DIFFERENTIABLE WHITE-BOX VIRTUAL ANALOG MODELING

Fabián Esqueda, Boris Kuznetsov and Julian D. Parker

Native Instruments GmbH
Berlin, Germany
firstname.lastname@native-instruments.de

ABSTRACT

Component-wise circuit modeling, also known as "white-box" modeling, is a well established and much discussed technique in virtual analog modeling. This approach is generally limited in accuracy by lack of access to the exact component values present in a real example of the circuit. In this paper we show how this problem can be addressed by implementing the white-box model in a differentiable form, and allowing approximate component values to be learned from raw input–output audio measured from a real device.

1. INTRODUCTION

Digital emulation of analog audio circuits, also known as virtual analog (VA) modeling, is a highly active and mature field of research within musical signal processing. Over more than 20 years, research in the field has covered a wide variety of audio systems such as analog synthesizers [1, 2], effects units [3–5] and vacuum tube guitar amplifiers [6].

VA modeling techniques are commonly divided into two main categories depending on their scope: "white-box" and "black-box". White-box modeling consists in deriving and discretizing the underlying governing equations of a circuit by means of circuit analysis and standard numerical methods. Due to the labor intensive nature of this process, particularly for the case of large circuits with multiple nonlinearities, a vast proportion of research in this field has focused on the development of automated general-purpose circuit modeling frameworks [7–11]. A key characteristic of white-box modeling is that it requires full knowledge of the circuit under study, ideally through circuit schematics and component datasheets. In cases where schematics are not readily available these have to be traced manually from the physical circuit. Furthermore, determining exact component values requires them to be measured in isolation from the circuit, which is not only impractical but can also compromise the integrity of the circuit.

The black-box approach, on the other hand, does not rely on access to the internals of the system under study and is based entirely on measurements. This process requires designing a general-purpose parametric structure that can be tuned manually or optimized automatically to replicate the input–output relationship exhibited by the captured data. For linear systems, this process is somewhat trivial, as this relationship is fully described by their impulse response. However, for the case of nonlinear systems with memory, such as guitar amplifiers and pedals, more complex structures and optimization techniques are required. Examples of black-box VA modeling techniques include the Volterra series [12], dynamic convolution [13, 14] and block-oriented structures such as Wiener-Hammerstein models [15, 16]. In some cases, partial knowledge of the internals of the system is used in the design of a black-box modeling system at the cost of generality. In such cases, the term "gray-box" modeling is commonly used [17].

The rise in popularity of Machine Learning (ML), and deep learning in particular, has seen the emergence of new lines of research in the field of VA modeling. A portion of this research has focused on the use of standard deep-learning architectures, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), to produce black-box models of analog audio systems such as vacuum tube amplifiers [18–21], guitar distortion circuits [22] and modulation effects [23]. Other research has formulated the problem in a form that could be described as gray-box or pseudo white-box [24], where the governing equations for the system under study are discovered and approximated from measurements of the input, output and internal state values. Approaches have also been proposed that attempt to characterise a wide class of audio-effects as transformations of the latent-space of an autoencoder-like structure [25].

In a recent study, Engel *et al.* pointed out that the use of standard deep-learning structures, known for their universality, can often lead to overengineered, inefficient and uninterpretable solutions for certain audio signal processing tasks [26]. To remedy this, the authors introduced the DDSP library, a set of *differentiable* classical DSP processors that can be integrated into larger deep learning models for end-to-end training using backpropagation. These differentiable blocks enable the encoding of structural and domain knowledge into the network, hence potentially reducing model size and training times.

In our previous work, we extended the scope of DDSP and studied the relationship between RNNs and IIR filters, proposing three differentiable IIR filter topologies [27]. As an example application, we presented a fixed (i.e. without user-facing parameters) VA model of a nonlinear guitar distortion pedal implemented using a differentiable Wiener-Hammerstein model consisting of two IIR filters and a multilayer perceptron (MLP), a class of feedfoward artificial neural network. This idea of utilizing differentiable DSP structures for VA modeling is developed further in a recent article by Nercessian *et al.*, where a general modeling architecture comprised of many cascaded differentiable biquad filters and nonlinear activations is proposed and applied to a guitar distortion circuit [28].

Following our previous research, in this paper we introduce the concept of *differentiable white-box VA modeling*. In a similar fashion to differentiable IIR filters, discretized circuit models can be optimized (i.e. trained) using backpropagation to fit raw input–output measurements from a real device. This technique allows us to improve the accuracy of white-box VA models by compen-

sating for errors introduced, for example, by mislabeled schematics, component tolerances or parasitic capacitances. Moreover, it can be used to uncover unknown component values and parameters, as well as potentiometer curves. The proposed approach to white-box system identification has the advantage that it can be integrated within larger deep learning models and can benefit from some of the features of ML frameworks, such as their ability to handle large amounts of data. Of related interest to this study is the work of Holmes and van Walstijn [29], who proposed the use of a Genetic Algorithm to optimize the parameters of a fixed nonlinear treble booster circuit.

This rest of this work is structured as follows. The proposed method will be outlined for the static case in Sec. 2, alongside a case study on a linear RC Filter, including results. In Sec. 3 we discuss how the method can be modified for the parametric case, with both a linear and a nonlinear case study, including results. Finally, Sec. 4 provides concluding remarks.

2. STATIC CIRCUITS

In this section, we introduce the proposed method for the case of static (i.e. non-parametric) circuits. As a first case study, a one-pole lowpass RC filter is considered.

2.1. Method Outline

In the continuous-time domain we can write the underlying system of ODEs governing a circuit in state-space form as

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{1}$$
$$\mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{u}(t)) \tag{2}$$

where the vectors $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{y}$ represent the states, inputs and outputs of the system, respectively. Throughout this work, dot notation is used to indicate a derivative taken with respect to time, i.e. $\dot{x} \equiv dx/dt$.

We can extend the state-space formulation to include parameters, giving us

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)\,;\, \boldsymbol{\lambda}) \tag{3}$$
$$\mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{u}(t)\,;\, \boldsymbol{\lambda}), \tag{4}$$

where vector $\boldsymbol{\lambda}$ is comprised of the system parameters which for the case of circuits have a physical significance, they represent the component values and other electrical constants. We then discretize (3) and (4) to get

$$\mathbf{x}^{n+1} = f_d\left(\mathbf{x}^{n+1} \dots \mathbf{x}^{n-k}, \mathbf{u}^{n} \dots \mathbf{u}^{n-k}\,;\, \boldsymbol{\lambda}\right) \tag{5}$$
$$\mathbf{y}^{n} = g_d\left(\mathbf{x}^{n} \dots \mathbf{x}^{n-k}, \mathbf{u}^{n} \dots \mathbf{u}^{n-k}\,;\, \boldsymbol{\lambda}\right), \tag{6}$$

where $x^n \equiv x[n]$ is shorter notation for the samples of a discrete-time signal. Functions f_d and g_d along with the value of k, which determines the maximum number of previous sample points used in the discretazion, will depend on the discretization/numerical method used.

The task is then to implement this discretized form of the system in a framework that allows automatic differentiation of computational graphs, e.g PyTorch or Tensorflow [30, 31], and train the parameters $\boldsymbol{\lambda}$ to minimise an appropriate loss metric, based on measured input–output data of a real example of the system. This implementation step is a relatively trivial task as these frameworks can handle the operations typically needed in circuit modeling, including matrix inversions and nonlinearities. It is worth noting that usual downsides of back-propagating through a recursive system apply, particularly the strong growth of computational complexity and memory usage as longer training sequences are considered, and the potential for vanishing gradients. Chen et al. [32] present a solution to these problems via the application of the adjoint state method to compute the gradient of system at a particular time-step. This is not applied in this work, but could be useful for more complex circuits if extended to support systems with input.

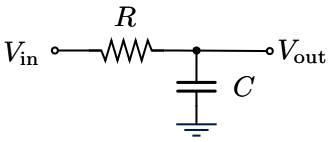

Figure 1: Schematic for a first-order RC lowpass filter.

The loss metric used for training in this work is the standard Mean Squared Error (MSE) applied directly in the time domain, but other metrics like Error to Signal Ratio (ESR) or some type of spectral or perceptually weighted loss metrics could be appropriate [26, 33]. Reference PyTorch implementations for the circuits considered in this study are provided in the accompanying repository[1].

2.2. Case Study: RC Filter

The first circuit considered in this study is the first-order RC lowpass filter shown in Fig. 1. Given the structure of this circuit, we can expect its governing ODE to be of the form

$$\dot{V}_{\text{out}}(t) = f\left(V_{\text{in}}(t), V_{\text{out}}(t)\,;\, [R, C]\right), \tag{7}$$

where $V_{\text{in}}(t)$ and $V_{\text{out}}(t)$ represent the input and state/output of the circuit, respectively, and f is a linear mapping. Using Kirchhoff's Voltage Law (KVL) and Kirchhoff's Current Law (KCL) we can determine function f, which gives us the ODE:

$$\dot{V}_{\text{out}}(t) = \frac{V_{\text{in}}(t) - V_{\text{out}}(t)}{RC}. \tag{8}$$

The product of parameters R and C forms what is known as the *RC time constant*, which has a unit of seconds and determines the cutoff frequency f_{c} (in Hz) of the filter by $f_{\text{c}} = 1/\left(2\pi RC\right)$.

Next, we discretize (8) using the trapezoidal rule, arriving at the difference equation

$$V_{\text{out}}^{n} = \frac{\rho\left(V_{\text{in}}^{n} + V_{\text{in}}^{n-1}\right) + (1-\rho)\,V_{\text{out}}^{n-1}}{1+\rho}, \tag{9}$$

where $\rho = T_{\text{s}}/(2RC)$ and T_{s} is the sampling period. The trapezoidal rule, which in the linear case is equivalent to the bilinear transform, is widely used in DSP due its desirable stability properties, as it maps poles located on the left-hand side of the s-plane to the inside of the unit circle. For the case of a discrete-time RC filter this stability condition is guaranteed as long as the RC time factor remains positive real.

[1] https://github.com/fabianesqueda/differentiable_va_modeling

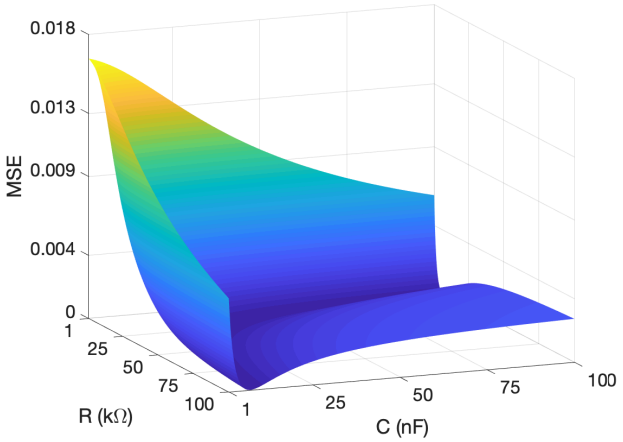

Figure 2: RC Filter Loss Surface

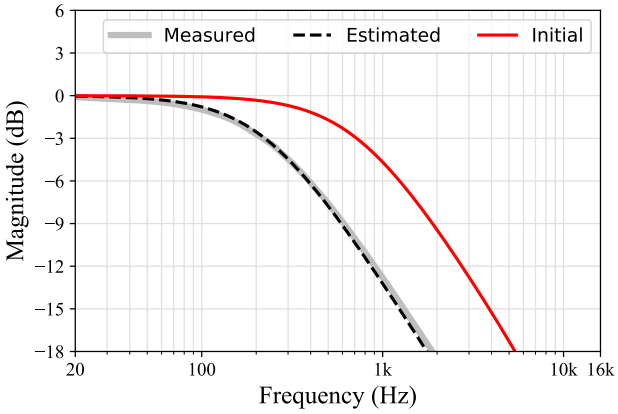

Figure 3: Magnitude response an RC lowpass filter before and after training, plotted against the measured target response.

2.2.1. *Training and Results*

Learning the parameters of an RC circuit involves finding the values of R and C that minimize a loss metric for a given dataset. To demonstrate this, a set of measurements from a physical RC circuit with nominal component values $R = 12\,\text{k}\Omega$ and $C = 68\,\text{nF}$ ($RC \approx 8.16 \times 10^{-4}\,\text{s}$ and $f_\text{c} \approx 195\,\text{Hz}$) were collected. The circuit was built on a breadboard and the input and output signals were recorded using an analog-to-digital converter (ADC). As in [24], unity gain op-amp buffers were placed between the measurement points and the ADC to prevent loading of the circuit. A 3-minute long audio signal composed of a logarithmic sine sweep, followed by a white noise ramp and a combination of guitar, bass and drum recordings, was used for these measurements. All measurements in this study were performed at a sampling rate of 96 kHz.

Figure 2 shows the MSE loss surface of (9) w.r.t. the measured data for different values of R and C. We can clearly see how the global minimum of this loss surface is not given by a single point, but rather by an infinitely-long line. This is due to the fact that – at least in theory – there exists an infinite combination of values of R and C that will yield any given cutoff frequency. A similar problem concerning multiple local minima in circuit parameter search spaces is reported in [29].

As an additional step before training, we introduce two scaling factors G_R and G_C so that ρ in (9) is redefined as

$$\rho = \frac{T_\text{s}}{2(G_\text{R}R)(G_\text{C}C)}. \tag{10}$$

Rather than learning optimal values for R and C directly, we train these two scaling factors instead. This nondimensionalization step is done to compensate for the difference in ranges between these two parameters, which are several orders of magnitude apart.

The RC filter was trained for 100 epochs on the measured data. To simulate a scenario in which the original component values are either wrong or unavailable, parameters R and C in the model were initialized to $4.7\,\text{k}\Omega$ and 47 nF, respectively (i.e. $RC \approx 0.22\,\text{ms}$ and $f_\text{c} \approx 720\,\text{Hz}$). Figure (3) shows the magnitude response of the RC filter before and after training, plotted against that of the real circuit. This result shows a good match between the trained and target responses, and corresponds to a learned RC time constant of approx. 0.745 ms (for $G_R = G_C = 1.837$) or $f_\text{c} \approx 224\,\text{Hz}$. This represents a 29 Hz difference w.r.t. the nominal cutoff frequency, indicating the system was also able to learn variations due to component tolerances. Due to the linear nature of this system, its behavior is characterized entirely by its impulse response, regardless of the driving input. Therefore, it is not necessary to evaluate the performance of the trained model on a separate validation set.

The results from this simple example demonstrate that the proposed technique, while unable to learn the exact component values of the reference circuit, was able to learn its RC time constant and hence reproduce its behavior accurately. As detailed in the following sections, this same approach can be scaled up to tackle larger circuits with multiple parameters and even nonlinearities.

3. PARAMETRIC CIRCUITS

Many circuits of interest have system parameters that can be varied according to user input, for example via the variable resistance of a potentiometer. The nature of the mapping between the user parameters and the system parameters can vary in complexity depending on the particular circuit and on the type of control mechanism used. For example, the control taper of a potentiometer is defined by the physical shape of the embedded resistive material, and may not follow an exact linear or logarithmic relationship. We therefore need to extend the model presented in Sec. 2.1 to account for this behavior. We can do this by splitting the system parameters $\boldsymbol{\lambda}$ into two portions, the static parameters $\boldsymbol{\lambda}_\text{s}$ and the variable parameters $\boldsymbol{\lambda}_\text{v}$, given by:

$$\boldsymbol{\lambda}_\text{v} = f_\text{v}(\boldsymbol{\Lambda}) \tag{11}$$

where $\boldsymbol{\Lambda}$ are the user-facing variable parameters, usually expressed as normalized values. The function f_v can be represented in the trainable system by an MLP or similar network. This process is known as *hyperconditioning* in the general case [34].

3.1. Case Study: FMV Tone Stack

The next circuit considered in this study is the passive RC network shown in Fig. 4. This circuit is known in the literature as the FMV

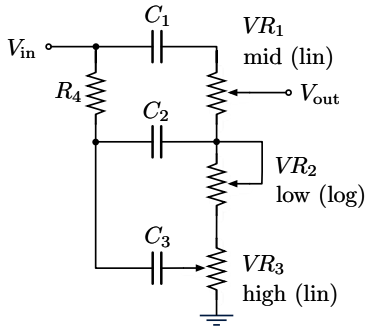

Figure 4: Schematic for the FMV Tone Stack.

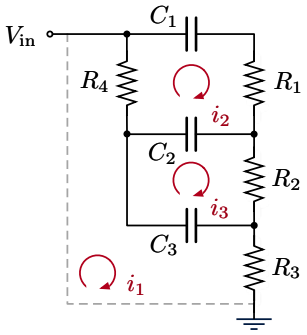

Figure 5: Equivalent representation of the FMV Tone Stack.

Tone Stack and corresponds to the parametric tone section commonly found in Fender, Marshall and Vox guitar amplifiers, hence its name. As reported in [35], the user-facing parameters in the FMV tone stack are non-orthogonal, which results in a complicated parametric frequency response.

We begin our analysis of the circuit by defining the equivalent representation shown in Fig. 5, where resistors

$$R_1 = VR_1, \quad R_2 = lVR_2 + (1-m)VR_3 \quad \text{and} \quad R_3 = mVR_3$$

are defined in terms of the the user-facing parameters l (low) and m (mid) ranging between $[0, 1]$. Next, we identify the three loops or meshes in the circuit, labeled i_{i-3} and highlighted in dark red in the schematic. Mesh i_1, denoted in Fig. 5 with the help of a dashed line, consists of V_{in}, R_4, C_3 and R_3. Using mesh current analysis we can derive a state-space representation of the circuit in terms of these three mesh currents, which can be written as

$$A\dot{\mathbf{i}}(t) + B\mathbf{i}(t) = \dot{\mathbf{u}}(t) \tag{12}$$

$$V_{\text{out}}(t) = C\mathbf{i}(t), \tag{13}$$

where

$$\mathbf{i} = [i_1,\, i_2,\, i_3]^T, \quad \mathbf{u} = [V_{\text{in}},\, 0,\, 0]^T, \tag{14}$$

$$A = \begin{bmatrix} R_3 + R_4 & -R_4 & 0 \\ R_4 & -(R_1 + R_4) & 0 \\ 0 & 0 & -R_2 \end{bmatrix}, \tag{15}$$

$$B = \begin{bmatrix} 1/C_3 & 0 & -1/C_3 \\ 0 & -(1/C_1 + 1/C_2) & 1/C_2 \\ 1/C_3 & 1/C_2 & -(1/C_2 + 1/C_3) \end{bmatrix}, \tag{16}$$

and $C = [mVR_3,\, hVR_1,\, R_2]$. Parameter h corresponds to the high control and also ranges from $[0, 1]$. As before, equation (12) is discretized using the trapezoidal rule, which after some algebraic manipulations, gives us the discrete-time expression

$$\mathbf{i}^n = \left(I + \frac{A^{-1}BT_s}{2}\right)^{-1} \left[\left(I - \frac{A^{-1}BT_s}{2}\right)\mathbf{i}^{n-1} + A^{-1}\left(\mathbf{u}^n - \mathbf{u}^{n-1}\right)\right], \tag{17}$$

where I is the identity matrix. The output to the tone stack is then computed through (13) as $V_{\text{out}}^n = C\,\mathbf{i}^n$.

3.1.1. *Training and Results*

In the same way as with the simple first-order RC filter, the proposed digital FMV tone stack can be trained to fit a set of input–output measurements from a real device. As before, we introduce scaling factors for each of the circuit components in the circuit, i.e. G_{VR1}, G_{VR2}, G_{VR3}, G_{R4}, G_{C1}, G_{C2} and G_{C3}, and redefine matrices A, B and C accordingly. Reference recordings were performed on the tone stack section of a Marshall-style guitar amplifier at six different user parameter settings. This was done without having to physically alter the real circuit by simply attaching crocodile clips to the leg of R_4 that connects with C_1, and to the wiper of VR_1. Recordings were performed following the procedure described in Sec. 2.2.1. This data was used to train an FMV tone stack model initialized with the component values published in [35] and shown in the second column of Table 1.

To account for the logarithmic curve of potentiometer VR_2 a small two-layer network of width one with a $\tanh$ activation was used to map the measured knob position to model parameter l. The input–output relationship of this small network is given by

$$f_{\text{v}}(x) = w_1 \tanh(w_2 x + b_2) + b_1, \tag{18}$$

where w_{1-2} and b_{1-2} are the weights and biases in each layer respectively. This small network has the same form as the general potentiometer mapping function proposed by Holmes and van Walstijn in [36]. Therefore, we initialized its weights and biases with the values proposed in said work (also shown in Table 1) and

	Value		
Name (λ)	Initial	Learned	G_λ
VR_1	250 kΩ	312 kΩ	1.2498
VR_2	1 MΩ	616 kΩ	0.6164
VR_3	25 kΩ	32 kΩ	1.2836
R_4	56 kΩ	29 kΩ	0.9081
C_1	250 pF	327.5 pF	1.3102
C_2	20 nF	17.3 nF	0.8652
$C3$	20 nF	16.8 nF	0.8408
w_1	0.566	0.5036	–
w_2	4.400	4.2547	–
b_1	−3.380	−3.3351	–
b_2	0.564	0.5016	–

Table 1: Intial and learned values for the FMV Tone Stack Model.

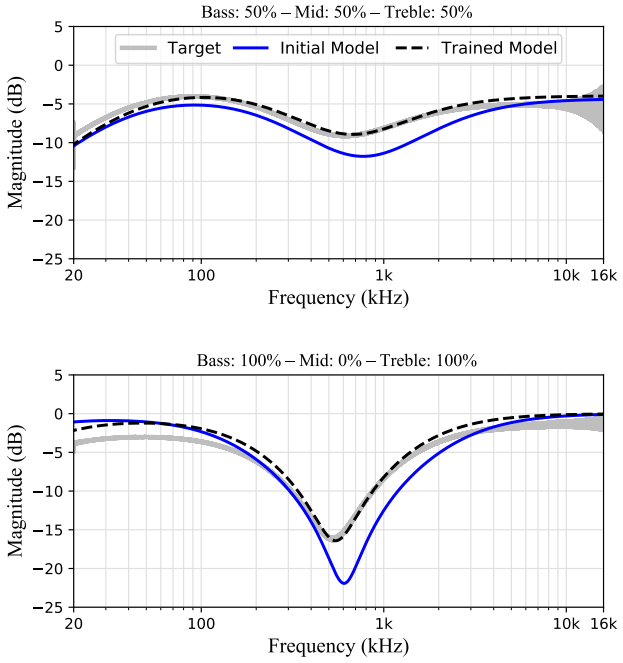

Figure 6: Magnitude response of the FMV tone stack for two different parameter settings before and after training, plotted against their corresponding target responses.

trained it in conjunction with the rest of the model parameters.

The tone stack model was trained for 100 epochs on the measured dataset. Figure 6 shows the magnitude response of the model before and after training for two different parameter settings. As shown by these results, the trained model achieves a better approximation of the target response w.r.t. its initial state. Learned parameters and scaling factors can be found in columns 3 and 4 of Table 1. This example demonstrates the ability of differentiable circuit models to handle user-facing parameters. An additional advantage of this differentiable model is that it can be easily integrated into a larger deep learning-based guitar amplifier modeling system, for instance, one where either STNs [24] or RNNs [21] are used to model the preamp and power amp sections.

3.2. Case Study: Ibanez TS-808 Overdrive Stage

As a final case study, we consider the nonlinear circuit shown in Figure 7, which corresponds to the overdrive stage of the Ibanez TS-808 guitar pedal. This circuit is configured as a non-inverting bandpass filter with clipping diodes in the feedback loop. In order to derive its governing ODE, we make use of the equation

$$I_\mathrm{D} = I_\mathrm{s}\left(e^{\frac{V_\mathrm{D}}{\eta V_\mathrm{T}}} - 1\right), \tag{19}$$

which describes the current-voltage relationship of a p–n junction. Parameters I_D and V_D are the current through and voltage across the diode, respectively, I_s is the reverse bias saturation current, V_T is the thermal voltage and η is the ideality factor of the diode, which typically ranges between 1 and 2 for a silicone diode.

Applying KVL and KCL to the circuit and using the substitution $V = V_\mathrm{out} - V_\mathrm{in}$ gives us the system equations

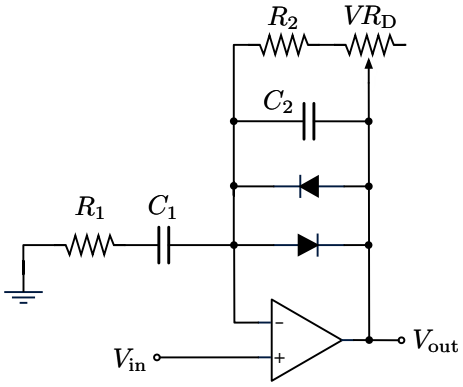

Figure 7: Schematic for the TS-808 Overdrive Stage

$$\dot{V}_\mathrm{x} = \dot{V}_\mathrm{in} - \frac{V_\mathrm{x}}{R_1 C_1} \qquad \text{and}$$

$$\dot{V} = \frac{V_\mathrm{x}}{R_1 C_2} - \frac{V}{R_\mathrm{F} C_2} - \frac{1}{C_1}\left(I_{\mathrm{s}_1} e^{\beta_1 V} - I_{\mathrm{s}_2} e^{-\beta_2 V}\right), \tag{20}$$

where $R_\mathrm{F} = R_2 + \sigma V R_\mathrm{D}$ for $\sigma \in [0, 1]$ and $\beta = 1/(\eta V_\mathrm{T})$. The user-facing parameter σ controls the gain level of the amplifier and hence the amount of overdrive. In [3], this same circuit is analyzed under the assumption that both diodes are identical. This leads to a more compact representation through the use of an exponential identity. In this work, we forego this simplification and instead treat both diodes separately, as this will allow us to learn any asymmetries in the system.

Discretizing the circuit equations using trapezoidal integration gives the us the discrete-time recursions

$$V_\mathrm{x}^n = \frac{V_\mathrm{in}^n - V_\mathrm{in}^{n-1} + (1-\rho)V_\mathrm{x}^{n-1}}{1+\rho}, \tag{21}$$

and

$$V^n = (1-\phi)\,V^{n-1} - \phi V^n - \frac{T_\mathrm{s}}{2C_2}\left[I_{\mathrm{s}_1}\left(e^{\beta_1 V^n} + e^{\beta_1 V^{n-1}}\right) - I_{\mathrm{s}_2}\left(e^{-\beta_2 V^n} + e^{-\beta_2 V^{n-1}}\right)\right] + \delta V_\mathrm{x}^n, \tag{22}$$

where

$$\rho = \frac{T_\mathrm{s}}{2R_1 C_1}, \quad \phi = \frac{T_\mathrm{s}}{2R_\mathrm{F} C_2} \quad \text{and} \quad \delta = \frac{T_\mathrm{s}}{2R_1 C_2}. \tag{23}$$

The output to the system is then given as $V_\mathrm{out}^n = V^n + V_\mathrm{in}^n$. Since (22) is implicit, it has to be resolved using an iterative root finding method such as Newton-Raphson.

3.2.1. *Training and Results*

The derived discrete-time model of the Tube Screamer overdrive circuit was implemented in PyTorch and, as in the previous examples, scaling factors were used for each of the nine model parameters (shown in the first column of Table 2). The Newton-Raphson method was used to iteratively resolve the implicit relation in the

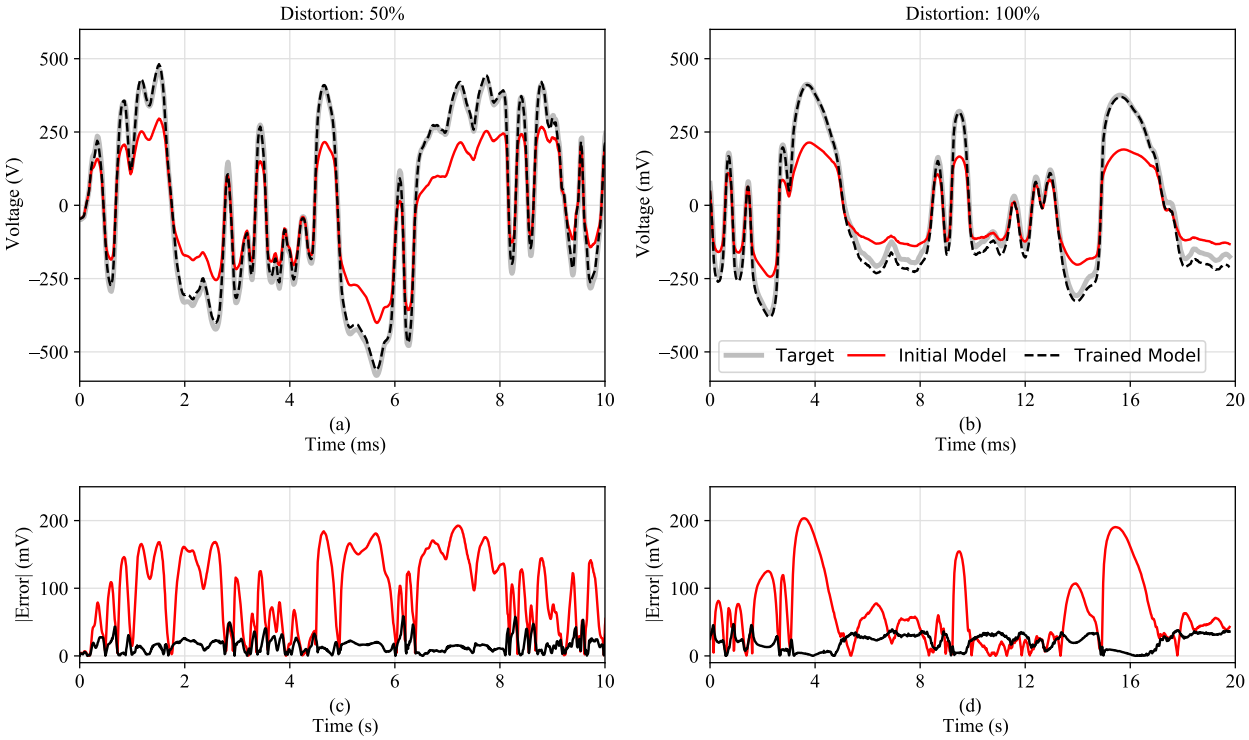

Figure 8: (a)–(b) Time domain response of the TS-808 model before and after training for two different parameter settings, plotted against their corresponding target responses, and (c)–(d) the absolute value of the error between them.

model. Training data was generated by collecting input and output measurements from a real TS-808 unit at three different overdrive levels, namely 0, 50 and 100%. Measurements were performed by attaching crocodile clips at the positive and output terminals of the op-amp, and using the setup described in Sec. 2.2.1. A 2-minute training signal composed by various segments of guitar, bass and drum recordings was used. The measurement setup was calibrated so that the maximum peak voltage seen at the input of the overdrive circuit was 200 mV, the same value used in [29]. A second dataset calibrated to have a peak voltage level of 500 mV was generated for validation. Model parameters were initialized according to the component values presented in [3][2], with the exception of I_s and β, which where initialized to 2 nA and $1/25$ mV ($\eta = 1$)[3], respectively. Like in the FMV tone stack, a 2-layer network with a $\tanh$ activation was used to account for the logarithmic relationship of VR_D.

	Value		
Name (λ)	Initial	Learned	G_λ
R_1	4.7 kΩ	4.49 kΩ.	0.9554
R_2	51 kΩ	50.3 kΩ	0.9867
VR_D	500 kΩ	513.1 kΩ	1.0262
C_1	47 nF	51.7 nF	1.1001
C_2	50 pF	52.6 pF	1.0520
I_{s_1}	2 nA	1.87 nA	0.9346
I_{s_2}	2 nA	1.88 nA	0.9423
β_1	40.0	19.99	0.4999
β_2	40.0	22.888	0.5722
w_1	0.566	0.4663	–
w_2	4.400	4.4138	–
b_1	−3.380	−3.3668	–
b_2	0.564	0.4643	–

Table 2: Intial and learned parameters for the TS-808 Overdrive Stage model.

The proposed model was trained for 100 epochs using 1024-sample sequences and batch size 512. The MSE values of the model before and after training were recorded as 6.4×10^{-3} and 1.5×10^{-3}, respectively. Figures 8(a) and (b) show the time-domain response of the Tube Screamer model at 50% and 100% overdrive for a segment of the validation dataset before and after training, plotted against the recorded waveforms. It is worth noting that due to the topology of the circuit, the output waveform is composed of a mix between the input and clipped signals. Therefore, the processed waveforms do not appear clipped in the traditional sense. These results show the initial model exhibited a good match during lower level passages but not throughout the entire signal range, hinting, for instance, that the clipping threshold of the diodes is wrong. This behavior is also exhibited in Figs. 8(c) and (d), which show the absolute value of the difference (in Volts) between the models and the reference measurements. The trained

[2] In [3], the diode ideality factor η is omitted from the model.

[3] According to several online sources, such as [37], the TS-808 circuit uses MA150 silicone diodes. Since the parameters for this diode model are not readily available, we initialize them to standard values and learn them instead from the measured data.

model, on the other hand, accurately follows the target waveform at different amplitude levels and provides a better match of the target response, as indicated by the reduced and more uniform level of error. Learned parameters and scaling factors can be found in columns 3 and 4 of Table 2. These results show the model was able to learn the asymmetric behavior of the diodes. As in the case of the FMV tone stack, the model can be integrated into a larger deep learning structure for end-to-end training.

The results from this nonlinear example validate the performance of the proposed method for the case of nonlinear circuits. Of particular interest in this example was the presence of an iterative root solver, which PyTorch was able to handle without any particular difficulties apart from resulting in considerably longer training times than those in the previous examples. However, it should be noted that the proposed method is independent of the numerical method used to discretize the circuit. As with any white-box VA model, the proposed method is limited by the accuracy of the discretization method employed. However, given the nature of the training process, the system will try to compensate for the effects of the discretization method by altering systems parameters. This may be an advantage in some cases, but it is worth noting this effect (along with the non-uniqueness of valid parameter values) means that the learnt parameter values may not exactly reflect those present in the real circuit. This could be partially addressed by making the discretization method parametric, and allowing it to be optimised alongside the system parameters [38]. An evaluation of the performance of different numerical methods in a differentiable context is a broad topic that falls outside the scope of this study and is therefore not examined any further.

4. CONCLUSIONS

In this work we introduced the concept of differentiable white-box VA modeling, a technique that can be used to learn the parameters of a circuit model from measurements of real devices. The proposed technique uses backpropagation and can be applied to both linear and nonlinear circuits in cases where components are either wrong or not readily available. Three case studies were presented, a fixed RC lowpass filter, the parametric tone section of a guitar amplifier and a nonlinear guitar overdrive circuit. We show how the derived models are able to approximate the response of the measured devices by learning optimal component values. The proposed method is independent of the numerical method used to discretize the ODEs of the circuit, which means a tailor-made scheme that best fits the requirements of the target implementation (e.g. in terms of CPU usage) can be used. The proposed differentiable models can be integrated into larger deep learning systems for end-to-end training.

5. ACKNOWLEDGEMENTS

We would like to thank Kurt J. Werner for the helpful comments and suggestions during the final preparations of this paper.

6. REFERENCES

[1] T. Stilson and J. Smith, "Alias-free digital synthesis of classic analog waveforms," in *Proc. Int. Computer Music Conf.*, Hong Kong, Aug. 1996, pp. 332–335.

[2] V. Välimäki, J. Pekonen, and J. Nam, "Perceptually informed synthesis of bandlimited classical waveforms using integrated polynomial interpolation," *J. Acoust. Soc. Am.*, vol. 131, no. 1, pp. 974–986, Jan. 2012.

[3] D. T. Yeh, J. S. Abel, and J. O. Smith, "Simplified, physically-informed models of distortion and overdrive guitar effects pedals," in *Proc. 10th Int. Conf. Digital Audio Effects (DAFx-07)*, Sept. 2007, pp. 189–196.

[4] D. T. Yeh, J. S. Abel, and J. O. Smith, "Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinary differential equations," in *Proc. 10th Int. Conf. Digital Audio Effects (DAFx-07)*, Bordeaux, France, Sept. 2007, pp. 197–204.

[5] R. Cauduro Dias de Paiva, J. Pakarinen, and V. Välimäki, "Reduced-complexity modeling of high-order nonlinear audio systems using swept-sine and principal component analysis," in *Proc. 125th AES Conv.*, Helsinki, Finland, March 2012.

[6] W. R. Dunkel, M. Rest, K. J. Werner, M. J. Olsen, and J. O. Smith, "The Fender Bassman 5F6–A family of preamplifier circuits–A wave digital filter case study," in *Proc. 19th Int. Conf. Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, Sept. 2016, pp. 263–270.

[7] D. T. Yeh, J. S. Abel, and J. O. Smith, "Automated physical modeling of nonlinear audio circuits for real-time audio effects—Part I: Theoretical development," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 18, no. 4, pp. 728–737, May 2010.

[8] M. Holters and U. Zölzer, "A generalized method for the derivation of non-linear state-space models from circuit schematics," in *Proc. European Signal Processing Conference (EUSIPCO-15)*, 2015, pp. 1073–1077.

[9] K. J. Werner, *Virtual analog modeling of audio circuitry using wave digital filters*, Ph.D. thesis, Stanford University, Stanford, CA, USA, Dec. 2016.

[10] G. De Sanctis and A. Sarti, "Virtual analog modeling in the wave-digital domain," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 715–727, May 2010.

[11] A. Falaize and T. Hélie, "Passive guaranteed simulation of analog audio circuits: A port-Hamiltonian approach," *Appl. Sci.*, vol. 6, no. 10, Apr. 2016.

[12] T. Hélie, "Volterra series and state transformation for real-time simulations of audio circuits including saturations: Application to the Moog ladder filter," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 747–759, May 2010.

[13] M. J. Kemp, "Analysis and simulation of non-linear audio processes using finite impulse responses derived at multiple impulse amplitudes," in *Proc. Audio Eng. Soc. 106th Conv.*, Munich, Germany, May 1999.

[14] A. Primavera, S. Cecchi, L. Romoli, M. Gasparini, and F. Piazza, "Approximation of dynamic convolution exploiting principal component analysis: Objective and subjective quality evaluation," in *Proc. Audio Eng. Soc. 133th Conv.*, San Francisco, USA, Oct. 2012.

[15] F. Eichas, S. Möller, and U. Zölzer, "Block-oriented modeling of distortion audio effects using iterative minimization,"

in *Proc. 18th Int. Conf. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, Dec. 2015, pp. 243–248.

[16] F. Eichas and U. Zölzer, "Black-box modeling of distortion circuits with block-oriented models," in *Proc. 19th Int. Conf. Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, Sept. 2016, pp. 39–45.

[17] F. Eichas, S. Möller, and U. Zölzer, "Block-oriented gray box modeling of guitar amplifiers," in *Proc. 20th Int. Conf. Digital Audio Effects (DAFx-17)*, Edinburgh, UK, Sept. 2017, pp. 184–191.

[18] Z. Zhang, E. Olbrych, J. Bruchalski, T. J. McCormick, and D. L. Livingston, "A vacuum-tube guitar amplifier model using long/short-term memory networks," in *Proc. IEEE SoutheastCon*, Saint Petersburg, FL, 2018.

[19] T. Schmitz and J.-J. Embrechts, "Nonlinear real-time emulation of a tube amplifier with a long short time memory neural-network," in *Proc. Audio Eng. Soc. 144th Conv.*, Milan, Italy, 2018.

[20] E.-P. Damskägg, L. Juvela, E. Thuillier, and V. Välimäki, "Deep learning for tube amplifier emulation," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Proc. (ICASSP 2019)*, Brighton, UK, 2019, pp. 471–475.

[21] A. Wright, E.-P. Damskägg, and V. Välimäki, "Real-time black-box modelling with recurrent neural networks," in *Proc. 22nd Int. Conf. on Digital Audio Effects (DAFx-19)*, Birmingham, UK, 2019.

[22] E.-P. Damskägg, L. Juvela, and V. Välimäki, "Real-time modeling of audio distortion circuits with deep learning," in *Proc. Int. Sound and Music Computing Conf. (SMC-19)*, Malaga, Spain, 2019, pp. 332–339.

[23] A. Wright and V. Välimäki, "Neural modelling of periodically modulated time-varying effects," in *Proc. 23rd Int. Conf. on Digital Audio Effects (DAFx-20)*, Vienna, Austria, 2020.

[24] J. Parker, F. Esqueda, and A. Bergner, "Modelling of nonlinear state-space systems using a deep neural network," in *Proc. 22nd Int. Conf. on Digital Audio Effects (DAFx-19)*, Birmingham, UK, 2019.

[25] M. Martinez Ramirez and J. Reiss, "End-to-end equalization with convolutional neural networks," in *Proc. 21st Int. Conf. on Digital Audio Effects (DAFx-18)*, Aveiro, Portugal, 2018, pp. 296–303.

[26] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *Proc. Int. Conf. on Learning Representations (ICLR)*, Addis Ababa, Ethiopia, 2020.

[27] B. Kuznetsov, J. D. Parker, and F. Esqueda, "Differentiable IIR filter for machine learning applications," in *Proc. 22nd Int. Conf. Digital Audio Effects (DAFx-20)*, Vienna, Austria, 2020.

[28] S. Nercessian, A. Sarroff, and K. J. Werner, "Lightweight and interpretable neural modeling of an audio distortion effect using hyperconditioned differentiable biquads," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Proc. (ICASSP 2021)*, Toronto, Canada, 2021, pp. 890–894.

[29] B. Holmes and M. van Walstijn, "Physical model parameter optimisation for calibrated emulation of the Dallas Rangemaster Treble Booster guitar pedal," in *Proc. 19th Int. Conf. Digital Audio Effects (DAFx-16)*, Brno, Czech Republic, Sept. 2016, pp. 39–45.

[30] A. Paszkeand et al., "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

[31] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, Software available from tensorflow.org.

[32] R. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *Proc. of the 32nd Int. Conf. on Neural Information Processing Systems (NeurIPS)*, 2018, pp. 6572–6583.

[33] A. Wright and V. Välimäki, "Perceptual loss function for neural modelling of audio systems," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Proc. (ICASSP 2020)*, Barcelona, Spain, 2020, pp. 251–255.

[34] J. Serra, S. Pascual, , and C. Segura, "Blow: A single-scale hyperconditioned flow for non-parallel raw-audio voice conversion," in *Proc. Conf. Neural Information Process. Syst. (NeurIPS)*, Vancouver, Canada, Dec. 2019.

[35] D. T. Yeh and J. O. Smith, "Discretization of the '59 Fender Bassman tone stack," in *Proc. 9th Int. Conf. Digital Audio Effects (DAFx-06)*, Montreal, Canada, Sept. 2006.

[36] B. Holmes and M. van Walstijn, "Potentiometer law modelling and identification for application in physics-based virtual analogue circuits," in *Proc. 22nd Int. Conf. Digital Audio Effects (DAFx-19)*, Birmingham, UK, Sept. 2019.

[37] ElectroSmash, "Tube Screamer Circuit Analysis," https://www.electrosmash.com/tube-screamer-analysis, Accessed April 10, 2021.

[38] M. Raissi, P Perdikaris, and G. Karniadakis, "Multistep neural networks for data-driven discovery of nonlinear dynamical systems," *arXiv:1801.01236*, 2018.