Trainable Projected Gradient Detector for Massive Overloaded MIMO Channels: Data-Driven Tuning Approach

SATOSHI TAKABE[1,2], **(Member, IEEE), MASAYUKI IMANISHI**[1],
TADASHI WADAYAMA[1], **(Member, IEEE), RYO HAYAKAWA**[3], **(Student Member, IEEE),**
AND KAZUNORI HAYASHI[4], **(Member, IEEE)**

[1]Nagoya Institute of Technology, Nagoya 466-8555, Japan
[2]RIKEN Center for Advanced Intelligence Project, Tokyo 103-0027, Japan
[3]Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
[4]Graduate School of Engineering, Osaka City University, Osaka 558-8585, Japan

Corresponding author: Satoshi Takabe (s_takabe@nitech.ac.jp)

ABSTRACT This paper presents a deep learning-aided iterative detection algorithm for massive overloaded multiple-input multiple-output (MIMO) systems where the number of transmit antennas n is larger than that of receive antennas m. Since the proposed algorithm is based on the projected gradient descent method with trainable parameters, it is named the trainable projected gradient-detector (TPG-detector). The trainable internal parameters, such as the step-size parameter, can be optimized with standard deep learning techniques, i.e., the back propagation and stochastic gradient descent algorithms. This approach is referred to as data-driven tuning and ensures fast convergence during parameter estimation in the proposed scheme. The TPG-detector mainly consists of matrix-vector product operations whose computational cost is proportional to mn for each iteration. In addition, the number of trainable parameters in the TPG-detector is independent of the number of antennas. These features of the TPG-detector result in a fast and stable training process and reasonable scalability for large systems. The numerical simulations show that the proposed detector achieves a comparable detection performance to those of the existing algorithms for the massively overloaded MIMO channels, e.g., the state-of-the-art IW-SOAV detector, with a lower computation cost.

INDEX TERMS Massive MIMO, overloaded MIMO, detection algorithm, deep learning.

I. INTRODUCTION

Multiple-input multiple-output (MIMO) signal processing is an indispensable wireless communication technology for achieving increased data transfer rates, enhanced reliability, and improved energy efficiency. In particular, *massive* MIMO systems have been widely studied because they can provide the high spectral efficiency required for upcoming communication technologies such as the 5th generation (5G) wireless network standard [1], [2]. Since tens or hundreds of antennas are used in a transmitter and receiver, signal detection for MIMO channels tends to be a computationally intensive task.

The associate editor coordinating the review of this manuscript and approving it for publication was Guan Gui.

It is thus worth studying practical massive MIMO detection algorithms which have both low computational complexity and reasonable bit error rate (BER) performance.

In a down-link massive MIMO channel with mobile terminals, a transmitter in a base station can have many antennas but a mobile terminal will have far fewer receive antennas because of restrictions on the cost, space, and power consumption. Such a system is known as an *overloaded MIMO* system, in which the number of transmit antennas n is larger than that of receive antennas m. Overloaded MIMO communications naturally arise in Internet of Things (IoT) wireless networks, i.e., data collection by a base station from a large number of sensor nodes can also be regarded as an up-link overloaded MIMO system because the number of sensor

nodes is typically greater than the number of receive antennas at the base station.

A practical detector for massive overloaded MIMO systems should have not only low computational complexity but also reasonable BER performance. However, achieving a reasonable balance between the complexity and detection performance is challenging. In fact, most naive MIMO detection algorithms have difficulties with either the detection performance or the computational complexity. For example, the zero-forcing detector and minimum mean square error (MMSE) detector [3] exhibit poor BER performance for overloaded MIMO channels, while optimal maximum likelihood (ML) detection is computationally intractable.

Among the detection algorithms for overloaded MIMO systems, several approximate ML detectors show a reasonable balance between complexity and performance. Search-based detection algorithms, such as slab-sphere decoding [4] and enhanced reactive tabu search (ERTS) [5], have been proposed for overloaded MIMO channels. Although these schemes show excellent detection performance, they are still computationally expensive for massive MIMO systems.

Some MIMO detection algorithms can be classified into the category of convex optimization-based algorithms. Fadlallah et al. proposed a MIMO detector based on ℓ_1-regularized minimization [6]. Recently, Hayakawa and Hayashi proposed an iterative detection algorithm with practical computational complexity based on *iterative weighted sum-of-absolute value* (IW-SOAV) optimization [7], [8]. The IW-SOAV provides a remarkable BER performance in the currently available overloaded MIMO detection algorithms with low computational complexity.

Recently, in the field of sparse signal recovery, deep learning techniques have attracted great interest because they can significantly improve the signal recovery performances of existing sparse signal recovery algorithms. Briefly, by unrolling the signal flow of an iterative algorithm, we can obtain a signal-flow graph similar to a feedforward neural network, where parameters are optimized by back propagation and stochastic gradient descent methods. Gregor and LeCun first proposed such an approach, called the learned iterative shrinkage-thresholding algorithm (LISTA) [9], which exhibits a sparse signal recovery performance far superior to that of the original ISTA [10]. Ito, Takabe, and Wadayama proposed the *trainable ISTA* (TISTA) [11], [12] which provides significantly faster convergence than ISTA and LISTA. Several new algorithms inspired by TISTA have also been developed: the TPG decoder for LDPC codes [13], OAMP-net for MIMO systems [14], DL-OAMP [15], [16], and SURE-TISTA [17].

The emergence of deep learning has also made a great impact on the design of algorithms for wireless communications. Deep learning-based MIMO detectors, such as the deep MIMO detectors (DMDs) in [18], [19], have been proposed based on the concept of end-to-end modeling of a detector by a neural network [20]–[22]. Although these algorithms exhibit a reasonable detection performance, they have poor

scalability because of the large number of tuning parameters, and their computational cost. It thus may be difficult to apply them to massive overloaded MIMO systems.

The goal of this paper is to propose a novel detection algorithm which is suitable for massive overloaded MIMO systems. Since the proposed algorithm is based on the trainable projected gradient (TPG) algorithm, it is called the TPG-detector [23]. The TPG-detector consists of two iterative steps: the gradient descent step and the soft projection step. These two steps include several trainable internal parameters that can be optimized with standard deep learning techniques, i.e., the back propagation and stochastic gradient descent algorithms. This approach is referred to as data-driven tuning and ensures the fast convergence of the parameter estimation in the proposed scheme.

This paper is organized as follows. In Section II, we introduce the concept of data-driven tuning for iterative algorithms and demonstrate it with a simple example. In Section III, we describe the problem setting of massive overloaded MIMO systems. Section IV is the main part of this paper, which introduces the proposed TPG-detector for massive overloaded MIMO systems. In Section V, the proposed algorithm's detection performance is compared with other algorithms such as the IW-SOAV. The last section is devoted to a summary of this paper. Appendix presents a brief review of the IW-SOAV.

II. DATA-DRIVEN TUNING

In this section, we first introduce our key design principle, called data-driven tuning, for numerical optimization algorithms. A simple example based on a toy problem related to MIMO detection problems is then presented to illustrate the basic idea behind data-driven tuning. In the numerical results, we observe the phenomenon of the data-driven acceleration of convergence for a projected gradient descent algorithm. The trainable algorithm shown in the example is then used as the base of the TPG-detector proposed in Section IV.

A. BASIC CONCEPT

We here introduce the concept of the *data-driven tuning of numerical optimization algorithms*, whose origin dates back to the work by Gregor and LeCun [9]. They unfolded an iterative optimization algorithm and embedded several trainable parameters such as matrices in gradient steps to improve its convergence performance. In general, by unfolding the iterative processes (Fig. 1 (a)) and by adding some trainable parameters, we obtain a multilayer signal-flow graph that is similar to a deep feedforward neural network (Fig. 1 (b)).

It is expected that the behavior of each process is controlled by the trainable parameters (black circles in Fig. 1(b)). If each process of the signal-flow graph is differentiable, these trainable parameters can be adjusted by standard deep learning techniques. Trainable parameters are tuned by minimizing the loss function between the supervised signal and the output at the end of the unfolded signal-flow graph. It is necessary to prepare sufficient training data to tune trainable parameters.
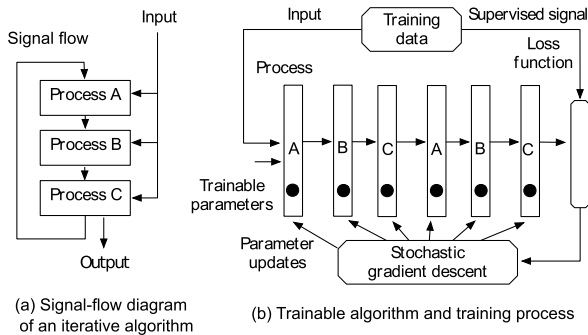
Input · · · Signal flow
Process A
Process B
Process C
Output

(a) Signal-flow diagram of an iterative algorithm

Input · Training data · Supervised signal
Process · Loss function
A B C A B C
Trainable parameters
Parameter updates · Stochastic gradient descent

(b) Trainable algorithm and training process

FIGURE 1. (a) A signal-flow diagram of an iterative algorithm, (b) data-driving tuning based on an unfolded signal-flow graph by supervised learning.

Fortunately, in problems involving wireless communications, the training data can be randomly generated according to a channel model.

In the training process, a randomly generated input is fed into the signal-flow graph and the corresponding supervised signal is also fed to the loss function (Fig. 1(b)). We apply back propagation and an SGD type parameter update (SGD, RMSprop, Adam, etc.) to optimize the parameters.

B. EXAMPLE OF DATA-DRIVEN TUNING
The aim of this subsection is to outline the use of data-driven tuning with a toy problem similar to the MIMO detection problem.

1) PROBLEM SETTING
Let us consider a simple quadratic optimization problem

$$\text{minimize}_{x\in\{-1,+1\}^n} \frac{1}{2}\|Ax - y\|_2^2, \qquad (1)$$

where $A \in \mathbb{R}^{n\times n}$ is a real-valued matrix and $\|\cdot\|_2$ represents the Euclidean norm. We assume that y is randomly generated based on a linear observation $y = A\tilde{x} + w \in \mathbb{R}^n$ where $\tilde{x}$ is a random vector uniformly distributed over $\{-1, +1\}^n$ and $w \in \mathbb{R}^n$ is an i.i.d. Gaussian random vector with zero mean and variance σ^2. The optimization problem is regarded as an ML estimation problem for the Gaussian linear vector channel, which belongs to NP-hard problems.

In order to approximately solve the problem efficiently, we consider a variant of the projected gradient (PG) algorithm. The PG algorithm considered here is given by

$$r_t = s_t + \gamma A^\top(y - As_t), \qquad (2)$$
$$s_{t+1} = \tanh(\xi r_t), \qquad (3)$$

where $t = 1, \ldots, T$ and $\tanh(\cdot)$ is calculated element-wise. In this paper, $H^\top$ represents the transpose of matrix H. The initial value is set to $s_1 = 0$.

There are two processes in the PG algorithm: In the gradient descent step (2), the vector r_t is updated along with the steepest descent direction of the objective function, i.e., $-\nabla\frac{1}{2}\|Ax - y\|_2^2 = A^\top(y - Ax)$. The parameter γ

corresponds to the step-size parameter which controls the convergence behavior such as the convergence to a fixed point and the convergence speed. In the projection step (3), we apply a soft projection function $\tanh(\cdot)$ to r_t in order to obtain the estimate s_{t+1} of the tth iteration. The soft projection ensures that the estimate takes a real value close to ±1. Although we can use a hard projection function onto the binary symbols $\{-1, +1\}$ instead of the soft projection, the PG algorithm fails to converge to a true signal as indicated in Section II-B.4. In this projection step, we have the parameter ξ which adjusts the softness of the soft projection. Note that this type of nonlinear projection has been commonly used in several iterative multiuser detection algorithms such as the soft parallel interference canceller [24].

2) TRAINABLE PG ALGORITHM
As described in the last subsection, the trainable algorithm can be constructed based on the PG algorithm by unfolding its iterative processes. We have the architecture of the trainable PG (TPG) algorithm given by

$$r_t = s_t + \gamma_t A^\top(y - As_t), \qquad (4)$$
$$s_{t+1} = \tanh(\xi r_t), \qquad (5)$$

with initial condition $s_1 = 0$. In the TPG algorithm, we have trainable parameters $\{\gamma_t\}_{t=1}^T$ in the gradient descent step.[1] Tuning these parameters also adjusts the step-size at each iteration. In the following discussion, the parameter ξ is treated as a fixed hyperparameter in Section II-B.3 and it is treated as a trainable parameter in Section II-B.4.

The trainable parameters are optimized by the standard mini-batch training. The ith training data point $d^i \triangleq (x^i, y^i)$ is randomly generated. The ith input signal $x^i \in \{-1, +1\}^n$ is chosen from the uniform distribution and the corresponding y^i is generated according to $y^i = Ax^i + w^i$ for a given A. The training dataset of size D, $\mathcal{D} \triangleq \{d^1, d^2, \ldots, d^D\}$, can be regarded as a mini-batch and is fed into the TPG algorithm simultaneously. In the following experiment, a matrix A is randomly generated for each mini-batch to simulate a realistic situation in which a channel matrix changes frequently. Here, each element of A follows the zero-mean Gaussian distribution with unit variance.

For the tth round of the training process, we feed a mini-batch with D training data points to the TPG algorithm to minimize the squared loss function

$$L(\Theta_t) \triangleq D^{-1} \sum_{d^i \in \mathcal{D}} \|x^i - \hat{x}_t(y^i)\|_2^2, \qquad (6)$$

where $\hat{x}_t(y) \triangleq s_{t+1}$ is the output of the TPG algorithm after t iterations and $\Theta_t \triangleq (\gamma_1, \ldots, \gamma_t)$ (or $\Theta_t \triangleq (\gamma_1, \ldots, \gamma_t, \xi)$) is a vector containing trainable parameters up to the tth iteration. A back propagation process evaluates the gradient $\nabla L(\Theta_t)$, which is used for updating trainable parameters Θ_t

---

[1]In the implementation, an alternative trainable parameter $\tilde{\gamma}_t^2$ is tuned instead of γ_t to satisfy $\gamma_t = \tilde{\gamma}_t^2 \geq 0$. This is also true for the TPG-detector in Section IV-A.
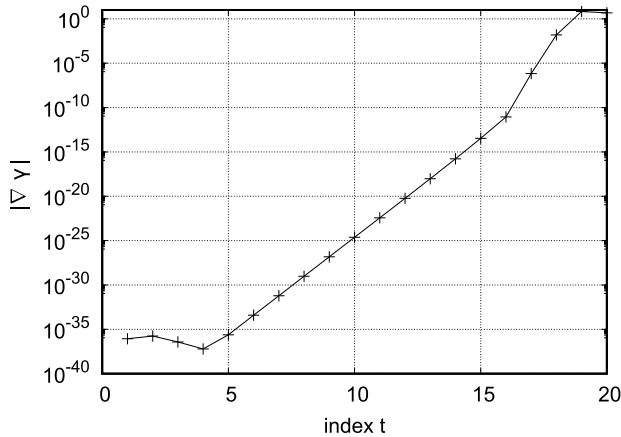
FIGURE 2. Absolute value of gradient of $\{\gamma_t\}_{t=1}^T$ in the TPG algorithm with single-shot training. The results obtained after the 10th mini-batch training by the Adam optimizer with learning rate 2.0×10^{-3} are $n = 1000$, $\sigma^2 = 4.0$, $D = 200$, and $\xi = 8.0$.
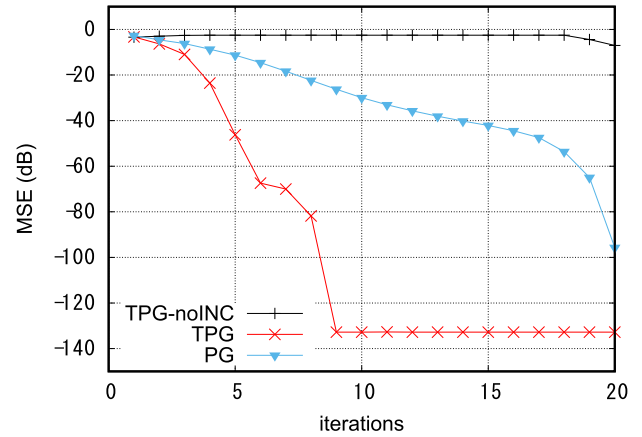

FIGURE 3. MSE as a function of the number of iterations. The curve (PG) represents the MSE of the plain PG algorithm with $\gamma = 6.5 \times 10^{-4}$ and $\xi = 6.0$. The curve (TPG) corresponds to the MSE of the TPG algorithm and the curve (TPG-noINC) corresponds to that of the TPG algorithm without incremental training. The parameter ξ is fixed at 8.0 for the TPG algorithm.

as $\boldsymbol{\Theta}_t := \boldsymbol{\Theta}_t + \boldsymbol{\Delta}$ where $\boldsymbol{\Delta}$ is given by an SGD type algorithm such as the Adam optimizer [25].

It should be remarked that a simple single-shot training using $t = T$ often fails to tune the training parameters accurately because of the vanishing gradient phenomenon. In the TPG algorithm, the phenomenon is caused by the fact that the derivative of the soft projection function (5) becomes close to zero almost everywhere. Figure 2 shows the absolute values of the gradient of the trainable parameters $\{\gamma_t\}_{t=1}^T$ in the TPG algorithm. We find that the gradient vanishes as the iteration index t becomes small. The parameter estimators fail to converge in the single-shot training (see Fig. 3). In order to improve the accuracy of the training process, we use the incremental training, which is effective in TISTA [11]. In the incremental training, the parameters $\{\gamma_t\}_{t=1}^T$ are sequentially trained from $\boldsymbol{\Theta}_1$ to $\boldsymbol{\Theta}_T$ in an incremental manner.

The details of the incremental training are as follows. First, $\boldsymbol{\Theta}_1$ is tuned by minimizing $L(\boldsymbol{\Theta}_1)$. After finishing the training of $\boldsymbol{\Theta}_1$, the values of the trainable parameters in $\boldsymbol{\Theta}_1$ are copied to the corresponding parameters in $\boldsymbol{\Theta}_2$. In other words, the results of the training for $\boldsymbol{\Theta}_1$ are carried over to $\boldsymbol{\Theta}_2$ as the initial values. For each round of the incremental training, which is called a *generation*, K mini-batches are processed.

3) EFFECT OF STEP-SIZE PARAMETER
We show the effects of the data-driven tuning in the TPG algorithm. Here, we treat the step size $\{\gamma_t\}_{t=1}^T$ as a trainable parameter, while ξ is treated as a hyperparameter. In the experiment, the dimension of matrix $\boldsymbol{A}$ is set to $n = 1000$ and we set $\sigma^2 = 4.0$. The number of iterations of the TPG algorithm is $T = 20$. We performed two types of training processes for the TPG algorithm to measure the effect of the incremental training. In the training process with incremental training, we used $K = 100$ mini-batches per generation. The mini-batch size was set to $D = 200$ and an Adam optimizer learning rate of 2.0×10^{-4} was used. In the training process without incremental training (named "TPG-noINC"

in Fig. 3), we used $K = 2000$ and $D = 200$, and the Adam optimizer learning rate was set to 2.0×10^{-3}. The initial values of the trainable parameters were $\gamma_t = 1.0 \times 10^{-4}$ ($t = 1, \ldots, T$). In this experiment, the softness parameter ξ was set as 8.0 for the TPG algorithm.

Figure 3 shows the mean squared error (MSE) as a function of the number of iterations of the plain PG algorithm when $\xi = 6.0$ and $\gamma = 6.5 \times 10^{-4}$ for the TPG algorithms with/without incremental training. The MSE after t iterations is defined by $10 \log_{10}(\mathbb{E}[\|\boldsymbol{x} - \hat{\boldsymbol{x}}_t(\boldsymbol{y})\|_2^2]/n)$ (dB) and estimated from 10^4 random samples. In the plain PG algorithm, we set $\gamma = 6.5 \times 10^{-4}$, which is the optimal value for $T = 20$ (see also Fig. 4).

From Fig. 3, it is found that the MSE of the TPG algorithm is remarkably lower than that of the plain PG algorithm. The MSE of the TPG algorithm is below -80 dB at $t = 8$, while the plain PG yields a smaller MSE after $t = 19$. In particular, the TPG algorithm shows a much faster convergence at $t = 9$, indicating that tuning the trainable parameters leads to fast convergence. This is an example of the data-driven acceleration of convergence from introducing data-driven tuning. Comparing the TPG algorithm with "TPG-noINC" in Fig. 3 without incremental training, we find that one-shot training fails to tune the trainable parameters accurately as predicted by Fig. 2. This indicates the importance of the incremental training in the data-driven tuning approach.

In Fig. 4, we show the relation between the step-size parameter γ and the MSE performance of the plain PG algorithm with $\xi = 6.0$. It can be observed that the parameter γ must be selected carefully to obtain appropriate convergence. In other words, the appropriate γ region for fast convergence is relatively narrow; specifically, we need to choose an initial estimate within the neighborhood of 6.5×10^{-4} to achieve -100 dB at $T = 200$. This means that optimization of the step size is critical even for the plain PG algorithm. In addition, the TPG algorithm achieves a lower
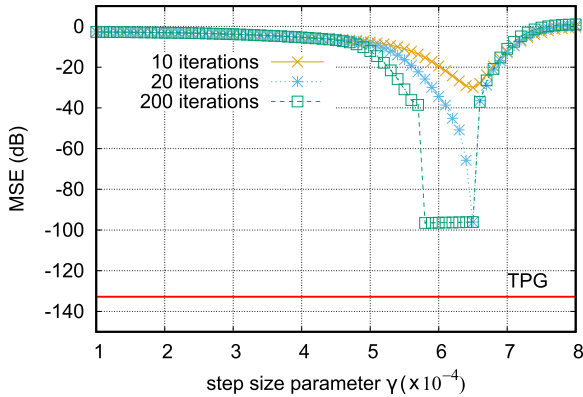
FIGURE 4. Relationship between the step-size parameter γ and the MSE performance of the plain PG algorithm with $\xi = 6.0$. The horizontal solid line represents the MSE of the TPG algorithm with $T = 20$ and $\xi = 8.0$.

FIGURE 5. MSE as a function of the number of iterations. The curves represent the MSE of the TPG algorithm with different fixed ξ or trainable ξ. The trained value is $\xi \simeq 8.560$.

MSE (around -130 dB), which cannot be achieved by the plain PG algorithm with 200 iteration steps. This fact implies that embedding a step-size parameter into each iteration step provides a substantial improvement in the speed of convergence and the accuracy of the solution.

4) EFFECT OF SOFTNESS PARAMETER

We further discuss the effect of the softness parameter ξ in (5) on the MSE performance. Figure 5 shows the MSE curves of the TPG algorithm with different values of ξ and the TPG algorithm with trainable ξ. The setting of the experiment is the same as above. As described above, the projection of the TPG algorithm uses the soft projection function instead of the hard-projection function corresponding to the $\xi \to \infty$ limit. The results show that a large fixed ξ is not appropriate in terms of the MSE. On the other hand, the TPG algorithm with small fixed ξ also shows a high MSE. It is thus crucial to tune not only the step-size parameter $\{\gamma_t\}_{t=1}^{T}$ but also the softness parameter ξ to fully utilize the benefits of the TPG algorithm. The curve with the label "TPG (ξ trained)" represents the MSE of the TPG with trainable ξ. In the experiment, we used $K = 10000$ due to the slow convergence of ξ. In Fig. 5, we can see that it outperforms other TPG algorithms with fixed ξ.

5) DISCUSSION

From the experimental results shown above, it is found that the TPG algorithm shows remarkable acceleration of the convergence speed, which we call data-driven acceleration. It is emphasized that the optimization problem considered here is randomized because its optimal solution and channel matrix are random variables. The randomized optimization problem thus has statistical properties in, e.g., gradient information and landscape of the cost function. Data-driven acceleration of convergence is a consequence of data-driven tuning that aims to learn the stochastic variations on the landscape of the objective functions. During the training process, trainable parameters are tuned to match the typical objective function.
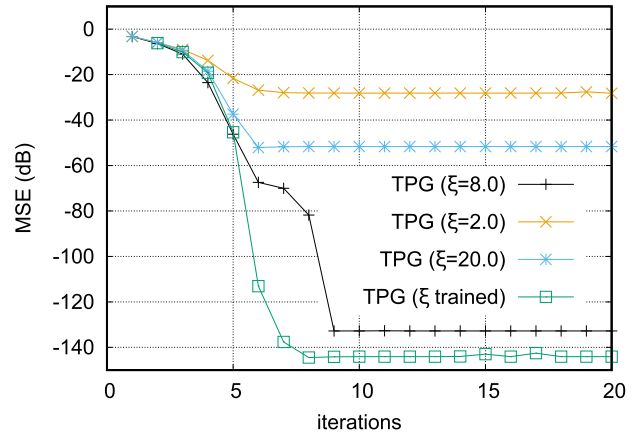
Most of the known acceleration techniques for gradient descent algorithms, such as the momentum methods, do not consider the statistical properties of the problems. On the other hand, data-driven acceleration does learn the statistical nature of the problem. The internal parameters controlling the behavior of the algorithm are adjusted to match the typical objective function via training processes. Data-driven acceleration is especially advantageous when implemented in detection algorithms because it reduces the number of iterations required without sacrificing the detection performance. This makes the algorithm faster and more computationally efficient.

III. OVERLOADED MIMO CHANNELS

In this section, we introduce the MIMO channel model used throughout the rest of the paper. The numbers of transmit and receive antennas are denoted by n and m, respectively. Our main interest lies in the overloaded MIMO scenario in which $m < n$ holds. It is also assumed that the transmitter does not use precoding and that the receiver perfectly knows the channel state information, i.e., the channel matrix.

Let $\tilde{x} \triangleq [\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n]^\mathsf{T} \in \tilde{\mathbb{S}}^n$ be a vector which consists of transmitted symbols $\tilde{x}_j$ ($j = 1, \ldots, n$) from the jth antenna. The symbol $\tilde{\mathbb{S}} \subset \mathbb{C}$ represents a signal constellation. We define $\tilde{y} \triangleq [\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_m]^\mathsf{T} \in \mathbb{C}^m$ as a vector with received symbols $\tilde{y}_i$ ($i = 1, \ldots, m$) by the ith antenna. Assuming a flat Rayleigh fading channel, the received symbol vector $\tilde{y}$ is given by

$$\tilde{y} = \tilde{H}\tilde{x} + \tilde{w}, \qquad (7)$$

where $\tilde{w} \in \mathbb{C}^m$ consists of zero-mean complex Gaussian random variables with covariance matrix $\sigma_w^2 I$. The matrix $\tilde{H} = (\tilde{h}_{i,j}) \in \mathbb{C}^{m \times n}$ is a channel matrix where $\tilde{h}_{i,j}$ is the path gain from the jth transmit antenna to the ith receive antenna. Each entry of $\tilde{H}$ independently follows a complex circular Gaussian distribution with zero mean and unit variance. It is convenient to derive an equivalent real-valued channel model

defined by

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}, \tag{8}$$

where

$$\boldsymbol{y} \triangleq \begin{bmatrix} Re(\tilde{\boldsymbol{y}}) \\ Im(\tilde{\boldsymbol{y}}) \end{bmatrix} \in \mathbb{R}^M, \quad \boldsymbol{H} \triangleq \begin{bmatrix} Re(\tilde{\boldsymbol{H}}) & -Im(\tilde{\boldsymbol{H}}) \\ Im(\tilde{\boldsymbol{H}}) & Re(\tilde{\boldsymbol{H}}) \end{bmatrix},$$

$$\boldsymbol{x} \triangleq \begin{bmatrix} Re(\tilde{\boldsymbol{x}}) \\ Im(\tilde{\boldsymbol{x}}) \end{bmatrix} \in \mathbb{S}^N, \quad \boldsymbol{w} \triangleq \begin{bmatrix} Re(\tilde{\boldsymbol{w}}) \\ Im(\tilde{\boldsymbol{w}}) \end{bmatrix} \in \mathbb{R}^M, \tag{9}$$

and $(N, M) \triangleq (2n, 2m)$. For $z \in \mathbb{C}$, $Re(z)$ and $Im(z)$ represent the real and imaginary part of z, respectively. The signal set $\mathbb{S}$ is the real counterpart of $\tilde{\mathbb{S}}$. The matrix $\boldsymbol{H} \in \mathbb{R}^{M \times N}$ is converted from $\tilde{\boldsymbol{H}}$. Similarly, the noise vector $\boldsymbol{w}$ consists of i.i.d. random variables following a Gaussian distribution with zero mean and variance $\sigma_w^2/2$. The signal-to-noise ratio (SNR) per receive antenna is then represented by

$$\mathrm{SNR} \triangleq \frac{E_s}{N_0} = \frac{2n}{\sigma_w^2}, \tag{10}$$

where $E_s \triangleq \mathbb{E}[||\tilde{\boldsymbol{H}}\tilde{\boldsymbol{x}}||_2^2]/m$ is the signal power per receive antenna and $N_0 \triangleq \sigma_w^2$ is the noise power per receive antenna. Throughout this paper, we assume the QPSK modulation format, i.e., $\tilde{\mathbb{S}} \triangleq \{1 + j, -1 + j, -1 - j, 1 - j\}$, which is equivalent to the BPSK modulation $\mathbb{S} \triangleq \{-1, +1\}$ in the corresponding real-valued channel model (8).

IV. TRAINABLE PROJECTED GRADIENT (TPG)-DETECTOR

The proposed algorithm, called the TPG-detector, is based on the TPG algorithm introduced in Section II-B. We first describe the details of the TPG-algorithm and discuss its time complexity. The key difference between the previously proposed trainable detector [23] and the TPG-detector described below is the improvement in the gradient step which leads to a significant performance improvement when the ratio $m/n(< 1)$ is sufficiently large. The TPG-detector has a lower computational cost than the OAMP-net [14] and requires a smaller number of trainable parameters than the DMD [18], [19]. These features lead to the low training and execution costs of the TPG-detector.

A. DETAILS OF TPG-DETECTOR

The ML estimation rule for the MIMO channel is given by

$$\hat{\boldsymbol{x}} \triangleq \operatorname*{argmin}_{\boldsymbol{x} \in \{-1, +1\}^N} \|\boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}\|_2^2. \tag{11}$$

An exhaustive search for the optimal solution is intractable when the system size is large because (11) is a non-convex optimization problem. Similar to Section II-B, we instead approximate the solution by using the TPG algorithm. The recursive formula of the TPG-detector is given by

$$\boldsymbol{r}_t = \boldsymbol{s}_t + \gamma_t \boldsymbol{W}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{s}_t), \tag{12}$$

$$\boldsymbol{s}_{t+1} = \tanh\left(\frac{\boldsymbol{r}_t}{|\theta_t|}\right), \tag{13}$$

where $t(= 1, \ldots, T)$ represents the index of an iteration step (or layer) and we use $\boldsymbol{s}_1 = \boldsymbol{0}$ as the initial value. The estimate of the algorithm after T iterations is given by $\hat{\boldsymbol{x}} = \boldsymbol{s}_{T+1}$.
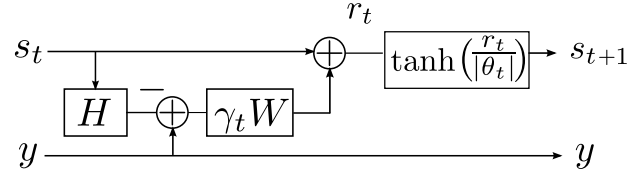

FIGURE 6. The tth layer of the TPG-detector. The trainable parameters are γ_t and θ_t.

The processes (12) and (13) correspond to the gradient descent step and the soft projection step, respectively, with soft projection function $\tanh(\cdot)$. The matrix $\boldsymbol{W}$ in the gradient step (12) is the linear MMSE (LMMSE)-like matrix defined by

$$\boldsymbol{W} \triangleq \boldsymbol{H}^\mathsf{T}(\boldsymbol{H}\boldsymbol{H}^\mathsf{T} + \alpha\boldsymbol{I})^{-1}, \tag{14}$$

where $\alpha \in \mathbb{R}$ is a trainable parameter. The matrix (14) also appears in the solution of the linear regression problem with a quadratic regularization term. We note that the OAMP-net [14] uses LMMSE-like matrices as well. The key difference from the TPG-detector is that the LMMSE-like matrix needs to be computed at each iteration in the OAMP-net. In fact, in the tth iteration step of the OAMP-net, the matrix is given as $\boldsymbol{W}_t$ by substituting the error variance with α in (14).

In our previous work [23], the Moore-Penrose pseudo-inverse matrix $\boldsymbol{H}^\mathsf{T}(\boldsymbol{H}\boldsymbol{H}^\mathsf{T})^{-1}$ was applied to the matrix $\boldsymbol{W}$ as in the orthogonal approximate message passing (OAMP) algorithm [26] and TISTA [11], [12], although a naive gradient descent method for (11) sets $\boldsymbol{H}^\mathsf{T}$ to $\boldsymbol{W}$ as in (4). However, as we will see in Fig. 8, the TPG-detector with the Moore-Penrose pseudo-inverse matrix has a poor detection performance when the ratio m/n is relatively large. Here, we instead use the LMMSE-like matrix (14) for $\boldsymbol{W}$.

Since it is critical to tune α to obtain a reasonable detection performance, the parameter α is also optimized in the training process. To reduce the number of trainable parameters and the computational cost, we assume that the same value of α is used at each iteration.

B. TRAINABLE PARAMETERS

The trainable parameters of the TPG-detector are $2T + 1$ real scalar variables α, $\{\gamma_t\}_{t=1}^T$, and $\{\theta_t\}_{t=1}^T$. The parameters $\{\gamma_t\}_{t=1}^T$ in the gradient step control the step size of the update. It should be remarked that similar trainable parameters are also introduced in the structure of TISTA [11], [12]. The parameters $\{\theta_t\}_{t=1}^T$ control the softness of the soft projection in (13). Different from the TPG algorithm, the trainable parameters depend on the iteration index t to increase the degree of freedom of the trainable parameters in the soft projection functions. The parameter α adjusts the degree of compensation for an ill-conditioned matrix $\boldsymbol{H}$. Apart from $\{\gamma_t\}_{t=1}^T$ and $\{\theta_t\}_{t=1}^T$, the TPG-detector uses the same parameter α for all iterations.

One of the advantages of the TPG-detector is that the number of trainable parameters is small, i.e., $O(T)$, and this

leads to a fast and stable training process. The number of trainable parameters in the TPG-detector does not depend on the number of antennas n and m although the DMD [18] contains $O(n^2 T)$ parameters in T layers. Since the computational cost for the training process is roughly proportional to the number of trainable parameters, the TPG-detector results in a remarkably fast training process, similarly to TISTA [12].

C. TIME COMPLEXITY

The computational complexity of the TPG-detector per iteration is $O(mn)$ because calculating the vector-matrix products $\boldsymbol{H s}_t$ and $\boldsymbol{W}(\boldsymbol{y} - \boldsymbol{H s}_t)$ takes $O(mn)$ computational steps. We need to calculate the LMMSE-like matrix $\boldsymbol{W}$ with $O(m^3)$ computational steps because the calculation involves an matrix inversion. However, the calculation of $\boldsymbol{W}$ is required only when $\boldsymbol{H}$ changes; i.e., the matrix inversion is not needed for each iteration of a TPG-detector if $\boldsymbol{H}$ is constant during the process. The total computational cost for the TPG-detector (for T iterations) without the initialization process for $\boldsymbol{W}$ is thus $O(mnT)$.

Another TISTA-based MIMO detection algorithm named OAMP-net [14] also uses the LMMSE matrix as a linear estimator. However, its computational cost is higher than that of the TPG-detector because the OAMP-net needs to compute a matrix inversion at each iteration [14]. The total computational complexity of the OAMP-net with T iteration steps is $O(m^3 T)$, which is larger than that of the TPG-detector. It should be emphasized that the computational cost affects not only the detection processes but also to the training processes.

D. TRAINING PROCESS

The TPG-detector is trained based on the incremental training described in Section II-B.2. The training data are generated randomly according to the channel model (8) with fixed variance σ_w^2 corresponding to a given SNR. As described in Section III, we assume a practical situation in which a channel matrix $\boldsymbol{H}$ is a random variable. According to this assumption, a matrix $\boldsymbol{H}$ is randomly generated for each mini-batch in the training process of the TPG-detector.

V. NUMERICAL RESULTS

In this section, we present the detection performance of the TPG-detector and compare it to that of other algorithms such as the IW-SOAV, which is known as one of the most efficient iterative algorithms for massive overloaded MIMO systems.

A. EXPERIMENTAL SETUP

In numerical experiments, we uniformly sample a transmitted vector $\boldsymbol{x}$ and generate random channel matrices for BER estimation. The BER is evaluated for a given SNR.

The TPG-detector was implemented with PyTorch 0.4.0 [27]. In this paper, a training process is executed with $T = 50$ rounds using the Adam optimizer [25]. To calculate the BER of the TPG-detector, a sign function $\mathrm{sgn}(z)$ which takes a

value of -1 if $z \leq 0$ and 1 otherwise is applied to the output $\boldsymbol{s}_{T+1}$ in an element-wise manner.

For comparison, we use the ERTS [5], IW-SOAV [8], and the standard MMSE detector. The ERTS is a heuristic algorithm based on a tabu search for overloaded MIMO systems. The parameters of ERTS are based on those given in the original work [5]. The computational complexity for executing the main loop of the ERTS algorithm is $O(N_{\mathrm{RTS}} n^2)$ where N_{RTS} is the maximum number of RTSs which is set to 500 in this paper.

The IW-SOAV is a double loop algorithm for massive overloaded MIMO systems (see the Appendix for a brief review). Its inner loop is the W-SOAV optimization, which recovers a signal using a proximal operator. Each round of the W-SOAV takes $O(mn)$ computational steps, which is comparable to that of the TPG-detector. After finishing an execution of the inner loop with K_{itr} iterations, several parameters are then updated in a re-weighting process based on a tentative recovered signal. This procedure is repeated L times in the outer loop. The total number of steps of the IW-SOAV is thus $K_{\mathrm{itr}} L$ and the total computational cost is $O(K_{\mathrm{itr}} Lmn)$ without pre-computation. In the following, we use the simulation results in [8] with $K_{\mathrm{itr}} = 50$. In this case, the computational cost of the TPG-detector is roughly equal to that of the IW-SOAV with $L = 1$ if m/n is a constant. If $L \geq 2$, i.e., the outer re-weighting process is executed, and the TPG-detector has lower computational complexity than the IW-SOAV.

B. MAIN RESULTS

1) SELECTION OF MATRIX W IN THE GRADIENT STEP

As described above, we choose the matrix $\boldsymbol{W}$ in the gradient step (12). The selection of the matrix will affect the detection performance of the algorithms as shown in the OAMP [26]. Before we compare the BER performance of the proposed TPG-detector with other detection algorithms, we numerically test the effect of the choice of $\boldsymbol{W}$. Figure 7 shows the detection performance of the TPG-detector with a different choice of $\boldsymbol{W}$ when $(n, m) = (150, 96)$. We examined three types of matrices: the matched filter matrix (MF) $\boldsymbol{H}^{\mathsf{T}}$, pseudo-inverse matrix (PINV) $\boldsymbol{H}^{\mathsf{T}}(\boldsymbol{H H}^{\mathsf{T}})^{-1}$, and LMMSE-like matrix (LMMSE) (14). From Fig. 7, we find that the LMMSE-like matrix outperforms other choices of $\boldsymbol{W}$ in a wide range of SNR.

Figure 8 shows the BER performances of PINV and LMMSE as a function of the ratio m/n when $n = 50$ and SNR= 20 dB. Although the BER performance of LMMSE is close to that of PINV when $m/n \leq 0.7$, LMMSE shows lower BER values when the ratio m/n is close to one. This is because the random matrix $\boldsymbol{H}$ tends to be ill-conditioned, i.e., the condition number of $\boldsymbol{H}$ increases as $m/n(< 1)$ increases. The LMMSE-like matrix successfully compensates the effect of the condition number. In fact, as shown in Fig. 9, the learned value of α becomes large in the high m/n region. Otherwise, the learned α is close to zero, which suggests that LMMSE corresponds to PINV without α. From these observations,
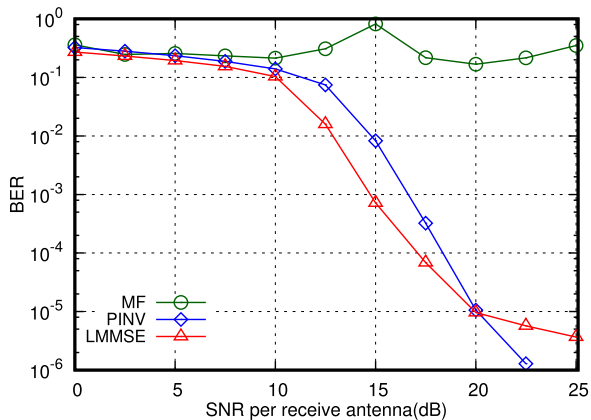
FIGURE 7. BER performances for $(n, m) = (150, 96)$ for different choices of W; SNR= 20 dB. The label "MF" stands for the matched filter which corresponds to the gradient of the original problem (11), i.e., $W \triangleq H^{\mathsf{T}}$, "PINV" represents the pseudo-inverse matrix $W \triangleq H^{\mathsf{T}}(HH^{\mathsf{T}})^{-1}$, and "LMMSE" is the LMMSE-like matrix (14).
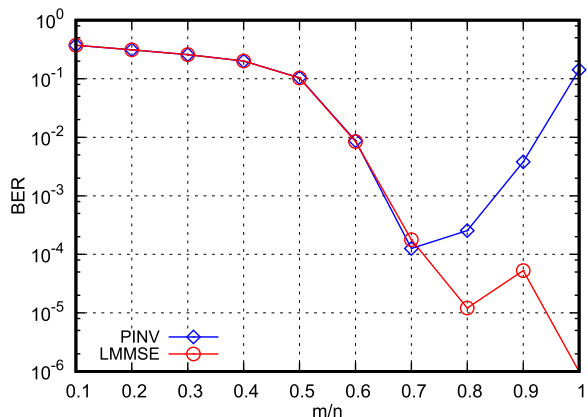
FIGURE 8. BER performances of the TPG-detector with different choices of W as a function of m/n; $n = 50$, SNR= 20 dB, $T = 50$. The label "PINV" represents the pseudo-inverse matrix $W \triangleq H^{\mathsf{T}}(HH^{\mathsf{T}})^{-1}$ and "LMMSE" is the LMMSE-like matrix (14).
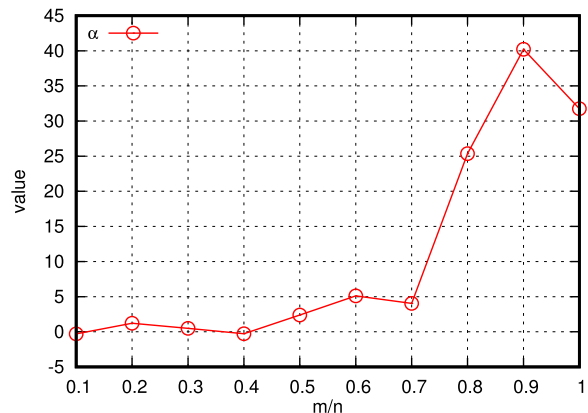
FIGURE 9. The learned values of α in the TPG-detector with the LMMSE-like matrix (14); $n = 50$, SNR= 20 dB, $T = 50$.

(a) t=1

(b) t=5

(c) t=10

FIGURE 10. Output s_{t+1} of the TPG-detector (blue bars) after (a) $t = 1$, (b) $t = 5$, (c) $t = 10$ iterations; $(n, m) = (25, 16)$, SNR= 20 dB, $T = 10$. Orange bars represent the true signals for comparison.

we set the LMMSE-like matrix to W in the proposed TPG-detector to treat the ill-conditioned matrix H.

2) SIGNAL DETECTION PROCESS

We next demonstrate the detection process of the TPG-detector. Figure 10 shows the output s_{t+1} of the TPG-detector
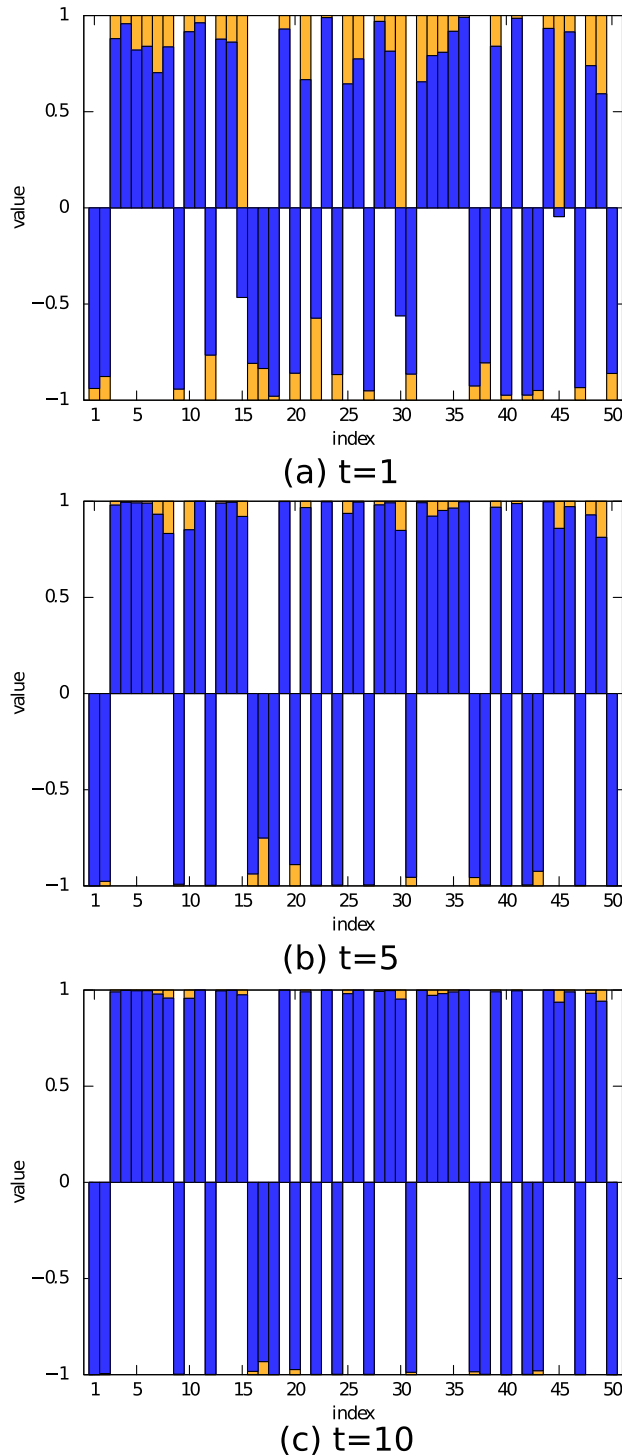
$(T = 10)$ after $t = 1, 5,$ and 10 iterations. The system size is $(n, m) = (25, 16)$ and the SNR is set to 20 dB. For comparison, the true signal is also shown in the figure. We find that the TPG-detector $(t = 1)$ mistakenly detects a few elements of the input signal as shown in Fig. 10 (a). After the 5th iteration, as shown in Fig. 10 (b), no bit errors
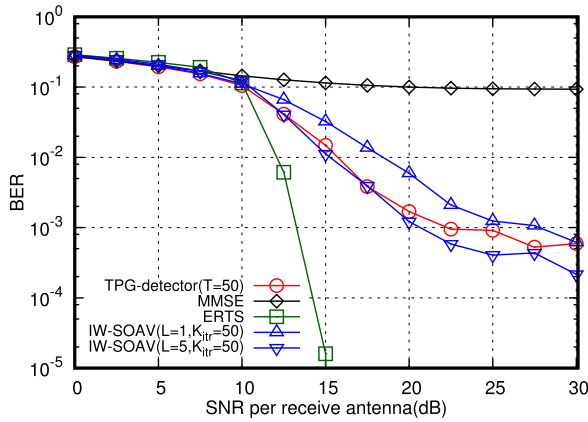
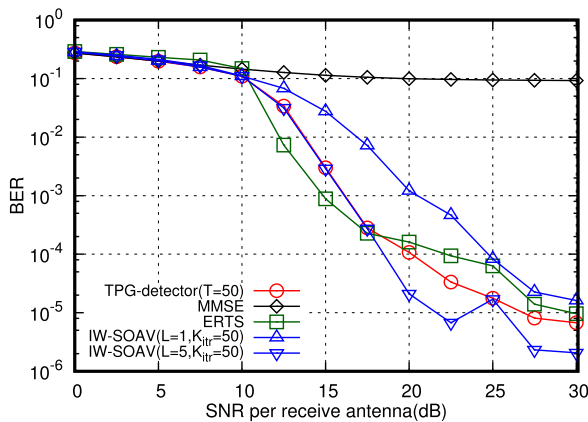**FIGURE 11.** BER performances for $(n, m) = (50, 32)$.

FIGURE 12. BER performances for $(n, m) = (100, 64)$.

FIGURE 13. BER performances for $(n, m) = (150, 96)$.

occur while some values of s_6 are not ± 1 because of the soft projection in (13). However, after some additional iterations, all elements of the output are sufficiently close to ± 1 (see Fig. 10 (c)) indicating that the MSE loss (6) is decreasing.

3) BER PERFORMANCES

We present the BER performance of each detector as a function of SNR for $(n, m) = (50, 32)$, $(100, 64)$, and $(150, 96)$ in Figs. 11-13, respectively. All the results show that the MMSE detector fails to detect transmitted signals reliably (BER $\simeq 10^{-1}$) because the system is underdetermined.

For $(n, m) = (50, 32)$ in Fig. 11, ERTS outperforms the other detection algorithms by a large margin when SNR is larger than 10 dB. It should be remarked that ERTS has a much greater time complexity (around several orders of magnitude) than that of the IW-SOAV (see Fig. 7 in [8]). Comparing the TPG-detector with the IW-SOAV, we find that the TPG-detector performs far better than the IW-SOAV ($L = 1$) and shows a BER performance close to the IW-SOAV ($L = 5$) when SNR is below 20 dB. Note that the computational cost for executing the TPG-detector with $T = 50$ is close to that of the IW-SOAV ($L = 1$). The IW-SOAV ($L = 5$) requires $K_{itr}L = 250$ iterations, which is 5 times as many as the number of iterations required for the TPG-detector with $T = 50$.
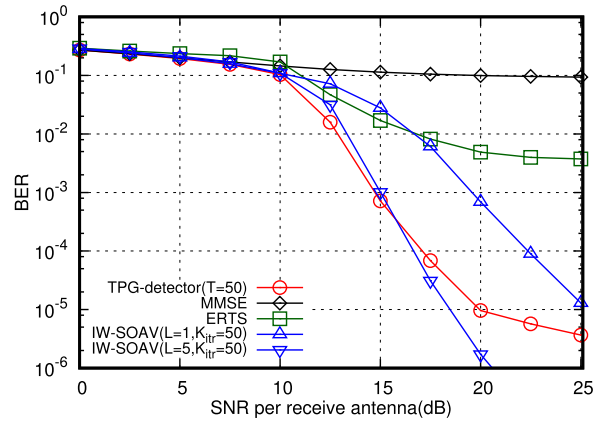
This implies that the TPG-detector can achieve a good detection performance with a relatively low computational cost.

For $(n, m) = (100, 64)$ in Fig. 12, ERTS detector shows the best BER performance in a middle SNR region where SNR is between 10 and 18 dB but the BER curve of ERTS is saturated after 20 dB. The TPG-detector and the IW-SOAV ($L = 1$ and $L = 5$) outperform ERTS in a high SNR regime. In such a regime, the TPG-detector exhibits a BER performance superior to that of the IW-SOAV ($L = 1$) for the entire range of SNR, i.e., the TPG-detector achieves an approximately 5 dB gain at BER $= 10^{-4}$ over the IW-SOAV ($L = 1$). More interestingly, the BER performance of the TPG-detector is fairly close to that of the IW-SOAV ($L = 5$) when the SNR is below 20 dB. When SNR $= 20$ dB, the BER estimate of the TPG-detector is 1.0×10^{-4} whereas that of the IW-SOAV ($L = 5$) is 2.1×10^{-5}.

Figure 13 shows the BER performance for $(n, m) = (150, 96)$. In this case, ERTS shows a poor BER performance, and cannot achieve a BER smaller than 10^{-3} for any SNR. The TPG-detector successfully recovers transmitted signals with lower BER than that of the IW-SOAV ($L = 1$). It again achieves about a 5 dB gain against the IW-SOAV ($L = 1$) at BER $= 10^{-5}$. In addition, the TPG-detector achieves the lowest BER when SNR $= 12.5$ dB. Although the IW-SOAV ($L = 5$) shows a considerable performance improvement when SNR > 15 dB, the gap between the curves of the TPG-detector and the IW-SOAV ($L = 5$) is only 2 dB at BER $= 10^{-5}$.

4) SYSTEM-SIZE DEPENDENCY

In Fig. 14, we show the BER performances of the TPG-detector and IW-SOAV ($L = 1$) as a function of the number of antennas n with the rate $m/n = 0.6$ fixed. The gap in their BER performances is especially large for SNR$= 20$ dB. We also find that the gain of the TPG-detector increases as n grows despite these algorithms having the same computational costs. It is confirmed that the TPG-detector outperforms low-complexity algorithms especially in the massive overloaded MIMO channels.
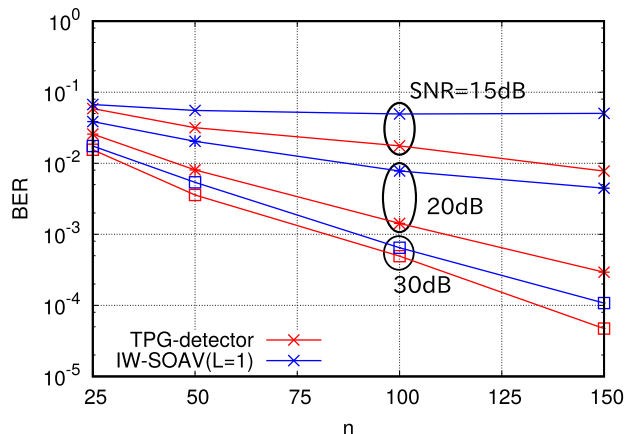
FIGURE 14. BER performances as functions of the number of antennas n for fixed rate $m/n = 0.6$ with SNR= 15, 20, 30 dB.

TABLE 1. Execution time for training processes of the TPG-detector.

(n, m)	$(50, 32)$	$(100, 64)$	$(150, 96)$
Exec. time (sec)	638.7	853.2	1203

5) TRAINED PARAMETERS

Figure 15 displays the learned parameters $\{\gamma_t\}_{t=1}^T$ and $\{|\theta_t|\}_{t=1}^T$ of the TPG-detector after training as a function of the iteration index $t(= 1, \ldots, T)$. We find that they exhibit a zigzag shape with a damping amplitude similar to that observed in TISTA [11]. The parameter γ_t, the step size of the linear estimator, is expected to accelerate the convergence of the signal recovery. Theoretical treatments for providing a reasonable interpretation of the characteristic shapes of the learned parameters are left as future work.

The trained values of α for different SNR values are shown in Fig. 16. We find that the parameter α depends on the value of SNR. In particular, the trained value decreases when SNR$\leq$ 7.5 dB. This tendency is similar to that of another parameter related to α in the IW-SOAV [8] using the LMMSE-like matrix. On the other hand, the trained value is non-monotonic unlike the IW-SOAV; specifically, it increases when SNR$<$ 7.5 dB. The parameter corresponding to α in the IW-SOAV should be chosen in advance by numerical simulations. The learning process in the TPG-detector easily tunes the parameter α in addition to other trainable parameters.

6) COMPUTATION TIME

We finally discuss the scalability of the TPG-detector to show the required computation time for training. The empirical execution time of the training process of the TPG-detector is measured by using a PC with GPU NVIDIA GeForce GTX 1080 and Intel Core i7-6700K CPU 4.0 GHz with 8 cores. Table 1 presents the execution time of the training processes with different n. Even for the case $(n, m) = (150, 96)$, we need only 20 minutes for training the TPG-detector and this result indicates that the training process of the TPG-detector is practical for fairly large systems.
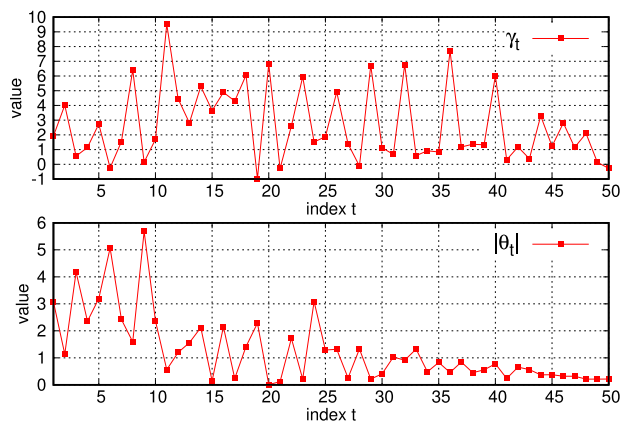

FIGURE 15. Sequences of learned parameters γ_t (top) and $|\theta_t|$ (bottom); $(n, m) = (150, 96)$, SNR = 20 dB, $1 \leq t \leq T = 50$. The trained value of α is 34.68.
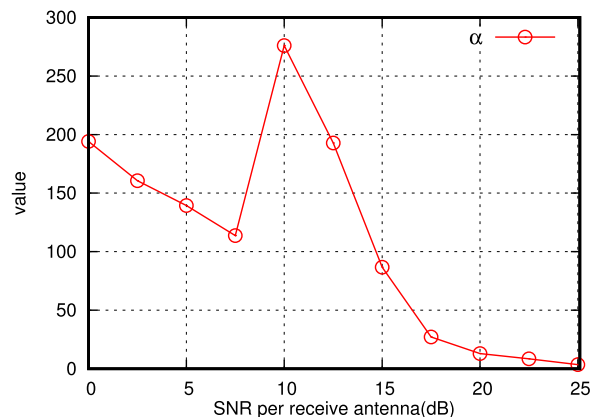

FIGURE 16. The trained α values for different SNR values; $(n, m) = (150, 96)$ and $T = 50$.

VI. CONCLUSION

In this paper, we proposed the TPG-detector, a deep learning-aided iterative decoder for massive overloaded MIMO channels. It is based on the concept of data-driven tuning using standard deep-learning techniques. The TPG-detector contains two trainable parameters for each layer: γ_t controlling the size of the gradient descent step and θ_t controlling the softness of the soft projection. In addition, the parameter α in the LMMSE-like matrix W (14) is also optimized in the training process. The total number of trainable parameters in T layers is thus $2T + 1$, which is significantly smaller than used in previous studies, such as for the DMD [18], [19]. This promotes fast and stable training for the TPG-detector.

The computational complexity of the TPG-detector with T iteration steps without the initialization on W is $O(mnT)$. This is an advantage over the OAMP-net [14] which needs a matrix inversion for each iteration and has a time complexity of $O(m^3T)$. This indicates that the TPG-detector is more scalable for massive MIMO systems in terms of the computational cost.

Numerical simulations show that the use of the LMMSE-like matrix successfully improves the BER performance of the TPG-detector even when the ratio m/n is relatively large, i.e., the channel matrix is ill-conditioned. It is also revealed that the proposed TPG-detector outperforms the state-of-the-art IW-SOAV ($L = 1$) by a large margin and achieves a comparable detection performance to the IW-SOAV ($L = 5$). The TPG-detector therefore can be seen as a promising iterative detector for massive overloaded MIMO channels providing an excellent balance between a low computational cost and a reasonable detection performance.

In this paper, we treat MIMO systems with QPSK modulation by separating the real and imaginary part of the signals. It is expected that the extension of the TPG-detector to 16QAM or 64QAM modulations is straightforward by using corresponding MMSE functions as a soft projection function (see [12] for details), which is left for a future work here. In contrast, this approach will fail for 2^K-PSK ($K > 2$) modulations because it neglects correlations in the modulation format. Instead, it is crucial to treat complex-valued signals directly as shown in some recent studies [28], [29]. As a trainable algorithm based on data-driven tuning, some of the authors recently proposed the complex-field TISTA for linear and nonlinear inverse problems in the complex domain [30]. Applying this approach to a massive overloaded MIMO system is a future research task.

APPENDIX
BRIEF REVIEW OF IW-SOAV
Here, we give a brief review of the IW-SOAV detector. The IW-SOAV is an effective iterative detection algorithm for massive overloaded MIMO systems proposed in [7], [8]. It is based on a variant of the Douglas-Rachford algorithm [31] which solves the following weighted SOAV (W-SOAV) optimization problem:

$$\hat{s} \triangleq \operatorname{argmin}_{z \in \mathbb{R}^{2n}} \left(\sum_{j=1}^{2n} w_j^+ |z_j - 1| + \sum_{j=1}^{2n} w_j^- |z_j + 1| + \frac{\alpha}{2} \|y - Hz\|_2^2 \right), \quad (15)$$

where z_j ($j = 1, \ldots, 2n$) is the jth element of z and $\alpha(> 0)$ is a constant. Here, we assume that each symbol x_j in the transmitted signal x is an independent random variable which takes a value of 1 w.p. w_j^+ and -1 w.p. $w_j^- \triangleq 1 - w_j^+$.

The IW-SOAV repeats the following procedures: (i) estimation of w_j^+ based on the detected signal and (ii) detection of the transmitted signal by solving the W-SOAV optimization (15). The IW-SOAV is thus a double-loop algorithm.

In the outer loop corresponding to procedure (i), the algorithm approximates $\{w_j^+\}$ for each transmitted symbol. The estimation is based on the approximate log likelihood ratio

which is given by

$$\hat{\Lambda}_j = \sum_{i=1}^{2m} \frac{2h_{i,j}\{y_i - (\hat{\mu}_i - h_{i,j}\hat{s}'_j)\}}{\hat{\sigma}_i^2 - h_{i,j}^2(1 - \hat{s}'^2_j)}, \quad (16)$$

where $h_{i,j}$ is the (i, j)th element of matrix H and $\hat{s}'$ represents a clipped signal of $\hat{s}$; specifically i.e., $\hat{s}'_j$ ($j = 1, \ldots, 2n$) takes a value of -1 if $\hat{s}_j < -1$, and 1 if $\hat{s}_j > 1$, and s_j otherwise. In addition, we define

$$\hat{\mu}_i \triangleq \sum_{k=1}^{2n} h_{i,k}\hat{s}'_k, \quad (17)$$

$$\hat{\sigma}_i^2 \triangleq \sum_{k=1}^{2n} h_{i,k}^2(1 - \hat{s}'^2_k) + \frac{\sigma_w^2}{2}, \quad (18)$$

for $i = 1, \ldots, 2m$. Then, the weight w_j^+ is calculated by

$$w_j^+ = \frac{e^{\hat{\Lambda}_j}}{1 + e^{\hat{\Lambda}_j}}. \quad (19)$$

In the inner loop corresponding to procedure (ii), the algorithm solves the W-SOAV optimization problem with an iterative process defined by the following recursive formula:

$$z_t = (I + \alpha\gamma H^\mathsf{T} H)^{-1}(r_t + \alpha\gamma H^\mathsf{T} y) \quad (20)$$
$$r_{t+1} = r_t + \theta_t(\phi_\gamma(2z_t - r_t) - z_t), \quad (21)$$

where $t(= 1, \ldots, K_{\text{itr}})$ denotes the iteration step, $\theta_t \in [\epsilon, 2 - \epsilon]$ is a constant, and $\phi_\gamma : \mathbb{R}^N \to \mathbb{R}^N$ is a component-wise function whose jth element $[\phi_\gamma(z)]_j$ is defined by

$$[\phi_\gamma(z)]_j \triangleq \begin{cases} z_j + \gamma & (z_j < -1 - \gamma) \\ -1 & (-1 - \gamma \le z_j < -1 - d_j\gamma) \\ z_j + d_j\gamma & (-1 - d_j\gamma \le z_j < 1 - d_j\gamma) \\ 1 & (1 - d_j\gamma \le z_j < 1 + \gamma) \\ z_j - \gamma & (1 + \gamma \le z_j), \end{cases} \quad (22)$$

with $d_j \triangleq w_j^+ - w_j^-$. The choice of the parameters $\gamma > 0$, $\epsilon \in (0, 1)$, and the initial value $r_0 \in \mathbb{R}^{2n}$ are arbitrary. In this W-SOAV optimizer, the transmitted symbol is received as $\hat{x} = z_{K_{\text{itr}}+1}$ after K_{itr} iteration steps. The IW-SOAV starts with $\hat{s} = 0$ and repeats L outer loops with K_{itr} inner loops. When all loops are finished, the sign function $\operatorname{sgn}(\cdot)$ is applied to the output $\hat{x}$ in an element-wise manner. The parameter α is fixed appropriately depending on SNR. In numerical experiments in Section V, we used $r_0 = 0$, $\epsilon = 0$, $\gamma = 1$, and $\theta_t = 1.9$ ($t = 1, \ldots, K_{\text{itr}}$), and set α as in [8].

The computational cost of each iteration of the IW-SOAV is $O(mn)$. Although it contains a matrix inversion operation which takes $O(n^3)$ computational steps, the inversion can be computed in advance. Since the total number of inner and outer loops is $K_{\text{itr}}L$, the computational cost of the IW-SOAV without pre-computation is $O(K_{\text{itr}}Lmn)$.
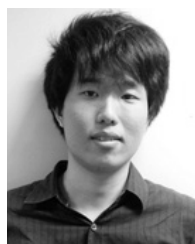
REFERENCES

[1] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[2] S. Yang and L. Hanzo, "Fifty years of MIMO detection: The road to large-scale MIMOs," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 1941–1988, 4th Quart., 2015.

[3] D. A. Shnidman, "A generalized nyquist criterion and an optimum linear receiver for a pulse modulation system," *Bell Syst. Tech. J.*, vol. 46, no. 9, pp. 2163–2177, Nov. 1967.

[4] K. K. Wong, A. Paulraj, and R. D. Murch, "Efficient high-performance decoding for overloaded MIMO antenna systems," *IEEE Trans. Wireless Commun.*, vol. 6, no. 5, pp. 1833–1843, May 2007.

[5] T. Datta, N. Srinidhi, A. Chockalingam, and B. S. Rajan, "Low-complexity near-optimal signal detection in underdetermined large-MIMO systems," in *Proc. IEEE NCC*, Feb. 2012, pp. 1–5.

[6] Y. Fadlallah, A. Aïssa-El-Bey, K. Amis, D. Pastor, and R. Pyndiah, "New decoding strategy for underdetermined MIMO transmission using sparse decomposition," in *Proc. Proc. 21st Eur. Signal Proc. Conf.*, Sep. 2013, pp. 1–5.

[7] R. Hayakawa, K. Hayashi, H. Sasahara, and M. Nagahara, "Massive overloaded MIMO signal detection via convex optimization with proximal splitting," in *Proc. 24th Eur. Signal Proc. Conf.*, Aug. 2016, pp. 1383–1387.

[8] R. Hayakawa and K. Hayashi, "Convex optimization-based signal detection for massive overloaded MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7080–7091, Nov. 2017.

[9] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. 27th Int. Conf. Mach. Learn.*, Jun. 2010, pp. 399–406.

[10] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math.*, vol. 57, no. 11, pp. 1413–1457, Nov. 2004.

[11] D. Ito, S. Takabe, and T. Wadayama, "Trainable ISTA for sparse signal recovery," in *Proc. IEEE Int. Conf. Commun. Workshops*, May 2018, pp. 1–6.

[12] D. Ito, S. Takabe, and T. Wadayama, "Trainable ISTA for sparse signal recovery," *IEEE Trans. Sig. Process.*, vol. 67, no. 12, pp. 3113–3125, Jun. 2019.

[13] T. Wadayama and S. Takabe, "Deep learning-aided trainable projected gradient decoding for LDPC codes," 2019, *arXiv:1901.04630*. [Online]. Available: https://arxiv.org/abs/1901.04630

[14] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "A model-driven deep learning network for MIMO detection," 2018, *arXiv:1809.09336*. [Online]. Available: https://arxiv.org/abs/1809.09336

[15] J. Zhang, H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning based on orthogonal approximate message passing for CP-free OFDM," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Aug. 2019, pp. 8414–8418.

[16] J. Zhang, C.-K. Wen, S. Jin, and G. Y. Li, "Artificial intelligence-aided receiver for A CP-free OFDM system: Design, simulation, and experimental test," 2019, *arXiv:1903.04766*. [Online]. Available: https://arxiv.org/abs/1903.04766

[17] M. Yao, J. Dang, Z. Zhang, and L. Wu, "SURE-TISTA: A signal recovery network for compressed sensing," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 3832–3836.

[18] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun.*, Jul. 2017, pp. 1–5.

[19] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, May 2019.

[20] T. J. O. , "Shea, T. Erpek, and T. C. Clancy, "Deep learning based MIMO communications," 2017, *arXiv:1707.07980*. [Online]. Available: https://arxiv.org/abs/1707.07980

[21] T. Wang, C.-K. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep learning for wireless physical layer: Opportunities and challenges," *China Commun.*, vol. 14, no. 11, pp. 92–111, 2017.

[22] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep learning for massive MIMO CSI feedback," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 748–751, Oct. 2018.

[23] S. Takabe, M. Imanishi, T. Wadayama, and K. Hayashi, "Deep learning-aided TPG projected gradient detector for massive overloaded MIMO channels," in *Proc. IEEE Int. Conf. Commun.*, Aug. 2019, pp. 1–9.

[24] D. Divsalar, M. K. Simon, and D. Raphaeli, "Improved parallel interference cancellation for CDMA," *IEEE Trans. Commun.*, vol. 46, no. 2, pp. 258–268, Feb. 1998.

[25] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980s

[26] J. Ma and L. Ping, "Orthogonal AMP," *IEEE Access*, vol. 5, pp. 2020–2033, 2017.

[27] (2014). *PyTorch*. [Online]. Available: https://pytorch.org

[28] C. Jeon, R. Ghods, A. Maleki, and C. Studer, "Optimality of large MIMO detection via approximate message passing," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aug. 2015, pp. 1227–1231.

[29] R. Hayakawa and K. Hayashi, "Reconstruction of complex discrete-valued vector via convex optimization with sparse regularizers," *IEEE Access*, vol. 6, pp. 66499–66512, 2018.

[30] S. Takabe and T. Wadayama, "Complex field-trainable ISTA for linear and nonlinear inverse problems," 2019, *arXiv:1904.07409*. [Online]. Available: https://arxiv.org/abs/1904.07409

[31] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, vol. 49. New York, NY, USA: Springer, 2009, pp. 185–212.

SATOSHI TAKABE (M'17) received the B.Sc., M.Sc., and Ph.D. degrees in multidisciplinary sciences from The University of Tokyo, Japan, in 2012, 2014, and 2017, respectively. He is currently an Assistant Professor with the Department of Computer Science, Nagoya Institute of Technology, Japan. He received the IEEE Information Theory Society Japan Chapter Young Researcher Best Paper Award, in 2018. His research interests include signal processing, information theory, and machine learning. He is a member of the IEICE, ORSJ, and JPS.

MASAYUKI IMANISHI received the B.E. and M.E. degrees from the Nagoya Institute of Technology, Japan, in 2017 and 2019, respectively. His research interests include signal processing and machine learning.

TADASHI WADAYAMA (M'96) was born in Kyoto, Japan, in 1968. He received the B.E., M.E., and D.E. degrees from the Kyoto Institute of Technology, in 1991, 1993, and 1997, respectively. In 1995, he started to work at the Faculty of Computer Science and System Engineering, Okayama Prefectural University, as a Research Associate. From 1999 to 2000, he stayed at the Institute of Experimental Mathematics, Essen University, Germany, as a Visiting Researcher. In 2004, he moved to the Nagoya Institute of Technology as an Associate Professor. Since 2010, he has been a Full Professor with the Nagoya Institute of Technology. His research interests include coding theory, information theory, and coding and signal processing for digital communication/storage systems. He is a member of the IEICE.

RYO HAYAKAWA (S'16) received the bachelor's degree in engineering and the master's degree in informatics from Kyoto University, Kyoto, Japan, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree with the Department of Systems Science, Graduate School of Informatics. His research interests include signal processing and wireless communication. He received the Telecom System Technology Student Award from the Telecommunications Advancement Foundation, in 2018. He is also a Student Member of the IEICE. He is also a Research Fellow (DC1) of the Japan Society for the Promotion of Science (JSPS).

KAZUNORI HAYASHI (S'97–M'03) received the B.E., M.E., and Ph.D. degrees in communication engineering from Osaka University, Osaka, Japan, in 1997, 1999, and 2002, respectively. He was an Assistant Professor, from 2002 to 2007, and an Associate Professor, from 2007 to 2017, with the Graduate School of Informatics, Kyoto University. He is currently a Professor with the Graduate School of Engineering, Osaka City University. His research interests include statistical signal processing for communication systems. He received the ICF Research Award from the KDDI Foundation, in 2008, the IEEE Globecom 2009 Best Paper Award, the IEICE Communications Society Best Paper Award, in 2010, the WPMC11 Best Paper Award, the Telecommunications Advancement Foundation Award, in 2012, and the IEICE Communications Society Best Tutorial Paper Award, in 2013. He is a member of APSIPA and ISCIE.

● ● ●