IEEE ComSoc IEEE Communications Society | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS Connecting the Mobile World | IEEE Transactions on Machine Learning in Communications and Networking

Received 9 January 2024; revised 1 May 2024; accepted 5 June 2024.
Date of publication 13 June 2024; date of current version 28 June 2024.

The associate editor coordinating the review of this article and approving it for publication was X. Chu.

Digital Object Identifier 10.1109/TMLCN.2024.3414296

Reservoir Computing-Based Digital Self-Interference Cancellation for In-Band Full-Duplex Radios

ZHIKAI LIU, HAIFENG LUO, AND THARMALINGAM RATNARAJAH (Senior Member, IEEE)

Institute for Imaging, Data and Communications (IDCOM), The University of Edinburgh, EH9 3JW Edinburgh, U.K.

CORRESPONDING AUTHOR: Z. LIU (Z.Liu-129@sms.ed.ac.uk)

This work was supported by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/T021063/1.

ABSTRACT Digital self-interference cancellation (DSIC) has become a pivotal strategy for implementing in-band full-duplex (IBFD) radios to overcome the hurdles posed by residual self-interference that persist after propagation and analog domain cancellation. This work proposes a novel reservoir computing-based DSIC (RC-DSIC) technique and compares it with traditional polynomial-based (PL-DSIC) and various existing neural network-based (NN-DSIC) approaches. We begin by delineating the structure of the RC and exploring its capability to address the DSIC task, highlighting its potential advantages over current methodologies. Subsequently, we examine the computational complexity of these approaches and undertake extensive simulations to compare the proposed RC-DSIC approach against PL-DSIC and existing NN-DSIC schemes. Our results reveal that the RC-DSIC scheme attains 99.84% of the performance offered by PL-based DSIC algorithms while requiring only 1.51% of the computational demand. Compared to many existing NN-DSIC schemes, the RC-DSIC method achieves at least 99.73% of its performance with no more than 36.61% of the computational demand. This performance justifies the viability of RC-DSIC as an effective and efficient solution for DSIC in IBFD, striking it is a better implementation method in terms of computational simplicity.

INDEX TERMS Digital self-interference cancellation, in-band full-duplex, reservoir computing.

I. INTRODUCTION

A. MOTIVATIONS

IN-BAND full-duplex (IBFD) radio is one of the promising techniques in the next generation of wireless communications to double the spectral efficiency and reduce the end-to-end latency by simultaneously transmitting and receiving at the same frequency [1], [2], [3]. However, the receiver is exposed to the transmitter of the same node that is interfered with by its transmitted signal, i.e., self-interference (SI). The SI could be more than 100 dB higher than the received signals from other nodes due to the proximity of the transmitter to its receiver, swamping any desired signal of interest. Several self-interference cancellation (SIC) techniques have been proposed in the past few years, and the efficiency of IBFD with SIC has been demonstrated through simulations and hardware experiments in various applications [2], [4], [5], [6]. These SIC techniques are mainly available in three domains: i) propagation domain ii) analog domain and iii) digital domain. We refer to the SIC in the propagation and analog domain as analog self-interference cancellation (ASIC) as they both suppress the SI before the analog-to-digital converter (ADC) at the receiver. DSIC is initiated after the ADC has converted the analog signal to a digital format. This stage leverages sophisticated digital signal processing algorithms to minimize any residual self-interference further not addressed by the ASIC phase [7], [8], [9], [10]. Moreover, given that DSIC can be readily implemented and adapted in the digital domain, while modifications to ASIC often necessitate hardware changes, which are more costly, less flexible, and more challenging, the focus of this work is on advancing DSIC schemes.

The transmitter and receiver hardware impairment has been modeled by Gaussian noise, which leads to residual self-interference, and its effect has been minimized using different optimization techniques in the literature; for example [8], the author considers an IBFD multi-user non-orthogonal multiple access communication system and optimizes the received signal-to-interference-plus-noise ratio (SINR) per unit power. In [9], the feasibility of spectrum sharing between a multiple-input multiple-output (MIMO) radar system and an IBFD MIMO cellular system is considered, where a joint transceiver design technique at the cellular system and users is proposed to maximize the probability of detection of the MIMO radar system, subject to constraints of quality of service of users and transmit power at the cellular system, null-space based waveform projection is used to mitigate the interference from radar system toward cellular system. The weighted sum rate maximization in IBFD multi-user multi-cell MIMO networks is considered at [7]. In [10], the author considers the IBFD MIMO interference channel, and two transmit beamforming design problems are solved: 1) sum-power minimization problem subject to rate constraints and 2) energy-efficiency maximization problem subject to individual power constraints.

In [11] and [12], the transmitter and receiver hardware non-linearity is modeled, and its effect has been mitigated by machine learning techniques. Specifically, [11] utilizes a robust channel NN-DSIC, where the neural network estimates the linear self-interference channel coefficients. While machine learning-based DSIC strategies have been rigorously studied, some challenges still need to be addressed. One of the most significant obstacles facing current DSIC algorithms is the high computational complexity and extended running time [12], which must be carefully managed to optimize performance. Therefore, the search for more efficient DSIC schemes continues to be a focal point for researchers in the field.

Reservoir computing has emerged as a compelling approach for enhancing DSIC in wireless networks. Existing DSIC algorithms usually require high accuracy in their channel estimation and are computationally intensive. However, RC's unique architecture enables it to model complex systems quickly and accurately, and it is thus suitable for estimating the SI channel. Originating from recurrent neural networks (RNNs) [13], RC is distinguished by its unique ‘‘reservoir’’ architecture [14], where neurons are sparsely and randomly interconnected [15]. This reservoir functions as a dynamic temporal memory, adeptly capturing the latent temporal features of incoming signals and mapping them into a high-dimensional space [16]. In this transformed space, linear readout mechanisms can effectively tackle complex nonlinear problems [17]. The advantages of incorporating RC into DSIC are twofold. Firstly, RC's adaptive nature and ‘one-shot learning’ capabilities enable rapid acclimation to fluctuating network conditions, a frequent occurrence in wireless communication systems [18]. This contrasts traditional DSIC algorithms, which often necessitate complex optimization schemes that may not be as agile in adapting to environmental changes. Secondly, RC demonstrates excellent computational efficiency by requiring fewer computational resources [19], enabling real-time processing and providing a scalable solution for DSIC. In summary, the integration of reservoir computing into DSIC schemes presents a promising advancement in the field of self-interference cancellation. By offering an amalgam of high performance, adaptability, and reduced computational complexity, RC has the potential to revolutionize current DSIC paradigms.

B. CONTRIBUTIONS

This study proposes an RC-DSIC scheme for IBFD wireless communications, balancing computational complexity and system performance. We introduced the concept and operation of RC, highlighting its suitability for DSIC. Then, we compared the complexity of our proposed RC-DSIC scheme against the traditional polynomial-based approach (PL), and many existing NN-based DSIC schemes, including the robust channel stable NN-aided method (SNN) [11], traditional NN-aided scheme (TNN) [20], hybrid-convolutional recurrent NN (HCRNN) scheme [21], hybrid-convolutional recurrent dense NN (HCRDNN) scheme [21], ladder-wise grid structure (LWGS) scheme [22], and moving-window grid structure (MWGS) scheme [22]. We have validated our theoretical findings through optimization simulations of the RC-DSIC parameters. The comprehensive simulation comparisons with the previously mentioned schemes highlight the RC-DSIC strategy's capability to reduce computational demands while significantly maintaining effective SI cancellation performance. The details of contributions are given as follows:

1) PROPOSING THE RC-DSIC SCHEME

We introduced an innovative RC-DSIC scheme for IBFD wireless communication systems. Initially, we detailed the reservoir initialization process, introducing parameters such as the number of neurons, spectral radius, leak rate, and scaling factor. To ensure effective training, we carefully selected pilot symbols to represent the dataset adequately. We then described the reservoir state update and the training process. We also analyzed non-linearity, illustrating how RC's capability for nonlinear dimensionality expansion enables the modeling of complex dynamics, thereby enhancing the DSIC process.

2) COMPREHENSIVE COMPLEXITY ANALYSIS

We comprehensively analyzed the computational complexity associated with our proposed RC-DSIC scheme. By comparing it with the complexities of existing SNN, TNN, HCRNN, HCRDNN, LWGS, MWGS, and PL schemes presented in the Appendix, we delineated the inherent advantages of our proposed RC-DSIC methodology. This analysis offered valuable insights into how the RC-DSIC scheme balances the cancellation performance and computational demands, positioning it as a viable scheme for DSIC.

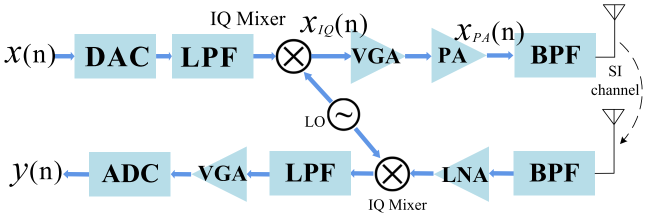

FIGURE 1. **A simplified model of an IBFD transceiver utilizing a shared local oscillator. A comprehensive version of this model is available in [4] and [20].**

3) THOROUGH SIMULATION VALIDATION

We conducted numerous experiments to evaluate the performance of the RC-DSIC scheme thoroughly. Our simulations were based on a dataset from an IBFD hardware testbed, equipped with National Instruments FlexRIO hardware, including two FlexRIO 5791R RF transceiver modules. Initially, our simulations optimized key parameters within the RC framework, laying the foundation for further analysis. We conduct subsequent simulations to compare the SI cancellation performance and computational complexity of the proposed RC-DSIC scheme against the existing DSIC methods. Furthermore, to investigate the SI cancellation performance of these DSIC schemes in low SNR (Signal-to-noise ratio) environments, we introduce additional noise into the dataset to evaluate their stability and robustness, highlighting our proposed method's practical value in real-world scenarios.

C. ORGANIZATION

The remainder of the paper is structured as follows. In Section II, we present the system model. In Section III, we introduced the proposed RC-DSIC scheme. In Section IV, the complexity analysis of the RC-DSIC scheme is presented. The simulation results are discussed in Section V. Finally, Section VI brings the paper to a conclusion. The complexity analysis of baseline algorithms is given in the Appendix.

D. NOTATIONS

Scalars, vectors, matrices, and sets are represented by the lower case (a), lower case boldface ($\mathbf{a}$), upper case boldface ($\boldsymbol{A}$), and calligraphic ($\mathcal{A}$) letters respectively. Transpose of matrices is denoted by $(.)^T$. Conjugate transpose is denoted by $(.)^H$. Conjugate is denoted by $(.)^*$. The notation $\|.\|_2$ denotes the l_2 norm. $|\mathcal{K}|$ denotes the cardinality of the set $\mathcal{K}$.

II. SYSTEM MODEL

This section describes a comprehensive system model for analyzing DSIC within an IBFD wireless communication system.

A. IN-BAND FULL DUPLEX MODEL

The IBFD transceiver integrates both a local transmitter and receiver while employing dual self-interference cancellation strategies, specifically analog and digital techniques [21]. In this system configuration, as depicted in Figure 1, the model is validated by experiments and widely adapted in DSIC studies [21], [22], [23]. Digital cancellation takes place before the analog-to-digital converter (ADC), aiming to eliminate any residual SI signal [21]. The digital signal $x(n)$ undergoes conversion to analog form via a digital-to-analog converter (DAC), then passes through a low-pass filter (LPF), and finally is modulated with the carrier frequency in an IQ mixer, where $n \in \{1, \ldots, L\}$ denotes each sample index and L represents the sample length [21]. The mixer imparts nonlinear distortion attributed to IQ imbalance, captured by the equation [23]:

$$x_{IQ}(n) = K_1 x(n) + K_2 x^*(n), \tag{1}$$

where $K_1, K_2 \in \mathbb{R}$, and $K_1 \gg K_2$. $x^*(n)$ presents the conjugate of $x(n)$. The signal, once modulated, is then subjected to amplification by the power amplifier (PA), which introduces additional nonlinear distortions. These distortions are represented through the parallel Hammerstein model, an established method for approximating such nonlinear effects. As per the conventional parallel-Hammerstein model [24], the PA's output signal, reflecting these nonlinearities, is formulated as:

$$x_{PA}(n) = \sum_{\substack{p=1 \\ p \text{ odd}}}^{P_{PA}} \sum_{m=0}^{M-1} h_p(m) \left| x_{IQ}(n-m) \right|^{p-1} x_{IQ}(n-m). \tag{2}$$

Here, $h_p(m)$ denotes the impulse response coefficients within the parallel Hammerstein model. The terms P_{PA} and M define the PA's nonlinearity order and memory length.

B. BASEBAND EQUIVALENT MODEL OF THE RECEIVED SI

At the receiver end of an IBFD node, the signal composition includes the SI signal, noise, and desired signals from remote nodes. For the sake of simplicity in this study, we assume the absence of desired signals from other IBFD nodes. Consequently, the remaining SI signal that persists following the RF cancellation stage undergoes a series of processing steps: it is filtered by a band-pass filter (BPF), amplified by a low noise amplifier (LNA), down-converted via an IQ mixer, and finally digitized by an ADC. Assuming that the ADC and potential baseband amplifiers are ideal, the down-converted and digitized received SI signal $y_{si}(n)$ can be modelled as

$$y_{si}(n) = \sum_{j=0}^{J-1} h_{si}(j) x_{\text{PA}}(n-j), \tag{3}$$

where J denotes the length of the SI channel, which is the number of previous samples of $x_{\text{PA}}(n)$ that affect the current SI signal received at the antenna, represented by $h_{si}(j)$. It describes how the signal that has passed through the PA is subsequently spread over time due to the SI channel's characteristics. By substitute (1), (2) in (3), the resulting SI signal $y_{si}(n)$ can be represented by the equation:

$$y_{si}(n) = \sum_{\substack{p=1\\ p \text{ odd}}}^{P_{PA}} \sum_{q=0}^{p} \sum_{m=0}^{M+J-1} h_{p,q}(m) x(n-m)^q x^*(n-m)^{p-q}, \tag{4}$$

where $h_{p,q}(m)$ denotes the impulse response of the channel, encompassing the cumulative effects of the PA, IQ mixer, and the SI channel [22]. The digital canceler in this system is tasked with two main objectives: first, to accurately estimate the effective channel coefficients $h_{p,q}(m)$, and second, to generate a precise replica of the SI signal denoted as $\tilde{y}_{si}(n)$ at the receiver. These objectives are achieved through a two-step process. In the first step, the digital canceler receives the transmitted baseband samples before digital-to-analog conversion and uses them for training purposes. During this training phase, it learns to create an accurate replica of the SI signal. In the second step, the trained digital canceler utilizes this replica to mitigate interference in the received post-ADC SI signal. The interference reduction is achieved by subtracting the replica from the post-ADC SI signal, resulting in the residual SI signal $y(n)$ represented by the equation:

$$y(n) = y_{si}(n) - \tilde{y}_{si}(n). \tag{5}$$

This separation of training and SI reconstruction processes ensures the effectiveness of the digital canceler in accurately estimating and cancelling the self-interference in the received signal.

C. KEY PERFORMANCE INDICATOR

The performance of the SI canceller is evaluated using the cancellation depth metric, quantified in decibels (dB). This metric measures the logarithmic ratio of the energy of the SI signal before suppression to the energy of the residual signal after suppression. The cancellation depth C_{dB} is mathematically defined as:

$$C_{dB} = 10 \log_{10} \left(\frac{\sum_{n=1}^{L} |y_{si}(n)|^2}{\sum_{n=1}^{L} |y(n)|^2} \right). \tag{6}$$

Here, C_{dB} provides a precise logarithmic measure of the power reduction achieved by the canceller. It quantifies the effectiveness of the SI suppression process by comparing the power levels before and after cancellation, giving a clear indication of the canceller's performance in reducing the SI signal.

III. RESERVOIR COMPUTING BASED DSIC SCHEME

In this section, we give a detailed explanation of how RC works and why it could be leveraged for DSIC. Reservoir computing employs a fixed, randomly initialized reservoir of neurons with sparse interconnections. This reservoir serves as a dynamic temporal memory, transforming incoming signals into a high-dimensional feature space where linear readout mechanisms can solve nonlinear problems. The effect of reservoir computing lies in its ability to perform one-shot learning, which enables quick adaptation to changing environments. Such characteristics make it a promising approach for digital processing.

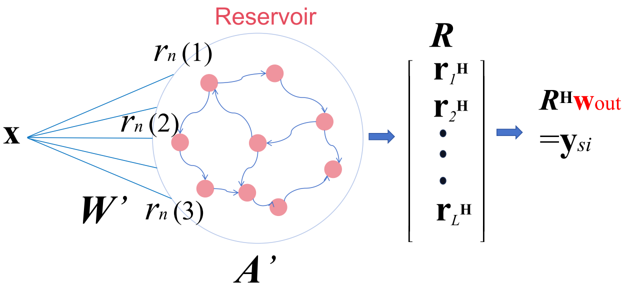
Reservoir
$r_n(1)$
$r_n(2)$
$r_n(3)$
x
W'
A'
R
$\mathbf{r}_1^{\mathrm{H}}$
$\mathbf{r}_2^{\mathrm{H}}$
$\mathbf{r}_L^{\mathrm{H}}$
$\boldsymbol{R}^{\mathrm{H}}\mathbf{W}_{\mathrm{out}}$
$=\mathbf{y}_{si}$

FIGURE 2. **Reservoir computing model.**

In an IBFD radio employing DSIC, the reservoir could be used to model the entire SI signal pathway from the transmitter input to the digital canceller. By feeding the known transmitted signal and the received signal (before DSIC) into the reservoir, the network can quickly learn the characteristics of the effective SI channel, including its nonlinearities and time variances. The high-dimensional feature space, transformed by the reservoir, could then be linearly read out to provide a highly accurate estimate for the SI channel coefficient vector $\hat{\mathbf{h}}_{p,q}$ that can derive the SI signal needed to be cancelled. These estimates can generate an accurate cancellation signal that, when subtracted from the received signal, ideally leaves only the noise, thus completing the DSIC process.

A. RESERVOIR INITIALIZATION

The first step of the RC-DSIC scheme is initializing the reservoir. The reservoir is architected as an echo state network (ESN), because of its computational efficiency in capturing space-temporal relationships. The initialization of parameters of the reservoir is significant, listed as follows:

- Number of neurons $N_{e,r}$: defining the reservoir capacity.
- Spectral radius ρ: determining the dynamic properties of the reservoir.
- Leak rate l_r: modulating the reservoir's temporal scale, with the rate itself being uniformly sampled from a range delineated by a minimum $\underline{l}_r$ and a maximum $\overline{l}_r$.
- Scaling factor σ: determining the influence of the input data on the reservoir's dynamics.
- Input weight matrix $\boldsymbol{W} \in \mathbb{R}^{N_{e,r} \times L}$: consisting of static random coefficients that determine the connections between the input and the reservoir nodes. It should be scaled by the scaling factor σ to modulate the magnitude of the input signals, expressed as:

$$\boldsymbol{W}' = \sigma \times \boldsymbol{W}. \tag{7}$$

- Reservoir matrix $\boldsymbol{A} \in \mathbb{R}^{N_{e,r} \times N_{e,r}}$: determining the connections of nodes within the reservoir, which are initialized randomly and remain fixed. The matrix elements are sparse, meaning only a small fraction of possible connections between nodes are non-zero. $\boldsymbol{A}$ should be normalized by its largest eigenvalue (its spectral radius $\rho_{\boldsymbol{A}}$) and then scaled by a desired spectral radius ρ. The scaling is performed as,

$$\boldsymbol{A}' = \frac{\rho}{\rho_{\boldsymbol{A}}}\boldsymbol{A}. \tag{8}$$

- Regularization factor λ: balancing the fit to the training data against the complexity of the readout weights.

The reservoir state is initialized to a $N_{e,r} \times 1$ zero vector $\mathbf{r}_0$.

B. PILOT SYMBOLS PREPARATION

Pilot symbols play a crucial role in wireless communication systems by facilitating channel estimation and synchronization. However, the inclusion of pilot symbols entails a trade-off with spectrum utilization, as dedicating resources to pilot transmission potentially reduces the bandwidth available for actual data transmission. Consequently, minimizing pilot overhead while ensuring sufficient data for accurate system training is a critical consideration in the design of DSIC schemes. This concern extends to the evaluation of pilot overhead's impact on performance, which is multifaceted and typically challenging to encapsulate within a closed-form expression. To address this, we explore the influence of different pilot overhead configurations through simulation.

In this study, the dataset $\mathcal{D}$ comprises sequences of transmitted and received signal pairs. To prepare pilot symbols for training the RC system, we partition $\mathcal{D}$ into two disjoint sets: the training pilot set $\mathcal{D}_{\text{train}}$ and the validation set $\mathcal{D}_{\text{val}}$. This partitioning is guided by the following relationship:

$$\delta = \frac{N_{\text{train}}}{L}, \tag{9}$$

where N_{train} represents the number of samples allocated to the training set $\mathcal{D}_{\text{train}}$, and δ denotes the predetermined proportion of the dataset designated as pilot symbols. The aim is to judiciously select pilot symbols in a manner that minimizes pilot overhead while ensuring robust training of the RC system.

We employ a systematic sampling strategy to ensure that the pilot symbols provide a representative snapshot of the entire dataset's dynamics and variations. The dataset $\mathcal{D}$ is chronologically indexed, and pilot symbols are extracted at regular intervals. This approach not only captures the essential characteristics of the dataset for effective system training but also addresses the critical concern of minimizing pilot overhead in DSIC schemes.

After segregating the pilot symbols, the remaining data, encapsulated within $\mathcal{D}_{\text{val}}$, is reserved for validation and testing purposes. Through these measures, we aim to balance the demand for pilot overhead against the necessity for efficient and effective interference cancellation, with the ultimate goal of optimizing DSIC scheme performance in real-world scenarios.

C. RESERVOIR STATE UPDATE

After the pilot symbols are prepared, the reservoir state update process commences. The number of time steps in RC is defined as L since each data point in the RC input sequence corresponds to a single time step, during which the reservoir integrates information from the current input with its existing state to capture temporal dynamics and dependencies. When dealing with input data of length L, where each data point is a discrete input to be processed, it is natural to define the number of time steps in RC as L. Each vector $\mathbf{r}_n$ of time slot n contributes to this matrix, creating a rich, high-dimensional representation of the input data as transformed by the reservoir's dynamic processes.

As depicted in Fig. 2, the input data $\mathbf{x} \in \mathbb{C}^{L\times 1}$ is processed by the scaled input weight matrix $\boldsymbol{W}'$ and then input to the reservoir. At each time step n, The update equation of the reservoir's state vector $\mathbf{r}_n \in \mathbb{C}^{N_{e,r}\times 1}$ is given by:

$$\mathbf{r}_{n+1} = (1 - l_r) \cdot \mathbf{r}_n + l_r \cdot f\left(\mathbf{A}'\mathbf{r}_n + \boldsymbol{W}'\mathbf{x}\right), \tag{10}$$

where f is the nonlinear activation function, chosen to be the *tanh* function. $\mathbf{r}_n$ represents the state vector of the reservoir at time step n. Each element $r_n(j)$ of the vector $\mathbf{r}_n$ represents the state of the j-th neuron at time step i. The term $(1 - l_r) \cdot \mathbf{r}_n$ accounts for the retention of the previous state's influence, modulated by the leak rate l_r, which allows for control over the memory capacity of the reservoir. The term $l_r \cdot f\left(\boldsymbol{A}'\mathbf{r}_n + \boldsymbol{W}'\mathbf{x}\right)$ accounts for the current state's update based on the input data and the internal dynamics dictated by the reservoir matrix $\boldsymbol{A}'$. The matrix $\boldsymbol{R} \in \mathbb{C}^{N_{e,r}\times L}$ comprising these evolving states $\mathbf{r}_n$, can capture the essence of the input data through the reservoir's dynamic response over time.

D. TRAIN THE READOUT MAP

The readout layer is a linear model trained using the regression technique. The goal is to find an optimal weight vector $\hat{\mathbf{w}}_{\text{out}} \in \mathbb{R}^{N_{e,r}\times 1}$ mapping the feature matrix $\boldsymbol{R}$ to the desired output $\mathbf{y}_{si}$, the self-interference signal to be canceled, expressed as,

$$\hat{\mathbf{w}}_{out} = \arg\min_{\mathbf{w}_{out}} \|\boldsymbol{R}^H \mathbf{w}_{out} - \mathbf{y}_{si}\|^2, \tag{11}$$

where $\mathbf{y}_{si} \in \mathbb{C}^{L\times 1}$ is the target output vector corresponding to the desired signal. $\boldsymbol{R}^H$ represents the conjugate transpose of matrix $\boldsymbol{R}$. To prevent overfitting, a regularization technique, ridge regression is used,

$$\hat{\mathbf{w}}_{out} = \arg\min_{\mathbf{w}_{out}} \left(\|\boldsymbol{R}^H \mathbf{w}_{out} - \mathbf{y}_{si}\|^2 + \lambda\|\mathbf{w}_{out}\|^2\right), \tag{12}$$

where λ is the regularization factor. The optimal weights $\mathbf{w}_{\text{out}}$ are solved for using closed-form solutions in the case of Ridge Regression for other types of regularization, expressed as,

$$\hat{\mathbf{w}}_{\text{out}} = (\boldsymbol{R}\boldsymbol{R}^H + \lambda\boldsymbol{I})^{-1}\boldsymbol{R}\mathbf{y}_{si}, \tag{13}$$

where $\boldsymbol{I}$ is the identity matrix. Once trained, the readout weights $\hat{\mathbf{w}}_{\text{out}}$ can be applied to the feature matrix $\boldsymbol{R}$ derived from new input signals to predict and cancel the self-interference signal in an IBFD communication system.

E. OPTIMIZATION AND VALIDATION

The optimization of the RC parameters is a critical step in enhancing the DSIC scheme's performance. Cross-validation, specifically K-fold cross-validation, is employed to fine-tune these parameters. This approach allows for a comprehensive evaluation of the model's performance across different subsets of the data, ensuring that the chosen parameters are not overly fitted to a specific portion of the dataset.

In K-fold cross-validation, the training pilot set $\mathcal{D}_{\text{train}}$ is divided into K distinct folds. Each fold serves as a validation set $\mathcal{D}_{\text{val}^{(k)}}$ in turn, while the remaining $K-1$ folds collectively form the complementary training set $\mathcal{D}_{\text{train}^{(-k)}}$. This process is represented as,

$$\mathcal{D}_{\text{train}} = \bigcup_{k=1}^{K} \mathcal{D}_{\text{train}^{(-k)}}, \mathcal{D}_{\text{train}} \cap \mathcal{D}_{\text{val}^{(k)}} = \emptyset, \forall k \in \{1, 2, \ldots, K\}. \tag{14}$$

For each fold k, the RC parameters—such as the number of neurons $N_{e,r}$, spectral radius ρ, leak rate l_r, input scaling σ, and regularization factor λ—are adjusted, and the model is trained on $\mathcal{D}_{\text{train}^{(-k)}}$. The performance of the trained model is then validated on $\mathcal{D}_{\text{val}^{(k)}}$. The optimization criteria are the cancellation depth defined in (6). The optimization simulation results will be presented in the subsequent sections.

F. NON-LINEARITY ANALYSIS

The aim of RC is to project the input into a higher-dimensional, nonlinear state space and then seek linear relationships within this space. The core of this RC's ability is embedded in its dynamic reservoir. The reservoir is a high-dimensional space represented by a network of interconnected neurons, each characterized by a unique state. The transformation from input $\mathbf{x}$ to the reservoir state $\boldsymbol{R}$ and subsequently to the output $\mathbf{y}_{si}$ is facilitated through a series of transformations that embed nonlinear characteristics into the output.

1) THE RESERVOIR AS A NONLINEAR PROJECTOR

As the reservoir state update equation given by Eq. (10), the reservoir is composed of a large number of interconnected neurons, creating a high-dimensional space. Each neuron's state is affected by its connections to other neurons and the input signal. This random, high-dimensional space is capable of representing complex patterns that are not discernible in lower-dimensional spaces.

2) NONLINEAR ACTIVATION FUNCTION

The activation function f, using *tanh* function, is pivotal in introducing nonlinearity into the system. This function can map the weighted sum of inputs and the current state into a nonlinear range, thus enabling the system to capture nonlinear patterns in the input data.

3) HIGH-DIMENSIONAL NONLINEAR MAPPING

The high-dimensional reservoir state $\boldsymbol{R}$ is mapped to the output $\mathbf{y}_{si}$ through a linear readout layer. The output is given by:

$$\mathbf{y}_{si} = \boldsymbol{R}^{H}\mathbf{w}_{\text{out}}. \tag{15}$$

The output weight vector $\mathbf{w}_{\text{out}}$ is trained to linearly combine the nonlinearly transformed states to produce the desired output. The reservoir transforms the input $\mathbf{x}$ into a high dimensional state matrix $\boldsymbol{R}$. This state vector represents a point in a high-dimensional space where linear separability of nonlinearly related inputs is achievable. The nonlinearity of the activation function ensures that the trajectory of the reservoir states corresponding to a time-varying input captures the inherent nonlinearities of the system.

In conclusion, the nonlinear mapping capability of RC is a result of the interplay between the high-dimensional reservoir, the nonlinear activation function, and the high-dimensional feature transformation. This intricate mechanism enables the one-dimensional linear input $\mathbf{x}$ to be mapped onto a P_{PA}-dimensional nonlinear output space $\mathbf{y}_{si}$, facilitating the modeling of complex dynamics and patterns.

IV. COMPLEXITY ANALYSIS

In this section, we analyze the complexity of the RC-DSIC scheme based on the number of floating-point operations (FLOPs) that consist of addition and multiplication operations. Each complex addition is counted as two real additions, and each complex multiplication is counted as three real multiplications and five real additions.

A. COMPLEXITY OF THE RC-DSIC SCHEME

In reservoir computing, the computational complexity is influenced by three key operations: the forward pass, weight calculation, and prediction. The ensuing discussion presents an in-depth analysis of each of these pivotal steps.

1) FORWARD PASS PROCESS

The forward process initializes and updates the reservoir state, a high-dimensional space where linear input combinations can be more easily separated. The concatenation of these states across all time steps will serve as the new feature matrix, which will be used for training the readout weights.

The forward pass is the phase where the input signal traverses through the reservoir to produce a new high-dimensional feature space. Essentially, this step transforms the input signal into the next reservoir state $\mathbf{r}_{n+1}$ by using the dynamics of the reservoir. The input includes the current input as well as $M+J$ past inputs, thus increasing the input vector size to $L \times (M+J)$. In the forward process as (10), the multiplication $\boldsymbol{W}'\mathbf{x}$ involves a matrix of size $N_{e,r} \times L \times (M+J)$. The multiplication $\boldsymbol{A}'\mathbf{r}_n$ involves a square matrix $\boldsymbol{A}'$ of size $N_{e,r} \times N_{e,r}$ and a vector $\mathbf{r}_n$ of size $N_{e,r}$. After the multiplications, the two results are summed and passed through an activation function tanh, which needs 6

FLOPs [21]. Therefore, the complexity of the forward pass process can be presented as

$$\begin{aligned}\mathcal{F}_{rc,f} &= 3N_{e,r}L(M+J)+2(N_{e,r}L(M+J)-1)\\ &\quad + 3N_{e,r}^2+2(N_{e,r}^2-1)+6N_{e,r},\\ &= 5N_{e,r}L(M+J)+5N_{e,r}^2+6N_{e,r}-4. \end{aligned} \tag{16}$$

In summary, this complexity formulation arises from the fundamental matrix-vector multiplications, summations, and the activation function application.

2) WEIGHT CALCULATION

The output weights are learned via a supervised learning algorithm using regularized least squares in the form of Ridge Regression. The goal is to find the optimal weight vector $\mathbf{w}_{out}$ that minimizes the loss function, defined as the mean squared error along with a regularization term. For the ridge regression solution, the matrix $\boldsymbol{R}$ is of size $N_{e,r} \times L$. So, $\boldsymbol{R}\boldsymbol{R}^H$ is a $N_{e,r} \times N_{e,r}$ matrix, where the complexity is $3N_{e,r}^2L$. The matrix $\mathbf{y}_{si}$ is of size $L \times 1$. The complexity for $\boldsymbol{R}\mathbf{y}_{si}$ is $3N_{e,r}L$. For the ridge regression solution, the term $(\boldsymbol{R}\boldsymbol{R}^H + \lambda I)^{-1}$ requires the inversion of a $N_{e,r} \times N_{e,r}$ matrix. The complexity for matrix inversion is $N_{e,r}^3$. Adding all these complexities, we get,

$$\mathcal{F}_{rc,w} = 3N_{e,r}^2L + 3N_{e,r}L + N_{e,r}^3. \tag{17}$$

In summary, the complexity of computing the optimal output weights involves multiplications and matrix inversion, each having its own specific computational costs that form the overall complexity.

3) PREDICTION STEP

The computational complexity of the prediction step hinges on the multiplication operations between the output weight vector and the reservoir state matrix. The prediction step is formulated as: $\mathbf{y}_{si} = \boldsymbol{R}^H\mathbf{w}_{\text{out}}$. The complexity of the multiplication operation is,

$$\mathcal{F}_{rc,p} = 3N_{e,r}^2\, L, \tag{18}$$

The cost is proportional to the number of time steps for which we are making predictions, as well as the number of neurons in the reservoir, capturing both the temporal and spatial aspects of the model's complexity. Combining all the steps, the overall time complexity for the model is,

$$\mathcal{F}_{rc} = 5N_{e,r}L(M+J)+6N_{e,r}^2L+N_{e,r}^3+5N_{e,r}^2+6N_{e,r}-4. \tag{19}$$

B. COMPLEXITY COMPARISON ANALYSIS

The computational complexities of the baseline DSIC algorithms are presented in the Appendix in detail. According to the comprehensive analysis, it is evident that the RC-DSIC scheme's capacity for achieving lower computational complexity relative to other DSIC approaches is rooted in its operational strategies and the specific architecture.

TABLE 1. **Parameter settings for simulations.**

Notations	Value
Memory of the PA M	3
SI channel length J	10
Sample length L	20480
Maximum nonlinearity order P_{PA}	5
Number of reservoir neurons $N_{e,r}$	20
Spectral radius ρ	1.5
Regularization factor λ	10^{-3}
Leak rate range midpoint $l_{r,m}$	0.2
Number of neurons of TNN scheme $N_{e,t}$	17
Number of training epochs of TNN scheme $N_{p,tnn}$	20
Number of neurons of SNN scheme $N_{e,s}$	17
Height, width, and depth of HCRNN filter C_h, C_w, C_d	12, 1, 1
Number of the HCRNN and HCRDNN filters F	3
Number of HCRNN recurrent layer neurons $N_{r,hcrnn}$	9
Number of training epochs of HCRNN scheme $N_{p,hcrnn}$	30
Number of HCRDNN recurrent layer neurons $N_{r,hcrdnn}$	5
Number of training epochs of HCRDNN scheme $N_{p,hcrdnn}$	30
Number of HCRDNN dense layer neurons N_d	12
Number of LWGS hidden layer neurons $N_{e,lwgs}$	10
Number of training epochs of LWGS scheme $N_{p,lwgs}$	30
Number of MWGS hidden layer neurons $N_{e,mwgs}$	12
Number of training epochs of MWGS scheme $N_{p,mwgs}$	30
Window size of MWGS scheme W_s	5

Reservoir computing relies on a fixed high-dimensional dynamic reservoir that processes inputs by mapping them into a higher dimensional space where linear separability of the data is more achievable. The fixed nature of the reservoir means that once set, the majority of the network does not require further adjustments, allowing the RC scheme to focus computational efforts on optimizing a smaller set of parameters. This process does not require backpropagation, the adaptation of the reservoir's weights during training, significantly reducing the computational burden. The only part of the system that is trained is a linear readout layer, which is much simpler to optimize compared to the entire network being trained in traditional and hybrid neural network models.

When comparing the RC-DSIC scheme to other NN-based schemes like TNN, SNN, HCRNN, HCRDNN, LWGS, and MWGS, it's evident that these models require more complex network structures (including multiple layers and specialized units like convolutional and recurrent layers). The RC-DSIC scheme, by contrast, involves sparse interconnections within the reservoir, significantly cutting down the number of required multiplications and additions during both the state update and readout phases. This sparse connectivity, along with the fixed nature of the reservoir weights, means that the computational overhead for updating the reservoir's state is substantially lower than that in fully connected layers of traditional or hybrid neural network models.

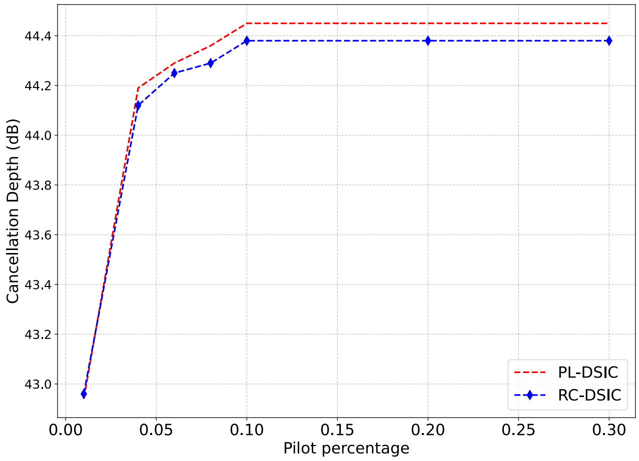

FIGURE 3. **Cancellation depth versus the pilot percentage.**

V. SIMULATION RESULTS

We conduct simulations using a real-world dataset from an IBFD hardware testbed, which utilized National Instruments FlexRIO equipment with two FlexRIO 5791R RF transceiver modules [25]. The dataset employed quadrature phase shift keying (QPSK) modulated orthogonal frequency-division multiplexing (OFDM) signals with a passband bandwidth of 10 MHz and 1024 carriers. The signals were sampled at a rate of 20 MHz, allowing for the observation of the signal sidebands. Each transmitted OFDM frame consisted of 20,480 baseband samples. The dataset was created with an average transmit power of 10 dBm, and a pass-band bandwidth of 10 Mhz, and the dual-antenna IBFD testbed configuration provided 53 dB of passive analog suppression. The source of noise in the dataset is attributed to oscillator phase noise [26]. To evaluate the performance of our proposed RC-DSIC scheme, we compare its self-interference cancellation performance with the traditional PL method, and various existing NN-based schemes, including the SNN [11], TNN [20], HCRNN [21], HCRDNN [21], LWGS [22] and MWGS [22]. All the simulation parameters are listed in Table 1. The parameters for the RC-DISC schemes were derived through optimizations, as detailed in the subsequent subsection. Other simulation parameters were carefully selected and standardized based on the benchmarks established in the referenced studies, ensuring they correspond to conditions that yielded optimal performance in those papers. This approach was adopted to facilitate a fair and meaningful comparison of our results with the findings of those studies. All results are averaged over multiple simulations.

A. PARAMETER OPTIMIZATION FOR THE RC-DSIC SCHEME

It should be noted that both RC and PL schemes critically depend on the accurate estimation of the self-interference channel and its non-linear characteristics to perform effective SI cancellation. This estimation process requires a specific amount of pilot symbols to be transmitted so that the schemes can learn or adapt to the channel properties accurately. More training pilots can enhance their performance at the cost of time-frequency resource efficiency. Figure 3 shows the achievable cancellation depth of RC-DSIC and PL-DSIC with increasing pilot percentages. As indicated in Figure 3, when the percentage of pilot symbols is set to 0.1, the performance of both the RC and PL schemes saturates, hence we select a training set proportion of 0.1 as pilot symbols for these methods. This strategy guarantees that the pilot symbols do not excessively consume transmission resources while enabling accurate self-interference channel estimation and non-linear characteristics.

However, NN-based schemes, characterized by their ability to learn from data, require a division of the available dataset into training and testing subsets. This allows the models to learn from a large portion of the data and validate their generalization capabilities on unseen data. This learning process inherently differs from the pilot symbol-based training in RC and PL schemes. For the simulations of NN-based DSIC schemes, 90% of the data is used for training, with the remaining 10% reserved for testing. This division ensures that the NN-based DSIC model is evaluated on unseen data, verifying its generalization capability and performance reliability. It should be noted that the NN-based DSIC schemes have 100% time-frequency resource efficiency even though we only use 10% data for testing.

The architecture of RC comprises a large number of interconnected neurons forming a dynamic reservoir. The behavior of this reservoir is highly nonlinear and complex, making it difficult to formulate an analytical solution for the optimal configuration mathematically. The interactions between neurons, the choice of activation functions, and the influence of various hyperparameters (such as the number of neurons, spectral radius, and leak rate) all contribute to the reservoir's dynamics. These factors introduce high-dimensional and nonlinear effects that are challenging to model mathematically in a closed-form manner. As a result, the optimal structure of an RC system is often found through empirical experimentation and iterative refinement. Researchers adjust the hyperparameters and reservoir architecture based on practical insights and domain knowledge. This empirical tuning process involves running numerous experiments with different configurations to identify settings that yield desirable performance for a specific task. The empirical tuning of RC parameters is a common practice in the field. Many studies on RC-based applications follow a similar methodology [18], [27], [28], emphasizing the importance of practical adjustments to achieve optimal performance.

To ascertain the optimized parameters for the RC-DSIC scheme, Figure 4 first illustrates the effect of the spectral radius ρ and the regularization factor λ on the cancellation depth. It is obvious that the cancellation depth increases as the spectral radius ρ ranges from 0.25 to 1.5 and diminishes when ρ extends from 1.5 to 2.5. The optimal cancellation depth is achieved when the spectral radius ρ equals 1.5. This

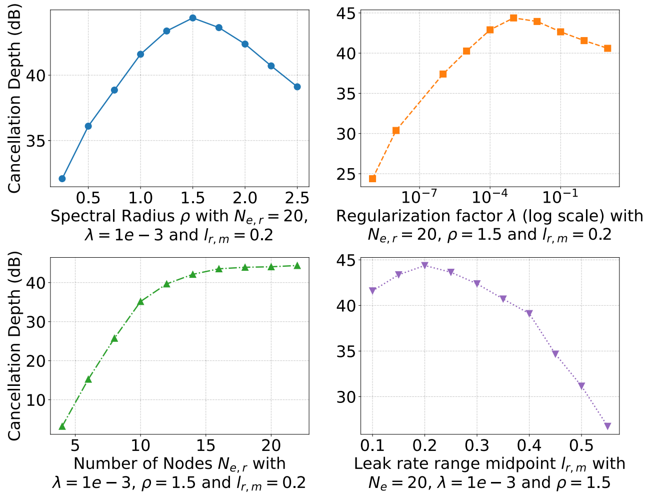

FIGURE 4. **Optimization for parameters of RC-DSIC scheme.**

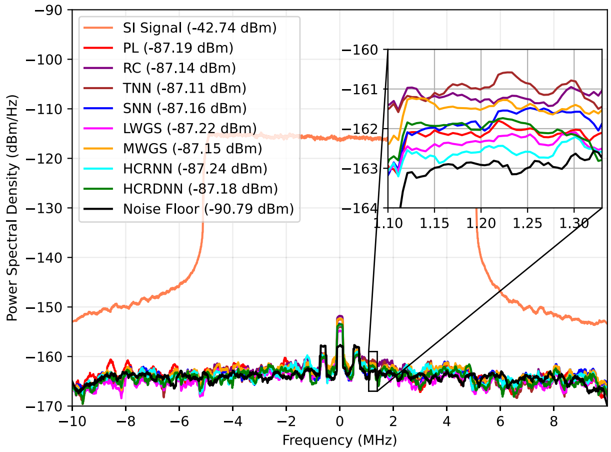

FIGURE 5. **PSD of the SI signal using the proposed and the existing DSIC schemes.**

phenomenon occurs because the spectral radius ρ affects the trade-off between the convergence speed and the stability of the reservoir. In the range of [0.25, 1.5], the growing cancellation depth reflects an enhancement in the reservoir's computational capacity, due to the ESN nearing the edge of criticality—characterized by rich dynamics and augmented memory capacity without sacrificing stability. When ρ surpasses 1.5, the network becomes unstable, with excessively correlated dynamics, undermining the input signal processing and the cancellation efficacy. The optimal spectral radius of 1.5 indicates the perfect balance between memory, stability, and processing capability within the reservoir. Concurrently, the cancellation depth is observed to increase when the regularization factor λ is increased from 10^{-9} to 10^{-3} and is reduced from 10^{-3} to 10. The peak cancellation depth is attained when λ is 10^{-3}. It indicates a regularization factor of 10^{-3} strikes a balance between underfitting and overfitting, allowing the reservoir computing model to maintain generality and achieve the highest cancellation depth through an optimal balance of model complexity and fitting precision.

This figure also illustrates the impact of the number of reservoir nodes ($N_{e,r}$) and the leak rate range midpoint ($l_{r,m}$) on the cancellation depth within a reservoir computing framework. The leak rate (l_r) is sampled from the range $[l_{r,m} - 0.025, l_{r,m} + 0.025]$. It is evident that the cancellation depth increases when $l_{r,m}$ increases from 0,1 to 0.2 and subsequently decreases when $l_{r,m}$ is increased from 0,2 to 0.5. The peak in cancellation depth at $l_{r,m} = 0.2$ denotes an optimal compromise, allowing the reservoir to adequately process historical information while retaining the flexibility to adjust to incoming data. Moreover, it is observed that as the number of reservoir nodes ($N_{e,r}$) increases, there is a corresponding increase in the cancellation depth, which converges when $N_{e,r}$ reaches 20. As $N_{e,r}$ expands, the system develops a more comprehensive representation space, enhancing its ability to discern and adapt to complex signal patterns and dependencies, ultimately leading to more effective cancellation. The stabilization of cancellation depth at $N_{e,r} = 20$ suggests that the system has attained a sufficient scale to handle the current informational load. Therefore, for the sake of computational efficiency, the number of reservoir nodes will be set to 20 in subsequent simulations. In summary, the RC-DSIC scheme's performance for the DSIC model is optimized at a spectral radius of $\rho = 1.5$, a regularization factor of $\lambda = 10^{-3}$, nodes number $N_{e,r} = 20$ and leak rate range midpoint $l_{r,m} = 0.2$. In subsequent simulations, these parameters will be held constant at these optimized levels to ensure maximal cancellation depth.

B. SI CANCELLATION PERFORMANCE COMPARISON

In Figure 5, we evaluate the power spectral density (PSD) of the SI signal after applying various cancellation techniques. The PSD of the SI signal prior to the application of any cancellation methods is depicted by the blue curve. Additionally, the black curve represents the PSD of the receiver's background noise when not transmitting, essentially the noise floor of the receiver. As evidenced by the figure, the RC-DSIC scheme can exhibit performance comparable to other DSIC schemes, being a mere 0.12 dB above the optimal DSIC performance.

The comparative analysis depicted in Figure 6 demonstrates the SI cancellation depth achieved by the proposed RC-DSIC scheme versus the traditional PL method and various NN-based DSIC schemes, with respect to the SI channel length J. As the SI channel length increases, the cancellation depth decreases due to several factors. Primarily, a longer SI channel length introduces more complex interference patterns, involving signals from multiple past transmissions that affect the current received signal. This complexity arises because the SI channel with a longer impulse response captures more echoes and reverberations, which represent a broader range of time delays and signal distortions. Consequently, the DSIC system must accurately model and cancel a more extensive array of interference effects, increasing the challenge of achieving deep cancellation. The extended channel memory demands more

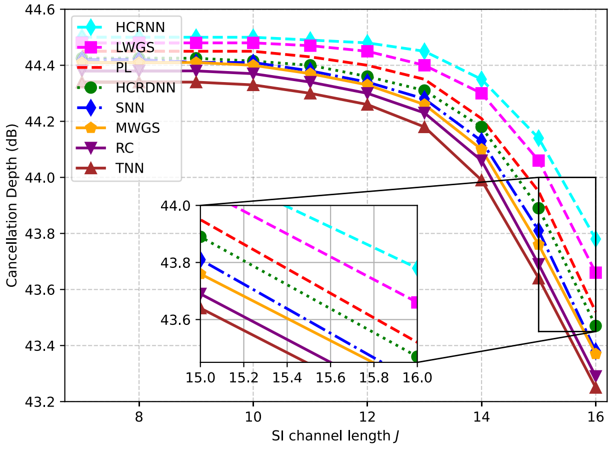

FIGURE 6. **Cancellation depth versus the SI channel length *J* of various schemes.**

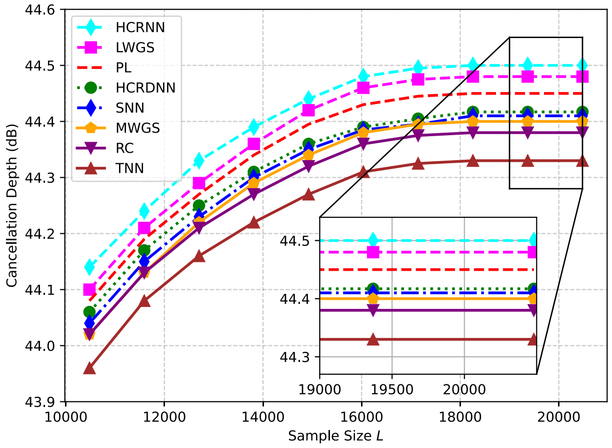

FIGURE 7. **Cancellation depth versus the sample length *L* of various schemes.**

sophisticated and computationally intensive models, straining the system's processing capabilities and adversely affecting its ability to deeply and effectively cancel interference, especially in real-time scenarios where computational efficiency is crucial.

Figure 7 presents a comparison of various DSIC schemes concerning the sample length L. As the total input data length increases in DSIC systems, the cancellation depth initially increases due to improved statistical accuracy and stability in the modelling of the self-interference channel. With more data, the algorithms have a more substantial basis to estimate and adapt to the interference characteristics, leading to more precise and effective cancellation. However, as DSIC systems process larger datasets, they require more computational resources, which do not scale linearly with the size of the data. This scaling challenge affects the algorithms' ability to operate efficiently in real-time environments, where timely processing is crucial. Additionally, the increased complexity of handling larger datasets can introduce delays, increase the computational burden, and limit the system's real-time performance capabilities. Thus, the initial gains in cancellation depth from using more extensive data can level off as the system encounters practical limits in processing capacity.

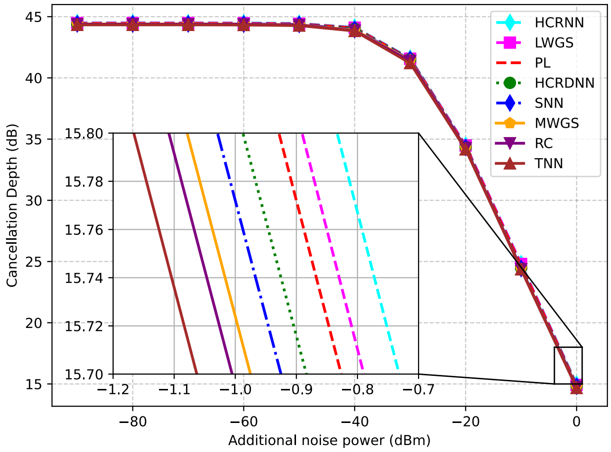

FIGURE 8. **Cancellation depth versus the additional noise power of various schemes.**

To research the DISC performance under a low signal-to-noise ratio (SNR) environment, Figure 8 compares DSIC schemes versus the additional noise power. As additional noise power is introduced, the effective SNR of the system decreases. A higher noise level masks the subtler features of the SI signal that these DSIC schemes learn to identify and cancel. This masking effect makes it increasingly difficult for the DSIC algorithms to distinguish between the noise and the actual SI, leading to a decreasing cancellation performance. This figure also suggests that RC-DSIC possesses a comparable level of resilience to noise as its counterparts. This resilience could be attributed to the inherent characteristics of RC, which can capture and process temporal patterns in data. RC's dynamic memory, provided by the reservoir, allows it to retain information over time, aiding in the identification of interference patterns even as noise levels increase. Its capacity to handle temporal dependencies efficiently enables it to adapt to increasing noise levels in a manner similar to more complex models.

C. COMPUTATIONAL COMPLEXITY COMPARISON

Figure 9 presents a comparison of computational complexity against training sample size L, with the analysis conducted over a range from 10,240 to 20,480 samples drawn from a real-world dataset totalling 20,480 samples. The observed trend of increasing computational complexity across all schemes as the sample size L rises is an expected outcome. This is because the complexity is directly proportional to the number of operations required to process each sample. Larger sample sizes mean more data points to process, which in turn increases the number of mathematical operations, such as multiplications and additions, that must be performed. Each scheme must apply its respective algorithms to a greater number of samples, thereby linearly scaling up the computational workload.

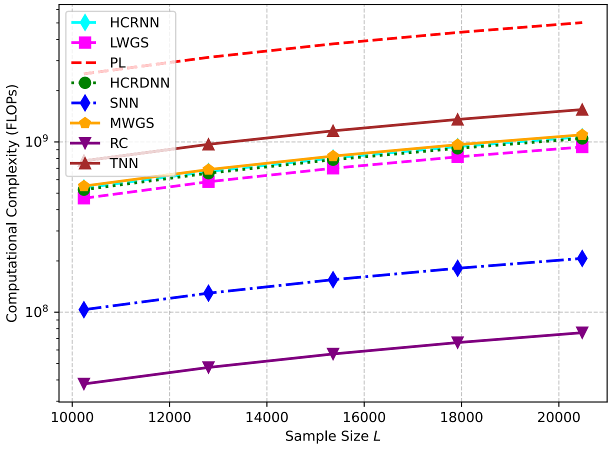

FIGURE 9. Computational complexity versus sample length *L*.

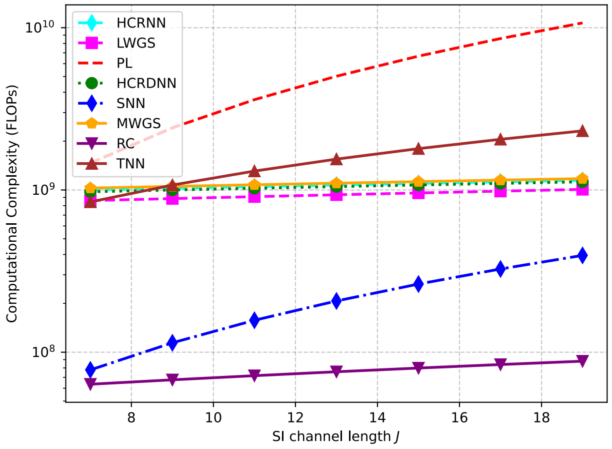

FIGURE 10. Computational complexity versus the SI channel length *J*.

Figure 10 illustrates the relationship between computational complexity and SI channel length (J) for various DSIC schemes. As J is extended, all schemes experience an uptick in computational complexity. This is because a longer SI channel length implies that the models must account for a more extended sequence of past inputs to predict or cancel the current sample. Each additional unit of SI channel requires the model to retain, process, and integrate an additional set of computations corresponding to the historical data. For the PL scheme, this means a higher number of polynomial terms to calculate; for RC and NN-based schemes, it involves additional nodes or states to update. This increased demand for historical data processing inherently escalates the number of computational steps, thus inflating the overall complexity.

Table 2 summarizes the performance comparison of various existing DSIC schemes against the proposed RC-DISC scheme, focusing on their computational complexity and the effectiveness of SI cancellation. The simulation parameters of this table are shown in Table 1. The "Relative complexity" indicates how many times the compared scheme's computational complexity is of the RC-DSIC scheme's complexity.

TABLE 2. Comparisons of the number of total FLOPs and SI cancellation effect for the existing DSIC schemes to the RC-DISC scheme.

Algorithms	FLOPs	DSIC depth (dB)	Relative complexity	Relative SIC depth (dB)
RC	75786116	44.38	× 1	+ 0.0
PL	5022269440	44.45	× 66.26	+ 0.07
TNN	1549168640	44.33	× 20.44	- 0.05
SNN	206970880	44.41	× 2.73	+ 0.03
HCRNN	915456000	44.50	× 12.08	+ 0.12
HCRDNN	890880000	44.41	× 11.76	+ 0.03
LWGS	933888000	44.48	× 12.32	+ 0.10
MWGS	1101004800	44.40	× 14.53	+ 0.02

The "Relative SIC depth" indicates the difference in the SI cancellation depth of each DSIC scheme compared to the RC-DSIC scheme, measured in dB. Notably, the RC scheme stands out for its exceptional efficiency, requiring significantly fewer FLOPs (75,786,116) while achieving an SI cancellation of 44.38 dB, which is similar to other DSIC schemes. In contrast, other algorithms exhibit a substantial increase in computational complexity. For instance, the traditional PL-based method exhibits a computational complexity that is 66.26 times that of the RC-DSIC scheme, yet it only yields a modest improvement of 0.07 dB in SIC depth. Existing NN-based methods, as exemplified by the SNN model, show at least 2.73 times the computational complexity of the RC-DSIC scheme. Among these NN-based methods, the highest improvement in SIC depth compared to the RC-DSIC scheme is a mere 0.12 dB, as demonstrated by the HCRDNN model. By calculations, the RC-DSIC approach achieves 99.84% of the efficacy provided by PL-based DSIC algorithms while necessitating only 1.51% of the computational resources. In comparison to various existing NN-DSIC strategies, the RC-DSIC technique delivers at least 99.73% of the performance while utilizing no more than 36.61% of the computational load. The SNN achieves relatively low computational complexity largely due to its adoption of a pre-trained neural network and an input preprocessing method. This analysis highlights the RC scheme's superior efficiency, achieving comparable or better SI cancellation with far less computational overhead than the other evaluated DSIC schemes.

VI. CONCLUSION

This study explored RC as a potential and lightweight solution for the digital self-interference cancellation for the IBFD model, comparing its performance against the traditional polynomial-based and the neural network-based DSIC approaches. Our findings reveal that while RC may not achieve the highest levels of performance, it excels in striking a commendable balance between efficiency and computational simplicity. The essence of RC's effectiveness lies in its ability to exploit the dynamical properties of reservoir computing, harnessing its inherent power to capture and utilize temporal dependencies in the self-interference signal. This innovative approach allows RC to provide efficient self-interference cancellation while significantly reducing computational complexity. This unique combination of capabilities positions RC as a promising solution for digital self-interference cancellation in IBFD wireless networks. Future research will delve deeper into optimizing RC parameters or integrating them with other emerging technologies to refine DSIC in IBFD wireless networks further.

APPENDIX

To enhance the readability and fluidity of our paper, emphasize the low complexity of the RC-DSIC algorithm, and increase the authenticity of the complexity comparisons in the simulations, we provide a detailed analysis of the computational complexities of the existing compared DSIC algorithms within the Appendix. Different from the complexity analysis in the reference papers, our focus is on the total computational complexities throughout the entire process.

A. COMPLEXITY OF THE PL-DSIC SCHEME

Drawing from Table 1 in [11], the computational complexity of the PL-DSIC scheme is described in the subsequent discussions.

1) BASIS MATRIX GENERATION

The initial phase of the scheme is generating the basis feature matrix. This feature matrix encapsulates the nonlinear characteristics intrinsic to self-interference. During the estimation phase, the complexity is devoid of real additions. Let the maximum nonlinearity order of the estimated model by P_{PA}. The multiplication complexity of this step can be quantified by

$$\mathcal{F}_{pl,b} = L(2n_p(P_{PA} - 1)), \tag{20}$$

where n_p represents the number of coefficients required for the polynomial-based DSIC scheme, which is represented as [20],

$$n_p = (M + J)\left(\frac{P_{PA} + 1}{2}\right)\left(\frac{P_{PA} + 3}{2}\right). \tag{21}$$

2) LEAST SQUARES ESTIMATION

After basis matrix generation, the scheme employs the least squares estimation technique. The additional and multiplication computational complexity of this phase are,

$$\mathcal{F}_{pl,la} = L(4n_p^2 - 2), \quad \mathcal{F}_{pl,lm} = L(6n_p^2 + 3n_p). \tag{22}$$

3) SELF-INTERFERENCE CANCELLATION

Finally, the derived weights are applied to cancel the self-interference. The additional and multiplication computational complexity of this phase are,

$$\mathcal{F}_{pl,sa} = L(2n_p - 2), \quad \mathcal{F}_{pl,sm} = L(3n_p). \tag{23}$$

Aggregating the complexities from each step, the overall computational complexity of the PL-DSIC scheme can be presented as,

$$\begin{aligned}\mathcal{F}_{pl} &= \mathcal{F}_{pl,b} + \mathcal{F}_{pl,la} + \mathcal{F}_{pl,lm} + \mathcal{F}_{pl,sa} + \mathcal{F}_{pl,sm} \qquad (24)\\ &= L\left(10n_p^2 + (2P_{PA} + 2)n_p - 4\right). \qquad (25)\end{aligned}$$

B. COMPLEXITY OF THE SNN AND TNN SCHEMES

To show the effectiveness of the reservoir computing scheme, the SNN approach proposed in [11] and TNN proposed in [20] are compared.

The input of the SNN-DSIC networks includes the transmitted signal in the digital domain $\mathbf{x}$ and an input generator $\mathbf{f}$. $\mathbf{f}$ is the convolution sum between the linear channel estimate $\hat{\mathbf{h}}_{ls}$ and $\mathbf{x}$. $\hat{\mathbf{h}}_{ls}$ denotes the vector of estimated coefficients acquired via LS estimation, used in the linear SI canceller for modeling and canceling the SI's linear component. The input of the SNN-DSIC networks is only the transmitted signal $\mathbf{x}$.

The neural network architectures of SNN and TNN both use multi-layer perception (MLP), which includes an input layer, a single hidden layer, and an output layer. The real and imaginary parts of the signal are fed alternately. The network's output is the estimated non-linear SI component $\hat{\mathbf{y}}_{si,\mathrm{nl}}$. Drawing from Table 2 and Table 3 in [11], the computational complexities of the SNN and TNN schemes involve the estimation and cancellation steps, which are presented as follows.

1) ESTIMATION STEPS

For the estimation step, the complexities of SNN and TNN arise from the basis matrix generation and least square estimation. However, TNN has extra steps, including the NN feedforward process, error calculation, and NN backpropagation, because SNN employs the neural network trained offline before deployment. The basis matrix generation involves no real additions and the real multiplication complexities of SNN and TNN are

$$\mathcal{F}_{snn,bg} = \mathcal{F}_{tnn,bg} = L(2n_p(P_{PA} - 1)). \tag{26}$$

The least-square estimation requires additional and multiplication complexities of SNN and TNN as

$$\mathcal{F}_{snn,la} = \mathcal{F}_{tnn,la} = L[16(M + J)^2 - 2], \tag{27}$$

$$\mathcal{F}_{snn,lm} = \mathcal{F}_{tnn,lm} = L[24(M + J)^2 + 6(M + J)]. \tag{28}$$

For TNN, the neural network feedforward process requires additional and multiplication complexities as,

$$\mathcal{F}_{tnn,fa} = N_{p,t}LN_{e,t}[2(M + J) + 3], \tag{29}$$

$$\mathcal{F}_{tnn,fm} = N_{p,t}LN_{e,t}[2(M + J) + 2]. \tag{30}$$

$N_{p,t}$ denotes the number of epochs in TNN training, and $N_{e,t}$ is the number of nodes in the hidden layer. Error calculation demands additional and multiplication complexities of TNN as

$$\mathcal{F}_{tnn,ea} = 4N_{p,t}L, \mathcal{F}_{tnn,em} = 4N_{p,t}L. \tag{31}$$

Finally, the neural network backpropagation complexities of TNN are

$$\mathcal{F}_{tnn,ba} = N_{p,t}LN_{e,t}[4(M+J)+2], \tag{32}$$

$$\mathcal{F}_{tnn,bm} = N_{p,t}LN_{e,t}[6(M+J)+6]. \tag{33}$$

2) NON-LINEAR CANCELLATION STEP

The first process in this step is the generation of the linear self-interference estimate $\hat{y}_{\mathrm{SI,lin}}$. The additional and multiplication complexities in this process of SNN and TNN are

$$\mathcal{F}_{snn,ca} = \mathcal{F}_{tnn,ca} = L\left[4(M+J)-2\right], \tag{34}$$

$$\mathcal{F}_{snn,cr} = \mathcal{F}_{tnn,ca} = L\left[6(M+J)\right]. \tag{35}$$

Then, the linear self-interference estimate from the received signal is subtracted. For SNN and TNN, the complexity of this process is characterized by a real addition complexity of $2L$ and no real multiplications. For SNN, before feeding the signal into the neural network for further interference cancellation, pre-processing is conducted. This step does not involve any real additions, but the real multiplication complexity is $L\left[3(M+J)\right]$.

The final component in the cancellation step is the generation of the non-linear self-interference estimate, represented as $\hat{y}_{\mathrm{SI,nl}}$. The additional and multiplication complexities of SNN and TNN for this component are

$$\mathcal{F}_{snn,pa} = LN_{e,s}\left[4(M+J)+3\right], \tag{36}$$

$$\mathcal{F}_{snn,pm} = LN_{e,s}\left[4(M+J)+2\right], \tag{37}$$

$$\mathcal{F}_{tnn,pa} = LN_{e,t}\left[2(M+J)+3\right], \tag{38}$$

$$\mathcal{F}_{tnn,pm} = LN_{e,t}\left[2(M+J)+2\right]. \tag{39}$$

where $N_{e,s}$ and $N_{e,t}$ denote the number of hidden neurons in the SNN and TNN neural networks respectively. The different inputs cause different complexities. Therefore, the overall computational complexity of the SNN and TNN scheme, when considering both real additions and multiplications, can be represented as

$$\begin{aligned}\mathcal{F}_{snn} &= L\{40(M+J)^2 + 19(M+J) + N_{e,s}[8(M+J)+5]\\ &\quad + 2\,n_p(P_{PA}-1) - 2\},\end{aligned} \tag{40}$$

$$\begin{aligned}\mathcal{F}_{tnn} &= L\{40(M+J)^2 + 16(M+J)\\ &\quad + N_{p,t}N_{e,t}[14(M+J)+13]\\ &\quad + N_{e,t}[4(M+J)+5] + 8N_{p,tnn}\\ &\quad + 2n_p(P_{PA}-1) - 2\}.\end{aligned} \tag{41}$$

C. COMPLEXITY OF THE HCRNN AND HCRDNN SCHEMES

The HCRNN and HCRDNN schemes are proposed in [21]. For the HCRNN-DSIC scheme, the number of FLOPs can be given by

$$\mathcal{F}_{hcrnn} = \mathcal{F}_{hcrnn,l} + \mathcal{F}_{hcrnn,c} + \mathcal{F}_{hcrnn,r} + \mathcal{F}_{hcrnn,o}, \tag{42}$$

where $\mathcal{F}_{hcrnn,l}$, $\mathcal{F}_{hcrnn,c}$, $\mathcal{F}_{hcrnn,r}$, and $\mathcal{F}_{hcrnn,o}$ denote the FLOPs required for the linear cancellation, convolutional, recurrent, and output layers of the HCRNN, respectively. The FLOPs required for the linear cancellation $\mathcal{F}_{hcrnn,l}$ can be expressed as,

$$\mathcal{F}_{hcrnn,l} = 3(M+J) + 7(M+J) - 2 = 10(M+J) - 2. \tag{43}$$

For the convolutional layer, $\mathcal{F}_{hcrnn,c}$ includes the complexities of convolution operations across all filters, applying the activation functions, and adding the bias values, expressed as,

$$\mathcal{F}_{hcrnn,c} = (2C_hC_wC_d + 1)(O_hO_wF), \tag{44}$$

where C_h, C_w, C_d are the height, width, and depth of the filter. O_h, O_w are the height and width of the output feature map. F denotes the number of filters. The activation functions are *relu* functions needing only one FLOP.

Furthermore, the number of FLOPs required for the recurrent layer includes operations for matrix multiplications, activation functions, and additions for biases, expressed as

$$\mathcal{F}_{hcrnn,r} = 2N_{r,hcrnn}\left(N_f + N_{r,hcrnn} + \frac{1}{2}\right), \tag{45}$$

with N_f as the number of input features and $N_{r,hcrnn}$ as the number of neurons in the HNRNN recurrent layer, and *ReLU* again serves as the activation function. The FLOPs for the output layer, which account for computing the weighted sum of inputs, applying an activation function, and adding biases to generate the final output, are given by

$$\mathcal{F}_{hcrnn,o} = N_o\left(2N_{r,hcrnn} + 1\right), \tag{46}$$

where N_o is the number of output layer's neurons. The only difference between the HCRNN and HCRDNN models lies in the inclusion of a dense layer in the HCRDNN, necessitating additional FLOPs for calculating the weighted inputs, activation functions, and biases at the dense layer neurons, denoted as,

$$\mathcal{F}_{hcrdnn,d} = N_d\left(2N_{r,hcrdnn} + 1\right), \tag{47}$$

where N_d is the number of neurons in the dense later. $N_{r,hcrdnn}$ is the number of neurons in the HNRDNN recurrent layer. So, in the computation of the HCRDNN's output layer FLOPs, N_o should be replaced with N_d. Let the number of training epochs of HCRNN and HCRDNN be $N_{p,hcrnn}$ and $N_{p,hcrdnn}$, respectively. The number of FLOPs for HCRNN and HCRDNN can be expressed as,

$$\begin{aligned}&\mathcal{F}_{hcrnn}\\ &= 2N_{p,hcrnn}L\,\Big[10(M+J) - 2 + (2C_hC_wC_d+1)O_hO_wF\\ &\quad +2N_{r,hcrnn}\left(N_f + N_{r,hcrnn} + N_o + \frac{1}{2}\right) + N_o\Big],\end{aligned} \tag{48}$$

$$\begin{aligned}&\mathcal{F}_{hcrdnn}\\ &= 2N_{p,hcrdnn}L\,\Big[10(M+J) - 2 + (2C_hC_wC_d+1)O_hO_wF\\ &\quad +2N_{r,hcrdnn}\left(N_f + N_{r,hcrdnn} + 2N_d + \frac{1}{2}\right) + 2N_d\Big],\end{aligned} \tag{49}$$

The factor of 2 in front of $N_{p,hcrnn}$ and $N_{p,hcrdnn}$ accounts for both the forward and backward pass complexities.

D. COMPLEXITY OF THE LWGS AND MWGS SCHEMES

The LWGS and MWGS schemes are introduced by [22]. Within the LWGS, the computational load, encompassing the multiplication of inputs or previous layer outputs by the weight matrix, the addition of bias terms, and the application of the ℂRELU activation function, is detailed as follows

$$\begin{aligned}
&\mathcal{F}_{lwgs}\\
&= 2N_{p,lwgs}L\left[3(\sum_{i=1}^{N_{e,lwgs}} i+M+J)+5(\sum_{i=1}^{N_{e,lwgs}} i+M+J)\right.\\
&\quad\left.+2(\sum_{i=1}^{N_{e,lwgs}} i+M+J)+2N_{e,lwgs}+6N_{e,lwgs}\right]\\
&= 2N_{p,lwgs}L\left[5N_{e,lwgs}(N_{e,lwgs}+1)+10(M+J)+8N_{e,lwgs}\right],
\end{aligned} \tag{50}$$

where $N_{e,lwgs}$ denotes the number of the hidden layer neurons of LWGS. Similarly, in the MWGS, the number of FLOPs can be expressed as

$$\begin{aligned}
&\mathcal{F}_{mwgs}\\
&= 2N_{p,lwgs}L\left\{3[M+J+W_s(N_{e,mwgs}-1)+N_{e,mwgs}]+\right.\\
&\quad 5[M+J+W_s(N_{e,mwgs}-1)+N_{e,mwgs}]+2[M+J+\\
&\quad \left.W_s(N_{e,mwgs}-1)+N_{e,mwgs}]+2N_{e,mwgs}+6N_{e,mwgs}\right\}\\
&= 2N_{p,mwgs}L\left\{10[M+J+W_s(N_{e,mwgs}-1)]+18N_{e,mwgs}\right\},
\end{aligned} \tag{51}$$

where W_s denotes the window size, and $N_{e,mwgs}$ is the number of the hidden layer neurons of MWGS.

REFERENCES

[1] H. Luo, A. Bishnu, and T. Ratnarajah, "In-band full-duplex radios in 6G networks: Implementation and applications," *Found. Trends Netw.*, vol. 13, no. 1, pp. 1–105, 2023.

[2] H. Luo, A. Bishnu, and T. Ratnarajah, "Design and analysis of in-band full-duplex private 5G networks using FR2 band," *IEEE Access*, vol. 9, pp. 166886–166905, 2021.

[3] J. Zhang, H. Luo, N. Garg, A. Bishnu, M. Holm, and T. Ratnarajah, "Design and analysis of wideband in-band-full-duplex FR2-IAB networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 4183–4196, Jun. 2022.

[4] D. Korpi, L. Anttila, and M. Valkama, "Nonlinear self-interference cancellation in MIMO full-duplex transceivers under crosstalk," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, pp. 1–15, Dec. 2017.

[5] D. Korpi, M. Heino, C. Icheln, K. Haneda, and M. Valkama, "Compact inband full-duplex relays with beyond 100 dB self-interference suppression: Enabling techniques and field measurements," *IEEE Trans. Antennas Propag.*, vol. 65, no. 2, pp. 960–965, Feb. 2017.

[6] A. Bishnu, M. Holm, and T. Ratnarajah, "Performance evaluation of full-duplex IAB multi-cell and multi-user network for FR2 band," *IEEE Access*, vol. 9, pp. 72269–72283, 2021.

[7] P. Aquilina, A. C. Cirik, and T. Ratnarajah, "Weighted sum rate maximization in full-duplex multi-user multi-cell MIMO networks," *IEEE Trans. Commun.*, vol. 65, no. 4, pp. 1590–1608, Apr. 2017.

[8] K. Singh, K. Wang, S. Biswas, Z. Ding, F. A. Khan, and T. Ratnarajah, "Resource optimization in full duplex non-orthogonal multiple access systems," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4312–4325, Sep. 2019.

[9] S. Biswas, K. Singh, O. Taghizadeh, and T. Ratnarajah, "Coexistence of MIMO radar and FD MIMO cellular systems with QoS considerations," *IEEE Trans. Wireless Commun.*, vol. 17, no. 11, pp. 7281–7294, Nov. 2018.

[10] A. C. Cirik, S. Biswas, S. Vuppala, and T. Ratnarajah, "Beamforming design for full-duplex MIMO interference channels–QoS and energy-efficiency considerations," *IEEE Trans. Commun.*, vol. 64, no. 11, pp. 4635–4651, Nov. 2016.

[11] D. H. Kong, Y. Kil, and S. Kim, "Neural network aided digital self-interference cancellation for full-duplex communication over time-varying channels," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6201–6213, Jun. 2022.

[12] M. Erdem, O. Gurbuz, and H. Ozkan, "Integrated linear and nonlinear digital cancellation for full duplex communication," *IEEE Wireless Commun.*, vol. 28, no. 1, pp. 20–27, Feb. 2021.

[13] L. Sun et al., "In-sensor reservoir computing for language learning via two-dimensional memristors," *Sci. Adv.*, vol. 7, no. 20, May 2021, Art. no. eabg1455.

[14] Y. K. Chembo, "Machine learning based on reservoir computing with time-delayed optoelectronic and photonic systems," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 30, no. 1, pp. 013111-1–013111-10, Jan. 2020.

[15] G. Tanaka et al., "Recent advances in physical reservoir computing: A review," *Neural Netw.*, vol. 115, pp. 100–123, Jul. 2019.

[16] D. J. Gauthier, E. Bollt, A. Griffith, and W. A. S. Barbosa, "Next generation reservoir computing," *Nature Commun.*, vol. 12, no. 1, p. 5564, Sep. 2021.

[17] Y. Zhong, J. Tang, X. Li, B. Gao, H. Qian, and H. Wu, "Dynamic memristor-based reservoir computing for high-efficiency temporal signal processing," *Nature Commun.*, vol. 12, no. 1, p. 408, Jan. 2021.

[18] M. Cucchi, S. Abreu, G. Ciccone, D. Brunner, and H. Kleemann, "Hands-on reservoir computing: A tutorial for practical implementation," *Neuromorphic Comput. Eng.*, vol. 2, no. 3, Sep. 2022, Art. no. 032002.

[19] F. M. Bianchi, S. Scardapane, S. Løkse, and R. Jenssen, "Reservoir computing approaches for representation and classification of multivariate time series," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 2169–2179, May 2021.

[20] A. Balatsoukas-Stimming, "Non-linear digital self-interference cancellation for in-band full-duplex radios using neural networks," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Kalamata, Greece, Jun. 2018, pp. 1–5.

[21] M. Elsayed, A. A. A. El-Banna, O. A. Dobre, W. Shiu, and P. Wang, "Hybrid-layers neural network architectures for modeling the self-interference in full-duplex systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6291–6307, Jun. 2022.

[22] M. Elsayed, A. A. A. El-Banna, O. A. Dobre, W. Shiu, and P. Wang, "Low complexity neural network structures for self-interference cancellation in full-duplex radio," *IEEE Commun. Lett.*, vol. 25, no. 1, pp. 181–185, Jan. 2021.

[23] M. Elsayed, A. A. A. El-Banna, O. A. Dobre, W. Shiu, and P. Wang, "Full-duplex self-interference cancellation using dual-neurons neural networks," *IEEE Commun. Lett.*, vol. 26, no. 3, pp. 557–561, Mar. 2022.

[24] M. Isaksson, D. Wisell, and D. Ronnow, "A comparative analysis of behavioral models for RF power amplifiers," *IEEE Trans. Microw. Theory Techn.*, vol. 54, no. 1, pp. 348–359, Jan. 2006.

[25] A. Balatsoukas-Stimming, A. C. Austin, P. Belanovic, and A. Burg, "Baseband and RF hardware impairments in full-duplex wireless systems: Experimental characterisation and suppression," *EURASIP J. Wireless Commun. Netw.*, vol. 2015, no. 1, pp. 1–11, Dec. 2015.

[26] Y. Kurzo, A. T. Kristensen, A. Burg, and A. Balatsoukas-Stimming, "Hardware implementation of neural self-interference cancellation," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 10, no. 2, pp. 204–216, Jun. 2020.

[27] K. Hamedani, L. Liu, and Y. Yi, "Energy efficient MIMO-OFDM spectrum sensing using deep stacked spiking delayed feedback reservoir computing," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 1, pp. 484–496, Mar. 2021.

[28] 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.