

MODELLING OF NONLINEAR STATE-SPACE SYSTEMS USING A DEEP NEURAL NETWORK

Julian D. Parker, Fabián Esqueda, André Bergner

Native Instruments GmbH.
Berlin, Germany

firstname.lastname@native-instruments.de

ABSTRACT

In this paper we present a new method for the pseudo black-box modelling of general continuous-time state-space systems using a discrete-time state-space system with an embedded deep neural network. Examples are given of how this method can be applied to a number of common nonlinear electronic circuits used in music technology, namely two kinds of diode-based guitar distortion circuits and the lowpass filter of the Korg MS-20 synthesizer.

1. INTRODUCTION

Virtual analog (VA) modelling is a well-established and active field of research within musical signal processing that focuses on the digital emulation of electrical or electro-mechanical systems used in music production. Previous VA research has studied a wide variety of music systems, such as analog filters and oscillators like those used in subtractive synthesis [1–5], guitar effects pedals [6–8], and guitar amplifiers [9], to name but a few examples. VA modelling can generally be separated into two broad categories, ‘white-box’ modelling and ‘black-box’ modelling.

White-box modelling, so called because it requires full knowledge of the structure of the device under study (e.g. via circuit schematics), focuses on deriving the underlying differential equations governing a system and then discretizing them to generate a numerical solution. For simple circuits this process can be performed manually, which typically allows for a tailored solution to the problem [10–12]. However, for more complicated systems this approach can quickly become unwieldy and the use of an automated general-purpose framework is generally preferred. Examples of these frameworks include state-space models [13, 14], the wave digital filter (WDF) formalism [15, 16], and port-Hamiltonian systems [17].

Black-box techniques, on the other hand, focus on measuring the system which is being modelled, and then using these measurements to provide parameters or coefficients to a standard modelling structure. Prominent forms of black-box modelling include Volterra series [18], dynamic convolution [19, 20] and Wiener-Hammerstein models [21, 22]. Some work has focused on automatically tuning a hand-designed model system for a specific class of systems, e.g. a guitar amplifier, using measurements from a specific example of such a system [23, 24]. This kind of approach is commonly named ‘grey-box’ as it requires some insight into the construction of the system.

Recent applications of Machine Learning (ML), specifically Convolutional Neural Networks (CNNs), to the topic of VA modelling. Copyright: © 2019 Julian D. Parker et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 Unported License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

elling [25] are promising but computationally expensive and limited to systems with no autonomous or higher-order nonlinear behaviour. Other recent work has examined the modelling of nonlinear time-series data by using a compound neural network structure including an autoencoder to estimate the internal state of the process and then model the evolution within this inferred space [26]. Both of these approaches do not consider measurements inside the system, and work with only input/output data. They are broadly therefore in the black-box category.

In this paper we present a new structure consisting of a deep neural network, embedded within a discrete-time state-space system. We call this structure a State Trajectory Network (STN). We show how this structure can be trained to approximate a number of the common circuits of music electronics using not just input and output measurements, but also measurements of signals within the circuits.

This paper is structured as follows. In Sec. 2 we describe the principles behind the method. In Sec. 3 we describe the proposed Artificial Neural Network (ANN) structure, and discuss both how it can be trained to fit a system, and how it can be applied to process signals. In Sec. 4 we apply the technique to a number of circuits and discuss the results. Sec.5 gives concluding remarks.

2. METHOD

The state-space (also known as phase-space) approach is a powerful mathematical formalism that can describe any system which can be characterised by a system of ODEs [27]. In continuous-time, it can be written as:

$$\dot{\mathbf{x}}(t) = f(\mathbf{u}(t), \mathbf{x}(t)) \quad (1)$$

$$\mathbf{y}(t) = g(\mathbf{u}(t), \mathbf{x}(t)) \quad (2)$$

where \mathbf{u} is the vector of inputs to the system, \mathbf{y} is the vector of outputs and \mathbf{x} are the ‘states’ of the system. This can also be written in a single equation form, by concatenating the states with the input or output:

$$\begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{pmatrix} = f_a \begin{pmatrix} \mathbf{u}(t) \\ \mathbf{x}(t) \end{pmatrix}. \quad (3)$$

The function f_a therefore completely describes the behaviour of the system. Note that in cases where the state of a system is directly taken as the output, there is no need for any \mathbf{y} variables. This is quite common in the types of systems we will be considering. The discrete-time analogue (given constant time-steps, where $n = \tau t$ with τ being the sampling interval) is given by:

$$\begin{pmatrix} \mathbf{x}_{n+1} \\ \mathbf{y}_n \end{pmatrix} = f_d \begin{pmatrix} \mathbf{u}_n \\ \mathbf{x}_n \end{pmatrix} \quad (4)$$

which allows the behaviour of the system to be calculated iteratively based on its inputs and previous states.

Much of the literature on VA modelling is concerned with deriving a set of equations in the form described by (3) which define the continuous time system of a particular circuit, and then using various methods to discretize it into the corresponding discrete-time state-space system to allow replication of its behaviour in a digital device like a computer. The discretization problem therefore becomes essentially the problem of transforming the function f into the function f_d .

2.1. Approximating f_d

If we have access to the system being modelled, we generally can also access the input and output signals $x(t)$ and $y(t)$. In many practical situations, e.g. when modelling an electrical circuit, we will also have access to the signals $x(t)$ corresponding to the states of the system. If we feed the system with test signals $x(t)$, and sample the state and output signals $x(t)$ and $y(t)$, respectively, we now have a large set of data-points describing the input and output relationship of the function f_d . If we want to replicate the behaviour of the system for arbitrary input, we can use this data to produce a new function \hat{f}_d that approximates f_d to an appropriate degree of accuracy.

This is essentially a regression task, and could be tackled via any number of standard techniques. However, it is well suited to the application of an artificial neural network (ANN) which are known to be universal approximators, i.e. they are capable of approximating any continuous function of real variables arbitrarily well [28, 29]. The system can then be simulated iteratively as usual, but using the approximated function \hat{f}_d instead of f_d derived analytically through discretization. The system can then be written as

$$\begin{matrix} x_{n+1} \\ y_n \end{matrix} = \hat{f}_d \begin{matrix} u_n \\ x_n \end{matrix} \quad (5)$$

The choice of what quantities to take as the states of the system is not as strict as it might first appear. Whilst the real states of the system are the quantities that store energy, and hence information (e.g. capacitors), other quantities in the system will be related to those states by a mapping (linear in the best case, but likely nonlinear). This means that as long as we have sufficient measured quantities compared to the number of states of the system we should be able to learn its dynamics. The formal upper bound on the number of independent measurements needed is given by the Whitney embedding theorem [30], where m is the number of states [30]. Additionally Taken's theorem [31] allows us to replace some (or all) of these measurements with time-delayed versions of another measurement. In practice, the dynamics in this transformed state-space may be more complicated, and hence it is preferable to use measurements as close as possible to the real states of the system.

3. NEURAL NETWORK ARCHITECTURE

The formulation of the problem in state-space terms is advantageous, as it means that the function we need to approximate is purely a static, i.e. memoryless, mapping. Consequently, network structures with embedded memory, such as recurrent neural networks (RNNs) [32], in particular LSTM networks [33] or

Figure 1: Proposed network structure, as it appears during training

echo state networks [34], which are standard candidates for sequence modelling, are unnecessary. The proposed method is also distinct from autoregressive modelling which predicts the next (often scalar) output based on a sequence of past observations [35], e.g. the celebrated WaveNet architecture [36]. Instead of considering only the input-output behavior of the system, we map from the state space into itself. Speaking in terms of dynamical systems, this means we are learning the piece-wise flow along the trajectory of the system [37]. The input can be considered as a parameter that influences this flow.

The core of the network is a Multilayer Perceptron (MLP), i.e. series of densely connected layers with an activation function. The number of layers and the layer width is tuned to suit the particular system being modelled, with small systems potentially needing only small networks. The activation function can be changed to fit the system, but generally saturating nonlinearities such as tanh produce good results. This intuitively makes sense, as the type of systems we are examining - electronic circuits, generally contain nonlinearities of the saturating type. Simpler activations such as Rectified Linear Units (ReLU) can be used, with the caveat that the size of the network will then generally need to be larger.

The states x_n and their values at the new time step x_{n+1} are likely to be closely related. We therefore structure this part of the network in a residual fashion using a skip-layer connection. Skip-layer connections have been successfully applied in different contexts [36, 38]. The implication of this is that the network is learning a residual of the state signals compared to their previous value, rather than the new values directly. This consistently improves training speed and accuracy in the case of the considered systems.

The proposed network structure is shown in Fig. 1. As the network is able to iteratively move along the trajectories of the learned state-space, we call this structure a State Trajectory Network (STN).

3.1. Training

In initial experiments with the structure, a Mean Squared Error (MSE) loss function was used for training:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2; \quad (6)$$

where Y and \hat{Y} are respectively the desired output and the actual output of the neural network, and n is the size of the training batch.

Using this loss function, the network structure showed some difficulty with training accurately. Investigation showed that the cause of this difficulty was large variance in the state residuals needed to fit the training data. This meant that MSE was prioritising the accuracy of the large residuals, and often leaving the smaller residuals completely incorrect. To combat this, a normalized version of MSE was used for the initial 10 epochs of the training:

$$MSE_{norm} = \frac{1}{n} \sum_{i=1}^n \frac{(Y_i - \hat{Y}_i)^2}{Y_i^2} \quad (7)$$

This seemed to provide a better foundation for further training using MSE, resulting in better overall accuracy. The Nesterov Adam (Nadam) optimizer was used, with learning rate and other meta-parameters set as recommended by Dozat [39]. Training and verification of the models was conducted using Keras [40] with TensorFlow [41] as a backend.

3.2. Application to sound synthesis and processing

Fig. 2 shows the structure of the network when used for processing or generating audio signals. The feedback connection representing the iterative calculation of the modelled system behaviour is shown. Also shown is an additional gain element used to scale the state residual calculated by the network. This element highlights another advantage of skip-level structure. Having direct access to the residuals is useful, as the residuals are entirely responsible for the time-evolution of the modelled system. Multiplying the residuals by an arbitrary gain allows us to effectively alter the time-scale of the simulated system. With some caveats, this allows the model to be run at sampling periods other than the one used to train the model. This gain can be defined as:

$$h = \frac{p}{t} = \frac{F_t}{F_p} \quad (8)$$

where p , t are the time-steps used for processing and for training respectively, and equivalently F_p and F_t are the sampling frequencies used for processing and for training.

Care must be taken when using this facility to run the system at a different sampling frequency than the one which it was trained for. Whilst the time-scale will be correctly altered, this does not make the system equivalent to one trained at the new sampling frequency. Running the system at a lower sampling frequency than used for the training raises the possibility of aliasing being introduced by elements of the learned behaviour that exceed the new Nyquist limit. Running the system at a higher sampling rate than it was trained at poses less risks, although the modelled system will potentially lack high-frequency behaviours that might have been present if the system had been trained at a higher sampling frequency. Oversampling compared to audio-rate is recommended in most situations, as the system does not utilize any specific anti-aliasing and hence will produce aliasing in the same situations as a normal nonlinear state-space system.

¹Note that the neural network output should not to be confused with y , the output of the system being modelled.

Figure 2: Proposed network structure in form used for processing signals.

4. MODELLING MEASURED SYSTEMS

In order to test the performance of the proposed method, a number of test circuits were built. A measurement signal was applied to the circuit, and the state and output signals recorded using an analog-to-digital converter (ADC). Unity gain op-amp buffers were used to isolate the measurement points from the load of the ADC. In some situations, node voltages were recorded instead of differential voltages across capacitors (i.e. the required states). In these cases, the differential voltage was recovered in post-processing by linear combination with other recorded node voltages.

For each circuit several models were trained, and then two selected for presentation here - one smaller and one larger network. All of the data referenced below is available on the accompanying website², including verification signals and model outputs.

4.1. Measurement Signal Design

Theoretically, any signal can be used to generate the measurements used to train the network. In practice, some important considerations have to be made. In contrast to other black-box modelling techniques, measurements with varying frequency input, e.g. a sine-sweep, are not strictly necessary to capture the behaviour of the system under consideration. This is because behaviour at all time-scales is captured in the learnt state trajectories. However, the chosen measurement signal does have a number of important impacts on the process.

The measurement signal defines the subset of the state space which will be sampled. In complicated systems, particular inputs could potentially be needed in order to access particular parts of the state space. A simple example of this could be a resonant filter circuit. In this case, an input signal at the resonant frequency would excite the circuit much more strongly than signals at other frequencies, potentially revealing nonlinear behaviours that only apply at high state magnitudes.

Care must also be taken to restrict the upper frequency of the measurement signal to significantly below the Nyquist frequency at the sampling rate used for the measurements. This is because the process of sampling the state signals and outputs is bandlimited. Components that exceed the Nyquist will be removed during the sampling process. Hence, harmonics produced in the real system may be lost. This means that state signals recorded in response to too high of a frequency will no longer accurately reflect the true state-space trajectory. This leads to measurements which appear to violate the constraint that each point in space space can correspond

²<https://github.com/julian-parker/DAFX-NLML>

to only one trajectory (effectively there are now further states in the bandlimiting filter of the sampling apparatus, which are not reflected in our system model).

In practice, the selection of a measurement signal is a balance between these two constraints, and can be tailored to the system under consideration. Typical input signals for the system can work well (e.g. guitar playing if modelling a distortion pedal), as can carefully specified sweeps. In this paper we used 10-second logarithmic sine-sweeps combined with low-level (-20dB) white noise. The noise was bandlimited to 20 kHz. This combined signal was then increased in amplitude linearly from zero to unity over half of the length of the measurement signal, in order to ensure the capture of sufficient data-points near the origin of the state-space. The minimum frequency of the sweep was taken to be 20 Hz, and the maximum frequency was chosen to be 10 kHz in order to cover a sufficient amount of the state-space without corrupting the data with spurious trajectories. A spectrogram of the measurement signal is shown in Fig. 3. A sampling rate of 48 kHz was used for measurement and training.

In all the examples presented here, training was done over 300 epochs, with a batch size of 256. Due to the small size of the networks under consideration, a GPU is not needed for the training process (and is in-fact slower than training on a CPU with SIMD functionality).

Figure 3: Spectrogram of sine-sweep and noise signal used to generate training data from the systems.

For time-domain verification of the results, two signals were chosen - a 500 Hz sawtooth wave, and a short passage of electric guitar. Short extracts from the sawtooth and guitar signals are shown in Fig. 4. The guitar signal contains a variety of signal levels, and the sawtooth was chosen to have a peak voltage of 0.5 V so as to stress the nonlinear behaviours in the circuits. Verification was primarily conducted in the time-domain, but a 1 kHz sinusoid with a peak voltage of 1 V was also used to check frequency-domain behaviour of the models.

Figure 4: Extracts of the signals used for verification of the modelled circuits

Figure 5: Schematic for the first-order diode clipper.

4.2. First-order diode clipper

The single-capacitor diode clipper, shown in Fig. 5, is a common object of examination in the VLSI literature, starting from the work of Yeh [10]. For small input signals, (i.e. peak voltages below approximately 0.6 V) the current flowing through the diodes is close to zero and the circuit reduces to a simple RC lowpass filter. For larger signal values the diodes will cause the output to saturate and the circuit becomes a nonlinear lowpass filter, with its instantaneous cutoff frequency rising as the circuit is driven harder.

The circuit looks simple, but has proven somewhat challenging for white-box techniques due to its inherent stiffness. A conventional discretization of this circuit will generally need to employ an implicit numerical scheme, with iterations necessary at each time-step in order to find the correct state value [42]. In this circuit, the output and the single state are the same quantity, i.e. the voltage across the single capacitor.

4.2.1. Training

Selecting training data is simple in this case. There is only one capacitor and hence state, and the value of this state is the output of the system. Fig. 6 shows a visualization of the training data. We can clearly see the saturating behaviour of the system as the prominent dark S-shape. Whilst the shape of the space may seem relatively simple, smaller networks had trouble fitting the curve shape exactly - even when using saturating activation functions.

Figure 6: Visualization of a subset of the state-space data used to train diode clipper model.

4.2.2. Results

The MSE for the chosen models, calculated over the entirety of the verification signals is:

Model struct.	Sawtooth	Guitar
2x128 ReLU	0:237 mV	0:276 mV
2x8 tanh	0:079 mV	0:135 mV

Fig. 7 shows extracts from the verification signals, processed with the chosen models and with the real circuit. The match appears to be good on both the sawtooth and guitar signal. Surprisingly, the larger ReLU based model shows lower accuracy. The frequency domain verification also shows a very close match to the real output, with odd and even harmonic levels matched very closely apart from a small amount of error in high-frequency even harmonics. The model output is indistinguishable from the output of the real circuit in casual listening.

the subsequent tone control. The behaviour of this circuit is similar to that of the first-order clipper, but more complex. The series capacitor introduces additional highpass filtering which in turn can cause asymmetry in the overall clipping behaviour [44].

This circuit exhibits stiffness properties similar to those of the first-order clipper when discretized using standard numerical techniques. The voltages across both capacitors were chosen as the states of the circuits. The first state, x_1 , was retrieved by measuring the node between the input resistor and series capacitor, and computing its difference with x_2 , which was measured directly at the output node.

4.3.1. Training

Again, it is valuable to visualize the measured training data to gain insight into the system. This can be seen in Fig. 9, where we see a projection of the data onto the x_1 - x_2 plane. We can still see the characteristic s-shape of the single diode-clipper. If we view the data in 3d, with x_1 providing the third dimension, we see that the s-shape is actually a manifold rotated around the x_2 axis. Training proceeded as in the single-capacitor case.

Figure 7: Results of processing the verification signals with two different trained diode-clipper models, compared to the output measured from the real circuit.

Figure 9: Visualization of state-space data used to train DS-1 model.

4.3. Second-order diode clipper (Boss DS-1)

4.3.2. Results

The MSE for the models, calculated over the entirety of the verification signals is:

Model struct.	Sawtooth	Guitar
2x128 ReLU	0:082 mV	0:191 mV
3x4 tanh	0:232 mV	2:67 mV

Figure 8: Schematic for the second-order diode clipper. Figure adapted from [43].

Again, we can see extracts from the verification signals in Fig. 10. The match is again very good, with a small advantage in accuracy from the larger ReLU based model. The frequency domain results show a very close match for odd harmonics, with even harmonics being a worse fit. The ReLU model in particular seems to produce stronger even harmonics than needed. The model output is again indistinguishable from the output of the real circuit in casual listening.

The next circuit considered in this study is the second-order diode clipper shown in Fig. 8. This circuit adds a capacitor between the input and the diodes, and is a simplified version of the core distortion section in the well-known Boss DS-1 pedal [43]. In this study we have omitted the preceding op-amp gain stage and

4.4. Sallen-Key Filter (Korg MS-20 REV2)

The last system examined is an adapted version of the lowpass filter circuit from the Korg MS-20, a classic monophonic analog

Figure 10: Results of processing the verification signals with two different trained DS-1 models, compared to the output measured from the real circuit.

vs x_2 plane. The 3-dimensional structure of the state space is already quite visible in this plot. Two main components can be seen, a saturating s-shaped manifold similar to that seen in the clipper circuits, along with a closed orbit corresponding to the self-oscillating behaviour.

Despite the more complex seeming behaviour of the circuit, training of the model networks proceeded more quickly than with the clipper circuits.

Figure 11: Visualization of state-space data used to train MS-20 Iter model.

synthesiser released by Korg in 1978. The particular version examined in this study corresponds to the second revision of the circuit, commonly known as ‘REV2’. This Iter is a Sallen-Key topology, and utilizes operational transconductance amplifiers (OTAs) as current-controlled gain elements used to set the cutoff of the Iter. An op-amp-based non-inverting amplifier with clipping diodes is used in the feedback path as a resonance control [45]. These diodes are responsible for the characteristic distorted sound of the Iter. The schematic for this circuit is shown in Fig. 13. For clarity, the cutoff control section has been omitted and ideal buffers have been used to represent the transistor buffers contained within the LM13700 OTAs [46].

This circuit was chosen for modelling as it exhibits strongly nonlinear self-oscillatory behaviour. This behaviour would not be possible to model using existing black-box techniques. For the purpose of this work, the parametric elements were set to fixed values. This meant feeding a constant control current to the OTAs to fix the cutoff, and fixing the resonance potentiometer at a particular gain. As in the DS-1 circuit, x_1 was computed by measuring the voltages at the outputs of the first OTA (IC 1) and the feedback amplifier, and computing their difference in post-processing. The second state of the circuit, x_2 , is simply the voltage difference between the output of the second OTA (IC 2) and ground. Since this node is connected to by a unit-gain buffer, it can be measured directly at the output Iter. For this circuit, the peak amplitude of the input sweep as seen by the circuit was adjusted to be

4.4.1. Training

The MS-20 Iter exhibits much more complex behaviour than the simpler clipper circuits. A plot of the training data is shown in Fig. 11, which again shows a projection of the data onto the

The MSE for the chosen models, calculated over the entirety of the verification signals is given by:

Model struct.	Sawtooth	Guitar
2x32 ReLU	2:01 mV	684 mV
3x4 tanh	1:29 mV	794 mV

The error values are somewhat higher than seen in the case of the clipper circuits, especially in the case of the guitar verification signal. As can be seen in Fig. 12, the behaviour of the circuit seems to be replicated rather well by the models with self-oscillation occurring at the correct frequency and being damped with increased input level as expected. The major difference between the measured and modelled results in the case of the guitar signal appears to be the phase of the self-oscillation. This can be explained by the fact that even very small errors in the learned state-space trajectory will accumulate to produce a phase difference when the dynamics are dominated by free self-oscillation. The phase of the oscillation isn't important perceptually in this case, so we conclude that the poor MSE numbers do not imply a poor model. Listening to the verification signals processed through the models confirms this, as they appear indistinguishable from the verification signals processed by the real circuit. The frequency-domain results also show a very close correspondence between the models and the real circuit, with small differences seen only in some higher harmonics.

4.5. Computational Complexity

We computed the number of floating point operations by counting all multiplications, additions, and nonlinear function evaluations in the network. For the activations functions we assume three operations for ReLUs (abs, add, mul) and an average of 30 operations for tanh. Assuming that the models are run at the same sampling

modelling of stateful systems outside of the virtual-analog domain, and also intend to investigate this avenue.

6. ACKNOWLEDGMENTS

Many thanks to Nikolaj Andersson for providing the guitar recording used for evaluation in this work, as well as participating in a previous very early incarnation of the research presented here.

7. 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] A. Huovilainen, "Non-linear digital implementation of the Moog ladder filter," in Proc. 7th Int. Conf. Digital Audio Effects (DAFx-04), Naples, Italy, Oct. 2004, pp. 61–64.
- [3] 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.
- [4] F. Fontana and M. Civolani, "Modeling of the EMS VCS3 voltage-controlled filter as a nonlinear filter network," IEEE Trans. Audio, Speech, Language Process., vol. 18, no. 4, pp. 760–772, Apr. 2010.
- [5] M. Rest, J. D. Parker, and K. J. Werner, "WDF modeling of a Korg MS-50 based non-linear diode bridge VCF," Proc. 20th Int. Conf. Digital Audio Effects (DAFx-17) Edinburgh, UK, Sept. 2017, pp. 145–141.
- [6] R. C. D. de Paiva, S. D'Angelo, J. Pakarinen, and V. Välimäki, "Emulation of operational amplifiers and diodes in audio distortion circuits," IEEE Trans. Circ. Syst. II: Express Briefs, vol. 59, no. 10, pp. 688–692, Oct. 2012.
- [7] F. Eichas, M. Fink, M. Holters, and U. Zölzer, "Physical modeling of the MXR Phase 90 guitar effect pedal," Proc. 17th Int. Conf. Digital Audio Effects (DAFx-14) Erlangen, Germany, Sept. 2014, pp. 153–158.
- [8] B. Holmes and M. van Walstijn, "Physical model parameter optimisation for calibrated emulation of the Dallas Rangemaster treble booster guitar pedal," Proc. 19th Int. Conf. Digital Audio Effects (DAFx-16) Brno, Czech Republic, Sept. 2016, pp. 47–54.
- [9] W. R. Dunkel, M. Rest, K. J. Werner, M. J. Olesen, and J. O. Smith III, "The Fender Bassman 5F6-A family of preamplifier circuits—A wave digital filter case study," Proc. 19th Int. Conf. Digital Audio Effects (DAFx-16) Brno, Czech Republic, Sept. 2016, pp. 263–270.
- [10] 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," Proc. 10th Int. Conf. Digital Audio Effects (DAFx-07) Bordeaux, France, Sept. 2007, pp. 197–204.
- [11] S. D'Angelo and V. Välimäki, "Generalized Moog ladder filter: Part II—explicit nonlinear model through a novel delay-free loop implementation method," IEEE/ACM Trans. Audio Speech Lang. Process., vol. 22, no. 12, pp. 1873–1883, Dec. 2014.
- [12] F. Esqueda, H. Pöntynen, J. D. Parker, and S. Bilbao, "Virtual analog models of the Lockhart and Serge wavefolder," Appl. Sci., vol. 7, no. 12, Dec. 2017.
- [13] 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.
- [14] M. Holters and U. Zölzer, "A generalized method for the derivation of non-linear state-space models from circuit schematics," Proc. European Signal Processing Conference (EUSIPCO-2015), pp. 1073–1077.
- [15] 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.
- [16] K. J. Werner, "Virtual analog modeling of audio circuitry using wave

Figure 12: Results of processing the verification signals with two different MS20 filter models, compared to the output measured from the real circuit for the same input.

frequency used for training, 92 kHz, we can then extrapolate this into a value in floating point operations per second (FLOPS):

Model	ops/sample	GFLOPS
DS-1 2x128 ReLU	34822	6.7
Clipper 2x128 ReLU	34307	6.6
MS-20 2x32 ReLU	2620	0.50
Clipper 2x8tanh	686	0.13
MS-20 3x4tanh	524	0.10
DS-1 3x4tanh	524	0.10

The values lie well within the bounds of real-time operation on a modern computer. However, the values should only be taken as a rough guideline for performance. The actual performance of the algorithm will depend strongly on the processor architecture used.

5. CONCLUSIONS

In this work we introduced a new technique for modelling the behaviour of nonlinear state-space systems, using a discrete-time state-space system with an embedded neural network. The network learns trajectories in state-space from measurements in a residual manner, and can be used to reproduce these trajectories in response to arbitrary input. We call the network structure a State Trajectory Network (STN). We showed how this structure can be used to accurately model the behaviour of a number of nonlinear circuits, using measurements from within the circuit to train the model. We also showed that the produced models are of sufficiently low computational complexity to be run for real-time audio usage.

Future work will concentrate on extending the STN to work with larger systems, and those with parametric control. We believe the STN may be an interesting structure for application to the

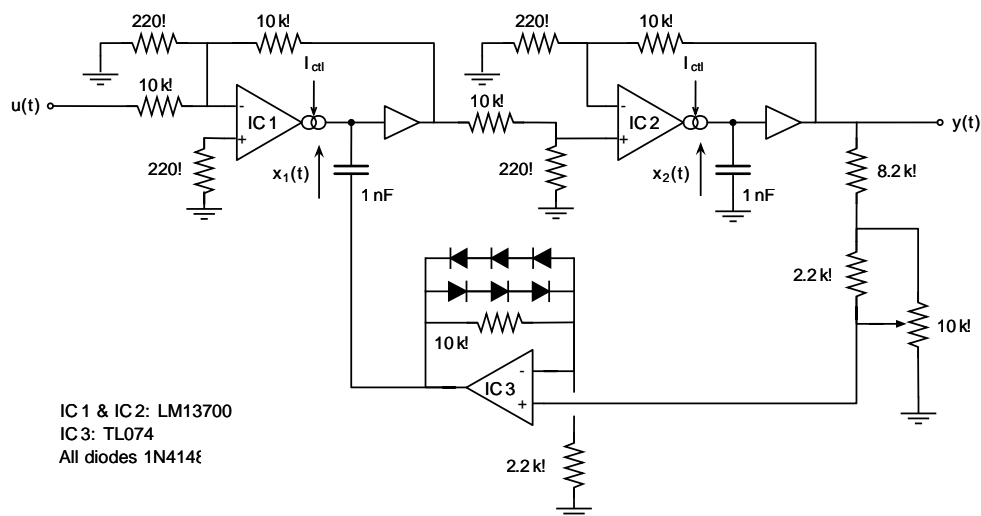


Figure 13: Simplified schematic of the Korg MS-20 filter, adapted from [46].

digital filters, Ph.D. thesis, Stanford University, Stanford, CA, USA, Dec. 2016.

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

[18] T. Hélie, “On the use of Volterra series for real-time simulations of weakly nonlinear analog audio devices: Application to the Moog ladder filter,” in *Proc. 9th Int. Conf. Digital Audio Effects (DAFx-06)*, Montreal, Canada, Sept. 2006, pp. 7–12.

[19] 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.

[20] 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.

[21] A. Novák, L. Simon, F. Kadlec, and P. Lotton, “Nonlinear system identification using exponential swept-sine signal,” *IEEE Trans. Instrum. Meas.*, vol. 59, no. 8, pp. 2220–2229, Oct. 2009.

[22] 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.

[23] 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.

[24] C. Kemper, “Musical instrument with acoustic transducer,” Patent No. US 8,796,530, 12 June 2008.

[25] E.-P. Damskågg, L. Juvola, E. Thuillier, and V. Välimäki, “Deep learning for tube amplifier emulation,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brighton, UK, May 2019.

[26] D. Masti and A. Bemporad, “Learning nonlinear state-space models using deep autoencoders,” in *Proc. 57th IEEE Conf. Dec. Control (CDC 2018)*, Miami, FL, USA, Dec. 2018, pp. 3862–3867.

[27] E. R. Scheinerman, *Invitation to dynamical systems*, Dover Publications, 1996.

[28] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989.

[29] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991.

[30] H. Whitney, “Differentiable manifolds,” *Annals of Mathematics*, pp. 645–680, 1936.

[31] F. Takens, “Detecting strange attractors in turbulence,” in *Dynamical systems and turbulence*, pp. 366–381. Springer, 1981.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, MIT press, 2016.

[33] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[34] H. Jaeger, “Adaptive nonlinear system identification with echo state networks,” in *Advances in Neural Inf. Proc. Systems*, 2003, pp. 609–616.

[35] R. J. Frank, N. Davey, and S. P. Hunt, “Time series prediction and neural networks,” *Journal of intelligent and robotic systems*, vol. 31, no. 1-3, pp. 91–103, 2001.

[36] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” *arXiv preprint arXiv:1609.03499*, 2016.

[37] J. M. Ginoux, *Differential geometry applied to dynamical systems*, vol. 66, World Scientific, 2009.

[38] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. 29th IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR 2016)*, Las Vegas, NV, USA, 2016, pp. 770–778.

[39] T. Dozat, “Incorporating Nesterov momentum into Adam,” in *Proc. 4th Int. Conf. Learn. Repres. (ICLR 2016)*, San Juan, Puerto Rico, May 2016.

[40] F. Chollet et al., “Keras,” <https://keras.io>, 2015.

[41] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., “Tensorflow: A system for large-scale machine learning,” in *Proc. 12th USENIX Symp. Oper. Sys. Des. Implement. (OSDI’16)*, 2016, pp. 265–283.

[42] F. G. Germain, “Fixed-rate modeling of audio lumped systems: A comparison between trapezoidal and implicit midpoint methods,” in *Proc. 20th Int. Conf. Digital Audio Effects (DAFx-17)*, Edinburgh, United Kingdom, Sept. 2017, pp. 168–175.

[43] Roland Corp., “Boss DS-1 Service Notes,” Dec. 1994.

[44] 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.

[45] T. E. Stinchcombe, “A study of the Korg MS10 and MS20 filters,” Aug. 2006.

[46] Korg Inc., “MS-20 Monophonic Synthesizer Service Manual,” 1978.