Automated Physical Modeling of Nonlinear Audio Circuits For Real-Time Audio Effects – Part I: Theoretical Development

David T. Yeh, *Member, IEEE,* Jonathan S. Abel, *Member, IEEE,* and Julius O. Smith, III, *Member, IEEE*

***Abstract*—This paper presents a procedural approach to derive nonlinear filters from schematics of audio circuits for the purpose of digitally emulating analog musical effects circuits in real time. This work, the first in a two-part series, extends a well-known efficient nonlinear continuous-time state-space formulation for physical modeling of musical acoustics to real-time modeling of nonlinear circuits. Rules for applying the formulation are given, as well as a procedure to derive simulation parameters automatically from circuit netlists. Furthermore, a related nonlinear discrete-time state-space algorithm is proposed to alleviate problems in solving particular circuit configurations. These methods were devised to solve non-convergence problems in the simulation of strongly saturated, nonparametric guitar distortion circuits such as the saturating diode clipper, which which is presented as an example derivation. Experimental considerations and sonic performance on various other circuits will be presented in a subsequent paper.**

***Index Terms*—Virtual analog, physical modeling synthesis, guitar distortion, amplifier modeling, circuit simulation, nonlinear filters, K-Method, EDICS: AUD-SYST**

I. Introduction

THIS research attempts to model and simulate digitally highly nonlinear circuits used primarily for electric guitar effects. Such efforts aim to preserve the heritage of circuits whose components, such as vacuum tubes, or particular vintage transistors, are becoming increasingly rare. Although a plethora commercial products purport to replicate the nonlinear processing of these circuits using simplified filters and memoryless nonlinearities, many are unsatisfied with the perceptual quality of such digital models [1].

This work takes a more accurate, physical modeling approach. Circuit schematics and accurate device model equations can sufficiently model circuit behavior. Analyzing these schematics using physical principles such as Kirchhoff's Current Law (KCL) and Kirchhoff's Voltage Law (KVL), one can derive a system of nonlinear equations that accurately replicates a circuit's input-to-output relationship over time. By exploiting the progress of contemporary digital computing power, modeling vintage circuits based on archives of their circuit schematics and device characteristics can ensure that the unique sound of these circuits will be available for future generations of musicians.

The authors are with the Center for Computer Research in Music and Acoustics (CCRMA), Department of Music, Stanford University, Stanford, CA 94305-8180, USA. This work was partially supported by a National Science Foundation Graduate Fellowship.

This work describes an efficient and robust method to simulate circuits in real time, solving the nonlinear system of ordinary differential equations that characterize circuit dynamics. The system accepts circuits described in netlist form, and produces parameters for a nonlinear system solver that can generate code for real-time simulation. This system facilitates physically-based real-time emulation of nonlinear circuits used for musical effects.

Here, after surveying the literature, we present in detail the K-method, originally developed to model nonlinear lumped systems in musical acoustics. We then present the Nodal K-method, a procedure for applying the K-method to nonlinear circuits as found in musical distortion circuits. This development is important because it gives insight into the well-known K-method and suggests how it could be automated for musical acoustics. It also provides a theoretical link to the Discrete K-method, presented last, which we recommend for general nonlinear circuits. Finally we give an example of the methods as applied to a simple nonlinear musical circuit, the diode clipper.

This work describes software that can be downloaded as Python and C++ code at http://ccrma.stanford.edu/~dtyeh/nkmethod10.

II. Previous Work

Many existing commercial products simulate the effect of classic analog circuits and give users access to a wide variety of sounds while replacing cumbersome physical gear, but often musicians find these emulations deficient [1]. Computer models of the circuits can be either perceptually or physically based, or a mixture of both. This work considers the physically based approach, which encompasses circuit simulation, or deriving a model by hand and solving it using an ordinary differential equation solver package. Often researchers develop custom solver software employing numerical integration techniques.

Circuit simulation is a mature field whose methods are well-documented in [2], [3]. Circuits can be specified as netlists – lists of circuit components specifying their model parameters and connectivity. The simulator automatically sets up and solves the nonlinear circuit equations to produce transient, or time-domain, responses to specified inputs. Popular circuit simulators such as SPICE (Simulation Program with Integrated Circuit Emphasis) cannot be used as real-time audio effects processors because they were designed for generality and offline use. SPICE simulation requires manual supervision because the solution to nonlinear equations often do not converge given a particular set of solver parameters. Furthermore, for highly saturating distortion circuits, the convergence and error control mechanism in SPICE tends to select a very small sampling period to achieve accuracy that far exceeds audible perceptibility. A simplified yet perceptually accurate simulation designed for audio effects processing would suffice with greatly relaxed computational requirements.

Alternatively one can use numerical integration to solve nonlinear systems of ordinary differential equations (ODEs) [4]. This requires the derivation of model equations that describe the behavior of the system, which is tedious and challenging for those without experience in musical acoustics or circuit analysis. There can be many different models that simulate the same system, depending on the choice of state variables and unknowns to represent the quantities in the system. This work presents one method of assigning these variables. Using off-the-shelf ODE packages also incurs some of the same problems of computational cost as circuit simulators because they are based upon the same principles of numerical integration and solving nonlinear systems iteratively.

Custom numerical simulation of lumped nonlinear systems in audio or musical acoustics based upon numerical integration has been studied extensively in the literature [5]–[11]. The computational musical acoustics / digital audio effects community has developed two prevailing methods for simulating ordinary differential equations with nonlinearities based on Wave Digital Filtering (WDF) principles (nonlinear WDFs [5]) or directly solving a nonlinear state-space system using Kirchhoff variables (K-Method [7]). 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. Again, the typical practitioner would experience difficulty applying these techniques to a circuit or system under consideration because they require the formulation of a model, and various formulations may exist.

This work extends attempts to simulate musical circuits using custom code based upon solving ordinary differential equations with numerical integration [10]–[14]. Previous work demonstrated the applicability of the K-method to audio distortion circuits [15]. While the results are promising in terms of accuracy and efficiency, these approaches all require intensive effort and problem-domain expertise to derive a model. Fusing the generality of circuit simulation techniques with the robust real-time efficiency of the K-method, this work increases the accessibility of custom modeling of musical circuits by automatically generating simulation code from a netlist description.

III. K-Method

Originally developed to model nonlinear systems in musical instruments, such as collision or reed dynamics, the K-Method derives a nonlinear filter that solves nonlinear state-space ODEs. The K-Method requires that an ODE modeling the system under consideration be derived in a particular form. The K-Method provides a general algorithm in matrix notation to solve a nonlinear ODE in this form. It does so by first discretizing the time derivative, then solving for the unknowns at the current time in terms of prior system state and inputs, which involves a nonlinear solver. If possible, solving the nonlinear function beforehand eliminates convergence problems during runtime. This results in an explicit and robust signal processing structure suitable for physical modeling of circuits in real time.

A. Form of system equations required by the K-Method

The K-Method solves ODEs of the form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{C}\mathbf{i}, \tag{1}$$

$$\mathbf{i} = \mathbf{f}\left(\mathbf{v}\right), \tag{2}$$

$$\mathbf{v} = \mathbf{D}\mathbf{x} + \mathbf{E}\mathbf{u} + \mathbf{F}\mathbf{i}, \tag{3}$$

where $\mathbf{x} \in \mathbb{R}^N$ is a vector representing the N state variables of the system, $\mathbf{u} \in \mathbb{R}^M$ represents the M inputs, and $\mathbf{i} \in \mathbb{R}^K$ incorporates the influence on the system dynamics by a nonlinear vector function $\mathbf{f}\left(\mathbf{v}\right) : \mathbb{R}^L \to \mathbb{R}^K$ with $\mathbf{v} \in \mathbb{R}^L$.

In most audio circuits, the vector $\mathbf{x}$ usually consists of the voltages across the capacitors in the circuit; vector $\mathbf{u}$ is usually voltage source signals and power supplies coming into the circuit. The symbol $\mathbf{i}$ is chosen for the nonlinear variable because nonlinear devices in a circuit, such as diodes, and transistor-like devices, are voltage-controlled current sources, giving a nonlinear mapping from the L controlling voltages $\mathbf{v}$ to the K nonlinear device currents.

The nonlinear dynamical system is partitioned into a dynamical part (1), and a purely static functional relationship that represents the overall nonlinearity of the circuit (2) and (3). The latter expresses the nonlinear relationship between the circuit variables often in implicit form, is often a transcendental equation, and must be solved by iterative methods.

In the dynamical portion of this formulation, coefficient matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ 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 $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{i}$ through coefficient matrices $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$.

Finally, coefficient matrices $\mathbf{L}$, $\mathbf{M}$, and $\mathbf{N}$ are needed to relate the system variables to the desired output variables of interest, vector $\mathbf{y}$,

$$\mathbf{y} = \mathbf{L}\mathbf{x} + \mathbf{M}\mathbf{u} + \mathbf{N}\mathbf{i}. \tag{4}$$

B. Discretization and solution

The K-Method solves the system of (1)–(3) with a nonlinear, recursive, discrete-time filter. Discretizing (1) by a stiffly stable numerical integration method [10] such as Backward Euler or the implicit trapezoidal rule we can derive the state update procedure for this filter.

For example, using Backward Euler

$$\dot{\mathbf{x}}_n = \alpha\left(\mathbf{x}_n - \mathbf{x}_{n-1}\right), \tag{5}$$

where $\alpha = 1/T$ and T is the sampling period and using subscripts to denote the time sample, we can use (1) to substitute for $\dot{\mathbf{x}}[n]$ in (5) and solve for

$$\mathbf{x}_n = \alpha \mathbf{H}\mathbf{x}_{n-1} + \mathbf{H}\left(\mathbf{B}\mathbf{u}_n + \mathbf{C}\mathbf{i}_n\right), \tag{6}$$

where

$$\mathbf{H} = \left(\alpha \mathbf{I} - \mathbf{A}\right)^{-1}, \tag{7}$$

which serves as a state update formula if the matrix is invertible. This can be used with (2) and (3) to find $\mathbf{i}_n$, the nonlinear device currents at sample n, using a nonlinear equation solver such as Newton's method. The following subsections demonstrate that one can solve for either $\mathbf{i}_n$ or $\mathbf{v}_n$, which give slightly different convergence properties.

1) Classical K-method: The only unknown in the state update (6) is $\mathbf{i}_n$, the outputs of the nonlinear function (2), or, for circuits, the nonlinear device currents, at the current time step. Combining (2) and (3) to eliminate $\mathbf{v}_n$,

$$\mathbf{i}_n = \mathbf{f}\left(\mathbf{D}\mathbf{x}_n + \mathbf{E}\mathbf{u}_n + \mathbf{F}\mathbf{i}_n\right). \tag{8}$$

Using (6) in (8) results in a nonlinear equation with one unknown,

$$\mathbf{i}_n = \mathbf{f}\left(\alpha \mathbf{D}\mathbf{H}\mathbf{x}_{n-1} + \left(\mathbf{D}\mathbf{H}\mathbf{B} + \mathbf{E}\right)\mathbf{u}_n + \left(\mathbf{D}\mathbf{H}\mathbf{C} + \mathbf{F}\right)\mathbf{i}_n\right). \tag{9}$$

The K-method defines

$$\mathbf{K} = \mathbf{D}\mathbf{H}\mathbf{C} + \mathbf{F}, \tag{10}$$

$$\mathbf{p}_n = \alpha \mathbf{D}\mathbf{H}\mathbf{x}_{n-1} + \left(\mathbf{D}\mathbf{H}\mathbf{B} + \mathbf{E}\right)\mathbf{u}_n \tag{11}$$

so (9) can be written

$$\mathbf{0} = \mathbf{f}\left(\mathbf{p}_n + \mathbf{K}\mathbf{i}_n\right) - \mathbf{i}_n. \tag{12}$$

This is a nonlinear equation with one parameter $\mathbf{p}_n$, which can be solved for unknown $\mathbf{i}_n$ using a nonlinear solver such as Newton's method.

If the mapping from $\mathbf{p}_n$ and $\mathbf{i}_n$ is functional, (12) is an implicitly defined nonlinearity with parameter $\mathbf{p}_n$,

$$\mathbf{i}_n = \mathbf{g}\left(\mathbf{p}_n\right), \tag{13}$$

a function representing the memoryless nonlinearity of the system. This condition can be rigorously described using the implicit mapping theorem (see [7] for details) as

$$\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\mathbf{K} - \mathbf{I}\right) \neq 0. \tag{14}$$

This happens to be the condition under which the linearized system in Newton's method can be solved. If the solution exists, then, locally, Newton's method will find the solution. It may be possible that there are more than one local solution. However, most audio amplification circuits will have a single stable operating point. Multiple operating points are often the result of regenerative, or positive, feedback, which is typically not found in audio amplification circuits.

This memoryless nonlinearity depends on the method of discretization because $\mathbf{K}$ depends on $\mathbf{H}$, which encapsulates the discretization parameters. Furthermore, (13) takes a linear transformation of states and inputs to generate an effective input to the original nonlinear function. Borin, et al. [7] noted that this can be done prior to runtime to generate the explicit function $\mathbf{g}(\mathbf{p}_n)$. In the case of highly saturating nonlinearities, convergence is difficult and offline computation allows manual tuning of the solver parameters to achieve an accurate solution. This method is designed to solve such nonparametric, highly saturating nonlinearities that would otherwise present convergence problems.

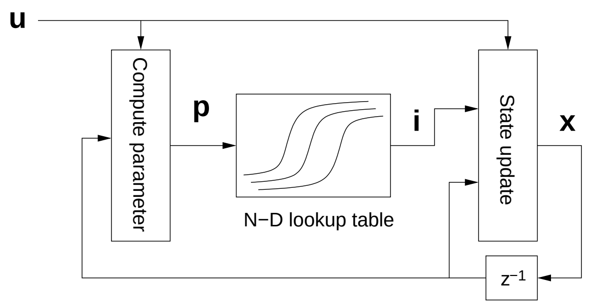

Figure 1. Nonlinear state-space filter corresponding to the K-method solution.

2) Modified K-method: It was found experimentally that iteration on the nonlinear terminal voltages converges faster than iteration on the device currents; therefore, we solve for $\mathbf{v}_n$ instead. This is likely due to the compressive nature of the voltages across device terminals as a function of current. Substitute (2) in (3) and eliminate $\mathbf{x}_n$ to derive a nonlinear equation for $\mathbf{v}_n$

$$\mathbf{0} = \mathbf{p}_n + \mathbf{K}\mathbf{f}(\mathbf{v}_n) - \mathbf{v}_n, \tag{15}$$

where the parameter $\mathbf{p}_n$ and matrix coefficient $\mathbf{K}$ are defined as previously for the classical K-method.

The existence of an explicit function relating the parameter to the nonlinear variable

$$\mathbf{v}_n = \mathbf{\Gamma}\left(\mathbf{p}_n\right) \tag{16}$$

is given by the condition

$$\det\left(\mathbf{K}\frac{\partial \mathbf{f}}{\partial \mathbf{v}} - \mathbf{I}\right) \neq 0, \tag{17}$$

which differs from (14) in the sequence of multiplication by $\mathbf{K}$.

C. Signal processing algorithm

Given the matrix coefficients as in Sec. III-A, the K-method can generate the following explicit state update procedure:

1) Compute $\mathbf{p}_n$ using (11),
2) Compute $\mathbf{i}_n = \mathbf{f}(\mathbf{\Gamma}(\mathbf{p}_n))$ by lookup into transformed nonlinear function (16) and (2),
3) Perform state update using (6),
4) Compute outputs using (4).

This is essentially a recursive nonlinear state space filter with a nonlinearity in the loop as shown in Fig. 1.

IV. Nodal K-Method

It is tedious to derive the ODE for each modeled circuit to use with the K-method. The Nodal K-method employs the Modified Nodal Analysis technique to automate this process, prescribing a procedure for deriving the ODE in the form required by the K-method; i.e., the NK-method automates the derivation of the K-method parameters described in Sec. III-A. The NK-method splits the process of modeling a circuit into an automated offline analysis phase, and a runtime nonlinear filter.

A. Modified Nodal Analysis

Contemporary circuit simulators set up nodal equations for circuits in a matrix formulation known as Modified Nodal Analysis (MNA) [2], [3]. MNA can be adapted to solve for the nonlinear state-space form required for the K-method.

MNA sets up a matrix system of equations for the nodal equations of a circuit

$$\mathbf{G}\nu = \mathbf{c}, \tag{18}$$

where $\mathbf{G}$ is the conductance matrix, $\nu \in \mathbb{R}^P$ represents the voltages of the P nodes in the circuits, $\mathbf{c} \in \mathbb{R}^P$ is a vector of any current sources into each of the nodes. Each row in the system corresponds to KCL for a node. Solving the matrix system yields the solution for the unknowns vector ν.

Each component type (resistor, capacitor, inductor, mutual inductance/transformer, diode, bipolar junction transistor) contributes to rows of ν and $\mathbf{c}$ and also to rows and columns of $\mathbf{G}$ corresponding to the component's terminal nodes. This allows straightforward programmatic construction of $\mathbf{G}$, ν and $\mathbf{c}$ by applying rules to a list of circuit components and their connectivity.

MNA augments the system with additional unknowns added as rows to the system to solve for, e.g., current into a voltage source. In this case, the source voltage is placed in a new row in source vector $\mathbf{c}$ and its unknown current augmented to the unknowns vector ν. Solving for these additional unknowns requires auxiliary equations, in this case, relating the source voltage to the node voltages.

B. Application to K-Method

To apply the K-Method to circuits, one chooses the state variables $\mathbf{x}$ to be the voltages across the terminals of capacitors and the currents through inductors. The inputs $\mathbf{u}$ represent the input current and voltage sources to the circuit. Nonlinear devices contribute nonlinear currents $\mathbf{i}$ computed by vector function $\mathbf{f}$, which maps control voltages $\mathbf{v}$ across the sense terminals to the terminal currents of the devices. Because voltages in ν correspond to nodes in the circuit, the controlling voltages are specified by the difference in node voltages of the two sense terminals, which are termed *controlling pairs*.

One can modify the MNA system to solve for parameters in the form of (1). Namely, decompose $\mathbf{c}$ into contributions from the right hand side of (1) and augment the unknowns vector with the state derivatives:

$$\mathbf{G}\nu = \mathbf{M}_1\mathbf{x} + \mathbf{M}_2\mathbf{u} + \mathbf{M}_3\mathbf{i}. \tag{19}$$

One then solves for the unknowns ν by inverting $\mathbf{G}$. This solves for all the node voltages, voltage source currents and state derivatives $\dot{\mathbf{x}}$ in terms of $\mathbf{x}$, $\mathbf{u}$, and $\mathbf{i}$, which gives expressions for K-method coefficients $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$. Each controlling voltage in $\mathbf{v}$ can be found by subtracting the node voltages in ν that correspond to its controlling pair to find $\mathbf{D}$, $\mathbf{E}$, and $\mathbf{F}$. The outputs $\mathbf{y}$ are usually voltages between certain nodes, so $\mathbf{L}$, $\mathbf{M}$, and $\mathbf{N}$ can be found in a similar way.

In the NK-method, each element contributes to a particular row or column of the MNA matrix system. Arrays keep track of these index for states, sources, nonlinearities, controlling pairs, and equation row. The number of rows in the MNA system corresponds to the number of nodes in the circuit plus the number of auxiliary equations. Each row in the linear system corresponds to a linearly independent equation that helps find the unknowns, which are node voltages and additional quantities such as voltage source terminal currents, and state derivatives.

Upon scanning the netlist, the parser counts the number of nodes, state elements, sources, controlling pairs and nonlinearities in the circuit, allocates space for the MNA system, and assigns indices to those elements. A second pass through the elements fills in the matrices at the indices assigned in the first pass with the value specified by the element.

Templates can be derived for each component type to fill in $\mathbf{G}$ and $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$. Templates also assign contributions from the nonlinear elements to $\mathbf{f}$ and its Jacobian, $\mathbf{J_f}(\mathbf{v})$. An example of the process is given in Sec. VII-A.

Table I shows the templates to build the Nodal K-method system in (19). An entry in the table like $\mathbf{G}[i,j] \leftarrow x$ indicates that the element adds x to the value at the ith row and jth column of matrix $\mathbf{G}$. The symbols n_+, n_- refer to the entries in unknowns vector ν that correspond to the circuit nodes of the $+$ and $-$ terminals of two terminal elements. The indices $\mathrm{eq_i}$, $\mathrm{state_i}$, or $\mathrm{src_i}$ denotes the entry in unknowns vector ν, state vector $\mathbf{x}$, or source vector $\mathbf{u}$ of the ith auxiliary equation, state element, or source element respectively. Index $\mathrm{nl_i}$ denotes the entry of the output of vector function $\mathbf{f}(\mathbf{v})$ corresponding to the ith nonlinear current equation, and $\mathrm{cp_j}$, the entry of the vector of controlling voltages $\mathbf{v}$ corresponding to the jth controlling pair. A definition in the table such as $\mathbf{u}[\mathrm{src_i}] \triangleq I$ indicates for example that the output current of current source I is assigned to the $\mathrm{src_i}$-th entry of source vector $\mathbf{u}$. As another example, $\mathbf{v}[\mathrm{cp_j}] \triangleq \nu[\mathrm{n}_+] - \nu[\mathrm{n}_-]$ says that the $\mathrm{cp_j}$-th entry of $\mathbf{v}$ is assigned to be the difference between the node voltages of the $+$ and $-$ terminals of the nonlinear element. These rules and others have been implemented in Python to automatically derive K-method parameters from a netlist.

V. Discrete K-Method

It was found that in certain circuits containing reactive elements in series, $\mathbf{G}$ in the Nodal K-method was singular. This is due to the nature of differential algebraic equations. Circuit simulators are able to handle a wide variety of circuits because they discretize circuit elements first and then apply MNA. Using the same approach with a nonlinear discrete time system, termed the DK-method, improves the robustness of the automated parameter derivation.

Table I
MNA TEMPLATES FOR NODAL K-METHOD

Resistor $R = 1/G$	
$\mathbf{G}\,[\mathrm{n}_+, \mathrm{n}_+]$	G
$\mathbf{G}\,[\mathrm{n}_+, \mathrm{n}_-]$	$-G$
$\mathbf{G}\,[\mathrm{n}_-, \mathrm{n}_+]$	$-G$
$\mathbf{G}\,[\mathrm{n}_-, \mathrm{n}_-]$	G
Capacitor C	
$\mathbf{G}\,[\mathrm{n}_+, \mathrm{eq_i}]$	C
$\mathbf{G}\,[\mathrm{n}_-, \mathrm{eq_i}]$	$-C$
$\mathbf{G}\,[\mathrm{eq_i}, \mathrm{n}_+]$	1
$\mathbf{G}\,[\mathrm{eq_i}, \mathrm{n}_-]$	-1
$\mathbf{M}_1\,[\mathrm{eq_i}, \mathrm{state_i}]$	1
$\nu\,[\mathrm{eq_i}] \triangleq \frac{dV_C}{dt}$	
$\mathbf{x}\,[\mathrm{state_i}] \triangleq V_C$	
VSource V	
$\mathbf{G}\,[\mathrm{n}_+, \mathrm{eq_i}]$	1
$\mathbf{G}\,[\mathrm{n}_-, \mathrm{eq_i}]$	-1
$\mathbf{G}\,[\mathrm{eq_i}, \mathrm{n}_+]$	1
$\mathbf{G}\,[\mathrm{eq_i}, \mathrm{n}_-]$	-1
$\mathbf{M}_2\,[\mathrm{eq_i}, \mathrm{src_i}]$	1
$\mathbf{u}\,[\mathrm{src_i}] \triangleq V$	
$\nu\,[\mathrm{eq_i}] \triangleq I_V$, voltage source current	
ISource I	
$\mathbf{M}_2\,[\mathrm{n}_+, \mathrm{src_i}]$	-1
$\mathbf{M}_2\,[\mathrm{n}_-, \mathrm{src_i}]$	1
$\mathbf{u}\,[\mathrm{src_i}] \triangleq I$	
Diode $I_d\left(\mathbf{v}\left[\mathrm{cp_j}\right]\right)$	
$\mathbf{M}_3\,[\mathrm{n}_+, \mathrm{nl_i}]$	-1
$\mathbf{M}_3\,[\mathrm{n}_-, \mathrm{nl_i}]$	1
$\mathbf{J_f}\,\left[\mathrm{nl_i}, \mathrm{cp_j}\right]$	$\left.\frac{\partial I_d}{\partial V_d}\right\|_{\mathbf{v}[\mathrm{cp_j}]}$
$\mathbf{v}\,\left[\mathrm{cp_j}\right] \triangleq \nu\,[\mathrm{n}_+] - \nu\,[\mathrm{n}_-]$	
$\mathbf{f}\,[\mathrm{nl_i}] \triangleq I_d\left(\mathbf{v}\left[\mathrm{cp_j}\right]\right)$	

The DK-method starts with a new set of templates for discretized reactive elements, and performs a different form of MNA to find state transition coefficient matrices and the nonlinear relationships between system variables. First we present a derivation of the discretized reactive element templates, known as companion circuits. Then we re-derive the coefficient matrices using these discretized reactive elements. This produces a new form of nonlinear filter similar to that of the K-method.

Like the NK-method, the DK-method splits the process of modeling a circuit into an automated analysis phase to derive runtime parameters from a netlist, and a robust runtime filter to simulate the circuit in real time.

A. Companion circuit

Component-wise discretization of circuit state elements results in companion circuits, which replace the original component in simulation with equivalent discrete-time components. This means that a different set of templates (shown in Tab. II)is used for these discretized reactive elements during construction of the MNA matrix equation. Figure 2 depicts a companion circuit for capacitors using trapezoidal rule discretization,

$$x[n] = x[n-1] + \frac{T}{2}\left(\dot{x}[n] + \dot{x}[n-1]\right). \tag{20}$$

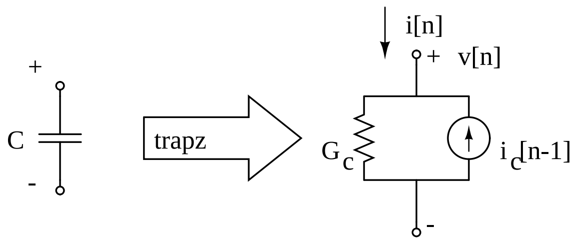

Figure 2. Companion circuit from discretization of capacitor.

B. Derivation of the state update

Trapezoidal rule discretization results in a state update that can be written in general as

$$x[n] = g\,v[n] + s\,x[n-1]. \tag{21}$$

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 both types of elements, $v[n]$ is the voltage across the device at time step n.

For systems with multiple state elements, we can write the state update (21) in vector form

$$\mathbf{x}[n] = \mathbf{G}_\mathrm{e}\mathbf{v}_\mathrm{e}[n] + \mathbf{S}\mathbf{x}[n-1], \tag{22}$$

where $\mathbf{G}_\mathrm{e}$ is a diagonal matrix comprising the scalars g relating the voltages across the elements $\mathbf{v}_\mathrm{e}$ to their simulation state variables $\mathbf{x}$, which depend on the discretization method. Diagonal matrix $\mathbf{S}$ multiplies each state by the appropriate sign s for state update.

These state elements can then be incorporated in MNA as conductances in matrix $\mathbf{G}$ and sources $\mathbf{c}$, with ν the vector of node voltages and unknown terminal currents. Table II shows the modified templates for building the MNA system (24) using companion circuits.

C. Solution of the discrete K-method

We seek an expression for the element variables in the form

$$\mathbf{v}_\mathrm{e}[n] = \mathbf{A}_\mathrm{e}\mathbf{x}[n-1] + \mathbf{B}_\mathrm{e}\mathbf{u}[n] + \mathbf{C}_\mathrm{e}\mathbf{i}[n], \tag{23}$$

where vector $\mathbf{v}_\mathrm{e}$ comprises the voltages across the state elements as computed by the coefficient matrices $\mathbf{A}_\mathrm{e}$, $\mathbf{B}_\mathrm{e}$, $\mathbf{C}_\mathrm{e}$, 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 $\mathbf{c}$ in MNA into contributions from the states (companion circuit sources), inputs, and nonlinear elements, we can write a system in the form

$$\mathbf{G}\nu[n] = \mathbf{M}_1\mathbf{x}[n-1] + \mathbf{M}_2\mathbf{u}[n] + \mathbf{M}_3\mathbf{i}[n]. \tag{24}$$

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

Table II
MODIFIED TEMPLATES FOR REACTIVE ELEMENTS IN DISCRETE K-METHOD

Capacitor C	
$\mathbf{G}\left[\mathrm{n}_+, \mathrm{n}_+\right]$	G_C
$\mathbf{G}\left[\mathrm{n}_+, \mathrm{n}_-\right]$	$-G_C$
$\mathbf{G}\left[\mathrm{n}_-, \mathrm{n}_+\right]$	$-G_C$
$\mathbf{G}\left[\mathrm{n}_-, \mathrm{n}_-\right]$	G_C
$\mathbf{M}_1\left[\mathrm{n}_+, \mathrm{state_i}\right]$	1
$\mathbf{M}_1\left[\mathrm{n}_-, \mathrm{state_i}\right]$	-1
$\mathbf{x}\left[\mathrm{state_j}\right] \triangleq= i_C$	
$G_C = T\,/\,2C$ for trapezoidal integration	
Inductor L	
$\mathbf{G}\left[\mathrm{n}_+, \mathrm{eq_i}\right]$	1
$\mathbf{G}\left[\mathrm{n}_-, \mathrm{eq_i}\right]$	-1
$\mathbf{G}\left[\mathrm{eq_i}, \mathrm{n}_+\right]$	-1
$\mathbf{G}\left[\mathrm{eq_i}, \mathrm{n}_-\right]$	1
$\mathbf{G}\left[\mathrm{eq_i}, \mathrm{eq_i}\right]$	Z_L
$\mathbf{x}\left[\mathrm{state_j}\right] \triangleq v_L$	
$\mathbf{v}_\mathrm{a}\left[\mathrm{eq_i}\right] \triangleq I_L$	
$Z_L = 2L\,/\,T$ for trapezoidal integration	
Mutual Inductance K	
$\mathbf{G}\left[\mathrm{eq_{L1}}, \mathrm{eq_{L2}}\right]$	M_{12}
$\mathbf{G}\left[\mathrm{eq_{L2}}, \mathrm{eq_{L1}}\right]$	M_{12}
$M_{12} = K\sqrt{L_1\,L_2}$	
$\mathrm{eq_{L1}}$, $\mathrm{eq_{L2}}$ is row # of I_{L1}, I_{L2} in $\mathbf{v}_\mathrm{a}$	

Subtracting node voltages in the solution ν corresponding to terminals of the state elements finds the element voltages $\mathbf{v}_\mathrm{e}$ (23). Combining (23) with (22) gives a state update equation

$$\mathbf{x}[n] = \mathbf{A}\mathbf{x}[n-1] + \mathbf{B}\mathbf{u}[n] + \mathbf{C}\mathbf{i}[n], \tag{25}$$

where coefficient matrices $\mathbf{A} = \mathbf{G}_\mathrm{e}\mathbf{A}_\mathrm{e} + \mathbf{S}$, $\mathbf{B} = \mathbf{G}_\mathrm{e}\mathbf{B}_\mathrm{e}$, $\mathbf{C} = \mathbf{G}_\mathrm{e}\mathbf{C}_\mathrm{e}$.

The terminal voltages, $\mathbf{v}[n]$, for the nonlinear devices can also be expressed implicitly as in the K-method as

$$\mathbf{0} = \mathbf{p}[n] + \mathbf{F}\mathbf{f}(\mathbf{v}[n]) - \mathbf{v}[n], \tag{26}$$

$$\mathbf{p}[n] = \mathbf{D}\mathbf{x}[n-1] + \mathbf{E}\mathbf{u}[n], \tag{27}$$

by subtracting rows of ν corresponding to the controlling pairs to find coefficient matrices $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$. As in the K-method (26) can often be solved for an explicit mapping

$$\mathbf{v}[n] = \mathbf{\Gamma}(\mathbf{p}[n]). \tag{28}$$

Output coefficients $\mathbf{L}$, $\mathbf{M}$, and $\mathbf{N}$ are also derived in the same manner.

D. Summary of the DK-method

As with the NK-method, the DK-method scans the netlist describing the circuit and builds the MNA system (24) according to the relevant rules in Tabs. I and II. It then solves for the DK-method parameters as derived above, which can be used to generate a recursive nonlinear filter for use in a runtime loop.

The runtime loop for this discrete-time K-method can be summarized as

1) Change of variable to parameter using (27),
2) Nonlinear function lookup $\mathbf{i}[n] = \mathbf{f}(\mathbf{\Gamma}(\mathbf{p}[n]))$ using solved nonlinearity (28), (2),
3) State update using (25),
4) Compute outputs using (4).

VI. NONLINEAR SOLVERS

The K-method and its variants require the use of a nonlinear root finder to solve the implicit nonlinear equations (12), (15), or (26) for the unknown nonlinear variable at time n. There are several methods to do this as described in [16]. These include fixed point iteration, and Newton's method, which is a particular form of fixed point iteration. We opt for Newton's method because it converges at a second-order rate, and is guaranteed to converge if initialized close to the solution [16]. Furthermore, circuits tend to be classified as "stiff" ODEs and fixed point iteration is known to perform poorly for this class of problems [4].

A. Newton's method parameters

Newton's method solves a systems of equations in the form

$$\mathbf{0} = \mathbf{R}(\mathbf{x}). \tag{29}$$

Newton's method finds values for $\mathbf{x}$ that cause the residual function, the right hand side of (29) $\mathbf{R}(\mathbf{x})$, to evaluate to $\mathbf{0}$. It does so by iterating through the process of taking a guess for $\mathbf{x}$, evaluating how far it is from the solution, and moving towards that solution. This measure of distance from the final solution requires the Jacobian, denoted $\mathbf{J}_\mathbf{R}(\mathbf{x})$, of the residual function $\mathbf{R}(\mathbf{x})$ with respect to $\mathbf{x}$ and evaluated at $\mathbf{x}$.

Newton's method is in general given by

$$\begin{aligned} \Delta\mathbf{x} &= -{\mathbf{J}_\mathbf{R}}^{-1}(\mathbf{x})\ \mathbf{R}(\mathbf{x}) \\ \mathbf{x} &:= \mathbf{x} + \Delta\mathbf{x} \end{aligned} \tag{30}$$

and iterating until the $\mathcal{L}_\infty$ norm is below some acceptable parameter value RELTOL

$$\|\Delta\mathbf{x}\|_\infty < \mathrm{RELTOL}.$$

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

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

$$\det\left(\mathbf{J}_\mathbf{R}(\mathbf{x})\right) \neq \mathbf{0}.$$

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

$$\|\mathbf{R}(\Delta\mathbf{x})\|_\infty < \mathrm{MAXRES}.$$

These two parameters RELTOL and MAXRES govern the accuracy of the Newton's method solver. In the Nodal K-method, the variable to be solved for is a voltage and RELTOL can be viewed as an noisy voltage error, which can be masked beneath other noise sources in a realistic system.

To make this explicit, we derive the equations corresponding to (30) for each proposed method. Because the K-method is a transient simulation, the following is solved at each time step, so the subscript n is dropped for notational simplicity. Furthermore, the residual function is parametric with respect to $\mathbf{p}$ and is thus denoted $\mathbf{R}_\mathbf{p}$.

1) Classical K-method: Defining the residual function (29) to be the right hand side of (12),

$$\mathbf{R}_\mathbf{p}(\mathbf{i}) = \mathbf{f}(\mathbf{p}+\mathbf{K}\mathbf{i}) - \mathbf{i}, \tag{31}$$

we find, by the matrix chain rule for differentiation,

$$\mathbf{J}_\mathbf{R}(\mathbf{i}) = \mathbf{J}_\mathbf{f}(\mathbf{p}+\mathbf{K}\mathbf{i})\mathbf{K} - \mathbf{I}, \tag{32}$$

where $\mathbf{J}_\mathbf{f}$ is the Jacobian of the device nonlinearities (2).

Using this in (30) we arrive at the linearized system to be solved repeatedly until $\mathbf{i}$, the outputs of the device nonlinearities at time n, converge:

$$\left(\mathbf{J}_\mathbf{f}(\mathbf{K}\mathbf{i}+\mathbf{p})\mathbf{K} - \mathbf{I}\right)\Delta\mathbf{i} = -\left(\mathbf{f}(\mathbf{K}\mathbf{i}+\mathbf{p}) - \mathbf{i}\right). \tag{33}$$

2) Modified K-method: Defining the residual function to be the right hand side of (15),

$$\mathbf{R}_\mathbf{p}(\mathbf{v}) = \mathbf{p} + \mathbf{K}\mathbf{f}(\mathbf{v}) - \mathbf{v}, \tag{34}$$

The Jacobian of this residual function is

$$\mathbf{J}_\mathbf{R}(\mathbf{v}) = \mathbf{K}\mathbf{J}_\mathbf{f}(\mathbf{v}) - \mathbf{I}, \tag{35}$$

where $\mathbf{J}_\mathbf{f}$ is again the Jacobian of the device nonlinearities (2). The following equation is then solved repeated until $\mathbf{v}$, the device terminal voltages at time n, converge:

$$\left(\mathbf{K}\mathbf{J}_\mathbf{f}(\mathbf{v}) - \mathbf{I}\right)\Delta\mathbf{v} = -\left(\mathbf{p} - \mathbf{v} + \mathbf{K}\,\mathbf{f}\,(\mathbf{v})\right). \tag{36}$$

3) Discrete K-method: Defining the residual function to be the right hand side of (26),

$$\mathbf{R}(\mathbf{v}) = \mathbf{p} + \mathbf{F}\mathbf{f}(\mathbf{v}) - \mathbf{v}, \tag{37}$$

The Jacobian of this residual function is

$$\mathbf{J}_\mathbf{R}(\mathbf{v}) = \mathbf{F}\mathbf{J}_\mathbf{f}(\mathbf{v}) - \mathbf{I}, \tag{38}$$

where $\mathbf{J}_\mathbf{f}$ is again the Jacobian of the device nonlinearities (2). The following equation is then solved repeated until $\mathbf{v}_n$, the device terminal voltages at time n, converge:

$$\left(\mathbf{F}\,\mathbf{J}_\mathbf{f}(\mathbf{v}) - \mathbf{I}\right)\,\Delta\mathbf{v} = -\left(\mathbf{p} + \mathbf{F}\,\mathbf{f}(\mathbf{v}) - \mathbf{v}\right). \tag{39}$$

4) Discussion: Note that the functions needed for each of the proposed methods depend only on the K-method coefficient matrices, which can be derived using MNA as described above. The only other information needed is the Jacobian of the nonlinear device outputs, which is typically included in the specification for the device model. The same device model equations can be used for each of the proposed methods to build up $\mathbf{f}(\mathbf{v})$ and $\mathbf{J}_\mathbf{f}(\mathbf{v})$ programmatically. This allows automation of the K-method given a netlist description of the circuit.

B. Convergence aids – Homotopy

For circuits with saturating nonlinearities, convergence at extremely large signal levels is difficult. One must provide initial conditions that are very close to the final solution for convergence to happen. It is difficult to know a priori when starting a simulation what initial conditions would be sufficient for the solver to converge. When precomputing the nonlinearity as done in the K-method, one needs to come up with a scheme for finding good initial conditions for every value of $\mathbf{p}_n$. Alternatively modifications to the Newton's method can improve convergence. One method that works well as implemented in this work is that of Newton homotopy [17], a globally convergent method, which is used in advanced circuit simulators to find dc solutions of nonlinear circuits. It parametrizes the residual function (29) and repeatedly solves a new residual function

$$\mathbf{R}_\mathrm{H}(\mathbf{x}_i, \rho_i) = \mathbf{R}(\mathbf{x}_i) - (1-\rho_i)\,\mathbf{R}(\mathbf{x}_0) = \mathbf{0}. \tag{40}$$

Parameter $\rho_i \in [0,1]$ is monotonically stepped for $i \in [1,\,N]$ until $\rho_N = 1$. At the i^th step, homotopy applies Newton's method to find $\mathbf{x}_i$, solving $\mathbf{R}_\mathrm{H}(\mathbf{x}_i,\rho_i)$, using the solution to the $i-1^\mathrm{th}$ step $\mathbf{x}_{n-1}$ as the initial condition. For a sufficiently small increment, $\Delta = \rho_i - \rho_{i-1}$, this initializes Newton's method for each homotopy step with valid solutions until $\rho = 1$ and the final solution is found. An appropriate increment for Δ can be found adaptively by choosing a value for $\Delta = \Delta^*$ and if convergence fails for ρ_i, then halving $\Delta = \Delta^*/2$ repeatedly until a sufficiently small Δ is found that produces convergence. This adaptive search for a good Δ has been implemented and found to be sufficient in practice for the various K-methods in this work.

One intuitive form of applying homotopy is slowly turning on the sources in the system so that the initial values for $\mathbf{x}$ during each intermediate application of Newton's method are always close to the actual intermediate solutions.

To make homotopy more concrete, the residual function for the DK-method would be

$$\mathbf{R}_\mathrm{H}(\mathbf{v}_i,\rho_i) = \mathbf{p}+\mathbf{F}\mathbf{f}(\mathbf{v}_i)-\mathbf{v}_i-(1-\rho_i)\,\left(\mathbf{p}+\mathbf{F}\mathbf{f}(\mathbf{v}_0)-\mathbf{v}_0\right), \tag{41}$$

where $\mathbf{v}_0 = \mathbf{0}$ is the zero vector.

It is obvious that homotopy takes much more computational effort than Newton's method. Because there is no guaranteed time bound for homotopy to converge, and because typically it requires far more total iterations than Newton's method, it is not appropriate for use as a real-time solver. However, because homotopy usually converges to a solution, it is ideal for use as an offline solver with the K-method.

C. Representation of the solved nonlinearity

In order to circumvent convergence problems at runtime, the nonlinearity (13), (16) or (28) was precomputed and stored as a table for real-time implementations of the nodal K-method. Several options exist for looking up this function at runtime.

The simplest option would be to do nearest neighbor approximation – that is, to use the value in the table that is

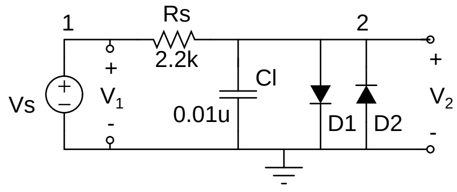

Figure 3. Schematic of the diode clipper with low-pass capacitors.

```
Vs       1  0  0V
Rs       1  2  2.2k
Cl       2  0  0.01u
D1       2  0  n914
D2       0  2  n914
.model  n914  D  (Vt=0.045  Is=2.52e-9  N=1.0)
```

Figure 4. Netlist for Fig. 3

closest to the lookup parameter. Alternatively, a more accurate and smooth function would result from multidimensional linear interpolation, which is the multidimensional extension of linear and bilinear interpolation as done in this work. The table was implemented using a uniform grid for constant-time accesses. Interpolation of grid points was performed using multidimensional interpolation, drawing upon the C++ code implementing `interpn` from the open-source project Octave [18].

The nonlinear multidimensional function (13) can also be implemented using one of many function approximation or nonlinear regression techniques [19]. These include multivariate splines [20], [21], neural networks [19], and support vector regression [19]. In particular, many have tried to model guitar distortion using neural networks, but the complexity of the memory in the nonlinearity makes this technique unreliable. With the K-method, one can instead perform nonlinear system identification [22] on a multidimensional, memoryless nonlinearity, which is a simpler problem for neural networks to solve.

Because Newton's method has guaranteed quadratic convergence if initialized with a guess close to the final solution, another possibility is to use a rough function approximation to initialize Newton's method, which then refines the result for greater accuracy.

Finally, note that many audio circuits are highly parametric. The nonlinearity (13) can be solved sweeping the parameter settings and including the parameter as another dimension in the function approximation. Function approximation allows for interpolation between settings and readily incorporates parametric changes while eliminating the risk of convergence failures during runtime.

VII. Application Example

This section demonstrates the nodal and discrete K-methods applied to the simple single state problem of the one-capacitor diode clipper using trapezoidal rule integration. The behavior of various numerical methods applied to the single-capacitor diode clipper has been studied extensively [10]. Figures 3 and 4 show the corresponding schematic and netlist.

The nonlinearities are due to the two diodes, which are nonlinear voltage-controlled current sources given by

$$I_d(V_d) = I_s\left(\exp\frac{V_d}{V_t} - 1\right), \tag{42}$$

where I_d is the diode current controlled by voltage V_d across the two terminals of the diode, I_s and V_t are physical parameters of the diode.

A. NK-method

This section demonstrates the derivation of the K-method coefficients from the netlist of the diode clipper.

Scanning the netlist determines that the diode clipper has three nodes, including ground. In general, the ground node and associated rows and columns are trimmed from the system, because otherwise the system would be overdetermined. Trimming is done as a postprocess step because it eliminates having to treat ground as a special case during parsing. The unknowns vector is then $\nu = \begin{bmatrix} V_1 & V_2 & I_V & \dot{V}_{Cl} \end{bmatrix}^T$. The last two unknowns require auxiliary equations to nodal analysis, which solve for the current through the voltage source, and the derivative of the capacitor voltage. These unknowns are indexed by the equation index eq_i.

The following describes how each element contributes to the MNA system, ignoring connections to ground for brevity. The voltage source adds an element to the source vector $\mathbf{u} = V_s$, and source current I_V to the unknowns vector. It contributes $\mathbf{G}[3,1] = 1$ and $\mathbf{M}_2[3] = 1$, relating the input source voltage to the nodal voltages, and $\mathbf{G}[1,3] = 1$, adding its current to KCL at node 1.

The resistor has two terminals, the n+ terminal is node 1 and the n- terminal is node 2. According to the template, Rs adds to $\mathbf{G}[1,1] = \mathbf{G}[2,2] = G = 1/R_s$ and $\mathbf{G}[1,2] = \mathbf{G}[2,1] = -G$, which expresses Ohm's law in matrix form.

The capacitor adds a state variable $\mathbf{x} = V_{Cl}$ to the system. It contributes $\mathbf{G}[2,4] = C_l$, adding the capacitor current $I = C_l\dot{V}_{Cl}$ to KCL at node 2, and $\mathbf{M}_1[4] = 1$, $\mathbf{G}[4,2] = 1$, relating the capacitor voltage to the nodal voltages.

The diodes contribute nonlinearities to the system $\mathbf{i} = \begin{bmatrix} i_{D1} & i_{D2} \end{bmatrix}^T$, and contribute $\mathbf{M}_3[2,1] = -1$, $\mathbf{M}_3[2,2] = 1$ respectively. The diode currents are a function of the controlling pair (2,0) between node 2 and ground. Duplicate controlling pairs can be detected and eliminated so that both diodes are controlled by the same controlling pair voltage $\mathbf{v} = \begin{bmatrix} (V_2 - 0) \end{bmatrix}$. They contribute to the global nonlinear current vector function $\mathbf{i}(\mathbf{v}) = \begin{bmatrix} i_{D1}(V_2) & i_{D2}(-V_2) \end{bmatrix}^T$.

The Jacobian can also be derived automatically, assigning the partial derivatives of the diode current to indices of the Jacobian matrix

$$\mathbf{J_f} = \begin{bmatrix} \left.\frac{\partial I_d}{\partial V_d}\right|_{V_2} \\ \left.\frac{\partial I_d}{\partial V_d}\right|_{-V_2} \end{bmatrix}. \tag{43}$$

The resulting MNA corresponding to (19) can be solved for the unknowns and manipulated as derived above to find the

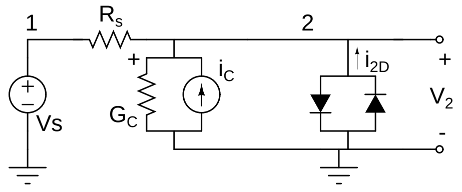

Figure 5. Companion circuit of single capacitor diode clipper.

K-method parameters:

$$\begin{bmatrix} G & -G & 1 & 0 \\ -G & G & 0 & C_l \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ I_V \\ \dot{V}_{Cl} \end{bmatrix} =$$
$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} [V_{Cl}] + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} [V_s] + \begin{bmatrix} 0 & 0 \\ -1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} i_{D1} \\ i_{D2} \end{bmatrix}. \quad (44)$$

Solving (44) for $\dot{V}_{Cl}$ finds the $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ coefficients

$$\dot{V}_{Cl} = \underbrace{-\frac{G}{C_l}}_{\mathbf{A}} V_{Cl} + \underbrace{\frac{G}{C_l}}_{\mathbf{B}} V_s + \underbrace{\begin{bmatrix} -\frac{1}{C} & \frac{1}{C} \end{bmatrix}}_{\mathbf{C}} \begin{bmatrix} i_{D1} \\ i_{D2} \end{bmatrix}, \quad (45)$$

which is the ODE studied in [10]. Likewise, solving for V_2 finds the controlling voltages as well as the output variable, and gives the $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$ and $\mathbf{L}$, $\mathbf{M}$, $\mathbf{N}$ coefficients,

$$V_2 = \underbrace{V_{Cl}}_{\mathbf{D},\mathbf{L}} + \underbrace{0}_{\mathbf{E},\mathbf{M}} V_s + \underbrace{\begin{bmatrix} 0 & 0 \end{bmatrix}}_{\mathbf{F},\mathbf{N}} \begin{bmatrix} i_{D1} \\ i_{D2} \end{bmatrix}.$$

B. DK-method

Alternatively, DK-method avoids having to solve for state derivatives because it first discretizes the state elements, converting them into equivalent companion circuits that contain memory, and results in simpler matrix equations. Figure 5 depicts the companion model for the diode clipper with the low pass capacitor replaced by its companion circuit. The parallel diodes connected with opposite polarities can be combined and written as

$$I_{2D}(V) = 2I_s \sinh(V/V_t). \quad (46)$$

Let the state be $\mathbf{x} = i_C$, the input $\mathbf{u} = V_s$, and the nonlinear currents $\mathbf{i} = i_{2D}$. Define $G = 1/R_s$, $G_C = T/2C$. All circuit variables correspond to time n unless otherwise noted. Following similar parsing procedures to the NK-method and ignoring the ground node for notational simplicity, the nodal DK-method setup corresponding to (24) for this circuit is

$$\begin{bmatrix} G & -G & 1 \\ -G & G+G_C & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ I_V \end{bmatrix} =$$
$$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} i_C[n-1] + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} V_s + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} i_{2D}. \quad (47)$$

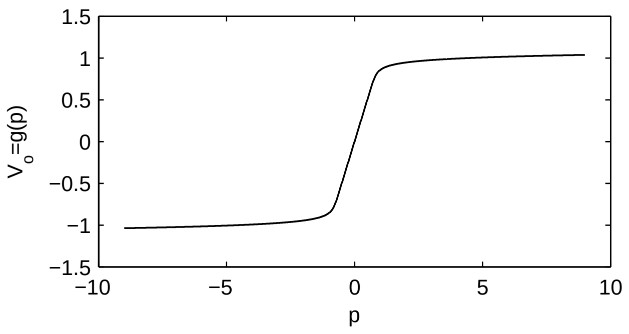

Figure 6. Explicit nonlinearity for single capacitor diode clipper.

Next we solve (47) for V_2 to find the voltage across capacitor, the state element, which corresponds to (23),

$$V_2 = \frac{1}{G_C+G} i_C[n-1] + \frac{G}{G_C+G} V_s + \frac{1}{G_C+G} i_{2D}. \quad (48)$$

Because this is also the controlling voltage and the output variable, this expression also gives $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$ and $\mathbf{L}$, $\mathbf{M}$, $\mathbf{N}$. Using (48) in (22) to solve for i_C derives a state update equation corresponding to (25) and gives $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$:

$$i_C = \frac{G_C-G}{G_C+G} i_C[n-1] + \frac{2G_C G}{G_C+G} V_s + \frac{2G_C}{G_C+G} i_{2D}. \quad (49)$$

Solving for the controlling variable V_2 requires solving the implicit equation (48), which corresponds to (26) and can be rewritten

$$0 = p + \frac{1}{G_C+G} i_{2D}(V_o) - V_o, \quad (50)$$

where V_o is an implicitly defined function

$$V_o = g(p) \quad (51)$$

of parameter

$$p = \frac{1}{G_C+G} i_C[n-1] + \frac{G}{G_C+G} V_s. \quad (52)$$

To compute the output given the input, first compute p by (52), then compute the nonlinear currents i_D using (51) and (46). Update the state using (49) and compute the output voltage using (48).

To illustrate the explicit nature of this computation despite the use of an implicit integration method, the explicit nonlinear function (51) is shown in Fig. 6.

VIII. Conclusions

We have derived first an extension to the K-method of simulating nonlinear musical acoustics to derive simulations of circuits for audio amplification, and second, a more robust, discrete-time version of the K-method for circuits. This work also introduces the use of homotopy, a globally convergent method, to aid Newton-Raphson solution of the nonlinear systems in the K-method. The method allows automatic generation of real-time filters that simulate the desirable nonlinearities of circuits for musical effects applications and greatly facilitates the process of implementing K-method models of nonlinear systems. This approach is general and can easily incorporate device models for a variety of exotic audio frequency devices including germanium transistors and vacuum tubes.

A recursive nonlinear filter in state-space form is the consequence of discretizing the system equations using standard implicit numerical integration formulas used in stiff ODE solvers. Although the nonlinearities require the use of iterative solvers, these can be computed offline, resulting in a runtime algorithm that is explicit. Such an efficient recursive nonlinear filter potentially offers an attractive alternative to computationally complex approximations of nonlinear systems such as Volterra series for applications outside of musical effects, e.g., linearization of communications circuits.

Given this model structure, future extensions to this work may include automatic nonlinear system identification of model parameters from measurements of real circuits using various excitation signals. Furthermore, the parametric nature of musical circuits demands research into the implementation of real-time parameter changes in this nonlinear filter structure. Further work can implement standard mechanical components and integrate waveguides into the netlist specification, thus enabling the automatic generation of musical synthesis routines from equivalent circuit descriptions.

REFERENCES

[1] J. Pakarinen and D. T. Yeh, “A Review of Digital Techniques for Modeling Vacuum-Tube Guitar Amplifiers,” *Comput. Music J.*, vol. 33, no. 2, pp. 85–100, 2009.

[2] A. Vladimirescu, *The Spice Book*. New York: Wiley, 1994.

[3] W. J. McCalla, *Fundamentals of Computer-Aided Circuit Simulation*. Boston: Kluwer Academic Publishers, 1987.

[4] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*. Hoboken, NJ: Wiley, 2003.

[5] A. Sarti and G. De Poli, “Toward nonlinear wave digital filters,” vol. 47, pp. 1654–1668, June 1999.

[6] A. Sarti and G. De Sanctis, “Systematic methods for the implementation of nonlinear wave-digital structures,” vol. 56, no. 2, pp. 460–472, Feb. 2009.

[7] G. Borin, G. De Poli, and D. Rocchesso, “Elimination of delay-free loops in discrete-time models of nonlinear acoustic systems,” vol. 8, no. 5, pp. 597–605, Sep. 2000.

[8] F. Avanzini and D. Rocchesso, “Efficiency, accuracy, and stability issues in discrete time simulations of single reed instruments,” *J. Acoust. Soc. Am.*, vol. 111, no. 5, pp. 2293–2301, May 2002.

[9] F. Fontana and F. Avanzini, “Computation of delay-free nonlinear digital filter networks: Application to chaotic circuits and intracellular signal transduction,” vol. 56, no. 10, pp. 4703–4715, Oct. 2008.

[10] D. T. Yeh, J. S. Abel, A. Vladimirescu, and J. O. Smith, “Numerical Methods for Simulation of Guitar Distortion Circuits,” *Comput. Music J.*, vol. 32, no. 2, pp. 23–42, 2008.

[11] A. Huovilainen, “Enhanced digital models for analog modulation effects,” in *Proc. 8th Int. Conf. Digital Audio Effects (DAFx-05)*, Madrid, Spain, Sept. 20-22 2005, pp. 155–160.

[12] ——, “Nonlinear digital implementation of the Moog ladder filter,” in *Proc. 7th Int. Conf. Digital Audio Effects (DAFx-04)*, Naples, Italy, Oct. 5–8, 2004, pp. 61–64.

[13] M. Karjalainen and J. Pakarinen, “Wave digital simulation of a vacuum-tube amplifier,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 5, Toulouse, France, 2006, pp. 153–156.

[14] M. Civolani and F. Fontana, “A nonlinear digital model of the EMS VCS3 voltage-controlled filter,” in *Proc. 11th Int. Conf. Digital Audio Effects (DAFx-08)*, Espoo, Finland, Sept. 1–4, 2008, pp. 35–42.

[15] D. T. Yeh and J. O. Smith, “Simulating guitar distortion circuits using wave digital and nonlinear state-space formulations,” in *Proc. 11th Int. Conf. Digital Audio Effects (DAFx-08)*, Espoo, Finland, Sept. 1–4, 2008, pp. 19–26.

[16] L. O. Chua, *Computer-Aided Analysis of Electronic Circuits*. Englewood Cliffs: Prentice Hall, 1975.

[17] 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,” vol. 21, no. 3, pp. 337 – 348, Mar. 2002.

[18] J. W. Eaton, *GNU Octave Manual*. Network Theory Limited, 2002. [Online]. Available: http://www.octave.org/

[19] T. Hastie, R. Tibshirani, and J. Friedman, *Nonlinear System Identification*. Berlin: Springer, 2008.

[20] J. H. Friedman, “Multivariate Adaptive Regression Splines,” *The Annals of Statistics*, vol. 19, no. 1, pp. 1–67, 1991.

[21] T. Kavli, “ASMOD – An algorithm for adaptive spline modeling of observation data,” *Int. Journal of Control*, vol. 58, no. 4, pp. 947–967, 1993.

[22] O. Nelles, *Nonlinear System Identification*. Berlin: Springer, 2001.

PLACE PHOTO HERE

David T. Yeh received the B.S. in Electrical Engineering and Computer Sciences at U.C. Berkeley, and the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University in 2009. His thesis topic concerned the modeling of musical distortion circuits such as stompboxes and vacuum tube audio amplifiers primarily used as guitar distortion effects. His research interests include acoustics, music information retrieval, audio signal processing and analog/digital system design. His graduate work has been funded by the Stanford Graduate Fellowship, the National Defense Science and Engineering Graduate Fellowship, and the National Science Foundation Graduate Fellowship.

From 2003-2005 he developed ultrasound medical imaging systems based on cMUTs as a student in the Khuri-Yakub Ultrasonics Lab at Stanford. In 2003 and 2006 he was with National Semiconductor as an analog IC design intern. In 2009 he consulted for Countryman Associates, designing analog circuits and adaptive signal processing algorithms for pro audio microphone applications. He is currently employed at a research lab to design analog/digital system for array imaging systems such as radar and ultrasound.

PLACE PHOTO HERE

Jonathan S. Abel received the S.B. in electrical engineering from MIT in 1982, where he studied device physics and signal processing. He received his M.S. and Ph.D. degrees in electrical engineering from Stanford University in 1984 and 1989, respectively. He is presently a Consulting Professor at the Center for Computer Research in Music and Acoustics (CCRMA) in the Music Department at Stanford University where his research interests include audio and music applications of signal and array processing, parameter estimation, and acoustics. From 1999 to 2007, he was a Co-Founder and Chief Technology Officer of the Grammy Award-winning Universal Audio, Inc. He was also Chief Scientist of Crystal River Engineering, Inc., and a Lecturer in the Department of Electrical Engineering at Yale University. As an industry consultant, Abel has worked with Apple, FDNY, L3, LSI Logic, NRL, SAIC, and Sennheiser, on projects in professional audio, GPS, medical imaging, passive sonar and fire department resource allocation.

PLACE PHOTO HERE

Julius O. Smith received the B.S.E.E. degree from Rice University, Houston, TX, in 1975. He received the M.S. and Ph.D. degrees in E.E. from Stanford University, Stanford, CA, in 1978 and 1983, respectively. He is presently a Professor of Music and Associate Professor of Electrical Engineering (by courtesy) at the Center for Computer Research in Music and Acoustics (CCRMA) at Stanford, teaching courses and pursuing research related to signal processing techniques applied to music and audio systems. For more information, see http://ccrma.stanford.edu/~jos/.