WAVE DIGITAL MODELING OF CIRCUITS WITH MULTIPLE ONE-PORT NONLINEARITIES BASED ON LIPSCHITZ-BOUNDED NEURAL NETWORKS

Oliviero Massi
DEIB
Politecnico di Milano
Milan, Italy
oliviero.massi@polimi.it

Edoardo Manino
Department of Computer Science
University of Manchester
Manchester, United Kingdom
edoardo.manino@manchester.ac.uk

Alberto Bernardini
DEIB
Politecnico di Milano
Milan, Italy
alberto.bernardini@polimi.it

ABSTRACT

Neural networks have found application within the Wave Digital Filters (WDFs) framework as data-driven input-output blocks for modeling single one-port or multi-port nonlinear devices in circuit systems. However, traditional neural networks lack predictable bounds for their output derivatives, essential to ensure convergence when simulating circuits with multiple nonlinear elements using fixed-point iterative methods, e.g., the Scattering Iterative Method (SIM). In this study, we address such issue by employing Lipschitz-bounded neural networks for regressing nonlinear WD scattering relations of one-port nonlinearities.

1. INTRODUCTION

Over recent years, numerous research works have focused on refining digital models of vintage analog audio circuits, like sound effect processors or synthesizers. The objective of these efforts is to faithfully capture their unique timbral characteristics through an accurate modeling of their circuit nonlinearities. Virtual Analog (VA) modeling [1] identifies the set of digital signal processing methods focused on digitally emulating analog audio circuits. These methods can be generally classified into *black-box* and *white-box* approaches. Black-box modeling approaches infer a global model of a reference circuit relying on pairs of observed input and output data [2, 3], while white-box approaches emulate the reference circuit by simulating the corresponding system of ordinary differential equations [4, 5, 6].

Among white-box techniques, Wave Digital Filters (WDFs) [7] have been widely employed in VA applications due to their inherent modularity, efficiency, and stability properties [6, 8, 9]. Introduced by A. Fettweis in the 1970s to design digital filter implementations of passive analog circuits [7], WDFs rely on a port-wise linear mapping of Kirchhoff pairs of variables (voltage and current) into pairs of wave variables (incident and reflected waves) with the introduction of a scalar free parameter per port called *port resistance*. Circuit elements and connection networks are modeled in a separate fashion by means of input-output blocks described by scattering equations. When interconnecting such input-output blocks, the introduced free parameters are set, through the so-called *adaptation* process, to eliminate as many implicit relations, also named as *delay-free-loops* (DFLs), as possible.

Unfortunately, not *all* DLFs can be removed for all circuital systems. On the one hand, linear circuit or circuits incorporating up to a single nonlinear one-port or multi-port element (characterized by an explicit mapping) can be implemented in the Wave Digital (WD) domain using stable discretization methods (e.g., trapezoidal rule) in a fully explicit fashion [10, 11]. On the other hand, this is not generally the case for circuits containing multiple nonlinearities, as not all the DFLs can be removed and iterative solvers are required. Hence, several fixed-point and Newton-Raphson (NR) methods have been developed in the literature for the solution of such circuits in the WD domain [12, 13, 14, 15, 16].

Amongst others, the Scattering Iterative Method (SIM) [15, 17] is an efficient fixed-point method which is able to solve generic circuits containing multiple nonlinear one-port elements using local one-dimensional solvers. SIM distinguishes itself from other WD fixed-point methods primarily by leveraging free parameters to enhance convergence speed [8, 15].

Lately, the widespread adoption of deep learning has prompted novel approaches for modeling nonlinear electrical components in WDFs. Specifically, the so-called hybrid WDFs combine data-driven neural models of nonlinear devices with traditional WD blocks [18, 19, 20, 11]. This integrated approach offers dual benefits. First, data-driven models enable the formulation of explicit nonlinear scattering relations. As already demonstrated by Canonical Piecewise Linear (CPWL) representations of nonlinear functions [21, 22], this eliminates the necessity of analytically deriving an explicit scattering relation for a given nonlinear component, which may not always be feasible. Second, when considering circuits with a single nonlinear component, thus characterized by explicit WD structures, these nonlinear component descriptions allow for highly computationally efficient implementations [11].

Unfortunately, the same methodology cannot be directly extended to circuits featuring multiple nonlinearities. This is due to the fact that many of the theoretical results concerning WD iterative methods are not readily transferable to the scenario involving multiple neural network-based WD blocks. Notably, conventional neural networks have arbitrarily large derivatives with respect to the input features [23], thus invalidating the assumptions required to guarantee convergence of fixed-point methods when simulating circuits with multiple one-port nonlinearities in the WD domain [15, 8, 17].

In this paper, we propose to recover the aforementioned convergence guarantees by training models with bounded derivatives. To this end, we adopt a recently-proposed class of models, named Lipschitz-bounded neural networks, which limit the magnitude of the neural model's Lipschitz constant by imposing architectural constraints [24, 25, 26]. In our experiments, we show that Lipschitz-bounded models are able to faithfully represent their analogue counterparts, while satisfying the given convergence bounds. In contrast, conventional machine learning techniques, like Sobolev training [27], produce models that consistently fail to meet the sufficient conditions for convergence.

Outlined below are our contributions:

- we are the first to model one-port nonlinear elements in WDFs with Lipschitz-bounded neural networks;
- we formally prove the connection between convergence guarantees of fixed-point methods and Lipschitz constant of data-driven blocks;
- we apply the SIM algorithm for the simulation of circuits featuring multiple neural network-based nonlinear one-port WD blocks preserving all theoretical guarantees on convergence;
- we extend the SIM convergence acceleration policy to this new data-driven scenario via a novel estimation technique for the slopes of the tangents passing through the operating points on the v-i curves of one-port nonlinearities.

The remainder of this manuscript is organized as follows. In Section 2, we provide the theoretical background on WDFs. Section 3 introduces the SIM algorithm. In Section 4, we show how to model multiple one-port nonlinearities using Lipschitz-bounded neural network-based WD blocks, and we extend the SIM algorithm to efficiently accommodate these novel data-driven blocks. In Section 5, we discuss an example of application of the proposed approach. Section 6 concludes this manuscript.

2. BACKGROUND ON WAVE DIGITAL FILTERS

The design of WDFs relies on a port-wise description of a reference analog circuit. In this approach, each pair of Kirchhoff variables, i.e., port voltage v and port current i, is replaced with a pair of voltage wave variables defined as [7]

$$a = v + Zi\,, \quad b = v - Zi\,, \tag{1}$$

where a and b are the incident wave and reflected wave, respectively, whereas $Z \neq 0$ is a free parameter known as port resistance. The inverse mapping of (1) is expressed as

$$v = \frac{a+b}{2}\,, \quad i = \frac{a-b}{2Z}\,. \tag{2}$$

2.1. Linear Elements

A broad class of linear one-port circuit elements, including resistors, resistive voltage/current sources, and dynamic elements like capacitors and inductors, when discretized using stable methods, can be described through the discrete-time Thévenin equivalent model [8]

$$v[k] = R_{\mathrm{g}}[k]i[k] + V_{\mathrm{g}}[k]\,, \tag{3}$$

where k is the sampling index, $v[k]$ is the port voltage, $i[k]$ is the port current, $R_{\mathrm{g}}[k] > 0$ is a constant or time-varying resistive parameter, and $V_{\mathrm{g}}[k]$ is a constant or time-varying voltage parameter. Using (2), the Thévenin equivalent model can be expressed in the WD domain as follows

$$b[k] = \frac{R_{\mathrm{g}}[k] - Z[k]}{R_{\mathrm{g}}[k] + Z[k]}a[k] + \frac{2Z[k]}{R_{\mathrm{g}}[k] + Z[k]}V_{\mathrm{g}}[k]\,. \tag{4}$$

The instantaneous dependence between $b[k]$ and $a[k]$ can be eliminated by setting $Z[k] = R_{\mathrm{g}}[k]$; in this case, (4) reduces to $b[k] = V_{\mathrm{g}}[k]$, and the linear one port-element is said to be *adapted* according to WDF theory [7].

2.2. Connection Networks

In the Kirchhoff domain, a N-port connection network [9] is characterized by a vector of port voltages $\mathbf{v} = [v_1, \ldots, v_N]^{\mathrm{T}}$ and a vector of port currents $\mathbf{j} = [j_1, \ldots, j_N]^{\mathrm{T}}$. Let $\mathbf{v}_{\mathrm{t}} \in \mathbb{R}^q$ be the vector of independent port voltages and $\mathbf{j}_{\mathrm{l}} \in \mathbb{R}^r$ be the vector of independent port currents, where $q + r = N$. Then, it is possible to write

$$\mathbf{v} = \mathbf{Q}^{\mathrm{T}}\mathbf{v}_{\mathrm{t}}\,, \quad \mathbf{j} = \mathbf{B}^{\mathrm{T}}\mathbf{j}_{\mathrm{l}}\,, \tag{5}$$

where $\mathbf{Q}$ is the $q \times N$ fundamental cut-set matrix and $\mathbf{B}$ is the $r \times N$ fundamental loop matrix [9]. Moreover, the orthogonality property $\mathbf{B}\mathbf{Q}^{\mathrm{T}} = \mathbf{0}$ holds true.

The WD realization of connection networks is a N-port scattering junction characterized by the wave variables

$$\mathbf{a}_{\mathrm{J}} = \mathbf{v} + \mathbf{Z}\mathbf{j}\,, \quad \mathbf{b}_{\mathrm{J}} = \mathbf{v} - \mathbf{Z}\mathbf{j}\,, \tag{6}$$

where $\mathbf{a}_{\mathrm{J}} = [a_{\mathrm{J}1}, \ldots, a_{\mathrm{J}N}]^{\mathrm{T}}$ is the vector of waves incident to the junction, $\mathbf{b}_{\mathrm{J}} = [b_{\mathrm{J}1}, \ldots, b_{\mathrm{J}N}]^{\mathrm{T}}$ is the vector of waves reflected by the junction, while $\mathbf{Z} = \mathrm{diag}[Z_1, \ldots, Z_N]$ is a diagonal matrix having port resistances as diagonal entries.

The scattering relation between $\mathbf{a}_{\mathrm{J}}$ and $\mathbf{b}_{\mathrm{J}}$ is given by $\mathbf{b}_{\mathrm{J}} = \mathbf{S}\mathbf{a}_{\mathrm{J}}$, where $\mathbf{S}$ is an $N \times N$ scattering matrix that can be computed using either of the two following equivalent equations [9]

$$\mathbf{S} = 2\mathbf{Q}^{\mathrm{T}}(\mathbf{Q}\mathbf{Z}^{-1}\mathbf{Q}^{\mathrm{T}})^{-1}\mathbf{Q}\mathbf{Z}^{-1} - \mathbf{I}\,, \tag{7}$$

$$\mathbf{S} = \mathbf{I} - 2\mathbf{Z}\mathbf{B}^{\mathrm{T}}(\mathbf{B}\mathbf{Z}\mathbf{B}^{\mathrm{T}})^{-1}\mathbf{B}\,, \tag{8}$$

with $\mathbf{I}$ being the $N \times N$ identity matrix.

2.3. Nonlinear Elements

Let us consider a generic nonlinear one-port element characterized in the Kirchhoff domain by an implicit nonlinear constitutive equation

$$h(v, i) = 0\,. \tag{9}$$

Substituting (2) into (9) results in a corresponding scattering relation in the WD domain which may not be expressible in explicit form

$$b = f(a, Z)\,, \tag{10}$$

where f is a nonlinear scalar function that depends both on the incident wave a and on the free parameter Z. To simplify notation, we omit explicit mention of the dependence on Z in the remainder of the paper. In this case, the resulting implicit scattering equation can be implemented in different ways, e.g., using a CPWL approximation [21], or a local one-dimensional NR solver [15]. Unlike most linear one-port elements, nonlinear elements cannot be adapted [7]. Nevertheless, even if the elimination of the instantaneous dependence of b on a is not possible, it can be "minimized" by dynamically adjusting the port resistance Z to align with the slope of the tangent at the current operating point on the v-i characteristic [15, 8].

As a relevant example of a nonlinear one-port element that we will use in our experiments in Sec. 5, let us consider the *extended Schockley diode model* discussed in [8], whose implicit constitutive equation is

$$h(v, i) = I_{\mathrm{s}}\left(\exp\left(\frac{v - R_{\mathrm{s}}i}{\eta V_{\mathrm{t}}}\right) - 1\right) + \frac{v - R_{\mathrm{s}}i}{R_{\mathrm{p}}} - i = 0\,, \tag{11}$$

where I_{s} is the saturation current, η is the ideality factor, V_{t} is the thermal voltage, while R_{s} and R_{p} are the series and shunt resistances of the p-n junction. A possible WD realization of such a model is the explicit scattering relation based on the Wright ω function discussed in [17].

3. SCATTERING ITERATIVE METHOD

The Scattering Iterative Method (SIM) [15] represents a WD iterative method designed for simulating circuits with multiple nonlinearities which has been previously used for the emulation of nonlinear audio circuits [8, 17].

Let us assume that the circuit is modeled in the WD domain using a single multi-port scattering junction to which all the elements are connected. We denote $\mathbf{a} = [a_1, \ldots, a_N]^{\mathrm{T}}$ as the vector of waves incident to the elements (reflected by the junction) and $\mathbf{b} = [b_1, \ldots, b_N]^{\mathrm{T}}$ as the vector of waves reflected by the elements (incident to the junction). Consequently, the vector of port voltages $\mathbf{v}$ is the same of Sec. 2.2, while the vector of port currents $\mathbf{i}$ is defined as $\mathbf{i} = -\mathbf{j}$, yielding $\mathbf{a} = \mathbf{b}_{\mathrm{J}}$ and $\mathbf{b} = \mathbf{a}_{\mathrm{J}}$.

At each sampling step k of the discrete-time simulation, SIM iteratively performs the following four stages:

Initialization The free parameters $Z_1[k], \ldots, Z_N[k]$ are set as close as possible to the tangent slope at the current working point on the v-i characteristic. For linear elements, this can be achieved by employing the adaptation conditions of traditional WDFs [7]. For nonlinear one-port elements, optimal slopes can be only estimated from the values of nth port current and port voltage at the operating point of the previous sample. Then, the updated $\mathbf{Z}[k]$ is used to update $\mathbf{S}[k]$ using (7) or (8).

Local Scattering Stage The waves $b_n^{(\gamma)}[k]$ reflected by adapted linear elements are computed as $b_n^{(\gamma)}[k] = V_{\mathrm{g}n}[k]$. When they are reflected by nonlinear one-port elements instead, they are determined as

$$b_n^{(\gamma)}[k] = f_n(a_n^{(\gamma-1)}[k]) , \tag{12}$$

where γ is the fixed-point index, and the generic nonlinear function $f_n(\cdot)$ is the WD mapping, e.g., (10), employed to model the element connected at port n.

Global Scattering Stage At each SIM iteration γ, the vector of waves incident to the elements $\mathbf{a}^{(\gamma)}[k]$ is computed as

$$\mathbf{a}^{(\gamma)}[k] = \mathbf{S}[k]\mathbf{b}^{(\gamma)}[k] . \tag{13}$$

Convergence Check Local Scattering Stage and Global Scattering Stage are iterated until the convergence condition

$$\|\mathbf{v}^{(\gamma)}[k] - \mathbf{v}^{(\gamma-1)}[k]\|_2 \leq \epsilon_{\mathrm{SIM}}$$

is met, where $\mathbf{v}^{(\gamma)}[k] = (\mathbf{a}^{(\gamma)}[k] + \mathbf{b}^{(\gamma)}[k])/2$, while ϵ_{SIM} is a small threshold, e.g., $\epsilon_{\mathrm{SIM}} = 10^{-5}$.

3.1. Considerations on SIM Convergence

The SIM update formula at each fixed-point iteration can be expressed by combining (12) and (13) as

$$\mathbf{a}^{(\gamma)} = \mathbf{Sf}\left(\mathbf{a}^{(\gamma-1)}\right) , \tag{14}$$

where the sampling index k is omitted for readability and $\mathbf{f}(\mathbf{a}) = [f_1(a_1), \ldots, f_N(a_N)]^{\mathrm{T}}$ is the vector of linear and nonlinear scattering functions related to the one-port elements.

According to (14), as extensively discussed in [17], a sufficient condition for SIM convergence is for the mapping $\mathbf{Sf}(\mathbf{a})$ to be contractive. This condition can be reformulated in terms of the spectral radius operator $\mathrm{srad}\,(\cdot)$, i.e., the largest eigenvalue in absolute value of the matrix in the argument, as

$$\mathrm{srad}\,(\mathbf{SJ_f}(\mathbf{a})) < 1 \quad \forall \mathbf{a} \in \mathbb{R}^N , \tag{15}$$

where $\mathbf{J_f}(\mathbf{a}) = \mathrm{diag}[f_1'(a_1), \ldots, f_N'(a_N)]$ is the Jacobian matrix of $\mathbf{f}(\mathbf{a})$. If the circuit is characterized by a lossless reciprocal connection network [17], it can be demonstrated that

$$\mathrm{srad}\,(\mathbf{SJ_f}(\mathbf{a})) \leq \mathrm{srad}\,(\mathbf{J_f}(\mathbf{a})) , \tag{16}$$

which reduces the convergence sufficient condition to

$$\mathrm{srad}\,(\mathbf{J_f}(\mathbf{a})) < 1 . \tag{17}$$

In the literature, it has been shown that the condition in (17) holds true assuming that all Z_n parameters are positive and each circuit element is characterized by a monotonically increasing v-i characteristic [15, 17]. Unfortunately, the same convergence considerations cannot be readily applied to scenarios where one-port nonlinearities are modeled by means of neural network-based WD blocks.

4. MODELING MULTIPLE ONE-PORT NONLINEARITIES USING NEURAL NETWORKS

The behavior of nonlinear electrical components can be characterized in a data-driven fashion based on measurement or simulation data. Once the Kirchhoff v-i characteristic of the one-port element has been acquired, it can be transformed into a WD domain dataset using (1) and a comprehensive set of values for the port resistance Z. Then, expressing the explicit WD scattering relation (10) for a nonlinear one-port element can be recast into a regression problem:

$$\hat{b} = \tilde{f}\,(a, \varrho(Z); \boldsymbol{\theta}_{\mathrm{WD}}) . \tag{18}$$

Here, $\tilde{f}$ denotes the function approximation provided by a suitable neural network architecture [28, 29, 30] whose parameters $\boldsymbol{\theta}_{\mathrm{WD}}$ are determined by solving the minimization problem

$$\boldsymbol{\theta}^{\star}_{\mathrm{WD}} = \arg\min_{\boldsymbol{\theta}_{\mathrm{WD}}} \mathcal{L}(b, \hat{b}) , \tag{19}$$

where $\mathcal{L}(b, \hat{b})$ is some regression loss function. As the port resistance parameter Z generally spans several orders of magnitude, the compression function ϱ is introduced to restrict its range of variation [18]. Assuming that $Z > 0$, we choose $\varrho(\cdot) = \ln(\cdot)$.

4.1. Convergence Conditions and Lipschitzness

As explained in Sec. 3.1, the convergence of a fixed-point method like SIM is guaranteed whenever (17) is satisfied. Here, we show that an explicit bound on the Lipschitz constant of the approximating function $\tilde{f}$ is sufficient to satisfy said convergence condition.

Definition 1. *A function* $\mathbf{g} : \mathbb{R}^d \to \mathbb{R}^e$ *is called Lipschitz continuous in norm* p *if there exists a constant* c *such that*

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d,\ \|\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y})\|_p \leq c\|\mathbf{x} - \mathbf{y}\|_p . \tag{20}$$

The smallest c *for which* (20) *is true is called the Lipschitz constant of* $\mathbf{g}$*, in short* $\mathrm{Lip}_p(\mathbf{g}) = c$.

Theorem 1. *Assume $g : \mathbb{R}^d \rightarrow \mathbb{R}$ is an arbitrary function over the reals such that its Lipschitz constant in p-norm is bounded by $\mathrm{Lip}_p(g) < 1$. Then, each partial derivative of g is also bounded by $\left|\frac{\partial g(\mathbf{x})}{\partial x_\imath}\right| < 1$ for all $\mathbf{x} = [x_1, \ldots, x_\imath, \ldots, x_d]^{\mathrm{T}}$.*

Proof. Assume that $c < 1$ in Definition 1 as per the statement of the theorem. Let $\mathbf{y} = \mathbf{x} + \epsilon_\imath$ vary from $\mathbf{x}$ only along dimension $\imath$. Then, we have

$$\frac{||g(\mathbf{x}) - g(\mathbf{x} + \epsilon_\imath)||_p}{||\epsilon_\imath||_p} < 1\,.$$

We can notice that both norms are taken over vectors of size one, thus reducing them to the absolute value for any p:

$$\frac{|g(\mathbf{x}) - g(\mathbf{x} + \epsilon_\imath)|}{|\epsilon_\imath|} < 1\,.$$

Finally, by taking the limit of the left-hand side for $\epsilon_\imath \rightarrow 0$, we recover the definition of partial derivative, and thus $\left|\frac{\partial g(\mathbf{x})}{\partial x_\imath}\right| < 1$. □

As per our discussion around (10), our data-driven nonlinear WD blocks are functions $\tilde{f}$ of the two variables a and Z. At the same time, the convergence conditions in Sec. 3.1 can be satisfied as long as $\left|\frac{\partial \tilde{f}(a)}{\partial a}\right| < 1$. In this sense, having the condition $\mathrm{Lip}_p(\tilde{f}) < 1$ is a stronger guarantee than needed, as it puts limits on $\left|\frac{\partial \tilde{f}(a)}{\partial Z}\right|$ too. However, the input variable Z can undergo any arbitrary rescaling, as shown in (18). As such, to avoid unnecessary constraints during training, it suffices to ensure that the scaled and zero centered data $\varrho(Z)$ have a slope much less than one.

4.2. Lipschitz-Bounded Neural Networks

Lipschitz-bounded neural networks are a class of models that aim at solving the minimization problem (19), while satisfying a given upper bound on the Lipschitz constant of $\tilde{f}$. More formally, we have

$$\boldsymbol{\theta}^\star_{\mathrm{WD}} = \arg\min_{\boldsymbol{\theta}_{\mathrm{WD}}} \mathcal{L}(b, \hat{b}) \quad \text{s.t.} \quad \mathrm{Lip}_p(\tilde{f}) \leq c\,.$$

While originally introduced to address the issue of adversarial examples in image classification models [31], Lipschitz-bounded neural networks have found applications in differential privacy [32], stable control [33], and reinforcement learning [34]. Overall, they work by constraining the Lipschitz constant of each neural layer via careful architectural choices.

A standard fully-connected layer ℓ is defined as

$$\mathbf{z} = \sigma(\mathbf{W}\mathbf{u} + \boldsymbol{\beta})\,, \tag{21}$$

where σ is a proper activation function, while $\mathbf{W}$ and $\boldsymbol{\beta}$ constitute a trainable weight matrix and bias vector, respectively. With respect to (21), the majority of available Lipschitz-bounded architectures focus on achieving $\mathrm{Lip}_2(\ell) \leq 1$ in Euclidean norm. Note that the specific p-norm does not prevent our application to WDF as per Theorem 1. Additionally, we can easily achieve $\mathrm{Lip}_2(\ell) \leq c$ by rescaling the output of ℓ by any arbitrary constant c. Hereafter, we introduce the neural architecture we use in our comparative experiments in Sec. 5:

Cayley-Orthogonal Layers (CAY) A possible reparametrization of the standard fully-connected layer (21) relies on an orthogonal weight matrix $\mathbf{D}$

$$\mathbf{z} = \sigma(\mathbf{D}\mathbf{u} + \boldsymbol{\beta})\,, \tag{22}$$

for any activation with slope $\sigma'(\cdot) \in [-1, 1]$. This can be achieved through the *Cayley Transform* [24], which maps a skew-symmetric weight matrix $\mathbf{W}$ into an orthogonal matrix $\mathbf{D}$ using the relation

$$\mathbf{D} = (\mathbf{I} - \mathbf{W})(\mathbf{I} + \mathbf{W})^{-1}\,.$$

Gradient-preserving activation functions like absolute value and group sort are recommended [35].

Hence, we implement $\tilde{f}$ with an architecture composed of such layers, ensuring that $\mathrm{Lip}_2(\ell) \leq 1$. Then, we rescale the neural network output by $c < 1$, thus enforcing $\left|\frac{\partial \tilde{f}(a)}{\partial a}\right| \leq c < 1$. This condition is sufficient to satisfy the convergence criterion in (17).

4.3. Data-driven Port Resistance Update for SIM with Neural Network-based WD blocks

Neural network-based WD blocks implementing (18) may not rely on any prior knowledge of the Kirchhoff domain constitutive equation for the modeled component (e.g., the v-i characteristic has been experimentally sampled). The lack of such information might prevent the estimation of the optimal tangent slope during the SIM Initialization Stage. However, neural network output derivatives can be computed using automatic differentiation (AD) [36]. Specifically, at each sampling step k, $\frac{\partial \tilde{f}_n(a_n)}{\partial a_n}$ can be calculated for the nth neural network-based WD block.

To establish a relation between $\frac{\partial \tilde{f}_n(a_n)}{\partial a_n}$ and the optimal tangent slope, let us rewrite (1) as [16, 17]

$$\begin{aligned} a_n &= v_n(i_n) + Z_n i_n = \phi_n(i_n)\,, \\ b_n &= v_n(i_n) - Z_n i_n = \psi_n(i_n). \end{aligned} \tag{23}$$

Assuming that $\phi_n(i_n)$ is invertible, the scattering relation of a generic element (10) can be rewritten as

$$b_n = f_n(a_n) = \psi_n(\phi_n^{-1}(a_n))\,. \tag{24}$$

The derivative of (24) with respect to a_n can be then obtained via

$$f'_n(a_n) = \frac{\psi'_n(\phi_n^{-1}(a_n))}{\phi'_n(\phi_n^{-1}(a_n))} = \frac{v'_n(i_n) - Z_n}{v'_n(i_n) + Z_n}\,, \tag{25}$$

where $v'_n(i_n)$ is the derivative of $v_n(i_n)$ with respect to i_n, which in turn corresponds to the tangent slope at the current working point. Solving (25) for $v'_n(i_n)$ yields the formula

$$v'_n(i_n) = -\frac{f'_n(a_n) + 1}{f'_n(a_n) - 1} \cdot Z_n\,, \tag{26}$$

which allows us to estimate the tangent slope as a coefficient dependent on $f'_n(a_n) = \frac{\partial \tilde{f}_n(a_n)}{\partial a_n}$ scaling the previously assigned value of Z_n. At the same time, (26) requires that $|f'_n(a_n)| < 1$ to ensure positive values of Z_n, aligning with the conditions discussed in Sec. 3.1.

Table 1: *Values of the parameters of the diode-based ring modulator circuit shown in Fig. 1.*

$R_{\rm in}$	$R_{\rm c}$	$R_{\rm d}$	$R_{\rm out}$	$C_{\rm a}$	$C_{\rm b}$	$C_{\rm d}$	$L_{\rm a}$	$L_{\rm b}$
80 Ω	1 Ω	50 Ω	600 Ω	1 nF	1 nF	1 nF	0.8 H	0.8 H

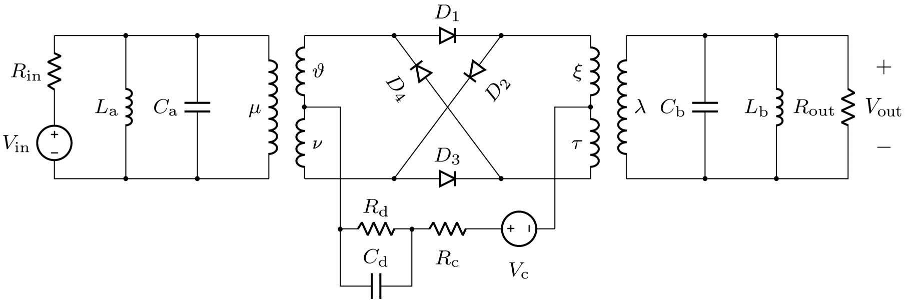


Figure 1: *Dynamic diode-based ring modulator.*

5. EXAMPLE OF APPLICATION

As a possible case study, let us consider the dynamic diode-based ring modulator circuit in Fig. 1. This circuit is characterized by two ideal 3-winding transformers, whose turn ratios are $\vartheta/\mu = \nu/\mu = \xi/\lambda = \tau/\lambda = 1/2$, where ϑ, ν, μ, ξ, τ and λ are the numbers of turns in each winding. The nonlinear diodes are four identical 1N914 diodes, whose extended Schockley model parameters, discussed in Sec. 2.3, are $I_{\rm s} = 2.52$ nA, $\eta = 1.752$, $V_{\rm t} = 26$ mV, $R_{\rm s} = 0.568$ Ω, and $R_{\rm p} = 10^5$ Ω. The parameters and corresponding values of the linear one-port elements are listed in Table 1.

The WD realization of the circuit in Fig. 1 is shown in Fig. 2. The WD structure is characterized by 13 one-port elements and by a 13-port junction $\mathcal{R}_1$ embedding the two ideal 3-winding transformers [8, 16]. The choice of a target circuit featuring diodes is motivated by the existence of efficient WD models of such nonlinearities in the literature on WDFs [8, 17]. This facilitates a rigorous evaluation of the performance of the proposed method.

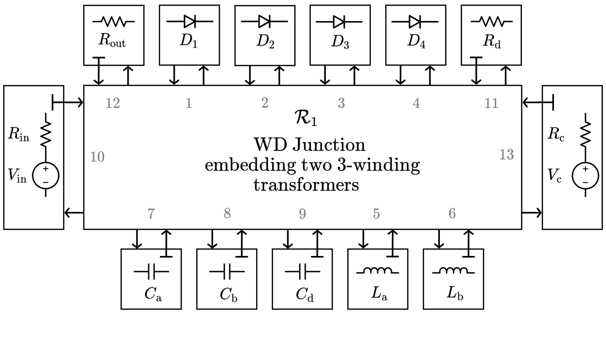


Figure 2: *WD implementation of the circuit shown in Fig.1.*

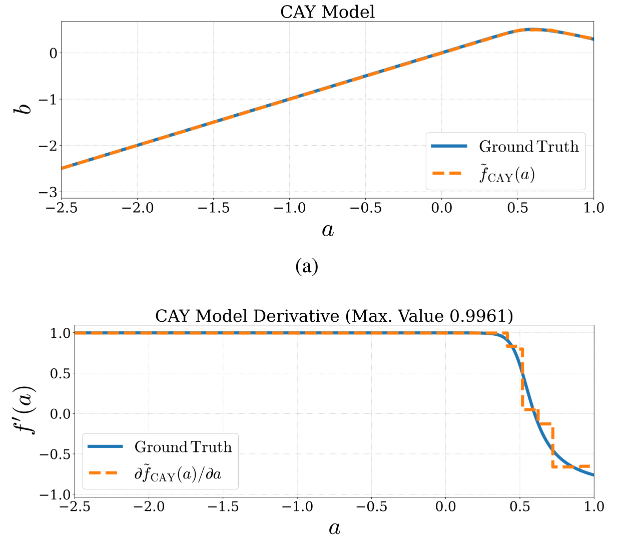
5.1. Dataset and Model Training

For our experiments, we generate a Kirchhoff domain dataset of the 1N914 diode from a Mathworks Simscape extended Schockley diode model implementation and the previously reported set of parameters. We sample the nonlinear diode model by linearly increasing the voltage v from -3V to 1V over the duration of 1 second and acquire the corresponding current i at a sampling rate $f_{\rm s} = 96$ kHz. The obtained v-i characteristic can be converted into corresponding WD variables using (1) and 150 values of Z in the range $[1, 10^5]$ Ω. To ensure adequate sampling of the Z, we split the range of interest into two sub-ranges. We linearly sample 50 values of Z from $[1, 10^3]$ Ω, and the remaining 100 values from $(10^3, 10^5]$ Ω. The compressed values $\varrho(Z) = \ln(Z)$ are then computed and zero centered. Out of the entire dataset, 80% is used for training, while the remaining 20% is set aside for evaluation purposes. Finally, batches containing 256 tuples $\big([a, \ln(Z)]^{\rm T}, b\big)$ are assembled for training purposes.

In our study, we compare the performance of the CAY neural network, built using Lipschitz-bounded layers as defined in (22), with that of a Multilayer Perceptron (MLP) utilizing standard fully connected layers described in (21). Both architectures comprise 2 layers with 32 hidden units each: the CAY neural network employs an absolute value activation function, while the MLP uses the ReLU activation function. The CAY model has an imposed upper bound on its Lipschitz constant of $c = 0.99975$. These models are implemented in Python using Pytorch [37] and are trained for 200 epochs using Adam [38] with a learning rate of 10^{-3} and the other default hyperparameters to minimize the Normalized Mean Squared Error (NMSE) loss function $\mathcal{L}(b, \hat{b}) := \frac{\sum_j (b_j - \hat{b}_j)^2}{\sum_j b_j^2}$.

The reparametrization of the CAY layer induces a strong regularization effect on the neural network, leading to difficulties in fitting the provided training dataset [31]. As evidence, the NMSE computed over the model predictions on the evaluation set is 1.26×10^{-3} for the CAY model and 9.89×10^{-7} for the MLP. However, let us compare the predictions of both models against the explicit WD extended Schockley diode model realization based on the Wright ω function [17], for $a \in [-2.5, 1]$ V and $Z = 10^2$ Ω, in Fig 3(a) and Fig 4(a), respectively. Although both implementations provide a satisfactory approximation of the WD nonlinear diode characteristic, it is noteworthy that the first-order derivative computed with AD of the MLP neural network in Fig 4(b) can locally exceed the value of one, unlike that of the Lipschitz-bounded CAY neural network in Fig 3(b).

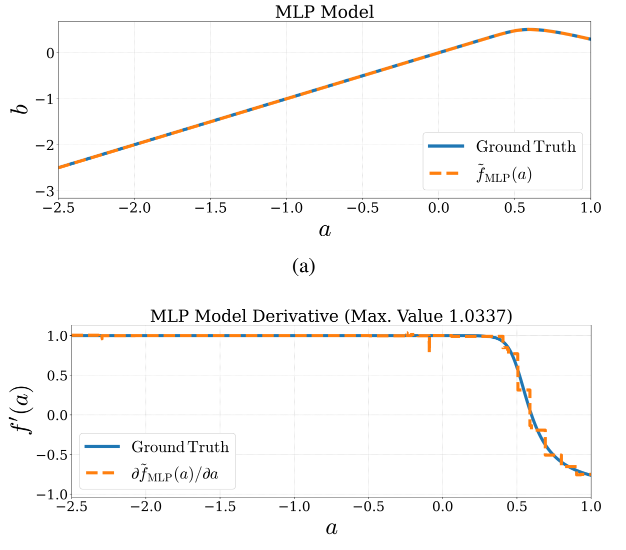
This issue can be also inspected globally by sampling the value of the derivative across the whole input space [39]. To do so, we extract random inputs with $a \in [-2.5, +2.5]$ V and $Z \in [1, 10^5]$ Ω and add a random perturbation with norm $||\epsilon||_2 = 10^{-2}$ to them. The maximum Lipschitz constant across more than 1 bil-



(a)

(b)

Figure 3: *Validation of the CAY-based 1N914 diode WD model: (a) CAY predictions (orange) vs. the ground truth (blue). (b) First-order derivatives computed with AD (orange) vs. the ground truth (blue).*



(a)

(b)

Figure 4: *Validation of the MLP-based 1N914 diode WD model: (a) MLP predictions (orange) vs. the ground truth (blue). (b) First-order derivatives computed with AD (orange) vs. the ground truth (blue).*

lion samples is 0.999747 for the CAY model and 1.288187 for the MLP model. Note that the former is very close to the upper bound of $c = 0.99975$ imposed during training.

5.2. Results

In this subsection, we discuss the numerical results derived from the SIM-based simulation of the nonlinear WD structure of Fig 2, where the nonlinear one-port WD blocks are implemented either using the Lipschitz-bounded CAY neural network or the MLP one.

The circuit is tested with input signal $V_{\mathrm{in}} = \sin(2\pi k f_{\mathrm{in}}/f_{\mathrm{s}})$ and carrier signal $V_{\mathrm{c}} = \sin(2\pi k f_{\mathrm{c}}/f_{\mathrm{s}})$, where k is the sampling index, $f_{\mathrm{s}} = 96$ kHz is the sampling frequency, while $f_{\mathrm{in}} = 1.5$ kHz and $f_{\mathrm{c}} = 500$ Hz are the fundamental frequencies of the input and of the carrier, respectively. All the simulation algorithms of the WD structures are also implemented as Python scripts using Pytorch and its Autograd engine [37], and are run on a laptop-mounted Intel Core i5-1240P 1.70 GHz CPU.

Simulation results related to five periods of the input signal are reported in Fig. 5(a) and Fig. 5(b), respectively. As it is possible to appreciate, both the neural network-based models closely match the Simscape simulation, with a deviation expressed in NMSE between the two output voltages of 1.12×10^{-4} for the CAY-based model and 1.43×10^{-4} for the MLP-based model.

As discussed in Sec. 4.3, at each sampling step k, the port resistances of the four nonlinear diodes Z_n, where $n \in [1, 4]$, are set equal to the data-driven estimate of $v_n'(i_n)$. A comparison between the proposed method and a standard analytic estimate relying on the constitutive equation of the extended Schockley diode model is shown in Fig. 6. For both neural network-based models, the estimated tangent slope at the current working point of the diodes mostly aligns with the analytic estimation. The appearing oscillations and drifts can be attributed to the piecewise linear activation functions employed, which result in a piecewise constant $\frac{\partial \tilde{f}_n(a_n)}{\partial a_n}$. We emphasize that in the absence of an explicit bound, the derivative of the MLP-based model might exceed one. This could lead to invalid estimates of (26), resulting in negative slope values and consequently disrupting the simulation.

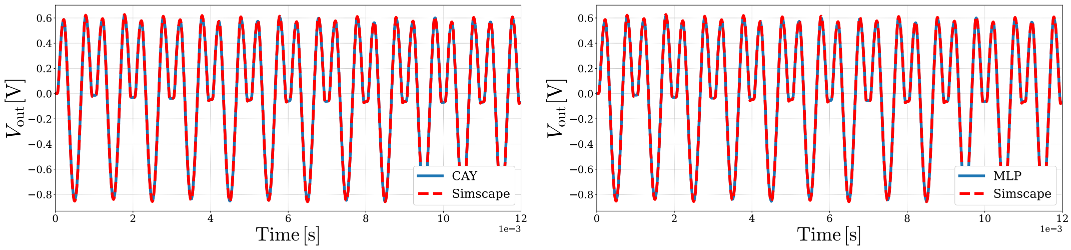
Finally, we conduct a quantitative analysis to assess the simulation speedup introduced by dynamically updating Z_n using (26), focusing just on the CAY-based model. We run 100 simulations over 12 ms of input signal keeping the diode port resistances fixed at the same value of 50 Ω and other 100 dynamically updating them. It is worth noting that, in the simulations in which Z_n is kept fixed, the Pytorch gradient calculation in each forward pass is deactivated [37]. Table 2 compares the average execution time of these simulations. Despite the increased computational load of computing the neural network gradients at each sampling step, our proposed method ensures SIM convergence on average in 7 iterations per sample, compared to the 37 without port resistance update. This in turn results in an overall simulation speedup of 1.92.

Table 2: *Comparison between the average execution time of SIM with and without the Z_n data-driven update.*

	Avg. Exec. Time	Speedup
SIM + Fixed Port Resistances	13.61 s	1
SIM + Proposed Approach	**7.08 s**	**1.92**

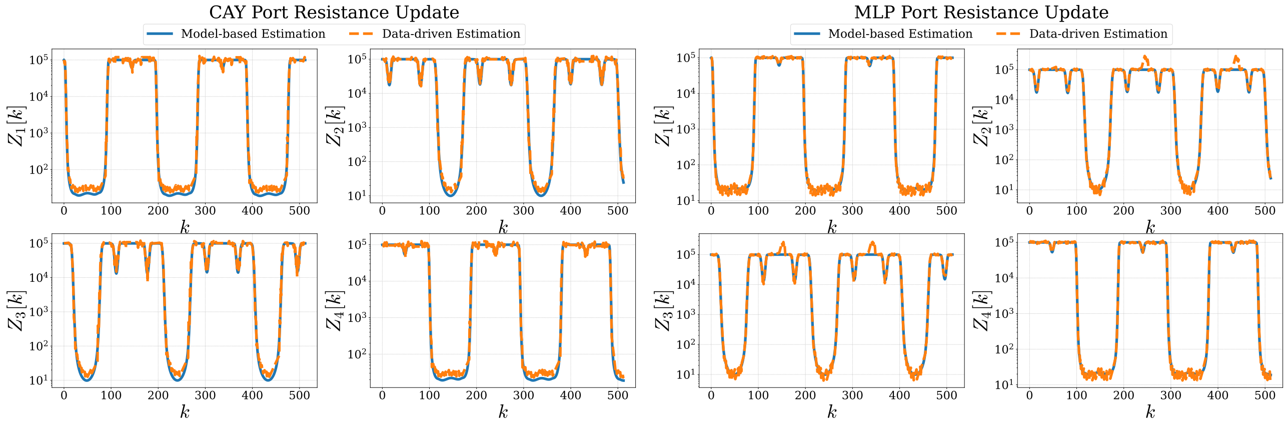
6. CONCLUSIONS

In this paper, we have framed the problem of modeling circuits with multiple one-port nonlinearities in the WD domain into a new



(a) (b)

Figure 5: *Voltage V_{out} measured across the resistor R_{out}. (a) Comparison between the CAY-based WD implementation and Mathworks Simscape. (b) Comparison between the MLP-based WD implementation and Mathworks Simscape.*



(a) (b)

Figure 6: *Estimated $v_n'(i_n[k])$ used to update the port resistance $Z_{1,2,3,4}[k]$ of the four diodes at each sample k: (a) CAY data-driven estimation (orange) vs. analytical estimation (blue). (b) MLP data-driven estimation (orange) vs. analytical estimation (blue)*

data-driven perspective. In particular, we have shown how the use of Lipschitz-bounded neural networks for modeling one-port nonlinearities in the WD domain allows us to recover all the theoretical guarantees on SIM convergence, along with its convergence acceleration policy.

Future work could involve applying this methodology to model multiple one-port nonlinearities with memory. Another noteworthy extension would involve modeling multiple multi-port elements in a fully data-driven fashion and suitably generalizing SIM.

7. REFERENCES

[1] V. Välimäki, S. Bilbao, J. O. Smith, J. S. Abel, J. Pakarinen, and D. Berners, *Virtual analog effects*, pp. 473–522, John Wiley & Sons, United Kingdom, 2nd edition, 2011.

[2] A. Wright, E.P. Damskägg, V. Välimäki, et al., “Real-time black-box modelling with recurrent neural networks,” in *Proc. 22nd Int. Conf. Digital Audio Effects (DAFx-19)*, Birmingham, UK, 2019, University of Birmingham.

[3] M. A. Martínez Ramírez, E. Benetos, and J. D. Reiss, “Deep learning for black-box modeling of audio effects,” *Applied Sciences*, vol. 10, no. 2, 2020.

[4] G. Borin, G. De Poli, and D. Rocchesso, “Elimination of delay-free loops in discrete-time models of nonlinear acoustic systems,” *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 5, pp. 597–605, 2000.

[5] A. Falaize-Skrzek and T. Hélie, “Simulation of an analog circuit of a wah pedal: a port-Hamiltonian approach,” in *Proc. 135th Audio Eng. Soc. (AES) Convention*, New York, USA, 2013, Audio Engineering Society.

[6] G. De Sanctis and A. Sarti, “Virtual analog modeling in the wave-digital domain,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 715–727, 2010.

[7] A. Fettweis, “Wave digital filters: Theory and practice,” *Proceedings of the IEEE*, vol. 74, no. 2, pp. 270–327, 1986.

[8] A. Bernardini, P. Maffezzoni, and A. Sarti, “Linear multistep discretization methods with variable step-size in nonlinear wave digital structures for virtual analog modeling,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 11, pp. 1763–1776, 2019.

[9] A. Bernardini, K. J. Werner, J. O. Smith, III, and A. Sarti, “Generalized wave digital filter realizations of arbitrary reciprocal connection networks,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 2, pp. 694–707, 2019.

[10] A. Sarti and G. De Sanctis, "Systematic methods for the implementation of nonlinear wave-digital structures," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 2, pp. 460–472, 2009.

[11] O. Massi, R. Giampiccolo, A. Bernardini, and A. Sarti, "Explicit vector wave digital filter modeling of circuits with a single bipolar junction transistor," in *Proc. 26th Int. Conf. Digital Audio Effects (DAFx23)*, Copenhagen, Denmark, 2023, pp. 172–179, Aalborg University.

[12] T. Schwerdtfeger and A. Kummert, "A multidimensional approach to wave digital filters with multiple nonlinearities," in *Proc. 22nd Eur. Signal Process. Conf. (EUSIPCO)*, Lisbon, Portugal, 2014, pp. 2405–2409.

[13] K. J. Werner, V. Nangia, J. O. Smith III, and J. S. Abel, "Resolving wave digital filters with multiple/multiport nonlinearities," in *Proc. 18th Int. Conf. Digital Audio Effects (DAFx-15)*, Trondheim, Norway, 2015, pp. 387–394, Norwegian University of Science and Technology.

[14] L. Kolonko, J. Velten, and A. Kummert, "Automatic differentiating wave digital filters with multiple nonlinearities," in *Proc. 28th Eur. Signal Process. Conf. (EUSIPCO)*, Amsterdam, Netherlands, 2021, pp. 146–150.

[15] A. Bernardini, P. Maffezzoni, L. Daniel, and A. Sarti, "Wave-based analysis of large nonlinear photovoltaic arrays," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 4, pp. 1363–1376, 2018.

[16] A. Bernardini, E. Bozzo, F. Fontana, and A. Sarti, "A wave digital Newton-Raphson method for virtual analog modeling of audio circuits with multiple one-port nonlinearities," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2162–2173, 2021.

[17] D. Albertini, A. Bernardini, and A. Sarti, "Scattering iterative method based on generalized wave variables for the implementation of audio circuits with multiple one-port nonlinearities," in *Proc. 150th Audio Eng. Soc. (AES) Convention*, Online, May 25-28 2021.

[18] J. Chowdhury and C. J. Clarke, "Emulating diode circuits with differentiable wave digital filters," in *19th Sound and Music Computing Conference*, Saint-Étienne, France, 2022, pp. 2–9, SMC Network.

[19] C. C. Darabundit, D. Roosenburg, and J. O. Smith, "Neural net tube models for wave digital filters," in *Proc. 25th Int. Conf. Digital Audio Effects (DAFx20in22)*, Vienna, Austria, 2022, pp. 153–160, Vienna University of Music and Performing Arts.

[20] O. Massi, A. I. Mezza, R. Giampiccolo, and A. Bernardini, "Deep learning-based wave digital modeling of rate-dependent hysteretic nonlinearities for virtual analog applications," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2023, no. 1, pp. 12, 2023.

[21] A. Bernardini and A. Sarti, "Canonical piecewise-linear representation of curves in the wave digital domain," in *Proc. 25th Eur. Signal Process. Conf. (EUSIPCO)*, Kos, Greece, 2017, pp. 1125–1129, IEEE.

[22] R. Giampiccolo, A. Bernardini, G. Gruosso, P. Maffezzoni, and A. Sarti, "Multiphysics modeling of audio circuits with nonlinear transformers," *J. Audio Eng. Soc*, vol. 69, no. 6, pp. 374–388, 2021.

[23] A. Virmaux and K. Scaman, "Lipschitz regularity of deep neural networks: analysis and efficient estimation," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[24] A. Trockman and J. Z. Kolter, "Orthogonalizing convolutional layers with the cayley transform," in *Proc. of Int. Conf. on Learning Representations (ICLR)*, 2021.

[25] L. Meunier, B. Delatte, et al., "A dynamical system perspective for Lipschitz neural networks," in *Proc. of the 39th Int. Conf. on Machine Learning*. 17–23 July 2022, pp. 15484–15500, PMLR.

[26] A. Araujo, A. J. Havens, et al., "A unified algebraic perspective on Lipschitz neural networks," in *Proc. of the 11th Int. Conf. on Learning Representations (ICLR)*, 2023.

[27] W. M. Czarnecki, S. Osindero, M. Jaderberg, G. Swirszcz, and R. Pascanu, "Sobolev training for neural networks," in *Advances in Neural Information Processing Systems*, 2017, vol. 30.

[28] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural networks*, vol. 4, no. 2, pp. 251–257, 1991.

[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[30] C. Huang, "Relu networks are universal approximators via piecewise linear or constant functions," *Neural Computation*, vol. 32, no. 11, pp. 2249–2278, 2020.

[31] B. Prach, F. Brau, G. Buttazzo, and C. H. Lampert, "1-Lipschitz layers compared: Memory, speed, and certifiable robustness," 2023.

[32] L. Béthune, T. Massena, T. Boissin, A. Bellet, F. Mamalet, Y. Prudent, C. Friedrich, M. Serrurier, and D. Vigouroux, "DP-SGD without clipping: The Lipschitz neural network way," in *Proc. of the 12th Int. Conf. on Learning Representations (ICLR)*, 2024.

[33] R. Wang and I. R. Manchester, "Youla-ren: Learning nonlinear feedback policies with robust stability guarantees," in *2022 American Control Conference (ACC)*, 2022, pp. 2116–2123.

[34] A. Russo and A. Proutiere, "Towards optimal attacks on reinforcement learning policies," in *2021 American Control Conference (ACC)*, 2021, pp. 4561–4567.

[35] C. Anil, J. Lucas, and R. Grosse, "Sorting out Lipschitz function approximation," in *Proc. of the 36th Int. Conf. on Machine Learning*, 2019, vol. 97, pp. 291–301.

[36] A. Verma, "An introduction to automatic differentiation," *Current Science*, pp. 804–807, 2000.

[37] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[39] E. Manino, R. S. Menezes, F. Shmarov, and L. C. Cordeiro, "Neurocodebench: a plain C neural network benchmark for software verification," 2023.