Brain-Inspired Wireless Communications: Where Reservoir Computing Meets MIMO-OFDM

Somayeh (Susanna) Mosleh, *Student Member, IEEE*, Lingjia Liu, *Senior Member, IEEE*, Cenk Sahin, *Student Member, IEEE*, Yahong Rosa Zheng, *Fellow, IEEE*, and Yang Yi, *Senior Member, IEEE*

***Abstract*— Reservoir computing (RC) is a class of neuromorphic computing approaches that deals particularly well with time-series prediction tasks. It significantly reduces the training complexity of recurrent neural networks and is also suitable for hardware implementation whereby device physics are utilized in performing data processing. In this paper, the RC concept is applied to detecting a transmitted symbol in multiple-input multiple-output orthogonal frequency division multiplexing (MIMO-OFDM) systems. Due to wireless propagation, the transmitted signal may undergo severe distortion before reaching the receiver. The nonlinear distortion introduced by the power amplifier at the transmitter may further complicate this process. Therefore, an efficient symbol detection strategy becomes critical. The conventional approach for symbol detection at the receiver requires accurate channel estimation of the underlying MIMO-OFDM system. However, in this paper, we introduce a novel symbol detection scheme where the estimation of the MIMO-OFDM channel becomes unnecessary. The introduced scheme utilizes an echo state network (ESN), which is a special class of RC. The ESN acts as a black box for system modeling purposes and can predict nonlinear dynamic systems in an efficient way. Simulation results for the uncoded bit error rate of nonlinear MIMO-OFDM systems show that the introduced scheme outperforms conventional symbol detection methods.**

***Index Terms*— Echo state network (ESN), multiple input multiple output (MIMO), nonlinear channel, orthogonal frequency division multiplexing (OFDM), power amplifier (PA), reservoir computing (RC), symbol detector.**

I. Introduction

ORTHOGONAL frequency division multiplexing (OFDM) is a promising multicarrier access techniques for wireless communication systems. OFDM converts a frequency-selective fading channel into a collection of parallel flat-fading subchannels. As a result, it provides robustness against narrowband interference, and lends to high spectral efficiency, enhanced channel capacity, and simplified transceiver structure [2]. Therefore, OFDM has been adopted in many modern telecommunication standards, such as DVB-T, 3GPP LTE/LTE-Advanced, and xDSL technologies.

On the other hand, OFDM also experiences some drawbacks. Most notable issues are the high peak-to-average power ratio (PAPR) and the sensitivity to both frequency offset and phase noise. Due to the issue of PAPR, a linear power amplifier (PA) is needed at the OFDM transmitter. The linearity requirement forces the PA to operate well below its saturation point leading to low energy efficiency. This is clearly undesirable for mobile devices, which usually have limited battery. Driving the PA closer to its saturation point is appealing, since it would increase the energy efficiency and prolong the battery life of a mobile device. However, driving the PA above the linear region results in nonlinear distortion effects. The nonlinear distortion makes it difficult to conduct symbol detection at the receiver.

In wireless communication systems, the transmitted signal undergoes degradation during propagation through the wireless channel. The combination of the multiple-input multiple-output (MIMO) and OFDM, referred to as the MIMO-OFDM, has been studied extensively in the industry and academia, due to its capability to provide high-rate transmission and robustness against multipath fading and other channel impairments. Accurate channel estimation is usually needed at the receiver to successfully detect transmitted symbols. Therefore, a major challenge of MIMO-OFDM systems lies in obtaining accurate channel state information (CSI) [3].

In general, CSI can be obtained through two methods [4], [5]. One is through blind channel estimation, which explores the statistical information of the channel and certain properties of the transmitted signals [5]. The other is through training-based channel estimation, which uses training signals sent by the transmitter, known *a priori* at the receiver [3]. Although the former has the advantage of incurring no overhead loss, it is only applicable to slowly time-varying channels due to its need for a long data record. This is also the main reason why the training-based method is widely adopted in most modern telecommunication systems, including *IEEE 802.16m* and 3GPP LTE/LTE-Advanced. In this paper, we focus on the training-based method and introduce a novel way to utilize these training signals for symbol detection.

The least square (LS) approach offers a decent channel estimation performance with relatively low complexity. However, this method is sensitive to the transmission noise and the

Manuscript received June 21, 2016; revised June 21, 2017 and September 26, 2017; accepted October 16, 2017. Date of publication December 7, 2017; date of current version September 17, 2018. The work of L. Liu and Y. Yi was supported by NSF under Grant ECCS-1731928. This paper was presented at the International Joint Conference on Neural Networks (IJCNN 2016) [1]. *(Corresponding author: Lingjia Liu.)*

S. Mosleh is with the Electrical Engineering and Computer Science Department, University of Kansas, Lawrence, KS 66045 USA.

L. Liu and Y. Yi are with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061 USA (e-mail: ljliu@ieee.org).

C. Sahin is with the Air Force Research Laboratory, Riverside, OH 45433 USA.

Y. R. Zheng is with the Electrical and Computer Engineering Department, Missouri University of Science and Technology, Rolla, MO 65409 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2766162

interpolation method. Another well-known channel estimation method is the linear minimum mean square error (LMMSE) estimator [2]. LMMSE is the optimal linear estimator that minimizes the expected value of mean squared error (MSE). In general, the LMMSE estimator has superior performance compared with the LS estimator. However, the LMMSE estimator has additional computational complexity.

In recent years, artificial neural networks (ANNs) gain momentum in providing effective solutions for various dynamic systems. An ANN usually consists of two components: the processing elements (called neurons) and the connections between them (termed parallel synaptic weights). Neural networks have a learning capacity like that of the human brain. These structures can be trained by various algorithms designed to adjust synaptic weights for each iteration in order to obtain the desired output for a given input. ANNs can be successfully applied for modeling nonlinear phenomenon of channel estimation, as they address complex classification problems by having the ability to form arbitrarily shaped nonlinear decision boundary regions [6]. Furthermore, they are known to perform complex mapping between their input and output space. Hence, networks of different architectures have been successfully applied to channel estimation. Two different ANN structures, namely, recurrent neural network (RNN) and multilayer perceptron (MLP), have been trained and tested for estimating wireless channels.

RNNs represent a very powerful generic tool, integrating both large dynamical memory and highly adaptable computational capabilities. The architecture of RNN consists of layers of processing units called neurons, interconnected by synapses, each having a specific weighted value. Compared with the more commonly used MLP which is mainly feedforward neural networks architecture, RNNs are distinctive in two ways: 1) the RNN exhibits dynamic behavior, due to the presence of feedback connections, and can maintain an activation, sometimes without an external input and 2) upon perturbation by an external input, the RNN produces and stores nonlinear transformations of the input history in its internal state, resulting in dynamic memory.

Both MLPs and RNNs have been used for channel estimations. MLP-based receivers have been mainly introduced in [7]–[11], while RNN-based receivers have been discussed in [12]–[14]. In both methods, ANNs are used to conduct channel estimation before applying for symbol detection. Furthermore, the study shows that RNN has superior estimation performance despite the large training complexity [15], [16]. This is because RNNs are capable of exploiting the underlying correlation within the data [17]. However, training a fully connected RNN in many cases is very difficult or even impossible [18]. Training algorithms for RNN are inherently difficult and follow a second-order gradient-descent method called Hessian-free optimization, which penalizes big changes in RNN activations and is likely to drive the learning process away from passing through many bifurcations [18]. This entails dynamic updates of the synaptic weights between all layers of neurons, rendering practical implementation highly infeasible. Furthermore, bifurcations in the training data can lead to nonconvergence [19]. Even when convergence occurs, the training process is computationally slow and intensive leading to poor performance of the modeling task [20]. In other words, most popular training methods for RNNs, such as backpropagation through time, real-time recurrent learning, and extended Kalman filter, are gradient-descent methods that are computationally expensive and often have extremely slow convergence. Therefore, these methods are generally suitable for small networks [21]. Therefore, an alternative way of training RNNs needs to be devised that is computationally simple while providing faster convergence.

Due to the difficulty of training RNNs, reservoir computing (RC) has recently attracted much attention. It has been shown in [21] and [22] that RC systems can outperform traditional RNNs in many cases. RC is especially suited for temporal data processing tasks in which the RNN is considered as a reservoir that produces and stores nonlinear transformations of the external input stimuli, which are then read out through linear connections at the output. RC gives important insights into RNNs, procuring practical machine learning tools as well as enabling computation with nonconventional hardware [23]. Echo state network (ESN) [21], [24] is one of the most popular RC systems. The basic concept of ESN is that RNNs can be trained without adapting each set of synaptic weights. Therefore, ESNs can outperform Hessian-free trained RNNs. Furthermore, the computational complexity of ESNs can be drastically reduced, since only one set of synaptic weights needs to be determined by a linear regression method. In essence, due to the computational complexity and the slow convergence of the training algorithms for RNNs, an ESN can provide a better performance [1], [25]–[27].

In this paper, we introduce an ESN-based symbol detector, which can overcome the issues created by the nonlinear distortion from the PA. In conventional approaches, accurate CSI estimates are required for the receiver to detect the transmitted symbol. The introduced method utilizes the RC architecture for symbol detection avoiding channel estimation and mitigating the nonlinear distortion from the PA. Simulation results for the uncoded bit error rate (BER) of the single-input-single-output (SISO) and MIMO-OFDM systems show the effectiveness of our scheme and its subsequent performance improvement.

The remainder of this paper is organized as follows. In Section II, we describe the system model and introduce the main assumptions required for the analysis conducted in this paper. Section III presents the concept of RC and demonstrates the design of the introduced ESN symbol detection scheme for MIMO-OFDM systems. Numerical results are shown in Section IV. An overview of the results, conclusion of this paper, and future work are presented in Section V.

II. Problem Formulation

In this paper, we consider a discrete-time baseband equivalent OFDM system, as shown in Fig. 1, where the transmitter consists of inverse discrete Fourier transform (IDFT), cyclic prefix (CP), and PA blocks, and the receiver consists of low noise amplifier, CP removal, and DFT blocks. Let N be the number of subcarriers of the underlying MIMO-OFDM

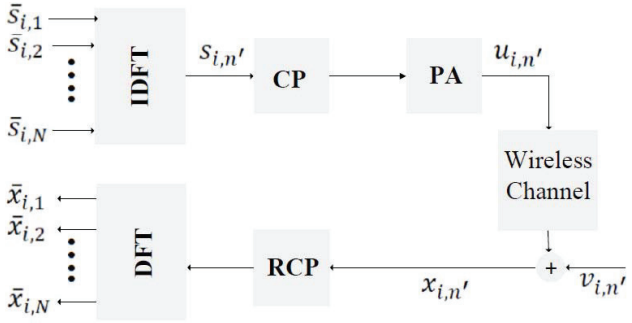


Fig. 1. Discrete-time equivalent baseband OFDM system.

system, $\bar{s}_{i,n}$ be the frequency-domain data symbol at the nth subcarrier and the ith symbol period, where $1 \leq n \leq N$ and $1 \leq i \leq I$ (I is the number of symbol periods), and $\bar{\mathbf{s}}(i) = [\bar{s}_{i,1}, \ldots, \bar{s}_{i,N}]^T \in \mathbb{C}^{N\times 1}$ be the vector containing the N data symbols of the ith symbol period. The frequency-domain data symbols $\bar{s}_{i,n}$ are assumed to be independently and identically distributed, with a uniform distribution over a quadrature amplitude modulation (QAM) or a phase shift keying alphabet. It is assumed that the transmitter does not have CSI. Thus, we consider that all the subcarriers have the same transmission power. Furthermore, we assume perfect symbol synchronization at the receive filter being matched to the transmit pulse shape filter.

The ith time-domain OFDM symbol is obtained by taking the IDFT of frequency-domain data symbols, that is

$$s_{i,n'} = \frac{1}{\sqrt{N}} \sum_{n=1}^{N} \exp\left[j2\pi(n-1)(n'-1)/N\right] \bar{s}_{i,n} \tag{1}$$

where $1 \leq n' \leq N$ and $j = \sqrt{-1}$. Note that (1) can be rewritten in the matrix form as $\mathbf{s}(i) = \mathbf{V}\bar{\mathbf{s}}(i)$, where $\mathbf{s}(i) = [s_{i,1}, \ldots, s_{i,N}]^T \in \mathbb{C}^{N\times 1}$ is the ith time domain symbol vector and $\mathbf{V} \in \mathbb{C}^{N\times N}$ is the IDFT matrix, with

$$[\mathbf{V}]_{p,q} = \frac{1}{\sqrt{N}} \exp[j2\pi\ (p-1)(q-1)/N], \quad 1 \leq p,\ q \leq N.$$

After the IDFT block, a CP of length M_{cp} is inserted in the symbols $s_{i,n}$ in order to ensure that the subcarriers are orthogonal, avoiding intersymbol interference (ISI) and intercarrier interference (ICI). However, this is accomplished only if the time dispersion from the channel is smaller than the duration of the CP. In fact, the CP is a copy of the last symbols $s_{i,N}$ at the beginning of the transmission block, inserted in the following way:

$$\mathbf{s}^{(cp)}(i) = [s_{i,(N-M_{cp}+1)}, \ldots, s_{i,N}\ \mathbf{s}^T(i)]^T \in \mathbb{C}^{(N+M_{cp})\times 1}.$$

The time-domain symbols with the CP are amplified by a PA, which can be represented using the Rapp model [28]. Note that when the PA is linear, the CP can be used to eliminate the ISI and ICI ensuring the orthogonality among subcarriers. However, as shown in sequel, when the PA is nonlinear, ICI will be introduced even when CP is used.

The nonlinear PA in the transmitter shown in Fig. 1 represents the nonlinear distortion imposed on the baseband signal. An accurate nonlinear PA model is the Rapp model [28]. Let $u_{i,n'}$, $1 \leq n' \leq N + M_{cp}$ be the time domain symbols after the PA, we have ($1 \leq n \leq N$)

$$u_{i,n+M_{cp}} = s_{i,n} G(|s_{i,n}|). \tag{2}$$

The PA gain $G(\cdot)$ can be expressed as

$$G(|s_{i,n}|) = \frac{A(|s_{i,n}|)\exp(j\phi(|s_{i,n}|))}{|s_{i,n}|} \tag{3}$$

where the term $|s_{i,n}|$ denotes the amplitude of $s_{i,n}$. Real amplifiers exhibit various magnitudes of nonlinearities. These are usually described by the amplitude transfer characteristics [also known as the amplitude modulation/amplitude modulation (AM/AM) conversion] and the phase transfer characteristics [also known as the amplitude modulation/phase modulation (AM/PM) conversion] of the amplifier. Note that since the PA model is memoryless, the PA output also has a CP. In (3), function $A(\cdot)$ represents the AM/AM conversion and function $\phi(\cdot)$ represents the AM/PM conversion. In the Rapp model, the phase distortion is assumed to be small enough so that it can usually be neglected. Therefore, the Rapp model can be characterized by the following AM/AM and AM/PM conversions [28]:

$$g(A) = \frac{\alpha A}{[1 + (\frac{\alpha A}{\beta})^{2r}]^{\frac{1}{2r}}}, \quad \phi(r) = 0 \tag{4}$$

where $A = |s_{i,n}|$ is the amplitude of the PA input signal, α is a small signal gain usually normalized to 1, β is the limiting output amplitude, and r controls the smoothness of the transition from linear operation to saturated operation. It is clear that $g(A) < \alpha A$ and $g(A) < \beta$. The amount of distortion introduced by the amplifier depends on the ratio ($\beta^2/\text{Var}[A]$), where $\text{Var}[A]$ is the average amplifier input power, and is characterized by the parameter ClipLevel[dB] defined by

$$\text{ClipLevel [dB]} = 10\log_{10}\frac{\beta^2}{\text{Var}[A]}. \tag{5}$$

Lower ClipLevel implies more severe nonlinear distortion. In addition, the severity of the nonlinear distortion depends on the modulation scheme. Higher order modulation schemes result in higher PAPR and more severe nonlinear distortion. We assume that PAs at all N_t transmit antennas are operating with the same r and β. According to the Bussgang theorem [29], the output of a nonlinear device can be divided into two parts: the useful degraded input replica and the uncorrelated nonlinear distortion. To be specific, we have

$$u_{i,n'} = s_u + n_d = \delta s_{i,n'} + n_d \tag{6}$$

where s_u and n_d are the useful part and distortion part of the output, respectively, and δ is a complex gain factor, which can be expressed as

$$\delta = \frac{\mathbb{E}\{u^*_{i,n'} s_{i,n'}\}}{\mathbb{E}\{s^*_{i,n'} s_{i,n'}\}}. \tag{7}$$

Here, $\mathbb{E}\{\cdot\}$ is the expected value and $(\cdot)^*$ represents the complex conjugate operation. The effect of the nonlinear amplifier depends on the operating point, which is the average power

of the input signals. Input back-off and output back-off [30] are two common parameters for verifying nonlinear distortion. They are defined as

$$\text{IBO} = 10\log\left(\frac{A_s^2}{P_{\text{in}}}\right), \quad \text{OBO} = 10\log\left(\frac{A_0^2}{P_{\text{out}}}\right) \tag{8}$$

where A_s is the PA input power at the saturation point, i.e., the input power corresponding to the maximum output power, P_{in} is the average PA input power, A_0 is the maximum output power, and P_{out} is the average output power. Note that due to the presence of PAs, a high PAPR introduces nonlinear ICI in the received signal if a high IBO is not used, which can significantly deteriorate the recovery of the information symbols. A high IBO results in a low energy efficiency of the PA and a low signal-to-noise ratio (SNR) at the receiver.

The PA output is transmitted through a frequency-selective wireless fading channel with impulse response denoted by h_m, $m = 0, \ldots, M$, where M is the delay spread of the underlying wireless channel. An additive white Gaussian noise (AWGN) of variance σ^2 is assumed at the channel output. At the receiver, the CP is removed from the time-domain received signal $x_{i,n'}$, $1 \le n' \le N + M_{cp}$. Thus, assuming that the length of the CP is greater than or equal to the channel delay spread, i.e., $M_{cp} \ge M$, the wireless channel can be represented by a circular convolution

$$x_{i,n'+M_{cp}} = \sum_{m=0}^{M} h_m u_{i,\text{Cir}(n'-m,N)} + v_{i,n'+M_{cp}} \tag{9}$$

where $1 \le n' \le N$ and $v_{i,n'+M_{cp}}$ is the AWGN component and the function $\text{Cir}(x, N)$ is defined as

$$\text{Cir}(x, N) \triangleq \begin{cases} x & 1 \le x \le N \\ x + N & 1 - N \le x \le 0. \end{cases} \tag{10}$$

Equation (9) can also be expressed in the matrix form as

$$\mathbf{x}(i) = \mathbf{H}\mathbf{u}(i) + \mathbf{v}(i) \tag{11}$$

where $\mathbf{x}(i) = [x_{i,M_{cp}+1}, \ldots, x_{i,M_{cp}+N}]^T \in \mathbb{C}^{N\times 1}$, $\mathbf{u}(i) = [u_{i,1}, \ldots, u_{i,N}]^T \in \mathbb{C}^{N\times 1}$, $\mathbf{v}(i) = [v_{i,M_{cp}+1}, \ldots, v_{i,M_{cp}+N}]^T \in \mathbb{C}^{N\times 1}$, and $\mathbf{H} \in \mathbb{C}^{N\times N}$ is the circulant matrix constructed from the channel impulse response h_m with

$$[\mathbf{H}]_{r,t} = \begin{cases} h_{r-t} & 0 \le r - t \le M \\ h_{r-t+N} & r - t \le M - N \\ 0 & 1 \le r, \quad t \le N. \end{cases} \tag{12}$$

The DFT of the received signals is calculated as

$$\bar{\mathbf{x}}(i) = \mathbf{V}^*\mathbf{x}(i) = \mathbf{V}^*\mathbf{H}\mathbf{V}\bar{\mathbf{u}}(i) + \bar{\mathbf{v}}(i) \tag{13}$$

where $\bar{\mathbf{x}}(i) \in \mathbb{C}^{N\times 1}$ is the vector of frequency-domain received signal at the ith symbol period, $\bar{\mathbf{u}}(i) = \mathbf{V}^*\mathbf{u}(i)$ is the frequency-domain version of $\mathbf{u}(i)$, and $\bar{\mathbf{v}}(i) = \mathbf{V}^*\mathbf{v}(i) \in \mathbb{C}^{N\times 1}$ is the frequency-domain noise vector, which is also AWGN with the same covariance matrix $\sigma^2\mathbf{I}_N$ as $\mathbf{v}(i)$.

III. Reservoir Computing and ESN Symbol Detection Method

Artificial RNNs analogous to the functioning of the human brain are often used to model nonlinear dynamic systems. An RNN is comprised of layers of abstract units called neurons, held together by synaptic connections, where the connection strength is indicated by a specified weight value. At least one feedback connection is present from the output to one of the layers. Training algorithms for RNNs are usually classified as gradient-descent methods, which reduce the specified training error parameter by iteratively converging to a local minimum. However, there are several drawbacks to gradient-descent based approaches [18]: 1) slow learning process; 2) difficulties in training large RNNs; 3) requirement of large memory; and 4) tendency to converge to local minimum.

Due to these problems, RC, a new paradigm of RNN training, becomes most popular recently. In RC approaches, a randomly generated RNN performs a nonlinear mapping upon perturbation by an external input. At the output of the RC, a linear readout is performed to estimate a target signal from the excited reservoir state. As a result, only the output weights are trained by a simple linear regression method. The reservoir exhibits short-term memory and the capability to preserve temporal data across distinct signals over time. Therefore, reservoir dynamics exhibit behavior similar to a spatiotemporal kernel whereby the input signal is projected to a high-dimensional space [31].

A variety of RC methods exist in the literature. Although they share the same basic concept, these methods have their own unique implementation: ESNs [21], [24], [32], [33], liquid state machines [17], backpropagation–decorrelation learning [34], temporal RNNs [35], and delay line feedback [36]–[38]. Among these methods, the ESN has been widely used in a variety of applications due to its implementation simplicity, scalability, and generalization capabilities.

A. Echo State Networks

ESNs are defined as an efficient and powerful computational model for approximating nonlinear dynamical systems and have been successfully applied in time-series prediction tasks. Indeed, to accurately predict the unseen values of the time series, the network would require a huge amount of memory. The ESN can utilize a massive short-term memory to develop an accurate dynamic model. Thus, a more accurate prediction of the time variation of the modeling task is obtained using ESNs. In principle, the ESN is an RNN with a nontrainable sparse recurrent part (reservoir) and a simple linear readout. A large RNN (of the order of hundreds of neurons) is used as a reservoir of dynamics, which can be excited by suitably presented input and output feedbacks. Connection weights in the ESN reservoir, as well as the input weights, are randomly generated and are not affected by the training. In order to compute the desired output dynamics, only the weights of connections from the reservoir to the output neurons are adjusted by the training.

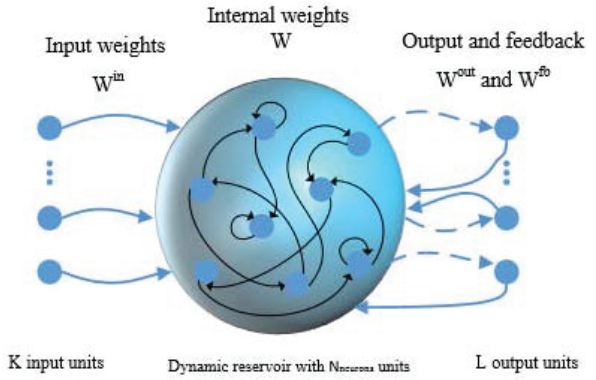


Fig. 2. Generic architecture of an ESN. Solid arrows show the fixed weights, while dashed arrows show the trainable weights.

B. Echo State Network Architecture

An ESN has three basic layers: the input layer, the dynamic reservoir, and the output layer. A generic architecture of the ESN is depicted in Fig. 2. The input layer is linked to the dynamic reservoir through the input weights $\mathbf{W}^{\text{in}}$. The dynamic reservoir has internal weights $\mathbf{W}$, which define the connections inside the reservoir. The dynamic reservoir is linked to the output layer through the output weights $\mathbf{W}^{\text{out}}$. The output is fed back to the dynamic reservoir through feedback weights $\mathbf{W}^{\text{fb}}$. Structurally, the main difference between an ESN and an RNN is the connectivity of neurons within the dynamic reservoir. The ESN is a sparsely connected RNN with $\mathbf{W}^{\text{in}}$, $\mathbf{W}$, and $\mathbf{W}^{\text{fb}}$ fixed *a priori* to randomly chosen values. In contrast with RNNs where the input and output weights are adjusted based on the minimization of the MSE, ESNs only calculate the output weights $\mathbf{W}^{\text{out}}$ leading from the dynamic reservoir to the output layer.

The basic idea behind ESN is to stimulate a random, large, and fixed RNN with an external input signal, which excites every neuron in the reservoir to generate nonlinear response signals, and to combine the desired output signals after training through a linear combination of the response signals. The main parameters of a typical ESN are defined in the following.

1) K input neurons and the discrete-time input sequence $\mathbf{u} = [u(1), \ldots, u(K)]^T \in \mathbb{C}^{K\times 1}$.
2) N_{neurons} neurons in the dynamic reservoir and the output of the dynamic reservoir which is the state vector $\mathbf{x} = [x(1), \ldots, x(N_{\text{neurons}})]^T \in \mathbb{C}^{N_{\text{neurons}}\times 1}$.
3) L output neurons and the discrete-time output sequence $\mathbf{y} = [y(1), \ldots, y(L)]^T \in \mathbb{C}^{L\times 1}$ which is corresponding to the input sequence $\mathbf{u}$.
4) Input weight matrix $\mathbf{W}^{\text{in}} \in \mathbb{C}^{N_{\text{neurons}}\times K}$.
5) Hidden layer weight matrix $\mathbf{W} \in \mathbb{C}^{N_{\text{neurons}}\times N_{\text{neurons}}}$.
6) output weight matrix $\mathbf{W}^{\text{in}} \in \mathbb{C}^{L\times\ (K+N_{\text{neurons}})}$.
7) Feedback weight matrix $\mathbf{W}^{\text{fb}} \in \mathbb{C}^{N_{\text{neurons}}\times L}$.
8) State activation function, f, for the reservoir which is a sigmoid function applied componentwise and is generally the tanh function.

For the ESN to be able to successfully perform the modeling task using the supervised learning algorithms, it must satisfy the echo state property. The key to understanding ESN training is the concept of echo states. Having echo states (or not having them) is a property of the network prior to training, that is, a property of the weight matrices $\mathbf{W}^{\text{in}}$, $\mathbf{W}$, and $\mathbf{W}^{\text{fb}}$. Intuitively, the echo state property implies, "if the network has been run for a very long time, the current network state is uniquely determined by the history of the input and the teacher forced output." The echo state property is a type of stability criteria, which states that the reservoir dynamics must be asymptotically state converging upon stimulation by an external input. In other words, the effect of the initial conditions must fade away with time. The echo state property is related to the algebraic properties of the weight matrix $\mathbf{W}$.

Remark 1: The existence of the echo state property may be verified in terms of separate necessary and sufficient conditions on the reservoir's weight matrix $\mathbf{W}$. The necessary and the sufficient conditions are that the spectral radius $\rho(\mathbf{W})$ and the largest singular value be less than one, respectively. In [21], the necessary condition is stated as a sufficient condition for the nonexistence of echo states when $\rho(\mathbf{W}) \geq 1$. The reason for this constraint is that if the underlying linear system is unstable, then the nonlinear system (resulting from the application of the input activation function) will also exhibit instability. From this point of view, the sufficient condition for the nonexistence of echo state property is $\rho(\mathbf{W}) \geq 1$. The necessary condition for the existence of echo state property results from this. From a system point of view, this requires that the nonlinear recurrent system can be locally asymptotically stable at the origin. For global asymptotic stability, a more restrictive sufficient condition is required. The proof of the sufficient condition is given by Jaeger in [21].

For practical purposes, the following procedure for determining weights seems to guarantee the echo state property (the target is to calculate the value of $\mathbf{W}^{\text{out}}$).

1) Generating the elements of $\mathbf{W}^{\text{in}}$ and $\mathbf{W}^{\text{fb}}$ from a uniform distribution over $[-1, 1]$.
2) Generating a sparsely random matrix $\mathbf{W}_0$ in the range $[-1, 1]$ and making sure that the mean value of all the weights are in near zero regime.
3) Scaling $\mathbf{W}_0$ by its highest eigenvalue $|\lambda_{\max}|$ and the spectral radius $\rho(\mathbf{W})$ with respect to $\mathbf{W} = (\rho(\mathbf{W})/|\lambda_{\max}|)\mathbf{W}_0$.

Even though the above procedure is used in many practical applications to guarantee the echo state property, it is not a sufficient condition [39]. To correctly define the echo state property, the statistical information of the input signal must also be taken into account [40]. It is worth noting that, the spectral radius $\rho(\mathbf{W})$ depends on the application and must be hand-tuned. In this paper, we have conditioned the spectral radius of the internal weight matrix to below unity, i.e., $\rho(\mathbf{W}) \in [0, 1]$, in order to ensure the echo state property.

C. ESN-Based Symbol Detection Scheme

Due to the nonlinear time-varying distortion of the wireless signal, we introduce an ESN as a black-box-time-domain symbol detector. To be specific, the wireless channel between the transmitter and the receiver is a multipath propagation environment that exhibits the properties of time variation and frequency selectivity. Transmitted signals undergo attenuation,

delay, and phase shift during propagation through the channel. Therefore, the wireless channel acts as a time-varying finite impulse response filter. In the conventional approach, successful detection of the transmitted signal often requires accurate CSI estimation and channel equalization. Unlike the traditional approach, we introduce a novel symbol detection scheme that does not require the explicit estimation of the CSI. The introduced scheme utilizes an ESN that acts as a black box for system modeling purposes. In this section, we describe the details of the ESN-based symbol detection. Furthermore, we will show that the behavior of the nonlinear time-variant system can be efficiently predicted and the consequent distortion can be reduced through our approach.

As mentioned earlier, a baseband OFDM system at the ith symbol period has a length-N complex-valued time-domain input sequence $\mathbf{s}(i) = [s_{i,1}, \ldots, s_{i,N}]^T$ and a length-N complex-valued time-domain output sequence $\mathbf{x}(i) = [x_{i,M_{cp}+1}, \ldots, x_{i,M_{cp}+N}]^T$, where N is the number of OFDM subcarriers. In addition, the channel impulse response has complex-valued coefficients, and it can be expressed as a complex vector $\mathbf{h} = [h_0, \ldots, h_M]^T$, where M is the delay spread of the underlying wireless channel. Assuming $M_{cp} \geq M$, the wireless channel can be represented by (9). Because the channel is complex, it is easy to see that the real (imaginary) part of output sequence depends not only on the real (imaginary) part of input sequence but also on its imaginary (real) part. For instance, let assume $M = 1$, then the channel impulse response has a single tap given by $h = \mathrm{Re}\{h_0\} + j\mathrm{Im}\{h_0\}$. Ignoring the AWGN, every output sample $x_{i,n'}$, $M_{cp} + 1 \leq n' \leq M_{cp} + N$, only depends on the corresponding input $s_{i,n}$, $1 \leq n \leq N$, and we have

$$\begin{aligned} \mathrm{Re}\{x_{i,n'}\} &= \mathrm{Re}\{s_{i,n}\}\mathrm{Re}\{h_0\} - \mathrm{Im}\{s_{i,n}\}\mathrm{Im}\{h_0\} \\ \mathrm{Im}\{x_{i,n'}\} &= \mathrm{Re}\{s_{i,n}\}\mathrm{Im}\{h_0\} + \mathrm{Im}\{s_{i,n}\}\mathrm{Re}\{h_0\} \end{aligned} \tag{14}$$

where $1 \leq i \leq I$ and I is the number of symbol periods.

It is clear that the input to the ESN is a complex-valued time-domain channel output, while the output is an estimate of a length-N complex-valued time-domain channel input sequence. However, the ESN can only operate on real numbers at its input and output nodes. Therefore, we introduce an ESN scheme with $2N_r$ input and $2N_t$ output nodes, in which the real and imaginary parts of the signals at both the input and output are fed to separate nodes. Therefore, in the SISO case, the number of ESN input and output nodes is equal to 2, while for the MIMO case, in which each transmitter and receiver equipped with two transmit and receive antennas, the number of ESN input and output nodes is equal to 4.

The block diagram of the introduced ESN-based symbol detector can be seen most clearly in Fig. 3. The inputs to the ESN are the real and imaginary parts of the length N complex-valued time-domain channel output sequence $y_{i,n}$, $1 \leq n \leq N + M_{CP}$, while the outputs are the real and imaginary parts of an estimate of $\mathbf{s}(i)$, denoted by $\hat{\mathbf{s}}(i)$. The outputs from the ESN symbol detector, i.e., $\mathrm{Re}\{\hat{\mathbf{s}}(i)\}$ and $\mathrm{Im}\{\hat{\mathbf{s}}(i)\}$, are combined to give $\hat{\mathbf{s}}(i) = \mathrm{Re}\{\hat{\mathbf{s}}(i)\} + j\mathrm{Im}\{\hat{\mathbf{s}}(i)\}$. Accordingly, the frequency domain detected transmitted symbol can be expressed as $\hat{\bar{\mathbf{s}}}(i) = \mathrm{DFT}\{\hat{\mathbf{s}}(i)\} = [\hat{\bar{s}}_{i,1} \ldots \hat{\bar{s}}_{i,N}]^T$.

An important consideration of the ESN-based symbol detector is the delay. This is the time it takes for the desired value to appear at the output of the ESN once the corresponding input is fed to the ESN. Let d_1 be the delay for the output node 1, and d_2 be the delay for the output node 2; both are nonnegative integers. The inputs to the ESN at time t are $x_{1,i,n'} = \mathrm{Re}\{x_{i,n'}\}$, and $x_{2,i,n'} = \mathrm{Im}\{x_{i,n'}\}$. Denote the values of the ESN output nodes by $z_{1,i,n}$ and $z_{2,i,n}$. The ESN's job is to compute estimates of the channel input sequence denoted by $\hat{s}_{i,n} = \mathrm{Re}\{\hat{s}_{i,n}\} + j\mathrm{Im}\{\hat{s}_{i,n}\}$, $n = 1, \ldots, N$. Having delays of d_1 and d_2 simply implies that

$$\begin{aligned} \mathrm{Re}\{\hat{s}_{i,n}\} &= z_{1,i,n+d_1} \\ \mathrm{Im}\{\hat{s}_{i,n}\} &= z_{1,i,n+d_2}. \end{aligned} \tag{15}$$

The delays d_1 and d_2 are incorporated to the training as follows. During the training, for a particular delay pair (d_1, d_2), the optimal output weights $\mathbf{w}_1$ and $\mathbf{w}_2$, both length-$N_n + 2$ vectors to the output nodes 1 and 2, respectively, are chosen, such that they minimize the total MSE as seen in (16), at the top of the next page, where the time series $\bar{s}_{i,n}$, $n = 1, \ldots, N$ is the teacher output. In other words, the training of the ESN is delay-specific. For each choice of the delay pair (d_1, d_2), we train the ESN and compute the training MSE given earlier. The ESN training MSE strongly depends on the delays for the particular realization of the channel impulse response (CIR). Since the CIR is not known in advance at the receiver, the optimal values of the delays (those that minimize the total MSE) are not known. Accordingly, during each training, the ESN is trained for different delay pairs, and the delay pair that minimizes the training MSE is chosen as the optimal delay

$$\left(d_1^*, d_2^*\right) = \arg \min_{(d_1,d_2)} \mathrm{MSE}^*(d_1, d_2). \tag{17}$$

It is worth noting that the closest pair of delays can be computed in $\mathcal{O}(n^2)$ time by performing a brute-force search, in which n is the number of delays in the lookup table.

Remark 2: In the block-fading wireless channel, the channel impulse response $\mathbf{h}$ is randomly generated from a wide-sense stationary uncorrelated scattering model and stays invariant for every K OFDM symbols [41]. Each element of $\mathbf{h}$ is a zero-mean complex Gaussian random variable with independent real and imaginary parts, and both parts have the same variance. The variance of each complex coefficient is set according to the power delay profile (PDP) of the channel $[\sigma_0^2, \ldots, \sigma_M^2]$, where $\sigma_\ell^2 = \mathbb{E}[h_\ell^* h_\ell]$. We use a one-sided truncated exponential function model for the PDP [41]. In particu-

$$\mathrm{MSE}^*(d_1, d_2) = \min_{\mathbf{w}_1, \mathbf{w}_2} \sum_{n=d_1}^{N+d_1-1} (z_{1,i,n}(\mathbf{w}_1) - \mathrm{Re}\{\bar{s}_{i,n-d_1}\})^2 + \sum_{n=d_2}^{N+d_2-1} (z_{2,i,n}(\mathbf{w}_2) - \mathrm{Im}\{\bar{s}_{i,n-d_2}\})^2 \tag{16}$$

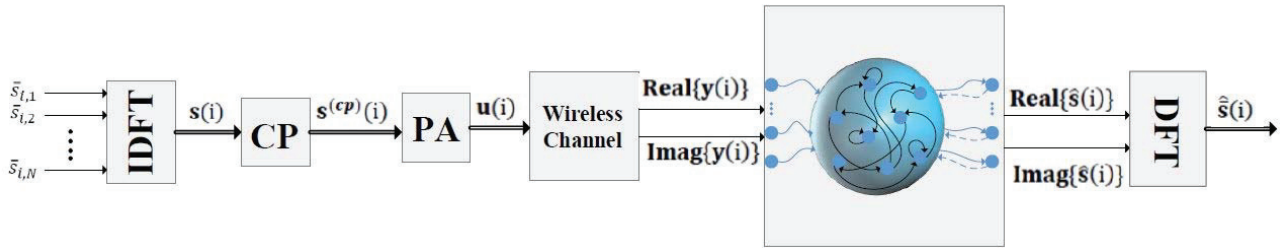


Fig. 3. Block diagram of the ESN-based symbol detector.

lar, σ_0^2 has the highest value, and the value of σ_ℓ^2 decays exponentially from $\ell = 0$ to $\ell = M$. On average, h_0 is the dominant coefficient having the largest magnitude. In that case, the optimal delay values are $d_1^* = d_2^* = 0$. Due to the randomness of the channel, it is possible that h_1 is the dominant CIR coefficient. In this case, the numerical results show that a delay value of at least 1 should be used to reduce the MSE; delays of 0 give poor MSE and BER results. This is intuitive, since in this case, the ESN estimates a sample before it has fully appeared at the ESN input. More details are discussed in Section IV. Additional parameters critical to the performance of the ESN estimator will be discussed in more detail in Section III-F.

D. Training of the Echo State Network

Given the ESN and the input–output sequences, the weights are trained to learn system characteristics. The available input–output sequences are divided into three parts:

1) an initial part, which serves the purpose of getting rid of initial transients in the network's internal states;
2) a training part, which is used in the actual learning procedure of adjusting the output weights;
3) a testing part, which is used to test the newly trained network on additional data.

Due to the complex structure of the dynamic reservoir, ESNs have a high capability for modeling complex dynamic systems. However, in order to limit the required computational effort during training, both online [42] and offline [33] training algorithms have been introduced. When using an offline mode of training, output weights are determined using teacher-forced outputs and inputs to obtain a collection of internal states in the state matrix. The calculated output weights can be used in the modeling task without any further modifications. In the online case, the output weights are updated in each time step, which assists in tracking time variance of the dynamic system. Note that in both the online and offline training algorithm of the ESN, only the output weights are updated, which guarantees the low computational requirement of the ESN approach. Both training modes are summarized in the following.

1) Offline Training Algorithm of the ESN:

Step 1: Generate an ESN following certain rules to ensure its echo state property. Note that once $\mathbf{W}^{\text{in}}$, $\mathbf{W}$, and $\mathbf{W}^{\text{fb}}$ are generated, they will not change during the entire training process.

Step 2: Feed the training data into the ESN to sample the network dynamics. When the training data are fed to the ESN, it activates the dynamic reservoir. First, the network's internal state vector $\mathbf{x}$ is initialized to zero, i.e., $\mathbf{x}(0) = \mathbf{0}$. Then, the network is driven by the training data for times $n = 0, \ldots, n_{\max}$ by presenting the input $\mathbf{u}(n)$ and forcing the output $\mathbf{y}(n-1)$ to the dynamic reservoir. The new state for the next time step is calculated as

$$\mathbf{x}(n+1) = f(\mathbf{W}^{\text{in}}\mathbf{u}(n+1) + \mathbf{W}\mathbf{x}(n) + \mathbf{W}^{\text{fb}}\mathbf{y}(n)) \tag{18}$$

where $\mathbf{u} = [x_{i,1}, \ldots, x_{i,N+M_{cp}}]^T$ and $\mathbf{y} = [\hat{s}_{i,1}, \ldots, \hat{s}_{i,N}]^T$, for $i = 1, \ldots, I$.

Step 3: Wash out the initial memory in the dynamic reservoir. The information from initial time steps, $n = 1, \ldots, n_0$, is not used for training, because the network's dynamics are partly determined by the initial arbitrary starting state of $\mathbf{x}(0) = \mathbf{0}$. By time n_0, the effects of the arbitrary starting state die out, and it is safe to assume that the network states are a pure reflection of the teacher-forced input and output. This is known as the initial washout time that is different according to different systems and the length of input sequence. For each time larger or equal to an initial washout time n_0, collect the input $\mathbf{u}(n)$ and the network state $\mathbf{x}(n)$ as a new row into a state-collecting matrix $\mathbf{S}$. In the end, the size of $\mathbf{S}$ works out to be $(n_{\max} - n_0 + 1) \times (N_{\text{neurons}} + K)$. Similarly, for each time larger than or equal to an initial washout time n_0, the sigmoid inverted teacher output $\tanh^{-1}\mathbf{y}(n)$ is collected in the teacher collecting matrix $\mathbf{T}$ of size $(n_{\max} - n_0 + 1) \times L$.

Step 4: Compute the output weights. Once the training is over, multiply the pseudoinverse of $\mathbf{S}$ with $\mathbf{T}$, to obtain a $(K + N_{\text{neurons}}) \times L$-sized matrix $\mathbf{W}^{\text{out}}$

$$(\mathbf{W}^{\text{out}})^T = \mathbf{S}^\dagger\mathbf{T} \tag{19}$$

where $\dagger$ denotes the pseudoinverse.

2) Online Training Algorithm of the ESN:

Step 1: Generate an ESH. In contrast to the offline training, there are four initial weight matrices that should be generated. Beside $\mathbf{W}^{\text{in}}$, $\mathbf{W}$, and $\mathbf{W}^{\text{fb}}$, the output weight $\mathbf{W}^{\text{out}}$ is also randomly generated. The rules for generating the weight matrices are the same as those described in the offline training.

Step 2: Calculate the states in the dynamic reservoir. This step is the same as that described in the offline training and (18) is used to calculate the states in the

dynamic reservoir. However, it is no longer required to collect the state at time n as a new row of a state matrix $\mathbf{S}$.

Step 3: Compute the estimated output of the ESN. The output of the ESN is calculated as

$$\hat{\mathbf{y}}(n+1) = f^{\text{out}}(\mathbf{W}^{\text{out}}(n)[\mathbf{u}(n+1); \mathbf{x}(n+1); \mathbf{y}(n)]) \tag{20}$$

where f^{out} is generally an identity or sigmoid function, $[;]$ indicates vector concatenation, and $\mathbf{W}^{\text{out}}(n)$ denotes the output weight matrix at time n. Input and output of the ESN are given by $\mathbf{u} = [x_{i,1}, \ldots, x_{i,N+M_{cp}}]^T$ and $\mathbf{y} = [\hat{s}_{i,1}, \ldots, \hat{s}_{i,N}]^T$, for $i = 1, \ldots, I$.

Step 4: Update the output weights. The estimated output is compared with the actual output in order to calculate the error vector $\mathbf{e}_y$. The output weights are updated, such that the mean square training error is minimized and can be updated as the following [42]:

$$\mathbf{W}^{\text{out}}(n+1) = \mathbf{W}^{\text{out}}(n)r + \eta \mathbf{x}(n+1)^T \mathbf{e}_y(n+1) + \gamma\, \mathbf{x}(n)^T \mathbf{e}_y(n) \tag{21}$$

where η is the learning gain and γ is the momentum gain, and each one is in the range of [0, 1].

Note that the value of the learning gain decides how fast the network learns. We could extract it, which would be the same as setting it to 1. In that case, the weights would change significantly whenever an error occurs. This tends to make the system unstable. On the other hand, the network will take longer time to learn under a small learning gain resulting in a slow convergence rate. However, the network will be more stable and resistant to noise (errors) and inaccuracies in the data. In general, we set $0.1 < \eta < 0.4$ depending upon how much error we expect in the inputs. Another effective approach increasing the convergence and stabilizing the training procedure is to add some momentum coefficient to the network. Adding a momentum term in the weight update helps avoid local minima, and also makes the dynamics of optimization more stable, since it is possible to use a smaller learning gain creating more stable learning and improved convergence rate. Typically, a value of $\alpha = 0.9$ is used.

It is noteworthy that adding a momentum term to the weight update is considered as a variation of the BP algorithm. The BP algorithm is widely used in ANN training. A momentum term is often added to the BP algorithm to accelerate and stabilize the learning procedure [43], in which the present weight updating increment is a combination of the present gradient of the error function and the previous weight updating increment. The choice of $\alpha = 0$ reduces this method to the original BP without a momentum term. The convergence of BP with a momentum term [see (21)] is considered by Bhaya and Kaszkurewicz [43] and Torii and Hagan [44]. They require the gradient of the error function to be a linear function of the weight. Especially, in [44], the learning rate and the momentum coefficient are restricted to constants. Consequently, the iteration procedure of BP with a momentum can be expressed as a stationary iteration. The convergence property is then determined by the eigenvalues of its iterative matrix. Due to the fact that the gradient of the error function is not a linear function of the weight for general activation functions, such as sigmoid functions, [45] generalized the result in [44] to a more general case and established the convergence of BP with a momentum term.

Remark 3: The goal of the training stage is to compute $\mathbf{W}^{\text{out}}$, such that an error measure, which is typically the MSE between the teacher signal and the trained output of the ESN, is minimized. A linear regression is usually performed on the states of the reservoir and the teacher output signals, and any suitable algorithm for linear regression can be implemented. It is worth noting that among the various linear regression algorithms, we used the matrix pseudoinverse for the computation of $\mathbf{W}^{\text{out}}$ due to its stability and feasibility.

Remark 4: Offline training is used for our implementation of the ESN-based symbol detector. It is optimal under MSE, and however, it cannot track time-varying channels like its online counterparts. The offline training can be a good fit for block-fading wireless channels where the channel remains invariant for a number of OFDM symbols. Furthermore, the offline training algorithm can be easily implemented in hardware.

E. Testing of the Echo State Network

The ESN ($\mathbf{W}^{\text{in}}$, $\mathbf{W}$, $\mathbf{W}^{\text{fb}}$, and $\mathbf{W}^{\text{out}}$) is now ready for use after the training. We can run the ESN on the test data, which has as initial condition on the last training time step (the neurons' states at time 0 in the testing phase are the neurons' states from time m in the training phase, where m is the size of the training sequence). The equations for the test phase can be expressed as

$$\hat{\mathbf{y}}(n) = f^{\text{out}}(\mathbf{W}^{\text{out}}[\mathbf{u}(n); \mathbf{x}(n); \mathbf{y}(n-1)]) \tag{22}$$

$$\mathbf{x}(n+1) = f(\mathbf{W}^{\text{in}}\mathbf{u}(n+1) + \mathbf{W}\mathbf{x}(n) + \mathbf{W}^{\text{fb}}\hat{\mathbf{y}}(n)) \tag{23}$$

where $\hat{\mathbf{y}}$ is the calculated output after the pseudoinverse calculation. To evaluate the performance, we usually use the normalized root mean squared error (NRMSE) which is defined as

$$\text{NRMSE} = \sqrt{\frac{\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2}{m \times \sigma_y^2}} \tag{24}$$

where σ_y^2 is the variance of the desired output signal $\mathbf{y}$, m is the testing sequence length, and $\hat{\mathbf{y}}$ is the output computed by the ESN after training.

Remark 5: In our ESN-based symbol detector, the reservoir activation function f in (18) is tanh and the output activation function f^{out} in (20) is an identity function. The tanh function is approximately linear in a small region around the origin and saturates at $+1/-1$ beyond this range. To keep the reservoir operation approximately linear, very small scaling of the input and output feedback weights are used for the ESN. It should be noted that although the sigmoid function shifts the training data near the origin to provide an approximate linear region of operation, the reservoir itself remains dynamic due to the existence of recurrent loops in the architecture.

F. Tuning the Echo State Network

As ESN is usually constructed by manually experimenting with a number of control parameters, in the following section we discuss in more detail on how to tune some of important ESN parameters in our ESN-based symbol detector.

1) Number of Neurons: One of the most important ESN parameters is the number of neurons in the reservoir. The relationship between N_{neurons} and the performance of the ESN-based symbol detector, i.e., BER, is critical. In general, it is desirable to have more neurons in the reservoir because this implies higher dimensionality. However, in the MIMO-OFDM system the training duration is usually fixed and is limited by the number of training signals of the system. As the number of neurons increases, the training per output weight decreases. The number of training samples is $2N$ and then the total number of output weights will be $2N_{\text{neurons}} + 4$. Therefore, the number of training samples per output weight is approximately N/N_{neurons} leading to a clear tradeoff between the reservoir dimensionality and the training quality. On the other hand, the channel is characterized by $2M$ real channel coefficients (M complex-valued with independent real and imaginary parts) that are generated independently. In concordance with the previous argument this implies that at a given SNR value the BER is a convex function of the number of neurons.

2) Spectral Radius: The spectral radius is a critical tuning parameter for the ESN. Usually the spectral radius is related to the input signal. However, if longer memory is needed, a higher spectral radius will be required. The downside of a longer spectral radius is longer time for the settling down of the network oscillations. Translating this into an experimental outcome means having a smaller region of optimality when searching for a good ESN with respect to some data sets. The spectral radius is considered as one of the most important tuning parameters of the ESN [33].

3) Weight Scaling: Input scaling is important for the ESN's ability to catch signal dynamics. If the input weights are too small, the network will be driven more by inner dynamics and thus lose the characteristics of the signal. If the input weights are too large, there will be no short-term memory and the inner states will be completely driven by the signal. Hence, the weight-scaling should be adjusted based on the input data.

4) Connectivity: Connectivity is another important parameter in the design of the ESN. It is defined as the number of nonzero weights from the total number of weights in the ESN. For example, for a 10 neuron network we will have 100 network weights; if we set the connectivity to 0.2 then the number of 0-valued weights will be $0.8 \times 100 = 80$. For our scenario, which considers a nonlinear case and uses a tanh activation function, the simulation results show that there is no effect on the connectivity value and the BER performance. In other words, fully connected networks perform as good as sparsely connected networks for this specific detection problem.

IV. Simulation Results

In this section, we present the simulation results for the ESN-based symbol detector. A step-by-step description of the introduced scheme is given as the following.

TABLE I
OFDM Parameters

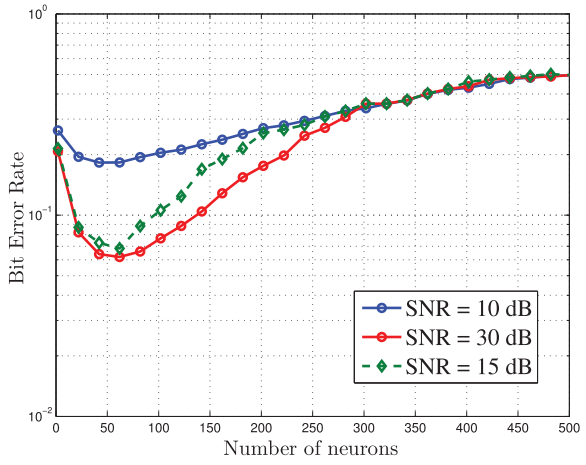
Number of OFDM Subcarriers	$N = 512$
Bandwidth	2 MHz
Doppler Frequency	100 Hz
Maximum Delay Spread	7μs
CIR duration	8

1) Both SISO-OFDM and MIMO-OFDM systems are simulated using a block-fading channel model whereby training signals are placed at the beginning of each frame. In this model, an independent channel realization occurs every K OFDM symbols. This specific value, K, is determined by physical parameters of the underlying wireless channel such as the Doppler frequency, system bandwidth, number of OFDM subcarriers, and channel impulse response duration. The system parameters used for performance evaluation are listed in Table I. Based on these parameters, the block-fading channel remains constant for $K = 19$ OFDM symbols. Separate linear and nonlinear block-fading channel models are developed for the current study. The linear block-fading channel is a multipath AWGN channel. The nonlinear PA model is incorporated into the channel model to form the nonlinear block-fading model. We assume that the parameters of the nonlinear PA are the same across all transmit antennas in the MIMO-OFDM system.
2) A CP of length $L - 1$ is added in the time-domain.
3) The channel output is $\mathbf{y}$ as shown in Fig. 3.
4) The ESN is generated randomly every 19 OFDM symbols. In the SISO-OFDM system, ESN has 2 input and 2 output nodes, while in an MIMO system with N_t transmit and N_r receive antennas, the ESN has $2N_r$ input and $2N_t$ output nodes so that in a pair of nodes one takes the real and the other takes the imaginary part of the signal. We consider an MIMO-OFDM system with 2 transmit and 2 receive antennas. The corresponding ESN has 4 input and output nodes. The training stage lasts for the duration of the first OFDM symbol and in the remaining symbols the ESN is tested for symbol detection. Each randomly generated ESN is trained with the optimal output delay values as discussed in Section III-C.
5) Withthe ESN-based symbol detector, we want to input $\mathbf{y}$, and discard all but the last N ESN output samples. The additional input samples are for the ESN transient. The CP samples of $\mathbf{y}$, the first $L - 1$ samples, have interference from the previous OFDM symbol. Our implementation takes this interference into account. The input to the ESN is the time-domain channel output $\mathbf{y}$, while the desired output from the ESN is an estimate of $\mathbf{s}(i)$, i.e., $\hat{\mathbf{s}}(i)$, where $\hat{\tilde{\mathbf{s}}}(i) = DFT(\hat{\mathbf{s}}(i))$ and $\hat{\tilde{\mathbf{s}}}(i)$ denotes the complex data symbols drawn from a signal constellation for transmission with an MIMO-ODFM system.

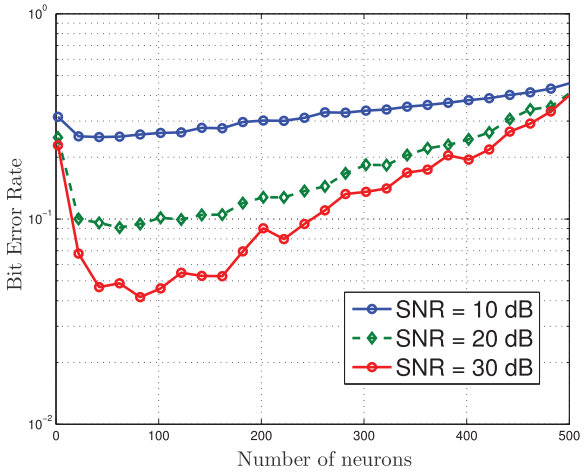
We choose quadrature phase shift keying and 16-QAM modulation schemes and set the parameters for the nonlinear PA, i.e., ClipLevel and Shape parameter, at 3 dB and 1, respectively. Uncoded BER is used as a performance measure to evaluate the effectiveness of our approach. The trained ESN is then used for transmission and BER curves as a function of the SNR are generated. For each SNR value, we simulate 4000 OFDM symbols whereby a new ESN is generated randomly every 19 OFDM symbols, and each randomly generated ESN is trained offline. To be specific, for each OFDM symbol we check whether the current OFDM symbol is for training or for data transmission. Two sets of simulations are performed. At first, the BER was simulated for different numbers of neurons in the reservoir to determine a suitable size of the reservoir. Using this value, we proceed to simulate the BER for different SNR values. The connectivity of the reservoir, which refers to the number of nonzero entries in the internal weight matrix $\mathbf{W}$, and the spectral radius are set at 20% and 0.98, respectively.

A. BER Versus Number of Neurons

In Figs. 4 and 5, we investigate the impact of the reservoir size on the BER of the SISO-OFDM and MIMO-OFDM system by increasing the number of reservoir neurons for 3 different SNR values: 10 dB, 20 dB, and 30 dB. Both the linear block-fading model and the nonlinear block-fading model where the nonlinear PA model is incorporated are simulated. The number of neurons in the reservoir is increased by increments of 20, starting from 2 neurons. It can be seen from the Fig. 4 that as the number of neurons increases, the BER of the ESN-based symbol detector first decreases and then increases. This is because as the the number of neurons increases a higher dimensionality becomes possible for resolving the inputs ï¼ˆsince more neurons are available for the per input sample for the same dimensions of the input). However, in the OFDM system the training signals are usually fixed. As the number of neurons increases, the training per output weight decreases. The number of training samples is $2N$ (N real and N imaginary). The total number of output weights is $2N_{\text{neurons}} + 4$. Thus, the number of training samples per output weight is approximately N/N_{neurons}. In short, Fig. 4 shows a clear tradeoff between the reservoir dimensionality and the training quality. As we increase the dimensionality, we decrease the training quality. On the other hand, the channel is characterized by $2L$ real channel coefficients (L complex-valued with independent real and imaginary parts). Therefore, we would expect the ESN-based symbol detector to do poorly when the number of neurons is less than $2L$. However, when the number of neurons is large, we also expect the ESN-based symbol detector to perform poorly due to fact that we do not have sufficient training signals. This implies that at a given SNR value the BER is a convex function of the number of neurons as shown in Fig. 4. In both cases (linear and nonlinear models), a convex shaped curve is obtained, with a minimum BER occurring at approximately the region from 50 to 100 neurons. With increasing SNR value, the magnitude of the minimum BER decreases as illustrated with both the linear and nonlinear



(a)



(b)

Fig. 4. BER versus number of reservoir neurons in SISO-OFDM. (a) Linear block-fading channel. (b) Nonlinear block-fading model.

block-fading models in Fig. 4. However, it is important to note that the value of the minimum BER all falls in the range from 50 to 100 neurons for the three SNR values.

For the MIMO-OFDM system, the BER shows a similar convex curve as given in Fig. 5, with the minimum BER occurring at approximately 50–100 neurons. The results are intuitive since as we have simulated an OFDM system with 512 subcarriers and used a reservoir with 20% connectivity, the number of nonsparse connections is around 500 when there are approximately 50 neurons in the reservoir. If the number of neurons in the reservoir is less than 50, there will be an inadequate number of reservoir connections compared to the number of subcarriers.

B. BER Versus SNR

From the analysis of the BER for different number of neurons, it was observed that the minimum BER occurred in the range from 50–100 neurons. Based on these observations, we selected a reservoir size to be 64 to perform the BER simulations for different SNR values. In this section, we simulated both SISO-OFDM and MIMO-OFDM systems. The OFDM

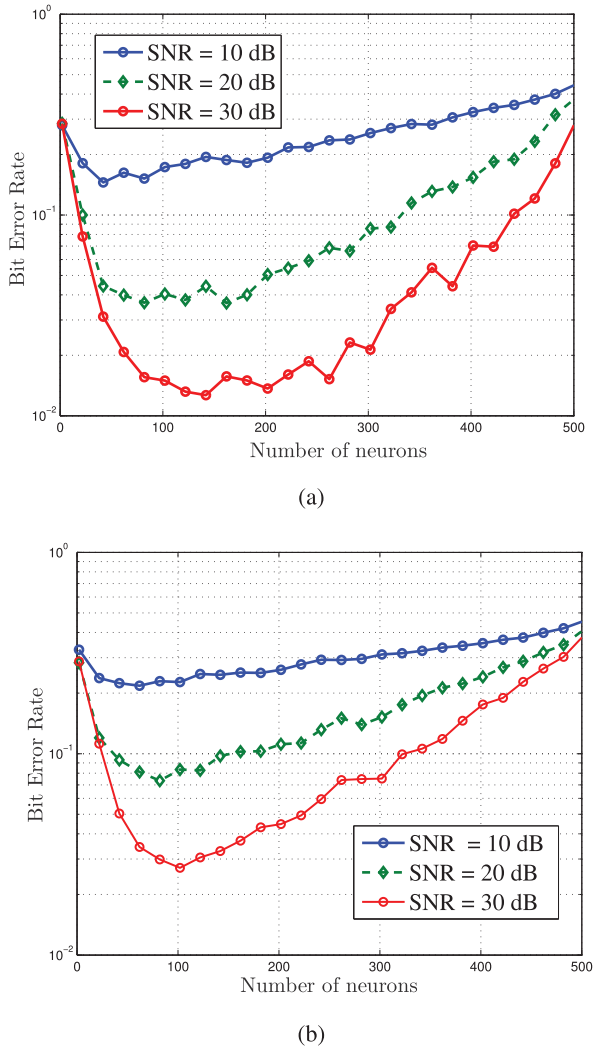


Fig. 5. BER versus number of reservoir neurons in MIMO-OFDM. (a) Linear block-fading channel. (b) Nonlinear block-fading model.

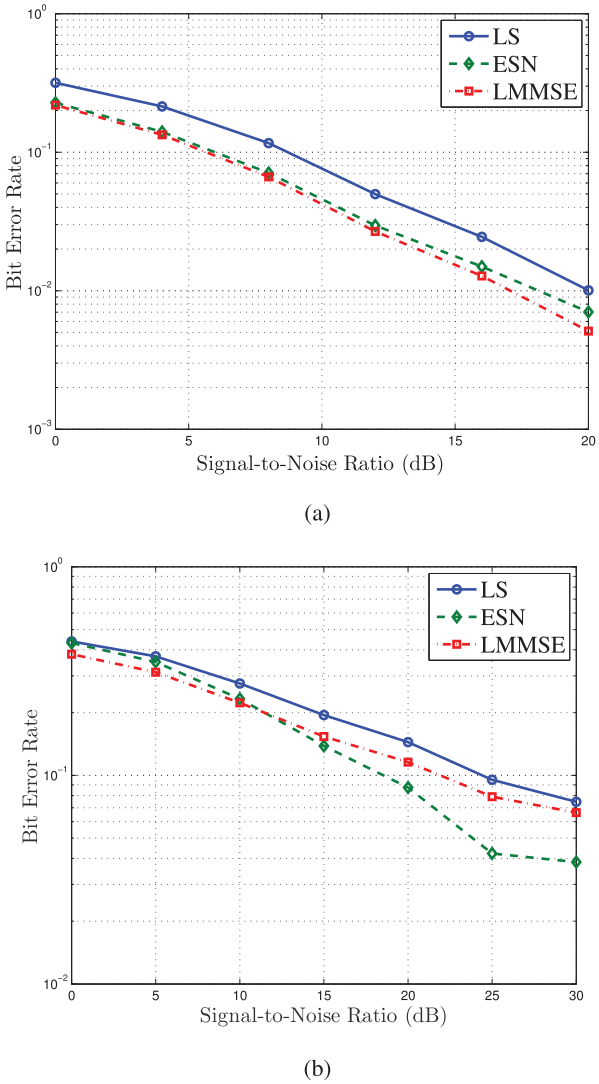


Fig. 6. BER versus SNR in SISO-OFDM. (a) Linear block-fading channel. (b) Nonlinear block-fading model.

system parameters are kept the same. For both SISO-OFDM and MIMO-OFDM systems, we perform simulations of the BER with increasing SNR for the ESN-based symbol detector along with the LSs channel estimator and the LMMSE channel estimator, for both the linear block-fading and the nonlinear block-fading models. Note that for LS and LMMSE cases, we first conduct channel estimation using LS and LMMSE based on training signals. After estimating the channel, conventional symbol detection is then conducted.

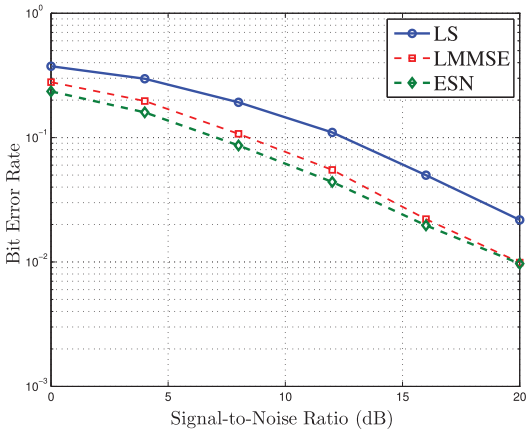
The BER for different SNR values is plotted in Fig. 6 for the SISO-OFDM system. For the linear block-fading channel, it is observed that the proposed scheme beats the LS channel estimator based symbol detection for all SNR values. Furthermore, it performs very close to the LMMSE channel estimator based symbol detection. It is important to note that both the ESN-based symbol detection and LS channel estimator based symbol detection does not require any statistical channel information. On the other hand, the LMMSE channel estimator based symbol detection requires channel statistical information.

For the nonlinear block-fading model in Fig. 6(b), the ESN-based approach outperforms both the LS estimator based approach, starting at approximately 5 dB and onward, and the LMMSE estimator based approach, starting at approximately 12 dB and beyond. With the increase of the SNR, the dominant source of signal distortion becomes the nonlinear PA. The PA introduces a larger magnitude of nonlinear distortion on the OFDM signal and this in turn gives rise to the high PAPR value. As the ESN can utilize a massive short-term memory, it can remember previous states up to order N_{neurons}, which indicates the reservoir size. This allows a more efficient resolution of the nonlinearity in the received signal; with feedback connections in the ESN, distortion from adjacent OFDM symbols can be cumulatively reduced.

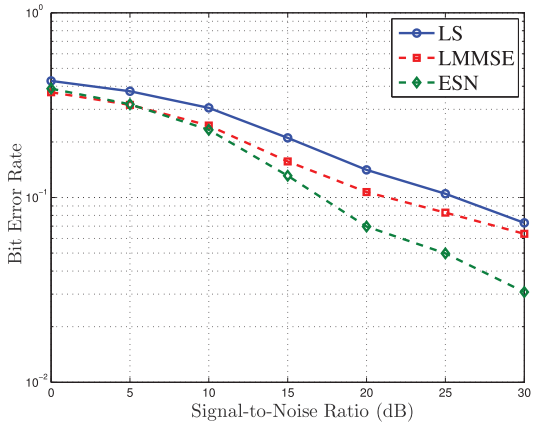
For the case of the MIMO-OFDM system in Fig. 7, we assumed that each transmit antenna has a nonlinear PA with the same parameters as given before. For every individual nonlinear PA present at each transmit antenna, the operation is assumed to be independent of other transmit antennas. Since the ESN now has 4 output nodes, we have 4 output delays. Every time we generate a new random ESN and train it

TABLE II
ESN PERFORMANCE IN TERMS OF TRAINING TIME, NMSE, AND PREDICTION ACCURACY

SNR	Training time	Testing runtime	Prediction Accuracy	NMSE Training	NMSE Testing
10 dB	0.7538	0.0540	77.92%	0.4300	0.5705
14 dB	0.7588	0.0543	84.77%	0.2857	0.3774
18 dB	0.7592	0.0544	90.24%	0.1869	0.2485
22 dB	0.7690	0.0550	93.94%	0.1321	0.1749
26 dB	0.7707	0.0550	95.56%	0.1082	0.1423
30 dB	0.7759	0.0552	96.91%	0.0872	0.1144



(a)



(b)

Fig. 7. BER versus SNR in MIMO-OFDM. (a) Linear block-fading channel. (b) Nonlinear block-fading model.

with different delay quadruples, one for each output node, and choose the one that minimizes the NMSE. The ESN BER curve, along with the LS BER curves and the LMMSE channel estimator BER curves, for both linear and nonlinear block-fading models are shown in Fig. 7. In the SISO-OFDM case, the ESN-based approach outperforms the LS and MMSE channel estimator based approach for all SNR values. The reason why the results of the linear block-fading channel for MIMO-OFDM systems are not consistent with that for SISO-OFDM systems is because in an MIMO system by increasing

TABLE III
COMPARISON BETWEEN ESN AND MLP

	SNR = 10 dB		SNR = 22 dB	
	ESN	MLP	ESN	MLP
Training time (s)	0.7991	3.0180	0.7690	10.2084
Testing run time (s)	0.0591	0.0153	0.0550	0.0124
MSE	0.5801	58.4504	0.1749	6.6466

the SNR, due to various impairments like fading, noise, scattering etc., signal corruption happens more frequently than the SISO case during wireless transmission. This introduces a larger magnitude of nonlinear distortion on the OFDM signal and in turn gives rise to the high PAPR value. Since the ESN can utilize a massive short-term memory, it can remember previous states up to the order of the number of neurons in the reservoir. This allows for a more efficient resolution of the nonlinearity in the received signal and with feedback connections in the ESN. In the nonlinear block-fading model, the ESN-based approach trails the performance of the LMMSE estimator based approach at around 10 dB. Beyond 10 dB, the ESN estimator outperforms both the LS and LMMSE estimators.

The training results of the ESN-based symbol detection in terms of training time/convergence time and accuracy, and ESN testing results in relation to prediction accuracy and the runtime are summarized in Table II. The BER is the percentage of bits that have errors relative to the total number of bits received in a transmission and is used to quantify a channel carrying data by counting the rate of errors in a data string. Accuracy, which shows the percentage of correctly detected symbols, which are detected correctly, can be defined as (1 - BER). To evaluate the ESN, we also calculate the NRMSE for the training and testing. The NRMSE is a frequently used measure of the difference between values predicted by a model and those actually observed in the environment that is being modeled. It is a good measure of how accurately the model predicts the response, and is the most important criterion if the main purpose of the model is prediction.

In the following, we compare the ESN performance with one of the most commonly used machine learning methods, namely MLP, for training and testing performance. Like ESN,

TABLE IV
THROUGHPUT COMPARISON BETWEEN ESN AND MLP

SNR	10 dB	14 dB	18 dB	22 dB	26 dB	30 dB
ESN	0.2208	0.1523	0.0976	0.0606	0.0444	0.0309
MLP	0.5002	0.5001	0.5000	0.4999	0.4997	0.5000

training of the MLP is done every 19 OFDM symbols. For each OFDM symbol, we check whether the current OFDM symbol is for training or for data transmission. Training is completed with a scaled conjugate gradient backpropagation algorithm and uses early stopping (the performance goal is set to 0.01). For each OFDM symbol, the input of the MLP is an $N \times 2N_r$ matrix. We used 200 validation examples. The number of hidden layers is 2 and the number of neurons in each layer is 64 and 128, respectively. We used sigmoid activation functions and the mean squared error cost function, because this combination avoids problems with upper/lower boundaries and local minima. The comparison is done for training time/convergence time, training and testing accuracy, and BER. The results are shown in Tables III and IV. Table IV demonstrates that even though the MLP does allow certain autonomy compared with the other machine learning techniques, its performance is usually strictly suboptimal when dealing with correlated data. Furthermore, Fig. 8 shows that even though MLP can be trained perfectly, the validation and test perform are poor due to the fact that the input data are highly correlated.

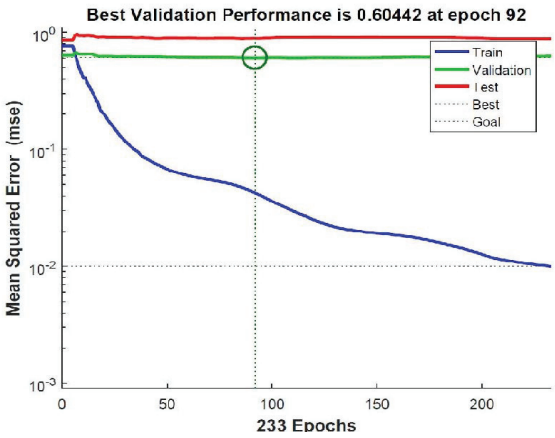


Fig. 8. Training and validation for MLP algorithm.

C. Complexity Analysis

In this section, we compare the computational complexity of our scheme with conventional ones. The computational complexity of the ESN-based symbol detector is associated with the updates of the weight matrix. Specifically, the proposed ESN-based symbol detector only needs to update the readout weight matrix, since training must occur only for the weights connecting the internal layer to the readout neuron. The computation complexity can be calculated by the total number of floating-point operations (FLOPs) [46]. Before explicating the complexity of each step, we first briefly summarize the number of FLOPs required for some basic operations: the product of two matrices of size $(m \times n)$ and $(n \times p)$ requires $m(n-1)p$ floating point addition and mnp floating point multiplication; the inversion of an $(n \times n)$ positive definite matrix takes n^3 floating point addition and n^3 floating point multiplication; and the complexity of the inner product operation between two $(n \times 1)$ vectors is $(2n-1)$. With these basic operations at hand, we now examine the complexity of the ESN-based, as well as the LS-based and MMSE-based methods.

The computational complexity of the ESN-based symbol detector is related to the readout matrix, which is given by (19). Hence, the implementation of $\mathbf{W}^{\text{out}}$ requires $(n_{\max} - n_0+1)^2 \times (N_{\text{neurons}}+K+n_{\max}-n_0)+(N_{\text{neurons}}+K)(n_{\max}-n_0+1)(n_{\max}-n_0)+L(N_{\text{neurons}}+K)(n_{\max}-n_0)$ floating point addition and $(n_{\max}-n_0+1)^2 \times (N_{\text{neurons}}+2+n_{\max}-n_0)(N_{\text{neurons}}+K)(n_{\max}-n_0+1)^2+L(N_{\text{neurons}}+K)(n_{\max}-n_0+1)$ floating point multiplication, where $n_{\max}$ and n_0 are the maximum time of training the network and the initial washout time, respectively. Therefore, the computational complexity of calculating $\mathbf{W}^{\text{out}}$ can be approximated by $(n_{\max}-n_0+1)^3$, which means that computational complexity for updating the weight matrix in the proposed scheme is related only to the training time. To compare the computation complexity of the ESN-based symbol detector, we must consider the computation complexity of the channel estimation algorithm along with the symbol detector scheme. Considering the number of complex additions and multiplications for each OFDM symbol as a complexity metric, the complexity of LS would be $(4N_r^3N_t + 2\ N_r^3 + N_r^2N_t - N_r^2 - N_rN_t)N$, while the complexity of LMMSE can be represented as $(6\ N_r^3 + 4\ N_r^3N_t + 3\ N_r^2N_t - 2\ N_rN_t)N$, where N_r is the number of receive antennas, N is the number of subcarriers, and N_t is the number of transmit antennas. Furthermore, at a receiver, a detector forms an estimate of the transmitted symbol by minimizing the distance between the received symbol and constellation points. The closest pair of points can be computed in $\mathcal{O}(n^2)$ time by performing a brute-force search, in which n is the number of constellation points.

V. CONCLUSION AND FUTURE WORK

In this paper, a novel ESN-based symbol detector is introduced for MIMO-OFDM systems. BER performance of the introduced symbol detector is compared with those of conventional symbol detectors based on channel estimation algorithms. Simulation results demonstrate the efficiency of our scheme in modeling channel behavior and compensating for nonlinear distortion.

As an extension of this paper, we will address other sources of nonlinearity, such as phase noise and Doppler shift. ESN with online training will also be an important future work.

REFERENCES

[1] S. Mosleht, C. Sahint, L. Liut, R. Y. Zheng, and Y. Yit, "An energy efficient decoding scheme for nonlinear MIMO-OFDM network using reservoir computing," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 1166–1173.

[2] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[3] L. Liu, R. Chen, S. Geirhofer, K. Sayana, Z. Shi, and Y. Zhou, "Downlink MIMO in LTE-Advanced: SU-MIMO vs. MU-MIMO," *IEEE Commun. Mag.*, vol. 50, no. 2, pp. 140–147, Feb. 2012.

[4] M. K. Ozdemir and H. Arslan, "Channel estimation for wireless OFDM systems," *IEEE Commun. Surveys Tuts.*, vol. 9, no. 2, pp. 18–48, 2nd Quart., 2007.

[5] L. Cheng, Y.-C. Wu, J. Zhang, and L. Liu, "Subspace identification for DOA estimation in massive/full-dimension MIMO systems: Bad data mitigation and automatic source enumeration," *IEEE Trans. Signal Process.*, vol. 63, no. 22, pp. 5897–5909, Nov. 2015.

[6] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 1999.

[7] G. Liodakis, D. Arvanitis, and I. O. Vardiambasis, "Neural network—Based digital receiver for radio communications ," *WSEAS Trans. Syst.*, vol. 3, no. 10, pp. 3308–3313, Nov. 2004.

[8] M. Jiang, C. Li, H. Li, and D. Yuan, "Channel tracking based on neural network and particle filter in MIMO-OFDM system," in *Proc. Int. Conf. Natural Comput. (ICNC)*, vol. 5. Oct. 2008, pp. 192–196.

[9] H. Cai and X.-H. Zhao, "MIMO-OFDM channel estimation based on neural network," in *Proc. 6th Int. Conf. Wireless Commun. Netw. Mobile Comput. (WiCOM)*, Sep. 2010, pp. 1–4.

[10] M. N. Seyman and N. Taşpınar, "Channel estimation based on neural network in space time block coded MIMO–OFDM system," *Digit. Signal Process.*, vol. 23, no. 1, pp. 275–280, Jan. 2013.

[11] C.-H. Cheng, Y.-P. Cheng, and W. Ching, "Using back propagation neural network for channel estimation and compensation in OFDM systems," in *Proc. 7th Int. Conf. Complex, Intell., Softw. Intensive Syst. (CISIS)*, Jul. 2013, pp. 340–345.

[12] C. Potter, G. K. Venayagamoorthy, and K. Kosbar, "RNN based MIMO channel prediction," *Signal Process.*, vol. 90, no. 2, pp. 440–450, Feb. 2010.

[13] G. Routray and P. Kanungo, "Rayleigh fading MIMO channel prediction using RNN with genetic algorithm," *Comput. Intell. Inf. Technol.*, vol. 250, pp. 21–29, Jan. 2011.

[14] K. K. Sarma and A. Mitra, "Modeling MIMO channels using a class of complex recurrent neural network architectures," *AEU-Int. J. Electron. Commun.*, vol. 66, no. 4, pp. 322–331, 2012.

[15] S. J. Nawaz, S. Mohsin, and A. A. Ikaram, "Neural network based MIMO-OFDM channel equalizer using comb-type pilot arrangement," in *Proc. Int. Conf. Future Comput. Commun.*, Apr. 2009, pp. 36–41.

[16] P. Gogoi and K. K. Sarma, "Channel estimation technique for STBC coded MIMO system with multiple ANN blocks," *Int. J. Comput. Appl.*, vol. 50, no. 13, pp. 10–14, Jul. 2012.

[17] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Comput.*, vol. 14, no. 11, pp. 2531–2560, 2002.

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

[19] K. Doya, "Bifurcations in the learning of recurrent neural networks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 6. May 1992, pp. 2777–2780.

[20] M. Lukoševičeus, "A practical guide to applying echo state networks," *Neural Netw., Tricks Trade*, vol. 7700, pp. 659–686, Aug. 2012.

[21] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks—With an Erratum note," GMD Nat. Res. Center Inf. Technol., Berlin, Germany, Tech. Rep. 148, Jan. 2010, pp. 1–47.

[22] X. Hinaut and P. F. Dominey, "On-line processing of grammatical structure using reservoir computing," in *Proc. 22nd Int. Conf. Artif. Neural Netw. Mach. Learn.*, Sep. 2012, pp. 596–603.

[23] M. Lukoševičius, H. Jaeger, and B. Schrauwen, "Reservoir computing trends," *KI-Künstliche Intell.*, vol. 26, no. 4, pp. 365–371, Nov. 2012.

[24] H. Jaeger, "Echo state network," *Scholarpedia*, vol. 2, no. 9, p. 2330, Sep. 2007.

[25] Y. Yi *et al.*, "FPGA based spike-time dependent encoder and reservoir design in neuromorphic computing processors," *Microprocess. Microsyst.*, vol. 46, pp. 175–183, Oct. 2016.

[26] C. Zhao, B. T. Wysocki, Y. Liu, C. D. Thiem, N. R. McDonald, and Y. Yi, "Spike-time-dependent encoding for neuromorphic processors," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 12, no. 3, pp. 23–46, Sep. 2015.

[27] A. M. Ehsan, H. An, Z. Zhou, and Y. Yi, "Design challenges and methodologies in 3D integration for neuromorphic computing systems," in *Proc. IEEE Int. Symp. Quality Electron. Design (ISQED)*, Mar. 2016, pp. 24–28.

[28] C. Rapp, "Effects of HPA-nonlinearity on a 4-DPSK/OFDM-signal for a digital sound broadcasting signal," in *Proc. 2nd Eur. Conf. Satellite Commun. (ECSC)*, Oct. 1991, pp. 179–184.

[29] A. Papoulis, *Probability - Random Variables and Stochastic Processes*, 2nd ed. Singapore: McGraw-Hill, 1985.

[30] P. Banelli and S. Cacopardi, "Theoretical analysis and performance of OFDM signals in nonlinear AWGN channels," *IEEE Trans. Commun.*, vol. 48, no. 3, pp. 430–441, Mar. 2000.

[31] A. Goudarzi, P. Banda, M. R. Lakin, C. Teuscher, and D. Stefanovic. (Jan. 2014). "A comparative study of reservoir computing for temporal signal processing." [Online]. Available: https://arxiv.org/abs/1401.2224

[32] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, Apr. 2004.

[33] H. Jaeger, "A tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the 'echo state network' approach," GMD German Nat. Res. Center Inform. Technol., Berlin, Germany, Tech. Rep. 159, Dec. 2013.

[34] J. J. Steil, "Backpropagation-decorrelation: Online recurrent learning with O(N) complexity," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2. Jul. 2004, pp. 843–848.

[35] P. Dominey, M. Arbib, and J. P. Joseph, "A model of corticostriatal plasticity for learning oculomotor associations and sequences," *J. Cognit. Neurosci.*, vol. 7, no. 3, pp. 116–311, 1995.

[36] A. Rodan and P. Tino, "Minimum complexity echo state network," *IEEE Trans. Neural Netw.*, vol. 22, no. 1, pp. 131–144, Jan. 2011.

[37] J. Li, L. Liu, C. Zhao, K. Hamedani, R. Atat, and Y. Yi, "Enabling sustainable cyber physical security systems through neuromorphic computing," *IEEE Trans. Sustain. Comput.*, to be published, doi: 10.1109/TSUSC.2017.2717807.

[38] J. Li, C. Zhao, K. Hamedani, and Y. Yi, "Analog hardware implementation of spike-based delayed feedback reservoir computing system," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 3439–3446.

[39] I. B. Yildiz, H. Jaeger, and S. J. Kiebel, "Re-visiting the echo state property," *Neural Netw.*, vol. 35, pp. 1–9, Nov. 2012.

[40] G. Manjunath and H. Jaeger, "Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks," *Neural Comput.*, vol. 25, no. 3, pp. 671–696, 2013.

[41] A. F. Molisch, *Wireless Communications*, 2nd ed. West Sussex, U.K.: Wiley, 2011.

[42] G. K. Venayagamoorthy, "Online design of an echo state network based wide area monitor for a multimachine power system," *Neural Netw.*, vol. 20, no. 3, pp. 404–413, Apr. 2007.

[43] A. Bhaya and E. Kaszkurewicz, "Steepest descent with momentum for quadratic functions is a version of the conjugate gradient method," *Neural Netw.*, vol. 17, no. 1, pp. 65–71, Jan. 2004.

[44] M. Torii and M. T. Hagan, "Stability of steepest descent with momentum for quadratic functions," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 752–756, May 2002.

[45] W. Wu, N. Zhang, Z. Li, L. Li, and Y. Liu, "Convergence of gradient method with momentum for back-propagation neural networks," *J. Comput. Math.*, vol. 26, no. 4, pp. 613–623, 2008.

[46] R. Hunger, "Floating point operations in matrix-vector calculus," Inst. Circuit Theory Signal Process., Tech. Univ. Munich, München, Germany, Tech. Rep., 2005.

Somayeh (Susanna) Mosleh (S'08) received the B.S. degree in electronics and the M.S. degree in communication systems from Yazd University, Yazd, Iran. She is currently pursuing the Ph.D. degree with the Department of Electrical Engineering and Computer Science, The University of Kansas, Lawrence, KS, USA.

From 2013 to 2014, she was with the Department of Electrical Engineering, University at Buffalo, Buffalo, NY, USA, as a Visiting Researcher. Her current research interests include reservoir computing, cooperative communications, caching, big data, and distributed designs.

Lingjia Liu (SM'15) 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.

From 2011 to 2017, he was an Associate Professor with the Electrical Engineering and Computer Science Department, The University of Kansas (KU), Lawrence, KS, USA. He spent over four years with the Mitsubishi Electric Research Laboratory and the Standards and Mobility Innovation Lab of Samsung Research America. He was leading Samsung's efforts on multiuser MIMO, CoMP, and HetNets in LTE/LTE-Advanced standards. He is currently an Associate Professor with the Electrical and Computer Engineering Department, Virginia Tech, Blacksburg, VA, USA. His current research interests include emerging technologies for 5G cellular networks, including machine learning for wireless networks, massive MIMO, massive MTC communications, and mmWave communications.

Dr. Liu received the Global Samsung Best Paper Award from Samsung Electronics, in 2008 and 2010, and the Air Force Summer Faculty Fellow from 2013 to 2017, the Miller Scholar at KU in 2014, the Miller Professional Development Award for Distinguished Research at KU in 2015, and the 2016 IEEE GLOBECOM Best Paper Award. He is currently an Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS, and an Associate Editor for the *EURASIP Journal on Wireless Communications and Networking* and the *International Journal of Communication Systems* (Wiley).

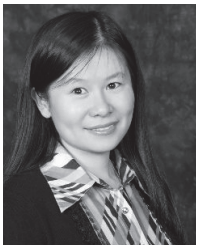
Cenk Sahin (S'06) received the B.S. degree (*summa cum laude*) in electrical and computer engineering from the University of Missouri–Kansas City, Kansas City, MO, USA, in 2008, and the M.S. and Ph.D. degrees (Hons.) in electrical engineering from The University of Kansas, Lawrence, KS, USA, in 2012 and 2015, respectively.

Since 2016, he has held a National Academies of Sciences National Research Council Research Associateship with the Sensors Directorate, Air Force Research Laboratory, Dayton, OH, USA. His current research interests include design and analysis of dual-function systems with radar and communication capabilities, signal processing methods for such systems, application of information theory and queueing theory to delay-sensitive wireless communications, modulation and coding techniques, spectrally efficient communications, and channel estimation for MIMO OFDM.

Yahong Rosa Zheng (F'15) received the B.S. degree in electrical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 1987, the M.S. degree in electrical engineering from Tsinghua University, Beijing, China, in 1989, and the Ph.D. degree from Carleton University, Ottawa, ON, Canada, in 2002.

From 2003 to 2005, she was an NSERC Post-Doctoral Fellow with the University of Missouri, Columbia, MO, USA. Since 2005, she has been a Faculty member of the Department of Electrical and Computer Engineering, Missouri University of Science and Technology, Rolla, MO, USA, where she is currently the Roy A. Wilkens Missouri Telecommunications Professor. She has published more than 60 journal papers and more than 100 conference papers in these areas. Her current research interests include digital signal processing, wireless communications, and wireless sensor networks.

Dr. Zheng was a recipient of an NSF CAREER Award in 2009. She has served as the technical program co-chair and the tutorial co-chair for many IEEE conferences. She served as an Associate Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS (2006–2008) and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY (2011–2016), and the IEEE JOURNAL OF OCEANIC ENGINEERING (since 2016). She has been a Distinguished Lecturer of the IEEE Vehicular Technology Society since 2015.



Yang Yi (SM'17) received the B.S. and M.S. degrees 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.

She is currently an Assistant Professor with the Bradley Department of Electronics and Communication Engineering, Virginia Tech, Blacksburg, VA, USA.

Her current research interests include very large scale integrated circuits and systems, computer aided design, and neuromorphic computing.

Dr. Yi was a recipient of 2016 Miller Professional Development Award for Distinguished Research, the 2016 United States Air Force Summer Faculty Fellowship, the 2015 NSF EPSCoR First Award, and the 2015 Miller Scholar. She is currently serving as an Associate Editor for the *Cyber Journal of Selected Areas in Microelectronics*. She has been serving on the Editorial Board of the *International Journal of Computational and Neural Engineering*. She has been selected as technical program committee members for various international conferences and symposiums.