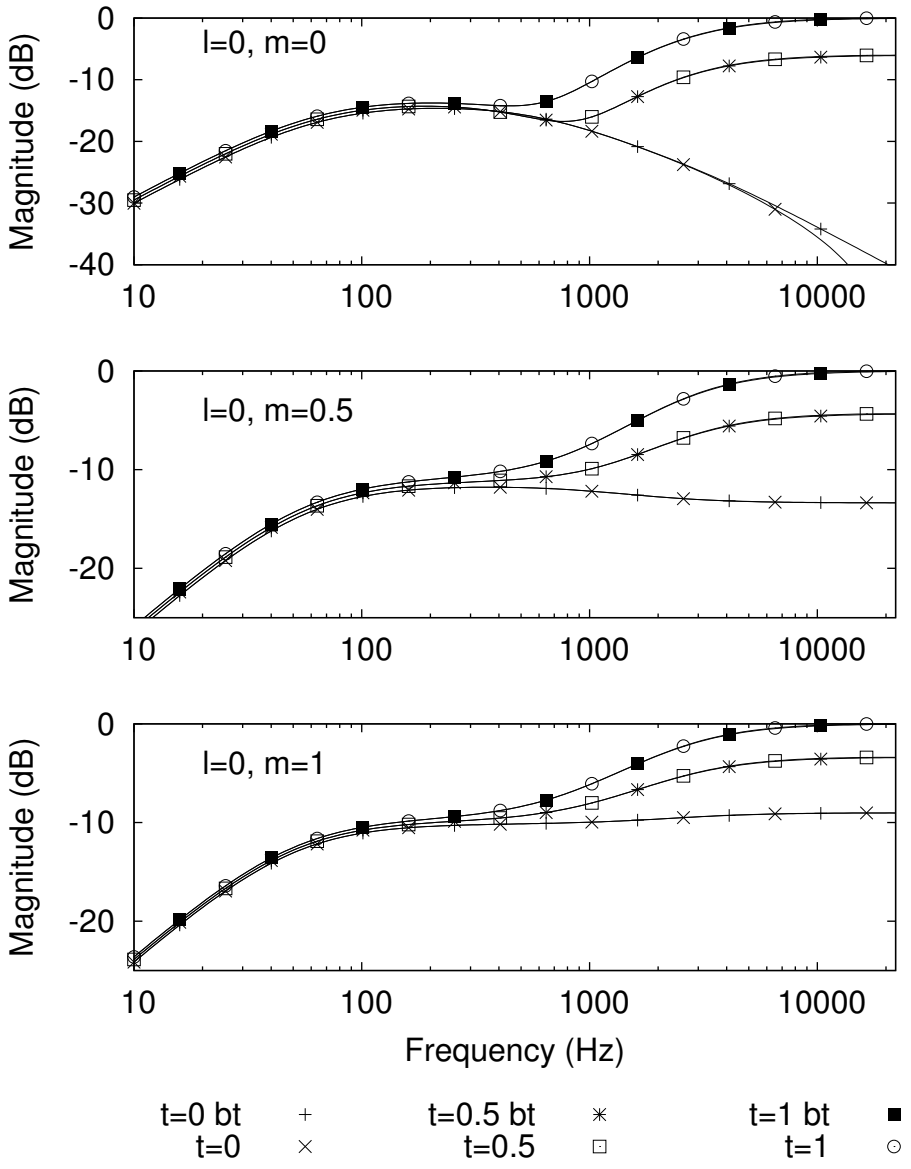


Figure 2.7: Comparison of filter magnitude response between original and discretized filters (bilinear transform (bt) with $f_s = 44.1\,\text{kHz}$), $l = 0$, where t, l, m are Treble, Low, and Mid controls.

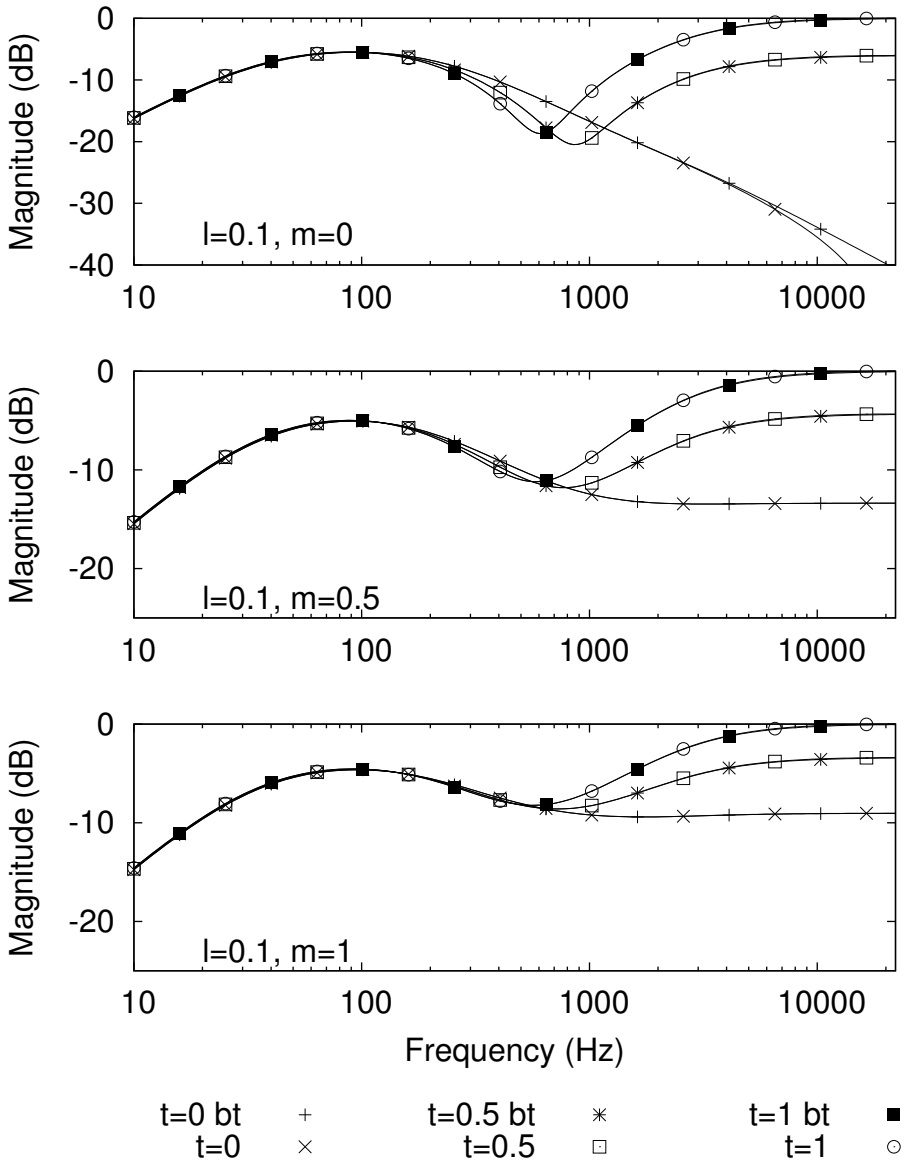


Figure 2.8: Comparison of filter magnitude response between original and discretized ($f_s =$ 44.1 kHz) filters, $l = 0.1$.

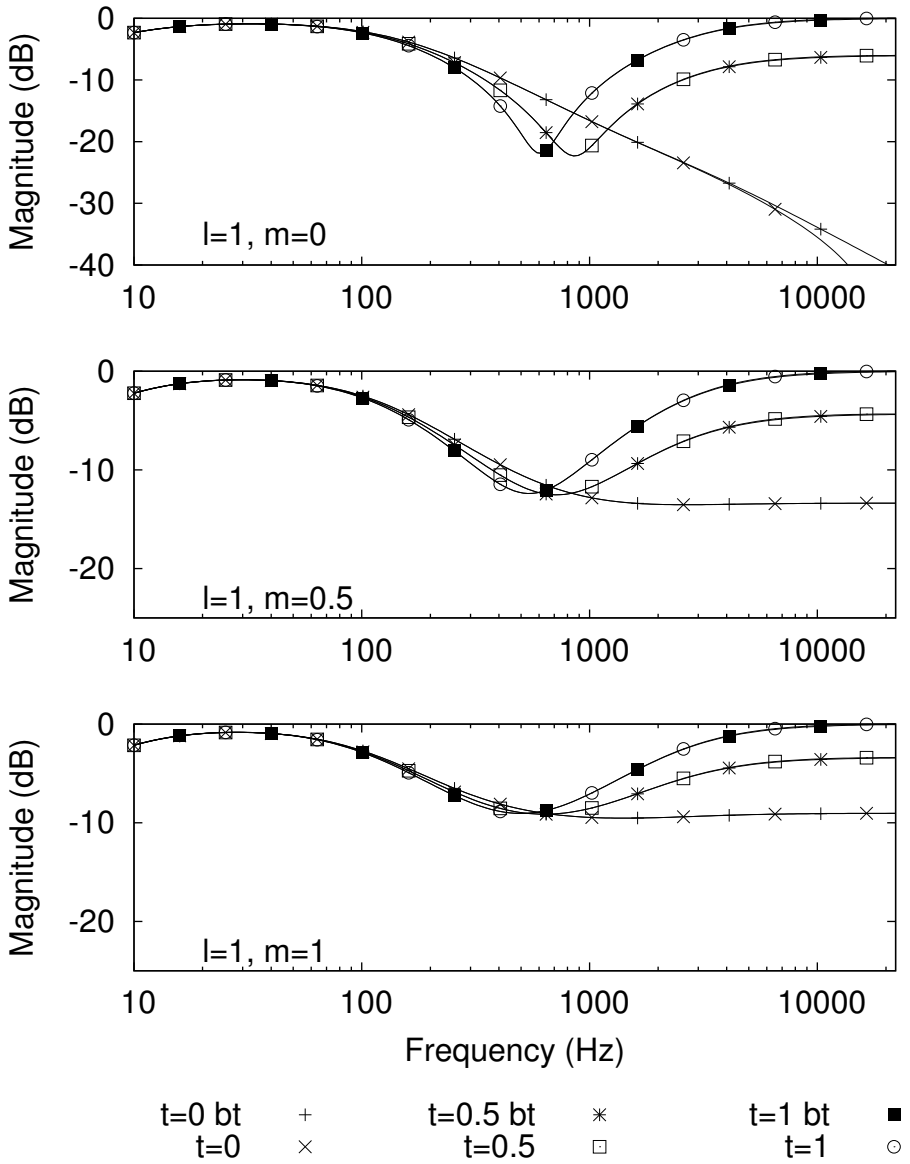


Figure 2.9: Comparison of filter magnitude response between original and discretized ($f_s =$ 44.1 kHz) filters, $l = 1$.

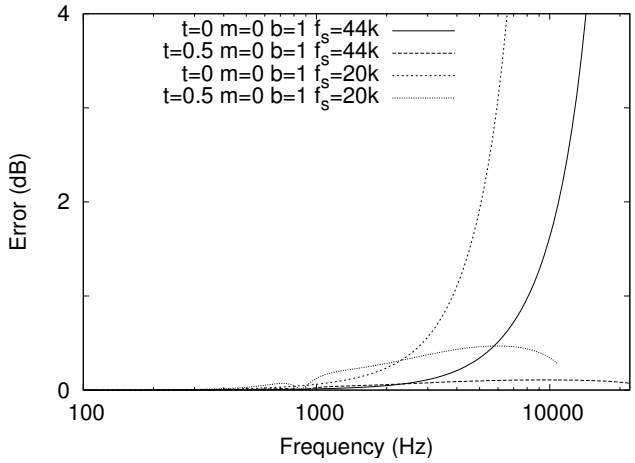


Figure 2.10: Error as difference between dB values of $H(s)$ and $H(z)$, for $f_s = 20$ and 44.1 kHz, and the noted tone settings.

and l that give the worst case results. The error is only meaningful for frequencies up through $f_s/2$.

The curves for $t = 0.5, m = 0, b = 1$ are characteristic of tone settings that give a high-pass response and have error within 0.5 dB for both cases of f_s.

The curves for $t = 0, m = 0, b = 1$ are characteristic of settings that give a low-pass response and exhibit a rapidly increasing error as frequency increases because the bilinear transform maps the null at infinite frequency to $f_s/2$. The error rises to 3 dB at roughly 6 kHz for $f_s = 20$ kHz, and at 13 kHz for $f_s = 44.1$ kHz. Because of the low-pass nature of these responses, the errors occur at frequencies where the magnitude is at least 10-20 dB lower than its peak value, making them perceptually less salient. Also, given that the frequency response of a typical guitar speaker is from 100 Hz to 6000 Hz, the deviations at higher frequencies would be inconsequential.

2.2.6.2 Implications of system poles and zeros for filter implementation

The plots exhibit the complex dependence of the frequency response upon the tone controls. The most obvious effect is that changing the *Middle* control also affects the treble response. The analytical form of the transfer function provides a way to find the poles and zeros of

the system as the settings are varied and gives insight into how the filter could be simplified to facilitate the implementation while maintaining accuracy.

Note that the tone stack is an entirely passive circuit composed of resistors and capacitors. This implies that the three poles of this system are all real. There is a zero at DC, leaving a pair of zeros that may be complex depending on the control settings. This also implies that the tone stack cannot be a resonant circuit although the pair of imaginary zeros can set up an anti-resonance, as evident in the notch seen in the frequency response plots.

Also note from (2.7) that none of the coefficients of the denominator depends on the treble control, t. The treble control therefore does not control the poles of the circuit but only adjusts the position of the zeros. This circuit can be decomposed into a weighted sum of terms that correspond to each pole by the partial fraction expansion. From this perspective, the treble control only affects the weighting of the different poles, but not the pole locations. The pole locations are controlled exclusively by the bass and middle knobs.

The existence of an analytical expression for the poles and zeros also informs the choice of c in the bilinear transform (2.2). The analytical expression allows the computation of frequency domain features such as local maxima or anti-resonance notches to be matched in the discrete-time domain.

2.2.7 Considerations for real-time implementation

The direct form filter with parameter update was implemented in LADSPA without any consideration given to efficient programming, and it runs in real time on an Intel P4 2.2 GHz processor. A third-order IIR implementation is an efficient yet accurate representation of the tone stack and should be preferred over higher-order filter designs. For the direct form implementation, one could amortize the cost of updating the coefficients over several audio frames because the tone controls need not respond as immediately as the audio signal.

Another efficient implementation of this filter would be to tabulate the coefficients for a robust digital structure, such as the lattice filter (Mitra, 2001). For the lattice filter, there are four coefficients that are functions of all three parameters and three that are functions of two parameters. A user does not need fine-grained stepping over the entire tone control space, and therefore a grid of 20 points in each dimension should suffice. This yields a

manageable table of 33,200 elements. Linear interpolation can provide an efficient way of further interpolating between grid points.

2.2.8 Approximate analysis with simplified circuits

Redrawing the tone stack in the form shown in Fig. 2.11 makes it easier to understand the design of the tone stack. Essentially, the tone stack is an entirely passive implementation of a fade between two parallel signal paths: one a DC blocker or high pass, and the other a cascade of a low-pass and high-pass filter. In the bottom path, the low-pass frequency should be higher than the high-pass frequency.

This interpretation clearly explains the shape of the magnitude response for this circuit: the midband notch results from the summation of the low frequency bandpass and the DC blocker. The weights of the bandpass and DC blocker outputs are controlled by the "Treble" knob, R1a and R1b. The low frequency shape is dominated by the response of the bandpass. The "Middle" knob R3 mainly controls the frequency of the low-pass and thus the upper cutoff frequency of the bandpass. The "Bass" knob R2 further adjusts the frequency of the high-pass section, or equivalently, the cut-on frequency of the bandpass.

2.2.9 Physically informed digital filter architecture

The insight offered by Sec. 2.2.8 provides a basis for designing a parametric digital tone stack with idealized characteristics. The treble control is a fader between a high-pass section and a second-order bandpass section. Owing to loading effects between the two sections, the quality-factor, or Q, of the notch also needs to be adjusted. The bandpass section is a cascade of a high-pass filter and a low-pass filter. Letting the user vary the weighting between high-frequency/low-frequency sections, the Q of the notch, the cut-on frequency of the high-frequency section and the cut-off/cut-on frequencies of the low frequency band allows this filter structure to emulate the response of any tone stack configuration, with potentially more intuitive and orthogonal controls.

Given this filter structure and a set of impulse responses from real circuits, one can formulate a system identification problem to solve for the above parameters automatically

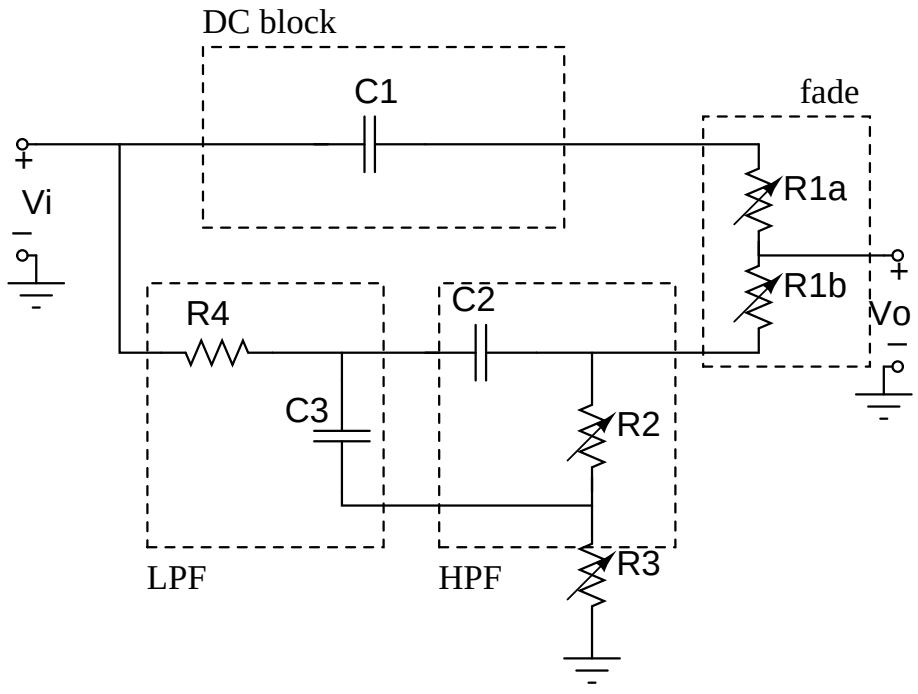


Figure 2.11: Analysis of the tone stack as two parallel sections

(Ljung and Soderstrom, 1983). Using time-domain impulse responses allows the problem to be formulated as least squares, e.g., stmcb in MATLAB.

Doing so, one can provide presets for a general parametric digital tone stack to emulate a selection of tasteful settings of the tone stack in iconic guitar amplifiers.

2.2.10 Conclusions

This work shows that the tone stack in the guitar amplifier can be parametrized exactly in the discrete-time domain and that the bilinear transform provides an outstanding frequency mapping for reasonable sampling rates. The transfer function for the physical tone stack was found as a function of its control parameters and component values using symbolic math software. This analysis provides a formula for updating the digital tone stack coefficients in a way that exactly emulates the physical circuit. The symbolic form of the transfer function also allows easy determination of the poles and zeros of the system and guides the design of a filter with simplified coefficients.

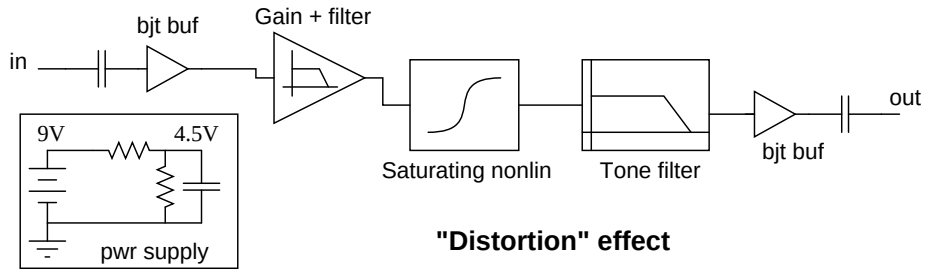


Figure 2.12: Block diagram of Distortion pedal.

2.3 Circuit Analysis of Distortion Pedal

The block diagram of the Boss DS-1 Distortion pedal (Roland Corp., 1980) is shown in Fig. 2.12. It is essentially a gain stage with a saturating nonlinearity sandwiched between filters. However, distortion from the bipolar transistor (BJT) emitter follower buffers and first gain stage are not negligible.

2.3.1 Emitter Follower buffers

A typical guitar pedal has an emitter follower (Fig. 2.13) at the input to buffer the signal from the guitar pickups, and another similar emitter follower at the output to drive the cable and subsequent load. The emitter follower topology is nominally linear in operation and flat in frequency response in the audio band. Typically it is used in conjunction with high-pass filters, whose cutoff frequency can be determined from the resistance and capacitance values. Here it is 3 Hz. The stage can be implemented as cascaded low-order high-pass filters. Implementation of high-pass filters is straightforward with the bilinear transform method of digitizing an analog prototype as described in Sec. 2.1.

2.3.2 Single bipolar transistor transimpedance gain stage

Gain can be provided by a single bipolar junction transistor (BJT) in a transimpedance gain topology shown in Fig. 2.14.

The frequency response is found from SPICE and digitized by finding the continuous-time poles and zeros, forming the transfer function and taking the bilinear transform. This

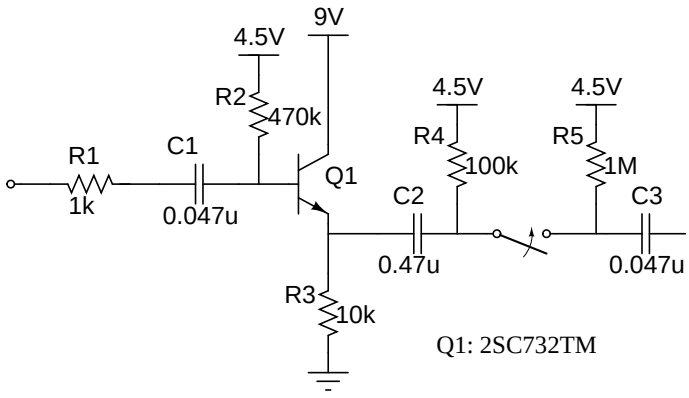


Figure 2.13: Input buffer: Emitter follower circuit.

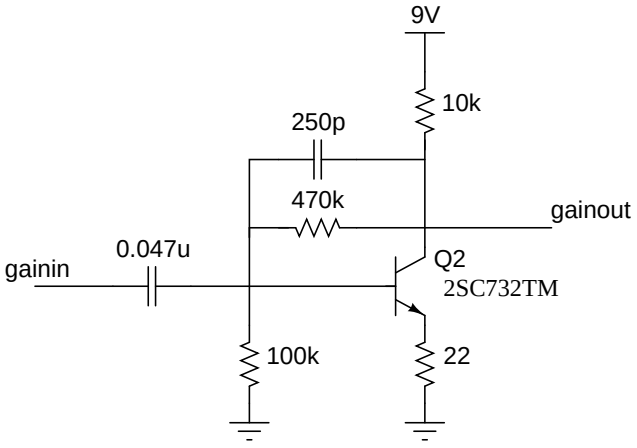


Figure 2.14: BJT transimpedance gain

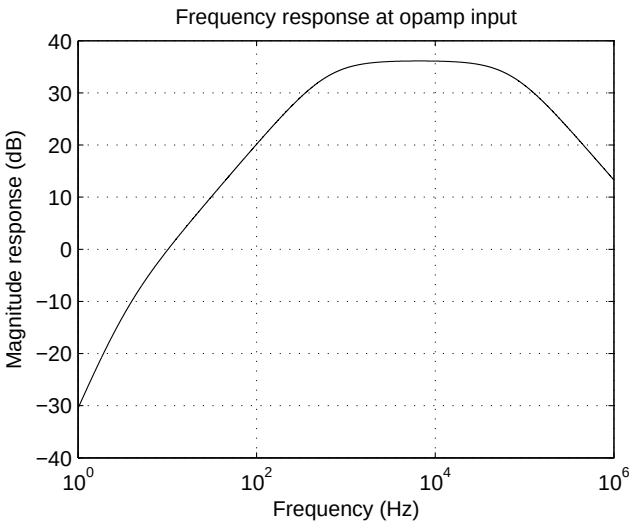


Figure 2.15: Frequency response of BJT stage

stage shows 36 dB of bandpass gain (Fig. 2.15). There are two zeros at DC, one pole at 3 Hz, one pole at 600 Hz, and another at 72 kHz, which is ignored because it is well outside the audio band. A transfer function is formed directly in (2.8):

$$H(s) = \frac{s^2}{(s+\omega_1)(s+\omega_2)}, \tag{2.8}$$

where the numerator is the product of two zeros at DC, s, and the denominator is the product of the poles at $\omega_1 = 2\pi 3$ and $\omega_1 = 2\pi 600$.

The bilinear transform applied to $H(s)$ with a sampling rate $f_s = 44.1$ kHz gives a second-order digital filter whose coefficients can be found using (2.5).

This stage introduces significant nonlinearity at large inputs, but this is neglected for now.

2.3.3 Op amp gain stage

Non-inverting op amp stages are common in guitar circuits because they minimize loading on the preceding stage. To analyze the circuit in Fig. 2.16 impedances are used in (2.1).

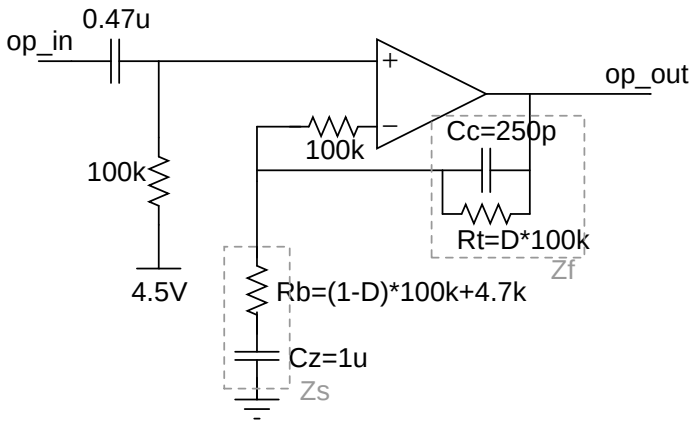


Figure 2.16: Operational amplifier gain stage

The final transfer function in factored form is given by (2.9).

$$H(s) = \frac{(s+\frac{1}{R_t C_c})(s+\frac{1}{R_b C_z})+\frac{s}{R_b C_c}}{(s+\frac{1}{R_t C_c})(s+\frac{1}{R_b C_z})} \tag{2.9}$$

where $R_t = D \cdot 100\text{k}\Omega$, $R_b = (1-D)100\text{k}\Omega + 4.7\text{k}\Omega$, $C_z = 1\mu\text{F}$, and $C_c = 250\,\text{pF}$. Capacitor C_z blocks DC to prevent the output from saturating in the presence of DC offset, while C_c stabilizes the op amp and contributes a low-pass pole. D ranges between $(0,1)$ and is the value of the "DIST" knob that controls the amount of gain before saturation and therefore the intensity of the distortion.

The frequency response is shown in Fig. 2.17 for values of D from 0 to 1 in increments of 0.1. This is a second-order stage than can be digitized directly by the bilinear transform, forming a second-order section with variable coefficients. The frequency response of this stage depends on the "DIST" knob. Notice that the frequency response at half the audio sampling rate, $|H(f = 22050)|$, is not zero and considerable warping will take place without oversampling or the filter design method by Orfanidis (1997).

This transfer function can be discretized by the bilinear transform (2.5), which also preserves the mapping of the "DIST" parameter.

The op amp provides the main nonlinearity of the Distortion effect. To first order, the op amp hard clips the signal at $V_{dd}/2$. In reality the op amp response is much slower because it is open loop and needs to recover from overdrive. It is also typically asymmetrical

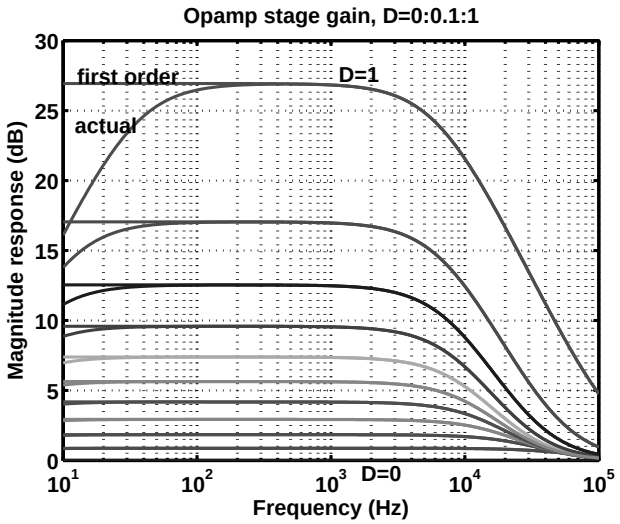


Figure 2.17: Frequency response of op amp gain stage, $D = 0 : 0.1 : 1$

in behavior, leading to significant even-order harmonics where otherwise only odd-order harmonics are expected. Refinements of the op amp clipping model can be based upon the macromodeling technique as done in SPICE to speed up simulations (Boyle et al., 1974). A black-box approach, the macromodeling technique emulates the input/output behavior of the op amp instead of simulating the behavior of its internal devices.

2.3.4 Diode clipper

2.3.4.1 Diode clipper filter

Following the op amp clipper in Fig. 2.16 is a RC low-pass filter with a diode limiter across the capacitor (Fig. 2.18). The diode clipper limits the voltage excursion across the capacitor to about a diode drop in either direction about signal ground. Section 3.4.1 derives in detail the ODE that describes the input output behavior of this circuit.

This is not a memoryless nonlinearity because it is a low-pass filter whose pole location changes with voltage. Fig. 2.19 depicts the input-output characteristic, which can be described as a "clipping" function, along with various analytic approximations based on hyperbolic tangent and arctangent. The curves are normalized so that the slopes about

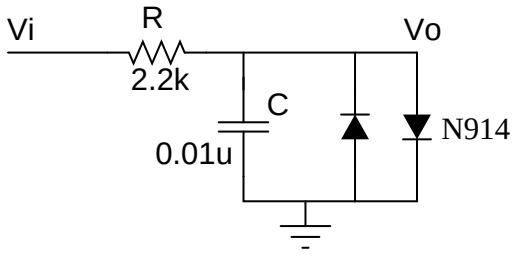


Figure 2.18: RC low-pass filter with diode limiter

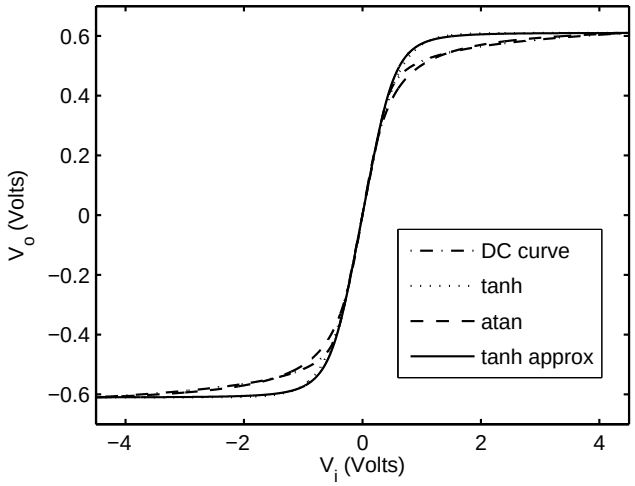


Figure 2.19: Static nonlinear transfer curves of the diode clipper compared: tabulated, tanh, arctan, approximation to tanh

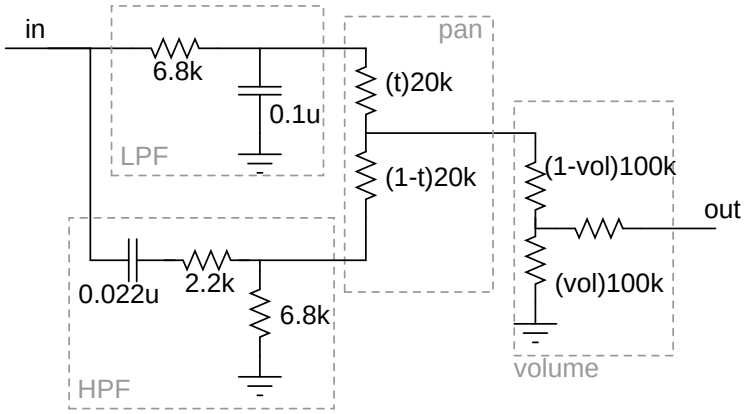


Figure 2.20: Tone circuit of Distortion pedal

$V_i = 0$ match visually and V_o at the extremes match. At high amplitude levels, the audible differences between the clipping functions are subtle.

2.3.4.2 Implementation of diode clipper nonlinearity

For efficiency, this nonlinearity is approximated as static, and the DC transfer curve is computed by setting $\frac{dV_o}{dt} = 0$ in (3.15), and tabulating the function $V_o = f(V_i)$ by Newton iteration. A nonuniform sampling of the input to output transfer curve is used that maintains a constant error percentage or signal to quantization noise ratio. The rationale for this is that, at small amplitudes, the curve is most linear with the highest gain, and most susceptible to quantization noise. At high levels, the nonlinearity is compressive, reducing the gain and quantization error. A logarithmic sampling with a floor about zero is chosen. Linear interpolation is used to further reduce quantization noise.

2.3.5 Tone stage

The tone stage (Fig. 2.20) is a highly interconnected passive network that cannot be accurately separated. However, an analysis of the circuit shows its design intent, and the error due to separating the blocks is less than that due to component tolerance in an actual circuit.

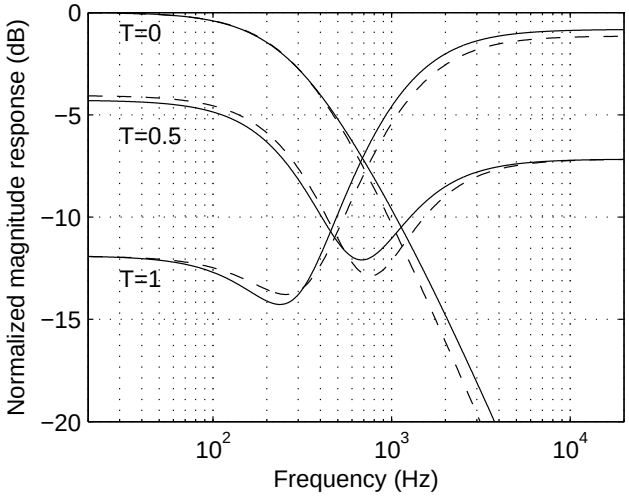


Figure 2.21: Distortion pedal tone circuit frequency response. Solid line is actual. Dashed line is digitized implementation.

This circuit involves a fade between high-pass filter and low-pass filter blocks. The fading affects the cutoff frequencies of the filters, but this effect is small. A digitization of this circuit can capture the essence of its operation, which is a loudness boost: a V-shaped equalization as commonly observed for tone circuits intended for electric guitars (Karjalainen et al., 2006; Yeh and Smith, 2006).

A full analysis is straightforward but tedious, so AC analysis is performed in SPICE, and the corner frequencies found graphically. The weightings for the fade are also determined by simulation. The high-pass corner frequency is $f_{hpf} = 1.16\,\text{kHz}$ and the low-pass corner frequency is $f_{lpf} = 320\,\text{Hz}$.

This is implemented digitally as a weighted sum of first-order high-pass and low-pass filters discretized by the bilinear transform rather than discretizing the complete transfer function. This simplification eliminates time-varying filters and the computation to update the coefficients, using static coefficients instead. Modeling a user-controlled parameter with greater accuracy is unnecessary because a user would not likely notice the difference in filter response.

The magnitude response of the original circuit is compared with the MATLAB approximation in Fig. 2.21. The responses are very similar with < 1 dB error in most cases.

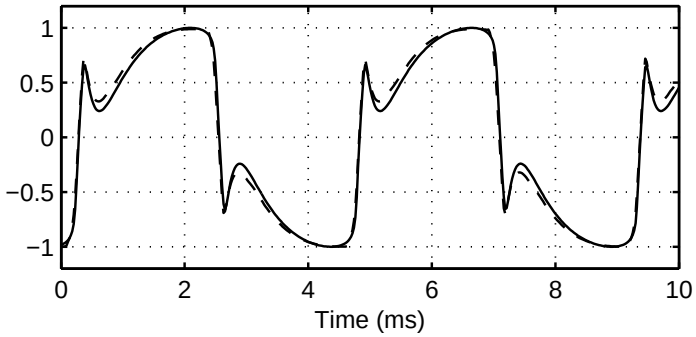
2.3.6 Experimental Results

An actual Boss DS-1 Distortion pedal was compared to the digital emulation for a 220 Hz sine signal with amplitude of 100 mV, and an exponential sine sweep. The settings on the actual pedal are adjusted until the spectrum resembles that of the digital version for the sine input. Adjustments were made approximately to match the difference in magnitude of the first two harmonics, and to match the position of notches in the frequency domain.

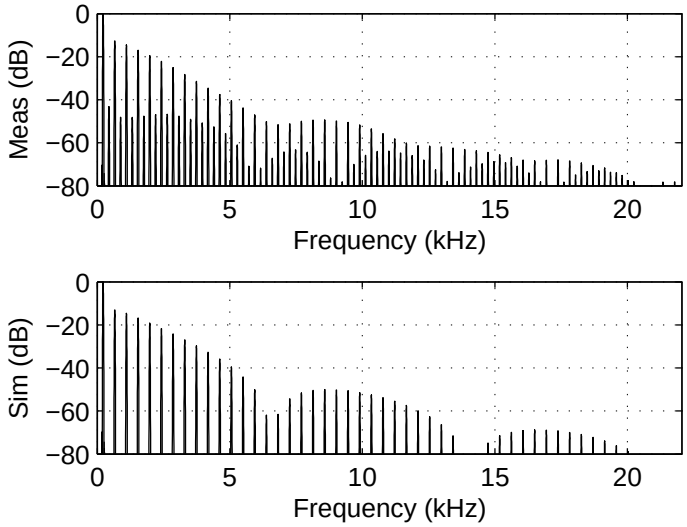
The time waveforms and magnitude spectra for the single-frequency excitation are shown in Fig. 2.22. The sinusoidal sweeps are represented by a log-frequency spectrogram (Brown, 1991) in Fig. 2.23 with 80-dB dynamic range.

The waveforms show a general similarity. The spectrograms indicate that frequency equalization is close. The measured spectra exhibit weak even-order harmonics (less than -40 dB or 1%) due to an even-order nonlinearity that is not modeled in this digital implementation. The dominant even-order nonlinearity is due to the BJT gain stage of Fig. 2.14 and will be considered in Sec. 5.3. The emulated version using the simplified algorithms sounds slightly brighter than the actual pedals because the static nonlinearity does not filter as much as a physical nonlinear filter, which will be addressed in Sec. 3.4. One artifact of using a static nonlinearity is visible in Fig. 2.23b as the presence of more pronounced notches in the spectrogram. In the physical case, the memory in the nonlinearity effectively provides feedback around the nonlinearity (see Sec. 4.3.1.2), causing more thorough mixing of signal components, and fills in the notches of the memoryless case.

The digitally emulated result also deviates from the measured one because there was no attempt to calibrate the model to the actual pedal with its particular component values and parameter settings. It is more representative of a circuit whose components happen to be exactly the values as in the schematic.

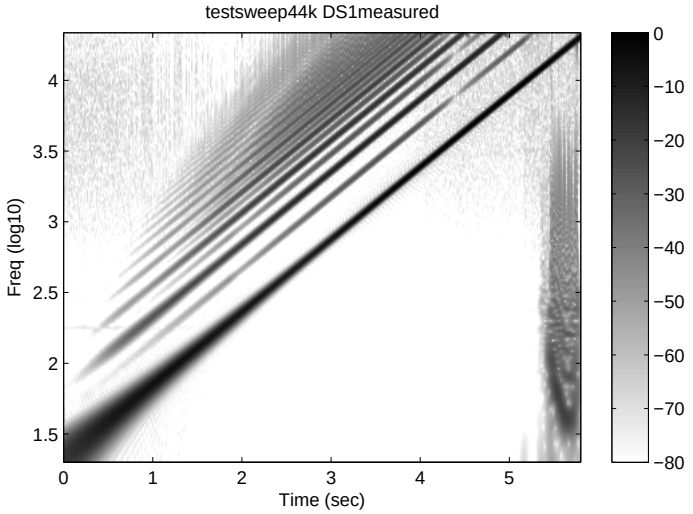


(a) Time response to 220 Hz sine, measured distortion pedal (dashed) and algorithm (solid)

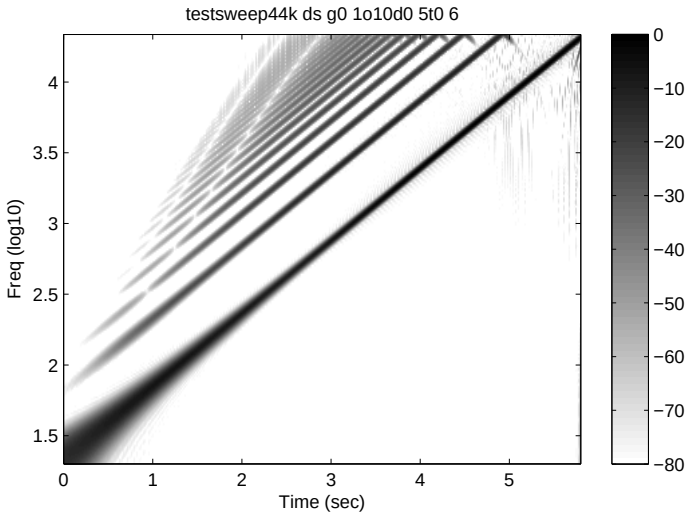


(b) Normalized spectrum of response to 220 Hz sine, distortion pedal (top), algorithm (bottom)

Figure 2.22: DS1 model verification: time response and output spectrum.



(a) Measured distortion pedal



(b) Distortion algorithm

Figure 2.23: Sine sweep log spectrogram of DS-1 pedal.

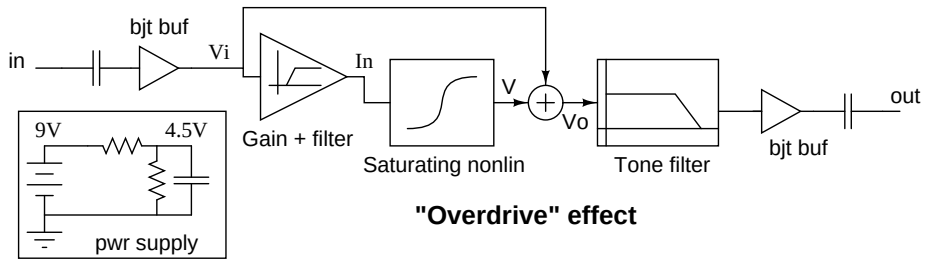


Figure 2.24: Block diagram of Overdrive pedal.

2.4 Circuit Analysis of Overdrive Pedal

The block diagram of an overdrive pedal, specifically the Ibanez Tube Screamer, is given in Fig. 2.24 (Keen, 2007). It is characterized by high-pass filters, followed by the summation of a high-pass filtered and clipped signal summed with the input signal. This is followed by low-pass tone filtering and a high pass in the output buffer. The following is an analysis of the circuit.

2.4.1 High-pass filters

The first stages of the overdrive pedal are high-pass filters with the following cutoff frequencies: $f_{c1} = 15.9\,\text{Hz}$, $f_{c2} = 15.6\,\text{Hz}$.

2.4.2 Non-inverting op amp with diode limiter

The non-inverting op amp (Fig. 2.25) of the overdrive pedal is similar to that of the distortion except the diode limiter is now across Z_f. The diode limiter essentially limits voltage excursion across the op amp, keeping it within ideal op amp conditions. The voltage at the minus input of the op amp is then the same as that on the plus terminal. This generates a current across Z_s,

$$I_n = \frac{V_{neg}}{Z_s} = V_i \frac{s}{R_1(s+\omega_z)}, \tag{2.10}$$

where $\omega_z = (R_1 C_z)^{-1}$, $R_1 = 4.7\text{k}\Omega$, $C_z = 0.047\mu\text{F}$.

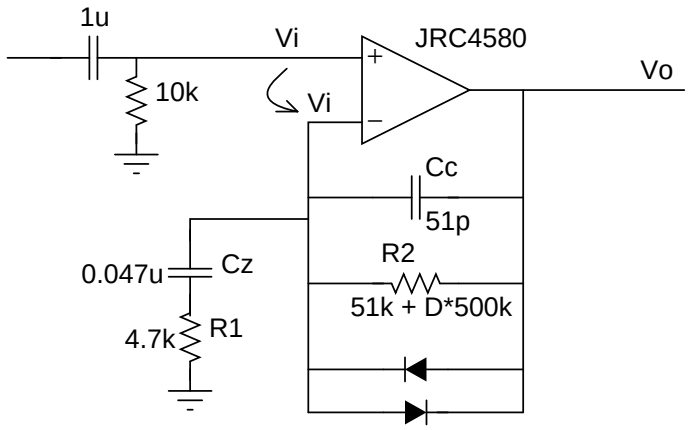


Figure 2.25: Clipping stage of overdrive pedal.

I_n flows through the components connected between the minus terminal and the output of the op amp. Circuit analysis produces the following equation:

$$I_n = \frac{V_o - V_i}{R_2} + C_c \frac{\mathrm{d}}{\mathrm{d}t}(V_o - V_i) + 2I_s \sinh(\frac{V_o - V_i}{V_t}) \tag{2.11}$$

Making a variable substitution $V = V_o - V_i$ yields,

$$\frac{\mathrm{d}V}{\mathrm{d}t} = \frac{I_n}{C_c} - \frac{V}{R_2 C_c} - \frac{2I_s}{C_c} \sinh(V/V_t), \tag{2.12}$$

where $C_C = 51\text{pF}$, $R_2 = 51\text{k} + D500\text{k}$, and $D \in (0,1)$, controlling the intensity of the overdrive. It can be seen that this ODE is the same as that for the Distortion pedal, (3.15), when I_n is replaced by V_i/R.

The arithmetic introduced by the variable substitution can be described in block diagram form as depicted in Fig. 2.24. The essence of the overdrive circuit is the summation of the input signal with the input filtered and clipped. The above variable substitution is solved for V_o:

$$V_o = V + V_i, \tag{2.13}$$

where V is obtained by solving (2.12).

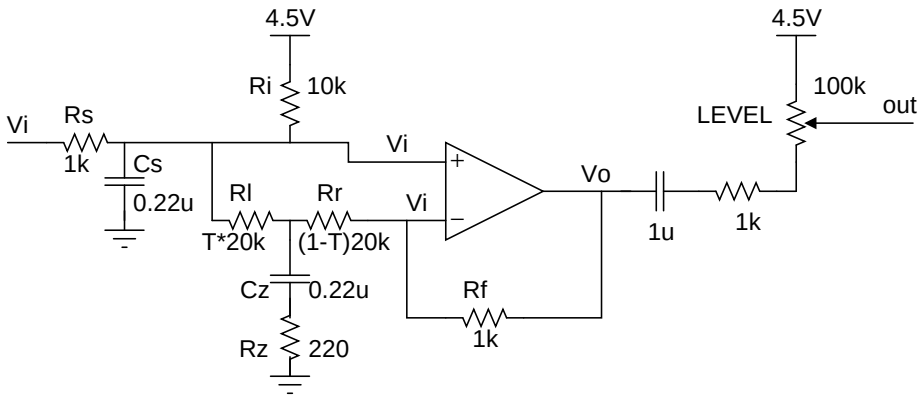


Figure 2.26: Overdrive tone circuit.

2.4.3 Tone stage

The tone stage (Fig. 2.26) can also be analyzed according to ideal op amp rules. The algebra is complicated, but the result is

$$\frac{V_o}{V_i} = \frac{(R_l R_f + Y)}{Y R_s C_s} \frac{s + W\omega_z}{(s+\omega_p)(s+\omega_z) + Xs}, \tag{2.14}$$

where

$$W = \frac{Y}{R_l R_f + Y},$$

$$X = \frac{R_r}{R_l + R_r} \frac{1}{(R_z + R_l \| R_r) C_z},$$

$$Y = (R_l + R_r)(R_z + R_l \| R_r),$$

$\omega_z = 1/\left(C_z\left(R_z + R_l \| R_r\right)\right)$, $\omega_p = 1/\left(C_s\left(R_s \| R_i\right)\right)$, $R_f = 1\text{k}$, $R_r = (1-T)20\text{k}$, $R_l = T20\text{k}$, $R_z = 220$, $C_z = 0.22\mu\text{F}$, $R_i = 10\text{k}$, $R_s = 1\text{k}$, $C_s = 0.22\mu\text{F}$, and $T \in (0,1)$ controls the cutoff frequency of the low pass.

This is a second-order transfer function with variable coefficients. Fig. 2.27 shows the essentially low-pass character of the magnitude response.

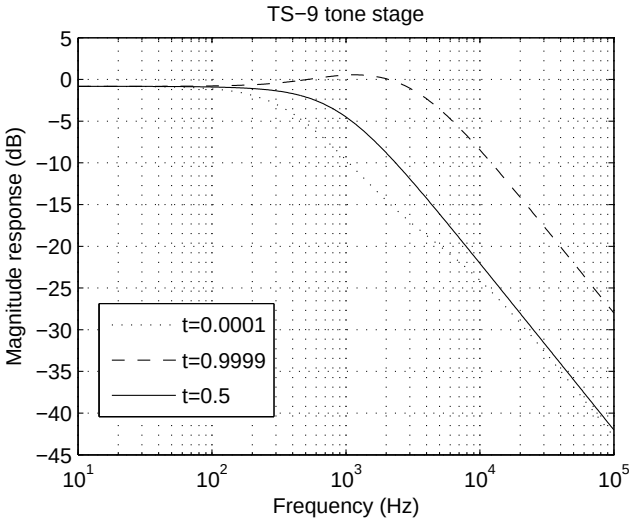


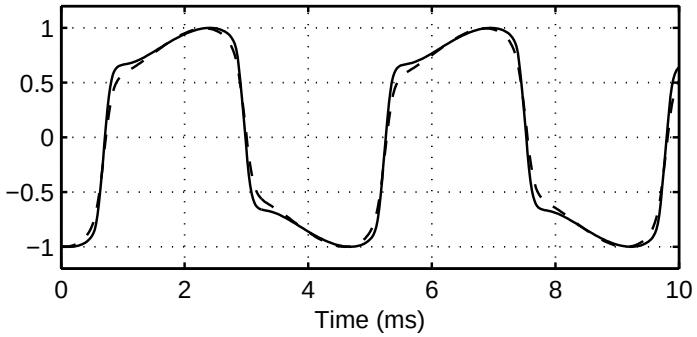
Figure 2.27: Overdrive tone circuit frequency response for $T = 0, 0.5, 1$.

2.4.4 Experimental Results

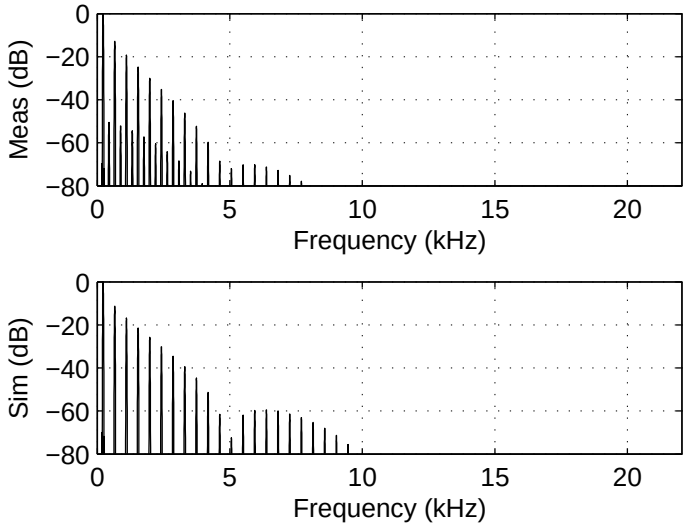
An actual Ibanez TS-9 Tube Screamer/Overdrive pedal was compared to the digital emulation for a 220 Hz sine signal with amplitude of 100 mV, and an exponential sine sweep. The settings on the actual pedal were adjusted until the spectrum resembled that of the digital version for the sine input. Adjustments were made approximately to match the difference in magnitude of the first two harmonics, and to match the position of notches in the frequency domain.

The time waveforms and magnitude spectra for the single-frequency excitation are shown in Fig. 2.28. The sinusoidal sweeps are represented by a log-frequency spectrogram (Brown, 1991) in Fig. 2.29 with 80-dB dynamic range.

As with the DS1, the static model for the tube screamer is missing low-level, even-order harmonics (-50 dB or less), most likely arising from the output BJT buffer, where the signal is largest. The diodes in the clipper are also not perfectly matched, which also contributes to an even-order nonlinearity. The model sounds brighter than the real circuit possibly because of the use of a static nonlinearity, and component variation in the tone stage.

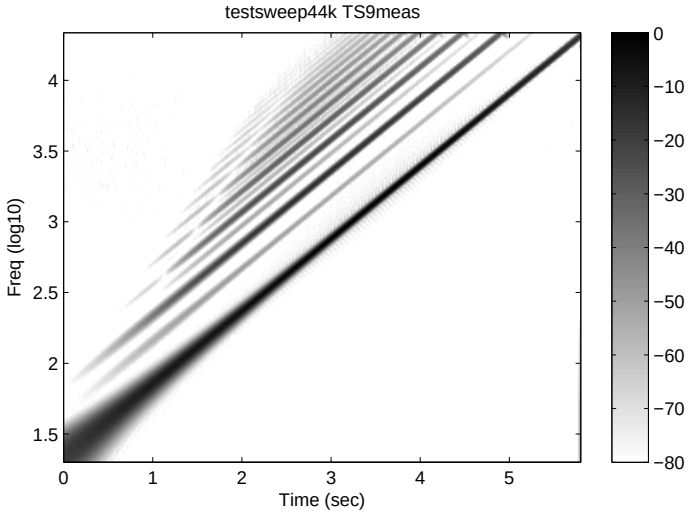


(a) Time response to 220 Hz sine, measured overdrive pedal (dashed) and algorithm (solid)

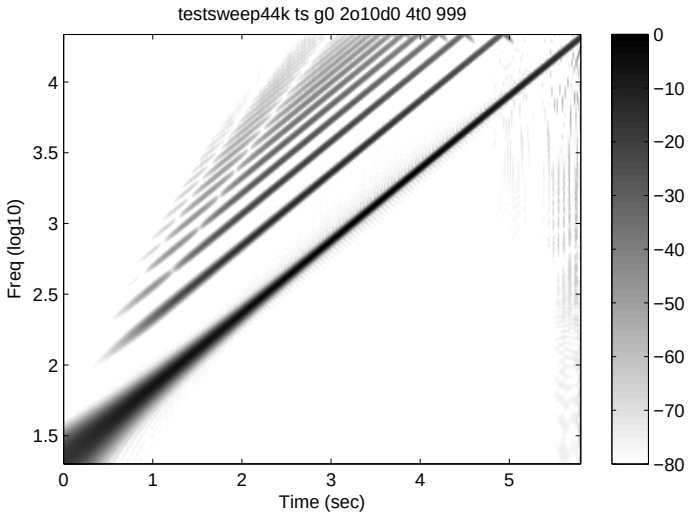


(b) Normalized spectrum of response to 220 Hz sine, overdrive pedal (top), algorithm (bottom)

Figure 2.28: Model verification of overdrive pedal: time response and output spectrum.



(a) Measured overdrive pedal



(b) Overdrive algorithm

Figure 2.29: Sine sweep log spectrogram of overdrive pedal.

2.4.5 Conclusions

The simplified, physically informed approach enables the design of distortion algorithms that emulate the behavior of analog prototypes. A first-pass design with no tuning is able to reproduce the salient characteristics of the effect.

While the result is not an exact emulation of the analog implementation, it provides a procedural basis for the design of a distortion algorithm, and a starting point from which further tuning can be done. The computational power needed is comparable to that available in commercially available guitar effects boxes because of the similar architecture comprising oversampling, low-order filters, and a tabulated nonlinearity.

In this work, BJT gain stage and op amp clipping behaviors are oversimplified. Nonlinearities are assumed to come from a single symmetrical diode clipper, which is not true under large-signal conditions. Improved models of remaining nonlinearities are the subject of subsequent chapters.

Chapter 3

Solution of Nonlinear Ordinary Differential Equations[1]

Electric guitarists prefer analog distortion effects over many digital implementations. The results in this chapter suggest reasons for this preference and proposes that detailed study of the electrical physics of guitar distortion circuits provides insight to design more accurate emulations. The work presented here introduces real-time emulation applied to guitar audio amplifiers in the form of a tutorial about relevant numerical methods and a case study. The results here suggest that simulating musical electronics using numerical methods in real time provides greater realism.

Analog guitar distortion effect devices known as solid-state distortion boxes commonly include a diode clipper circuit with an embedded low-pass filter. These distortion-effect devices can be modeled and accurately simulated as Ordinary Differential Equations (ODEs). A survey and a comparison of the basic numerical integration methods are presented as they apply to simulating circuits for audio processing, with the widely used diode clipper presented as an example.

[1]This chapter is adapted from Yeh et al., "Numerical Methods for Simulation of Guitar Distortion Circuits," Computer Music Journal, 32:2 (Summer, 2008), pp. 23–42.

3.1 Prior Work in Ordinary Differential Equation Solvers

Numerical solution of ODEs is a mature topic in applied mathematics, and many sophisticated algorithms exist for efficiently attaining accurate solutions and speed (Gear, 1971; Press et al., 1992; Shampine, 1994; Stoer and Bulirsch, 2002). The MATLAB scientific computing environment features a rich suite of ODE solvers (Shampine and Reichelt, 1997) that can be used for experimentation and gaining experience with the solution of ODEs. The circuit simulator SPICE (McCalla, 1987; Vladimirescu, 1994) is essentially a nonlinear ODE solver.

More advanced simulation techniques were subsequently developed based upon relaxation methods, which take advantage of loosely coupled circuit nodes in digital circuits to speed up the simulation (Newton and Sangiovanni-Vincentelli, 1984). Nonlinear analog circuits for audio processing, however, are typically highly coupled, possibly with global feedback, and still require the use of traditional, or "direct," ODE methods (White and Sangiovanni-Vincentelli, 1987).

ODE solvers use numerical integration methods to approximate a solution to the differential equation. SPICE typically offers the choice of Backward Euler, trapezoidal rule, and BDF (often referred to as Gear), which are implicit but stable. The popular explicit Runge-Kutta method of order four has great accuracy and is easy to use, but it is computationally expensive (Press et al., 1992). The extrapolation technique (Stoer and Bulirsch, 2002) is an efficient way to dramatically boost the accuracy of the solution, but requires greater implementation effort and complexity than the simple methods.

3.2 Numerical Methods

The basic ODE solvers use numerical integration to solve equations of the form

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(t, \mathbf{x}, \mathbf{y}), \tag{3.1}$$

where $\mathbf{x}$ is the system state; $\mathbf{f}(t, \mathbf{x}, \mathbf{u})$ is a nonlinear function that computes the time derivative of $\mathbf{x}(t)$, and depends on the current state $\mathbf{x}(t)$ and encompasses the input $\mathbf{u}(t)$ to the

system. Time t is the independent variable of integration for ordinary differential equations that describe circuits. For systems of equations in state-space notation, the state is described by a vector whose elements are nonlinear functions, and $\mathbf{f}(t, \mathbf{x}, \mathbf{u})$ is the derivative with respect to time of that vector. (Note that representing state by the variable $\mathbf{x}$ is chosen to be consistent with state-space notation; traditional numerical-analysis literature tends to use y to represent system state.)

In the case of a linear constant-coefficient differential equation, (3.1) becomes

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = \mathbf{A}\,\mathbf{x}(t) + \mathbf{B}\,\mathbf{u}(t), \tag{3.2}$$

where the eigenvalues of $\mathbf{A}$ are the poles of the system. Digital filters are efficient solvers of this special case of ODEs.

Like digital filters, explicit methods depend on states only from previous time steps. In contrast, implicit formulas depend on current state, forming a delay-free loop, and require iteration if the ODE is nonlinear. Newton's method, including its variants, is the most popular solver, in part because it is scalable to higher dimensions. For single-dimensional equations, bisection or bracketing provides predictable convergence under general conditions (Press et al., 1992).

In the numerical methods literature, ODE methods are notated with subscripts denoting the time index, superscripts for the current iterate, prime for the derivative, and h for step size (i.e., sampling period), for example,

$$y'_{n+1} = \frac{y_{n+1} - y_n}{h}.$$

In this chapter, methods are presented using square brackets to denote the time index, a dot to represent the time derivative, and T for step size, as is typically done for digital filters.

3.2.1 Integration Formulas

An ODE solver finds $x[n]$ given (3.1), where $t = nT$. For each discrete time n, the derivative operator in the ODE is evaluated by the application of a numerical integration formula.

3.2.1.1 Forward Euler

The most basic one is the Forward Euler (FE) method:

$$x[n] = x[n-1] + T\dot{x}[n-1], \tag{3.3}$$

where $x[n]$ is the system state at discrete time n, T is the sampling interval, and $\dot{x}$ is given in (3.1). Forward Euler is an explicit, first-order-accurate method, also known as forward difference.

3.2.1.2 Backward Euler

Another such formula is the Backward Euler (BE) method:

$$x[n] = x[n-1] + T\dot{x}[n]. \tag{3.4}$$

Backward Euler, also known as backward difference, is similar to Forward Euler, except the time derivative is evaluated at the current time point, resulting in an implicit, first-order-accurate method.

3.2.1.3 Implicit Trapezoidal Rule

Related to the above Euler methods is the Trapezoidal Rule (TR) method, given by

$$x[n] = x[n-1] + \frac{T}{2}\left(\dot{x}[n] + \dot{x}[n-1]\right), \tag{3.5}$$

which uses instead the average of the derivatives at times n and $n-1$, resulting in an implicit, second-order-accurate method.

It is also equivalent to the discretization of a continuous-time transfer function by the bilinear transform (4.1). The bilinear transform is the only practical order-preserving discretization method that does not introduce artificial damping, that is, turning unstable continuous-time poles into stable discrete-time ones (Smith III, 2008). The WDF implementation of the trapezoidal rule was also tried, but it was found to produce exactly the same results while requiring more operations; therefore, it is not presented here.

3.2.1.4 Backward Difference Formula Order 2

Another numerical integration method, the Backward Difference Formula (BDF2), is commonly used in circuit simulation, and it deserves mention here. It is a multi-step implicit method that only requires a single function evaluation of the time derivative of (3.1) per iteration:

$$x[n] = \frac{4}{3}x[n-1] - \frac{1}{3}x[n-2] + \frac{2T}{3}\dot{x}[n]. \tag{3.6}$$

3.2.1.5 Explicit Runge-Kutta Order 4

Finally, a popular higher-accuracy-order one-step method is the explicit fourth-order Runge-Kutta formula (RK4):

$$\begin{aligned}
k_1 &= T\,g(n-1, x[n-1]),\\
k_2 &= T\,g(n-1/2, x[n-1] + k_1/2),\\
k_3 &= T\,g(n-1/2, x[n-1] + k_2/2),\\
k_4 &= T\,g(n, x[n-1] + k_3),
\end{aligned}$$

$$x[n] = x[n-1] + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6}, \tag{3.7}$$

where $g(m,x) = f(t[m], x, u[m])$ and $f(t,x,u)$ is as defined in (3.1). Note that RK4 requires function evaluations every half sample, and therefore it requires input at twice the sampling rate of the output, also noted by Huovilainen (2004). On the contrary, the expanded bandwidth of the distorted output signal requires a sampling rate higher at the output than at the input to reduce aliasing. Implicit Runge-Kutta constructions also have been developed extensively (Gear, 1971; Butcher, 1987; Fränken and Ochs, 2001).

3.2.2 Newton's Method for Solving Nonlinear Equations

At time n, the implicit method must be solved for current state $\mathbf{X} = \mathbf{x}[n]$ and can be rewritten in the general vector form

$$\mathbf{0} = \mathbf{F}(\mathbf{X}), \tag{3.8}$$

which represents a nonlinear root-finding problem. In one dimension, one can easily search for values of $\mathbf{X}$ that set (3.8) to zero using the bisection method (Press et al., 1992).

For higher dimensions we need to use the general Newton's method (Press et al., 1992), which linearizes (3.8) with respect to $\mathbf{X}$ and solves the resulting system repeatedly until convergence. Newton's method requires the Jacobian, denoted $\mathbf{J_F}(\mathbf{X})$, of (3.8) with respect to $\mathbf{X}$ and evaluated at $\mathbf{X}$. For most implicit ODE methods, this can be easily computed from the Jacobian of $f(t,x,u)$ (3.1), denoted as $\mathbf{J}_f(\mathbf{X})$.

Newton's method is in general given by

$$\begin{aligned} \Delta\mathbf{X} &= -\mathbf{J}_\mathbf{F}^{-1}(\mathbf{X})\,\mathbf{F}(\mathbf{X}) \\ \mathbf{X} &:= \mathbf{X} + \Delta\mathbf{X}, \end{aligned} \tag{3.9}$$

and iterating until $\Delta\mathbf{X}$ is below some acceptable value. Usually LU decomposition and back substitution solves (3.9) instead of taking the matrix inverse owing to its salubrious numerical properties. It converges rapidly if the iteration is started with an initial guess for $\mathbf{X}$ that is close to the final solution. Typically, this guess is the previous state $\mathbf{x}[n-1]$, which works well when the system is oversampled and successive samples are close in value to each other.

Newton's method requires the Jacobian to be invertible at every point in the desired solution space or

$$\det(\mathbf{J}_F(\mathbf{x})) \neq \mathbf{0}.$$

Convergence is achieved when the $\mathscr{L}_\infty$ norm is below some acceptable parameter value RELTOL

$$\|\Delta\mathbf{X}\|_\infty < \text{RELTOL}. \tag{3.10}$$

It converges rapidly if the iteration is started with an initial guess for $\mathbf{X}$ that is close to the final solution.

To verify convergence to a valid solution, the residual at convergence is computed as $\mathbf{F}(\mathbf{x})$. The solution $\mathbf{x}$ is accepted if

$$\|\mathbf{F}(\Delta\mathbf{x})\|_\infty < \text{MAXRES}. \tag{3.11}$$

These two parameters RELTOL and MAXRES govern the accuracy of the Newton's method solver. If $\mathbf{X}$ is a voltage, RELTOL can be viewed as an noisy voltage error, which can be masked beneath other noise sources present in a realistic system.

3.2.2.1 Homotopy

In many problems, Newton's method must be initialized close to the solution to converge. In typical usage, proximity to the solution is unknown; therefore, we propose the use of homotopy to aid convergence. Homotopy was originally developed in circuit simulator research for finding DC solutions of circuits, and conditions for its use are discussed in (Melville et al., 1993; Ushida et al., 2002).

The particular form of homotopy used here is known as Newton homotopy (Ushida et al., 2002). It solves the system by starting from the trivial solution and parametrizing the problem such that, along the parametric path, the algorithm can move incrementally and always find a solution that is in the region of convergence for Newton's method.

For homotopy, start with an initial guess

$$x^* = 0.$$

Assume a parameter λ, which is incremented in the range $\lambda \in [0,1]$.

Solve the problem

$$\mathrm{H}(x,\lambda) = \mathrm{F}(x) + (\lambda - 1)\mathrm{F}(x^*) = 0$$

by Newton's method initializing the solution with $x = x^*$ for $\lambda = 0$. When x converges to a solution for the given λ, increment λ and solve $\mathrm{H}(x,\lambda) = 0$, initializing x with the solution x for the previous λ.

3.2.2.2 Semi-Implicit Methods

Given the observation that guitar-distortion systems are highly oversampled to suppress aliasing, the implicit methods can be modified to evaluate only one step of the Newton method iteration. This is known as the semi-implicit method (Press et al., 1992), which has

constant cost. Effectively, this converts implicit integration methods into explicit form by removing constraints on the final ΔX, which also relinquishes control over the error.

The expressions to evaluate ΔX for the two most well known implicit methods are given below; semi-implicit BDF2 can be similarly derived. In the following analysis, it is observed that $X = x[n-1]$, and $f(n,X)$ is understood to be shorthand for $f(t[n],X,u[n])$. Extension to systems with multiple states is straightforward.

The Semi-Implicit Backward Euler (BE s-i) is given by

$$\Delta X = \frac{x[n-1] - X + T\,f(n,X)}{1 - T\,J_f(X)} = \frac{T\,f(n,X)}{1 - T\,J_f(X)}. \tag{3.12}$$

The Semi-Implicit Trapezoidal Rule (TR s-i) is given by

$$\begin{aligned}\Delta X &= \frac{x[n-1] - X + 0.5T\,\left(f(n,X) + f(n-1,x[n-1])\right)}{1 - 0.5T\,J_f(X)} \\ &= \frac{0.5T\,\left(f(n,X) + f(n-1,x[n-1])\right)}{1 - 0.5T\,J_f(X)}.\end{aligned} \tag{3.13}$$

3.3 Accuracy of Numerical Integration Methods

3.3.1 Local Truncation Error

The traditional measure of accuracy is Local Truncation Error (LTE), which is the lowest-order difference between the full Taylor Series expansion of the solution and the result of the method. For example,

$$\dot{x}(t) = x(t) - x(t-T)$$

is first-order-accurate because the error is proportional to T as $T \to 0$. The trapezoidal rule, on the other hand, exhibits an error proportional to T^2 as $T \to 0$, so it is second-order-accurate. Manifestations of this error are aliasing and frequency warping. Oversampling reduces error by the accuracy order of the method. For example, it is known that the trapezoidal rule has the smallest truncation error of any method of order two (McCalla,

1987). Specifically, its truncation error decreases as one-twelfth the square of the sampling interval ("second-order convergence").

3.3.2 Stability

Using an approximation of the derivative to solve the ODE may introduce numerical stability problems. The standard stability analysis for numerical integration methods applied to nonlinear ODE involves linearizing the system, and applying numerical integration. For example, consider a linearized system as described by (3.2) with $B = 0$. Substituting this into the Forward Euler method (3.3) yields

$$x_n = (\mathbf{I} + T\mathbf{A})\, x_{n-1}.$$

This recurrence relation is stable if $|1 + T\lambda| < 1$ for each eigenvalue λ of matrix $\mathbf{A}$. The stability of an ODE method depends on the ratio between the sampling frequency and the largest eigenvalue of the system $\mathbf{A}$. For a nonlinear system, the eigenvalues may depend on the operating point.

Essentially, the various integration methods are different ways of mapping the continuous-system poles to discrete-time poles. If all resulting discrete system poles lie within the unit circle, the discretization yields a stable numerical solution.

3.3.2.1 Explicit Methods

The plot of the region of stability on the complex $T - s$ plane (the s-plane normalized times T) forms a bounded region where the method is stable. Explicit methods, such as Forward Euler and explicit RK4, result in polynomial stability conditions (Stoer and Bulirsch, 2002), which trace out an external boundary of the stability region in the s-plane (Fig. 3.1). Consequently, this places a limit on the largest magnitude negative eigenvalue the system may have to assure bounded behavior.

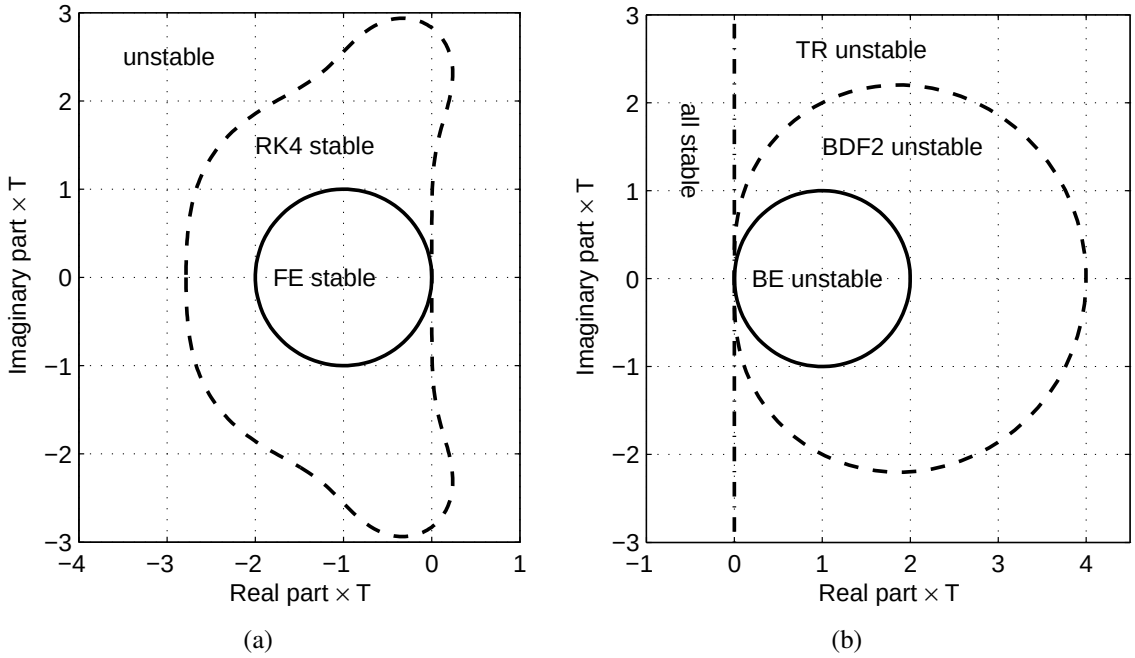


Figure 3.1: (a) Regions of stability for explicit methods Forward Euler (FE) and Runge-Kutta 4 (RK4) are inside boundary; (b) regions of stability for implicit methods Backward Euler (BE), BDF2, and trapezoidal rule (TR) are outside boundary.

3.3.2.2 Implicit Methods

For implicit methods, the stability region extends to infinity in the negative half-plane (Fig. 3.1), thereby placing no limit on the maximum magnitude of an eigenvalue of a system (if it is not complex) and allowing a low sampling rate. For trapezoidal, Backward Euler, and BDF2, the regions encompass the entire left half-plane, so all stable continuous-time systems will map to stable discrete-time systems ("A-stability"). Backward Euler and BDF2 will introduce artificial damping to higher frequency poles, possibly causing some unstable poles to be mapped to stable poles in the digital domain. The trapezoidal rule is the bilinear transform, mapping the complex frequency axis onto the unit circle and introducing no additional damping. An A-stable method will always converge to a stable result as long as the nonlinear solver converges.

3.3.2.3 Stiff Stability

For the ODEs found in analog circuits, it has been found in practice that implicit methods drastically reduce the sampling-rate requirement relative to explicit methods and are ultimately more efficient (McCalla, 1987). Circuit simulation problems often have eigenvalues that are highly separated in value, a property known as "stiffness" in the ODE literature, requiring a long time scale to compute the solution, and a small time step for stability when using explicit methods. Stiffly stable solvers place no requirement on the minimum sampling rate needed to ensure a bounded solution. Instead, considerations for accuracy such as aliasing govern the choice of step size. None of the explicit methods can be stiffly stable (Stoer and Bulirsch, 2002), because they require a minimum sampling rate to operate properly.

3.3.3 Considerations for Application to Audio Distortion Circuits

3.3.3.1 Error

The typical implementation of an ODE solver targets applications with different error requirements than real-time audio. Error for audio is best defined spectrally and perceptually

in the short-time frequency domain using masking information as in perceptual audio coding (Bosi and Goldberg, 2003). In this work we evaluate accurate behavior in the audio band between 0–20 kHz and neglect masking effects, or errors beyond 20 kHz. The error criterion for general solvers adaptively adjusts the variable step size to an excessively small value. When a stiffly stable method is used, an audio-band error criterion greatly improves efficiency by allowing a larger step size, as explained in the following.

3.3.3.2 Oversampling

As shown in Sec. 1.5, aliasing due to nonlinearities in digital implementations of distortion requires oversampling to attenuate the aliases. With the $8\times$ oversampling used in typical guitar distortion, all implicit ODE methods are very accurate between DC and 20 kHz, and frequency warping effects are far out of the audio band. Aliasing concerns dominate the choice of oversampling rate; it is therefore advisable to focus on simple methods with low computational complexity.

3.4 Case study: Diode clipper circuit

The diode-clipper circuit with an embedded low-pass filter forms the basis of both diode clipping "distortion" and "overdrive" or "tube screamer" effects pedals (Yeh et al., 2007a), and it is found in many other products that implement guitar distortion using solid-state circuitry. This popular circuit block is taken as a case example to evaluate the performance and feasibility of using numerical integration methods to solve nonlinear ODEs in real time for an audio system and how it compares to a static nonlinearity approximation. (The terms "diode clipper" and "diode limiter" refer to the same circuit and are used interchangeably.)

The Boss DS-1 circuit (Roland Corp., 1980) is a distortion pedal, and its schematic can be approximately divided into blocks as described in Sec. 2.3 and shown in Fig. 3.2. Here we focus on the saturating nonlinearity block, which is the diode clipper. We also developed a real-time audio plug-in based upon this model of the DS-1, which includes a custom solver for the ODE of the diode clipper. More detailed models should also include the nonlinearities of the buffer stages and the gain / filter block.

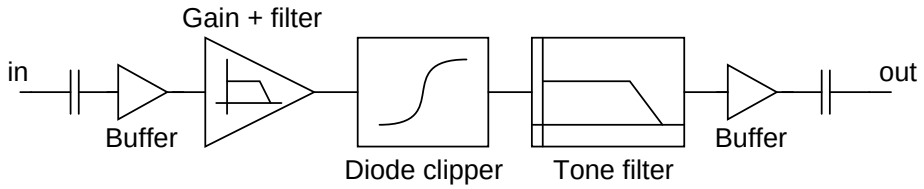


Figure 3.2: Partitioning scheme and block diagram for the Boss DS-1 circuit.

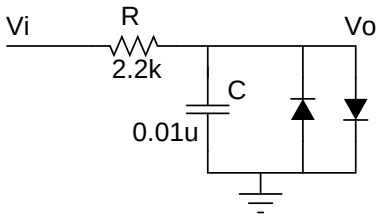


Figure 3.3: RC low-pass filter with diode limiter.

3.4.1 Diode Clipper Equation

The diode clipper in guitar circuits is typically a first-order resistor-capacitor (RC) low-pass filter with a diode limiter across the capacitor (Fig 3.3). The diode clipper limits the voltage excursion across the capacitor to about a diode drop (the diode "turn-on" voltage, approximately 0.7 V) in either direction about signal ground.

A common first-order approximation of the diode is a piecewise linear function, which is equivalently a switch model. A higher-order model is chosen here:

$$I_d = I_s \left(\exp \frac{V_d}{V_T} - 1 \right), \tag{3.14}$$

where I_d and V_d are the diode current and voltage, respectively. The reverse saturation current I_s, and thermal voltage V_T of the device are model parameters that can be extracted from measurement. Real diodes are more complicated (Muller et al., 2002), but this model is accurate enough for the range of values used in the clipper circuit. This model also possesses continuity in its first derivative, which simplifies convergence when solving the circuit equations using Newton's method.

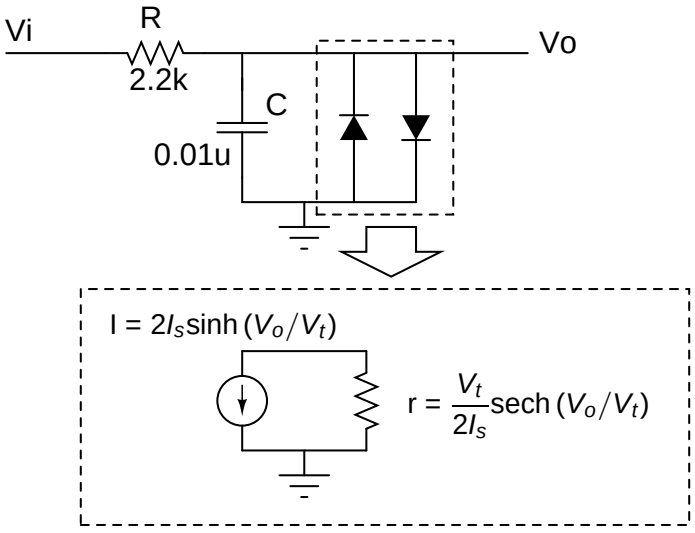


Figure 3.4: Linearized diode-clipper circuit.

The nonlinear ODE of the diode can be derived from Kirchhoff's laws (Yeh et al., 2007a):

$$\frac{\mathrm{d}V_o}{\mathrm{d}t} = \frac{V_i(t) - V_o}{RC} - 2\frac{I_s}{C}\sinh(V_o/V_T), \tag{3.15}$$

where V_i and V_o are the input and output signals, respectively. For this work, the parameters are $R = 2.2\,\mathrm{k}\Omega$, $C = 10\,\mathrm{nF}$, $I_s = 2.52\,\mathrm{nA}$, and $V_T = 45.3\,\mathrm{mV}$.

The linearized model (Fig. 3.4) of this circuit suggests that this nonlinearity has memory, because it yields a low-pass filter whose pole location depends on the state of the circuit. The results in Möller et al. (2002) also suggest that circuit nonlinearities have memory. Even with simulated data, which should have negligible measurement noise, the technique for extracting the nonlinear transfer curves of a tube amplifier produces a noisy result owing to hysteresis.

In general, although the basic nonlinearities of these devices are quasi-static for audio frequencies, placing the statically nonlinear device into a circuit with reactive devices will produce a nonlinear ordinary differential equation. This is especially true for voltage-mode circuits, where voltage is the signal variable, as in many amplifier circuits. The nonlinearity must be solved against the constraints imposed by the circuit, always yielding an implicit expression for the transfer curve mapping input voltage to output voltage, as demonstrated

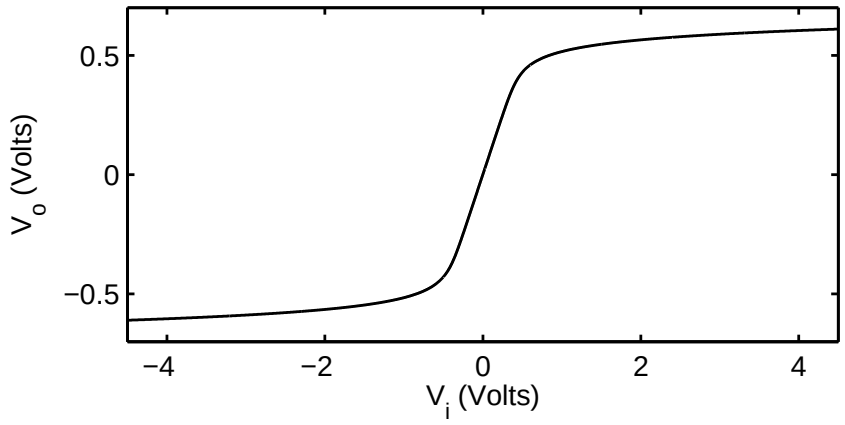


Figure 3.5: Tabulated static nonlinearity of the diode clipper solved from implicit nonlinear relationship.

previously. Reactive components apply complex constraints, yielding a nonlinearity with embedded memory, or equivalently, a filter with an embedded nonlinearity, as noted by Huovilainen (2004).

In the framework of Sec. 3.2, the ODE for the diode clipper is given by (3.15), where V_o is the state x, and V_i is the input u. For Newton's method, because $x \in \mathbb{R}^1$, the Jacobian $J_F(X)$ is simply the derivative of $F(X)$ with respect to X.

The left-half plane eigenvalue, or pole, of the diode clipper can be found from the small-signal linearization of the circuit in Fig. 3.4. When V_o is large, the linearized diode resistance will dominate R, making this eigenvalue approximately

$$\lambda_{\text{clip}} \approx -\frac{2I_s}{CV_t}\cosh\left(V_o/V_t\right). \tag{3.16}$$

Although this system only has one eigenvalue, it is intended to process audio input and needs to run for a time scale that is very large compared to the time constant of the eigenvalue. This system can thus be considered "stiff" in the general sense of the term.

3.4.2 Approximation of Diode Clipper ODE by Static Nonlinearity and Digital Filters

To evaluate the significance of the memory in the nonlinearity and to demonstrate what is lacking when static nonlinearities are used, an approximation to the ODE solver using a static nonlinearity with a prefilter was derived.

The nonlinearity used (Fig. 3.5) is the DC approximation of the actual nonlinearity, generated from the ODE by setting the time derivative of the output voltage in (3.15) to zero

$$\frac{\mathrm{d}V_o}{\mathrm{d}t} = \frac{V_i(t) - V_o}{RC} - 2\frac{I_s}{C}\sinh\left(V_o/V_T\right) = 0,$$

and solving via Newton's method. This is implemented using a lookup table as in Yeh et al. (2007a) and is a type of waveshaping distortion.

The pre-filter was heuristically derived by comparing simulation runs of a highly oversampled trapezoidal method with this static nonlinearity approximation and choosing a low-pass cutoff frequency that best approximates the phase shift over a wide range of frequencies. The result, a first-order low-pass filter with a cutoff frequency at approximately $\omega_c = 2.8\,/\,RC$ placed before the nonlinearity, also reduces aliasing compared to using no pre-filter.

3.4.3 Comparative Results

The basic methods presented in the previous section were applied to the ODE of the diode clipper and compared for various input signals. In all cases the methods were run with $8\times$ oversampling. The iterations terminate when the correction given by Newton's method for the previous iterate is less than 5 mV. This is acceptable, because, in an actual circuit, noise from the components is greater than this. In some cases, a $32\times$ oversampled trapezoidal rule result also serves as a highly accurate reference for comparison. In the Boss DS-1 (Yeh et al., 2007a), the signal is hard-clipped to ± 4.5 V by an operational amplifier gain-stage before being smoothed by the diode clipper. Therefore, the test inputs are normalized to 4.5 V.

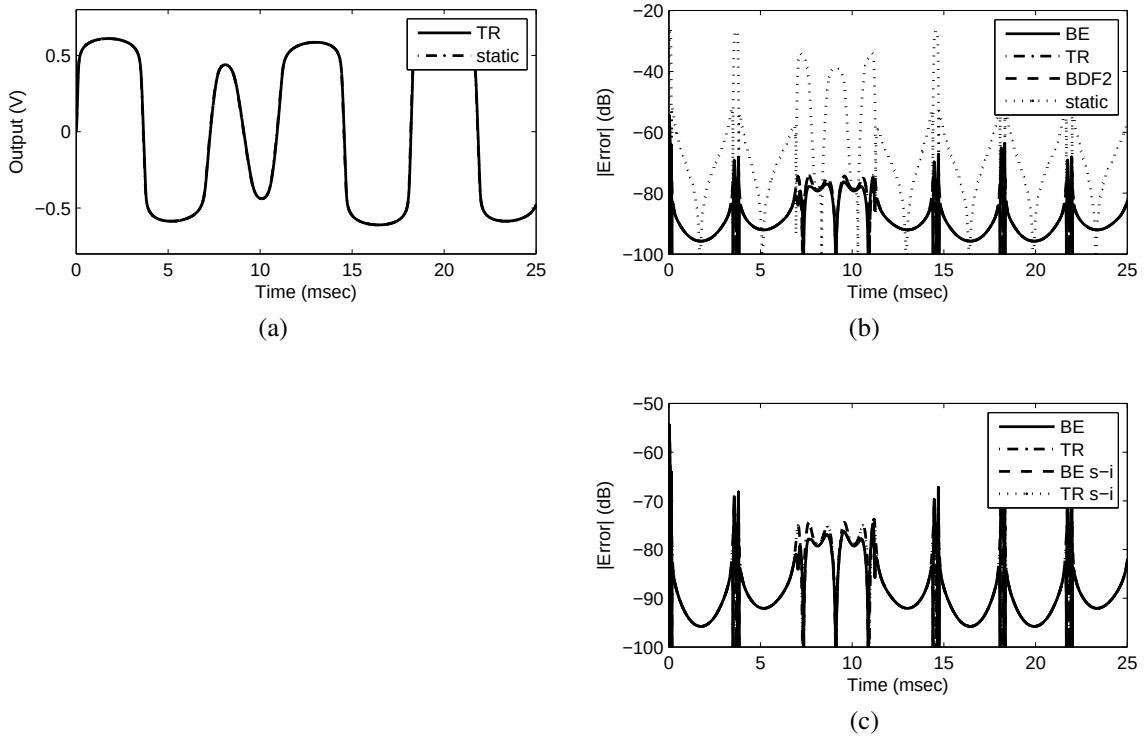


Figure 3.6: Time-domain results for 110 Hz + 165 Hz input. (a) Waveforms for trapezoidal (TR) and static approximation, 8× oversampling. They are indistinguishable in the figure. (b) Error for Backward Euler (BE), TR, BDF2, and static approximation. TR and BDF2 are almost identical, both being second order. (c) Error for BE, TR, and semi-implicit versions BE s-i, TR s-i. Only BE and TR can be distinguished here, because semi-implicit is practically identical to fully implicit for low-frequency input.

3.4.3.1 Two-Tone Sine

A dual-tone excitation (110 and 165 Hz, 4.5-V peak) was applied at the input of the diode-clipper using each of the stable integration methods. The implicit and semi-implicit methods generate almost identical time-domain responses. Figure 3.6a shows only the TR and static approximation. Figures 3.6b and 3.6c plot the error of the 8×-oversampled methods relative to the 32×-oversampled reference. All of the numerical methods exhibit similar profiles with low error. The second-order-accurate methods TR and BDF2 have almost identical error. Semi-implicit and implicit versions of the same method have almost identical error for these low frequencies. The static approximation shows noticeably larger error

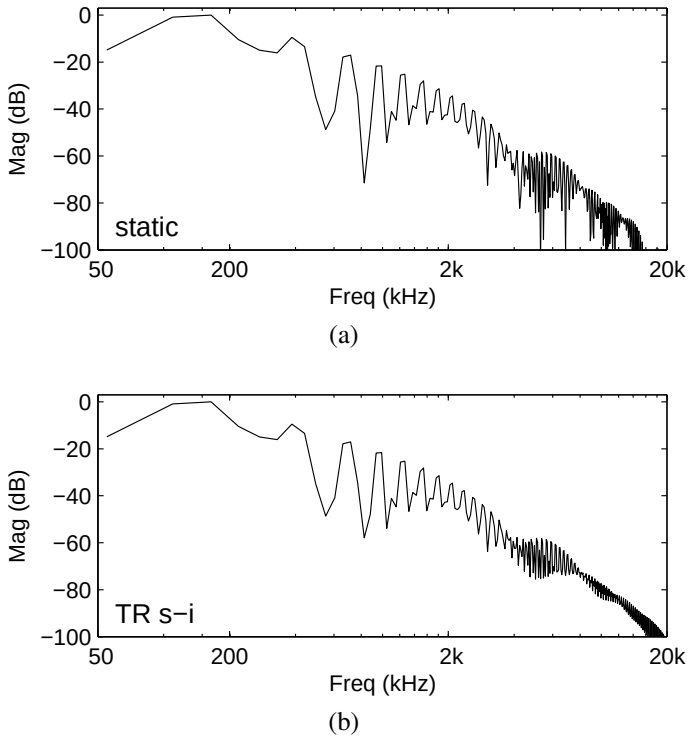


Figure 3.7: Peaks in spectra of responses to 110 Hz + 165 Hz input, connected by solid lines, for (a) semi-implicit trapezoidal (TR s-i) and (b) static nonlinearity approximation. The other methods are practically identical to TR s-i.

than the numerical solvers, but it is typically less than -20 dB, or 10%, a good engineering approximation.

A spectral comparison better represents the audible differences. The numerical solvers all produce similar output spectra and are represented in Fig. 3.7 by the semi-implicit trapezoidal rule. Only the peaks are plotted and connected by straight lines for better visual discrimination. This is compared to the static approximation, which is a close approximation that is extremely accurate in the first few harmonics and reproduces the overall contour of the spectrum. The ODE smoothes the response at higher frequencies, whereas the static approximation exhibits sharp nulls in the peaks' heights.

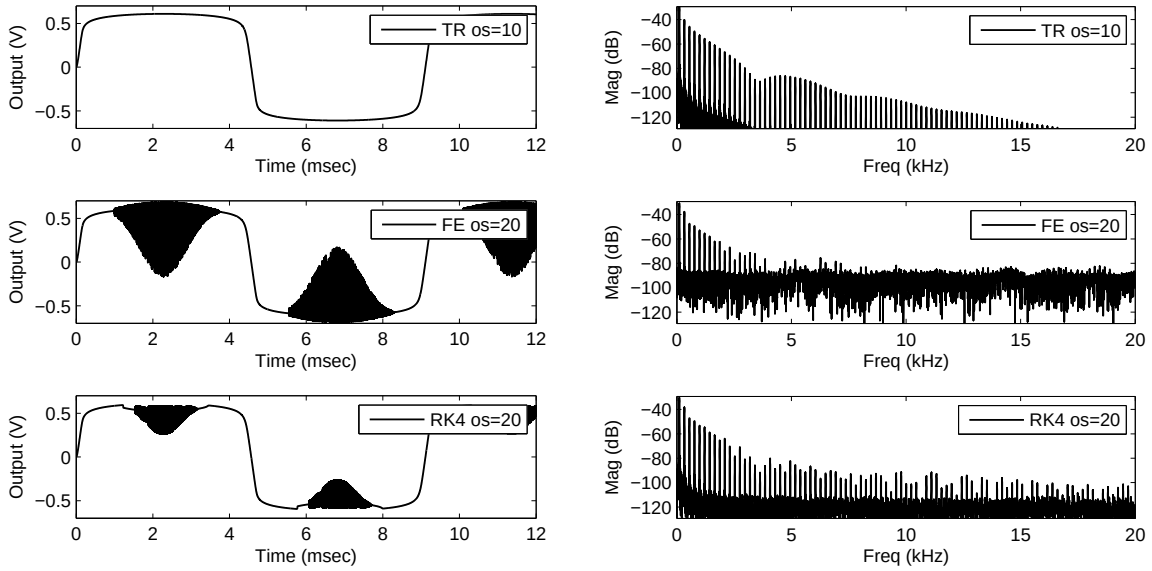


Figure 3.8: Waveforms (left) and frequency spectra (right) demonstrating time domain instability and broadband spectral noise of explicit methods Forward Euler (FE) and fourth order explicit Runge-Kutta (RK4), both oversampled at 20×, compared to Trapezoidal rule (TR) oversampled at 10×.

3.4.3.2 Explicit Methods: Stability Problems

A test sinusoid of 4.5 V and 110 Hz was applied to the clipper ODE discretized by the Forward Euler and fourth-order explicit RK4 methods with 20× oversampling to demonstrate the high oversampling needed for a stable simulation of a system with a high frequency pole.

Figure 3.8 shows the results, compared with using the trapezoidal rule, with 10× oversampling. The time waveform follows the curve of the solution but becomes unstable at high signal values, when the diode has high conductance. Considering the linearized system about that operating point, the high small-signal conductance of the diode contributes to a high-frequency pole that lies outside the method's stability region. The spectra resemble that of the TR solution in the low frequencies, but the instability manifests as a broadband noise floor that sounds particularly grating and cannot be removed by simple filtering. It was found experimentally that, to achieve stable results, very high oversampling factors (38× for Forward Euler and 30× for RK4) were needed.

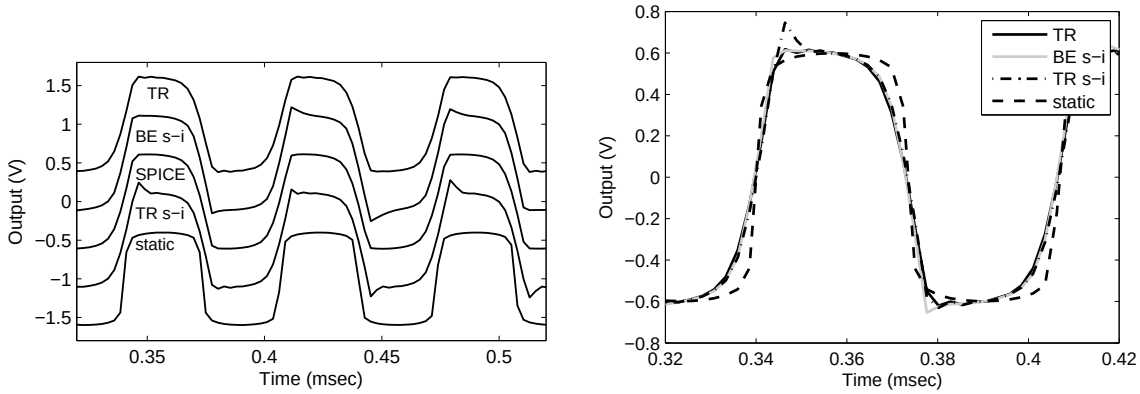


Figure 3.9: Time-domain waveforms for 15,001 Hz input. (a) Five curves are plotted with offsets to facilitate visual discrimination. Top to bottom plotted with offsets in parentheses: trapezoidal (+1V), Semi-implicit Backward Euler (+0.5V), SPICE (+0V), semi-implicit trapezoidal (-0.5V) and the static approximation (-1V), all with 8× oversampling. (b) The methods are also overlaid for comparison.

3.4.3.3 Single High-Frequency Sine and Verification with SPICE

A high-level, high-frequency sine-wave excitation (4.5 V, 15,001 Hz) reveals inadequacies in the semi-implicit methods, which exhibit overshoot in the time-domain plots (Fig. 3.9) and spurious tones in the frequency domain.

The same input was provided to the SPICE simulator LTspice (Linear Technology, 2007), which has WAV-file import capability, using trapezoidal rule integration. This comparison verifies the methods and approximations used in this work. An exact time-domain match to SPICE cannot be expected because of differences in convergence criteria and numerical handling. SPICE uses an adaptive step size to control error, and it applies linear interpolation when interfacing with the WAV sound-file format.

The spectra of trapezoidal method, semi-implicit trapezoidal method, and static approximation at 8× oversampling are plotted in Fig. 3.10 against the result generated by trapezoidal-rule integration with 32× oversampling, which represents the accurate solution. The static nonlinearity correctly reproduces the magnitude of the fundamental.

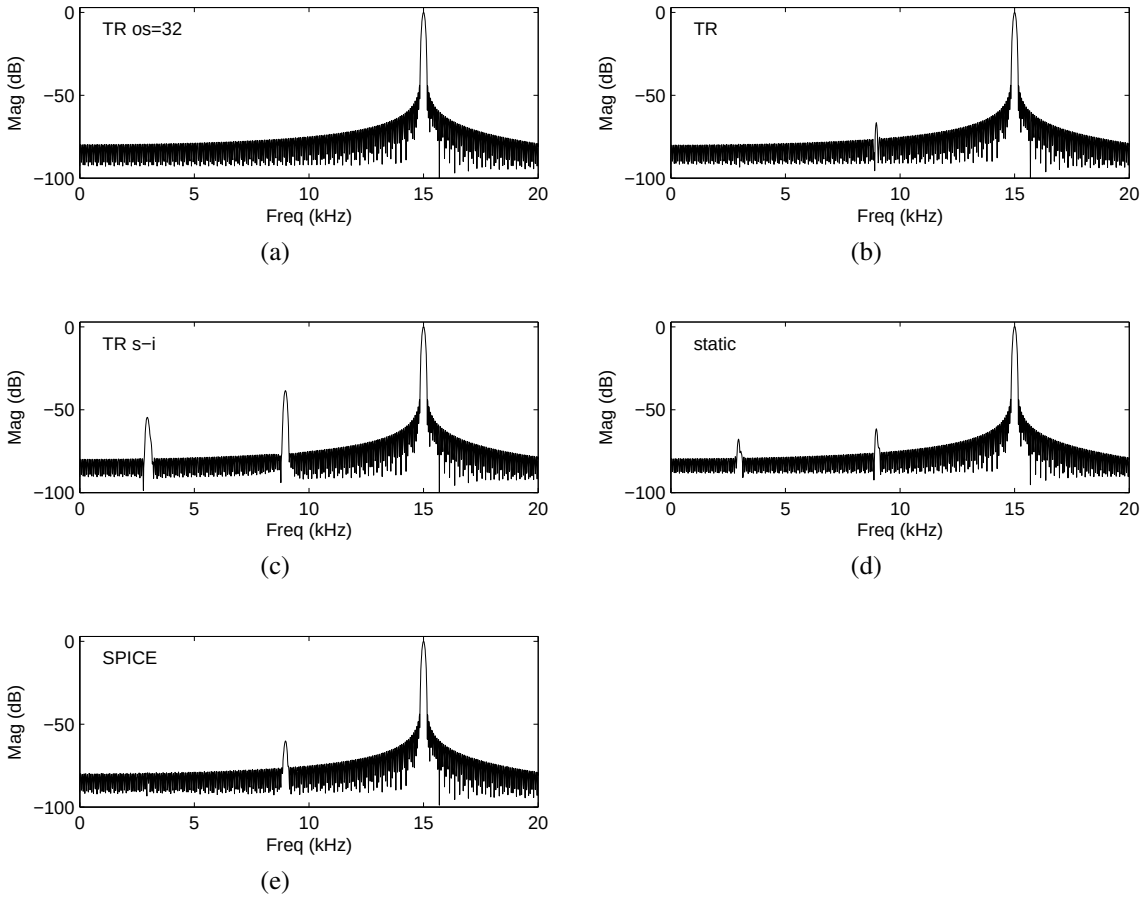


Figure 3.10: Magnitude response to 15,001 Hz, 4.5 V input: (a) 32× oversampled reference using TR, (b) TR, (c) TR s-i, and (d) static, each at 8×oversampling, and (e) LTspice, which linearly interpolates output to the 8× oversampling grid.

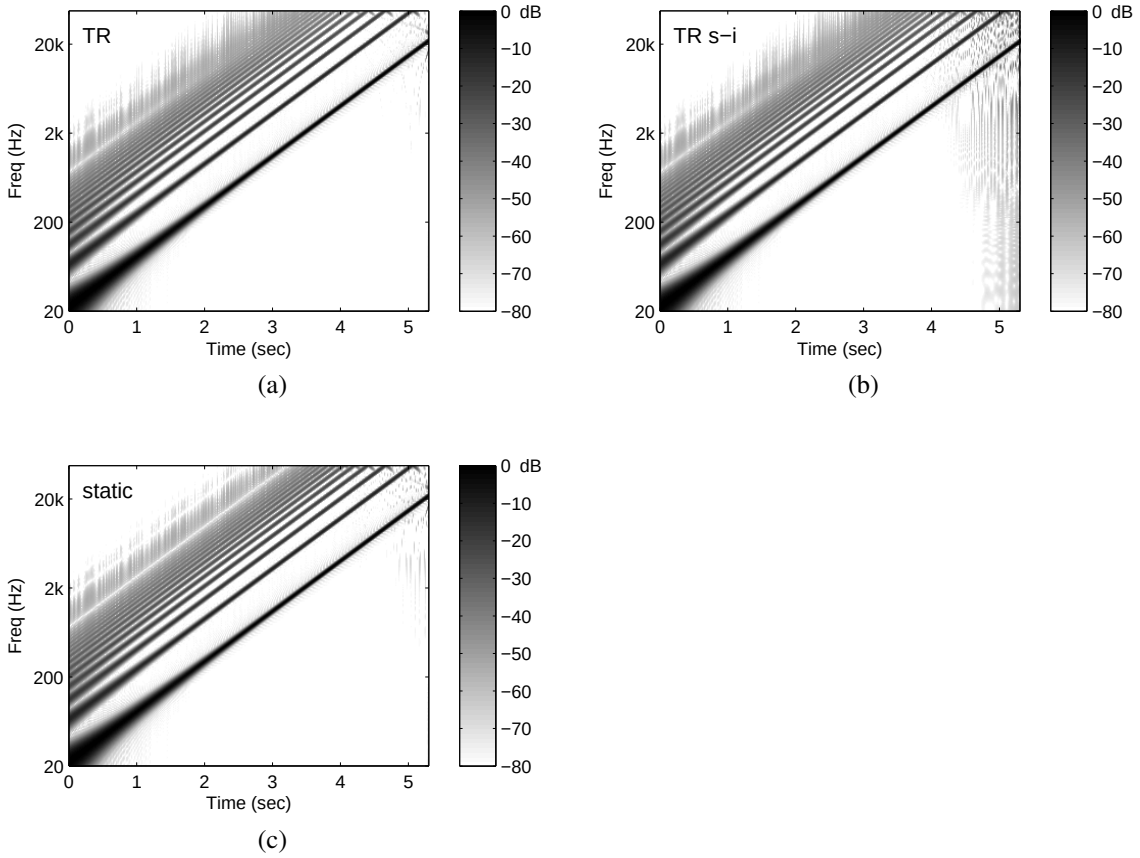


Figure 3.11: Log spectrogram of diode-clipper response to sine sweep using (a) TR, (b) TR s-i, and (c) static approximation methods, for $8\times$ oversampling, $f_s = 48\,\text{kHz}$.

3.4.3.4 Sine Sweep

A high-amplitude, sinusoidal, exponential frequency sweep from 20 Hz to 20 kHz was processed by the methods. The output was downsampled to 96 kHz and displayed as a log spectrogram. All of the ODE methods produce almost identical output if stable, so only the spectrograms for trapezoidal rule, its semi-implicit version, and the static nonlinearity are shown in Fig. 3.11. The ODE methods exhibit a blurring of the higher frequencies absent in the static case and also reduce aliasing. The static nonlinearity produces very little aliasing at $8\times$ oversampling. The semi-implicit methods overshoot for strong high-frequency components, which manifests as a strong aliasing-like characteristic.

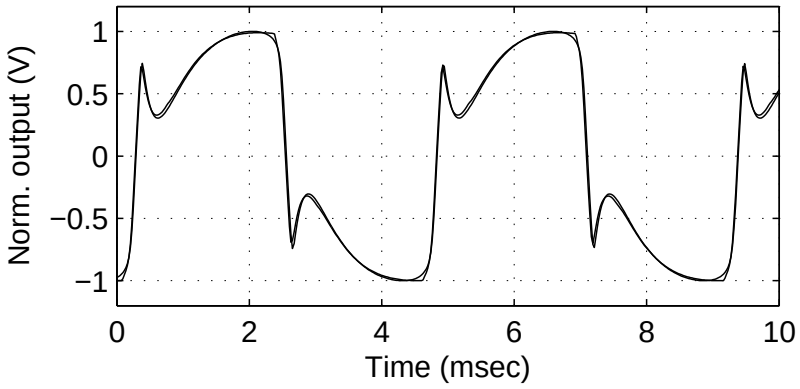


Figure 3.12: Time response to 220-Hz, 100-mV input signal of measured Boss DS-1 (solid) and simulation with TR (dashed), normalized to 1 V. They are almost exactly overlaid in this view. The DS1 exhibits a sharper corner at the right edge of the waveform.

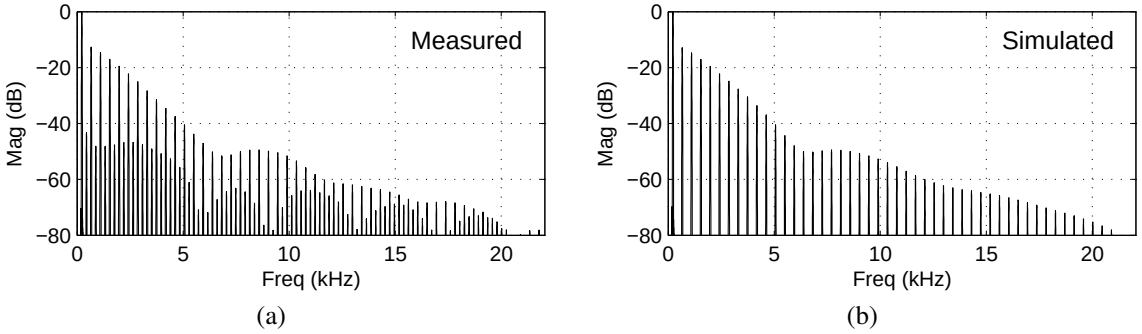


Figure 3.13: Spectra of Fig. 3.12: (a) Boss DS-1; (b) simulation with TR.

3.4.3.5 Comparison with Measurement

Finally, the entire block diagram of Fig. 3.2 was simulated, using an $8\times$-oversampled trapezoidal method on the diode clipper, and compared to output from an actual DS-1 (Figs. 3.12 and 3.13). The input to each system was a 220-Hz sine wave with 100-mV amplitude. Arbitrary knob settings were chosen in simulation, and the knobs on the actual device were turned until the spectrum exhibited a similar envelope. The differences in output are most likely caused by the asymmetric nonlinearity of the bipolar transistor-gain stage, which provides even-order harmonics and which was neglected in this model. The additional harmonics cause a subtle audible difference (an increased level of additional tones), which

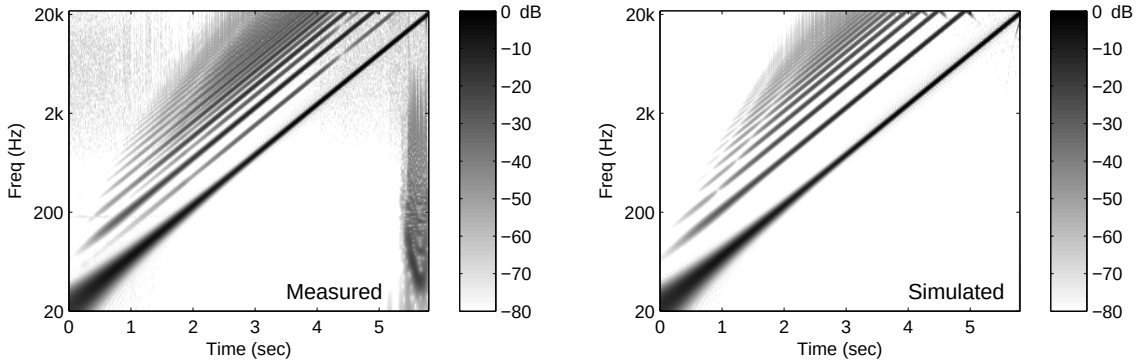


Figure 3.14: Log spectrograms of sine sweep from 20 Hz – 20 kHz: (a) DS-1 and (b) simulated model.

should be included in more accurate models. Note that the use of the ODE solver on the diode clipper significantly improves upon the static approximation used in Sec. 2.3.6.

Likewise, log spectrograms of the two devices in response to an exponential-frequency sweep from 20 Hz to 20 kHz are compared in Fig. 3.14. The use of the diode clipper with ODE solver also appears to reduce aliasing relative to the static approximation used for the spectrograms of Sec. 2.3.6. The measured device exhibits device noise seen above 1 kHz and features an impulse-like, subharmonic output at 5.3 s when the circuit no longer can track the high-level, high-frequency input. This behavior is not sonically pleasing and perhaps need not be modeled.

3.4.4 Computational Cost

Because the number of iterations in an ODE solver that employs Newton's method is related to the input in a complicated way, an empirical measurement of cost is made. For the $8\times$-oversampled rate, the number of iterations per sample required for a tolerance of 5 mV is plotted in Figs. 3.15–3.17, along with the moving average over a frame size of 256 samples (32 samples at 48 kHz). The inputs used are an exponential sine sweep from 20 Hz – 20 kHz and 4.5 V amplitude, and two signals representative of typical input, recorded from an electric guitar with Humbucking pickups. The first signal was an open E "power chord"

Method	X	f-calls	Avg f-calls/sample
FE	38	1	38.0
RK4	30	4	120.0
BE	8	$2n$	28.8
BE s-i	8	2	16.0
TR	8	$1+2n$	36.8
TR s-i	8	3	24.0
BDF2	8	$2n$	28.8
BDF2 s-i	8	2	16.0
static	8	Lookup	–

Table 3.1: Cost comparison of methods in terms of function calls (f-calls) to either the time derivative (3.15) or its Jacobian. Oversampling X required is also shown. For implicit methods, $n = 1.8$, the average number of iterations per sample over a frame. Base sampling rate is 48 kHz.

with a peak input of 1 V, amplified by 60 dB, and hard-clipped to 4.5 V before processing by the ODE, and a single-note riff with a bend, normalized to a peak of 4.5 V.

Although extremely high-level, high-frequency input causes problems with convergence as indicated by the sine sweep, the number of iterations per sample never exceeds eight. When guitar signals are extremely amplified (60 dB for example), more iterations are needed at the clipping threshold, because the signal changes very abruptly. However, when used in a frame-based audio processing system, assuming a conservative frame size of 32 samples at 48 kHz (256 at 8×), the average number of iterations per sample never exceeds 1.8.

The cost per sample in terms of function calls to compute either the time-derivative (3.1) or its Jacobian in the iterative methods is shown in Table 3.1. The cost of computing the derivative is assumed to be similar to the cost of computing the Jacobian; therefore, the cumulative number of function calls represents the cost of the method. The cost is normalized per audio sample at the base sampling rate of 48 kHz. For iterative methods, the number of iterations n averaged over the 32-sample frame is assumed to be 1.8, as suggested herein.

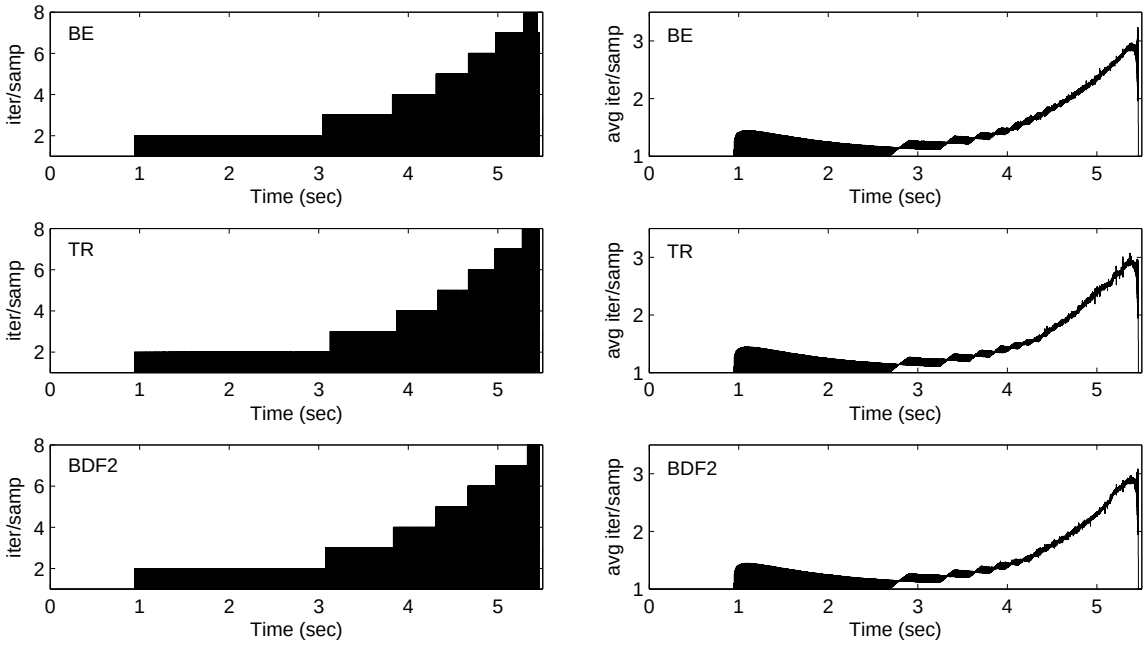


Figure 3.15: Exponential sine sweep from 20–20kHz. Left: number of iterations per sample; right: moving average of iterations using a frame size of 256 samples. Top to bottom: BE, TR, BDF2.

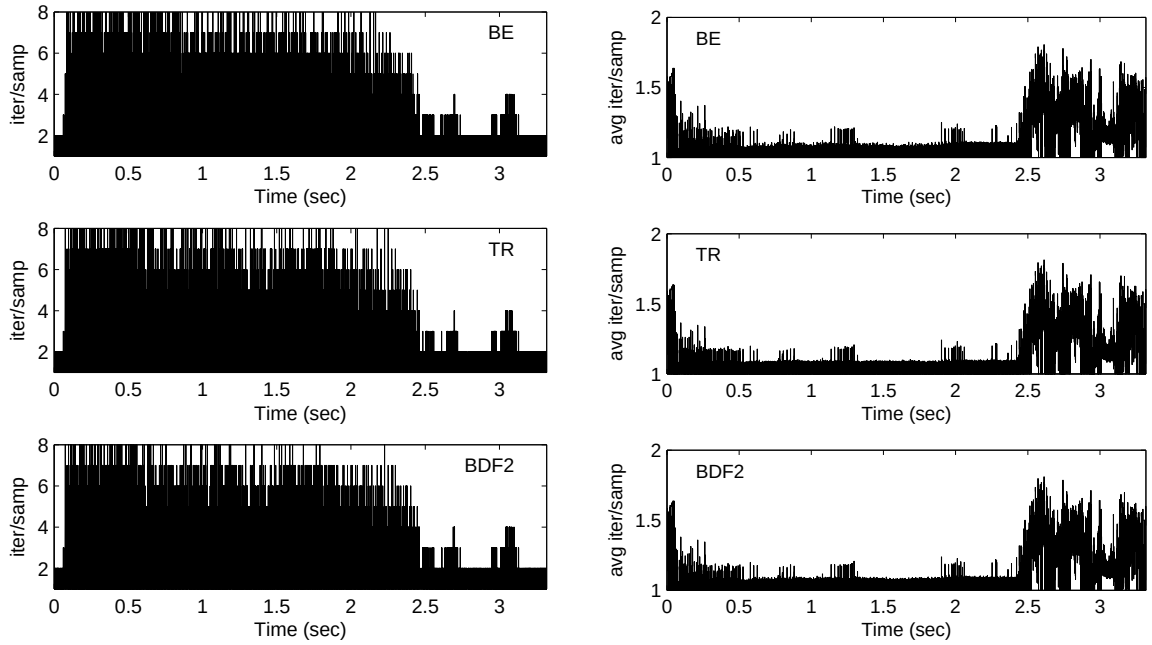


Figure 3.16: Power chord. Left: number of iterations per sample; right: moving average of iterations using a frame size of 256 samples. Top to bottom: BE, TR, BDF2.

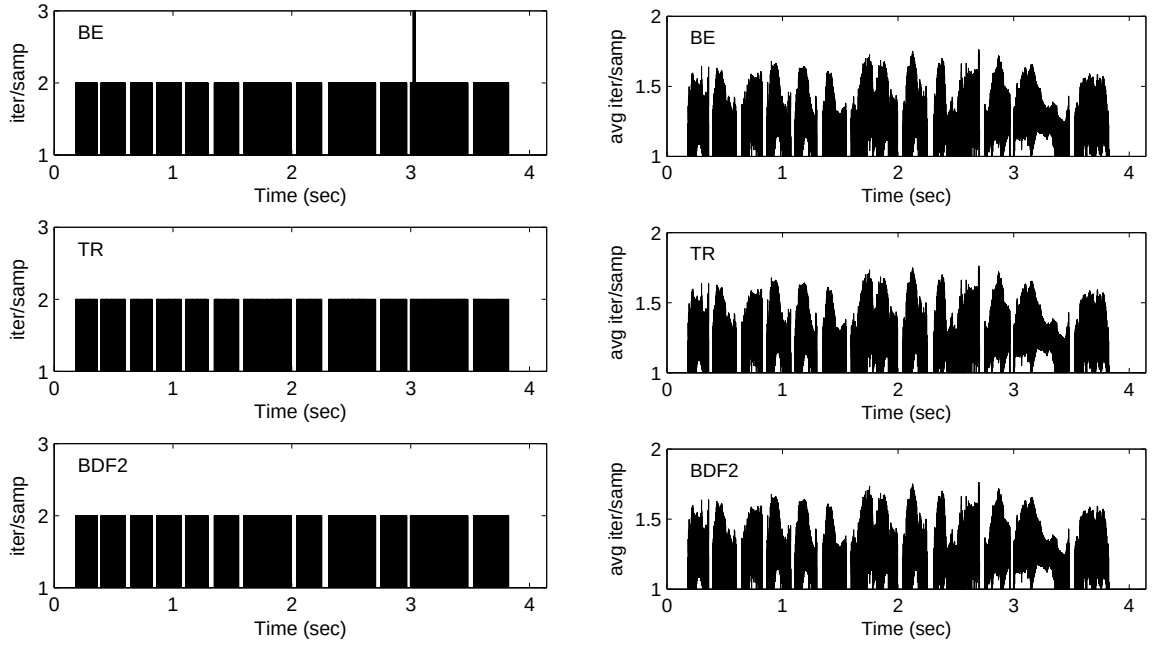


Figure 3.17: Single-note riff with a bend. Left: number of iterations per sample; right: moving average of iterations using a frame size of 256. Top to bottom: BE, TR, BDF2.

3.4.5 Discussion

3.4.5.1 Choice of Method

The evaluation of cost confirms prior findings in the circuit-simulation literature that implicit methods are preferred over explicit ones for simulation of general circuits. The explicit methods, although simple, do not produce reliably accurate results for the diode clipper ODE, as measured in the frequency domain, unless they are impractically highly oversampled. This need for excessive oversampling ($38\times$ for FE and $30\times$ for explicit RK4) was found to be necessary to avoid numerical instability associated with a high-frequency pole in the physical model. Huovilainen (2004) successfully applied an explicit method, because the Moog filter is typically weakly nonlinear. When systems become strongly nonlinear, they may operate in device regions that result in high-frequency poles, which cause stability problems with explicit methods as shown here. Convergence by Newton's method for circuits with strongly nonlinear regions is also difficult in general.

For audio-frequency input, the differences between the methods are negligible in the audio band because the process is well oversampled to reduce aliasing, especially when dealing with clipping-type distortions. The oversampling causes the errors of the various accuracy-order methods to be very low in the audio band and makes the effect of frequency warping insignificant. Thus, complicated higher order-accuracy methods such as extrapolation techniques or implicit Runge-Kutta are unnecessary. It would seem then that a stable method of low order would be sufficient while guaranteeing bounded output if a convergent nonlinear solver is used. The time-domain outputs of the semi-implicit methods show significant ringing for high-frequency inputs, but this is an extreme case because high amplitudes at these frequencies are rarely encountered in practical guitar signals as demonstrated by the sound examples.

3.4.5.2 Real-Time Considerations

A prototype implementation, with no particular efforts to write efficient code, demonstrates that the iterated implicit methods run in real time up to $8\times$ oversampling on a contemporary CPU (Intel Core 2 Duo, 1.6 GHz).

Experience in circuit-simulator development has shown that most of the computational effort takes place within the iterative loop to evaluate the detailed nonlinear models of circuit devices (McCalla, 1987). This was also demonstrated by Karjalainen and Pakarinen (2006), who showed that tabulation of the device models significantly speeds up the iterative solution in modeling a tube preamplifier with WDF. The computational complexity in the WDF approach is similar to that here, which suggests that full implicit solution of circuits should be feasible.

Scaling to circuits with more nodes should be possible as long as device models remain simple to compute or if they are tabulated. This is true especially because the goal of guitar distortion is to model amplifiers with vacuum tubes whose characteristics should be measured and tabulated for greatest realism. Even simplified models, though, may prove sufficient to capture the dynamics or character of the circuit.

3.5 Explicit discretization of the Moog ladder filter: why it is stable

In general, for guitar distortion circuits with strongly clipping nonlinearities, Newton's method with convergence aids is required to solve the nonlinear ODE. However, there are several examples of circuits with weak nonlinearities for which the system eigenvalue is bounded and therefore can be simulated with an explicit method (Huovilainen, 2004, 2005).

The nonlinear ODE derived and discretized by Huovilainen (2004) for the Moog ladder filter is

$$\frac{dV_c}{dt} = \frac{I_c}{C}\left(\tanh\left(\frac{V_{in}}{2V_t}\right) - \tanh\left(\frac{V_c}{2V_t}\right)\right)$$

Huovilainen uses the explicit Forward Euler method to discretize this system. Recall that explicit methods have a bounded region of stability for the system eigenvalue normalized by sample rate. Forward Euler has the largest stability region of the explicit methods that are not Runge-Kutta.

To consider the stability of this system, we need to find its linearized system eigenvalue. Let $V_{in} = 0$ to simplify the analysis.

$$\frac{dV_c}{dt} = F(V_c) = -\frac{I_c}{C}\tanh\left(\frac{V_c}{2V_t}\right)$$

Linearize this equation about an operating point V_C.

$$\frac{dv_c}{dt} = \left.\frac{\partial F}{\partial V_c}\right|_{V_c=V_C} v_c = -\frac{I_c}{C}\operatorname{sech}^2\left(\frac{V_C}{2V_t}\right) v_c$$

The eigenvalue of this system is thus $\lambda = -\frac{I_c}{C}\operatorname{sech}^2\left(\frac{V_C}{2V_t}\right)$. Contrast this with the eigenvalue of the diode clipper, which is $\lambda_{\text{clip}} \propto -\cosh(V_O/V_t)$. The functions $\operatorname{sech}^2(x)$ and $\cosh(x)$ are shown in Fig. 3.18 for comparison. Note that the eigenvalue for the Moog has a maximum and falls to zero as V_C increases, while the eigenvalue for the diode clipper will rise very rapidly with V_O. The Moog ODE has the convenient property that the system eigenvalue is bounded, and its maximum value is that in the middle of the nominal operating range. It can thus be guaranteed to be stable over its entire operating range with an explicit discretization.

To generalize, explicit methods can be used if the $\mathrm{f}(x,t)$ of an ODE in normal form

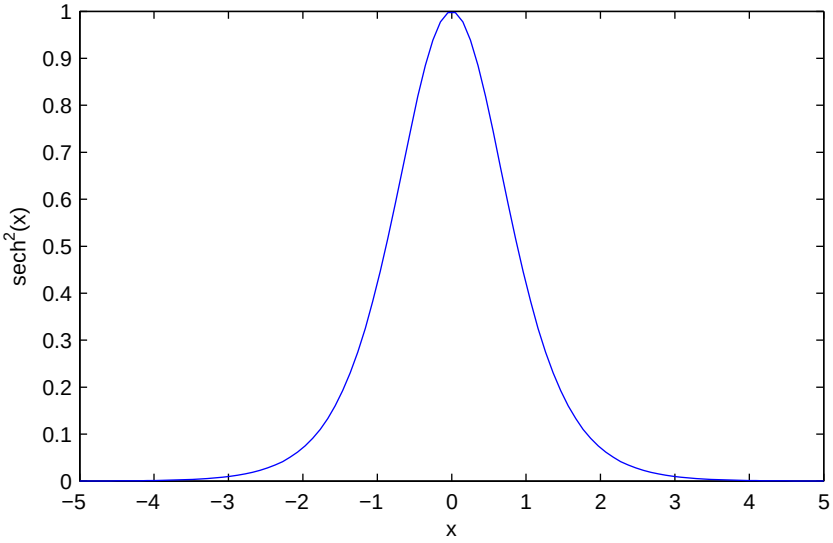
$$\dot{x} = \mathrm{f}(x,t)$$

is given by a compressive, continuous function of state variable x so that its first derivative (partial with respect to x) will be bounded. When applied to circuit elements with memory, this means that the nonlinear conductance seen across the terminals of the capacitor must be a compressive function of the capacitor voltage:

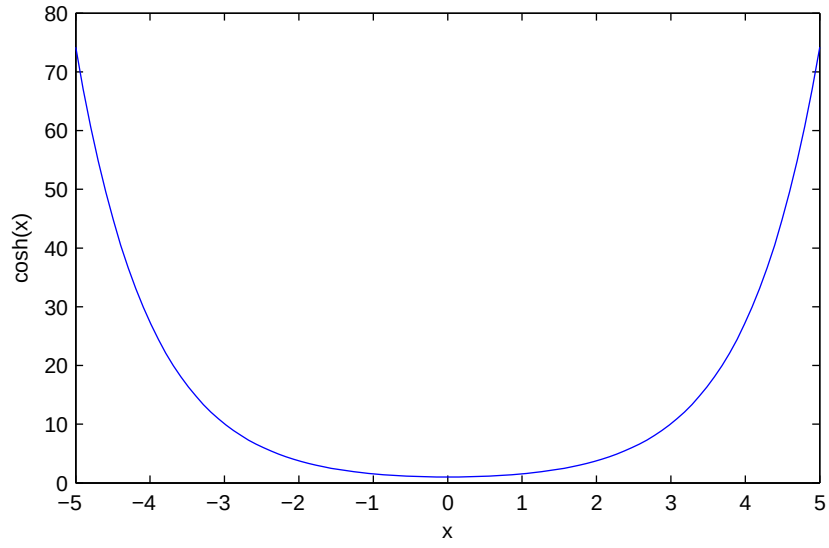
$$C\dot{v} = \mathrm{g}(v,t)$$

For inductors the nonlinear resistance seen across the terminals of the inductor must be a compressive function of the inductor current:

$$L\dot{i} = \mathrm{h}(i,t)$$



(a) $\text{sech}^2(x)$



(b) $\cosh(x)$

Figure 3.18: Comparison of $\text{sech}^2(x)$ and $\cosh(x)$

The DC nonlinearity of the Moog

$$0 = \frac{dV_c}{dt} = \frac{I_s}{C} - \frac{I_c}{C}\tanh\left(\frac{V_c}{2V_t}\right)$$
$$I_s = I_c \tanh\left(\frac{V_c}{2V_t}\right)$$

is also characterized by an explicit relationship between the input current signal I_s , and the state variable V_c. On the other hand, the diode clipper is inherently a nonlinear relationship between the input voltage V_i and the state/output variable V_o.

Huovilainen (2005) also discretized several other circuits by the Forward Euler method. These include an operational transconductance amplifier based all pass with a feedforward capacitor, and a JFET allpass implementation. In both these cases, the current going into a capacitor is a saturating function of the capacitor voltage and consequently the resulting filter algorithm can be guaranteed to be stable throughout its operating range.

This knowledge informs the choice of explicit or implicit methods when discretizing simple nonlinear ODEs found by inspection and approximation of the full circuit equations. In general, because circuits may have multiple and coupled states with implicitly defined nonlinear transfer characteristics, it becomes difficult to guarantee stability using explicit methods. Furthermore, the implicitly defined nonlinear functions need to be solved anyway by iteration. Consequently, general circuit simulation resorts to guaranteed stable implicit methods.

3.6 Conclusions

In general, while many electronic devices (e.g., diodes, vacuum tubes) might be characterized by a static nonlinearity, placing one into a circuit creates a nonlinear ODE. Often this can be approximated by a static nonlinearity, which can be derived from the nonlinear ODE as demonstrated; however, solving the ODE provides more accurate results.

Although explicit ODE methods can be used in cases where the poles of the system are known never to cross the stability boundary, the pathological case of the diode clipper and

the findings of the circuit simulation literature indicate that, in general, implicit methods are required.

Because the nonlinearities in typical guitar distortion are strong, bandwidth is expanded by a large factor, necessitating large oversampling rates. This constraint on the sampling rate causes the different methods, if stable, to be negligibly different in the audio frequency band.

It was found that, for realistic input signals, the number of iterations required for implicit methods to converge at $8\times$ oversampling is not overwhelmingly costly (typically no more than eight iterations), because the signal changes smoothly across samples. This suggests the use of semi-implicit methods. Both implicit and semi-implicit ODE solvers were successfully implemented in a real-time audio plug-in emulating a simplified block diagram of the Boss DS-1.

Chapter 4

Numerical Simulation of General Lumped Systems

This chapter discusses systematic approaches to apply the methods of Chapter 3 to more complex systems with multiple state variables and multidimensional nonlinearities.

Numerical simulation of lumped nonlinear systems has been studied extensively in the literature (Sarti and De Poli, 1999; De Sanctis et al., 2003; Borin et al., 2000; Fontana et al., 2004; Huovilainen, 2005; Avanzini and Rocchesso, 2002; Yeh et al., 2008; Sarti and De Sanctis, 2009). The computational musical acoustics / digital audio effects community has developed two prevailing methods for simulating ordinary differential equations with nonlinearities based on wave digital principles or directly solving a nonlinear state-space system. Both methods have been applied to the same types of problems in nonlinear musical acoustics and are also applicable to certain classes of nonlinear circuits used for musical effects.

This work extends attempts to simulate musical circuits based upon solving ordinary differential equations (Huovilainen, 2004; Karjalainen and Pakarinen, 2006; Pakarinen, 2008; Yeh et al., 2008). The motivation for this work is to investigate block-based modeling techniques (Rabenstein et al., 2007) applied to a more complete simulation of electronic circuits used in musical effects processing. Because circuits naturally divide into stages which may interact with adjacent stages, they are apt for description as a two-way signal flow diagram as in (Rabenstein et al., 2007; Fettweis, 1986). This work presents examples of how

circuits may be represented as blocks in such a modeling scheme. Various schemes exist that account for the mutual interaction between blocks (Rabenstein et al., 2007; Fontana et al., 2004; Avanzini et al., 2005; Fontana and Avanzini, 2008; Fettweis, 1986; Sarti and De Poli, 1999; Sarti and De Sanctis, 2009; Smith III, 2008).

This chapter reviews the wave digital formulation for representing circuits as digital filters, and formulations based upon state-space with memoryless nonlinearity, also known as the "K-method" of computational musical acoustics. In addition, it reviews the predominant formulation for circuit simulation and applies this technique to derive parameters for the K-method in a systematic fashion. The subsequent chapter demonstrates the applicability of these techniques to circuits in guitar electronics: the bright switch, the diode clipper, a transistor amplifier, and a triode amplifier.

4.1 Wave Digital Filter

An alternative formulation to the ODE problem is to express the signals and states in terms of wave variables and to apply component-wise, or local, discretization (Fettweis, 1986) at a uniform sample rate. This formulation is known as the Wave Digital Principle, and the resulting ODE solvers are Wave Digital Filters (WDF). WDFs typically apply trapezoidal-rule integration in the form of the Bilinear Transform

$$s = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}} \tag{4.1}$$

because it preserves stability across continuous- and discrete-time domains, but other passive ODE methods with greater orders of accuracy have been developed for WDFs as well (Fränken and Ochs, 2001, 2002). A procedure to automatically generate the minimal WDF from an arbitrary circuit topology has been developed (Meerkötter and Fränken, 1996; Fränken et al., 2005). The nonlinear WDF has been investigated extensively (Meerkötter and Scholz, 1989; Felderhoff, 1996; Sarti and De Poli, 1999; Sarti and De Sanctis, 2009), and it has been used to simulate the ODE of a simplified vacuum tube preamplifier circuit for guitar distortion (Karjalainen and Pakarinen, 2006).

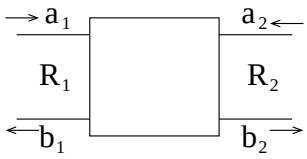


Figure 4.1: Two port scattering element defines network blocks using incident waves a, reflected waves b, and a port resistance R.

When using the bilinear transform (4.1) for discretization, the WDF formulation is equivalent to trapezoidal-rule integration and results in an identical state trajectory as the iterations are being solved. This occurs because, in the WDF, the nonlinearity is still expressed and solved in terms of Kirchhoff variables (currents and voltages), requiring a conversion from the wave variables.

4.1.1 Wave Digital Formulation

The wave digital formulation (Fettweis, 1986) views the linear N-port circuit network as a scattering junction, replacing voltage V and current I variables that define a single port by incident A and reflected B waves, and a port impedance R, as depicted in Fig. 4.1 for a two-port. Doing so allows instantaneous reflections to be eliminated by matching port impedances when ports are connected together, resulting in a computable wave digital filter (WDF) structure.

The variable transformation to the wave domain is

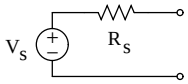
$$\begin{aligned} A &= V + RI \\ B &= V - RI \end{aligned} \tag{4.2}$$

and the inverse formulation exists $\forall R \geq 0$.

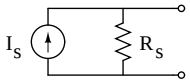
4.1.1.1 Wave digital elements

In the wave digital filter, circuit elements such as resistors, capacitors and inductors become port impedances and delay, if applicable, as shown in Tab. 4.1. They are computed

Element	Port impedance	Reflected wave
Resistor R	$R_p = R$	$b[n] = 0$
Capacitor C	$R_p = T/2C$	$b[n] = a[n-1]$
Inductor L	$R_p = 2L/T$	$b[n] = -a[n-1]$
Short circuit	$R_p = X$	$b[n] = -a[n]$
Open circuit	$R_p = X$	$b[n] = a[n]$
Voltage source V_s	$R_p = X$	$b[n] = -a[n] + 2V_s$
Current source I_s	$R_p = X$	$b[n] = a[n] + 2R_pI_s$
Terminated V_s	$R_p = R_s$	$b[n] = V_s$
Terminated I_s	$R_p = R_s$	$b[n] = R_pI_s$



Terminated V_s



Terminated I_s

Table 4.1: Wave digital elements: Port impedances and reflected waves. The last two elements are sources lumped with a resistance as shown. Elements with unmatched port resistances can be set to any positive port resistance $R_p = X$.

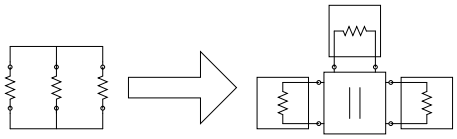
by substituting (4.2) into the Kirchhoff definitions of the elements and computing the reflected waves due to the change in impedance from the port to the element. Usually the port resistance is chosen such that the instantaneous reflection is matched, resulting in a reflection-free port (RFP).

4.1.1.2 Wave digital adaptors

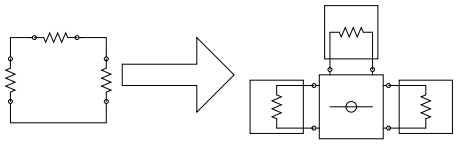
The topology of the circuit is represented by adaptors, which compute the scattering among ports from their port impedances. Because connections can often be described in terms of series and parallel electrical arrangements, series and parallel adaptors have been studied in detail and are used for connecting elements in wave digital filters. Parallel and series adaptors in the three-port case are shown schematically in Fig. 4.2. The scattering relations for elements and adaptors are always derived by substituting (4.2) into the Kirchhoff equations defining the element or adaptor.

The scattering relations for N-port parallel adaptors are given by

$$\gamma_v = \frac{2G_v}{G_1 + G_2 + \cdots + G_n}$$



(a) Parallel



(b) Series

Figure 4.2: Three-port adaptors and corresponding circuit schematic

$$a_p = \gamma_1 a_1 + \gamma_2 a_2 + \cdots + \gamma_n a_n$$
$$b_\nu = a_p - a_\nu,$$

where γ_ν are the scattering parameters for each port ν, a_p is a parallel junction wave variable, and b_ν is the reflected wave for each port ν.

The scattering relations for N-port series adaptors are given by

$$\gamma_\nu = \frac{2R_\nu}{R_1 + R_2 + \cdots + R_n}$$

$$a_s = a_1 + a_2 + \cdots + a_n$$
$$b_\nu = a_\nu - \gamma_\nu a_s$$

where γ_ν are the scattering parameters for each port ν, a_p is a series junction wave variable, and b_ν is the reflected wave for each port ν.

Note that both parallel and series adaptors have linear complexity with the number of ports, as opposed to the N^2 complexity of a generic scattering junction. The reader is referred to the comprehensive tutorial (Fettweis, 1986) for the scattering relations of other elements and adaptors.

Each adaptor can have at most one reflection free port (RFP) whose impedance is matched to the equivalent impedance of all the other ports combined. Because of this,

adaptors naturally form a directed tree structure with the RFP oriented towards the root of the tree. The entire tree in the classical WDF formulation naturally admits only one reflection free element, which must be placed at the root of the tree. All other elements must not have an instantaneous reflection. While the tree arrangement is suggested in (Fettweis, 1986), a more through development is presented in (De Sanctis et al., 2003; Sarti and De Sanctis, 2009).

4.1.1.3 Nonlinear wave digital elements

Nonlinear elements are also derived by substitution of (4.2) into the Kirchhoff definition of the element, and solving for the reflected wave b (Meerkötter and Scholz, 1989). This often produces an instantaneous reflection, which in the classical WDF, must be placed at the root of the tree. Thus, WDFs can handle only a single nonlinearity, although modified approaches might be able to handle more (Petrausch and Rabenstein, 2004).

Wave digital filters can also implement nonlinear reactances such as capacitors and inductors by defining an auxiliary port to the element whose wave variables correspond to the state variables of the reactance. For example, for a nonlinear capacitor $Q = C(V)V$, the wave variables at the auxiliary port would be defined in terms of charge Q and voltage V. The transformed variable definitions can then be used in the defining equations of the nonlinear reactance. The reader is referred to (Sarti and De Poli, 1999; Felderhoff, 1996) for detailed derivations and usage.

4.1.1.4 Other wave digital filter considerations

Because WDFs are often arranged in a tree structure, with data dependency flowing from the leaves to the root, and then from the root back down to the leaves, it is convenient to label the wave signals in terms of u for signals going up the tree, and d for signals traveling down the tree (De Sanctis et al., 2003; Rabenstein et al., 2007; Sarti and De Sanctis, 2009). Otherwise, naming signals in terms of incident and reflected waves between WDF components can quickly become unwieldy.

Once the WDF elements and adaptors are defined, the WDF tree is sufficient to represent its computational structure. The examples in Chapter 5 will present their algorithms as WDF trees.

4.2 Modified Nodal Analysis / Transient Simulation

Perhaps the most well-known circuit simulator is SPICE (Simulation Program with Integrated Circuit Emphasis) (Nagel and Pederson, 1973; Nagel, 1975). Claims of using circuit simulation techniques to emulate guitar distortion often conjure up the concept of using the numerical and matrix techniques found in SPICE to solve the equations that describe circuit behavior, although demonstrable examples of this are rare. Over the past 30 years, the circuit CAD (Computer Aided Design) community has converged upon an approach for simulating the time-domain behavior of nonlinear circuits.

Nonlinear time-domain simulation is referred to as transient simulation in SPICE. SPICE and other popular simulators use component-wise discretization of reactive components (namely capacitors and inductors), and modified nodal analysis (MNA) to build up a linear system that can be solved for the unknowns (Vladimirescu, 1994; McCalla, 1987).

4.2.1 Nodal Analysis

Modified nodal analysis solves for the unknowns of any circuit based upon Kirchhoff's Current Law (KCL) at each node and a few auxiliary equations.

Nodal analysis formulates Ohm's Law in matrix form. The current flowing out of node 1 into node 2 through a resistor of conductance G is given by Ohm's Law as

$$I_R = GV_R = G(V_1 - V_2).$$

An expression of the conservation of charge, KCL states that the sum of currents into a node is zero

$$\sum I_n = 0.$$

KCL at node 1 then takes the form

$$G_{12}(V_1 - V_2) + G_{13}(V_1 - V_3) + ... = 0.$$

Any independent current sources into node 1 can be added to the RHS:

$$G_{12}(V_1 - V_2) + G_{13}(V_1 - V_3) + ... = \sum I_s \tag{4.3}$$

Independent current sources are known, and the unknowns are the node voltages

$$V_1,\ V_2,\ V_3,\ \ldots$$

Each node introduces an unknown and requires the application of KCL at that node, giving for the whole circuit N independent equations to determine uniquely the N unknowns. Conceptually, nodal analysis is simply the matrix representation of a system of KCL equations like 4.3 as a unique set of N linear equations needed to determine N unknowns.

$$\mathbf{M}\mathbf{V} = \mathbf{I} \tag{4.4}$$

The unknowns in vector $\mathbf{V}$ typically represent the voltages at each node referenced to ground. Each row in $\mathbf{M}$ contains the conductance coefficients for KCL at the node represented by the same row in $\mathbf{V}$. Any independent current source into that node adds to the corresponding row in the RHS vector $\mathbf{I}$.

Using KCL at each of the nodes results in an overdetermined system. Often one node is designated ground, its KCL deleted from the system, and all of the other node voltages are then referred to the potential at the ground node.

4.2.2 Modified Nodal Analysis (MNA)

Sometimes the circuit variables are related in a way other than through Ohm's law. The linear system (4.4) can incorporate these additional equations describing the new unknowns

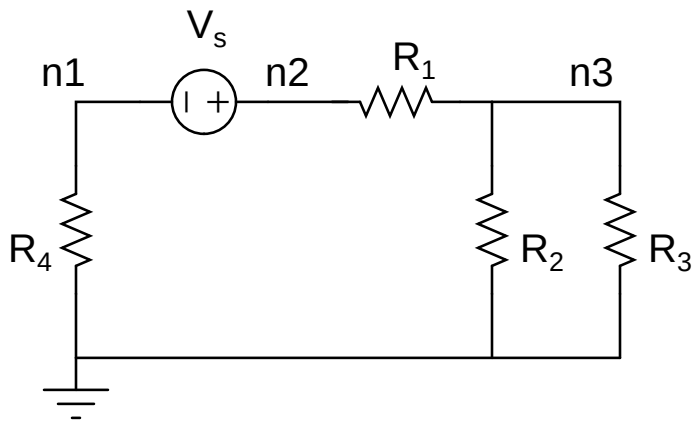


Figure 4.3: Example circuit to illustrate MNA

and their relationship with the other variables. Modified nodal analysis simply means augmenting nodal analysis with the required unknowns and constraints to solve the system in the most straightforward way.

4.2.2.1 Ideal voltage source

For example, a voltage source pins the relationship between two node voltages and cannot be described in terms of Ohm's Law. It requires a new equation to be added to the system (4.4). For a voltage source of value V_S with the positive node at n1 and the negative node at n2

$$V_{\text{n1}} - V_{\text{n2}} = V_S \tag{4.5}$$

The voltage source also supplies to the KCL equations at its two terminals as much current as needed to maintain the voltage drop across its terminals. This means that its current I_{V_S} is an unknown and is added to the **V** vector of unknowns.

4.2.2.2 An example

We illustrate MNA with an example. The circuit in Figure 4.3 has an ideal voltage source and some resistors. Treating ground as a known voltage, there are three unknown node voltages. Voltage source current is another unknown, resulting in a system of four equations with four unknowns.

The MNA setup with $G_n = 1/R_n$ for this circuit is

$$\begin{bmatrix} G_4 & 0 & 0 & -1 \\ 0 & G_1 & -G_1 & 1 \\ 0 & -G_1 & G_1+G_2+G_3 & 0 \\ -1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_{\mathrm{n1}} \\ V_{\mathrm{n2}} \\ V_{\mathrm{n3}} \\ I_{V_s} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ V_s \end{bmatrix}$$

4.2.3 Component-wise discretization

Component-wise discretization applies numerical integration formulas as discussed in Sec. 3.2 to approximate the derivatives associated with reactive components. The discretized components can then be represented as equivalent subcircuits with memory that can be incorporated into the circuit under analysis.

4.2.3.1 Capacitors

The differential equation in time relating capacitor voltage and current is

$$I = C\frac{dV}{dt}$$

Applying the trapezoidal rule to the derivative results in

$$v[n] - v[n-1] = \frac{T}{2C}\left(i[n] + i[n-1]\right)$$

Solving for current at the present time step n

$$i[n] = G_C v[n] + i_C[n-1] \tag{4.6}$$

where

$$i_C[n] = -\left(G_C v[n] + i[n]\right) \tag{4.7}$$

Equation (4.6) can be interpreted as a conductance $G_C = T\,/\,2C$, the equivalent discretized conductance of a capacitor, and a current source holding the state of the capacitor. The circuit representation of this is known as a companion circuit (Chua, 1975;

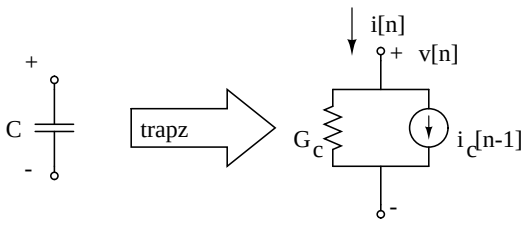


Figure 4.4: Discretized companion circuit for capacitor

Vladimirescu, 1994), shown in Fig. 4.4. The modified nodal representation can readily incorporate this description of a capacitor as a conductance and a current source. Additional code corresponding to the local discretization of the capacitor updates its state from the current and voltage of the capacitor at time n, using (A.3).

4.2.3.2 Inductors and mutual inductance

The companion circuit for inductors usually uses the Thevenin equivalent, that is, representing the companion circuit as a resistor and voltage source in series, and is similarly derived.

To handle inductors and mutual inductance in a unified fashion, we derive inductors differently.

Inductors are defined generally by a set of two equations. The magnetic flux Φ in an inductor depends on the current through that inductor (self inductance) and other coupled loops as well (mutual inductance). For n coupled inductors, the flux in inductor 1 is given by

$$\Phi_1 = L_1 I_1 + M_{12} I_2 + \ldots + M_{1n} I_n \tag{4.8}$$

The law of induction relates flux to voltage through a time derivative

$$V = \frac{d\Phi}{dt} \tag{4.9}$$

Applying the trapezoidal rule to the derivative results in

$$\phi[n] - \phi[n-1] = \frac{T}{2}\left(v[n] + v[n-1]\right)$$

Solving for voltage at the present time step n

$$v[n] = \frac{2}{T}\phi[n] + v_L[n-1] \tag{4.10}$$

where

$$v_L[n] = -\left(\frac{2}{T}\phi[n] + v[n]\right) \tag{4.11}$$

The inductor thus acts like a voltage source and adds two unknowns to MNA: flux and the voltage source current. The additional constraints added to the system to solve for these two unknowns are (4.8) and the companion circuit voltage source voltage as in (4.5).

4.2.4 Nonlinear devices

The nonlinear devices of interest in audio are diodes, and transistor-like devices, which are expressed as nonlinear voltage-controlled current sources. As current sources, they show up in the RHS vector. They also contribute conductance terms to the **M** matrix, and must be linearized about an operating point. Solving the linearized system repeatedly and updating the RHS vector and conductance contribution using voltages from the solution vector at each step is equivalent to using Newton's method.

4.2.5 Algorithmic construction of MNA

MNA is popular in circuit simulation because the algorithms to construct the matrix equation are simple and do not require graph theory.

Tables 4.2, 4.3 display the contribution of a circuit element to the row and column in **M** corresponding to a node voltage or other unknown variable. The rightmost column shows the contribution to the row in the vector of knowns **I**.

Resistor $G = 1/R$	col n+	col n-	
row n+	G	$-G$	
row n-	$-G$	G	

Capacitor C	col n+	col n-	**I**
row n+	G_C	$-G_C$	$-i_C$
row n-	$-G_C$	G_C	i_C

Current Source I	col n+	col n-	**I**
row n+			$-I$
row n-			I

Voltage Source V	col n+	col n-	col i_V	**I**
row n+			1	
row n-			-1	
row i_V	1	-1		V

Table 4.2: Transient Modified Nodal Analysis construction table for two terminal devices. Each element has two nodes labeled n+ and n-. Each element contributes the table entry to the corresponding column and row in the **M** "conductance" matrix, and currents to the corresponding rows in the **I** vector.

Inductor L	col n+	col n-	col ϕ_L	col i_L	**I**
row n+				1	
row n-				-1	
row i_L	1	-1	$-\gamma$		v_L
row ϕ_L			-1	L	

Mutual Coupling K	col i_{L1}	col i_{L2}
row ϕ_{L1}		M_{12}
row ϕ_{L2}	M_{12}	

Table 4.3: Transient Modified Nodal Analysis construction table for inductors and mutual inductance. Inductors add two unknowns to **V**: inductor flux ϕ_L and inductor current i_L. Discretization results in a voltage source representing the state of the inductor. Discretization constant $\gamma = 2/T$ for trapezoidal rule. Mutual coupling K between inductors L_1, L_2 yields a mutual inductance $M_{12} = K\sqrt{L_1 L_2}$

4.3 State-space approaches with memoryless nonlinearity

The following presents specific cases of a general class of solvers for problems that can be formulated in state-space with a memoryless nonlinearity (K-method) (Borin et al., 2000). This bears some similarity to Chua's hybrid circuit equations formulation (Chua, 1975). Assuming N memory elements and nonlinear devices, the hybrid formulation treats these elements as an N-port terminal, which is connected to the rest of the linear circuit. Relative to MNA, the hybrid formulation reduces the size of the matrix equation that needs to be solved to determine the unknown quantities for each sample. In a similar way, we observe that the nonlinear part can be extracted as a separate system to minimize the dimensionality of the matrix equation needed to solve this nonlinear part using Newton's method. Furthermore, the solution to the nonlinear part can be tabulated before runtime to eliminate the need for iterative methods inside the real-time loop.

4.3.1 Iteration on currents (Classical K-method)

For circuit nonlinearities in the audio frequency band, a memoryless, or static, nonlinearity well-characterizes circuit elements such as vacuum tubes, bipolar transistors (BJTs), and field effect transistors (FETs). Memoryless nonlinearities can even represent nonlinear capacitances after a change of state variable from current to charge. Nonlinearities with DC hysteresis such as remanence flux in magnetic cores of ferrite core inductors and transformers, however, cannot be represented as static.

This observation that most circuit nonlinearities are static leads to the following partitioned state-space representation of the differential equations for a circuit, developed by Borin et al. (2000) for the solution of nonlinear ordinary differential equations that arise in musical acoustics.

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{C}\mathbf{i}, \tag{4.12}$$

$$\mathbf{i} = \mathbf{f}(\mathbf{v}) \tag{4.13}$$

$$\mathbf{v} = \mathbf{D}\mathbf{x} + \mathbf{E}\mathbf{u} + \mathbf{F}\mathbf{i} \tag{4.14}$$

where $\mathbf{x}$ is a vector representing the state of the system, $\mathbf{u}$ represents the input, and $\mathbf{i}$ represents the contribution to the time derivative of the state from the nonlinear part through a nonlinear vector function $\mathbf{f}$. The symbol $\mathbf{i}$ is chosen for the nonlinear variable because typically nonlinear devices such as diodes, and transistor-like devices are voltage-controlled current sources, giving a nonlinear mapping from control voltages $\mathbf{v}$ across the sense terminals to the terminal currents.

The nonlinear dynamical system is partitioned into a dynamical part (4.12), and a purely static functional relationship that represents the overall nonlinearity of the circuit (4.14). The latter often expresses the nonlinear relationship between the circuit variables in implicit form, is often a transcendental equation, and must solved by iterative methods.

In the dynamical portion of this formulation, matrix multiplies by $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ represent linear combinations of the state, inputs, and nonlinear part that affect the evolution of the state. The nonlinear contribution is defined implicitly with respect to $\mathbf{i}$ and in general also depends on a linear combination of $\mathbf{x}$ and $\mathbf{u}$ (matrix multiplies by $\mathbf{D}$, $\mathbf{E}$, and $\mathbf{F}$).

4.3.1.1 Deriving the classical K-method

Discretizing (4.12) by the trapezoidal rule (equivalently, the bilinear transform) we can solve for $\mathbf{x}_n$, the circuit state at time n, and write

$$\mathbf{x}_n = \mathbf{H}(\alpha\mathbf{I}+\mathbf{A})\mathbf{x}_{n-1} + \mathbf{HB}(\mathbf{u}_n+\mathbf{u}_{n-1}) + \mathbf{HC}\,(\mathbf{i}_n+\mathbf{i}_{n-1}) \tag{4.15}$$

with $\mathbf{H} = (\alpha\mathbf{I}-\mathbf{A})^{-1}$, and $\alpha = 2/T$ if no frequency prewarping is used.

Discretization partitions the ODE into a state update (4.15) and a nonlinear system of equations. To derive this nonlinear system we substitute (4.15) into (4.14)

$$\begin{aligned}\mathbf{v}_n &= \mathbf{D}\,(\mathbf{H}(\alpha\mathbf{I}+\mathbf{A})\mathbf{x}_{n-1} + \mathbf{HB}(\mathbf{u}_n+\mathbf{u}_{n-1}) + \mathbf{HC}\,(\mathbf{i}_n+\mathbf{i}_{n-1})) + \mathbf{Eu}_n + \mathbf{Fi}_n \\ &= \underbrace{\mathbf{DH}\,((\alpha\mathbf{I}+\mathbf{A})\mathbf{x}_{n-1} + \mathbf{B}(\mathbf{u}_n+\mathbf{u}_{n-1}) + \mathbf{Ci}_{n-1}) + \mathbf{Eu}_n}_{\mathbf{p}_n} + (\mathbf{DHC}+\mathbf{F})\,\mathbf{i}_n\end{aligned} \tag{4.16}$$

Notice that we can parametrize (4.16) with respect to

$$\mathbf{p}_n = \mathbf{DH}\,((\alpha\mathbf{I}+\mathbf{A})\mathbf{x}_{n-1} + \mathbf{B}(\mathbf{u}_n+\mathbf{u}_{n-1}) + \mathbf{Ci}_{n-1}) + \mathbf{Eu}_n \tag{4.17}$$

because these terms are constant over the iterations to solve $\mathbf{i}_n$.

Next substituting (4.16) into (4.13) gives the implicit equation describing the nonlinearity of the circuit with this parametrization

$$\mathbf{i}_n = \mathbf{f}(\mathbf{p}_n + \mathbf{K}\mathbf{i}_n) \tag{4.18}$$

where

$$\mathbf{K} = \mathbf{DHC} + \mathbf{F} \tag{4.19}$$

If the mapping between $\mathbf{p}_n$ and $\mathbf{i}_n$ is functional, (4.18) can be represented as

$$\mathbf{i}_n = \mathbf{g}(\mathbf{p}_n) \tag{4.20}$$

– a function representing the memoryless nonlinearity of the system. This memoryless nonlinearity depends on the method of discretization because $\mathbf{K}$ depends on $\mathbf{H}$, which encapsulates the discretization parameters. Furthermore, (4.18) takes a linear transformation (4.17) of states and inputs to generate an effective input to the original nonlinear function. Borin et al. (2000) noted that this can be done prior to runtime to generate the explicit function $\mathbf{g}(\mathbf{p}_n)$.

4.3.1.2 Classical K-method summary

With discretization by trapezoidal rule, the K-Method procedure for state update can be summarized as

1. $\mathbf{p}_n = \mathbf{DH}\left((\alpha\mathbf{I} + \mathbf{A})\mathbf{x}_{n-1} + \mathbf{B}(\mathbf{u}_n + \mathbf{u}_{n-1}) + \mathbf{C}\mathbf{i}_{n-1}\right) + \mathbf{E}\mathbf{u}_n$
2. $\mathbf{i}_n = \mathbf{g}(\mathbf{p}_n)$
3. $\mathbf{x}_n = \mathbf{H}(\alpha\mathbf{I} + \mathbf{A})\mathbf{x}_{n-1} + \mathbf{HB}(\mathbf{u}_n + \mathbf{u}_{n-1}) + \mathbf{HC}(\mathbf{i}_n + \mathbf{i}_{n-1})$

In words, the runtime procedure performs a linear change of variable from current input and discretized system state (including past input and nonlinear contributions) to an effective input for a memoryless nonlinear function. The state update routine then takes the past state, current input and nonlinear contributions to compute the current state. This is illustrated in Fig. 4.5.

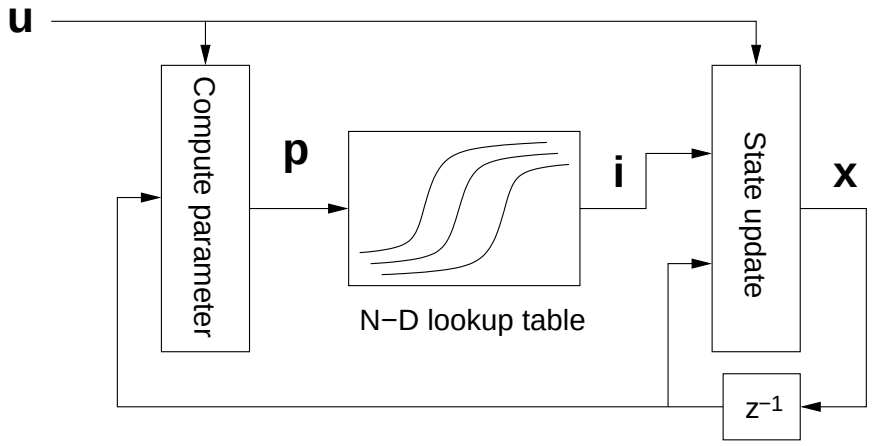


Figure 4.5: Signal flow diagram of K-method filter

Output y is generally expressed as a linear combination of the inputs, states, and nonlinear part

$$\mathbf{y}_n = \mathbf{A}_o\mathbf{x}_n + \mathbf{B}_o\mathbf{u}_n + \mathbf{C}_o\mathbf{i}_n.$$

using coefficient matrices $\mathbf{A}_o$, $\mathbf{B}_o$, $\mathbf{C}_o$.

Once this procedure is defined, the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$, $\mathbf{A}_o$, $\mathbf{B}_o$, $\mathbf{C}_o$, and the nonlinear device equations completely describe the system under consideration.

4.3.1.3 Newton's method for classical K-method

Newton's method is described in Sec. 3.2.2. The unknowns for the classical K-method are the device terminal currents $\mathbf{i}_n$ at each time step. Writing (4.18) in residual form

$$\mathbf{G}(\mathbf{i}_n) = \mathbf{0} = \mathbf{f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n) - \mathbf{i}_n \tag{4.21}$$

we find, by the matrix chain rule for differentiation,

$$\mathbf{J}_\mathbf{G}(\mathbf{i}_n) = \mathbf{J}_\mathbf{f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n)\mathbf{K} - \mathbf{I}. \tag{4.22}$$

Finally we arrive at the linearized system to be solved for the classical K-method:

$$\left(\mathbf{J}_\mathbf{f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n)\mathbf{K} - \mathbf{I}\right)\Delta\mathbf{i}_n = -\left(\mathbf{f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n) - \mathbf{i}_n\right). \tag{4.23}$$

Let N and M be the lengths of $\mathbf{i}$ and $\mathbf{v}$ respectively. Assuming that $N = M$, each iteration solving for $\Delta\mathbf{i}$ requires three dense matrix-vector multiplies of dimension N and four vector adds of length N.

4.3.1.4 Homotopy

To simplify offline tabulation of $\mathbf{g}(\mathbf{p}_n)$, (4.20) we propose the use of homotopy (Sec. 3.2.2.1) to help convergence for arbitrary input values when an appropriate initial guess is unavailable.

Applying homotopy amounts to solving the following system for the terminal currents $\mathbf{i}_n$:

$$(\mathbf{J_f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n)\,\mathbf{K} - \mathbf{I})\,\Delta\mathbf{i}_n = -\left(\mathbf{f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n) - \mathbf{i}_n + (\lambda - 1)\,\mathbf{G}(\mathbf{i}_n^*)\right) \tag{4.24}$$

With trivial solution $\mathbf{i}_n^* = \mathbf{0}$, $\mathbf{G}\,(\mathbf{i}_n^* = \mathbf{0}) = \mathbf{f}\,(\mathbf{p}_n)$ so that (4.24) becomes

$$(\mathbf{J_f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n)\,\mathbf{K} - \mathbf{I})\,\Delta\mathbf{i}_n = -\left(\mathbf{f}(\mathbf{K}\mathbf{i}_n + \mathbf{p}_n) - \mathbf{i}_n + (\lambda - 1)\,\mathbf{f}\,(\mathbf{p}_n)\right) \tag{4.25}$$

4.3.2 Iteration on terminal voltages (VK-method)

Alternatively, treating the controlling sense voltages of the nonlinear devices as the unknowns instead of the terminal currents we write equations to solve for the terminal voltages of the nonlinear devices.

The linear part of the system remains the same as in (4.12)

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{C}\mathbf{i}$$

where the terminal currents $\mathbf{i} = \mathbf{f}(\mathbf{v})$; $\mathbf{v}$ is the vector of control voltages.

The nonlinear system of equations corresponding to (4.14) is rewritten in terms of the new unknown $\mathbf{v}$.

$$\mathbf{v} = \mathbf{D}\mathbf{x} + \mathbf{E}\mathbf{u} + \mathbf{F}\mathbf{f}(\mathbf{v}) \tag{4.26}$$

Discretizing (4.12) by the trapezoidal rule we can solve for $\mathbf{x}_n$, the circuit state at time n, and write

$$\mathbf{x}_n = \mathbf{H}(\alpha\mathbf{I}+\mathbf{A})\mathbf{x}_{n-1} + \mathbf{HB}(\mathbf{u}_n+\mathbf{u}_{n-1}) + \mathbf{HC}\left(\mathbf{f}(\mathbf{v}_n)+\mathbf{f}(\mathbf{v}_{n-1})\right) \tag{4.27}$$

with $\mathbf{H} = (\alpha\mathbf{I}-\mathbf{A})^{-1}$, and $\alpha = 2/T$ as in the classical K-Method.

Substituting (4.27) in (4.26) and solving for a residual function parametrized by $\mathbf{p}_n$

$$\begin{aligned}\mathbf{v}_n &= \mathbf{D}\left(\mathbf{H}\left(\alpha\mathbf{I}+\mathbf{A}\right)\mathbf{x}_{n-1} + \mathbf{HB}\left(\mathbf{u}_n+\mathbf{u}_{n-1}\right) + \mathbf{HC}\left(\mathbf{f}(\mathbf{v}_n)+\mathbf{f}(\mathbf{v}_{n-1})\right)\right) + \mathbf{Eu}_n + \mathbf{Ff}(\mathbf{v}_n)\\ &= \mathbf{DH}\left(\left(\alpha\mathbf{I}+\mathbf{A}\right)\mathbf{x}_{n-1} + \mathbf{B}\left(\mathbf{u}_n+\mathbf{u}_{n-1}\right) + \mathbf{Cf}(\mathbf{v}_{n-1}) + \mathbf{Cf}(\mathbf{v}_n)\right) + \mathbf{Eu}_n + \mathbf{Ff}(\mathbf{v}_n)\end{aligned}$$

$$\mathbf{0} = \underbrace{\mathbf{DH}\left(\left(\alpha\mathbf{I}+\mathbf{A}\right)\mathbf{x}_{n-1} + \mathbf{B}\left(\mathbf{u}_n+\mathbf{u}_{n-1}\right) + \mathbf{Cf}(\mathbf{v}_{n-1})\right) + \mathbf{Eu}_n}_{\mathbf{p}_n} - \mathbf{v}_n + \mathbf{DHCf}(\mathbf{v}_n) + \mathbf{Ff}(\mathbf{v}_n)$$

Using $\mathbf{K}$ as previously defined (4.19)

$$\mathbf{G}(\mathbf{v}_n) \equiv \mathbf{p}_n - \mathbf{v}_n + \mathbf{Kf}(\mathbf{v}_n) = \mathbf{0} \tag{4.28}$$

where the change of variable to the parameter of the nonlinearity

$$\mathbf{p}_n = \mathbf{DH}\left(\left(\alpha\mathbf{I}+\mathbf{A}\right)\mathbf{x}_{n-1} + \mathbf{B}\left(\mathbf{u}_n+\mathbf{u}_{n-1}\right) + \mathbf{Cf}(\mathbf{v}_{n-1})\right) + \mathbf{Eu}_n \tag{4.29}$$

is identical to (4.17).

4.3.2.1 Newton's method

To complete the derivation, we find the Newton's method parameters for iterating on the control voltages. The Jacobian of the residual function (4.28) is

$$\mathbf{J}_\mathbf{G}(\mathbf{v}) = \mathbf{KJ}_\mathbf{f}(\mathbf{v}) - \mathbf{I} \tag{4.30}$$

The following equation is then solved for the terminal voltages

$$\left(\mathbf{KJ}_\mathbf{f}(\mathbf{v}_n) - \mathbf{I}\right)\Delta\mathbf{v}_n = -\left(\mathbf{p}_n - \mathbf{v}_n + \mathbf{Kf}(\mathbf{v}_n)\right) \tag{4.31}$$

For N nonlinear current sources in the system, each iteration requires two dense matrix-vector multiplies of dimension N, and three vector adds of length N, and is slightly more efficient than the classical K-method.

4.3.2.2 Homotopy

Applying homotopy, the system to be solved to obtain $\Delta\mathbf{v}$ is

$$(\mathbf{K}\mathbf{J}_{\mathbf{f}}(\mathbf{v}) - \mathbf{I})\,\Delta\mathbf{v}_n = -\left(\mathbf{p}_n - \mathbf{v}_n + \mathbf{K}\mathbf{f}(\mathbf{v}_n) + (\lambda - 1)\,\mathbf{G}(\mathbf{v}_n^*)\right) \tag{4.32}$$

Observing that $\mathbf{G}\left(\mathbf{v}_n^* = \mathbf{0}\right) = \mathbf{p}_n$ because $\mathbf{f}\left(\mathbf{v} = \mathbf{0}\right) = \mathbf{0}$ (i.e., all nonlinear devices have zero terminal current for zero applied voltage), (4.32) becomes

$$(\mathbf{K}\mathbf{J}_{\mathbf{f}}(\mathbf{v}) - \mathbf{I})\,\Delta\mathbf{v}_n = -\left(\mathbf{p}_n - \mathbf{v}_n + \mathbf{K}\mathbf{f}(\mathbf{v}_n) + (\lambda - 1)\,\mathbf{p}_n\right)$$

$$(\mathbf{K}\mathbf{J}_{\mathbf{f}}(\mathbf{v}) - \mathbf{I})\,\Delta\mathbf{v}_n = -\left(\lambda\mathbf{p}_n - \mathbf{v}_n + \mathbf{K}\mathbf{f}(\mathbf{v}_n)\right) \tag{4.33}$$

4.3.3 Discussion

We have derived equations to solve a general K-method system by iterating on either voltage or current. Iterating on voltage is slightly more efficient than iterating on current because, in typical nonlinear device equations, the voltage is a compressive function of the current, whereas the current is an expansive function of the voltage.

The numerical stability of iterating on terminal currents versus voltages needs to be evaluated case by case, but empirically we have found that solving for voltage has a larger region of convergence and requires fewer iterations than solving for current.

Researchers originally developed homotopy as an advanced method for solving the difficult problem of finding DC solutions of a circuit. For transient analysis, circuit simulators normally resort to decreasing the time step so that using the previous solution as the initial guess will fall in the region of convergence for Newton's method.

Because a fixed-step size is desired for real-time simulation, we apply the homotopy technique to facilitate the convergence problem in transient simulation. This allows the

robust offline solution of a nonlinear dynamical circuit and eliminates the unreliable process of trying to converge in real-time.

For the large input levels common in guitar distortion circuits, computation of $\mathbf{g}(\mathbf{p}_n)$ by Newton's method requires manual supervision to initialize the iterations within the region of convergence. The simulation examples in Ch. 5 do not converge by Newton's method for large inputs without some method to improve convergence, such as varying the time-step, or homotopy. When homotopy is used with smooth and continuous device equations, convergence is very robust, and reduces the need for manual tuning of simulation parameters. In particular, Newton's method failed to converge for the simulation of the BJT amplifier in Sec. 5.5.2, and, instead, homotopy was used to obtain the results shown.

4.4 Systematic derivation of K-method systems (NK-method)

In the course of this research it was found that analyzing each circuit for implementation manually was tedious and error-prone. A novel contribution of this work is to develop a system for generating the parameters for the K-method formulation given the netlist description of a circuit. It derives the K-method parameters using an algorithm similar to MNA and is known as the Nodal K-method (NK-method).

4.4.1 Development of NK-method

For NK-method, we desire a system in the form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{C}\mathbf{i},$$

$$\mathbf{i} = \mathbf{f}(\mathbf{v})$$

$$\mathbf{v} = \mathbf{D}\mathbf{x} + \mathbf{E}\mathbf{u} + \mathbf{F}\mathbf{i}$$

The unknowns desired are the time derivatives of the state elements. The state $\mathbf{x}$, inputs $\mathbf{u}$, and nonlinear devices currents $\mathbf{i}$ are considered knowns.

We observe that MNA (4.4) sets up a system of equations based upon circuit laws to solve for unknowns. Solving the system, the solution vector $\mathbf{V}$ will be described in terms of only variables given in the RHS vector $\mathbf{I}$. We set up a matrix system of equations in the following form, decomposing the RHS vector into the three known vectors and their coefficient matrices

$$\mathbf{MV} = \mathbf{M}_1\mathbf{x} + \mathbf{M}_2\mathbf{u} + \mathbf{M}_3\mathbf{i} \tag{4.34}$$

Essentially we augment the MNA formulation to solve for all the unknown node voltages along with the variables in $\dot{\mathbf{x}}$ expressing capacitor dynamics, etc. For example

$$\mathbf{V} = \begin{bmatrix} V_1 & \dots & \dot{V}_{C1} \end{bmatrix}^T$$

Inverting $\mathbf{M}$ in (4.34) we can solve for the vector of unknowns

$$\mathbf{V}[n] = \mathbf{A}'\mathbf{x}[n-1] + \mathbf{B}'\mathbf{u}[n] + \mathbf{C}'\mathbf{i}[n] \tag{4.35}$$

Taking the rows of $\mathbf{A}' \triangleq \mathbf{M}^{-1}\mathbf{M}_1$, $\mathbf{B}' \triangleq \mathbf{M}^{-1}\mathbf{M}_2$, $\mathbf{C}' \triangleq \mathbf{M}^{-1}\mathbf{M}_3$ corresponding to the state derivatives gives $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ of the K-method. The $\mathbf{v}$ vector expresses voltage differences across the controlling terminals of the nonlinear elements. Given a list of nonlinear devices, the NK-method builds the list of *controlling pairs* of nodes. Taking the rows in $\mathbf{A}'$, $\mathbf{B}'$, $\mathbf{C}'$ corresponding to the two nodes of a controlling pair and subtracting them gives the row in $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$ for that controlling pair.

4.4.2 Element tables for NK-method

NK-method sets up the system (4.34) in a fashion similar to MNA because each kind of circuit element contributes particular elements to the system matrices. These are summarized in the following tables. The main difference is that elements contribute to $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$ instead of $\mathbf{I}$. Also, nonlinear elements are considered pure current sources and do not contribute to $\mathbf{M}$. Reactive elements are also handled differently, adding additional variables and, hence, rows to $\mathbf{M}$. Similar tables can be derived for inductors, mutual inductors (transformers), FETs, and vacuum tube tetrodes/pentodes, but are not shown here.

Resistor $R = 1/G$

M	**col n+**	**col n-**
row n+	G	$-G$
row n-	$-G$	G

Capacitor C

Adds unknown $\dot{V}_C$ to row eq_i in $\mathbf{V}$,

Adds state V_C to row state_i in $\mathbf{x}$

M	**col n+**	**col n-**	**col eq_i**
row n+			C
row n-			$-C$
row eq_i	1	-1	

$\mathbf{M}_1$	col state_i
row eq_i	1

VSource V

Adds source V to row src_i in $\mathbf{u}$,

Adds unknown current I_V to row eq_i in $\mathbf{V}$

M	**col n+**	**col n-**	**col eq_i**
row n+			1
row n-			-1
row eq_i	1	-1	

$\mathbf{M}_2$	col src_i
row eq_i	1

ISource *I*

Adds source *I* to row src_i in **u**

$\mathbf{M}_2$	**col src_i**
row n+	-1
row n-	1

Junction Diode

Adds one controlling pair (n+, n-) to row cp_i in **v**

Adds one nonlinear current $I_d(\mathbf{v}[\text{cp\_i}])$ to row nl_i in **i**

$\mathbf{M}_3$	**col nl_i**
row n+	-1
row n-	1

$\mathbf{J_f}$	col cp_i
row nl_i	$\frac{\partial I_d}{\partial V_d}\Big\vert_{\mathbf{v}[\text{cp\_i}]}$

Bipolar Junction Transistor

Has nodes labeled with indices nc, nb, ne for collector, base, and emitter

Adds controlling pairs V_{be}, $V_{bc}$to rows be_i, bc_i in **v**

Adds nonlinear currents $I_e(\mathbf{v}[\text{be\_i}], \mathbf{v}[\text{bc\_i}])$, $I_c(\mathbf{v}[\text{be\_i}], \mathbf{v}[\text{bc\_i}])$ to rows e_i, c_i in **i**

$\mathbf{M}_3$	**col e_i**	**col c_i**
row nc		-1
row nb	1	1
row ne	-1	

$\mathbf{J_f}$	col be_i	col bc_i
row e_i	$\frac{\partial I_e}{\partial V_{be}}\Big\vert_{(\mathbf{v}[\text{be\_i}],\mathbf{v}[\text{bc\_i}])}$	$\frac{\partial I_e}{\partial V_{bc}}\Big\vert_{(\mathbf{v}[\text{be\_i}],\mathbf{v}[\text{bc\_i}])}$
row c_i	$\frac{\partial I_c}{\partial V_{be}}\Big\vert_{(\mathbf{v}[\text{be\_i}],\mathbf{v}[\text{bc\_i}])}$	$\frac{\partial I_c}{\partial V_{bc}}\Big\vert_{(\mathbf{v}[\text{be\_i}],\mathbf{v}[\text{bc\_i}])}$

Vacuum Tube Triode

Has nodes labeled with indices na, ng, nk for anode, grid, and cathode

Adds controlling pairs V_{gk}, $V_{ak}$to rows gk_i, ak_i in **v**

Adds nonlinear currents $I_a(\mathbf{v}[\text{gk\_i}], \mathbf{v}[\text{ak\_i}])$, $I_g(\mathbf{v}[\text{gk\_i}], \mathbf{v}[\text{ak\_i}])$ to rows a_i, g_i in **i**

$\mathbf{M}_3$	**col a_i**	**col g_i**
row na	-1	
row ng		-1
row nk	1	1

$\mathbf{J_f}$	col gk_i	col ak_i
row a_i	$\frac{\partial I_a}{\partial V_{gk}}\Big\vert_{(\mathbf{v}[\text{gk\_i}],\mathbf{v}[\text{ak\_i}])}$	$\frac{\partial I_a}{\partial V_{ak}}\Big\vert_{(\mathbf{v}[\text{gk\_i}],\mathbf{v}[\text{ak\_i}])}$
row g_i	$\frac{\partial I_g}{\partial V_{gk}}\Big\vert_{(\mathbf{v}[\text{gk\_i}],\mathbf{v}[\text{ak\_i}])}$	$\frac{\partial I_g}{\partial V_{ak}}\Big\vert_{(\mathbf{v}[\text{gk\_i}],\mathbf{v}[\text{ak\_i}])}$

4.5 DK-method - Discrete State-Space with Memoryless Nonlinearity

A limitation of the K-Method when applied to nonlinear, continuous-time state-space systems is that often systems are expressed as Differential Algebraic Equations (DAE).

$$\mathbf{M}\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}, \mathbf{u})$$

There is a "mass-matrix" $\mathbf{M}$ that is often singular and cannot be represented in the K-Method form. It is well known in the DAE community that, in general, $\mathbf{M}$ may be singular in circuits, and methods to circumvent this problem are an active area of research.

The discretization for transient simulations in SPICE produces effective conductances for the state elements. These contribute to nonsingular conductance matrices. Instead of solving for continuous-time state-space representations of nonlinear circuits as in the K-Method, we propose to discretize the state elements first and then solve the resulting system for the compact state-space representation. This approach fuses the concepts of the K-Method (Borin et al., 2000), the hybrid state-space circuit (Chua, 1975), and transient MNA formulations (Vladimirescu, 1994). We call this the Discrete K-method (DK-method)

4.5.1 Component-wise discretization

Component-wise discretization of circuit state elements results in companion circuits as done for transient MNA. Appendix A derives the discretized inductors and capacitors for the DK-method using both trapezoidal rule and Backward Euler. The discretization results in a state update that can be written in general as

$$x[n] = G\,v[n] + s\,x[n-1] \tag{4.36}$$

For capacitors, $G = 2G_C(T)$, where $G_C(T)$ is the discrete conductance of the capacitor, a function of the sampling rate T, $x[n]$ is the capacitor equivalent source current at time n, and $s = -1$. For inductors $G = 2$, $x[n]$ is the inductor equivalent voltage at time n, and $s = 1$.

For systems with multiple state elements, we can write the state update (4.36) in vector form

$$\mathbf{x}[n] = \mathbf{G}\mathbf{v}[n] + \mathbf{S}\mathbf{x}[n-1] \tag{4.37}$$

where $\mathbf{G}$ is a diagonal matrix comprising the scalars relating elements' circuit variables v (element currents or voltages) to their simulation state variables x, which depend on the discretization method. Diagonal matrix $\mathbf{S}$ multiplies each state by the appropriate sign s for state update.

4.5.2 Modified nodal analysis

These state elements can then be incorporated in MNA

$$\mathbf{M}\mathbf{V} = \mathbf{I}$$

as conductances in matrix $\mathbf{M}$ and sources $\mathbf{I}$, where $\mathbf{V}$ is the vector of node voltages and unknown terminal currents.

We seek an expression for the element variables in the form

$$\mathbf{v}[n] = \mathbf{A_e}\mathbf{x}[n-1] + \mathbf{B_e}\mathbf{u}[n] + \mathbf{C_e}\mathbf{i}[n] \tag{4.38}$$

where vector $\mathbf{v}$ comprises the voltages across or the current through the state elements, vector $\mathbf{u}$ comprises the sources in the circuit, and vector $\mathbf{i}$ comprises the terminal currents of the nonlinear elements.

By decomposing the source vector $\mathbf{I}$ in (4.4) into contributions from states, inputs, and nonlinear elements, we can write a system in the form

$$\mathbf{M}\mathbf{V}[n] = \mathbf{M_1}\mathbf{x}[n-1] + \mathbf{M_2}\mathbf{u}[n] + \mathbf{M_3}\mathbf{i}[n] \tag{4.39}$$

Owing to the conductances of the discretized state elements, $\mathbf{M}$ will be nonsingular and the system can be solved to find the node voltages/solution currents $\mathbf{V}[n]$ as in classical MNA. Unlike MNA, which solves the sparse system by LU decomposition, we must invert

M to find the matrices that multiply **x**, **u**, and **i**, yielding expressions for the nodal solution in terms of the given variables.

$$\mathbf{V}[n] = \mathbf{A}^{'}\mathbf{x}[n-1] + \mathbf{B}^{'}\mathbf{u}[n] + \mathbf{C}^{'}\mathbf{i}[n] \tag{4.40}$$

Rows of $\mathbf{A}' \triangleq \mathbf{M}^{-1}\mathbf{M}_1$, $\mathbf{B}' \triangleq \mathbf{M}^{-1}\mathbf{M}_2$, $\mathbf{C}' \triangleq \mathbf{M}^{-1}\mathbf{M}_3$ correspond to the weights of the known variables to give the nodal voltage solution. Subtracting pairs of rows corresponding to terminal voltages of elements finds the voltages across those elements. Expressions for element currents can be found from the appropriate rows as well. This now yields the desired

$$\mathbf{v}[n] = \mathbf{A_e}\mathbf{x}[n-1] + \mathbf{B_e}\mathbf{u}[n] + \mathbf{C_e}\mathbf{i}[n] \tag{4.41}$$

giving the element voltages or currents **v**. Combining (4.41) with (4.37) gives a state update equation

$$\mathbf{x}[n] = \mathbf{A}\mathbf{x}[n-1] + \mathbf{B}\mathbf{u}[n] + \mathbf{C}\mathbf{i}[n] \tag{4.42}$$

where $\mathbf{A} = \mathbf{G}\mathbf{A_e} + \mathbf{S}$, $\mathbf{B} = \mathbf{G}\mathbf{B_e}$, $\mathbf{C} = \mathbf{G}\mathbf{C_e}$.

4.5.3 Solving the nonlinearity

4.5.3.1 Newton's method

Borrowing from the key idea of the K-method to use a change of variable and solve the nonlinearity ahead of time, and observing that the number of unknowns in the nonlinear system can be minimized by solving for the control voltages of the nonlinearities, we write an equation to describe the control voltages by again subtracting the appropriate rows of (4.40):

$$\mathbf{v_{nl}} = \mathbf{D}\mathbf{x}[n-1] + \mathbf{E}\mathbf{u}[n] + \mathbf{F}\,\mathbf{f}(\mathbf{v_{nl}}[n]) \tag{4.43}$$

where $\mathbf{i} = \mathbf{f}(\mathbf{v_{nl}})$

We observe that (4.43) can be rewritten as

$$\mathbf{0} = \mathbf{D}\mathbf{x}[n-1] + \mathbf{E}\mathbf{u}[n] + \mathbf{F}\,\mathbf{f}(\mathbf{v_{nl}}[n]) - \mathbf{v_{nl}}[n] \tag{4.44}$$

and solved for the control voltages $\mathbf{v_{nl}}$ by Newton's method as for the K-method. The solution can also be parametrized with respect to

$$\mathbf{p}[n] = \mathbf{D}\mathbf{x}[n-1] + \mathbf{E}\mathbf{u}[n] \tag{4.45}$$

yielding a system

$$\mathbf{G}(\mathbf{v_{nl}}) = \mathbf{0} = \mathbf{p} + \mathbf{F}\,\mathbf{f}(\mathbf{v_{nl}}) - \mathbf{v_{nl}} \tag{4.46}$$

giving an implicit relation between $\mathbf{p}[n]$ and $\mathbf{v_{nl}}[n]$. Because $\mathbf{i}[n]$ is needed in the state update expression, the function to be precomputed should be $\mathbf{i}[n] = \mathbf{g}(\mathbf{p}[n])$.

Equation (4.46) can be solved by Newton's method. The Jacobian of the residual is

$$\mathbf{J_G}(\mathbf{v}) = \mathbf{F}\,\mathbf{J_f}(\mathbf{v}) - \mathbf{I} \tag{4.47}$$

and the linearized system to solve per iteration is

$$(\mathbf{F}\,\mathbf{J_f}(\mathbf{v}) - \mathbf{I})\,\Delta\mathbf{v} = -\left(\mathbf{p} + \mathbf{F}\,\mathbf{f}(\mathbf{v_{nl}}) - \mathbf{v_{nl}}\right) \tag{4.48}$$

where $\mathbf{J_f}$ is the Jacobian of the nonlinear current vector function with respect to the vector of control voltages, and $\mathbf{J_G}$ is the Jacobian of the residual function (4.46) with respect to the control voltages.

4.5.3.2 Homotopy

Using homotopy, the system to be solved at each iteration becomes

$$(\mathbf{F}\,\mathbf{J_f}(\mathbf{v}_n) - \mathbf{I})\,\Delta\mathbf{v} = -\left(\mathbf{p}_n + \mathbf{F}\,\mathbf{f}(\mathbf{v}_n) - \mathbf{v}_n + (\lambda - 1)\,\mathbf{G}(\mathbf{v}_n^*)\right)$$

Since $\mathbf{v}_n^* = \mathbf{0}$, $\mathbf{G}(\mathbf{v}_n^*) = \mathbf{p}_n$,

$$\begin{aligned}(\mathbf{F}\,\mathbf{J_f}(\mathbf{v}_n) - \mathbf{I})\,\Delta\mathbf{v} &= -\left(\mathbf{p}_n + \mathbf{F}\,\mathbf{f}(\mathbf{v}_n) - \mathbf{v}_n + (\lambda - 1)\,\mathbf{p}_n\right)\\ (\mathbf{F}\,\mathbf{J_f}(\mathbf{v}_n) - \mathbf{I})\,\Delta\mathbf{v} &= -\left(\mathbf{F}\,\mathbf{f}(\mathbf{v}_n) - \mathbf{v}_n + \lambda\,\mathbf{p}_n\right)\end{aligned} \tag{4.49}$$

4.5.4 Summary

The runtime loop for this hybrid K-method/transient MNA is also depicted by Fig. 4.5 and can be summarized as

1. $\mathbf{p}[n] = \mathbf{D}\mathbf{x}[n-1] + \mathbf{E}\mathbf{u}[n]$ (change of variables)
2. $\mathbf{i}[n] = \mathbf{f}(\mathbf{p}[n])$ (nonlinear function lookup)
3. $\mathbf{x}[n] = \mathbf{A}\mathbf{x}[n-1] + \mathbf{B}\mathbf{u}[n] + \mathbf{C}\mathbf{i}[n]$ (state update)

This represents the discrete-time numerical solution to the circuit in a compact form involving only matrix multiplies and function lookups, with no need for iteration during runtime.

4.5.5 DK-method Element Tables

The element tables for DK-method are the same as for NK-method except for the memory elements, which are discretized in the same fashion as for transient MNA.

Capacitor C

Adds state i_C to row state_i in $\mathbf{x}$

$G_C = T\,/\,2C$ for trapezoidal integration

M	**col n+**	**col n-**
row n+	G_C	$-G_C$
row n-	$-G_C$	G_C

$\mathbf{M}_1$	col state_i
row n+	1
row n-	-1

Inductor L

Adds state v_L to row state_i in $\mathbf{x}$

Adds unknown I_L to row eq_i in $\mathbf{V}$, which also contributes to KCL at terminal nodes

$Z_L = 2L\,/\,T$ for trapezoidal integration

M	**col n+**	**col n-**	**col eq_i**
row n+			1
row n-			-1
row eq_i	-1	1	Z_L

$\mathbf{M}_1$	col state_i
row eq_i	1

Mutual Inductance K between inductors $L1$, $L2$

For eq_i$_{L1}$, eq_i$_{L2}$, $M_{12} = K\sqrt{L1\,L2}$

M	**col eq_i$_{L1}$**	**col eq_i$_{L2}$**
row eq_i$_{L1}$		M_{12}
row eq_i$_{L2}$	M_{12}	

Chapter 5

Applications of Selected Simulation Methods

The following examples apply the WDF and K-method formulations to various circuits found in guitar electronics. We performed all simulations for Secs. 5.1–5.4, first presented in (Yeh and Smith, 2008), using custom code in MATLAB and utilizing Newton's method to solve any nonlinear equations.

5.1 Bright switch/filter

The bright switch (Fig. 5.1) is commonly found in guitar amplification circuits and sometimes in the electronics of the electric guitar itself as part of a volume knob implemented as a resistive voltage divider. When engaged, it provides a low impedance path for high frequencies to bypass part of the voltage divider.

The WDF tree for the bright switch is shown in Fig. 5.2 and its signal processing algorithm can be derived by inspection from this tree. The capacitor was chosen to be at the top of the tree because, when it is disconnected, the open circuit at the top of the tree produces an instantaneous reflection. Using the reflection-free wave digital capacitor sets its port impedance and requires a parallel two-port adapter P2 to match the impedance of the rest of the tree. The bright switch is suited to implementation as a WDF because of its

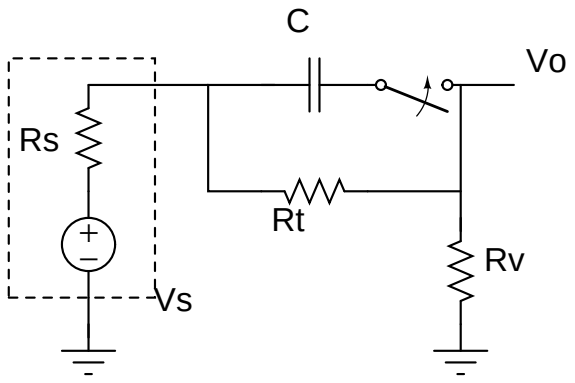


Figure 5.1: Schematic of the bright switch from guitar electronics.

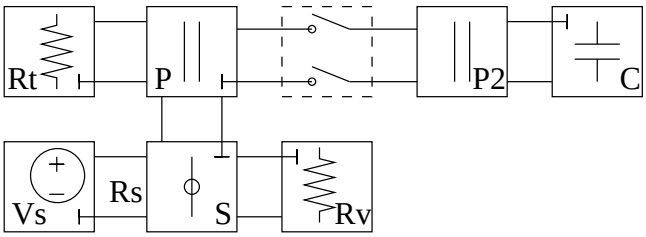


Figure 5.2: WDF tree to implement the bright switch. Adaptor P2 is root.

switched nature. The computational blocks in a WDF correspond directly to the physical circuit elements, which can easily be rearranged to reflect a different structure.

Figure 5.3 shows the magnitude response of the bright switch simulated using the values $C = 120\text{pF}$, $R_t = (1 - \text{vol})\text{M}\Omega$, $R_v = (\text{vol})\text{M}\Omega$, $R_s = 100\text{k}\Omega$, where $\text{vol} \in [0, 1]$ is the value of the volume potentiometer.

5.2 Two-capacitor diode clipper

The behavior of various numerical methods applied to the single capacitor diode clipper was studied extensively in Sec. 3.4. Here we consider the diode clipper including the effects of the DC blocking capacitor. The nonlinearities cause the pole of the high-pass frequency to move depending on the signal level. Because this pole is at low frequencies, this could have a notably audible effect, especially in the presence of transients.

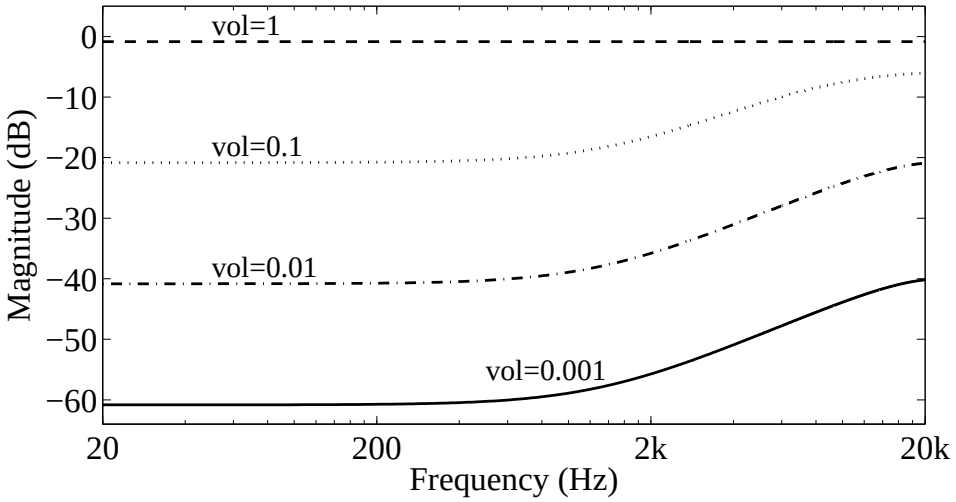


Figure 5.3: Magnitude response of the volume attenuator with bright switch engaged for values of volume as shown.

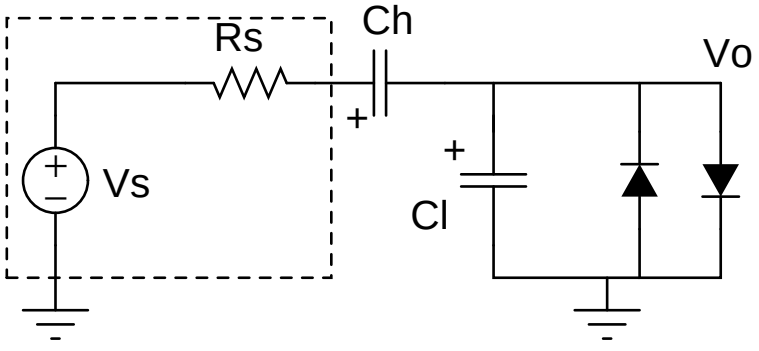


Figure 5.4: Schematic of the diode clipper with high-pass and low-pass capacitors.

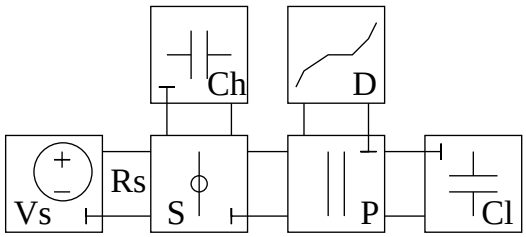


Figure 5.5: WDF tree of the two-capacitor diode clipper. Diode D is the root of the tree.

The two-capacitor diode clipper is shown in Fig. 5.4. The two diodes are two physically separate nonlinearities but can be combined into a single equivalent nonlinearity summing their currents by KCL because they are connected in parallel. In the WDF, the two diodes are modeled by the voltage-controlled current source

$$I_d(V) = 2I_s \sinh\left(V/V_t\right), \tag{5.1}$$

where $I_d(V)$ is the current through the two diodes, I_s and $V_t$are physical parameters of the diodes, and V is the controlling voltage across the diodes.

Device parameters for the following simulations are $R_s = 2.2\text{k}\Omega$, $C_h = 0.47\mu\text{F}$, $C_l = 0.01\mu\text{F}$, $I_s = 2.52 \times 10^{-9}\text{A}$, and $V_t = 45.3\text{mV}$.

5.2.1 WDF implementation

The current state of WDF technology is well suited for modeling circuits connected in series and parallel and with a single one-port nonlinearity. The diode clipper is a prime example of this. The tree corresponding to the computational structure of the WDF for the diode clipper is shown in Fig. 5.5. The input is the voltage source V_s and the output is the junction voltage of the parallel adaptor.

The nonlinear relationship between the incident and reflected waves to this block in the WDF is derived by substituting the wave variable definitions (4.2) into (5.1) and solving the resulting implicitly defined nonlinear function for $b = f(a)$ using numerical methods.

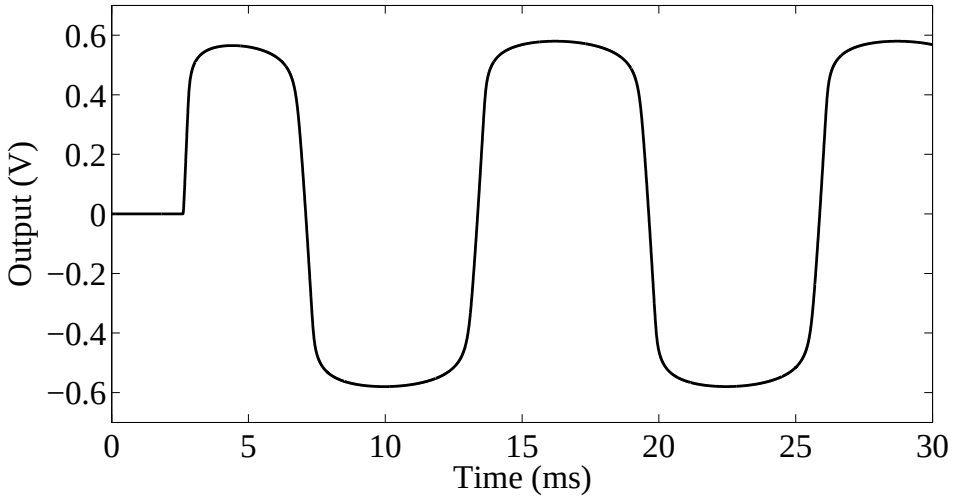


Figure 5.6: Simulated output of two-capacitor diode clipper for sine input with frequency 80 Hz, amplitude 4.5 V. Identical output is produced by WDF and K-method.

Specifically, the nonlinear equation to be solved is

$$2I_s \sinh\left(\frac{a+b}{2V_t}\right) - \frac{a-b}{2R_p} = 0.$$

Conditions for which a solution exists are given in (Meerkötter and Scholz, 1989; Sarti and De Poli, 1999).

This nonlinear relationship $b = f(a)$ is then placed at the top of the tree representing the rest of the diode clipper to prevent delay-free loops in the signal processing algorithm.

This example demonstrates the power of the WDF formulation to build up algorithms modularly to simulate circuits. The additional high-pass capacitor C_h is added to the single capacitor diode clipper by replacing the original resistive voltage source (inside the dotted box in Fig. 5.4) with the series combination of the source and the capacitor. Thus, the structure of the WDF derives directly from the connectivity of the modeled circuit.

5.2.2 K-Method implementation

The matrices for the K-method representation of the diode clipper can be found by application of KCL at the two nodes with unknown voltages.

These equations are

$$I_{Ch} = C_h \dot{V}_{Ch} = G_s(V_s - V_x)$$

$$C_l \dot{V}_o = C_h \dot{V}_{Ch} - I_d(V_o).$$

Choosing the state variables to be the voltages across the two capacitors $\mathbf{x} = \begin{bmatrix} V_o & V_{Ch} \end{bmatrix}^T$ with the polarity of the voltages indicated by + on the capacitors in Fig. 5.4, we solve for $\dot{V}_o$ and $\dot{V}_{Ch}$ using $V_x = V_o + V_{Ch}$ to define the intermediate node voltage in terms of the state variables, and set $u = V_s$ to be the input. We let the nonlinear part $i = I_d(V_o)$, which makes $v = V_o$ the input to the nonlinearity. The state variable V_o is also the output.

$$\dot{V}_o = \frac{G_s}{C_l}(V_s - V_o - V_{Ch}) - \frac{1}{C_l} I_d(V_o)$$

$$\dot{V}_{Ch} = \frac{G_s}{C_h}(V_s - V_o - V_{Ch})$$

The resulting K-method matrices are

$$\mathbf{A} = \begin{bmatrix} -G_s/C_l & -G_s/C_l \\ -G_s/C_h & -G_s/C_h \end{bmatrix}, \qquad \mathbf{D} = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} G_s/C_l \\ G_s/C_h \end{bmatrix}, \qquad \mathbf{E} = \begin{bmatrix} 0 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} -1/C_l \\ 0 \end{bmatrix}, \qquad \mathbf{F} = \begin{bmatrix} 0 \end{bmatrix}.$$

5.2.2.1 Simulation results

The output of the diode clipper simulated using an input signal of 80 Hz, 4.5 V amplitude, is plotted in Fig. 5.6. The sampling rate was 8× oversampled the audio sampling rate of 48000 Hz to reduce signal aliasing in the output. Identical output was produced by the nonlinear WDF and the K-method. Notice the first cycle of the output has a different period than the steady-state response, indicating that the high-pass capacitor does indeed affect the response of the circuit to transients and should be included in models.

5.2.2.2 Comparative discussion

The signal flow diagram to update the state for both methods involve linear operations followed by a static nonlinearity and more linear operations. The nonlinearity is also present in the discrete-time feedback loop, which alters the order of the nonlinearity.

Considering the nonlinearity as a separate operation of comparable cost, the WDF requires only 4 multiplies and 8 adds to implement the diode clipper. In contrast, the K-method using a straightforward implementation of matrix-vector multiplication requires 13 multiplies and 12 adds. However, it is not straightforward to derive WDFs for the examples that follow.

5.3 Common-emitter transistor amplifier with feedback

Figure 5.7 shows the common-emitter amplification stage from the Boss DS-1 (Yeh et al., 2007a), which employs a bipolar junction transistor (BJT) in the shunt-shunt feedback configuration, giving rise to a transimpedance amplifier. The feedback resistor exists mainly to bias the base of the BJT at a desired operating point. The circuit also features mild emitter degeneration as is common with these amplifiers, which reduces the gain and improves the small-signal linearity of the stage. Because of the high gain from node b to node c, this stage is highly sensitive to the DC bias voltage of node b, which is determined by the design of the circuit. Using incorrect resistor values affects the output swing, which in turn influences the shape and symmetry of the clipped output.

The design values for this circuit are $R_i = 100\text{k}\Omega$, $R_c = 10\text{k}\Omega$, $R_l = 100\text{k}\Omega$, $R_f = 470\text{k}\Omega$, $R_e = 22\Omega$, $C_i = 0.047\mu\text{F}$, $C_f = 250p\text{F}$, and $C_o = 0.47\mu\text{F}$.

5.3.1 Bipolar Junction Transistor (BJT) device model

Figure 5.8 depicts generic model for the bipolar junction transistor comprising voltage-controlled current sources. The BJT has three terminals, the collector, base, and emitter, whose currents are controlled by voltages across two pairs of the terminals, $V_{be} = V_b - V_e$, $V_{bc} = V_b - V_c$. By conservation of current, only two of the terminal current definitions are needed to completely describe the current-voltage (I-V) characteristics. We use $I_b(V_{be}, V_{bc})$

and $I_c(V_{be}, V_{bc})$ here. Semiconductor devices such as the BJT also have nonlinear resistances and capacitors, which require more detailed models; however, for simplicity, we assume that we can neglect these effects for the signal levels of this circuit in the audio frequency band.

A complete, yet simple, physically derived model for computer simulation, the Ebers-Moll model (Muller et al., 2002) defines the following current-voltage (I-V) characteristics:

$$I_e = \frac{I_S}{\alpha_F}\left[\exp\left(V_{be}/V_T\right) - 1\right] - I_S\left[\exp\left(V_{bc}/V_T\right) - 1\right] \tag{5.2}$$

$$I_c = I_S\left[\exp\left(V_{be}/V_T\right) - 1\right] - \frac{I_S}{\alpha_R}\left[\exp\left(V_{bc}/V_T\right) - 1\right] \tag{5.3}$$

$$I_b = \frac{I_S}{\beta_F}\left[\exp\left(V_{be}/V_T\right) - 1\right] + \frac{I_S}{\beta_R}\left[\exp\left(V_{bc}/V_T\right) - 1\right] \tag{5.4}$$

Device parameters for this simulation are $V_T = 26\,\text{mV}$, $\beta_F = 200$, $\beta_R = 0.1$, $\alpha_R = \beta_R/(1+\beta_R)$, $I_s = 6.734 \times 10^{-15}\text{A}$. The reader is referred to textbooks on electronic devices (Muller et al., 2002) for detailed interpretation of these parameters.

5.3.2 K-Method formulation

Using the generic description of Fig. 5.8 for I-V characteristics, we find the K-method matrices for this circuit. Again defining the state to be the voltages across each of the three capacitors, $\mathbf{x} = \begin{bmatrix} V_{Ci} & V_{bc} & V_{Co} \end{bmatrix}^T$ with the polarity of the voltages indicated by + on the capacitors in Fig. 5.7, we can use KCL to find equations at each of the nodes, and solve for $\dot{\mathbf{x}}$. The inputs are $\mathbf{u} = \begin{bmatrix} V_i & V_{CC} \end{bmatrix}^T$, the input voltage and the supply rail. The nonlinearity is given by

$$\mathbf{i} = \begin{bmatrix} I_b(V_{be}, V_{bc}) & I_c(V_{be}, V_{bc}) \end{bmatrix}^T,$$

the currents at the base and collector terminals of the BJT, and requires an input $\mathbf{v} = \begin{bmatrix} V_{be} & V_{bc} \end{bmatrix}^T$. The output is found from

$$V_o = V_i - V_{Ci} - V_{bc} - V_{Co}.$$

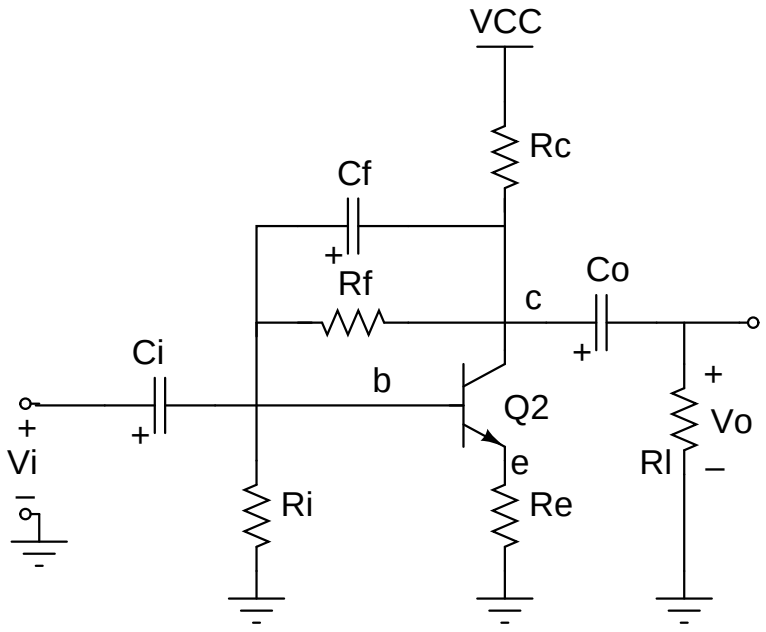


Figure 5.7: Schematic of the common-emitter amplifier with feedback.

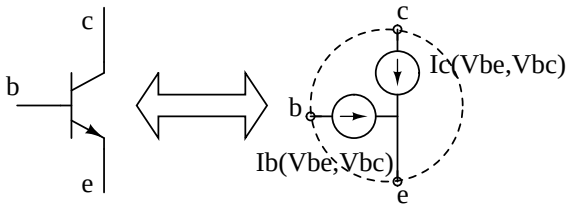


Figure 5.8: Generic BJT device model.

The K-method matrices then give the appropriate linear combinations of these variables using conductance $G_x = 1/R_x$ in place of the corresponding resistance:

$$\mathbf{A} = \begin{bmatrix} -\frac{G_c+G_l+G_i}{C_i} & -\frac{G_c+G_l}{C_i} & -\frac{G_l}{C_i} \\ -\frac{G_c+G_l}{C_f} & -\frac{G_c+G_l+G_f}{C_f} & -\frac{G_l}{C_f} \\ -\frac{G_l}{C_o} & -\frac{G_l}{C_o} & -\frac{G_l}{C_o} \end{bmatrix},$$

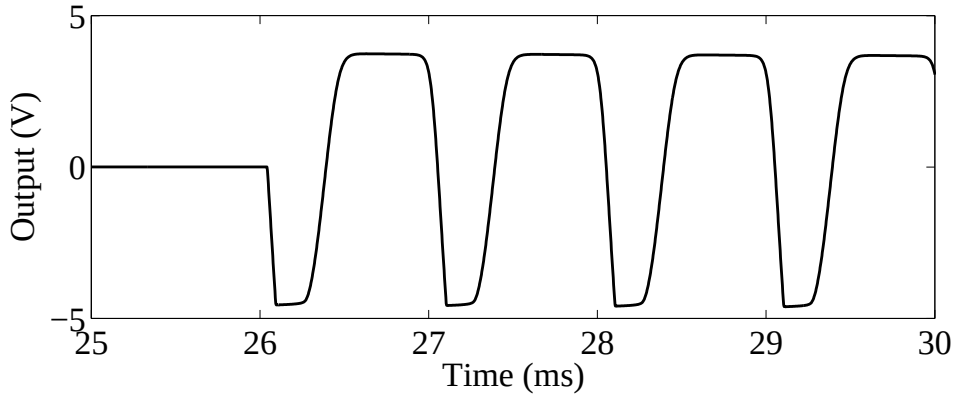


Figure 5.9: Output of the common-emitter amplifier for sine input of 0.2 V, 1 kHz.

$$\mathbf{B} = \begin{bmatrix} \frac{G_c+G_l+G_i}{C_i} & -\frac{G_c}{C_i} \\ \frac{G_c+G_l}{C_f} & -\frac{G_c}{C_f} \\ \frac{G_l}{C_o} & 0 \end{bmatrix}, \qquad \mathbf{D} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 1/C_i & 1/C_i \\ 0 & 1/C_f \\ 0 & 0 \end{bmatrix}, \qquad \mathbf{E} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},$$

$$\mathbf{F} = \begin{bmatrix} -R_e & -R_e \\ 0 & 0 \end{bmatrix}.$$

The formulation given admits a generic device model. While the Ebers-Moll model for a BJT is used here specifically, it is a simple model that does not account for the many nonidealities of real devices. In practice, the distortion performance of the circuit is highly sensitive to the accuracy of the device models, which usually represent some simplification of reality. This formulation allows for the use of tabulated device models obtained experimentally for greatest accuracy.

5.3.3 Simulation results

The BJT amplifier was simulated at a sampling rate of $8\times$ the audio rate 48000 Hz. The results are shown in Fig. 5.9. Note the asymmetry of the duty cycle of the output given a sine wave input. This is due to the asymmetry in the nonlinearity: one polarity clips at a lower level than the other. This causes an offset at DC, which is being filtered out by the

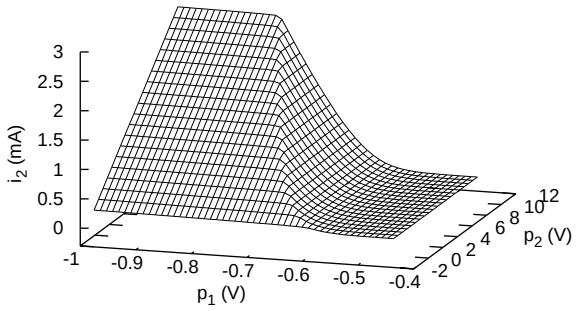


Figure 5.10: BJT K-method nonlinearity $i_2 = f(\mathbf{g}(\mathbf{p}))$

DC blocking capacitor at the output, causing the the bias of the output waveform to shift downward after the initial transient at 26 ms (not shown). A slowly shifting bias would affect the distortion of subsequent nonlinear stages.

The memoryless nonlinearity of the K-method filter giving the collector current as a function of the two parameters is plotted in Fig. 5.10.

5.4 Common-cathode triode amplifier with supply bypass

In guitar circuits, the ubiquitous common-cathode triode amplifier stage (Fig. 5.11) provides preamplification gain. Several of these stages can be cascaded for a high-gain amplifier. This circuit is essentially the same configuration as a BJT common-emitter amplifier. The grid resistor R_g and parasitic Miller capacitance C_f are shown explicitly in this simulation circuit. The cathode resistor R_k determines the operating bias point for the circuit. Often a bypass capacitor C_k is placed across the cathode resistor to counteract the effects of gain degeneration caused by the resistor, and gives a bandpass gain.

For this simulation, the circuit design used is $R_g = 70\text{k}\Omega$, $R_k = 1500\Omega$, $R_p = 100\text{k}\Omega$, $R_i = 1\text{M}\Omega$, $C_i = 0.047\mu\text{F}$, $C_f = 2.5p\text{F}$, $C_k = 25\mu\text{F}$.

5.4.1 Triode device model

The triode differs slightly from the BJT in the device model. While the BJT is controlled by the voltages across the base-emitter, and base-collector ports, owing to different operating principles, the triode is controlled by the voltages across the gate-cathode and cathode-anode ports. The triode device model is shown in Fig. 5.12.

The classic Child-Langmuir triode equation for the plate current (Spangenberg, 1948) is used here as a proof of concept:

$$I_p = K\left(E_d\left(\frac{1+\text{sign}(E_d)}{2}\right)\right)^{3/2}, \quad \text{where}$$

$$E_d = \mu V_{gk} + V_{pk},$$

and grid current $I_g = 0$. For the 12AX7 triode in this simulation, $\mu = 83.5$, $K = 1.73 \times 10^{-6}\text{A}/\text{V}^{3/2}$ (Leach, 1995).

The Child-Langmuir model allows the plate-cathode voltage to become negative while plate-cathode current is positive when the grid voltage is sufficiently high. This unphysical behavior demonstrates the inaccuracy of the model in a common region of operation for guitar distortion.

The Child-Langmuir equation is admittedly a poor model for simulation; however, the K-method formulation admits a general two-port description of the triode, so any of the multitude of triode models developed for circuit simulation in SPICE can be ported to this method. In particular, this formulation accounts for the effects of grid conduction (not used with this model), which is claimed to be sonically significant (Maillet, 1998).

5.4.2 K-method formulation

While a similar circuit was simulated using the wave digital formulation (Karjalainen and Pakarinen, 2006), the two-port nonlinear device does not yet readily admit a wave digital representation, and ad hoc means were necessary to generate a WDF. Alternatively, the K-method allows direct simulation of the common-cathode circuit in Fig. 5.11.

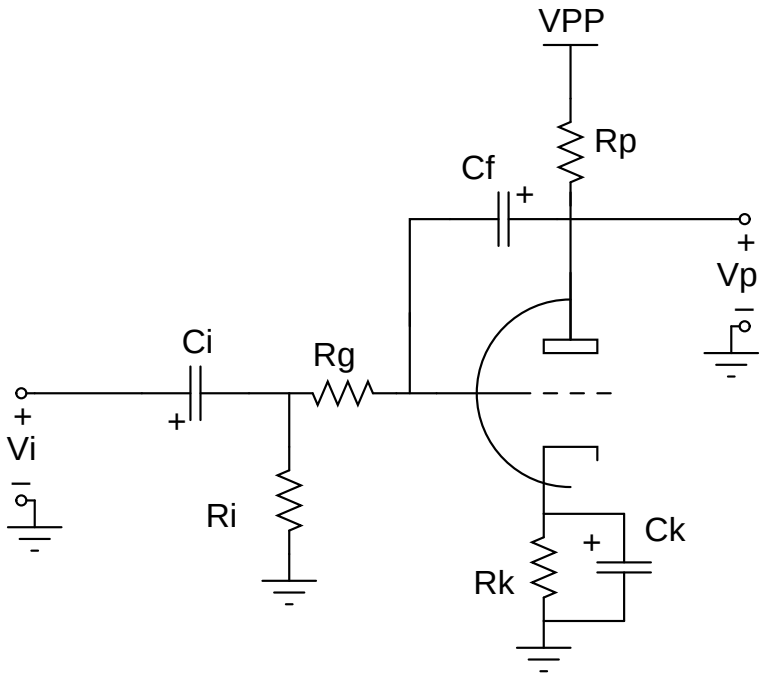


Figure 5.11: Schematic of the common-cathode triode amplifier.

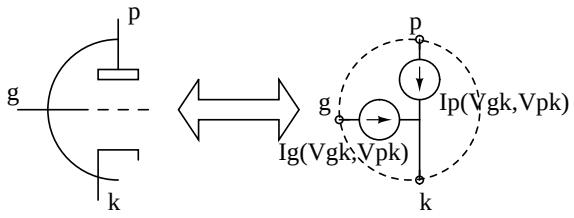


Figure 5.12: Generic triode device model.

The state vector is the voltages across each of the capacitors $\mathbf{x} = \begin{bmatrix} V_{Ci} & V_{Cf} & V_{Ck} \end{bmatrix}^T$, with the polarity of the voltages indicated by + on the capacitors in Fig. 5.11. The inputs are $\mathbf{u} = \begin{bmatrix} V_i & V_{PP} \end{bmatrix}^T$, the input voltage and the supply rail. The nonlinearity is given by

$$\mathbf{i} = \begin{bmatrix} I_g(V_{gk}, V_{pk}) & I_p(V_{gk}, V_{pk}) \end{bmatrix}^T,$$

the currents through the grid and plate terminals, and requires an input $\mathbf{v} = \begin{bmatrix} V_{gk} & V_{pk} \end{bmatrix}^T$, the voltages across the grid-cathode, and plate-cathode ports. The K-method matrices, using conductance $G_x = 1/R_x$ in place of the corresponding resistance, are then

$$\mathbf{A} = \begin{bmatrix} \frac{-\left(\left(G_i+G_g\right)G_p+G_iG_g\right)}{C_i(G_g+G_p)} & \frac{G_gG_p}{C_i(G_g+G_p)} & 0 \\ \frac{G_gG_p}{C_f(G_g+G_p)} & \frac{-G_gG_p}{C_f(G_g+G_p)} & 0 \\ 0 & 0 & \frac{-G_k}{C_k} \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} \frac{\left(\left(G_i+G_g\right)G_p+G_iG_g\right)}{C_i(G_g+G_p)} & \frac{-G_gG_p}{C_i(G_g+G_p)} \\ \frac{-G_gG_p}{C_f(G_g+G_p)} & \frac{G_gG_p}{C_f(G_g+G_p)} \\ 0 & 0 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} \frac{G_g}{C_i(G_g+G_p)} & \frac{G_g}{C_i(G_g+G_p)} \\ \frac{G_p}{C_f(G_g+G_p)} & \frac{-G_g}{C_f(G_g+G_p)} \\ \frac{1}{C_k} & \frac{1}{C_k} \end{bmatrix},$$

$$\mathbf{D} = \begin{bmatrix} \frac{-G_g}{G_p+G_g} & \frac{-G_p}{G_p+G_g} & -1 \\ \frac{-G_g}{G_p+G_g} & \frac{G_g}{G_p+G_g} & -1 \end{bmatrix},$$

$$\mathbf{E} = \begin{bmatrix} \frac{G_g}{G_p+G_g} & \frac{G_p}{G_p+G_g} \\ \frac{G_g}{G_p+G_g} & \frac{G_p}{G_p+G_g} \end{bmatrix}, \qquad \mathbf{F} = \begin{bmatrix} \frac{-1}{G_p+G_g} & \frac{-1}{G_p+G_g} \\ \frac{-1}{G_p+G_g} & \frac{-1}{G_p+G_g} \end{bmatrix}.$$

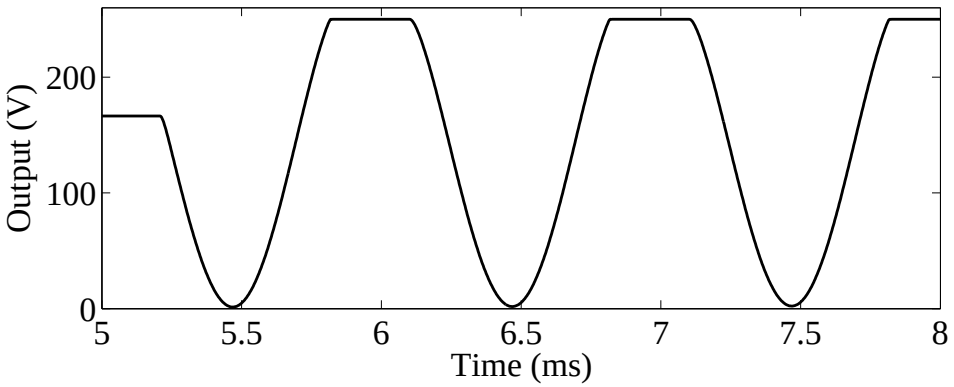


Figure 5.13: Plate voltage of common-cathode amplifier for sine input of 2.8 V, 1000 Hz.

The output is taken to be the plate voltage and can be found by $V_{pk} + V_k$ during simulation. This output contains a bias voltage and needs to be high-pass filtered for use in an audio plugin.

5.4.3 Simulation results

The tube preamp was simulated using the Child-Langmuir triode model at a sampling rate of $8\times$ the audio rate 48000 Hz. The plate voltage for an input of 2.8 V, 1000 Hz, is plotted in Fig. 5.13. Notice that this device model has an unrealistically sharp cutoff, causing the truncated tops of the waveforms in the figure.

The memoryless nonlinearity of the K-method filter giving the plate current as a function of the two parameters is plotted in Fig. 5.14.

5.5 DK-method examples

This section presents two example applications of the DK-method to clarify its usage and implications.

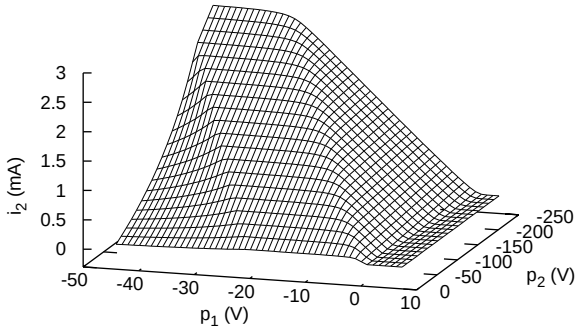


Figure 5.14: Common-cathode triode amplifier K-method nonlinearity $i_2 = f(\mathbf{g}(\mathbf{p}))$.

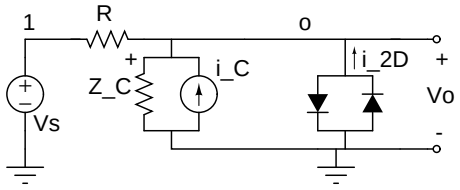


Figure 5.15: Companion circuit of single capacitor diode clipper.

5.5.1 Single capacitor diode clipper derivation

The following develops the equations for the nodal DK-method applied to the single-capacitor diode clipper of Sec. 3.4 using trapezoidal rule integration. The intent is to provide a concrete example illustrate the abstract development in Sec. 4.5.

Figure 5.15 depicts the companion model to the single capacitor diode clipper. The trapezoidal rule capacitor companion circuit replaces the single capacitor in Fig. 3.3. The parallel diodes connected with opposite polarities are represented collectively by (5.1).

Defining $G = 1/R$, $G_C = T/2C$, using all circuit variables at time n unless otherwise noted, and ignoring the ground node for notational simplicity, the nodal DK-method setup

for this circuit is

$$\begin{bmatrix} G & -G & 1 \\ -G & G+G_C & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_o \\ I_v \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} i_C[n-1] + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} V_s + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} i_{2D}. \quad (5.5)$$

Solving (5.5) for V_o yields

$$V_o = \frac{1}{G_C+G} i_C[n-1] + \frac{G}{G_C+G} V_s + \frac{1}{G_C+G} i_{2D}. \quad (5.6)$$

Using (5.6) in (4.37) derives a state update equation.

$$\begin{aligned} i_C &= \frac{2G_C}{G_C+G} i_C[n-1] + \frac{2G_C G}{G_C+G} V_s + \frac{2G_C}{G_C+G} i_{2D} - i_C[n-1] \\ &= \frac{G_C-G}{G_C+G} i_C[n-1] + \frac{2G_C G}{G_C+G} V_s + \frac{2G_C}{G_C+G} i_{2D}. \end{aligned} \quad (5.7)$$

In this circuit, V_o is also the controlling variable, so (5.6) also expresses (4.43) and can be rewritten

$$0 = p[n] + \frac{1}{G_C+G} i_{2D}(V_o) - V_o, \quad (5.8)$$

where V_o is an implicitly defined function

$$V_o = g(p) \quad (5.9)$$

of parameter

$$p[n] = \frac{1}{G_C+G} i_C[n-1] + \frac{G}{G_C+G} V_s[n]. \quad (5.10)$$

To compute the output given the input, first compute $p[n]$ by (5.10), then compute the nonlinear currents $i_D[n]$ using (5.9) and (5.1). Update the state using (5.7) and compute the output voltage using (5.6). To illustrate the explicit nature of this computation despite the use of an implicit integration method, the explicit nonlinear function (5.9) is shown in Fig. 5.16. Device parameters for this example are $R = 2.2\text{k}\Omega$, $C = 0.01\mu\text{F}$, $I_s = 2.52 \times 10^{-9}$ A, and $V_t = 45.3\,\text{mV}$.

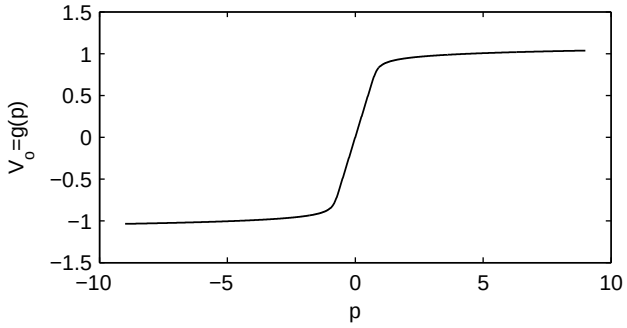


Figure 5.16: Explicit nonlinearity for single capacitor diode clipper.

5.5.2 Common Emitter BJT amplifier

This section examines the results of applying the DK-method using Backward Euler integration on the common emitter BJT amplifier of Sec. 5.3.

The netlist parser was implemented in Python, and the simulation code/nonlinear solver was implemented in C++. In particular, nonlinear device models included in the system are silicon pn diodes, Ebers-Moll models for bipolar junction transistors, and Koren models for triodes and tetrodes (Koren, 1996). The code for this project is available at http://ccrma.stanford.edu/∼dtyeh/nkmethod10.

The Backward Euler discretization was applied at $8\times$ oversampling of $f_s = 48\,\text{kHz}$. To provide a highly accurate reference, a $72\times$ oversampled algorithm was also evaluated. For both cases RELTOL $= 10^{-4}$, MAXRES $= 10^{-3}$ using the homotopy solver. The $8\times$ oversampled BE algorithm was used to generate linearly interpolated tables using the homotopy solver. To explore the effects of approximation error, we tested a coarsely sampled table with 25 steps for each dimension. The maximum of the gradient of table for nonlinearities is approximately 0.7 mV, which is the worst case error of the table. In comparison, we tested a finely sampled table with 1000 steps for each dimension. The maximum of the gradient of this table is approximately 19 μV. These methods were all compared to a simulation using the same device model parameters in LTspice IV (Linear Technology, 2007), using Trapezoidal Rule integration, default simulation parameters, a maximum time step of 2.6 μs, and saving the output to a wave file at 384 kHz, which applies linear interpolation to

fit SPICE's output to a fixed sampling grid. Comparison to SPICE verifies the correctness of our implementation.

5.5.2.1 Two-tone sinusoidal test

To explore the spectral response, we tested the methods with a two-tone excitation

$$V_i = A_1 \sin(2\pi f_1 t) + A_2 \sin(2\pi f_2 t) \tag{5.11}$$

with $f_1 = 110\,\text{Hz}$, $f_2 = 165\,\text{Hz}$. Ten milliseconds of silence precede the signal to allow the circuit to settle and to observe the transient response.

To test the response to a large signal excitation, we applied $A_1 = A_2 = 1\,\text{V}$. Figure 5.17 shows the time responses V_o of the five methods tested. The responses are very similar in the time domain, with the notable exception being the coarse table, which shows some deviations in the lower parts of the waveform.

To observe the spectral accuracy of the methods, the FFT was applied to a Hann windowed, one-second excerpt of the response. Because the spectrum from 20 Hz–20 kHz is very dense, we only plot the peaks corresponding to the harmonics of 55 Hz, the fundamental of the output signal after modulation due to nonlinearity, in Fig. 5.18.

The profiles of the harmonic peaks are very similar, except for the coarse table. The sampling of the table becomes more important for smaller signals because the SNR due to the approximation error is smaller; therefore, we also tested the methods using $A_1 = A_2 = 0.1\,\text{V}$. The results are shown in Figs. 5.19 and 5.20. The error in the frequency domain due to sampling the table is on the order of the maximum of the gradient of the table, but with attenuation and frequency rolloff due to filtering effects in this circuit.

5.5.2.2 High-frequency single-tone sinusoid

To explore the effects of very large higher frequency signals, we input single-tone, high-frequency sinusoids at 1 V to the methods. Often very large-amplitude or high-frequency signals will cause convergence difficulties in the nonlinear solvers of these methods. These tests are an extreme case because realistic electric guitar signals will not have frequency

components with 1 V amplitude at such high frequencies due to natural rolloff of acoustic systems. However, these signals serve to examine the robustness of the methods.

Figures 5.21 and 5.22 show the response to 1 kHz, and Figs. 5.23 and 5.24 display the response to 11.111 kHz, a frequency chosen to spread aliases throughout the spectrum. The fixed step-size methods cause severe overshoot at high frequencies, which is absent in the SPICE version owing to its implementation of adaptive time step. The table methods perform similarly to the methods with solvers. Aliasing is the dominant problem for such a high-amplitude, high-frequency signal, but note that all methods correctly represent the energy of the fundamental.

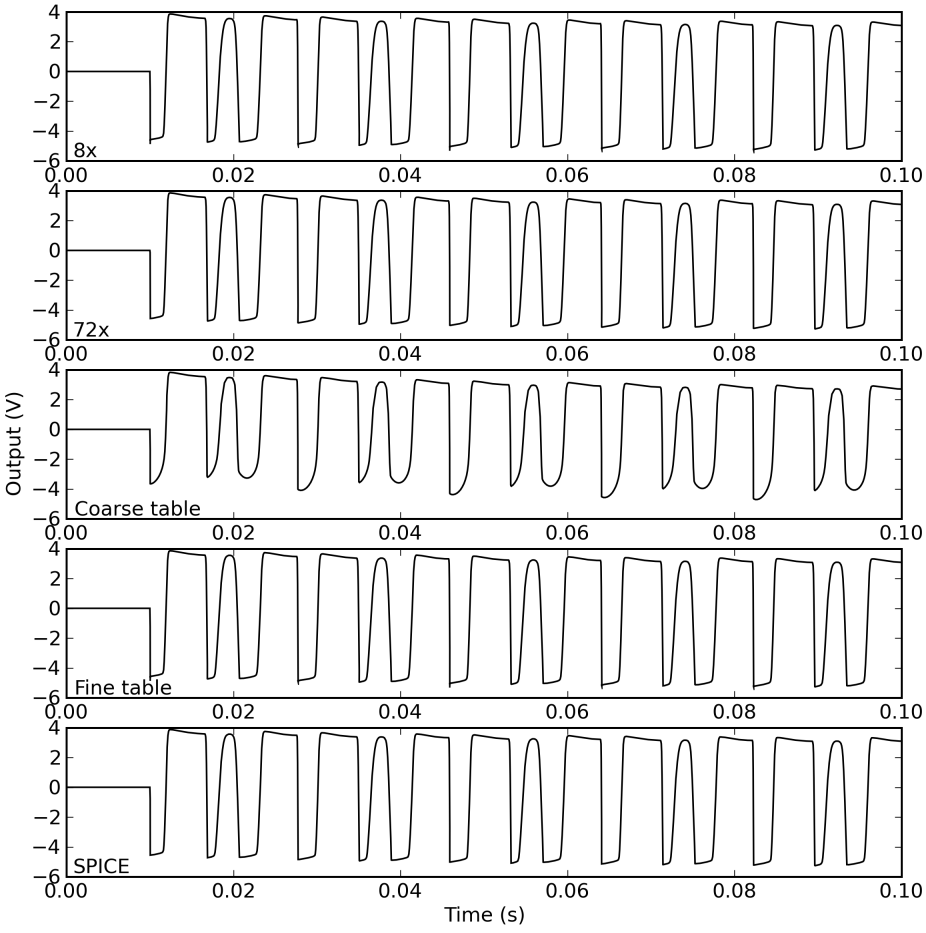


Figure 5.17: BJT amplifier response to 110 and 165 Hz sinusoids, each 1 V.

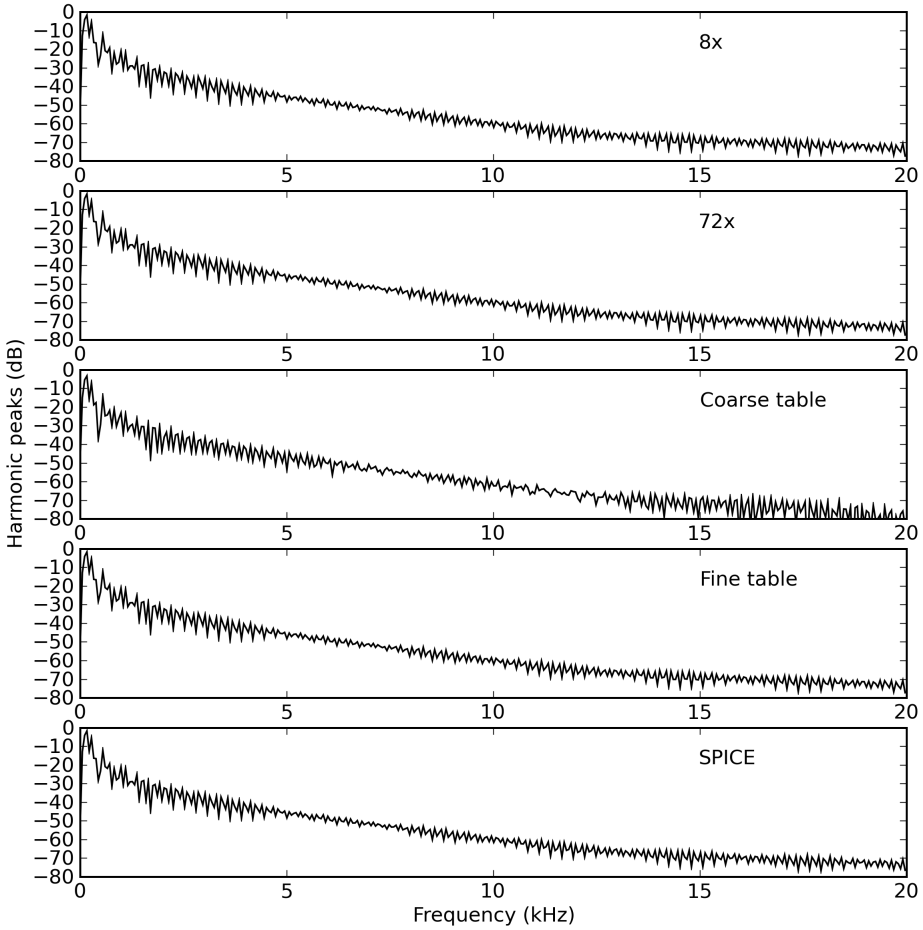


Figure 5.18: BJT amplifier response to 110 and 165 Hz sinusoids, each 1 V, harmonic peaks, $f_0 = 55$Hz.

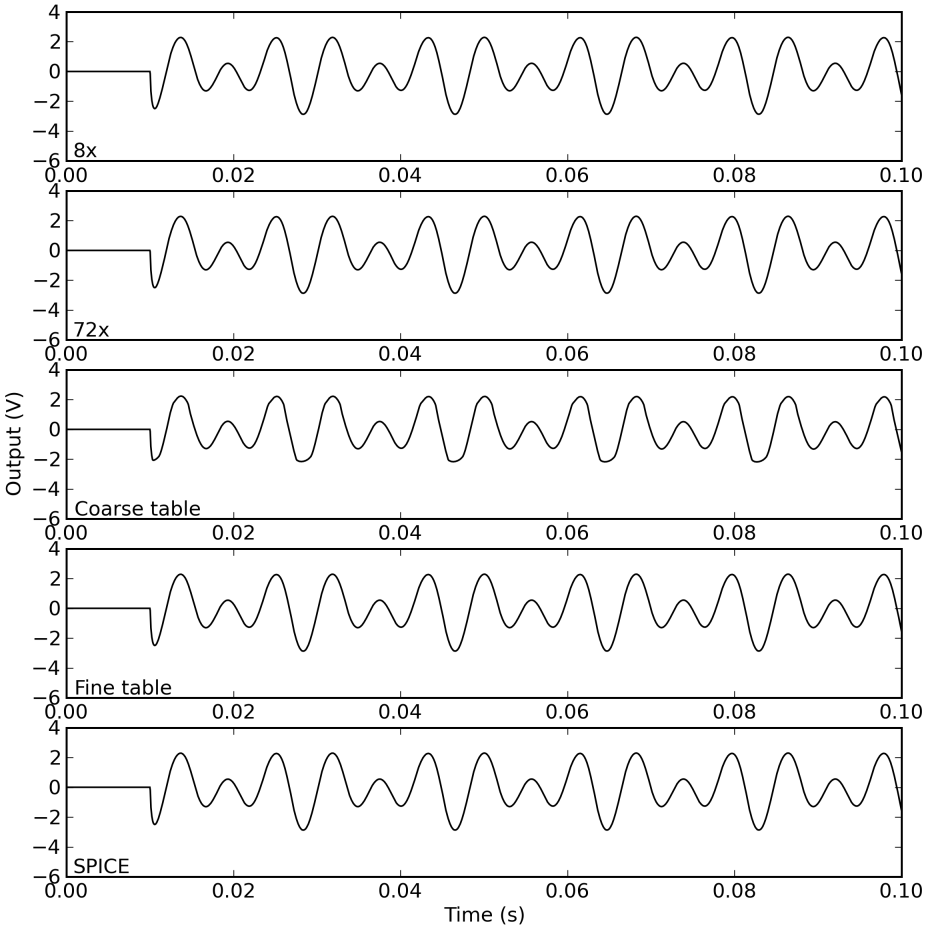


Figure 5.19: BJT amplifier response to 110 and 165 Hz sinusoids, each 0.1 V.

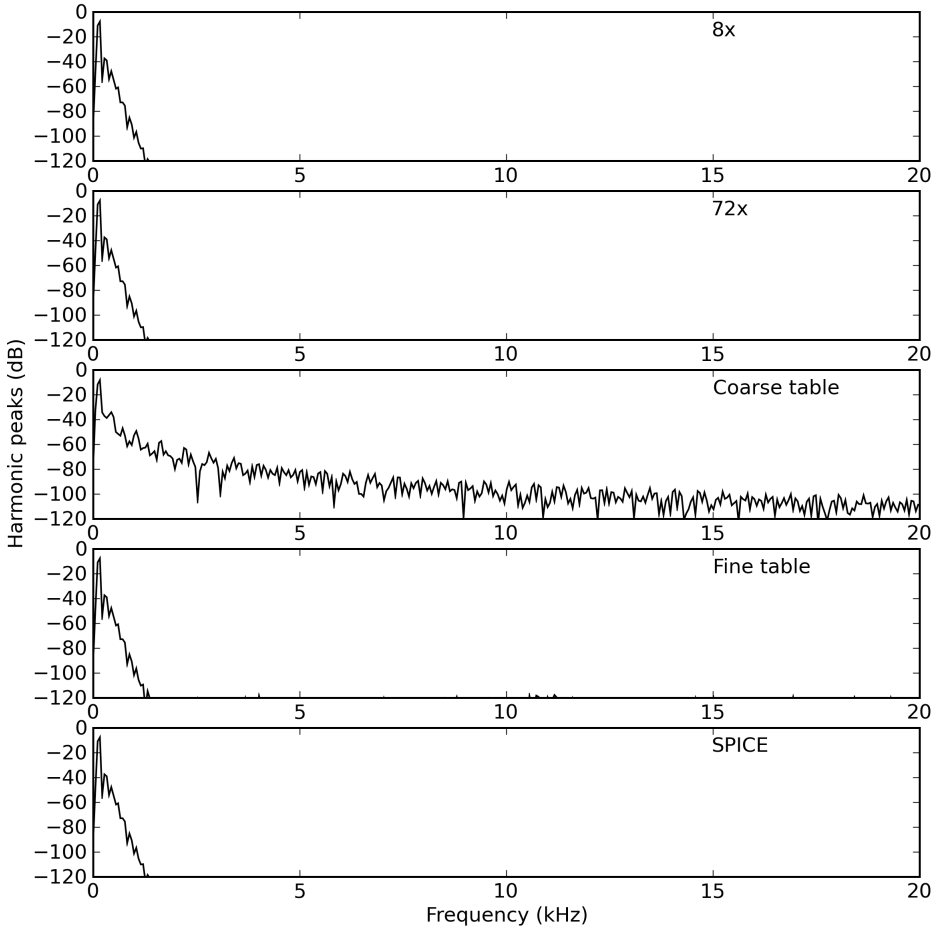


Figure 5.20: BJT amplifier response to 110 and 165 Hz sinusoids, each 0.1 V, harmonic peaks, $f_0 = 55$Hz.

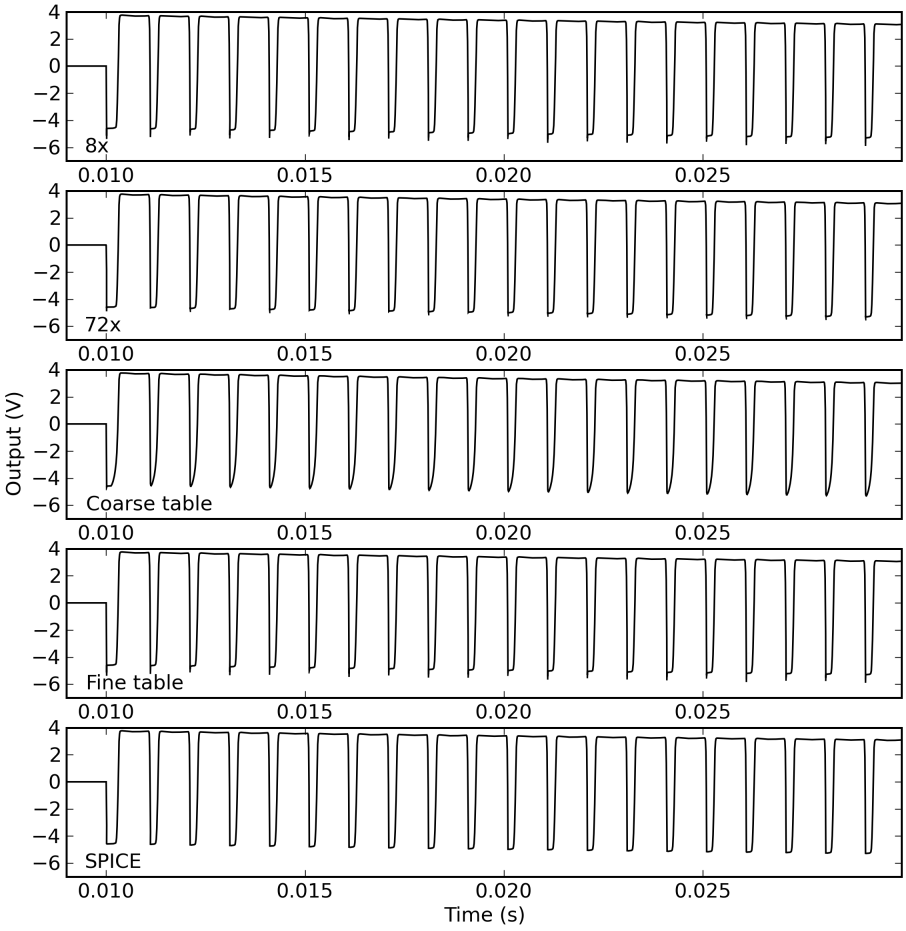


Figure 5.21: BJT amplifier response to 1 kHz, 1 V input.

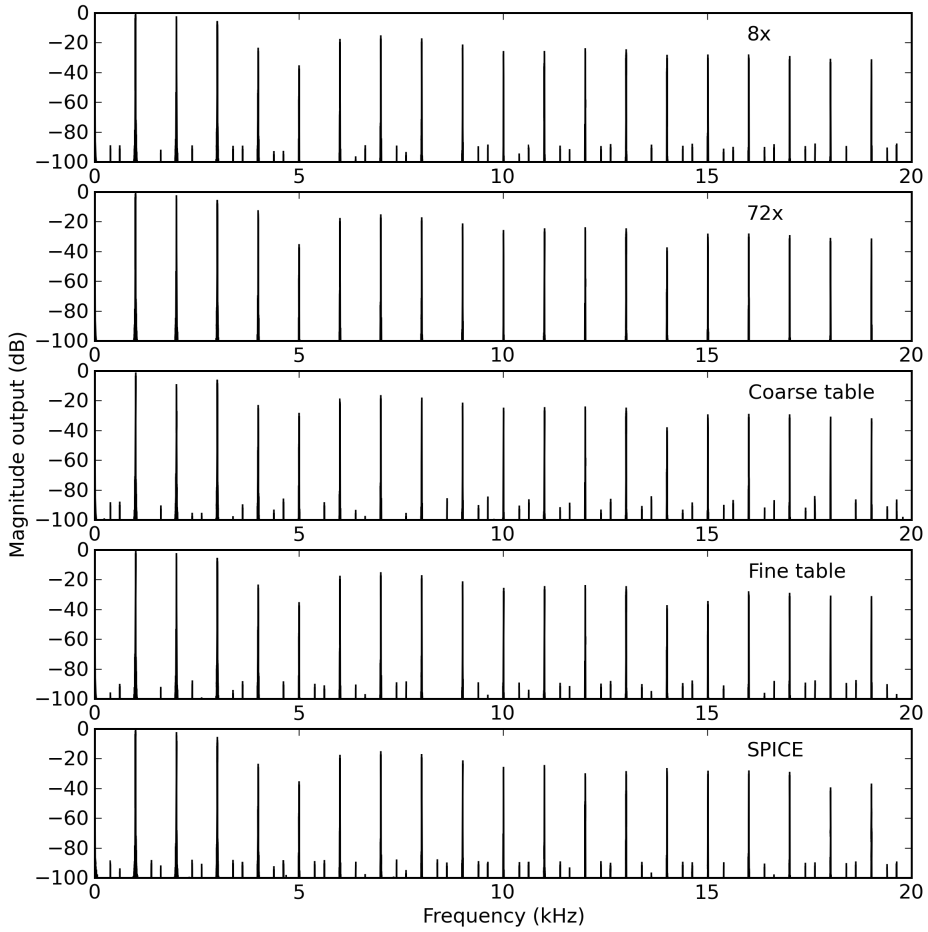


Figure 5.22: BJT amplifier response (magnitude spectrum) to 1 kHz, 1 V input.

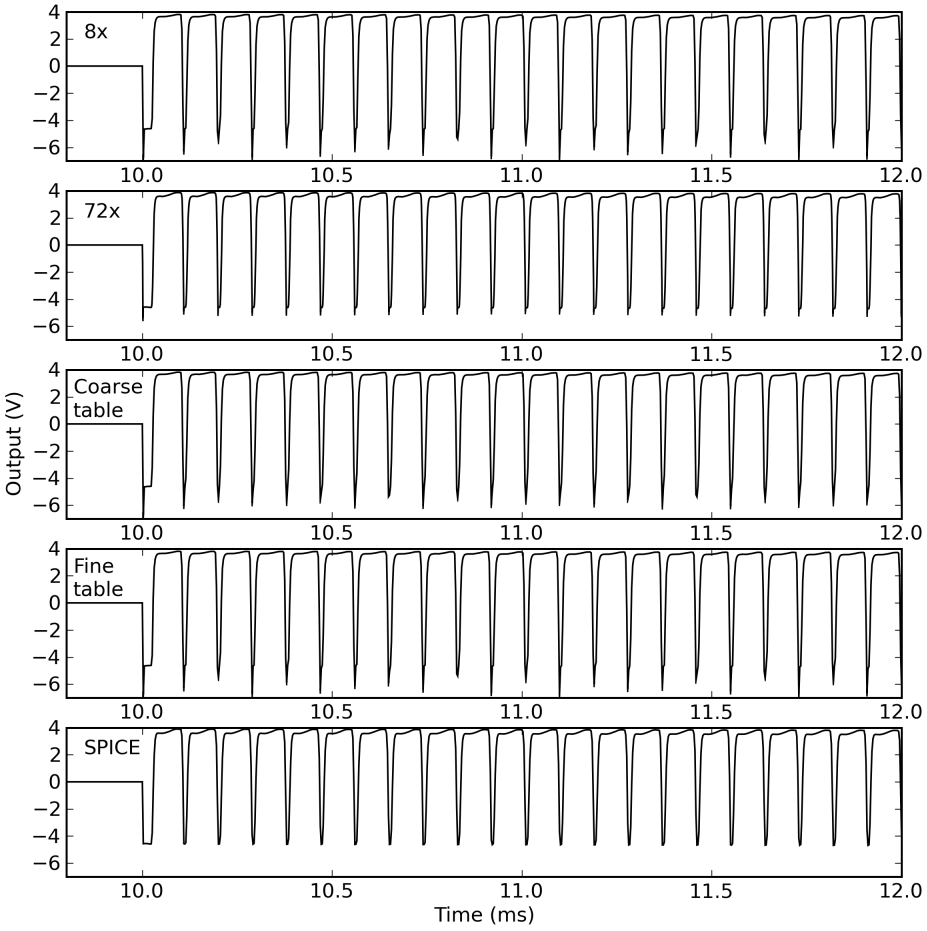


Figure 5.23: BJT amplifier response to 11 kHz, 1 V input.

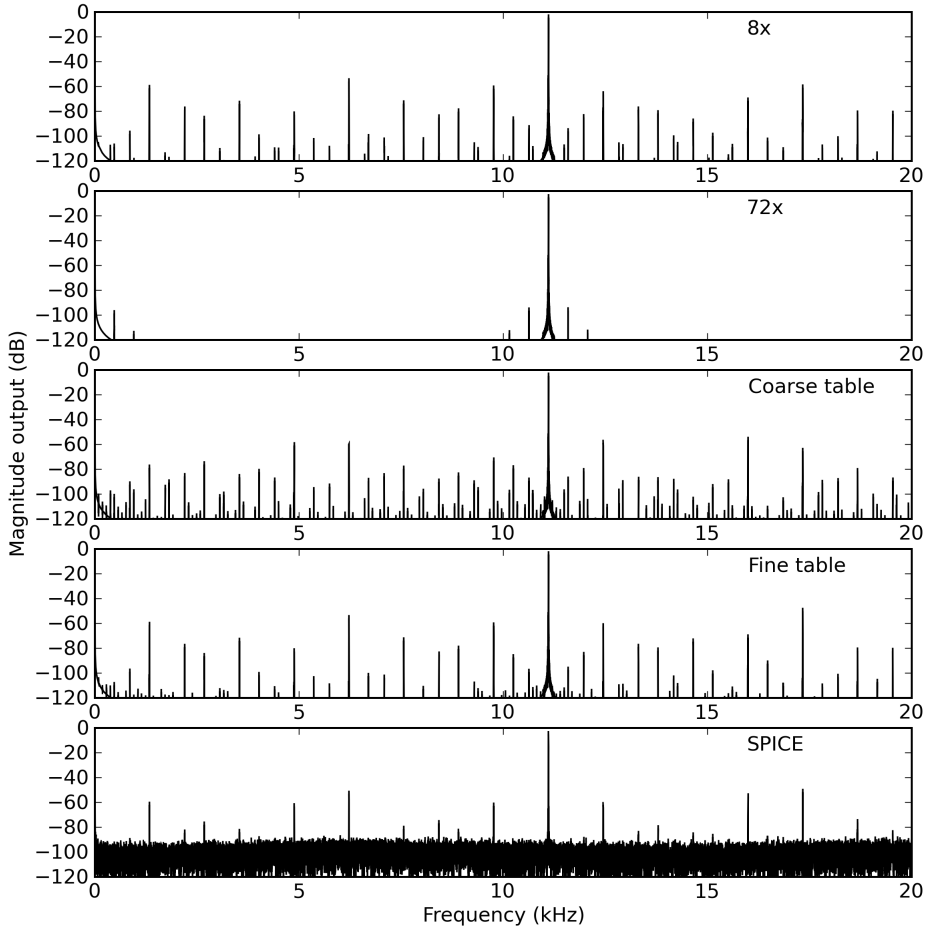


Figure 5.24: BJT amplifier response (magnitude spectrum) to 11 kHz, 1 V input.

5.6 Conclusions

The nonlinear methods developed for computational musical acoustics are readily applied to musical circuits simulation.

For the K-method, states should correspond to the natural state elements of the circuit, namely capacitors and inductors. Choosing the appropriate state variables facilitates derivation of the nonlinear state-space equations and aids interpretation of the resulting system.

For solving nonlinear systems, both the WDF and K-method are conceptually similar in the overall order of operations. Both first compute a linear combination of state and inputs – this is used as an input to a nonlinear function. Then to update the states they compute linear combinations of these variables with the outputs of the nonlinearity.

As seen from the results, the problem of aliasing dominates the choice of sampling rate. The problem remains even at computationally expensive sampling rates, although it may be perceptually acceptable. High input levels used for guitar distortion cause Newton's method to fail, requiring the use of homotopy. However, homotopy, though robust, is not suitable for real-time use because it dramatically increases the computational complexity. At high input levels with strongly saturating nonlinearities, the solvers require many iterations, making systems that use solvers unattractive for real-time use. Using tables trades off accuracy and flexibility for robustness and speed. Furthermore, it is more difficult to handle parameter changes in the circuit with a table.

Chapter 6

Conclusions

This work addresses the demand for high-quality musical distortion processing algorithms that mimic the sound of original analog equipment by proposing that detailed circuit analysis and simulation itself leads to an effective implementation.

These models are created by first inspecting the circuits to determine the signal path and to decompose the circuit into blocks that implement linear and nonlinear stages in the signal path. The linear stages are implemented by analyzing the circuits either symbolically or numerically and designing linear filters that well approximate the desired frequency response. The nonlinear stages are implemented by describing the circuit using a netlist and processing it with a system that derives parameters for a general nonlinear filter implementation. This is effectively a customized approach to circuit simulation that is appropriate for real-time audio signal processing.

The prevailing method of using parametric filters and nonlinearities in cascade requires laborious tuning by hand to determine the correct parameters. In contrast, as proposed here, modeling the signal path of circuits by a stagewise approach yields digital emulations that capture the salient features of the circuit on a first pass without the need for tweaking. Further tuning of the models and parameters can be done should an exact match be required. This method provides a well-defined procedure to deriving digital emulations of distortion circuits. Furthermore, whereas the parametric approach in essence samples the behavior of the circuit for particular operating conditions, a physically derived emulation better mimics the behavior over a broad range of conditions.

In the process of conducting this research, it has been found that oversampling by 2–8$\times$ dramatically improves performance of the algorithms in terms of reduced aliasing and ease of convergence.

For approaches that solve the ordinary differential equations of circuits, it has been found that except for special cases where the input signal can be guaranteed to fall within certain specifications, explicit numerical integration algorithms such as Forward Euler or explicit 4th order Runge-Kutta can lead to instability. Also, experience has shown that in general, the output signal of a circuit is implicitly computed from a set of nonlinear equations owing to feedback or delay free loops both within the topology of the circuit and in the model equations themselves. Because this requires iteration anyway, one should always first consider using A-stable, implicit numerical integration algorithms such as Backward Euler or Trapezoidal rule as a general approach to solving the circuit equations. Finally, the K-method concept demonstrates that these implicit integration algorithms can be solved before runtime, eliminating the cost disadvantage of an iterative approach.

The WDF is an efficient approach, having linear computational complexity when simulating multi-port systems, but it cannot handle multidimensional nonlinearities easily. Further study needs to be done to apply WDFs to more complicated nonlinear circuits.

Deriving the parameters for the K-method approach can often prove to be unwieldy and an automated procedure was developed to compute these parameters from a description of the circuit. One can envision using existing schematic capture utilities to generate the netlist from a graphical schematic of the circuit and drive the process of creating a real-time emulation of the circuit in software. Although symbolic software can be used to ease the derivation of the parameters in explicit analytical form, the resulting expressions are often very complicated due to the nature of circuit analysis. A direct numerical approach is thus recommended for reasons of efficiency and generality.

6.1 Contributions

1. This work provides an analysis of typical circuits found in guitar distortion for future reference (parametric tone stack, diode clipper, opamp filters, Distortion and Overdrive signal paths, single-ended single-stage amplifiers).

2. Various discretization methods were analyzed in terms of accuracy and suitability for audio processing.

3. This work provides rules for applying WDF and K-methods to simulating circuits, demonstrating the techniques via several examples.

4. The K-method was extended to allow automated derivation of parameters from a netlist description of the circuit (NK-method).

5. The K-method was derived for the discrete-time case to facilitate automated derivation of parameters (DK-method).

6. This work introduces iteration on the control voltages and the usage of homotopy to improve convergence.

6.2 Future work

6.2.1 Nonlinear modeling

As the K-method simulates nonlinear systems recursively, it may serve as an efficient alternative to the Volterra series for modeling and compensating nonlinear systems. Using a state estimator, one might be able to invert the system and be able to linearize nonlinearities with memory.

It would also be worthwhile to investigate system identification approaches to deriving parameters for a known K-method structure (matrix coefficients, nonlinear function) using input/output measurements of real nonlinear systems.

6.2.2 Function approximation

The dimension of the table lookup for the stored nonlinearity in K-method grows with the number of nonlinear devices in the circuit. A straightforward table lookup is thus impractical for circuits with more than two transistors or vacuum tubes. However, function approximation approaches such as neural networks or nonlinear regression may hold promise for efficiently providing means to implement these high-dimensional lookup functions.

6.2.3 Further algorithm development

Wave digital filters offer computational efficiency, robustness to coefficient quantization, and facilitate interfacing with wave variables, making them a worthwhile subject of study. Representation of multiport nonlinearities and filter synthesis for general active circuits is still an open problem.

For implementation purposes, further work can be done on the fixed point properties of the K-method and its sensitivity to coefficient roundoff error. Better theoretical understanding of the stability properties of the K-method can help to ensure robust runtime behavior.

6.2.4 Improved component modeling

Accurate simulation on a physical modeling basis requires electronic device equations that accurately model the nonlinearities. Device models for bipolar junction transistors were designed with circuit simulation in mind. This is not the case with currently available vacuum-tube models, which tend to result in unreliable simulations due to discontinuities in the model or poorly behaved regions in the curve fits. Vacuum-tube device models need to be improved before nonlinear computer simulation of vacuum-tube circuits can become realistic. Once accurate, numerically robust device models are available, they can be readily used with these two methods for solving nonlinear ordinary differential equations.

The result of precomputing the nonlinearity in K-method only applies when the nonlinearities themselves are inherently memoryless. Some circuit nonlinearities such as the DC hysteresis in a magnetic core do have memory and further work is still to be done to implement these in an efficient way.

6.2.5 Extension to nonlinear musical acoustics simulation

Furthermore, the K-method was borrowed from computational musical acoustics, where it was used to model nonlinear mechanical systems, and applied to circuits. This work presents an automated system to derive K-method parameters, which can be applied again

to mechanical equivalent circuits in future work. It would be interesting to include waveguides in the netlist description format and allow complete specifications of acoustic simulations as netlists. These specifications can then generate real-time code, facilitating the implementation of nonlinear simulations of musical acoustics.

6.3 Final thoughts

This work represents a first thorough investigation of implementing distortion algorithms by emulating circuits using a systematic and general approach. It is hoped that these results will inform and inspire future researchers or practitioners in virtual analog modeling.

Simulating the circuit is the most accurate way to reproduce the nuances of the distortion and to ensure that it behaves with appropriate complexity. Like physical modeling of musical instruments, modeling the circuit component-wise allows parametric behavior to be modeled correctly. Because audio circuits are typically small, it is conceivable that in the near future, full simulations can be done in real time. If successful, this could harness the existing skill of circuit designers and hobbyists, who can then experiment with their ideas and implement them on a real-time digital platform. Because component models are not limited by physical reality and availability, they can experiment with various parameters of the device electronics (tubes, transistors, etc.) and even invent fictional ones to discover if their conjectures about the nature of various phenomena in musical electronics are true. Continued development in real-time circuit simulation (e.g., “real-time SPICE”) of musical electronics has the potential to spark a paradigm shift in digital audio effects.

Appendix A

Derivations of DK-Method Elements

A.1 Discrete capacitors

A.1.1 Trapezoidal Rule

For discrete capacitors using trapezoidal rule integration,

$$\begin{aligned} I &= C\frac{\mathrm{d}V}{\mathrm{d}t} \\ \int I(t)\mathrm{d}t &= C\int \mathrm{d}V \\ T/2\,(I[n]+I[n-1]) &= C(V[n]-V[n-1]) \\ I[n] &= 2C/T\,V[n]-2C/T\,V[n-1]-I[n-1] \end{aligned}$$

Let $Y_C = 2C/T$, and let an equivalent current source represent the state

$$i_C[n] = Y_C V[n] + I[n] \tag{A.1}$$

This equation can be added to the KCL of corresponding rows in the MNA.

$$\begin{aligned} I[n] &= Y_C V[n] - (Y_C V[n-1] + I[n-1]) \\ &= Y_C V[n] - i_C[n-1] \end{aligned} \tag{A.2}$$

Combining (A.1) and (A.2) gives an expression for state update

$$\begin{aligned} i_C[n] &= Y_C V[n] + Y_C V[n] - i_C[n-1] \\ &= 2Y_C V[n] - i_C[n-1] \end{aligned} \tag{A.3}$$

A.1.2 Backward Euler

$$\begin{aligned} I[n]/C &= \left.\frac{\mathrm{d}V}{\mathrm{d}t}\right|_n \\ I[n]/C &= \left(V[n] - V[n-1]\right)/T \\ I[n] &= C/T\left(V[n] - V[n-1]\right) \\ I[n] &= C/T\,V[n] - C/T\,V[n-1] \end{aligned}$$

Let $Y_C = C/T$:

$$i_C[n] = Y_C V[n]$$

$$I[n] = Y_C V[n] - i_C[n-1]$$

State update unlike trapezoidal rule does not depend on $i_C[n-1]$

$$i_C[n] = Y_C V[n]$$

A.2 Discrete inductors

A.2.1 Trapezoidal Rule

For discrete mutual inductors using trapezoidal rule integration, flux Φ,

$$\Phi_1[n] = L_1 I_1[n] + M I_2[n] \tag{A.4}$$

$$\begin{aligned}\Phi_1[n] &= T/2V_1[n] + T/2V_1[n-1] + \Phi_1[n-1] \\ &= T/2V_1[n] + \text{state}_1[n-1] \end{aligned} \tag{A.5}$$

and likewise for inductor 2.

The following derives the state update equation for inductors

$$\begin{aligned}\text{state}_L[n] &= T/2V_L[n] + \Phi_L[n] \\ &= T/2V_L[n] + T/2V_L[n] + \text{state}_L[n-1] \\ &= TV_L[n] + \text{state}_L[n-1] \end{aligned} \tag{A.6}$$

Equations (A.4) and (A.5) can be used in the MNA matrices. Inductors add two unknown variables, Φ_L and V_L to the system.

By equating the right hand sides of (A.4) and (A.5) a single equation to be added to the MNA system can be derived. In this simplified implementation, each inductor adds 1 unknown, I_L, and 1 equation to the system.

$$L_1 I_1[n] + M_{12} I_2[n] + ... = T/2\, V_1[n] + \text{state}_1[n-1]$$

Rearranging to define discretized impedances,

$$\frac{2L_1}{T} I_1[n] + \frac{2M_{12}}{T} I_2[n] + ... = V_1[n] + 2/T\ \text{state}_1[n-1]$$

Letting $Z_{L1} = 2L_1/T$, $Z_{M12} = 2M_{12}/T$, the inductor state is then equivalently a voltage generator on the RHS,

$$v_L = 2/T\ \text{state}_L$$

The state update equation in terms of v_L becomes

$$v_L[n] = 2V_L[n] + v_L[n-1] \tag{A.7}$$

Finally,

$$Z_{L1}I_1[n] + Z_{M12}I_2[n] + ... = V_1[n] + v_{L1}[n-1] \tag{A.8}$$

Equation (A.8) defines a single equation that is added to the MNA formulation for each inductor.

A.2.2 Backward Euler

For discrete mutual inductors using Backward Euler integration,

$$\Phi_1[n] = L_1 I_1[n] + M I_2[n] \tag{A.9}$$

$$\Phi_1[n] = T\, V_1[n] + \Phi_1[n-1] \tag{A.10}$$

and likewise for inductor 2. Unlike trapezoidal rule, the BE formula is the state update.

Equations (A.9) and (A.10) can be used in the MNA matrices. Inductors add two unknown variables, Φ_L and V_L to the system and equation

$$L_1 I_1[n] + M_{12} I_2[n] + ... = T\, V_1[n] + \Phi_1[n-1]$$

Rearranging to define discretized impedances,

$$\frac{L_1}{T} I_1[n] + \frac{M_{12}}{T} I_2[n] + ... = V_1[n] + 1/T\, \Phi_1[n-1]$$

Letting $Z_{L1} = L_1/T$, $Z_{M12} = M_{12}/T$, the inductor state is then equivalently a voltage generator on the RHS,

$$v_L = 1/T\, \Phi_L$$

The state update equation in terms of v_L becomes

$$v_L[n] = V_L[n] + v_L[n-1] \tag{A.11}$$

Finally,

$$Z_{L1}I_1[n] + Z_{M12}I_2[n] + ... = V_1[n] + v_{L1}[n-1] \tag{A.12}$$

Equation (A.12) defines a single equation that is added to the MNA formulation for each inductor.

Bibliography

J. Abel. Private comm. Private comm., 2006.

J. S. Abel and D. P. Berners. A technique for nonlinear system measurement. In *Proceedings of the 121st Audio Engineering Society Convention*, San Francisco, CA, Oct. 2006.

D. Arfib. Digital Synthesis of Complex Spectra by Means of Multiplication of Nonlinear Distorted Sine Waves. *Journal of the Audio Engineering Society*, 27(10):757–768, 1979.

F. Avanzini and D. Rocchesso. Efficiency, accuracy, and stability issues in discrete time simulations of single reed instruments. *Journal of the Acoustical Society of America*, 111(5):2293–2301, May 2002.

F. Avanzini, F. Fontana, and D. Rocchesso. Efficient computation of nonlinear filter networks with delay-free loops and applications to physically-based sound models. In *Proceedings of the 4th International Workshop on Multidimensional Systems, (NDS 2005)*, pages 110–115, Wuppertal, Germany, July 2005.

D. P. Berners and J. S. Abel. Discrete-time shelf filter design for analog modeling. In *Proceedings of the 115th Audio Engineering Society Convention*, New York, Oct. 2003.

G. Borin, G. De Poli, and D. Rocchesso. Elimination of delay-free loops in discrete-time models of nonlinear acoustic systems. *IEEE Transactions on Speech and Audio Processing*, 8(5):597–605, Sept. 2000.

M. Bosi and R. Goldberg. *Introduction to Digital Audio Coding and Standards*. Klewer Academic Publishers, 2003.

S. Boyd and L. O. Chua. Fading Memory and the Problem of Approximating Nonlinear Operators with Volterra Series. *IEEE Transactions on Circuits and Systems*, 32(11): 1150 – 1161, Nov. 1985.

G. R. Boyle, D. O. Pederson, B. M. Cohn, and J. E. Solomon. Macromodeling of integrated circuit operational amplifiers. *IEEE Journal of Solid-State Circuits*, 9:353–364, Dec 1974.

J. C. Brown. Calculation of a constant-Q spectral transform. *Journal of the Acoustical Society of America*, 89:425–434, 1991. Matlab code available at `http://web.media.mit.edu/~brown/cqtrans.htm`.

J. C. Butcher. *The Numerical Analysis of Ordinary Differential Equations*. Wiley, Hoboken, NJ, 1987.

L. O. Chua. *Computer-Aided Analysis of Electronic Circuits*. Prentice Hall, Englewood Cliffs, 1975.

M. Civolani and F. Fontana. A nonlinear digital model of the EMS VCS3 voltage-controlled filter. In *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx-08)*, pages 35–42, Espoo, Finland, Sept. 1–4, 2008.

D. V. Curtis, K. L. Chapman, C. C. Adams, and Fender Musical Instruments. Simulated tone stack for electric guitar. United States Patent 6222110, 2001.

G. De Sanctis, A. Sarti, and S. Tubaro. Automatic synthesis strategies for object-based dynamical physical models in musical acoustics. In *Proceedings of the 6th International Conference on Digital Audio Effects (DAFx-03)*, London, UK, Sept. 2003. URL `http://www.elec.qmul.ac.uk/dafx03/proceedings/pdfs/dafx39.pdf`.

M. Doidic and et al. Tube modeling programmable digital guitar amplification system. U.S. Patent 5789689, Aug. 16 1998.

Duncan Amps. Tone Stack Calculator. Internet, 2006. URL `http://www.duncanamps.com/tsc/`. Available online, accessed 24 Mar. 2006.

T. Felderhoff. A new wave description for nonlinear elements. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, volume 3, pages 221–224, Atlanta, USA, May 1996.

Fender Music Instruments Corp. 5F6-A Schematic. Internet, 1999. URL `http://www.ampwares.com/ffg/bassman_narrow.html`. Available online, accessed 24 Mar 2006.

P. Fernández-Cid, J. Quirós, and P. Aguilar. MWD: Multiband Waveshaping Distortion. In *Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99)*, Trondeim, Norway, 1999.

A. Fettweis. Wave digital filters: Theory and practice. *Proceedings of the IEEE*, 74(2):270 – 327, Feb. 1986.

F. Fontana and F. Avanzini. Computation of delay-free nonlinear digital filter networks: Application to chaotic circuits and intracellular signal transduction. *IEEE Transactions on Signal Processing*, 56(10):4703–4715, Oct. 2008.

F. Fontana, F. Avanzini, and D. Rocchesso. Computation of nonlinear filter networks containing delay-free paths. In *Proceedings of the 7th International Conference on Digital Audio Effects (DAFx-04)*, pages 113–118, Naples, Italy, Oct. 2004.

D. Fränken and K. Ochs. Synthesis and Design of Passive Runge-Kutta Methods. *AEÜ International Journal of Electronics and Communications*, 55(6):417–425, 2001.

D. Fränken and K. Ochs. Improving Wave Digital Simulation by Extrapolation Techniques. *AEÜ International Journal of Electronics and Communications*, 56(5):327–336, 2002.

D. Fränken, J. Ochs, and K. Ochs. Generation of Wave Digital Structures for Networks Containing Multiport Elements. *IEEE Transactions on Circuits and Systems—Part I: Regular Papers*, 52(3):586–596, 2005.

R. Gallien and K. Robertson. Programmable tone control filters for electric guitar. United States Patent Application US 2007/0168063 A1, 2007.

W. C. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1971.

T. Goetze. CAPS, the C* Audio Plugin Suite. Internet, 2005. URL `http://quitte.de/dsp/caps.html`. Available online, accessed 29 Mar 2006.

F. Gustafsson and et al. System and Method for Simulation of Non-Linear Audio Equipment. U.S. Patent Application Publication US2004/0258250 A1, 2004.

T. Hèlie. On the use of Volterra series for real-time simulations of weakly nonlinear analog audio device: application to the Moog ladder filter. In *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-06)*, pages 7–12, Montreal, Quebec, Canada, Sept. 18–20, 2006.

R. Hoffmann-Burchardi. Digital simulation of the diode ring modulator for musical appliations. In *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx-08)*, pages 165–168, Espoo, Finland, Sept. 1–4, 2008.

A. Huovilainen. Nonlinear digital implementation of the Moog ladder filter. In *Proceedings of the 7th International Conference on Digital Audio Effects (DAFx-04)*, pages 61–64, Naples, Italy, Oct. 5–8, 2004.

A. Huovilainen. Enhanced digital models for analog modulation effects. In *Proceedings of the 8th International Conference on Digital Audio Effects (DAFx-05)*, pages 155–160, Madrid, Spain, Sept. 20-22 2005.

M. Karjalainen and J. Pakarinen. Wave digital simulation of a vacuum-tube amplifier. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 5, pages 153–156, Toulouse, France, 2006.

M. Karjalainen, T. Mäki-Patola, A. Kanerva, and A. Huovilainen. Virtual air guitar. *Journal of the Audio Engineering Society*, 54(10):964–980, Oct. 2006.

R. G. Keen. The Technology of the Tube Screamer. Internet, 2007. URL `http://www.geofex.com/fxtech.htm`. Available online, accessed 22 Mar 2007.

M. J. Kemp. Audio Effects Synthesizer with or without Analyzer. U.S. Patent 7039194, 2006.

N. Koren. Improved vacuum tube models for SPICE simulations. *Glass Audio*, 8(5):18, 1996. URL `http://www.normankoren.com/Audio/Tubemodspice_article.html`. Available online, accessed 24 May 2009.

R. Kuehnel. *Circuit Analysis of a Legendary Tube Amplifier: The Fender Bassman 5F6-A*. Pentode Press, Seattle, 2nd edition, 2005. URL `http://www.pentodepress.com/contents.html`.

R. Kuroki and et al. Digital Audio Signal Processor with Harmonics Modification. U.S. Patent 5841875, 1998.

M. Le Brun. Digital Waveshaping Synthesis. *Journal of the Audio Engineering Society*, 27 (4):250–266, 1979.

W. M. Leach. SPICE Models for Vacuum-Tube Amplifiers. *Journal of the Audio Engineering Society*, 43(3):117–126, May 1995.

Linear Technology. SwitcherCAD III/LTspice Users Guide. Internet, 2007. URL `http://ltspice.linear.com/software/scad3.pdf`. Available online, accessed 3 Nov. 2007.

L. Ljung and T. L. Soderstrom. *Theory and Practice of Recursive Identification*. MIT Press, Cambridge, MA, 1983.

J.-C. Maillet. A generalized algebraic technique for modeling triodes. *Glass Audio*, 10(2): 2–9, 1998.

W. L. Martens and A. Marui. Psychophysical Calibration of Sharpness of Multiparameter Distortion Efffects Processing. In *Proceedings of the 114th Audio Engineering Society Convention*, New York, 2003. Preprint no. 5739.

W. J. McCalla. *Fundamentals of Computer-Aided Circuit Simulation*. Kluwer Academic Publishers, Boston, 1987.

K. Meerkötter and D. Fränken. Digital Realization of Connection Networks by Voltage-Wave Two-Port Adaptors. *AEÜ International Journal of Electronics and Communications*, 50(6):362–367, 1996.

K. Meerkötter and R. Scholz. Digital simulation of nonlinear circuits by wave digital filter principles. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, volume 1, pages 720–723, May 1989.

R. C. Melville, L. Trajković, S. C. Fang, and L. T. Watson. Artificial Parameter Homotopy Methods for the DC Operating Point Problem. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12(6):861 – 877, June 1993.

R. D. Middlebrook, V. Vorpérian, and J. Lindal. The N Extra Element Theorem. *IEEE Transactions on Circuits and Systems—Part I: Fundamental Theory and Applications*, 45(9):919–935, Sept. 1998.

S. K. Mitra. *Digital Signal Processing*. McGraw-Hill, Boston, 2nd edition, 2001.

S. Möller, M. Gromowski, and U. Zölzer. A measurement technique for highly nonlinear transfer functions. In *Proceedings of the 5th International Conference on Digital Audio Effects (DAFx-02)*, pages 203–206, Hamburg, Germany, Sep. 26-28 2002.

R. S. Muller, T. L. Kamins, and M. Chan. *Device Electronics for Integrated Circuits*. Wiley, Hoboken, NJ, 3 edition, 2002.

L. W. Nagel. SPICE2: A Computer Program to Simulate Semiconductor Circuits. Technical Report ERL Memo No. UCB/ERL-M75/520, University of California, Berkeley, May 1975.

L. W. Nagel and D. O. Pederson. SPICE (Simulation Program with Integrated Circuit Emphasis). Technical Report ERL Memo No. ERL-M382, University of California, Berkeley, Apr. 1973.

A. R. Newton and A. L. Sangiovanni-Vincentelli. Relaxation-Based Electrical Simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 3(4): 308–331, 1984.

L. S. H. Ngia. Separable nonlinear least-squares methods for efficient off-lineand on-line modeling of systems using Kautz and Laguerre filters. *IEEE Transactions on Circuits and Systems—Part II: Analog and Digital Signal Processing*, 48(6):562–579, June 2001.

S. J. Orfanidis. Digital Parametric Equalizer Design with Prescribed Nyquist-Frequency Gain. *Journal of the Audio Engineering Society*, 45(6):444–455, June 1997.

J. Pakarinen. *Modeling of Nonlinear and Time-Varying Phenomena in the Guitar*. PhD thesis, Helsinki University of Technology, 2008.

S. Petrausch and R. Rabenstein. Wave digital filters with multiple nonlinearities. In *Proceedings of the XII European Signal Processing Conference (EUSIPCO)*, volume 1, pages 77–80, Vienna, Austria, Sept. 2004.

A. Pittman. *The Tube Amp Book 4.1th Edition*. Groove Tubes, San Fernando, 2002.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, MA, second edition, 1992.

R. Rabenstein, S. Petrausch, A. Sarti, G. De Sanctis, C. Erkut, and M. Karjalainen. Block-Based Physical Modeling for Digital Sound Synthesis. *IEEE Signal Processing Magazine*, 24(2):42–54, Mar. 2007.

D. Rife and J. Vanderkooy. Transfer-function measurement with maximum-length sequences. *Journal of the Audio Engineering Society*, 37(6):419–444, June 1989.

Roland Corp. *Boss DS-1 Service Notes*, Dec. 26 1980.

F. Santagata, A. Sarti, and S. Tubaro. Non-Linear Digital Implementation of a Parametric Analog Tube Ground Cathode Amplifer. In *Proceedings of the 10th International Conference on Digital Audio Effects (DAFx-07)*, pages 169–172, Bordeaux, France, 2007.

M. Sapp, J. Becker, and C. Brouër. Simulation of Vacuum-Tube Amplifiers. *Journal of the Acoustical Society of America*, 105(2):1331, 1999.

A. Sarti and G. De Poli. Toward nonlinear wave digital filters. *IEEE Transactions on Signal Processing*, 47:1654–1668, June 1999.

A. Sarti and G. De Sanctis. Systematic methods for the implementation of nonlinear wave-digital structures. *IEEE Transactions on Circuits and Systems—Part I: Regular Papers*, 56(2):460–472, Feb. 2009.

J. Schattschneider and U. Zölzer. Discrete-Time Models for Nonlinear Audio Systems. In *Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99)*, Trondheim, Norway, 1999.

J. Schimmel. Using nonlinear amplifier simulation in dynamic range controllers. In *Proceedings of the 6th International Conference on Digital Audio Effects (DAFx-03)*, pages 203–206, London, UK, Sep. 8-11 2003.

J. Schimmel and J. Misurec. Characteristics of the Broken Line Approximation and its Use in Distortion Audio Effects. In *Proceedings of the 10th International Conference on Digital Audio Effects (DAFx-07)*, pages 161–164, Bordeaux France, 2007.

A. M. Schneider, J. T. Kaneshige, and F. D. Groutage. Higher Order s-to-z Mapping Functions and Their Application in Digitizing Continuous-Time Filters. *Proceedings of the IEEE*, 79(11):1661–1674, Nov. 1991.

L. F. Shampine. *Numerical Solution of Ordinary Differential Equations*. Chapman and Hall, New York, 1994.

L. F. Shampine and M. W. Reichelt. The MATLAB ODE Suite. *SIAM Journal on Scientific Computing*, 18:1–22, 1997.

J. O. Smith III. *Techniques for Digital Filter Design and System Identification with Application to the Violin*. PhD thesis, Elec. Eng Dept., Stanford University (CCRMA), June 1983. CCRMA Technical Report STAN-M-14,.

J. O. Smith III. *Physical Audio Signal Processing: Digital Waveguide Modeling of Musical Instruments and Audio Effects*. available on-line, June 2008. URL `http://ccrma.stanford.edu/~jos/pasp/`.

K. Spangenberg. *Vacuum Tubes*. McGraw-Hill, New York, 1st edition, 1948.

J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer-Verlag, New York, third edition, 2002.

C. R. Sullivan. Extending the Karplus-Strong algorithm to synthesize electric guitar timbres with distortion and feedback. *Computer Music Journal*, 14(3):26–37, 1990.

H. Thornburg. Antialiasing for Nonlinearities: Acoustic Modeling and Synthesis Applications. In *Proceedings of the 1999 International Computer Music Conference*, Beijing, China, 1999.

A. Ushida, Y. Yamagami, Y. Nishio, I. Kinouchi, and Y. Inoue. An Efficient Algorithm for Finding Multiple DC Solutions Based on the SPICE-Oriented Newton Homotopy Method. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(3):337 – 348, Mar. 2002.

V. Välimäki and A. Huovilainen. Oscillator and Filter Algorithms for Virtual Analog Synthesis. *Computer Music Journal*, 30(2):19–31, 2006.

A. Vladimirescu. *The Spice Book*. Wiley, New York, 1994.

C. Wan and A. M. Schneider. Extensions of the Weighted-Sample Method for Digitizing Continuous-Time Filters. *IEEE Transactions on Signal Processing*, 49(8):1627–1637, Aug. 2001.

J. K. White and A. L. Sangiovanni-Vincentelli. *Relaxation Techniques for the Simulation of VLSI CIrcuits*. Kluwer, Boston, MA, 1987.

D. T. Yeh and J. O. Smith. Discretization of the '59 Fender Bassman tone stack. In *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-06)*, pages 1–6, Montreal, Quebec, Canada, 2006. URL `http://www.dafx.ca/proceedings/papers/p_001.pdf`.

D. T. Yeh and J. O. Smith. Simulating guitar distortion circuits using wave digital and nonlinear state-space formulations. In *Proceedings of the 11th International Conference on Digital Audio Effects (DAFx-08)*, pages 19–26, Espoo, Finland, Sept. 1–4, 2008.

D. T. Yeh, J. Abel, and J. O. Smith. Simplified, physically-informed models of distortion and overdrive guitar effects pedals. In *Proceedings of the 10th International Conference on Digital Audio Effects (DAFx-07)*, pages 189–196, Bordeaux, France, Sept. 10–14, 2007a.

D. T. Yeh, J. Abel, and J. O. Smith. Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinary differential equations. In *Proceedings of the 10th International Conference on Digital Audio Effects (DAFx-07)*, pages 197–204, Bordeaux, France, Sept. 10–15, 2007b.

D. T. Yeh, J. S. Abel, A. Vladimirescu, and J. O. Smith. Numerical Methods for Simulation of Guitar Distortion Circuits. *Computer Music Journal*, 32(2):23–42, 2008.

U. Zölzer, editor. *DAFX – Digital Audio Effects*. J. Wiley & Sons, 2002.