Universal Approximation of Linear Time-Invariant (LTI) Systems Through RNNs: Power of Randomness in Reservoir Computing

Shashank Jere, *Graduate Student Member, IEEE*, Lizhong Zheng, *Fellow, IEEE*, Karim Said, and Lingjia Liu, *Senior Member, IEEE*

***Abstract*—Recurrent neural networks (RNNs) are known to be universal approximators of dynamic systems under fairly mild and general assumptions. However, RNNs usually suffer from the issues of vanishing and exploding gradients in standard RNN training. Reservoir computing (RC), a special RNN where the recurrent weights are randomized and left untrained, has been introduced to overcome these issues and has demonstrated superior empirical performance especially in scenarios where training samples are extremely limited. On the other hand, the theoretical grounding to support this observed performance has yet been fully developed. In this article, we show that RC can universally approximate a general linear time-invariant (LTI) system. Specifically, we present a clear signal processing interpretation of RC and utilize this understanding in the problem of approximating a generic LTI system. Under this setup, we analytically characterize the optimum probability density function for configuring (instead of training and/or randomly generating) the recurrent weights of the underlying RNN of the RC. Extensive numerical evaluations are provided to validate the optimality of the derived distribution for configuring the recurrent weights of the RC to approximate a general LTI system. Our work results in clear signal processing-based model interpretability of RC and provides theoretical explanation/justification for the power of randomness in randomly generating instead of training RC's recurrent weights. Furthermore, it provides a complete optimum analytical characterization for configuring the untrained recurrent weights, marking an important step towards explainable machine learning (XML) to incorporate domain knowledge for efficient learning.**

***Index Terms*—Reservoir computing, echo state network, neural network, deep learning, system identification and approximation, explainable machine learning.**

I. Introduction

THE rise of deep learning methods [1] in recent times has been unprecedented, owing largely to their remarkable success in fields as diverse as image classification [2], speech recognition [3] and language translation [4], among many others. Specifically, recurrent neural networks (RNNs) are known to be universal approximators for dynamical systems under general conditions [5], making them suitable for applications involving temporally correlated data. Therefore, RNNs are well suited to sequential tasks such as sentence sentiment classification [6], language translation [7], video frame analysis [8], [9] as well as recently in receive processing tasks such as symbol detection [10] in wireless communications. More recently, RNNs have also been adapted to be applied in natural language processing (NLP) tasks to emulate the remarkable success of transformers [11] while avoiding their high computational and memory complexity. However, vanilla RNNs exhibit the problem of vanishing and exploding gradients [12] when trained using the backpropagation through time (BPTT) algorithm [13]. Long short-term memory (LSTM) networks [14] alleviate this problem to a certain degree by incorporating additional internal gating procedures [15], [16] and thus, deliver more robust performance compared to vanilla RNNs [17]. On the other hand, LSTMs require significantly more training data due to their richer modeling capabilities, thereby posing a challenge when the training data is inherently limited, e.g., in the physical (PHY) and medium access control (MAC) layers of modern wireless systems where the over-the-air (OTA) training data is extremely limited. To balance this trade-off, randomized recurrent neural networks [18] have been a topic of active investigation. A general randomized RNN consists of an untrained hidden layer with recurrent units, which non-linearly projects the input data into a high-dimensional feature space, and a trained output layer which scales and combines the outputs of the hidden layer in a linear fashion. Reservoir Computing (RC) [19] is a specific paradigm within the class of randomized RNN approaches where the echo state network (ESN) [20] is a popular implementation of the general RC framework.

In RC architectures including the ESN, typically only the output layer of the network is trained using pseudo-inversion or Tikhonov regularization, while the weights of the input layers and the hidden layers are fixed after initialization based on a certain pre-determined distribution. This particular feature of RC significantly reduces the amount of required training making it uniquely suitable for applications where the number of training samples is extremely limited. Furthermore, since the recurrent weights are randomly generated and fixed, RC completely

Manuscript received 14 July 2023; revised 7 February 2024; accepted 24 March 2024. Date of publication 23 April 2024; date of current version 3 July 2024. This work was supported by U.S. National Science Foundation (NSF) under Grant NSF/CNS-2003059. The guest editor coordinating the review of this manuscript and approving it for publication was Prof. Aylin Yener. *(Corresponding author: Lingjia Liu.)*

Shashank Jere, Karim Said, and Lingjia Liu are with the Bradley Department of ECE at Virginia Tech, Wireless@Virginia Tech, Blacksburg, VA 24061 USA (e-mail: ljliu@ieee.org).

Lizhong Zheng is with the EECS Department, Massachussets Institute of Technology, Cambridge, MA 02139 USA.

Digital Object Identifier 10.1109/JSTSP.2024.3387274

avoids the issues of vanishing and exploding gradients that commonly occur in the standard RNN training. Despite its limited training, RC has demonstrated impressive performance in many sequential processing applications including NLP tasks, e.g., decoding grammatical structure from sentences [21], learning word-to-meaning mappings [22], in video frame analysis tasks such as event detection in visual content [23], as well as in stock market prediction [24]. Recently, RC has found great appeal in various wireless applications, especially in the PHY/MAC layer receive processing with extremely scarce OTA training data. For example, ESNs and its extensions have been utilized to construct symbol detectors for 5G and Beyond 5G multiple antenna systems [10], [25], [26], [27]. In addition, the ESN has been applied to effectively combat inter-symbol interference (ISI) and improve detection performance in a chaotic baseband wireless communication system [28]. Furthermore, ESN-based deep reinforcement learning has been introduced for dynamic spectrum access in 5G networks to provide improved sample efficiency and convergence rate over traditional RNN structures [29]. Beyond conventional wireless communications, RC has also found utility in equalization for optical transmission [30] and signal classification in optoelectronic oscillators [31].

Although RNNs and its variants including RC have shown superior empirical performance in various sequence processing tasks, a fundamental theoretical understanding of their effectiveness using classical tools remains largely unexplored. As discussed in [32], "lack of explainability" is one of the top five challenges in applying machine learning approaches to applications with limited training data such as telecommunication networks, which have traditionally been designed based on a mixture of theoretical analysis, wireless channel measurements, and human intuition and understanding. In fact, the traditional approach has proven amenable for domain experts to resort to either theoretical analysis or computer simulations to validate wireless system building blocks. Therefore, it is desirable for neural network models to have similar levels of explainability especially when designed for wireless systems, and in general for applications with specifications-limited or cost-prohibitive procedures of obtaining training data samples.

A. State-of-the-art in Explainable Machine Learning (XML)

Even though deep neural networks have been effective in various applications, they are still largely perceived as black-box functions converting features in input data to classification labels or regression values at their output. With the growing real-world application of neural network models in sensitive areas such as autonomous driving and medical diagnostics, there is an increasing need to develop a deep understanding of the inner workings of such models. This has given birth to the field of Explainable Machine Learning (XML) which has seen important developments in recent times. A useful overview of Layer-Wise Relevance Propagation (LRP), which is an explainability technique for deep neural networks that uses propagation of relevance information from the output to the input layers, is provided in [33]. An information-theoretic approach towards opening the black box of neural networks was provided in [34] building upon the information bottleneck (IB) principle. SHAP (SHapley Additive exPlanations), which is a model interpretation framework built on the principles of game theory, was introduced in [35]. Outside of neural network models, the work in [36] introduces the concept of local explanation vectors, applying the technique to support vector machines (SVMs). While these works introduce useful interpretation and explanation frameworks, a first principles-based approach that utilizes a signal processing-oriented understanding is largely missing or not yet fully developed for most neural network architectures.

Among studies exploring the theoretical explanations behind the success of RC in time-series problems, one of the first is [37], which introduces a functional space approximation framework for a better understanding of the operation of ESNs. Another recent work of note is [38] which shows that an ESN without nonlinear activation is equivalent to vector autoregression (VAR). [39] makes the case for ESNs being universal approximators for ergodic dynamical systems. The effectiveness of RC in predicting complex nonlinear dynamical systems such as the Lorenz and the Rössler systems was studied in [40], while [41] investigated the tuning and optimization of the length of the fading memory of RC systems. Our previous work in [42] derived an upper bound on the Empirical Rademacher Complexity (ERC) for single-reservoir ESNs and showed tighter generalization for ESNs as compared to traditional RNNs, while simultaneously demonstrating the utility of the derived bound in optimizing an ESN-based symbol detector in multi-antenna wireless receivers. Other statistical learning theory-based works such as [43] also attempt to bound the generalization error for RC using slightly modified Rademacher-type complexity measures. In our previous work [44], we introduce a signal processing analysis of the ESN and present a complete analytical characterization of the optimum untrained recurrent weight for an ESN with a single neuron when employed in the wireless channel equalization task. While the works in existing literature provide interesting insights using information-theoretic or statistical learning-theoretic principles, a lucid signal processing understanding coupled with complete analytical characterizations using conventional tools has not been established yet. With this in mind, we aim to accomplish the following two objectives in this work: i) Gain a theoretical understanding of why randomly generated reservoir weights provide good empirical performance for function approximation, and ii) develop a systematic methodology to configure this random generation of reservoir weights incorporating prior information or domain knowledge. With these objectives, we provide an outline in the next section for the set of problems considered, the overall approach adopted and the steps taken to solve them.

B. Our Contributions

The main contributions of this work are summarized below:

1) First, we consider the "atomic" problem of approximating the impulse response of a first-order infinite impulse response (IIR) filter using an ESN consisting of two neurons in the reservoir with fixed reservoir weights and with linear

activation. Formulating this as an orthogonal projection problem, we precisely calculate the corresponding approximation error and derive an exact scaling law that relates this approximation error to the distance between the ESN's poles (i.e., the recurrent reservoir weights).

2) Second, continuing with the impulse response of the first-order IIR system as the target function, we consider the problem of approximating its impulse response using an ESN with multiple neurons having randomly generated weights. Optimizing the corresponding approximation error, we derive the optimum probability density function (PDF) to configure the random generation of the ESN reservoir weights.
3) Third, we generalize this result by showing that the derived optimum PDF for approximating a first-order IIR system is also optimum for approximating general higher-order LTI systems that can be written as a linear combination of first-order poles.
4) Fourth, we show that under linear activation, a reservoir with random and sparse interconnections between its constituent neurons has an equivalent representation as a reservoir with non-interconnected neurons.
5) Finally, via extensive numerical evaluations, we empirically confirm the following: i) Validity of the derived approximation error scaling law, and ii) Optimality of the derived optimum PDF for configuring the ESN reservoir weights when applied to the task of approximating a first-order IIR and higher-order LTI systems.

The rest of the paper is organized as follows. Section II presents the problem formulation for the task of LTI system approximation using an ESN. Section III presents our approach and analysis to derive the optimum distribution for configuring the random generation of ESN reservoir weights. Section IV outlines the training procedure of the ESN and briefly outlines overfitting concerns in this scenario. Numerical evaluations to validate the theoretical findings in the preceding sections are presented in Section V. Finally, we provide concluding remarks and directions for future work in Section VI.

Notation: $\mathbb{R}$: set of real numbers; $\mathcal{U}(a,b)$: uniform distribution with support $[a,b]$; $\mathcal{N}(\mu,\sigma^2)$: Gaussian (normal) distribution with mean μ and variance σ^2; $\mathbb{E}[\cdot]$: Expectation operator, $\mathsf{VAR}(\cdot)$: Variance operator; c and C denote scalars, $\mathbf{c}$ denotes a column vector; $\|\cdot\|_2$: ℓ_2-norm; $\langle \mathbf{a},\mathbf{b}\rangle = \mathbf{b}^T\mathbf{a}$: inner product of vectors $\mathbf{a},\mathbf{b}\in\mathbb{R}^n$. $\mathbf{C}$ denotes a matrix; $(\cdot)^T$: matrix transpose; $(\cdot)^\dagger$: Moore-Penrose matrix pseudo-inverse. $\lambda(\mathbf{C})$ denotes the spectrum (set of eigenvalues) of $\mathbf{C}$. $p_{\mathrm{A}}(\cdot)$ denotes the probability density function (PDF) of a random variable α. $\Pr(E)$ denotes the probability of event E. (a,b) denotes an open interval and $[a,b]$ denotes a closed interval for $a,b\in\mathbb{R}$. W.L.O.G. stands for "without loss of generality". We define the following terms to have this specific meaning in the remainder of the paper: i) "Training": Data-driven optimization of neural network (NN) model weights via backpropagation-based or single-shot algorithms (e.g., least squares), ii) "Randomly generating": The process of generating NN model weights in an i.i.d. manner by sampling them from a pre-determined and unoptimized distribution, iii) "Configuring": The process of generating NN model weights in an i.i.d. manner by sampling them from an analytically derived distribution taking into account prior information or domain knowledge.

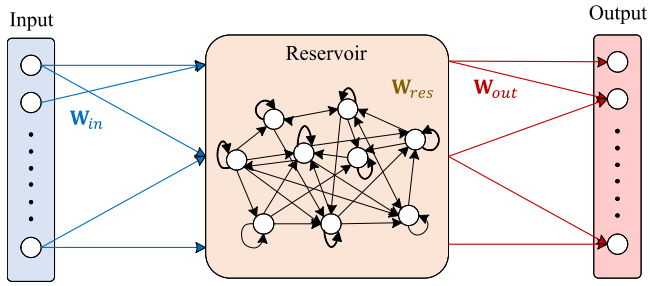

Fig. 1. Echo state network (ESN) with a single reservoir.

II. Problem Formulation

A. Randomized RNN: RC and the Echo State Network (ESN)

In the context of a randomized RNN [18] and more specifically an ESN, a general learning problem can be defined by the tuple $(\mathcal{Z},\mathcal{P},\mathcal{H},\ell)$, where:

- $\mathcal{X}$ and $\mathcal{Y}$ are the input and output spaces respectively. In our case, $\mathcal{X}\in\mathbb{R}^{D\times T}$ represents a time sequence of length T. The output space is $\mathcal{Y}\in\mathbb{R}^{K\times T}$ or $\mathcal{Y}\in\{0,1\}^{K\times T}$, depending on whether the network is being employed for regression or classification respectively in the sequence-to-sequence setting. In the sequence-to-vector setting, we have $\mathcal{Y}\in\mathbb{R}^K$ or $\mathcal{Y}\in\{0,1\}^K$. Here, D and K are the input and output dimensions respectively.
- $\mathcal{Z}=\mathcal{X}\times\mathcal{Y}$ represents the joint input-output space.
- $\mathcal{P}$ is the space of probability distributions defined on $\mathcal{Z}$.
- $\mathcal{H}$ is the space of all function approximators $h:\mathcal{X}\to\mathcal{Y}$ where h denotes the neural network function.
- The loss function $\ell(\cdot)$ is defined as $\ell:\mathcal{Y}\times\mathcal{Y}\to\mathbb{R}$.

Define an input sequence $\mathbf{U}=[\mathbf{u}[1],\mathbf{u}[2],\ldots,\mathbf{u}[T]]$ of length T such that $\mathbf{u}[n]\in\mathbb{R}^D$ and $\mathbf{U}\in\mathbb{R}^{D\times T}$ for the discrete-time indices $n=1,2,\ldots,T$. Note that each data sample $\mathbf{u}(n)$ in the time series $\mathbf{U}$ is a (column) vector of dimension D. For every training sequence $\mathbf{U}$, a label (ground truth) sequence $\mathbf{G}$ is available to train the network, where $\mathbf{G}=[\mathbf{g}[1],\mathbf{g}[2],\ldots,\mathbf{g}[T]]$ such that $\mathbf{g}[n]\in\mathbb{R}^K$ for a (sequence-to-sequence) regression task and $\mathbf{g}[n]\in\{0,1\}^K$ for a (sequence-to-sequence) classification task. The training set Z^N of size N is then defined as the set of input-label tuples $\mathsf{Z}^N:=\{(\mathbf{U}_1,\mathbf{G}_1),(\mathbf{U}_2,\mathbf{G}_2),\ldots,(\mathbf{U}_N,\mathbf{G}_N)\}$, where Z^N is generated i.i.d. according to some (unknown) joint input-output probability distribution $\mathsf{P}(\cdot,\cdot)\in\mathcal{P}$. The general setup described above is applicable to a time series problem with any recurrent deep learning model. Within the class of randomized RNNs, we consider a single reservoir ESN containing M neurons with random and sparse interconnections (among other possibilities) and a single output (readout) weights matrix. This structure is depicted in Fig. 1. Next, we define the input, output and the model weights of the ESN in the following:

- $\mathbf{x}_{\mathrm{res}}[n]\in\mathbb{R}^M$ is the reservoir state vector at time index n.

- $\mathbf{X}_{\text{res}} = [\mathbf{x}_{\text{res}}[1], \ldots, \mathbf{x}_{\text{res}}[T]] \in \mathbb{R}^{M\times T}$ is defined as the "reservoir states matrix" of the individual states from $n = 1$ to $n = T$.
- $\mathbf{x}_{\text{in}}[n] \in \mathbb{R}^D$ denotes the ESN input. $\mathbf{y}[n] \in \mathbb{R}^K$ denotes the ESN output.
- $\mathbf{W}_{\text{in}} \in \mathbb{R}^{M\times D}$ is the input weights matrix, $\mathbf{W}_{\text{res}} \in \mathbb{R}^{M\times M}$ is the reservoir weights matrix, $\mathbf{W}_{\text{out}} \in \mathbb{R}^{K\times M}$ is the output weights matrix.

For a point-wise nonlinear activation function $\sigma(\cdot)$, the state update equation and the output equation are respectively:

$$\mathbf{x}_{\text{res}}[n] = \sigma\left(\mathbf{W}_{\text{res}}\mathbf{x}_{\text{res}}[n-1] + \mathbf{W}_{\text{in}}\mathbf{x}_{\text{in}}[n]\right), \tag{1}$$

$$\mathbf{y}[n] = \mathbf{W}_{\text{out}}\mathbf{x}_{\text{res}}[n]. \tag{2}$$

In this setup, $\mathbf{W}_{\text{in}}$ and $\mathbf{W}_{\text{res}}$ are randomly generated, i.e., initialized from a certain pre-determined but arbitrary distribution, e.g., the uniform or Gaussian distributions, and then kept fixed throughout the training and inference (test) stages. Unlike vanilla RNNs and its variants where all network weights are trained using BPTT, the only trainable network parameter in the ESN is the output weights matrix $\mathbf{W}_{\text{out}}$, which is trained using a pseudoinverse-based closed-form linear update rule. This greatly reduces the number of trainable parameters, as well as the training computational complexity, lending well to applications with limited training data availability. Additionally, the sparsely interconnected nature of $\mathbf{W}_{\text{res}}$ is controlled via the hyperparameter named 'sparsity' (denoted as κ), which represents the probability of an element of $\mathbf{W}_{\text{res}}$ being zero. The internal reservoir structure of Fig. 1 depicts this random and potentially sparse nature of the interconnections between the constituent neurons.

B. Approximating an Atomic LTI System With an ESN

Consider the target LTI system characterized by the following causal time-domain impulse response:

$$\mathbf{s}_\alpha[n] = \begin{cases} \alpha^n, & n \geqslant 0 \\ 0, & n < 0 \end{cases} = \alpha^n u[n], \tag{3}$$

where $\alpha \in (-1, 1)$ and $u[n]$ is the discrete-time unit step function. Thus, the target system to be approximated by the ESN is described by the time-domain impulse response characterized by the infinite-dimensional vector $\mathbf{s}_\alpha \in \mathbb{R}^\infty$. We choose this as the first case to analyze since the time-domain impulse response of a large class of general LTI systems can be written exactly as a linear combination of first order IIR impulse responses of (3) [45], i.e.,

$$\mathbf{h}[n] = \sum_{i=1}^{N_0} w_i \mathbf{s}_{\alpha_i}[n], \tag{4}$$

where $w_i \in \mathbb{R}$ are the combining weights, thereby making the extension to the general case feasible given the analysis for the simple case of (3). This is shown in Section III-E.

In this work, we consider a simplified version of the more general ESN described in Section II-A. Specifically, we consider a simple reservoir construction where the individual neurons are disconnected from each other and only consist of unit delay self-feedback loops. This translates to $\mathbf{W}_{\text{res}}$ being a diagonal matrix. Next, for tractability of analysis, we consider linear activation such that $\sigma(\cdot)$ in (1) is an identity mapping. As shown in our previous work [44], a single neuron with linear activation can be modeled as a first-order infinite impulse response (IIR) filter with a single pole. This is illustrated in Fig. 2, where a single neuron or "node" inside the reservoir simply implements a first-order autoregressive process AR(1) with a feedback weight a. The system response for the IIR filter in Fig. 2 is given by

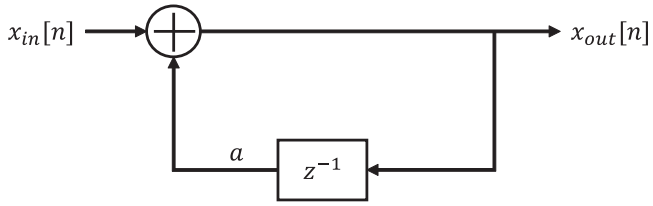

Fig. 2. Modeling a neuron in the reservoir with linear activation as a single-pole IIR filter.

$$H_0(z) = \frac{X_{\text{out}}(z)}{X_{\text{in}}(z)} = \frac{1}{1 - az^{-1}}. \tag{5}$$

Finally, we consider the input weights to be unity, as their effect is absorbed in the output weights when the activation employed in the reservoir is linear.

With the aforementioned preliminaries laid out, the ESN design problem for LTI system approximation can be articulated as follows. Consider an ESN reservoir as a collection of non-interconnected neurons with fixed corresponding reservoir (recurrent) weights $\{\beta_m\}_{m=1}^M$, where each $\beta_m \in (-1, 1)$ to ensure stability of its impulse response $\beta_m^n u[n]$. Such a reservoir with non-interconnected neurons has also been considered for neuromorphic computing in an experimental setting using photonic hardware [46], thus highlighting its practical applicability. We would like to choose $\{\beta_m\}_{m=1}^M$ such that their weighted combination approximates the normalized target $\frac{\mathbf{s}_\alpha}{\|\mathbf{s}_\alpha\|_2}$ with a low approximation error, i.e.,

$$\frac{\mathbf{s}_\alpha}{\|\mathbf{s}_\alpha\|_2} \approx \sum_{m=1}^{M} W_m \mathbf{s}_{\beta_m}, \tag{6}$$

where $\mathbf{s}_{\beta_m}[n] = \beta_m u[n]$. Note that target normalization is imperative to ensure that the mean approximation error across multiple LTI system realizations (i.e., values of "parameter" α) is not dominated by realizations for which α is closer to 1 or -1 over those for which α is closer to 0. With this formulation, the normalized target has unit norm. This can also be written as the system function in terms of the z-transform as

$$S_\alpha(z) \approx \sum_{m=1}^{M} W_m S_{\beta_m}(z), \tag{7}$$

which can be expanded as

$$\frac{\sqrt{1-\alpha^2}}{1 - \alpha z^{-1}} \approx \sum_{m=1}^{M} \frac{W_m}{1 - \beta_m z^{-1}}. \tag{8}$$

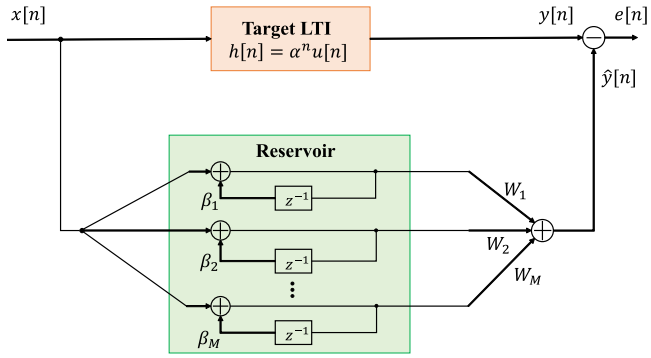

Fig. 3. Approximating an LTI system belonging to a known model family (e.g., first-order IIR system) with a linear non-interconnected reservoir ESN.

Thus, the system being estimated is an infinite impulse response (IIR) system with a single pole α, where the ESN attempts to estimate this IIR impulse response as a weighted combination of M IIR impulse responses characterized by the random poles $\{\beta_m\}$, which are kept fixed during training of the output weights $\{W_m\}$ and during test. This problem can be characterized as a system "approximation" or "identification" problem, whereby a linear ESN with a reservoir of non-interconnected neurons with randomly generated or configured weights attempts to reproduce the output of the unknown LTI system belonging to a known model family, in this case, a single-pole IIR system. The problem setup is depicted in Fig. 3.

C. ESN Initialization and Training

The process of initializing the ESN reservoir weights (random generation or configuration) and subsequent training of output weights consists of three components: i) a target function $f(\cdot;\alpha)$ to be approximated, ii) a linear subspace $\mathbf{\Omega}$ spanned by the reservoir basis functions, and iii) an approximation $\widehat{f}(\cdot;\beta_1,\ldots,\beta_M)$ of the target function in $\mathbf{\Omega}$. For the LTI system approximation task, the target function is the normalized impulse response of the system, given by $f(\cdot;\alpha)=\frac{\mathbf{s}_\alpha}{\|\mathbf{s}_\alpha\|_2}$. The subspace spanned by the reservoir neurons is given by $\mathbf{\Omega}=\text{span}(\mathbf{s}_{\beta_1},\ldots,\mathbf{s}_{\beta_M})$. For a general loss function $\mathcal{L}(f;\widehat{f})$, the training procedure finds the output combining weights $\{W_m\}$ such that the approximation $\widehat{f}(\cdot;\beta_1,\ldots,\beta_M)$ lying in $\mathbf{\Omega}$ minimizes $\mathcal{L}(f;\widehat{f})$. With the ℓ_2 norm as the loss function $\mathcal{L}(f;\widehat{f})$, the ESN training procedure finds $\widehat{f}(\cdot;\beta_1,\ldots,\beta_M)$ as the ℓ_2 training loss minimizing approximation, implying that it is the orthogonal projection of $f(\cdot;\alpha)$ onto $\mathbf{\Omega}$. The corresponding approximation error is then referred to as the "projection error". This setup is illustrated in Fig. 4. The projection error can be written as the following ℓ_2-loss:

$$\varepsilon=\left\|\frac{\mathbf{s}_\alpha}{\|\mathbf{s}_\alpha\|_2}-\sum_{m=1}^{M}W_m^*\mathbf{s}_{\beta_m}\right\|_2^2, \tag{9}$$

where

$$\{W_m^*\}=\arg\min_{\{W_m\}}\left\|\frac{\mathbf{s}_\alpha}{\|\mathbf{s}_\alpha\|_2}-\sum_{m=1}^{M}W_m\mathbf{s}_{\beta_m}\right\|_2^2, \tag{10}$$

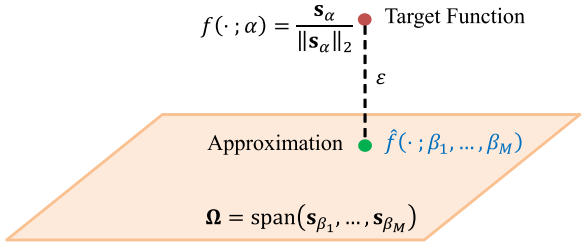

Fig. 4. Learning a single-pole IIR system: An orthogonal projection view.

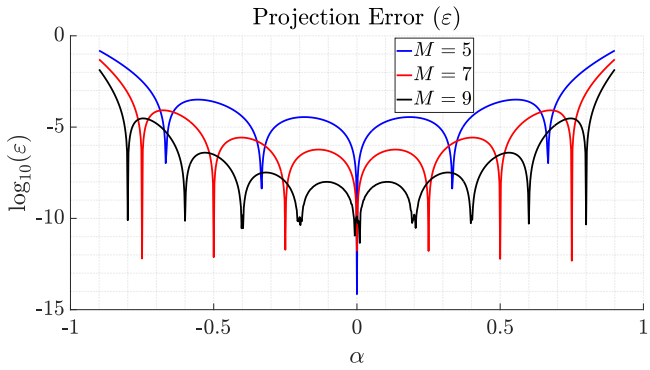

Fig. 5. Projection Error (ε) of (12) versus α for $M=5,7,9$ ESN poles evenly spaced in $(-1,1)$. The local minima represent the locations of the poles $\{\beta_m\}_{m=1}^M$ in each case.

are the optimum output weights given by

$$\mathbf{w}=\mathbf{\Sigma}^{-1}\mathbf{r}. \tag{11}$$

Here, $\mathbf{w}\triangleq[W_1^*\ \ldots\ W_M^*]^T\in\mathbb{R}^M$, and the projection error can be shown to be

$$\varepsilon=1-\mathbf{r}^T\mathbf{\Sigma}^{-1}\mathbf{r}. \tag{12}$$

Here, $\|\mathbf{s}_\alpha\|_2^2=\sum_{n=0}^{\infty}\alpha^{2n}=\frac{1}{1-\alpha^2}$, and $\mathbf{r}\in\mathbb{R}^{M\times 1}$ and $\mathbf{\Sigma}\in\mathbb{R}^{M\times M}$ are respectively defined as

$$\mathbf{r}\triangleq\frac{1}{\|\mathbf{s}_\alpha\|_2}\begin{bmatrix}\langle\mathbf{s}_{\beta_1},\mathbf{s}_\alpha\rangle\\ \vdots\\ \langle\mathbf{s}_{\beta_M},\mathbf{s}_\alpha\rangle\end{bmatrix}, \tag{13}$$

and $[\mathbf{\Sigma}]_{i,j}\triangleq\langle\mathbf{s}_{\beta_i},\mathbf{s}_{\beta_j}\rangle$, where $[\mathbf{\Sigma}]_{i,j}$ is the element of $\mathbf{\Sigma}$ in the ith row and the jth column. As an example, the projection error of (12) is plotted in Fig. 5 with M evenly spaced poles in the interval $(-1,1)$ for $M=5,7,9$.

The projection error of (12) is intrinsically linked to the training loss when $\{W_m\}$ are trained with finite labeled data samples. Specifically, for an impulse input, i.e., $\|\mathbf{x}\|_2=1$, the data-driven training loss is lower bounded by the projection error ε. This is because the projection error makes use of the "optimum" output combining weights $\{W_m^*\}$, computing which requires knowledge of α. This inherently assumes that an infinite number of data samples are available for training $\{W_m\}$ so that they converge to $\{W_m^*\}$. Thus, the projection error of (12) is the lowest achievable training loss for an impulse input ($\|\mathbf{x}\|=1$). Then, the loss metric $\mathcal{L}(\cdot;\cdot)$ to be used to optimize the fixed

reservoir weights $\{\beta_m\}$ can be defined as the projection error of (9), i.e.,

$$\mathcal{L}(\alpha;\beta_1,\ldots,\beta_M) \triangleq \varepsilon. \tag{14}$$

Since we are interested in designing a single ESN with reservoir weights that provides a low approximation error *on average*, we model α as a random variable with a known prior PDF $p_{\mathrm{A}}(\cdot)$. For example, system identification tasks in acoustic and electromechanical servo systems employ frequency-domain methods [47], [48] in practice to empirically deduce the distribution of the system poles or the modes of a given LTI system. Then, the ultimate ESN design goal is to analytically choose the fixed optimum reservoir weights $\{\beta_1^*,\ldots,\beta_M^*\}$ according to

$$\{\beta_1^*,\ldots,\beta_M^*\} = \underset{\{\beta_1,\ldots,\beta_M\}}{\arg\min}\ \mathbb{E}_{\alpha\sim p_{\mathrm{A}}(\cdot)}\left[\mathcal{L}(\alpha;\beta_1,\ldots,\beta_M)\right], \tag{15}$$

so that the expected projection error is minimized, where the expectation is taken over the target function parameter α.

Determining the optimum $\{\beta_m^*\}_{m=1}^M$ individually can be intractably challenging. Therefore, we take an alternative approach of treating each β_m as a random variable such that an individual pole β_m is sampled i.i.d. from the PDF $p_{\mathrm{B}}(\cdot)$. Instead of finding the optimum reservoir weights individually, we attempt to find the optimum probability distribution in terms of its PDF $p_{\mathrm{B}}^*(\cdot)$, from which the poles $\{\beta_1,\ldots,\beta_M\}$ can be "configured" by sampling from $p_{\mathrm{B}}^*(\cdot)$ in an i.i.d. manner. Therefore, the reservoir optimization problem can be reformulated as determining the optimum PDF $p_{\mathrm{B}}^*(\cdot)$ of the ESN pole distribution which satisfies

$$p_{\mathrm{B}}^*(\cdot) = \underset{p_{\mathrm{B}}(\cdot)}{\arg\min}\,\mathbb{E}_{\{\beta_1,\ldots,\beta_M\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}(\cdot)}\left[\mathbb{E}_{\alpha\sim p_{\mathrm{A}}(\cdot)}\left[\mathcal{L}(\alpha;\beta_1,\ldots,\beta_M)\right]\right]. \tag{16}$$

In the next section, we describe a method of solving this optimization problem by using a local approximation.

III. Reservoir Optimization

A. Nearest Neighbors Approximation

As $M\to\infty$, the projection error $\mathcal{L}(\alpha;\beta_1,\ldots,\beta_M)$ can be estimated by making a "nearest neighbors approximation". This approximation states that in the neighborhood of a given α, the approximation error due to $\{\beta_m\}_{m=1}^M$ is dominated by the two ESN poles closest to α. In this treatment, we assume that $\alpha\sim\mathcal{U}(-\alpha_0,\alpha_0)\triangleq p_{\mathrm{A}}(\cdot)$, where $0<\alpha_0<1$. Then, the nearest neighbors approximation states that

$$\mathcal{L}(\alpha;\beta_1,\ldots,\beta_M)\approx\widetilde{\mathcal{L}}(\alpha;\beta_1,\ldots,\beta_M), \tag{17}$$

where the "surrogate loss" $\widetilde{\mathcal{L}}$ is defined as

$$\widetilde{\mathcal{L}}(\alpha;\beta_1,\ldots,\beta_M)\triangleq\mathcal{L}(\alpha;\beta^{(1)},\beta^{(2)}). \tag{18}$$

Here, $\beta^{(1)}$ and $\beta^{(2)}$ are the two ESN poles that are closest to α, i.e., its two nearest neighbors, with $\beta^{(1)},\beta^{(2)}\subset\{\beta_1,\ldots,\beta_M\}$. In this treatment, we define a local neighborhood $\mathcal{R}$ as the interval containing $\beta^{(1)}$, α and $\beta^{(2)}$, i.e., it is the interval containing the LTI system pole α and only the two nearest ESN poles β_1 and β_2. With the approximation of (17), the optimization problem can be stated as

$$p_{\mathrm{B}}^*(\cdot) = \underset{p_{\mathrm{B}}(\cdot)}{\arg\min}\,\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}(\cdot)}\left[\mathbb{E}_{\alpha\sim p_{\mathrm{A}}(\cdot)}\left[\widetilde{\mathcal{L}}\left(\alpha;\{\beta_m\}_{m=1}^M\right)\right]\right]. \tag{19}$$

In the following sequence of steps, we denote $p_{\mathrm{B}}^*(\cdot)$ as p_{B}^* for brevity of notation. If the projection error corresponding to the problem in (19) is ε_1, i.e.,

$$\begin{aligned}
\varepsilon_1 &= \min_{p_{\mathrm{B}}}\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}}\left[\mathbb{E}_{\alpha\sim p_{\mathrm{A}}(\cdot)}\left[\widetilde{\mathcal{L}}(\alpha;\beta_1,\ldots,\beta_M)\right]\right],\\
&= \min_{p_{\mathrm{B}}}\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}}\left[\sum_{\mathcal{R}}\int_{u\in\mathcal{R}}p_{\mathrm{A}}(u)\widetilde{\mathcal{L}}(\alpha;\beta_1,\ldots,\beta_M)du\right],\\
&= \min_{p_{\mathrm{B}}}\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}}\left[\sum_{\mathcal{R}}\int_{u\in\mathcal{R}}p_{\mathrm{A}}(u)\mathcal{L}\left(\alpha;\beta^{(1)},\beta^{(2)}\right)du\right],\\
&\leqslant \min_{p_{\mathrm{B}}}\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}}\left[\sum_{\mathcal{R}}\int_{u\in\mathcal{R}}p_{\mathrm{A}}(u)\cdot\sup_{\alpha\in\mathcal{R}}\mathcal{L}\left(\alpha;\beta^{(1)},\beta^{(2)}\right)du\right],\\
&= \min_{p_{\mathrm{B}}}\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}}\left[\sum_{\mathcal{R}}\Pr(\alpha\in\mathcal{R})\cdot\sup_{\alpha\in\mathcal{R}}\mathcal{L}\left(\alpha;\beta^{(1)},\beta^{(2)}\right)\right],
\end{aligned} \tag{20}$$

where the dummy variable u denotes a particular realization of the random variable α. Since $\Pr(\alpha\in\mathcal{R})$ is constant regardless of the location of the small neighborhood $\mathcal{R}$ in the entire range of $[-\alpha_0,\alpha_0]$ for $\alpha\sim p_{\mathrm{A}}(\cdot)$, the optimization problem can be stated as the min-max formulation given by

$$p_{\mathrm{B}}^* = \underset{p_{\mathrm{B}}}{\arg\min}\,\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}}\left[\sum_{\mathcal{R}}\sup_{\alpha\in\mathcal{R}}\left[\mathcal{L}(\alpha;\beta^{(1)},\beta^{(2)})\right]\right]. \tag{21}$$

As we shall see in the next section, obtaining the exact expression for $\sup_{\alpha\in\mathcal{R}}[\mathcal{L}(\alpha;\beta^{(1)},\beta^{(2)})]$ can be intractably challenging. Instead, we derive an upper bound $B_\varepsilon^{(\mathcal{R})}$ for this term so that $\sup_{\alpha\in\mathcal{R}}[\mathcal{L}(\alpha;\beta^{(1)},\beta^{(2)})]\leq B_\varepsilon^{(\mathcal{R})}$. Therefore, we define the "optimum" distribution $p_{\mathrm{B}}^*(\cdot)$ as that which solves the following optimization problem:

$$p_{\mathrm{B}}^*(\cdot) = \underset{p_{\mathrm{B}}(\cdot)}{\arg\min}\,\mathbb{E}_{\{\beta_m\}\overset{\text{i.i.d.}}{\sim}p_{\mathrm{B}}}\left[\sum_{\mathcal{R}}B_\varepsilon^{(\mathcal{R})}\right]. \tag{22}$$

In the next section, we derive an expression for $B_\varepsilon^{(\mathcal{R})}$.

B. Error Bound With Nearest Neighbors Approximation

We analyze the behavior of the error of (12) in the small neighborhood $\mathcal{R}$ where the two nearest neighbors of $\alpha\in\mathcal{R}$, $\beta^{(1)}$ and $\beta^{(2)}$ are denoted simply as β_1 and β_2 for clarity of notation and $\beta_1<\beta_2$ W.L.O.G. Thus, for the purpose of this analysis, $\mathcal{R}$ is defined as the interval containing β_1, α and β_2. With this disclaimer, in the following analysis, we denote the projection error of (12) as $\varepsilon_{(2)}$, where the subscript (2) denotes

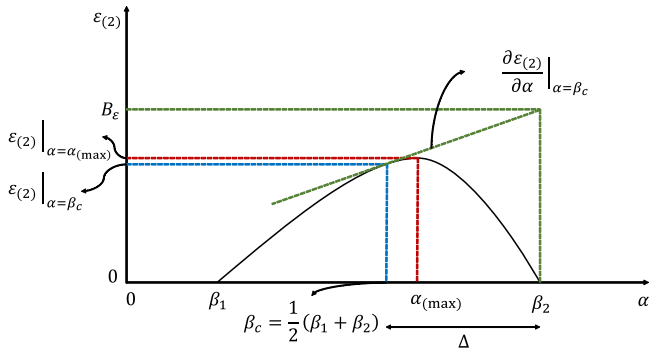

Fig. 6. Nearest Neighbors Approximation: Projection error $\varepsilon_{(2)}$ plotted in the neighborhood $\mathcal{R}$ with poles β_1 and β_2 on its edges.

the fact that we are evaluating a 2-nearest neighbors error in a small neighborhood $\mathcal{R}$. $\varepsilon_{(2)} \triangleq \mathcal{L}(\alpha;\beta_1,\beta_2)$ can be evaluated by substituting for $\mathbf{r}$ and $\mathbf{\Sigma}$ with $M=2$ in (12). After further manipulation, it can be obtained as

$$\varepsilon_{(2)} = 1 - \frac{(1-\alpha^2)(1-\beta_1\beta_2)}{(\beta_1-\beta_2)^2}\left(\frac{(1-\beta_1^2)(1-\beta_1\beta_2)}{(1-\alpha\beta_1)^2} - 2\frac{(1-\beta_1^2)(1-\beta_2^2)}{(1-\alpha\beta_1)(1-\alpha\beta_2)} + \frac{(1-\beta_2^2)(1-\beta_1\beta_2)}{(1-\alpha\beta_2)^2}\right). \tag{23}$$

Since our focus is on the small neighborhood $\mathcal{R}$, we quantify the density of packing of the two ESN poles β_1 and β_2 by defining them around a mid-point β_c as $\beta_1=\beta_c-\Delta$ and $\beta_2=\beta_c+\Delta$, where $\Delta\geq 0$ and $\beta_c \triangleq \frac{1}{2}(\beta_1+\beta_2)$.

We are interested in the trend followed by the maximum value of this error within $\mathcal{R}$ as a function of Δ. However, obtaining an expression for the true maximum error $\varepsilon_{(2)}^{(\max)}$ by finding the stationary point inside $\mathcal{R}$ can be intractably tedious. Therefore, instead of finding $\varepsilon_{(2)}^{(\max)}$, we attempt to find an upper bound B_ε on $\varepsilon_{(2)}^{(\max)}$. With this final goal, we state the following proposition.

Proposition 1: An upper bound on the maximum error in $\mathcal{R}$ is given by

$$B_\varepsilon = \varepsilon_{(2)}^{(\mathrm{mid})} + \Delta\left|\frac{\partial\varepsilon_{(2)}}{\partial\alpha}\Big|_{\alpha=\beta_c}\right|, \tag{24}$$

where $\varepsilon_{(2)}^{(\mathrm{mid})} \triangleq \varepsilon_{(2)}(\beta_c)$.

This can be seen with the aid of Fig. 6 which plots the projection error $\varepsilon_{(2)}$ with the nearest neighbors approximation for $\beta_1,\ \beta_2>0$ W.L.O.G. It can be observed that B_ε of (24) is one of the possible upper bounds on the true maximum error $\varepsilon_{(2)}|_{\alpha=\alpha_{\max}}$. Since $\varepsilon_{(2)}$ is a concave function of α and has exactly one local maximum inside $\mathcal{R}$, the claim of Proposition 1 always holds within $\mathcal{R}$ which is bounded by exactly one ESN pole on either side. Then, $\varepsilon_{(2)}$ evaluated at β_c is given in the following lemma.

Lemma 1: The 2-nearest neighbors-based projection error $\varepsilon_{(2)}$, evaluated at the mid-point, i.e., $\alpha=\beta_c$ of the small neighborhood $\mathcal{R}$ is given by

$$\varepsilon_{(2)}^{(\mathrm{mid})} = \frac{1}{(1-\beta_c^2)^4}\Delta^4 + O(\Delta^6). \tag{25}$$

The complete proof of this result is provided in Appendix A. This is an important outcome, indicating that the neighborhood error has a power scaling law with Δ given by Δ^4. Similarly, an expression for $\frac{\partial\varepsilon_{(2)}}{\partial\alpha}\Big|_{\alpha=\beta_c}$ is given in the following lemma.

Lemma 2: The rate of change of $\varepsilon_{(2)}$ in $\mathcal{R}$, evaluated at the mid-point $\alpha=\beta_c$ is given by

$$\frac{\partial\varepsilon_{(2)}}{\partial\alpha}\Big|_{\alpha=\beta_c} = \frac{4\beta_c}{(1-\beta_c^2)^5}\Delta^4 + O(\Delta^6). \tag{26}$$

The complete derivation for this result is given in Appendix B. With the results of Lemma 1 and Lemma 2, we can use Proposition 1 to state the following theorem.

Theorem 1: An upper bound on the worst-case (highest) projection error in $\mathcal{R}$ is given by

$$B_\varepsilon = \frac{1}{(1-\beta_c^2)^4}\Delta^4 + \frac{4|\beta_c|}{(1-\beta_c^2)^5}\Delta^5 + O(\Delta^7). \tag{27}$$

This result follows from directly substituting (25) and (26) in (24) of Proposition 1. Note that a tighter bound on the true maximum error $\varepsilon_{(2)}^{(\max)}$ can be obtained by evaluating the RHS of (24) at an $\alpha=\alpha^*$ that is closer to the true maximizing point $\alpha_{(\max)}$, instead of at the mid-point $\alpha=\beta_c$. However, it can be shown that such a tighter bound also exhibits a minimum dependence of Δ^4. With either upper bound, the conclusion is that the worst-case projection error in $\mathcal{R}$ obeys a scaling law versus Δ with the minimum exponent 4 and no lower than that, i.e., the error scales at least as Δ^4, which is a significant result.

C. Deriving the Optimum ESN Pole Distribution

Theorem 1 expresses an upper bound on the approximation error of an LTI system pole α using only two ESN poles β_1 and β_2, as a function of the distance between the poles $\Delta=\frac{|\beta_2-\beta_1|}{2}$. Now, we revert to the problem of approximating α using M ESN poles $\{\beta_m\}_{m=1}^M$ that are "configured" by sampling them in an i.i.d. manner from the PDF $p_{\mathrm{B}}(\cdot)$. As $M\to\infty$, we now define a neighborhood $\mathcal{R}$ as an infinitesimally small interval over $(-\alpha_0,\alpha_0)$ in which the PDF $p_{\mathrm{B}}(\cdot)$ is constant with value $p_{\mathrm{B}}(\mathcal{R})$. We denote the length of this interval as $|\mathcal{R}|$. Then, for a particular realization of α say $\alpha^{(\mathcal{R})}$ lying inside $\mathcal{R}$, the nearest neighbors approximation states that the approximation error is given by the two ESN poles say $\beta^{(1,\mathcal{R})}$ and $\beta^{(2,\mathcal{R})}$ that are closest to $\alpha^{(\mathcal{R})}$. Denote the corresponding minimum distance between them as $\Delta^{(\mathcal{R})}=\frac{|\beta^{(2,\mathcal{R})}-\beta^{(1,\mathcal{R})}|}{2}$. The upper bound on the highest error in $\mathcal{R}$ is given by (27), which we write as $B_\varepsilon^{(\mathcal{R})}(\Delta(\mathcal{R}))$. Then, the contribution of this particular realization $\alpha^{(\mathcal{R})}$ to the average approximation error across all realizations of α is given by $C^{(\mathcal{R})}=p_{\mathrm{A}}(\alpha^{(\mathcal{R})})\cdot|\mathcal{R}|\cdot B_\varepsilon^{(\mathcal{R})}$. To satisfy the min-max

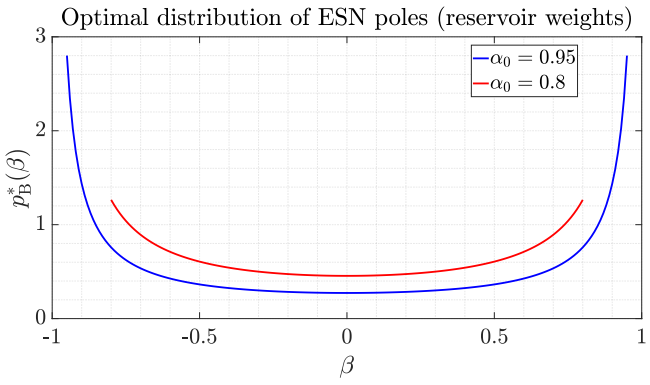

Fig. 7. Optimum PDF $p_{\mathrm{B}}^*(\beta)$ curves for $\alpha_0 = 0.95$ and $\alpha_0 = 0.8$.

optimization objective of (20) and thus, that of (21), we require that $C^{(\mathcal{R})}$ remain constant across all such neighborhoods, i.e., for any two neighborhoods $\mathcal{R}$ and $\mathcal{R}'$, we require $C^{(\mathcal{R})} = C^{(\mathcal{R}')}$. Since $p_{\mathrm{A}}(\cdot)$ is constant and as $|\mathcal{R}|$ does not depend on M or $\Delta(\mathcal{R})$, we require $B_\varepsilon^{(\mathcal{R})} = B_\varepsilon^{(\mathcal{R}')}$. Using only the leading terms of (27), this becomes

$$\begin{aligned} \frac{\left(\Delta^{(\mathcal{R})}\right)^4}{(1-\beta_c(\mathcal{R})^2)^4} &= \frac{\left(\Delta^{(\mathcal{R}')}\right)^4}{(1-\beta_c(\mathcal{R}')^2)^4}, \\ \Rightarrow \left(\Delta^{(\mathcal{R})}\right)^4 &\propto (1-\beta_c(\mathcal{R})^2)^4, \\ \Rightarrow \Delta^{(\mathcal{R})} &\propto (1-\beta_c(\mathcal{R})^2). \end{aligned} \tag{28}$$

Now, $\Delta^{(\mathcal{R})} \propto \frac{1}{M \cdot p_{\mathrm{B}}(\mathcal{R})}$. Thus, the optimum PDF $p_{\mathrm{B}}^*(\cdot)$ must vary in $\mathcal{R}$ as

$$p_{\mathrm{B}}^*(\mathcal{R}) \propto \frac{1}{1-\beta_c(\mathcal{R})^2}. \tag{29}$$

Since this relationship must hold in every infinitesimally small $\mathcal{R}$, we can write the PDF $p_{\mathrm{B}}^*(\cdot)$ in terms of the realization β of the random variable representing an ESN pole. Hence, we replace $\beta_c(\mathcal{R})$ with β to write the optimum $p_{\mathrm{B}}^*(\beta)$ for the "global" allocation of ESN poles as

$$p_{\mathrm{B}}^*(\beta) = \frac{1}{C}\frac{1}{(1-\beta^2)}, \tag{30}$$

where the PDF normalizing constant C is found by solving $\int_{-\alpha_0}^{\alpha_0} \frac{C}{1-\beta^2} d\beta = 1$ for $|\alpha_0| < 1$, giving $C = \log\left(\frac{1+\alpha_0}{1-\alpha_0}\right)$. As an example for $\alpha_0 = 0.95$, $C = 3.6636$ and the optimum ESN pole (reservoir weight) distribution is

$$p_{\mathrm{B}}^*(\beta) = \frac{0.273}{1-\beta^2}. \tag{31}$$

The optimum PDF curves for $\alpha_0 = 0.95$ and $\alpha_0 = 0.8$ are plotted in Fig. 7. Recognizing that $\Delta \propto \frac{1}{M}$, where M is the number of neurons in the reservoir, Theorem 1 provides a practical scaling law for the ESN projection error and by extension, the training loss, i.e., $\varepsilon_{(2)}^{(\max)} \propto \frac{1}{M^4}$. Such a direct scaling relationship of the training loss as a function of the model size is currently not available for more traditional neural network architectures.

D. Incorporating Prior Distributions on Pole of LTI System

In the derivation of $p_{\mathrm{B}}^*(\beta)$ in Section III-C, the prior distribution of the pole of the unknown LTI system was assumed to be uniform, i.e., $\alpha \sim \mathcal{U}(-\alpha_0, \alpha_0)$. However, the PDF of the optimum distribution for the poles $\{\beta_m\}$ can be adjusted for any other prior distribution of α. This result is stated in the following corollary.

Corollary 1.1: Given an optimum probability density function (PDF) $p_{\mathrm{B}}^*(\cdot)$ of the ESN poles $\{\beta_m\}$ for the unknown LTI system pole α distributed uniformly as $\alpha \sim p_{\mathrm{A}}(\cdot) \triangleq \mathcal{U}(-\alpha_0, \alpha_0)$, the optimum ESN pole distribution changes to $q_{\mathrm{B}}^*(\cdot) \propto p_{\mathrm{B}}^*(\cdot) \cdot (q_{\mathrm{A}}(\cdot))^{1/4}$, if the prior distribution on α changes from $p_{\mathrm{A}}(\cdot)$ to $q_{\mathrm{A}}(\cdot)$.

We provide a sketch of a proof for this result using the same argument as that in Section III-C. For $\alpha \sim q_{\mathrm{A}}(\cdot)$, where $q_{\mathrm{A}}(\cdot)$ is a non-uniform PDF, the contribution of a particular realization $\alpha^{(\mathcal{R})}$ to the average approximation error across all realizations of α is now $C^{(\mathcal{R})} = q_{\mathrm{A}}(\alpha^{(\mathcal{R})}) \cdot |\mathcal{R}| \cdot B_\varepsilon^{(\mathcal{R})}$. Optimizing the min-max objective of (20) requires $C^{(\mathcal{R})} = C^{(\mathcal{R}')}$ for any two neighborhoods $\mathcal{R}$ and $\mathcal{R}'$. However, since $q_{\mathrm{A}}(\cdot)$ is no longer constant across neighborhoods, this becomes

$$q_{\mathrm{A}}\left(\alpha^{(\mathcal{R})}\right) \cdot B_\varepsilon^{(\mathcal{R})} = q_{\mathrm{A}}\left(\alpha^{(\mathcal{R}')}\right) \cdot B_\varepsilon^{(\mathcal{R}')}. \tag{32}$$

Using only the leading terms in $B_\varepsilon^{(\mathcal{R})}$ and $B_\varepsilon^{(\mathcal{R}')}$, we get

$$\begin{aligned} q_{\mathrm{A}}\left(\alpha^{(\mathcal{R})}\right) \cdot \frac{\left(\Delta^{(\mathcal{R})}\right)^4}{(1-\beta_c(\mathcal{R})^2)^4} &= q_{\mathrm{A}}(\alpha^{(\mathcal{R}')}) \cdot \frac{\left(\Delta^{(\mathcal{R}')}\right)^4}{(1-\beta_c(\mathcal{R}')^2)^4}, \\ \Rightarrow \left(\Delta^{(\mathcal{R})}\right)^4 &\propto \frac{\left(1-\beta_c(\mathcal{R})^2\right)^4}{q_{\mathrm{A}}(\alpha^{(\mathcal{R})})}. \end{aligned} \tag{33}$$

Recognizing that $\Delta^{(\mathcal{R})} \propto \frac{1}{M \cdot q_{\mathrm{B}}(\mathcal{R})}$, we get

$$q_{\mathrm{B}}^*(\mathcal{R}) \propto \frac{(q_{\mathrm{A}}(\alpha^{(\mathcal{R})}))^{1/4}}{(1-\beta_c(\mathcal{R})^2)}. \tag{34}$$

Thus, the modified optimum PDF $q_{\mathrm{B}}^*(\cdot)$ for a global allocation of ESN poles can be written in terms of a general pole realization β, similar to Section III-C as

$$q_{\mathrm{B}}^*(\beta) \propto \frac{(q_{\mathrm{A}}(\beta))^{1/4}}{(1-\beta^2)} = p_{\mathrm{B}}^*(\beta)(q_{\mathrm{A}}(\beta))^{1/4}, \tag{35}$$

giving the result in Corollary 1.1.

Thus, if the system pole α follows a known non-uniform distribution $q_{\mathrm{A}}(\cdot)$, the "optimum" distribution to sample the ESN reservoir weights from is not simply $q_{\mathrm{A}}(\cdot)$ itself, but is rather a function of $q_{\mathrm{A}}(\cdot)$ which is further skewed by the universal optimum PDF $p_{\mathrm{B}}^*(\cdot)$. This is an important insight which informs that configuring the ESN reservoir weights according to the same distribution as the LTI system pole is in fact sub-optimal and a better initialization strategy exists.

E. Extension to Higher-Order LTI Systems

In the preceding sections, we have considered the atomic problem of approximating a first-order IIR system having a single pole using an ESN consisting of randomly selected

reservoir weights (poles) and trained output weights. We now generalize the target function to a higher-order LTI system, in particular, a linear combination of first-order poles [45], i.e., $\mathbf{s}_u = \sum_{k=1}^{K} v_k \mathbf{s}_{\alpha_k}$, for some weights $v_k \in \mathbb{R}$, $k = 1, \ldots, K$. This is written in the transform domain as

$$S_u(z) = \sum_{k=1}^{K} \frac{v_k}{1 - \alpha_k z^{-1}}. \tag{36}$$

We would like to approximate this higher-order system with an ESN consisting of a random collection of poles $\{\beta_{m,k}\}$ corresponding to each system pole realization α_k. This approximation can be written as

$$S_u(z) \approx \sum_{k=1}^{K} \left(\sum_{m=1}^{M} \frac{W_{m,k}}{1 - \beta_{m,k}} \right). \tag{37}$$

Denoting the projection error incurred in approximating each first-order component $\mathbf{s}_{\alpha_k}$ as ε_k, we recognize that an upper bound on $\mathsf{VAR}(\varepsilon_k)$ has been obtained as $\mathsf{VAR}(\varepsilon_k) \leq B_{\varepsilon_k}$ in Theorem 1. Then, the variance of the total approximation error ε across all K poles is given by $\mathsf{VAR}(\varepsilon) = \mathsf{VAR}(\sum_{k=1}^{K} \varepsilon_k)$. Since the LTI system poles $\{\alpha_k\}$ are not independent in general, an upper bound on $\mathsf{VAR}(\varepsilon)$ can be obtained as

$$\mathsf{VAR}(\varepsilon) \leq K^2 \cdot \max_k \left(\mathsf{VAR}(\varepsilon_k) \right) \leq K^2 \cdot \max_k \left(B_{\varepsilon_k} \right). \tag{38}$$

Therefore, the optimum PDF minimizing the approximation error of a single first-order IIR system also minimizes the same for a linear combination of such poles.

F. Reservoir With Random and Sparse Interconnections

The conventional ESN in state-of-the-art practice uses a reservoir that is sparsely connected with randomly weighted interconnections between the constituent neurons. In the case of non-interconnected neurons, the reservoir weights matrix is $\mathbf{W}_{\text{res}} = \text{diag}(\{\beta_m\}_{m=1}^{M})$. However, this is not the case for a sparsely interconnected reservoir. Performing the eigenvalue decomposition of the general sparse (non-diagonal) $\mathbf{W}_{\text{res}}$,

$$\mathbf{W}_{\text{res}} = \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^{-1}, \tag{39}$$

where $\mathbf{Q} \in \mathbb{C}^{M \times M}$ is the matrix containing the eigenvectors of $\mathbf{W}_{\text{res}}$. For a non-interconnected reservoir, $\mathbf{W}_{\text{res}} = \mathbf{\Lambda}$ and $\mathbf{Q} = \mathbf{I}_M$. On the other hand, for a random and sparsely interconnected reservoir, the elements of $\mathbf{W}_{\text{res}}$ induce a corresponding distribution in $\mathbf{\Lambda}$ such that the elements of $\mathbf{\Lambda}$ may no longer be independent [49]. However, the projection error due to a general random sparsely interconnected reservoir ESN will always be lower bounded by the projection error due to a non-interconnected reservoir with its weights sampled i.i.d. from $p_{\text{B}}^*(\cdot)$. Although $p_{\text{B}}^*(\cdot)$ has been derived for the case of non-interconnected neurons, we will show in this section that even with random and sparse (weighted) interconnections between the neurons, where the recurrent and interconnection weights are drawn from a uniform distribution, the projection error in this case is still lower bounded by the projection error with $\{\beta_m\} \overset{\text{i.i.d.}}{\sim} p_{\text{B}}^*(\cdot)$. This can be seen by invoking the state update and output equations for the linear ESN, i.e.,

$$\mathbf{x}_{\text{res}}[n] = \mathbf{W}_{\text{res}} \mathbf{x}_{\text{res}}[n-1] + \mathbf{W}_{\text{in}} \mathbf{x}_{\text{in}}[n] \tag{40}$$

$$\mathbf{x}_{\text{out}}[n] = \mathbf{W}_{\text{out}} \mathbf{x}_{\text{res}}[n] \tag{41}$$

Substituting (39) in (40), we get

$$\begin{aligned} &\mathbf{x}_{\text{res}}[n] = \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^{-1} \mathbf{x}_{\text{res}}[n-1] + \mathbf{W}_{\text{in}} \mathbf{x}_{\text{in}}[n], \\ \Rightarrow\ &\widetilde{\mathbf{x}}_{\text{res}}[n] = \mathbf{\Lambda} \widetilde{\mathbf{x}}_{\text{res}}[n-1] + \widetilde{\mathbf{W}}_{\text{in}} \mathbf{x}_{\text{in}}[n], \end{aligned} \tag{42}$$

where $\widetilde{\mathbf{x}}_{\text{res}}[n] \triangleq \mathbf{Q}^{-1} \mathbf{x}_{\text{res}}[n]$ and $\widetilde{\mathbf{W}}_{\text{in}} \triangleq \mathbf{Q}^{-1} \mathbf{W}_{\text{in}}$. Using $\mathbf{Q}\mathbf{Q}^{-1} = \mathbf{I}_M$ in (41), we get

$$\mathbf{x}_{\text{out}}[n] = \widetilde{\mathbf{W}}_{\text{out}} \widetilde{\mathbf{x}}_{\text{res}}[n], \tag{43}$$

where $\widetilde{\mathbf{W}}_{\text{out}} = \mathbf{W}_{\text{out}} \mathbf{Q}$.

Thus, a general *linear* ESN with random and sparse interconnections between its reservoir neurons can be diagonalized and the analysis for its optimization is the same as that for a reservoir without interconnections, i.e., for $\mathbf{W}_{\text{res}} = \mathbf{\Lambda}$. We will empirically show in Section V that a linear reservoir with random interconnections does not provide additional performance gain and is still bounded by the performance of the non-interconnected reservoir ESN with weights sampled from the optimal $p_{\text{B}}^*(\cdot)$. This conclusion holds in general for reservoirs with linear activation, i.e., the best performance for a reservoir with linear activation will only be achieved for the case of non-interconnected neurons with $\{\beta_m\}$ configured by sampling i.i.d. from $p_{\text{B}}^*(\cdot)$ In other words, $p_{\text{B}}^*(\cdot)$ is the optimum PDF to sample $\{\beta_m\}$ from only when the neurons are not interconnected.

Studying the impact of nonlinear activation to derive the optimum PDF, even with non-interconnected neurons can be challenging. Although local approximations of the state update equation around the zero state can be obtained using the Jacobian, which is an approach used in stability analysis [50], this is generally analytically tractable only for specific activation functions, e.g., the hyperbolic tangent (tanh) function. Alternative approaches may include incorporating the nonlinear activation by modeling the state update equation as a higher-order autoregressive process, up to an order that may admit tractable analysis towards the optimum PDF. Finally, operator theoretic methods [51] could be a possible solution for handling the nonlinear activation, however their tractability towards deriving the optimum PDF remains to be studied. The effect of nonlinear activation on random interconnections between neurons will be addressed in our future work.

IV. Training With Limited Samples

In the preceding sections, we have considered the orthogonal projection of an LTI system's impulse response on to the subspace spanned by the reservoir of the ESN, and solved the problem of finding the optimum basis for this subspace. The optimum output weights $\{W_m^*\}$ for the linear combination of these basis functions are given by (11). However, this makes use of the knowledge of the particular realization of α or viewed alternatively, requires infinitely many samples to learn $\{W_m^*\}$. In practice, however, we do not observe or know the true model of the system being simulated, but have access to

only a limited number of labeled input-output data samples. Under this scenario, the output weights $\mathbf{w} \triangleq [W_1\ W_2\ \ldots\ W_M]^T$ are trained with limited training data using the conventional approach of least squares optimization of the ℓ_2 regression loss. For a training sequence consisting of input-output pairs $\{(x_1, y_1), \ldots, (x_L, y_L)\}$, $\mathbf{w}$ is estimated as

$$\widehat{\mathbf{w}} = \left(\boldsymbol{y}^T \mathbf{X}_{\text{res}}^{\dagger}\right)^T, \tag{44}$$

where $\boldsymbol{y} \triangleq [y_1\ y_2\ \ldots\ y_L]^T \in \mathbb{R}^{L\times 1}$ is the ground truth and $\mathbf{X}_{\text{res}} \in \mathbb{R}^{M\times L}$ is the reservoir states matrix containing the state vector $\mathbf{x}_{\text{res}}[n]$ from $n = 1$ to $n = L$ in its columns. When multiple sequences are used for training, the training rule is modified as

$$\widehat{\mathbf{w}} = \left(\bar{\boldsymbol{y}}^T \bar{\mathbf{X}}_{\text{res}}^{\dagger}\right)^T, \tag{45}$$

where $\bar{\boldsymbol{y}} \in \mathbb{R}^{N_p L}$ is the concatenated ground truth across N_p training sequences, and $\bar{\mathbf{X}}_{\text{res}} \in \mathbb{R}^{M\times N_p L}$ is the concatenated reservoir states matrix. The availability of only a finite number of labeled training data samples leads to the well-known issue of *model selection*. In the context of ESNs, this translates into selecting an optimum reservoir size M such that the test loss is minimized while avoiding an excessively large reservoir size that may lead to overfitting. The Akaike Information Criterion (AIC) [52] is a well-known model selection criterion that penalizes large model sizes. The main AIC result can be written as

$$\begin{aligned}&\arg\min_{M} D\left(p_{X,Y}(x,y;\alpha)||p_{X,Y}(x,y;\beta_m,W_m)\right)\\ &= \arg\min_{M} D\left(\widehat{p}_{X,Y}(x,y;\alpha)||p_{X,Y}(x,y;\beta_m,W_m)\right) + \frac{M}{N_p L},\end{aligned} \tag{46}$$

where $p_{X,Y}(x, y; \alpha)$ denotes the true unknown joint distribution with parameter α from which the input-output sample pairs are generated, i.e., the unknown LTI system. $p_{X,Y}(x, y; \beta_m, W_m)$ denotes the joint distribution generated by the ESN model with parameters $\{\beta_m\}$ and $\{W_m\}$ and $D(p||q)$ denotes the Kullback-Leibler (KL) divergence between two probability distributions with PDFs $p(\cdot)$ and $q(\cdot)$. Since we cannot observe the true joint distribution $p_{X,Y}(x, y; \alpha)$ in practice and only observe a finite number of input-output samples, we only have access to the empirical joint distribution $\widehat{p}_{X,Y}(x, y; \alpha)$. Thus, the argument of the LHS of (46) is representative of the test loss, while the argument of the first term on the RHS of (46) is representative of the training loss computed using a finite number N_p of input-output pair sequences, for which a scaling law as a function of M has been derived in Theorem 1. The second term on the RHS $\frac{M}{N_p L}$ represents the overfitting penalty imposed by the AIC. Combining this observation with the result of Theorem 1, we get

$$\mathcal{L}_{\text{test}} \propto \frac{1}{M^4} + \frac{M}{N_p L} \tag{47}$$

With this relationship, we can derive an order for the optimum reservoir size M^* which minimizes the test loss $\mathcal{L}_{\text{test}}$. This is obtained by first setting

$$\frac{d\mathcal{L}_{\text{test}}}{dM} \propto -\frac{4}{M^5} + \frac{1}{N_p L} = 0. \tag{48}$$

Solving this, we can obtain an order of magnitude for the optimum reservoir size M^* as

$$M^* = O\left((N_p L)^{1/5}\right). \tag{49}$$

Note that this result does not give the exact reservoir size in terms of number of neurons, but is rather an approximation of the order of the optimum reservoir size needed to minimize the testing loss. Furthermore, the AIC is one of many model selection criteria, e.g., Bayesian Information Criterion (BIC), Generalized Information Criterion (GIC), among others [53]. However, such model selection criteria is beyond the scope of this paper. A statistical learning theory-inspired model selection criteria for ESN-based multi-antenna wireless symbol detection is developed in our previous work [42].

V. Numerical Evaluations

In this section, we provide numerical evaluations to validate the theoretical results derived in the preceding sections. Specifically, our objective is to experimentally verify the result of Theorem 1 and validate the optimality of the distribution $p_{\text{B}}^*(\cdot)$ for the reservoir weights under various scenarios.

A. Sampling From the Optimum Distributions

For the case of uniformly distributed system pole α, we use the Von Neumann rejection sampling method (accept-reject algorithm) [54] to draw i.i.d. samples from the optimal reservoir weights distribution $p_{\text{B}}^*(\cdot)$, as well as the modified optimum PDF $q_{\text{B}}^*(\cdot)$ for non-uniformly distributed α, as shall be seen in Section V-E. Alternatively, its empirical form [55] can avoid computing the PDF scaling constant.

B. Projection Error Scaling law From Theorem 1

The main result of Theorem 1 is a scaling law for the projection error as a function of the reservoir size in neurons. This is a key result that also translates to the rate of decrease in the training loss when training a standard *linear* ESN under limited training data. The projection error ε of (12) is simulated over 10^5 Monte-Carlo runs for $\alpha \sim \mathcal{U}(-0.95, 0.95)$ for an ESN with a non-interconnected reservoir, i.e., $\mathbf{W}_{\text{res}} = \text{diag}(\{\beta_m\}_{m=1}^M)$. The resulting plot of the empirical error versus M is shown in Fig. 8. We can observe that the simulated projection error using reservoir weights $\{\beta_m\}$ configured using the optimum PDF $p_{\text{B}}^*(\cdot)$ is significantly lower than the error obtained using reservoir weights drawn from $\mathcal{U}(-0.95, 0.95)$. Additionally, the empirical ε approximately displays an M^{-4} dependence when its reservoir weights are configured using $p_{\text{B}}^*(\beta)$, compared to approximately an M^{-2} dependence displayed when the weights are sampled from $\mathcal{U}(-0.95, 0.95)$, indicating a good match between theory and numerical evaluations.

In addition to plotting the projection error, we also validate the scaling law via the empirical sequence approximation error

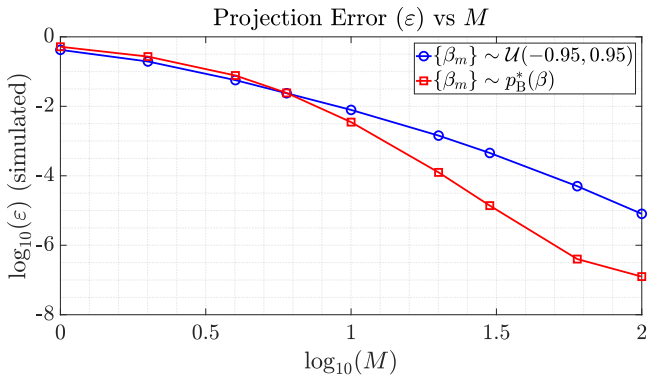

Fig. 8. Validation of the scaling law for the projection error (ε) of (12).

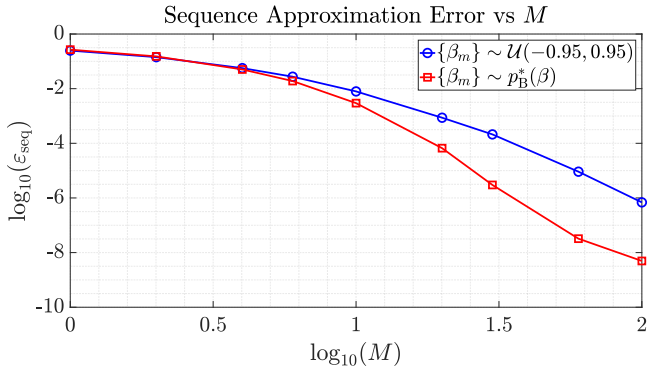

Fig. 9. Validation of the scaling law for the sequence approximation error ($\varepsilon_{\mathrm{seq}}$) for sequence length $L = 1000$.

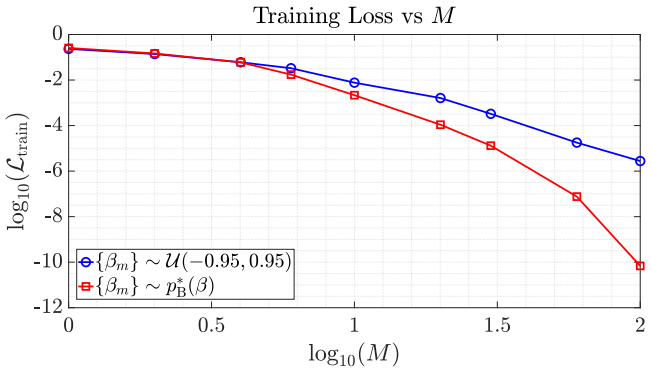

Fig. 10. Training loss versus reservoir size M for ESN trained with finite training samples.

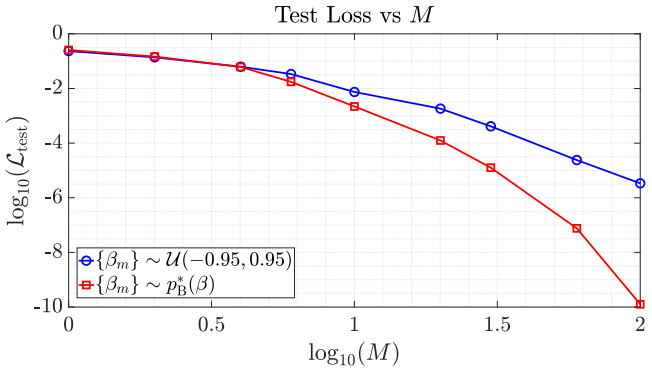

Fig. 11. Test loss versus reservoir size M for ESN trained with finite training samples.

$\varepsilon_{\mathrm{seq}}$, defined as

$$\varepsilon_{\mathrm{seq}} = \frac{1}{N_{\mathrm{sim}} L} \sum_{i=1}^{N_{\mathrm{sim}}} \left\| \mathbf{y}_{\mathrm{LTI}}^{(i)} - \mathbf{y}_{\mathrm{ESN}}^{(i)} \right\|_2^2, \tag{50}$$

where $\mathbf{y}_{\mathrm{LTI}}^{(i)} \in \mathbb{R}^L$ and $\mathbf{y}_{\mathrm{ESN}}^{(i)} \in \mathbb{R}^L$ are the sequences each of length L output by the unknown LTI system being simulated and by the ESN approximation respectively in the ith Monte-Carlo run. Note that the output weights $\mathbf{w} \in \mathbb{R}^M$ for the sequence approximation task are computed using (11), i.e., they are selected as the optimum values $\{W_m^*\}$ that result from orthogonal projection given the value of the realization of α in each run. This is plotted in Fig. 9 for a sequence length $L = 1000$ over $N_{\mathrm{sim}} = 10^5$ runs. As with the simulated projection error, Fig. 9 shows that $\varepsilon_{\mathrm{seq}}$ also exhibits a dependence of approximately M^{-4} for $\{\beta_m\} \sim p_{\mathrm{B}}^*(\cdot)$ and that of approximately M^{-2} for $\{\beta_m\} \sim \mathcal{U}(-0.95, 0.95)$. In summary, these numerical evaluations provide strong confirmation for the validity of the derived theoretical optimum distribution of the internal reservoir weights.

C. Training and Test Loss Under Limited Training Data

Recall that computing the projection error ε via (12) required knowledge of the particular realization of α in each run, or alternatively the availability of infinitely many training samples. However, with limited training data as in practice, we can verify that a similar scaling trend versus M and improvement in performance in terms of the training and test losses is obtained when configuring the weights using $p_{\mathrm{B}}^*(\cdot)$ compared to randomly generating them from $\mathcal{U}(-\alpha_0, \alpha_0)$. To validate this, the linear ESN is trained with $N_p = 1$ training sequence of length $L = 500$ samples, i.e., $\widehat{\mathbf{w}}$ is computed using (45). Next, it is tested with $N_d = 10$ test sequences of the same length. The empirical training loss $\mathcal{L}_{\mathrm{train}} \triangleq \frac{1}{N_{\mathrm{sim}} N_p L} \sum_{i=1}^{N_{\mathrm{sim}}} \|\bar{\mathbf{y}}_{\mathrm{LTI,train}}^{(i)} - \bar{\mathbf{y}}_{\mathrm{ESN,train}}^{(i)}\|_2^2$ is plotted in Fig. 10, where $\bar{\mathbf{y}}_{\mathrm{LTI,train}}^{(i)} \in \mathbb{R}^{N_p L}$ is the concatenated training output from the LTI system and $\bar{\mathbf{y}}_{\mathrm{ESN,train}}^{(i)} \in \mathbb{R}^{N_p L}$ is the concatenated ESN output during training respectively in the i^{th} Monte-Carlo run, with $N_{\mathrm{sim}} = 5 \times 10^4$.

We can observe that the ESN with optimally sampled reservoir weights shows a significantly lower training loss and approximately obeys the M^{-4} scaling law. The more important practical performance metric, namely the empirical test loss $\mathcal{L}_{\mathrm{test}} \triangleq \frac{1}{N_{\mathrm{sim}} N_d L} \sum_{i=1}^{N_{\mathrm{sim}}} \|\bar{\mathbf{y}}_{\mathrm{LTI,test}}^{(i)} - \bar{\mathbf{y}}_{\mathrm{ESN,test}}^{(i)}\|_2^2$ is plotted in Fig. 11, where $\bar{\mathbf{y}}_{\mathrm{LTI,test}}^{(i)} \in \mathbb{R}^{N_d L}$ is the concatenated LTI system output during test and $\bar{\mathbf{y}}_{\mathrm{ESN,test}}^{(i)} \in \mathbb{R}^{N_d L}$ is the concatenated ESN output during test respectively in the i^{th} Monte-Carlo run. Therefore, the derived optimum PDF for the reservoir weights can provide up to 4 orders of magnitude improvement in the test loss at higher reservoir sizes, indicating a huge performance gain that

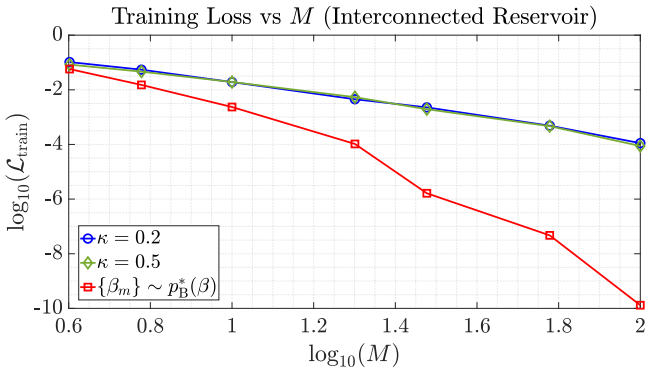

Fig. 12. Training loss versus reservoir size M under finite training samples for ESN with random interconnections between neurons.

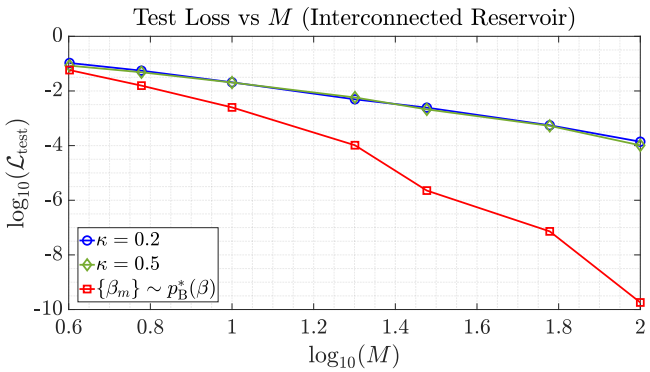

Fig. 13. Test loss versus reservoir size M under finite training samples for ESN with random interconnections between neurons.

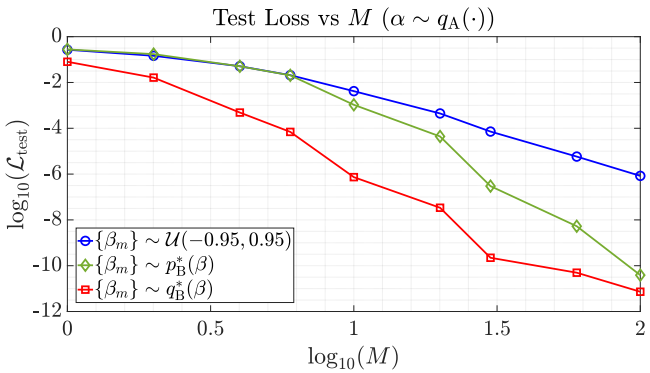

Fig. 14. Test loss versus reservoir size M with a changed prior PDF on α given by $q_{\text{A}}(\cdot)$ (non-uniform distribution).

can be achieved without any additional training complexity. Note that in the simulation of a simple system such as a first-order IIR system, it would typically take a model of a significantly larger size, i.e., reservoir with many more neurons to start observing the overfitting effect in the test loss $\mathcal{L}_{\text{test}}$.

D. Interconnected Linear Activation Reservoir

In order to validate our finding from Section III-F that interconnections between neurons in the reservoir is equivalent to a non-interconnected reservoir with modified input and output weights matrices, we replicate the evaluations of Section V-C, but with a non-diagonal $\mathbf{W}_{\text{res}}$, i.e., with random and sparse interconnections between the reservoir neurons. The sparsity of connections is controlled via the hyperparameter 'sparsity' (denoted as κ) which represents the probability of each element of $\mathbf{W}_{\text{res}}$ being 0. Furthermore, the spectral radius of $\mathbf{W}_{\text{res}}$ is set to 0.95, i.e., $\rho(\mathbf{W}_{\text{res}}) = \max|\lambda(\mathbf{W}_{\text{res}})| = 0.95$ for the cases of random and sparsely interconnected reservoirs, with corresponding weights drawn i.i.d. from $\mathcal{U}(-0.95, 0.95)$.

From both Figs. 12 and 13, we can observe that the ESN with a non-interconnected reservoir with weights configured using the optimum PDF $p_{\text{B}}^*(\cdot)$ greatly outperforms the ESN model with a sparsely interconnected reservoir with weights randomly generated from $\mathcal{U}(-0.95, 0.95)$, i.e., the state-of-the-art practice. At higher reservoir sizes, e.g., $M = 100$, we can see a gain of up to 6 orders of magnitude in the test loss. Additionally, for a fixed spectral radius, a change in the sparsity of the reservoir from $\kappa = 0.2$ to $\kappa = 0.5$ does not result in any observable change in the trends of the training and the test losses. This confirm our hypothesis from Section III-F that for linear activation, random interconnections between neurons do not provide additional performance gain.

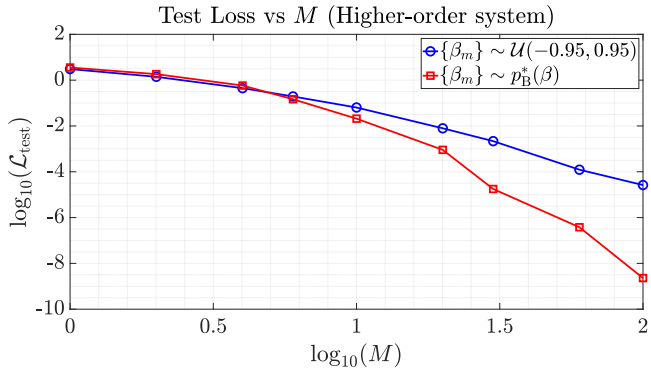

Fig. 15. Test loss versus reservoir size M for a 5-th order LTI system.

E. Simulating Change in Prior Distribution of System Pole

In this section, the result of Corollary 1.1 is validated through simulations. Specifically, we consider a changed prior distribution of α given by $q_{\text{A}}(\cdot) \triangleq \mathcal{N}(0.7, 10^{-2})$. Following the same settings as in the previous sections for data-driven training of the output weights, the ESN reservoir weights are now configured by sampling from the modified optimum PDF $q_{\text{B}}^*(\cdot)$ using the result of Corollary 1.1. The corresponding test loss for this experiment is plotted in Fig. 14. Furthermore, we also plot the test loss for the case of $\{\beta_m\}$ initialized from $p_{\text{B}}^*(\cdot)$, which is optimized for $\alpha \sim \mathcal{U}(-0.95, 0.95)$ and not for $\alpha \sim q_{\text{A}}(\cdot)$. Compared to randomly generating the reservoir weights $\{\beta_m\}$ from $\mathcal{U}(-0.95, 0.95)$, configuring them using $p_{\text{B}}^*(\cdot)$ or $q_{\text{B}}^*(\cdot)$ both result in much improved performance. However, the performance achieved with $q_{\text{B}}^*(\cdot)$ which is optimized for the modified prior PDF $q_{\text{A}}(\cdot)$ is even better than that using $p_{\text{B}}^*(\cdot)$ which is optimized for a uniform prior PDF $p_{\text{A}}(\cdot)$. This clearly validates

Corollary 1.1 and demonstrates the value in adapting the reservoir initialization strategy to the available domain knowledge.

F. Simulating Higher-Order LTI Systems

In this section, we empirically verify the optimality of $p_{\mathrm{B}}^{*}(\cdot)$ when approximating higher-order LTI systems of the form given in (36), i.e., a linear combination of first-order poles. Specifically, we consider a 5-th order system by substituting $K = 5$ in $\mathbf{s}_u = \sum_{k=1}^{K} \mathbf{s}_{\alpha_k}$, where $\{\alpha_k\}$ are sampled i.i.d. from $\mathcal{U}(-0.95, 0.95)$. The corresponding test loss is plotted in Fig. 15. Similarly to the first-order system approximation task, we can see an improvement of up to 4 orders of magnitude at moderate to higher reservoir sizes. This validates the applicability of the derived optimum PDF to higher-order LTI systems.

VI. Conclusion and Future Work

In this work, we have introduced a clear signal processing approach to understand the echo state network (ESN), a powerful architecture of the Reservoir Computing (RC) family, belonging to the broader class of randomized recurrent neural networks. Employing the linear ESN to approximate a simple linear time-invariant (LTI) system, we provide a precise scaling law obeyed by the approximation error and a complete analytical characterization of the optimum probability density function (PDF) that can be used to configure the ESN's reservoir weights, which are otherwise randomly generated in a pre-determined and arbitrary fashion in state-of-the-art practice. Numerical evaluations demonstrate the optimality of the derived optimum PDF by showing a gain of up to 4 orders of magnitude at moderate to high reservoir sizes. This demonstrates the practical applicability and realizable performance gains by virtue of the analysis in this work. Extension of this analysis to complex-valued ESNs and developing an understanding of the impact of nonlinear activation is part of future investigation. Additionally, deriving the optimum weights distribution for the wireless channel equalization task given statistical knowledge of the channel is also included in future work.

Appendix A
Proof of Lemma 1

Proof: Substituting $\alpha = \beta_c \triangleq \frac{1}{2}(\beta_1 + \beta_2)$ in (23) and the substitution $\beta_1 = \beta_c - \Delta$ and $\beta_2 = \beta_c + \Delta$, we can arrive at the following expression after some manipulation,

$$\begin{aligned}\varepsilon_{(2)}^{(\mathrm{mid})} &= \frac{\Delta^4}{\left(1 + \beta_c^4 - \beta_c^2(2 + \Delta^2)\right)^2}, \\ &= \frac{\Delta^4}{(1-\beta_c^2)^4 \left(1 - \frac{2\beta_c^2\Delta^2}{(1-\beta_c^2)^2} + \frac{\beta_c^4\Delta^4}{(1-\beta_c^2)^4}\right)}. \end{aligned} \tag{51}$$

To perform a Taylor series expansion up to the second power for the term $C_4^{(\mathrm{mid})} \triangleq \frac{1}{\left(1 - \frac{2\beta_c^2\Delta^2}{(1-\beta_c^2)^2} + \frac{\beta_c^4\Delta^4}{(1-\beta_c^2)^4}\right)}$, recall the Taylor series expansion for $\frac{1}{1+x}$ for $x \ll 1$ given by

$$\frac{1}{1+x} \approx 1 - x + x^2 + O(x^3). \tag{52}$$

Applying this to $C_4^{(\mathrm{mid})}$, we obtain

$$\begin{aligned} C_4^{(\mathrm{mid})} &\approx 1 - \left(\frac{\beta_c^4\Delta^4}{(1-\beta_c^2)^4} - \frac{2\beta_c^2\Delta^2}{(1-\beta_c^2)^2}\right) \\ &\quad + \left(\frac{\beta_c^4\Delta^4}{(1-\beta_c^2)^4} - \frac{2\beta_c^2\Delta^2}{(1-\beta_c^2)^2}\right)^2, \\ &= 1 + \frac{2\beta_c^2\Delta^2}{(1-\beta_c^2)^2} + \frac{3\beta_c^4\Delta^4}{(1-\beta_c^2)^4} + O(\Delta^6). \end{aligned} \tag{53}$$

Using this approximation in (51), we arrive at Lemma 1,

$$\varepsilon_{(2)}^{(\mathrm{mid})} = \frac{1}{(1-\beta_c^2)^4}\Delta^4 + O(\Delta^6). \tag{54}$$

■

Appendix B
Proof of Lemma 2

Proof: With the substitutions $\beta_1 = \beta_c - \Delta$, $\beta_2 = \beta_c + \Delta$ and a sequence of algebraic manipulations, we can arrive at the following expression for the derivative of the neighborhood error w.r.t. α, evaluated at the mid-point $\alpha = \beta_c \triangleq \frac{1}{2}(\beta_1 + \beta_2)$,

$$\left.\frac{\partial \varepsilon_{(2)}}{\partial \alpha}\right|_{\alpha=\beta_c} = \frac{4\beta_c(1 - \beta_c^2 + \Delta^2)}{\left((1-\beta_c^2)^2 - \beta_c^2\Delta^2\right)^3}\Delta^4. \tag{55}$$

Since a power series expansion for $\left.\frac{\partial \varepsilon_{(2)}}{\partial \alpha}\right|_{\alpha=\beta_c}$ in terms of Δ is required, we first obtain a Taylor series expansion for the term $\frac{1}{((1-\beta_c^2)^2 - \beta_c^2\Delta^2)^3}$ as follows. Rewriting this term as

$$\begin{aligned} &\frac{1}{\left((1-\beta_c^2)^2 - \beta_c^2\Delta^2\right)^3} \\ &\quad = \frac{1}{(1-\beta_c^2)^6\left(1 - \frac{\beta_c^6\Delta^6}{(1-\beta_c^2)^6} + \frac{3\beta_c^4\Delta^4}{(1-\beta_c^2)^4} - \frac{3\beta_c^2\Delta^2}{(1-\beta_c^2)^2}\right)}. \end{aligned} \tag{56}$$

Next, we perform an expansion for the term

$$C_6^{(\mathrm{bound})} \triangleq \frac{1}{1 + \left(\frac{3\beta_c^4\Delta^4}{(1-\beta_c^2)^4} - \frac{3\beta_c^2\Delta^2}{(1-\beta_c^2)^2} - \frac{\beta_c^6\Delta^6}{(1-\beta_c^2)^6}\right)}, \tag{57}$$

using the Taylor series $\frac{1}{1+x} \approx 1 - x$, for small x. Thus,

$$C_6^{(\mathrm{bound})} \approx 1 + \frac{3\beta_c^2\Delta^2}{(1-\beta_c^2)^2} - \frac{3\beta_c^4\Delta^4}{(1-\beta_c^2)^4} + \frac{\beta_c^6\Delta^6}{(1-\beta_c^2)^6}. \tag{58}$$

Simplifying (55), we get

$$\left.\frac{\partial \varepsilon_{(2)}}{\partial \alpha}\right|_{\alpha=\beta_c} = \frac{4\beta_c(1-\beta_c^2)\Delta^4 + 4\beta_c\Delta^6}{(1-\beta_c^2)^6} C_6^{(\mathrm{bound})}. \tag{59}$$

Substituting with the Taylor expansion for $C_6^{(\mathrm{bound})}$ from (58), it follows that

$$\begin{aligned} \left.\frac{\partial \varepsilon_{(2)}}{\partial \alpha}\right|_{\alpha=\beta_c} &\approx \left(\frac{4\beta_c(1-\beta_c^2)\Delta^4 + 4\beta_c\Delta^6}{(1-\beta_c^2)^6}\right) \\ &\quad \times \left(1 + \frac{3\beta_c^2\Delta^2}{(1-\beta_c^2)^2} - \frac{3\beta_c^4\Delta^4}{(1-\beta_c^2)^4}\right. \end{aligned}$$

$$+\frac{\beta_c^6\Delta^6}{(1-\beta_c^2)^6}\Bigg),$$
$$=\frac{4\beta_c}{(1-\beta_c^2)^5}\Delta^4+O(\Delta^6), \tag{60}$$

yielding the result of Lemma 2. ■

REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[3] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE 2013 Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 6645–6649.

[4] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.

[5] K. Funahashi and Y. Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," *Neural Netw.*, vol. 6, no. 6, pp. 801–806, 1993.

[6] M. M. Agüero-Torales, J. I. Abreu Salas, and A. G. López-Herrera, "Deep learning and multilingual sentiment analysis on social media data: An overview," *Appl. Soft Comput.*, vol. 107, 2021, Art. no. 107373.

[7] K. Cho et al., "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1724–1734.

[8] D. Güera and E. J. Delp, "Deepfake video detection using recurrent neural networks," in *Proc. IEEE 15th Int. Conf. Adv. Video Signal Based Surveill.*, 2018, pp. 1–6.

[9] B. Zhao, X. Li, and X. Lu, "CAM-RNN: Co-attention model based RNN for video captioning," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5552–5565, Nov. 2019.

[10] S. S. Mosleh, L. Liu, C. Sahin, Y. R. Zheng, and Y. Yi, "Brain-inspired wireless communications: Where reservoir computing meets MIMO-OFDM," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4694–4708, Oct. 2018.

[11] B. Peng et al., "RWKV: Reinventing RNNs for the transformer era," *Findings of the Assoc. Comput. Linguistics: EMNLP 2023*. Association for Computational Linguistics, Dec. 2023, pp. 14048–14077. [Online]. Available: https://aclanthology.org/2023.findings-emnlp.936.

[12] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1310–1318.

[13] P. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[16] S. Hochreiter and J. Schmidhuber, "LSTM can solve hard long time lag problems," in *Proc. Adv. Neural Inf. Process. Syst.*, 1997, pp. 473–479.

[17] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, 2020, Art. no. 132306.

[18] C. Gallicchio, A. Micheli, and P. Tino, "Randomized recurrent neural networks," in *Proc. Eur. Symp. Artif. Neural Netw.*, 2018. [Online]. Available: https://www.esann.org/sites/default/files/proceedings/legacy/es2018-6.pdf

[19] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, 2009.

[20] M. Lukoševičius, *A Practical Guide to Applying Echo State Networks*. Berlin Heidelberg: Springer, 2012, pp. 659–686.

[21] X. Hinaut and P. F. Dominey, "On-Line processing of grammatical structure using reservoir computing," in *Proc. Artif. Neural Netw. Mach. Learn.–ICANN*, 2012, pp. 596–603.

[22] A. Juven and X. Hinaut, "Cross-situational learning with reservoir computing for language acquisition modelling," in *Proc. Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.

[23] A. Jalalvand, G. Van Wallendael, and R. Van De Walle, "Real-time reservoir computing network-based systems for detection tasks on visual contents," in *Proc. 7th Int. Conf. Comput. Intell., Commun. Syst. Netw.*, 2015, pp. 146–151.

[24] W.-J. Wang, Y. Tang, J. Xiong, and Y.-C. Zhang, "Stock market index prediction based on reservoir computing models," *Expert Syst. With Appl.*, vol. 178, 2021, Art. no. 115022.

[25] Z. Zhou, L. Liu, and H.-H. Chang, "Learning for detection: MIMO-OFDM symbol detection through downlink pilots," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3712–3726, Jun. 2020.

[26] Z. Zhou, L. Liu, S. Jere, J. Zhang, and Y. Yi, "RCNet: Incorporating structural information into deep RNN for online MIMO-OFDM symbol detection with limited training," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3524–3537, Jun. 2021.

[27] J. Xu, Z. Zhou, L. Li, L. Zheng, and L. Liu, "RC-Struct: A structure-based neural network approach for MIMO-OFDM detection," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, pp. 7181–7193, Sep. 2022.

[28] H.-P. Ren, H.-P. Yin, C. Bai, and J.-L. Yao, "Performance improvement of chaotic baseband wireless communication using echo state network," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6525–6536, Oct. 2020.

[29] H.-H. Chang, L. Liu, and Y. Yi, "Deep echo state Q-network (DEQN) and its application in dynamic spectrum sharing for 5G and beyond," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 3, pp. 929–939, Mar. 2022.

[30] F. Da Ros, S. M. Ranzini, H. Bülow, and D. Zibar, "Reservoir-computing based equalization with optical pre-processing for short-reach optical transmission," *IEEE J. Sel. Topics Quantum Electron.*, vol. 26, no. 5, Sep./Oct. 2020, Art. no. 7701912.

[31] H. Dai and Y. K. Chembo, "Classification of IQ-modulated signals based on reservoir computing with narrowband optoelectronic oscillators," *IEEE J. Quantum Electron.*, vol. 57, no. 3, Jun. 2021, Art. no. 5000408.

[32] R. Shafin, L. Liu, V. Chandrasekhar, H. Chen, J. Reed, and J. C. Zhang, "Artificial intelligence-enabled cellular networks: A critical path to beyond-5G and 6G," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 212–217, Apr. 2020.

[33] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, *Layer-Wise Relevance Propagation: An Overview*. Cham, Switzerland: Springer, 2019, pp. 193–209.

[34] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," 2017, *arXiv:1703.00810*.

[35] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4768–4777.

[36] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller, "How to explain individual classification decisions," *J. Mach. Learn. Res.*, vol. 11, pp. 1803–1831, Aug. 2010.

[37] M. C. Ozturk, D. Xu, and J. C. Príncipe, "Analysis and design of echo state networks," *Neural Comput.*, vol. 19, no. 1, pp. 111–138, Jan. 2007.

[38] E. Bollt, "On explaining the surprising success of reservoir computing forecaster of chaos? The universal machine learning dynamical system with contrast to VAR and DMD," *Chaos*, vol. 31, 2021, Art. no. 013108.

[39] A. G. Hart, J. L. Hook, and J. H. Dawes, "Echo state networks trained by tikhonov least squares are L2(μ) approximators of ergodic dynamical systems," *Physica D: Nonlinear Phenomena*, vol. 421, 2021, Art. no. 132882.

[40] A. Haluszczynski and C. Räth, "Good and bad predictions: Assessing and improving the replication of chaotic attractors by means of reservoir computing," *Chaos*, vol. 29, no. 10, 2019, Art. no. 103143.

[41] T. L. Carroll, "Optimizing memory in reservoir computers," *Chaos*, vol. 32, no. 2, 2022, Art. no. 023123.

[42] S. Jere, R. Safavinejad, and L. Liu, "Theoretical foundation and design guideline for reservoir computing-based MIMO-OFDM symbol detection," *IEEE Trans. Commun.*, vol. 71, no. 9, pp. 5169–5181, Sep. 2023.

[43] L. Gonon, L. Grigoryeva, and J.-P. Ortega, "Risk bounds for reservoir computing," *J. Mach. Learn. Res.*, vol. 21, no. 240, pp. 1–61, 2020.

[44] S. Jere, R. Safavinejad, L. Zheng, and L. Liu, "Channel equalization through reservoir computing: A theoretical perspective," *IEEE Wireless Commun. Lett.*, vol. 12, no. 5, pp. 774–778, May 2023.

[45] A. V. Oppenheim, A. S. Willsky, and S. H. Nawab, *Signals & Systems*, 2nd ed. Hoboken, NJ, USA: Prentice-Hall, Inc., 1996.

[46] K. Sozos, A. Bogris, P. Bienstman, G. Sarantoglou, S. Deligiannidis, and C. Mesaritakis, "High-speed photonic neuromorphic computing using recurrent optical spectrum slicing neural networks," *Commun. Eng.*, vol. 1, no. 1, p. 24, 2022, doi: 10.1038/s44172-022-00024-5.

[47] X. Liu, "A new method for the pole estimation of linear time-invariant systems using singular value decomposition," *J. Sound Vib.*, vol. 310, no. 4, pp. 998–1013, 2008.

[48] Y. Zhang, Z. Zhang, X. Xu, and H. Hua, "Modal parameter identification using response data only," *J. Sound Vib.*, vol. 282, no. 1, pp. 367–380, 2005.
[49] A. Edelman and N. R. Rao, "Random matrix theory," *Acta Numerica*, vol. 14, pp. 233–297, 2005.
[50] F. M. Bianchi, L. Livi, and C. Alippi, "Investigating echo-state networks dynamics by means of recurrence analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 2, pp. 427–439, Feb. 2018.
[51] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge, U.K.: Cambridge Univ. Press, 2019.
[52] H. Akaike, *Information Theory and an Extension of the Maximum Likelihood Principle*. New York, NY, USA: Springer, 1998, pp. 199–213.
[53] P. Stoica and Y. Selen, "Model-order selection: A review of information criterion rules," *IEEE Signal Process. Mag.*, vol. 21, no. 4, pp. 36–47, Jul. 2004.
[54] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, vol. 2. Berlin, Germany: Springer, 1999.
[55] B. S. Caffo, J. G. Booth, and A. C. Davison, "Empirical supremum rejection sampling," *Biometrika*, vol. 89, no. 4, pp. 745–754, 2002.

Shashank Jere (Graduate Student Member, IEEE) received the B.S. degree in electrical and electronic engineering from Nanyang Technological University, Singapore, in 2014, and the M.S. degree in electrical engineering from the University of California at Los Angeles, Los Angeles, CA, USA, in 2016. From 2016 to 2019, he was a Platform and Product Development Engineer with Qualcomm Technologies Inc., San Diego, CA, USA. He is currently working toward the Ph.D. degree with the Wireless@VT, Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA. His research interests include wireless communications, optimization, deep learning, information theory and statistical learning theory.

Lizhong Zheng (Fellow, IEEE) received the B.S. and M.S. degrees from Tsinghua University, Beijing, China, in 1994 and 1997, respectively, and the Ph.D. degree from the University of California, Berkeley, CA, USA, in 2002. Since 2002, he has been with the Department of Electrical Engineering and Computer Sciences, Massachusetts Institute of Technology, Cambridge, MA, USA, where he is currently a Professor of electrical engineering and computer sciences. His research interests include information theory, wireless communications, and statistical inference. He was the recipient of the Eli Jury Award from UC Berkeley in 2002, IEEE Information Theory Society Paper Award in 2003, NSF CAREER Award in 2004, and AFOSR Young Investigator Award in 2007. He became an IEEE Fellow in 2016.

Karim Said received the B.Sc. degree from Mansoura University, Mansoura, Egypt, in July 2006, and the M.S. and Ph.D. degrees from the Virginia Polytechnic institute and State University (Virginia Tech), Blacksburg, VA, USA, in 2012 and 2017, respectively. He is currently a Research Scientist with Virginia Tech, working on Waveform Design for 6G and Machine Learning for Wireless Communications

Lingjia Liu (Senior Member, IEEE) received the B.S. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, and the Ph.D. degree in electrical and computer engineering from Texas A&M University, College Station, TX, USA. He is currently a Professor and Bradley Senior Faculty Fellow with the ECE Department, Virginia Tech (VT), Blacksburg, VA, USA, and is also the Director of Wireless@Virginia Tech, a center focusing on wireless technology. He spent more than four years working with the Mitsubishi Electric Research Laboratory (MERL) and the Standards and Mobility Innovation Lab, Samsung Research America (SRA), where he was the recipient of Global Samsung Best Paper Award in 2008 and 2010. He was leading Samsung's efforts on multiuser MIMO, CoMP, and HetNets in 3GPP LTE/LTE-Advanced standards. His research interests include enabling technologies for 5G-Advanced/6G networks, including machine learning for wireless networks, massive MIMO, massive MTC communications, and mmWave communications. His research received eight Best Paper Awards. In 2021, he was the recipient of VT College of Engineering Dean's Award for Excellence in Research.