IEEE Open Journal of Vehicular Technology

Received 15 May 2024; accepted 30 May 2024. Date of publication 6 June 2024;
date of current version 21 June 2024. The review of this article was coordinated by Editor Sumei Sun.

Digital Object Identifier 10.1109/OJVT.2024.3410834

Dynamic Conjugate Gradient Unfolding for Symbol Detection in Time-Varying Massive MIMO

TOLUWALEKE OLUTAYO (Student Member, IEEE) AND BENOIT CHAMPAGNE (Senior Member, IEEE)

Department of Electrical and Computer Engineering, McGill University, Montreal, QC H3A 0G4, Canada

CORRESPONDING AUTHOR: TOLUWALEKE OLUTAYO (e-mail: toluwaleke.olutayo@mail.mcgill.ca).

This work was supported by an Alliance Grant from the Natural Sciences and Engineering Research Council of Canada, with InterDigital Canada as industry sponsor. An earlier version of this article was presented in part at the IEEE LATINCOM, Rio de Janeiro, Brazil, Nov 30–Dec 03, 2022 [DOI: 10.1109/LATINCOM56090.2022.10000560].

ABSTRACT This article addresses the problem of symbol detection in time-varying Massive Multiple-Input Multiple-Output (M-MIMO) systems. While conventional detection techniques either exhibit subpar performance or impose excessive computational burdens in such systems, learning-based methods which have shown great potential in stationary scenarios, struggle to adapt to non-stationary conditions. To address these challenges, we introduce innovative extensions to the Learned Conjugate Gradient Network (LcgNet) M-MIMO detector. Firstly, we expound Preconditioned LcgNet (PrLcgNet), which incorporates a preconditioner during training to enhance the uplink M-MIMO detector's filter matrix. This modification enables the detector to achieve faster convergence with fewer layers compared to the original approach. Secondly, we introduce an adaptation of PrLcgNet referred to as Dynamic Conjugate Gradient Network (DyCoGNet), specifically designed for time-varying environments. DyCoGNet leverages self-supervised learning with Forward Error Correction (FEC), enabling autonomous adaptation without the need for explicit labeled data. It also employs meta-learning, facilitating rapid adaptation to unforeseen channel conditions. Our simulation results demonstrate that in stationary scenarios, PrLcgNet achieves faster convergence than LCgNet, which can be leveraged to reduce system complexity or improve Symbol Error Rate (SER) performance. Furthermore, in non-stationary scenarios, DyCoGNet exhibits rapid and efficient adaptation, achieving significant SER performance gains compared to baseline cases without meta-learning and a recent benchmark using self-supervised learning.

INDEX TERMS Massive MIMO (M-MIMO), symbol detection, conjugate gradient, model-based learning, non-stationary channels, online adaptation, meta-learning.

I. INTRODUCTION

With the ever-growing demand for higher data rates and more reliable wireless communication systems, Massive Multiple-Input Multiple-Output (MIMO) technology has emerged as a promising solution to address the challenges of future wireless networks [2]. This paradigm leverages an extensive array of antennas at the base station to simultaneously serve multiple users, thereby significantly enhancing spectral efficiency and interference mitigation [3]. However, the task of detecting data symbols at the receiver in M-MIMO systems remains highly computationally demanding, particularly when dealing with very large numbers of antennas and users. Conventional algorithms, which exhibit satisfactory performance in small-scale MIMO systems as referenced in [4], [5], [6], prove to be computationally prohibitive for M-MIMO setups. Consequently, there is a critical necessity to develop low-complexity detectors capable of delivering good performance in such systems.

Recent innovations in deep learning have shown remarkable potential in addressing the complexity of M-MIMO detection algorithms [7]. Relying on Deep Neural Networks (DNNs), these learning approaches can be broadly

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

categorized into two types: i) data-based and ii) model-based. Data-based approaches aim to find optimal parameters for a DNN detector by minimizing a cost function over a training data set. For instance, in [8], a Convolutional Neural Network-based Likelihood Ascent Search (CNNLAS) detection algorithm is introduced that integrates a graphical detection model and exhibits robustness against channel estimation errors compared to alternative methods. A more recent approach, outlined in [9], employs a data-driven DNN architecture incorporating reservoir computing and a multi-head self-attention mechanism specifically tailored for dynamic wireless scenarios. Furthermore in [10], a data-driven implementation of the Soft Interference Cancellation (SIC) symbol detection algorithm, named DeepSIC is introduced. DeepSIC leverages the generalization capabilities of DNNs to detect transmitted symbols across both linear and nonlinear channels. In spite of the performance benefits offered by these data-driven DNN approaches, they face several drawbacks, including relatively large training data set requirements and lengthy training times. Moreover, their black-box nature poses challenges in understanding and improving the performance of the trained networks.

In contrast, model-based networks leverage domain knowledge, enabling the reduction of training data and facilitating comprehension of their operation [11]. In the field of wireless communications, an effective framework for model-based learning is provided by algorithm unfolding, commonly known as *deep unfolding* [12]. This technique unfolds the iterations of an inference algorithm by introducing additional trainable parameters to enhance model capacity, and subsequently utilizing gradient-based methods to optimize the network's performance [13]. Recently, deep unfolding has emerged as a popular approach to learning-based M-MIMO symbol detection. In [14], a symbol detector named DetNet is introduced that unfolds the iterations of a projected gradient descent algorithm, demonstrating performance akin to a baseline semi-definite relaxation detector but with significantly reduced runtime. However, DetNet's reliance on very large parameter space (e.g., over 2 million parameters for a 64×32 MIMO system) results in substantial storage and computational costs. In [15], a model-based detector named OAMPNet is introduced which unfolds the Orthogonal Approximate Message Passing (OAMP) algorithm, showcasing improved performance compared to the base OAMP algorithm. However, OAMPNet's computational complexity increases significantly in the case of M-MIMO detection due to the need to perform large matrix inversion in each layer. To reduce the complexity of model-based detector, a so-called Learned Conjugate Gradient Network (LcgNet) is proposed in [16]. LcgNet unfolds the Conjugate Gradient (CG) descent MIMO detector into a layer-wise neural network that can learn universal step size values. Although LcgNet outperforms the CG detector on which it is based, it still requires a substantial number of model parameters, leading to extended training time and increased detection complexity. In [1], an extension of LcgNet termed Preconditioned Learned Conjugate Gradient Network (PrLcgNet) is proposed, whereby a preconditioning scheme is employed to achieve similar or better performance with fewer layers.

While learning-based methods for M-MIMO detection have shown great promise in stationary scenarios, they encounter substantial challenges in non-stationary environments. These methods typically involve training model parameters on dedicated pilot blocks and subsequently applying the trained model to detect data blocks. This demands a model with ample capacity, i.e., capable of generalizing across diverse channel conditions and characterized by a large number of parameters. An alternative strategy involves adapting the detector to time-varying channel conditions, as exemplified by [17], where a model-based detector known as MMNet is introduced. MMNet's architecture draws inspiration from iterative soft-thresholding algorithms and employs a novel training algorithm that exploits temporal and spectral correlations present in slowly-varying MIMO channels to expedite model retraining. However, application of MMNet in rapidly-varying channels poses major challenges due to loss of temporal correlation.

Recent advancements in the area of symbol detection, as evidenced by [18], [19], propose innovative strategies like Forward Error Correction (FEC) and error detection mechanisms to generate training samples for model adaptation. This circumvents the need for transmitting pilots solely for model retraining, thus reducing training overhead and enhancing the feasibility of real-time applications. However, the methods in [18], [19] are not conceived for MIMO detection, but instead to enhance Viterbi and BCJR algorithms, respectively, in SISO systems with memory. Studies such as [20], [21], [22], advocate for the adoption of meta-learning to facilitate the optimization of model parameter initialization, streamlining the process of rapid online adaptation. In particular, [22] introduces a meta-learning scheme utilizing pilots from previous transmissions of IoT devices to train a demodulator for adaptation to new channel conditions. Nevertheless, extension of these schemes to M-MIMO detection under time-varying channel conditions remains largely uncharted.

In this paper, drawing inspiration from these recent developments in online model-based learning, we introduce a novel M-MIMO symbol detector named Dynamic Conjugate Gradient Network (DyCoGNet), specially designed to enhance detection efficiency in time-varying environments. In contrast to earlier approaches such as MMNET, DyCoGNet builds upon PrLcgNet by integrating a novel combination of deep unfolding, FEC-aided self-supervised learning, and meta-learning strategies. Specifically, DyCoGNet unfolds the preconditioned conjugate gradient algorithm by incorporating learnable parameters, facilitating efficient adaptation to rapid channel variations in M-MIMO systems. Furthermore, DyCoGNet adopts a self-supervised training mechanism utilizing FEC to generate additional training data for online model adaptation during the data detection phase. Finally, DyCoGNet employs meta-learning to enhance its adaptability to time-varying channels, by allowing to learn optimal

parameter initialization with minimal retraining. The main contributions of our work are summarized as follows:

- We expound and refine PrLcgNet, an extension of LcgNet featuring a preconditioning technique optimizing the receiver's filter matrix spectrum. This approach effectively minimizes the eigenvalue spread, enhancing the convergence behavior of the iterative algorithm, which in turn can be leveraged to improve Symbol Error Rate (SER) performance or reduce computational complexity.
- We examine the challenges posed by time-varying channels in the design of M-MIMO detectors, and develop a FEC-aided self-supervised learning approach to facilitate detector adaptation. This approach, which exploits reliable detection of transmitted symbols, allows the detector to dynamically adjust model parameters estimated from pilot blocks, during subsequent detection of data blocks, effectively handling outdated information through pilot-free online re-training.
- We investigate the use of meta-learning as a means to expedite the adaptation process through proper initialization of the model parameters. Specifically, this technique is incorporated in our approach to learn optimal parameters for model initialization, thereby enhancing adaptability with minimal retraining as the channel evolves.
- We develop a new model-based M-MIMO detector called DyCoGNet, as an extension of PrLcgNet, tailored for wireless environments with time-varying channels. DyCoGNet relies on the integration of CG deep unfolding, FEC-aided online self-supervised adaptation, and meta-learning to ensure computationally efficient adaptation of model parameters in time-varying environments. We emphasize that DyCoGNet operates without the reliance on perfect channel knowledge, setting it apart from other model-based approaches.
- A detailed complexity analysis of PrLcgNet and DyCoGNet is presented, including comparisons to benchmark approaches. It is seen that both PrLcgNet and DyCoGNet can lead to substantial reduction in computational complexity compared to these approaches in practical applications.
- We conduct extensive simulations to assess the newly proposed detectors under both stationary (spatially correlated Rayleigh fading MIMO channels) and non-stationary (Gauss-Markov and 3GPP TDL-A [23] MIMO channels) conditions. We appraise the efficacy of the preconditioning used in PrLcgNet, showcasing its ability to expedite convergence and reduce SER when compared to LcgNet in stationary environments. We also assess DyCoGNet in non-stationary scenarios with high-user mobility, where it demonstrates significant advantages in terms of SER performance over other baseline detectors.

The rest of this paper is structured as follows. Section II introduces the system model for M-MIMO symbol detection and reviews the CG algorithm. Section III presents a systematic derivation of the PrLcgNet detection network, along with a discussion of training and computational complexity. Section IV introduces DyCoGNet as an extension of PrLcgNet, by integrating the FEC-aided self-learning and meta-learning techniques alongside CG deep unfolding; a complexity analysis of DyCoGNet is also provided. Section V presents the results of numerical simulations over stationary and time-varying channels. Finally, Section VI concludes the work.

Notations - Boldface uppercase letters denote matrices, boldface lowercase letters denote column vectors, and lowercase letters denote scalars. Operators $\Re(\cdot)$ and $\Im(\cdot)$ take the real and imaginary parts of their arguments, respectively. The notations $\mathbf{A}^{-1}$, $\mathbf{A}^H$, and $\mathbf{A}^T$ respectively stand for the inverse, conjugate transpose, and transpose of matrix $\mathbf{A}$ The 2-norm of vector $\mathbf{x}$ is denoted by $\|\mathbf{x}\|$ and the Frobenius norm of matrix $\mathbf{A}$ by $\|\mathbf{A}\|_F$. The complex and real vector spaces of size $m \times n$ are denoted by $\mathbb{C}^{m\times n}$ and $\mathbb{R}^{m\times n}$, respectively. The symbol $\odot$ denotes the Hadamard product, and $\mathbf{I}_n$ represents an identity matrix of size n. For random vectors, $\mathcal{CN}(\mathbf{0}, \mathbf{R})$ and $\mathcal{N}(\mathbf{0}, \mathbf{R})$ respectively denote the complex circular and real Gaussian distributions with zero mean and correlation matrix $\mathbf{R}$, respectively, while $\mathbb{E}[\cdot]$ represents the expectation operator.

II. M-MIMO SYSTEM MODEL AND CG ALGORITHM

In this section, we lay the foundations of our approach by introducing the M-MIMO system model, exposing the challenges of large vector symbol detection, and providing essential background on the CG iterative detector.

A. M-MIMO SYSTEM MODEL FOR SYMBOL DETECTION

We consider a narrowband multiuser M-MIMO communication channel with N_t single antenna transmitters and a receiving base station equiped with N_r antennas. The forward (uplink) baseband signal model for this system is given by:

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{1}$$

where $\mathbf{y} \in \mathbb{C}^{N_r}$ is the received signal vector, $\mathbf{H} \in \mathbb{C}^{N_r \times N_t}$ is the channel matrix, $\mathbf{s} \in \mathcal{S}^{N_t}$ is the vector of transmitted symbols, $\mathcal{S} \subset \mathbb{C}$ is a finite symbol constellation, and $\mathbf{n} \in \mathbb{C}^{N_r}$ is an additive white Gaussian noise vector with element variance σ_n^2, i.e., $\mathbf{n} \sim \mathcal{CN}(\mathbf{0}, \sigma_n^2 \mathbf{I}_{N_r})$. We assume that the constellation symbols in $\mathcal{S}$ are normalized to yield unit variance and let $\gamma \triangleq \mathbb{E}[\|\mathbf{H}\mathbf{s}\|^2]$ denote the total received signal power.

The goal of MIMO symbol detection is to recover the unknown transmitted vector $\mathbf{s}$ from the received signal vector $\mathbf{y}$ in (1). The Bayes-optimal detector is the Maximum A Posteriori (MAP) detector which outputs as estimate, the symbol that maximizes the posterior probability $p(\mathbf{s}|\mathbf{y})$, i.e.:

$$\hat{\mathbf{s}}_{\text{MAP}} = \arg\max_{\mathbf{s}\in\mathcal{S}^{N_t}} p(\mathbf{s}|\mathbf{y}). \tag{2}$$

Assuming a uniform input prior distribution and knowledge of the channel matrix $\mathbf{H}$ at the receiver,[1] the MAP estimate

[1]Several efficient schemes are available for the estimation of the channel matrix $\mathbf{H}$ in MIMO systems, see e.g., [24].

reduces to a Maximum Likelihood (ML) estimate given by:

$$\hat{\mathbf{s}}_{\mathrm{ML}} = \arg\max_{\mathbf{s}\in\mathcal{S}^{N_t}} p(\mathbf{y}|\mathbf{s}) = \arg\max_{\mathbf{s}\in\mathcal{S}^{N_t}} p_{\mathbf{n}}(\mathbf{y}-\mathbf{H}\mathbf{s},) \tag{3}$$

where $p_{\mathbf{n}}(\cdot)$ is the noise probability density function. For white Gaussian noise, the ML estimation reduces to a constrained discrete Euclidean distance minimization problem:

$$\hat{\mathbf{s}}_{\mathrm{ML}} = \arg\min_{\mathbf{s}\in\mathcal{S}^{N_t}} \|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2. \tag{4}$$

However, this optimal ML detection is NP-hard due to the finite constellation constraint, making it intractable for M-MIMO applications with large dimensionality.

In past years, several sub-optimal detectors have been proposed to address this challenge [6]. In M-MIMO systems, simple linear detection algorithms, such as the Zero-Forcing (ZF) and Linear Minimum Mean Square Error (LMMSE) detectors, can achieve good performance, particularly due to the channel hardening phenomenon [25]. The main approach employed by linear detectors is to filter the received signal $\mathbf{y}$ to obtain a soft estimate $\hat{\mathbf{s}}$ of the transmitted symbol vector, and then map the elements of $\hat{\mathbf{s}}$ to $\mathcal{S}$ using the minimum distance criterion. Specifically, in the case of the LMMSE detector, the filtering operation can be expressed as follows:

$$\hat{\mathbf{s}} = \left(\mathbf{H}^H\mathbf{H} + \sigma_n^2\mathbf{I}_{N_r}\right)^{-1}\mathbf{H}^H\mathbf{y}. \tag{5}$$

However, the matrix inverse in (5) can incur excessive computational complexity for large-dimensional matrices prevalent in M-MIMO scenarios.

B. CONJUGATE GRADIENT DESCENT ALGORITHM

The CG detector is an iterative algorithm that approaches the performance of the LMMSE detector without the need for complex matrix inversion. To facilitate the description of the CG algorithm and the learning-based algorithms discussed in subsequent sections, we reformulate the detection problem in terms of real-valued quantities, defined as [26]:

$$\mathbf{y}_r = \begin{bmatrix}\Re(\mathbf{y})\\ \Im(\mathbf{y})\end{bmatrix} \in \mathbb{R}^{2N_r},\quad \mathbf{s}_r = \begin{bmatrix}\Re(\mathbf{s})\\ \Im(\mathbf{s})\end{bmatrix} \in \mathbb{R}^{2N_t}, \tag{6}$$

$$\mathbf{n}_r = \begin{bmatrix}\Re(\mathbf{n})\\ \Im(\mathbf{n})\end{bmatrix} \in \mathbb{R}^{2N_r}, \tag{7}$$

$$\mathbf{H}_r = \begin{bmatrix}\Re(\mathbf{H}) & -\Im(\mathbf{H})\\ \Im(\mathbf{H}) & \Re(\mathbf{H})\end{bmatrix} \in \mathbb{R}^{2N_r\times 2N_t}. \tag{8}$$

In terms of these variabes, the M-MIMO system model in (1) can be equivalently expressed as:

$$\mathbf{y}_r = \mathbf{H}_r\mathbf{s}_r + \mathbf{n}_r, \tag{9}$$

while the LMMSE soft receiver output in (5) becomes:

$$\hat{\mathbf{s}}_r = \left(\mathbf{H}_r^T\mathbf{H}_r + \frac{\sigma_n^2}{2}\mathbf{I}\right)^{-1}\mathbf{H}_r^T\mathbf{y}_r = \mathbf{A}^{-1}\mathbf{b}, \tag{10}$$

where matrix $\mathbf{A} \triangleq \mathbf{H}_r^T\mathbf{H}_r + \frac{\sigma_n^2}{2}\mathbf{I}$ is symmetric positive definite and vector $\mathbf{b} \triangleq \mathbf{H}_r^T\mathbf{y}_r$.

The LMMSE detector in (10) can be reformulated as the solution to a standard quadratic optimization problem, i.e.:

$$\mathcal{K}(\mathbf{s}_r;\mathbf{A},\mathbf{b}) \triangleq \frac{1}{2}\mathbf{s}_r^T\mathbf{A}\mathbf{s}_r - \mathbf{b}^T\mathbf{s}_r. \tag{11}$$

$$\hat{\mathbf{s}}_r = \arg\min_{\mathbf{s}_r} \mathcal{K}(\mathbf{s}_r;\mathbf{A},\mathbf{b}). \tag{12}$$

The CG algorithm is a highly efficient optimization technique that effectively explores the curvature of the objective function in (11). By aligning the search directions with the eigenvectors of the Hessian matrix $\mathbf{A}$, the algorithm efficiently corrects the unaccounted component of the previous search direction in the iterative solution computation process [27]. This property makes it a powerful tool for solving quadratic optimization problems. Specifically, the CG algorithm generates a set of $L \le 2N_t$ search directions $\mathbf{d}_i \in \mathbb{R}^{2N_t}$, $i \in \{0,\ldots,L-1\}$, that are $\mathbf{A}$-orthogonal to each other, i.e., $\mathbf{d}_i^T\mathbf{A}\mathbf{d}_j = 0,\ \forall i \neq j$. At each iteration, the algorithm updates the current estimate of the desired solution $\hat{\mathbf{s}}_r$ and calculates a new conjugate direction using a residual vector (or steepest descent direction) denoted as $\mathbf{r}_i$. The algorithm starts by initializing $\hat{\mathbf{s}}_r$ to a suitable value, $\mathbf{s}_{r,0}$, which can be chosen as $\mathbf{0}$ in the absence of prior knowledge. The first conjugate direction $\mathbf{d}_0$ and residual vector $\mathbf{r}_0$ are then initialized as follows:

$$\mathbf{d}_0 = \mathbf{r}_0 \triangleq \mathbf{b} - \mathbf{A}\mathbf{s}_{r,0} \tag{13}$$

The update equations at the i-th iteration ($i = 0, 1, \ldots, L-1$) are given by:

$$\alpha_i = \frac{\mathbf{r}_i^T\mathbf{r}_i}{\mathbf{d}_i^T\mathbf{A}\mathbf{d}_i}, \tag{14}$$

$$\mathbf{s}_{r,i+1} = \mathbf{s}_{r,i} + \alpha_i\mathbf{d}_i, \tag{15}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha_i\mathbf{A}\mathbf{d}_i, \tag{16}$$

$$\beta_i = \frac{\mathbf{r}_{i+1}^T\mathbf{r}_{i+1}}{\mathbf{r}_i^T\mathbf{r}_i}, \tag{17}$$

$$\mathbf{d}_{i+1} = \mathbf{r}_{i+1} + \beta_i\mathbf{d}_i. \tag{18}$$

The CG algorithm possesses the desirable property of guaranteed convergence, ensuring that the optimal solution to (12) is found within a maximum of $2N_t$ iterations [27].

III. PRLCGNET DETECTION NETWORK DEVELOPMENT

We begin this section by providing an overview of LcgNet in its original form and motivating the need for improvements. Subsequently, we present the step-by-step integration of a novel preconditioner to create PrLcgNet. We also elaborate on the training process, explaining how the network's parameters are optimized. Finally, we conduct a comprehensive complexity analysis to compare the computational requirements of PrLcgNet with LcgNet.

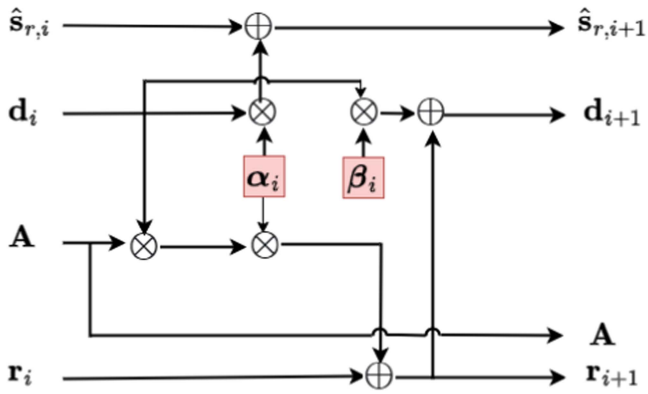


FIGURE 1. LcgNet *i*-th layer.

A. LCGNET

The CG detector's complexity results from the multiple matrix-vector multiplications and divisions needed to calculate the scalar step-sizes α_i and β_i in (14)–(18). To reduce detection complexity, the deep learning network proposed in [16], termed LcgNet, offers an efficient alternative. In LcgNet, the step-sizes are learned offline over a training set of transmitted and received symbols, making them parameters of the network and hence eliminating the need for their explicit calculations during detection. The LcgNet architecture is based on the flow graph illustrated in Fig. 1, where each CG algorithm iteration corresponds to a layer in LcgNet.

Two variants of LcgNet exist: LcgNetS and LcgNetV. In LcgNetS, the learnable parameters of the network at layer i are scalars, represented by $\boldsymbol{\theta}^{(i)} = \{\alpha_i, \beta_i\}$, while in LcgNetV, these parameters are extended to vectors of size $2N_t$, i.e., $\boldsymbol{\theta}^{(i)} = \{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i\}$. LcgNetV demonstrates superior performance compared to LcgNetS across various channel models and SNR levels [16]. Consequently, our analysis exclusively focuses on the performance enhancement of LcgNetV. The computations performed by the i^{th} layer of LcgNetV are described by the following equations:

$$\mathbf{s}_{r,i+1} = \mathbf{s}_{r,i} + \boldsymbol{\alpha}_i \odot \mathbf{d}_i \tag{19}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \boldsymbol{\alpha}_i \odot (\mathbf{A}\mathbf{d}_i) \tag{20}$$

$$\mathbf{d}_{i+1} = \mathbf{r}_{i+1} + \boldsymbol{\beta}_i \odot \mathbf{d}_i \tag{21}$$

These recursions are employed for signal detection as well as for training the model using backpropagation.

Although LcgNetV outperforms the CG detector on which it is based, it still requires a large number of model parameters, leading to extended training time and detection complexity.

B. THE PROPOSED PRLCGNET

Preconditioning is a powerful technique employed to enhance the condition number of a matrix, leading to more efficient and stable iterative solutions in a variety of problems [28]. Consider a symmetric, positive semi-definite matrix $\mathbf{M}$ that serves as an approximation to another matrix $\mathbf{A}$, with the advantage of being easier to invert. The solution to the linear equation $\mathbf{A}\mathbf{s}_r = \mathbf{b}$ can be obtained indirectly and more efficiently by solving the preconditioned equation:

$$\left(\mathbf{M}^{-1}\mathbf{A}\right)\mathbf{s}_r = \mathbf{M}^{-1}\mathbf{b}. \tag{22}$$

If $\kappa(\mathbf{M}^{-1}\mathbf{A}) \ll \kappa(\mathbf{A})$, where $\kappa(\cdot)$ denotes the condition number of its matrix argument, then iterative methods like the CG algorithm can rapidly find the solution to (22) compared to the original problem. However, directly applying the CG algorithm to obtain the solution to the preconditioned equation in (22) is not straightforward as $\mathbf{M}^{-1}\mathbf{A}$ is not necessarily symmetric positive semi-definite. To address this problem, the Cholesky decomposition of the preconditioning matrix, $\mathbf{M} = \mathbf{E}\mathbf{E}^T$, is used to transform $\mathbf{A}\mathbf{s}_r = \mathbf{b}$ into the following symmetric positive semi-definite form:

$$\left(\mathbf{E}^{-1}\mathbf{A}\mathbf{E}^{-T}\right)\bar{\mathbf{s}}_r = \mathbf{E}^{-1}\mathbf{b}. \tag{23}$$

where $\bar{\mathbf{s}}_r \triangleq \mathbf{E}^T\mathbf{s}_r$. The transformed matrix $\mathbf{E}^{-1}\mathbf{A}\mathbf{E}^{-T}$ has the same eigenvalues as $\mathbf{M}^{-1}\mathbf{A}$, but since it is symmetric, the CG algorithm can be readily applied to find the solution to (23). In practice, the Cholesky decomposition $\mathbf{M} = \mathbf{E}\mathbf{E}^T$ needs not be explicitly computed (see [29], Chap. 12). In effect, the update equations to solve problem (23) iteratively can be expressed in terms of the untransformed variable $\mathbf{s}_r$. Specifically, at the i-th iteration ($i = 0, \ldots, L-1$), we have:

$$\alpha_i = \frac{\mathbf{r}_i^T\mathbf{M}^{-1}\mathbf{r}_i}{\mathbf{d}_i^T\mathbf{A}\mathbf{d}_i}, \tag{24}$$

$$\mathbf{s}_{r,i+1} = \mathbf{s}_{r,i} + \alpha_i\mathbf{d}_i, \tag{25}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha_i\mathbf{A}\mathbf{d}_i, \tag{26}$$

$$\beta_i = \frac{\mathbf{r}_{i+1}^T\mathbf{M}^{-1}\mathbf{r}_{i+1}}{\mathbf{r}_i^T\mathbf{M}^{-1}\mathbf{r}_i}, \tag{27}$$

$$\mathbf{d}_{i+1} = \mathbf{M}^{-1}\mathbf{r}_{i+1} + \beta_i\mathbf{d}_i, \tag{28}$$

where $\mathbf{s}_{r,0} = \mathbf{0}$, $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{s}_{r,0}$, and $\mathbf{d}_0 = \mathbf{M}^{-1}\mathbf{r}_0$.

We now introduce PrLcgNet. This network leverages the unfolding of the preconditioned CG algorithm by incorporating the learning of augmented step-sizes in a manner similar to LcgNetV, but with the integration of the preconditioner matrix $\mathbf{M}$ in the updating of the search direction $\mathbf{d}_i$ at each layer. The PrLcgNet algorithm for M-MIMO detection is shown in Algorithm 1. It uses as main inputs the matrix $\mathbf{A}$ and vector $\mathbf{b}$ from (10), along with a pre-computed inverse preconditioner matrix $\mathbf{M}^{-1}$, and outputs the desired estimate $\mathbf{s}_r$ of the transmitted symbol vector.

We note that implementing PrLcgNet as described necessitates the computation of the matrix multiplication $\mathbf{M}^{-1}\mathbf{r}_{i+1}$ at each layer or iteration. This matrix multiplication can potentially result in substantial computational costs, contingent on the choice of the preconditioner $\mathbf{M}$. The simplest preconditioner involves a diagonal matrix with diagonal entries identical to those of $\mathbf{A}$ i.e., $\mathbf{M}^{-1} = \mathbf{D}^{-1}$ where $\mathbf{D} = \text{diag}(\mathbf{A})$. This approach is known as *diagonal preconditioning* or *Jacobi preconditioning*. More sophisticated preconditioners

Algorithm 1: PrLcgNet M-MIMO Detector.

Input : Matrix $\mathbf{A}$ and vector $\mathbf{b}$
Inverse preconditioner matrix $\mathbf{M}^{-1}$
Initial step-sizes $\boldsymbol{\alpha}_0$ and $\boldsymbol{\beta}_0$
Output: Solution vector $\hat{\mathbf{s}}_r$

Initialize $\mathbf{s}_{r,0} = \mathbf{0}$, $\mathbf{r}_0 = \mathbf{b}$, $\mathbf{d}_0 = \mathbf{M}^{-1}\mathbf{r}_0$
for $l = 0, 1, \ldots, L-1$ **do**
  $\mathbf{s}_{r,l+1} = \mathbf{s}_{r,l} + \boldsymbol{\alpha}_l \odot \mathbf{d}_l$
  $\mathbf{r}_{l+1} = \mathbf{r}_l - \boldsymbol{\alpha}_i \odot (\mathbf{A}\mathbf{d}_l)$
  $\mathbf{d}_{l+1} = \mathbf{M}^{-1}\mathbf{r}_{l+1} + \boldsymbol{\beta}_l \odot \mathbf{d}_l$
end
return $\hat{\mathbf{s}}_r = \mathbf{s}_{r,L}$

have been proposed in the literature, including the Gauss-Seidel, and Symmetric Successive Over-Relaxation (SSOR) methods, among others. [30], [31], [32].

PrLcgNet, by integrating a preconditioner within its network, can substantially enhance algorithm convergence. The preconditioner improves the network's ability to learn optimal parameters, resulting in lower residual error during training and decreased online detection complexity. Notably, PrLcgNet achieves comparable performance to LcgNet with fewer layers, as will be shown in Section V.

C. TRAINING

During this phase, the network is provided with training data consisting of inputs and labels, and its parameters are adjusted to provide the bet fit between the model outputs and the labels. Consider a dataset of M training samples, $\mathcal{D} = \{\mathbf{y}_r^{(m)}, \mathbf{s}_r^{(m)}\}_{m=1}^{M}$, where $\mathbf{y}_r^{(m)}$ denotes the received signal vector corresponding to transmitted symbol vector $\mathbf{s}_r^{(m)}$. Let $\boldsymbol{\theta} \triangleq \{\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(L)}\}$ denote the complete set of learnable parameters of the PrLcgNet detector, where L is the number of layers; let $\Phi(\,\cdot\,;\boldsymbol{\theta}) : \mathbb{C}^{N_r} \to \mathcal{S}^{N_t}$ represent the receiver mapping dictated by these parameters; and let $\hat{\mathbf{s}}_r^{(m)}(\boldsymbol{\theta}) \triangleq \Phi(\mathbf{y}_r^{(m)};\boldsymbol{\theta})$ denote the output of PrLcgNet in response to $\mathbf{y}_r^{(m)}$. During the training process, the received signal vectors $\mathbf{y}_r^{(m)}$ serve as input to the network, while the symbol vectors $\mathbf{s}_r^{(m)}$ are used as labels. The optimal parameters of the network are obtained by minimizing a loss function, defined as the mean square error between the network outputs and the labels:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{M}\sum_{m=1}^{M} \left\| \hat{\mathbf{s}}_r^{(m)}(\boldsymbol{\theta}) - \mathbf{s}_r^{(m)} \right\|^2. \tag{29}$$

Following a similar approach to [16], the training dataset $\mathcal{D}$ is generated by randomly selecting transmit symbols from the symbol constellation to form the vectors $\mathbf{s}^{(m)}$. Subsequently, channel matrices $\mathbf{H}^{(m)}$ and additive noise vectors $\mathbf{n}^{(m)}$ are obtained by sampling appropriate statistical distributions, and the received signal vectors $\mathbf{y}^{(m)}$ are obtained by applying the noisy M-MIMO channel model (1). Subsequently, backpropagation based on gradient descent is employed to obtain the optimal network parameters minimizing (28), as further discussed in Section V.

D. COMPLEXITY ANALYSIS OF PRLCGNET

Table 1 presents a comparison of the complexity between the CG detector and the trained PrLcgNet and LcgNet. The numbes of layers/iterations for the CG detector, LcgNet, and PrLcgNet are denoted as L_0, L_1, and L_2, respectively. Firstly, for an equal number of layers/iterations,($L_0 = L_1 = L_2$) the CG detector exhibits a higher complexity due to the matrix-vector multiplication entering the computation of α_i. In contrast, both LcgNet and PrLcgNet are designed in a manner that obviates this requirement, since $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are already learned parameters integrated into the network architecture. We note that our work predominantly utilizes Jacobi preconditioning. Consequently, when we compare LcgNet and PrLcgNet under the same number of layers/iterations, they exhibit equivalent complexities. As shown in Section V, opting for $L_2 < L_1$ offers a good performance-complexity trade-off, since PrLcgNet can still achieve similar or better performance as LcgNet in terms of average training Normalized Mean Square Error (NMSE) and SER. The implication is a notable reduction in detection complexity making PrLcgNet an efficient alternative while maintaining competitive performance.

IV. DYCOGNET: EXTENDING PRLCGNET TO DYNAMIC WIRELESS ENVIRONMENTS

In this section, we introduce DyCoGNet, an extension of the PrLcgNet detector specifically designed to overcome the challenges posed by non-stationary radio environments. Distinctively, DyCoGNet leverages the principles of FEC-aided self-supervised learning alongside the integration of meta-learning, to achieve rapid and effective adaptation of model parameters. We first present the time-varying M-MIMO block-fading channel model of interest, laying the foundation for understanding the challenges of learning-based symbol detection in such environments. Next, we delve into online training using self-supervised learning and explain how it can be leveraged to achieve autonomous adaptation using training data generated from the FEC process. We then unwraps the inner workings of meta-learning and conceive a novel approach for enabling rapid adjustment of the network model by learning optimal parameter initializations. Lastly, we summarize the resulting DyCoGNet algorithm and present an analysis of its computational complexity.

A. BLOCK-FADING CHANNEL MODEL AND PROBLEM FORMULATION

We now extend the system model (1) and consider a more general time-varying block-fading M-MIMO channel. The channel's input-output relation within a normalized coherence interval of integer duration T, referred to as a block,[2] can be

[2]Formally, we define $T \triangleq \lfloor T_C/T_S \rfloor$, where T_C and T_S respectively denote the coherence time and symbol duration (in seconds), and $\lfloor\cdot\rfloor$ denotes the floor function.

TABLE 1. Complexity of the CG Based Detectors

Variables	CG Detector [16]	LcgNet [16]	PrLcgNet
$\boldsymbol{\alpha}_i$	$4N_t^2 + \mathcal{O}(N_t)$	N/A	N/A
$\boldsymbol{\beta}_i$	$\mathcal{O}(N_t)$	N/A	N/A
$\mathbf{s}_{r,i}$	$\mathcal{O}(N_t)$	$\mathcal{O}(N_t)$	$\mathcal{O}(N_t)$
$\mathbf{r}_i$	$4N_t^2 + \mathcal{O}(N_t)$	$4N_t^2 + \mathcal{O}(N_t)$	$4N_t^2 + \mathcal{O}(N_t)$
$\mathbf{d}_i$	$\mathcal{O}(N_t)$	$\mathcal{O}(N_t)$	$\mathcal{O}(N_t)$
	$L_0(8N_t^2 + \mathcal{O}(N_t))$	$L_1(4N_t^2 + \mathcal{O}(N_t))$	$L_2(4N_t^2 + \mathcal{O}(N_t))$

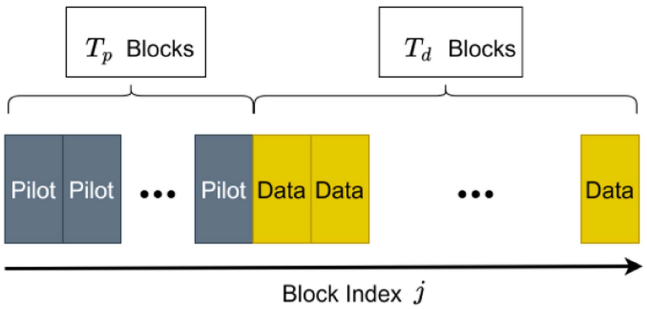


FIGURE 2. Allocation of pilot and data blocks over consecutive coherence intervals for block-fading channel model.

compactly written in matrix notation as follows [33]:

$$\mathbf{Y}_j = \mathbf{H}_j \mathbf{S}_j + \mathbf{N}_j \tag{30}$$

where $j \in \mathbb{N}_+$ is the block index, $\mathbf{Y}_j = [\mathbf{y}_{j1}, \ldots, \mathbf{y}_{jT}] \in \mathbb{C}^{N_r \times T}$ contains the signal vectors received at the N_r BS antennas, $\mathbf{H}_j \in \mathbb{C}^{N_r \times N_t}$ is the channel propagation matrix, $\mathbf{S}_j = [\mathbf{s}_{j1}, \ldots, \mathbf{s}_{jT}] \in \mathcal{S}^{N_t \times T}$ contains the symbol vectors transmitted from the N_t user antennas, and $\mathbf{N}_j \in \mathbb{C}^{N_r \times T}$ is an additive noise term.

The system model presented in (30) describes a channel with block-wise variations, i.e., changes between successive values of the block index j, which can be attributed to the dynamical nature of typical wireless communication environments. For the purpose of applying learning-based methods to symbol detection over time-varying channels, we proceed as in [34] and consider the scenario illustrated in Fig. 2, where a total of T_p pilot blocks and T_d data blocks are transmitted sequentially. The blocks indexed $j \in \mathcal{T}_p \triangleq \{1, \ldots, T_p\}$ represent pilot signals known at the receiver and employed for training the DNN-based symbol detector, while blocks indexed $j \in \mathcal{T}_d \triangleq \{T_p + 1, \ldots, T_p + T_d\}$ represent unknown data blocks. The data blocks are encoded using FEC, enabling the receiver to detect and correct errors in the decoding phase.

Learning-based symbol detectors such as PrLcgNet typically assume that the training and testing data are obtained based on channel realizations randomly drawn from a fixed, consistent distribution. Nevertheless, owing to the dynamic nature of radio channel distributions, deploying this type of detector in real-world scenarios encounters several challenges. Indeed, to ensure model generality, the training must encompass a wide range of channel distributions, necessitating in turn a large number of model parameters to effectively capture this variability. Furthermore, the performance of the trained detector may degrade significantly in the face of radio conditions not previously encountered or well represented by the training data. Our primary objective, therefore, is to conceive an "adaptive" training algorithm that allows PrLcgNet to accurately recover the transmitted data blocks $\{\mathbf{S}_j\}_{j\in\mathcal{T}_d}$ from the received signal blocks $\{\mathbf{Y}_j\}_{j\in\mathcal{T}_d}$ over a time-varying M-MIMO channel, by expanding the available joint knowledge of the transmitted and received pilot blocks, i.e., $\{\mathbf{S}_j, \mathbf{Y}_j\}_{j\in\mathcal{T}_p}$.

Specifically, we would like to design an improved training mechanism whereby the learnable parameters of PrLcgNet are optimally adjusted on-line to achieve a low estimation error for each data block, taking into account underlying changes in the M-MIMO channel. As in Section III-C, let $\boldsymbol{\theta} = \{\boldsymbol{\alpha}_l, \boldsymbol{\beta}_l\}_{l=1}^{L}$ denote the learnable parameters of the PrLcgNet detector and let $\boldsymbol{\Phi}(\cdot\,; \boldsymbol{\theta})$ represent the corresponding receiver mapping. Also let $\hat{\mathbf{s}}_{ji}(\boldsymbol{\theta}) \triangleq \boldsymbol{\Phi}(\mathbf{y}_{ji}; \boldsymbol{\theta})$ denote the detector output corresponding to received signal vector $\mathbf{y}_{ji}$ and let $\hat{\mathbf{S}}_j(\boldsymbol{\theta}) = [\hat{\mathbf{s}}_{j1}(\boldsymbol{\theta}), \ldots, \hat{\mathbf{s}}_{jT}(\boldsymbol{\theta})]$. Ideally, the optimal parameters $\boldsymbol{\theta}_j$ for the j-th block would be obtained as:

$$\boldsymbol{\theta}_j = \arg\min_{\boldsymbol{\theta}} \frac{1}{T} \left\| \hat{\mathbf{S}}_j(\boldsymbol{\theta}) - \mathbf{S}_j \right\|_F^2, \quad j \in \mathcal{T}_d. \tag{31}$$

However, the transmitted symbol blocks $\mathbf{S}_j$ are a priori unknown at the receiver and it is therefore not possible to carry out the above optimization. Adapting PrLcgNet parameters to maintain low SER in the face of changing channels therefore poses a significant challenge.

B. SELF-SUPERVISED LEARNING FOR AUTONOMOUS ADAPTATION

Inspired by the approach in [18], we leverage FEC to implement self-supervised learning. In this approach, error correction is employed at the receiver to generate additional training data during the detection phase, following the initial training phase. The use of pilot-free online training enables the receiver to dynamically adjust its model parameters and track channel variations, thereby handling outdated information.

The operation of our proposed self-supervised learning approach relies on the following block dependent loss function:

$$\mathcal{L}_j(\boldsymbol{\theta}) \triangleq \frac{1}{T} \| \hat{\mathbf{S}}_j(\boldsymbol{\theta}) - \bar{\mathbf{S}}_j \|_F^2, \quad j \in \mathcal{T}_p \cup \mathcal{T}_d \tag{32}$$

where $\bar{\mathbf{S}}_j$ represents the label matrix at the j-th block. This loss function applies during both the training phase using pilot blocks and the detection phase using data blocks, with different choices of label matrix as will be explained shortly. We let $\boldsymbol{\theta}_j$ denote the model parameter values optimizing the above loss function, i.e.:

$$\boldsymbol{\theta}_j = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_j(\boldsymbol{\theta}) \tag{33}$$

An approximate solution to (33) can be obtained via gradient-based optimization with a designated learning rate denoted as $\mu > 0$, yielding the iterative update equation:

$$\boldsymbol{\theta}_j^{(t)} = \boldsymbol{\theta}_j^{(t-1)} - \mu \nabla_{\boldsymbol{\theta}} \mathcal{L}_j \left(\boldsymbol{\theta}_j^{(t-1)} \right). \tag{34}$$

Here $t = 1, \ldots, I_\theta$ represents the iteration index, while the total number of iterations is designated as I_θ.

During pilot transmission (i.e., for $j \in \mathcal{T}_p$), we set $\bar{\mathbf{S}}_j = \mathbf{S}_j$ to adjust the detector parameters. This is possible since $\mathbf{S}_j$ is known at the receiver during the training phase. The (near) optimal parameter values $\boldsymbol{\theta}_j$ obtained after I_θ iterations of the gradient-based update (34), will be used as initial condition for the next block's iterations.

During data transmission (i.e., for $j \in \mathcal{T}_d$), the PrLcgNet receiver utilizes the currently received signal block $\mathbf{Y}_j$ and the model parameters from the previous block $\boldsymbol{\theta}_{j-1}$ to obtain an estimate of transmitted symbol vectors for the current block, denoted as $\hat{\mathbf{S}}_j(\boldsymbol{\theta}_{j-1})$. This estimate is then passed through a FEC decoder to recover the original uncoded message bits from the transmitter side, represented by $\mathcal{W}_j$. We assume that the decoder provides a confidence score about the accuracy of its error detection and correction operation.[3] Depending on this confidence level (i.e., whether or not it exceeds a preset threshold), one of two possibilities arises:

- If the receiver is confident in the decoded message $\mathcal{W}_j$, the later is re-encoded to form the transmitted data block $\mathbf{S}_j$. This re-encoded data is then used along with the received signal block $\mathbf{Y}_j$ as the training data to update the model parameters in the current block. That is, we set $\bar{\mathbf{S}}_j = \mathbf{S}_j$ as the label matrix in (32) and then carry on the optimization of $\mathcal{L}_j(\boldsymbol{\theta})$ as per (33)–(34).
- If the receiver is not confident, the model parameters remain unchanged in the current block, i.e., $\boldsymbol{\theta}_j = \boldsymbol{\theta}_{j-1}$

Performing gradient steps as outlined in (34) requires an initial value $\boldsymbol{\eta}_j \triangleq \boldsymbol{\theta}_j^{(0)}$, a hyperparameter intrinsically tied to the training process. Until now, and as posited in [18], it has been assumed that $\boldsymbol{\eta}_j = \boldsymbol{\theta}_{j-1}$, based on the assumption of a relatively smooth variation of the channel across blocks, implying that a detector trained on data from the $(j-1)$-th block is reasonably effective in detecting data from the j-th block. However, this presumption falls short when grappling with rapidly changing channels, where substantial shifts in the channel's characteristics can occur from one block to the next. Herein, we use meta-learning [35] to tackle this issue in the determination of $\boldsymbol{\eta}_j$, as explained below.

C. META-LEARNING FOR FAST ADAPTATION

Inspired by the framework proposed in [34], we introduce a meta-learning strategy aimed at accelerating the adaptation process of the detector under rapidly varying channel conditions. Our approach utilizes long-term channel variation trends, represented by pairs $(\mathbf{Y}_j, \bar{\mathbf{S}}_j)$ dynamically stored in a fixed-size heap-based buffer, to derive optimal initializations $\boldsymbol{\eta}_j$, enabling quicker adaptation. Specifically, the content of the buffer at the j-th frame, denoted by $\mathcal{B}_j$, is updated on a first-in-first-out basis as follows:

- If $j \in \mathcal{T}_p$, or if $j \in \mathcal{T}_d$ and the detection is successful, the pair $(\mathbf{Y}_j, \bar{\mathbf{S}}_j)$ is inserted as a new element of the buffer, as represented by $\mathcal{B}_j = \mathcal{B}_{j-1} \sqcup (\mathbf{Y}_j, \bar{\mathbf{S}}_j)$.
- If $j \in \mathcal{T}_d$ but the detection is not successful, the content of the buffer is not modified, i.e., $\mathcal{B}_j = \mathcal{B}_{j-1}$.

Following the principles of support and query tasks as outlined in [35], meta-learning involves the selection of a subset of training data from $\mathcal{B}_j$, consisting of all pairs $(\mathbf{Y}_k, \bar{\mathbf{S}}_k)$, such that $(\mathbf{Y}_{k-1}, \bar{\mathbf{S}}_{k-1})$ also belongs to $\mathcal{B}_j$. For convenience, we let $\Lambda_j = \{k \mid (\mathbf{Y}_{k-1}, \bar{\mathbf{S}}_{k-1}) \text{ and } (\mathbf{Y}_k, \bar{\mathbf{S}}_k) \in \mathcal{B}_j\}$ denote the corresponding set of indices. In this setup, the support task involves estimating a transmitted symbol block $\mathbf{S}_{k-1}$ for every $k \in \Lambda_j$, while the query task focuses on the estimation of the subsequent block $\mathbf{S}_k$.

To explain how meta-learning can generate an initial value $\boldsymbol{\eta}_j = \boldsymbol{\theta}_j^{(0)}$ for (34), let us temporarily denote the learnable model parameters[4] by $\boldsymbol{\eta}$. In the support task, the symbol estimate from the PrLcgNet output $\hat{\mathbf{S}}_{k-1}(\boldsymbol{\eta})$ is used to compute the loss $\mathcal{L}_{k-1}(\boldsymbol{\eta})$ using (32). A single gradient descent step yields:

$$\boldsymbol{\chi}_k(\boldsymbol{\eta}) \triangleq \boldsymbol{\eta} - \delta \nabla_{\boldsymbol{\eta}} \mathcal{L}_{k-1}(\boldsymbol{\eta}), \tag{35}$$

where $\delta > 0$ denotes the step-size. These updated model parameters are then used for obtaining $\hat{\mathbf{S}}_k(\boldsymbol{\chi}_k(\boldsymbol{\eta}))$ in the query task. This process is repeated for all block indices $k \in \Lambda_j$. The primary objective is to minimize the meta-learning loss function which is defined as follows:

$$\tilde{\mathcal{L}}_j(\boldsymbol{\eta}) \triangleq \sum_{k \in \Lambda_j} \mathcal{L}_k(\boldsymbol{\chi}_k(\boldsymbol{\eta})). \tag{36}$$

Finally, $\boldsymbol{\eta}_j$ is obtained as the model parameters optimizing the above loss function, i.e.:

$$\boldsymbol{\eta}_j = \arg\min_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_j(\boldsymbol{\eta}). \tag{37}$$

Similar to Section IV-B, the solution to (37) is approximated through gradient-based optimization with a specified learning

[3] For instance, in the case of Reed-Solomon codes employed in this work, such a measure of the confidence is provided by the normalized Hamming distance between the re-encoded bits and those obtained from the hard decision output of the detector, following a similar approach as in [18], [34].

[4] Both $\boldsymbol{\theta}$ and $\boldsymbol{\eta}$ represent possible values of the model parameters, defined over the search space $\mathbb{R}^{4N_tL}$. We use the two variables to better differentiate between the search processes used for symbol detection and for meta-learning.

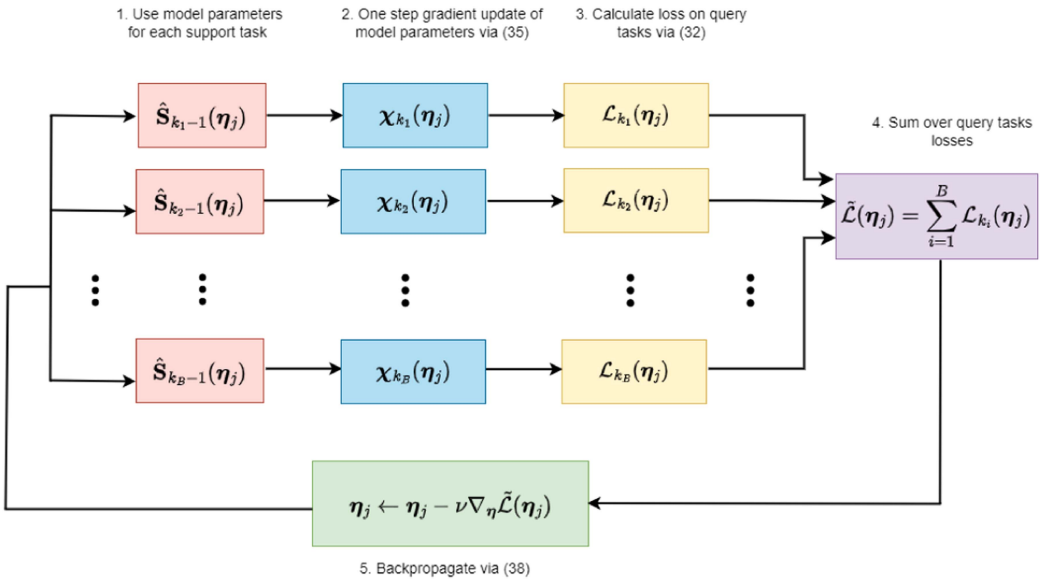


FIGURE 3. Meta-training process to obtain initialization hyperparameter η_j.

rate $\nu > 0$, leading to the iterative update:

$$\boldsymbol{\eta}_j^{(t)} = \boldsymbol{\eta}_j^{(t-1)} - \nu \nabla_{\boldsymbol{\eta}} \tilde{\mathcal{L}} \left(\boldsymbol{\eta}_j^{(t-1)} \right) \tag{38}$$

where $t = 1, \ldots, I_\eta$ indicates the iteration index, and I_η is the total number of iterations. The meta-learning optimization process for hyperparameter $\boldsymbol{\eta}_j$ is illustrated in Fig. 3 where for convenience, we set $\Lambda_j = \{k_1, k_2, \ldots, k_B\}$ and drop the reliance on the iteration index t for the gradient based optimization (38). This meta-learning hyperparameter optimization is performed at the receiver after every F received signal blocks. A smaller F results in more frequent updates, thereby enhancing the accuracy of the parameters. However, this comes at the expense of increased computational complexity.

The amalgamation of self-supervised online learning and meta-learning with the PrLcgNet detector, initially crafted for stationary environments, culminates in the creation of DyCoGNet. DyCoGNet fundamentally extends PrLcgNet's capabilities to accommodate evolving channel conditions. The mechanics of this adaptive process are concisely encapsulated in Algorithm 2.

D. COMPLEXITY ANALYSIS OF DYCOGNET

As discussed in Section III-D, the detection process complexity of DyCoGNet, which incorporates PrLcgNet as its base detector, is $\mathcal{O}(LN_t^2)$. It is crucial, however, to also consider the additional complexity introduced by the adaptive mechanisms of DyCoGNet. This assessment must include the computational overhead associated with pilot training, self-supervised online learning, and meta-learning. Within this adaptive framework, both pilot training and self-supervised online learning involve I_θ iterations per block. The computational burden for each iteration of gradient descent depends on the forward and backward passes through the network and the block size. Specifically, for a block size of T, the complexity per gradient descent iteration is approximately $\mathcal{O}(LN_t^2\, T)$. Moreover, DyCoGNet conducts optimization of its initialization hyperparameters over I_η learning iterations, scheduled periodically every F blocks. As a result, meta-learning effectively averages I_η/F iterations per block. Although this approach does increase the computational load by adding an average of I_η/F gradient computations per block, it potentially reduces the total number of required gradient steps compared to a purely online training regimen. This reduction is primarily attributed to the enhanced convergence speed of online training driven by meta-learning techniques.

V. NUMERICAL RESULTS

In this section, we present the numerical results of our proposed M-MIMO detectors, PrLcgNet for stationary settings and its extension DyCoGNet, for non-stationary settings. We begin by outlining the proposed detectors parameters and detailing channel parameters for both scenarios. For stationary scenarios, we explore PrLcgNet's convergence behavior and analyze its SER performance in comparison with established baseline detectors. For non-stationary M-MIMO scenarios, we evaluate the SER performance of DyCoGNet, benchmarking it against baseline detectors. Finally, we assess the robustness of DyCoGNet in scenarios with imperfect Channel State Information (CSI) at the receiver

A. EXPERIMENTAL SETUP

In our investigation, we consider a narrow-band M-MIMO system with $N_t = 32$ transmit antennas and $N_r = 64$ receive antennas, utilizing a 4-QAM modulation scheme. In stationary settings, perfect CSI knowledge is assumed at the receiver. In non-stationary settings, perfect CSI is assumed during the

Algorithm 2: DyCoGNet Adaptation on Block.

Input : Buffer content $\mathcal{B}_{j-1}$, model parameter $\boldsymbol{\theta}_{j-1}$, initialization hyperparamter $\boldsymbol{\eta}_{j-1}$, self-supervised learning iterations I_θ, meta-learning iterations I_η, learning rates μ, δ, ν, meta-learning interval F

Output: Hyperparameter $\boldsymbol{\eta}_j$, model weights $\boldsymbol{\theta}_j$, buffer content $\mathcal{B}_j$

Receive $\mathbf{Y}_j$
// Updating heap-based buffer
if $j \in \mathcal{T}_p$ **then**
  $\mathcal{B}_j = \mathcal{B}_{j-1} \sqcup \{(\mathbf{Y}_j, \bar{\mathbf{S}}_j)\}$
else
  Decode $\mathbf{Y}_j$ to obtain $\mathbf{W}_j$
  if *decoding is confident* **then**
    Remodulate $\mathbf{W}_j \to \bar{\mathbf{S}}_j$
    $\mathcal{B}_j = \mathcal{B}_{j-1} \sqcup \{(\mathbf{Y}_j, \bar{\mathbf{S}}_j)\}$
  else
    $\mathcal{B}_j = \mathcal{B}_{j-1}$
// Meta-learning every F blocks
if j *mod* $F == 0$ **then**
  for $t = 1, \ldots, I_\eta$ **do**
    $\tilde{\mathcal{L}}_j \leftarrow 0$
    for $k \in \Lambda_j$ **do**
      $\tilde{\mathcal{L}}_j \leftarrow \tilde{\mathcal{L}}_j + \mathcal{L}_k(\boldsymbol{\chi}_k(\boldsymbol{\eta}))$
    $\boldsymbol{\eta}_j \leftarrow \boldsymbol{\eta}_j - \nu \nabla_{\boldsymbol{\eta}} \tilde{\mathcal{L}}_j(\boldsymbol{\eta}_j)$
else
  $\boldsymbol{\eta}_j = \boldsymbol{\eta}_{j-1}$
// Online learning phase
if $j \in \mathcal{T}_p$ *or* $j \in \mathcal{T}_d$ *and decoding is confident* **then**
  $\boldsymbol{\theta}_j = \boldsymbol{\eta}_j$
  for $t = 1, \ldots, I_\theta$ **do**
    $\boldsymbol{\theta}_j \leftarrow \boldsymbol{\theta}_j - \mu \nabla_{\boldsymbol{\theta}} \mathcal{L}_j(\boldsymbol{\theta}_j)$
else
  $\boldsymbol{\theta}_j \leftarrow \boldsymbol{\theta}_{j-1}$

pilot transmission phase only; the receiver then uses the channel estimate obtained from the last pilot block during the data detection phase. That is, we do not assume real-time access to the actual channel matrices in the detection phase.

In stationary settings, we explore two primary models for the channel matrix $\mathbf{H}$. The first model employs uncorrelated Rayleigh fading, i.e., the entries of $\mathbf{H}$ are generated as independent and identically distributed (IID) random variables. The second model utilizes spatially correlated Rayleigh fading which is simulated using the widely recognized Kronecker product formulation [36]:

$$\mathbf{H} = \mathbf{R}_r^{\frac{1}{2}} \mathbf{U} \mathbf{R}_t^{\frac{1}{2}}. \tag{39}$$

Here, the entries of matrix $\mathbf{U} \in \mathbb{C}^{N_r \times N_t}$ are IID Rayleigh variables, while $\mathbf{R}_r \in \mathbb{C}^{N_r \times N_r}$ and $\mathbf{R}_t \in \mathbb{C}^{N_t \times N_t}$ represent the spatial correlation matrices at the receiver and transmitter, respectively. The entries of these matrices are obtained from the following exponential model [37]:

$$r_{ij} = \begin{cases} p^{j-i}, & \text{if } i \le j \\ r_{ji}, & \text{if } i > j \end{cases} \tag{40}$$

where ρ is the correlation coefficient and $|\rho| < 1$. In this work, ρ is set to be 0.5.

For the dynamic time-varying setting, we explore two different scenarios. Firstly, the Gauss-Markov block fading channel model [38] is utilized where the channel vector $\mathbf{h}_k(j)$ from user k to the base station in block j is governed by:

$$\mathbf{h}_k(j) = \alpha \mathbf{h}_k(j-1) + \sqrt{1-\alpha^2} \mathbf{R}_k^{1/2} \mathbf{n}_k(j), \ k = 1, \ldots, N_t \tag{41}$$

Here, the parameter $\alpha = J_0(2\pi f_d T_C)$, where $J_0(\cdot)$ signifies the zeroth order Bessel function of the first kind, plays a key role in determining the channel's temporal correlation coefficient. Specifically, f_d corresponds to the maximum Doppler frequency, while T_C signifies the coherence time. Furthermore, $\mathbf{R}_k$ denotes the spatial correlation matrix for user k, and $\mathbf{n}_k(j) \sim \mathcal{CN}(\mathbf{0}, \mathbf{I}_{N_r})$ is the channel additive noise. Secondly, our exploration extends to examining channels generated according to the TDL-A MIMO channel model from the 3GPP TR 38.901 standard [23].

For all experiments performed in the time-varying setting, we set the number of gradient descent iterations as $I_\theta = I_\eta = 100$, the learning rates $\mu = \nu = \delta = 10^{-3}$, the heap-based buffer size is set to 5, and the meta-learning interval is set to $F = 5$. Following [34], online self-supervised training is only used when the normalized Hamming distance between the re-encoded bits and those obtained from the hard decision output of the detector is less than a threshold of 0.02.

We assess detector performance using two key metrics: the SER and the average NMSE, the latter being expressed mathematically as:

$$\text{NMSE} = \frac{\mathbb{E}\{\|\hat{\mathbf{s}} - \mathbf{s}\|^2\}}{\mathbb{E}\{\|\mathbf{s}\|^2\}}. \tag{42}$$

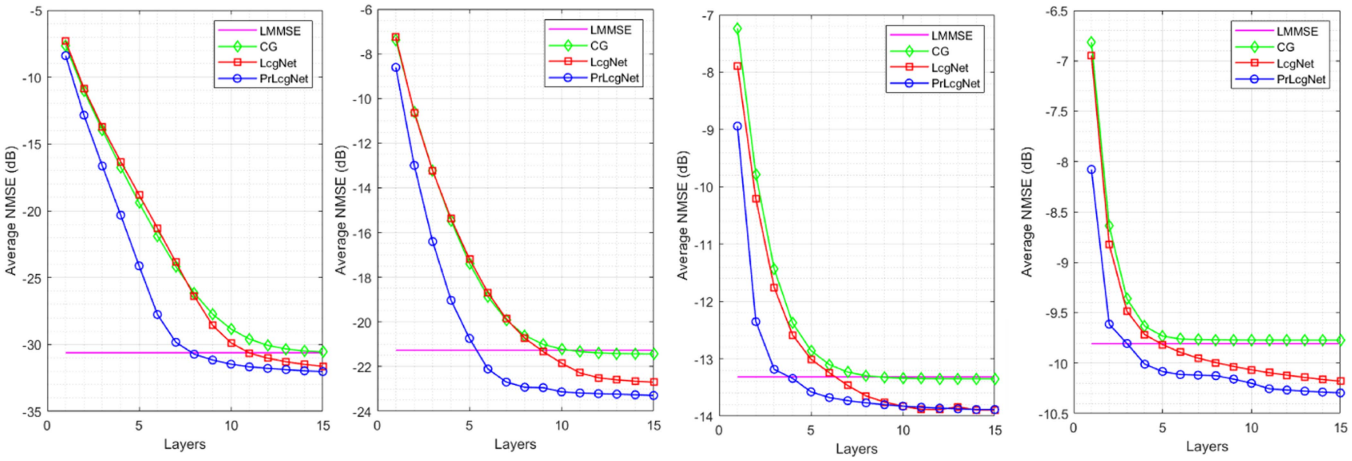
These metrics are evaluated for different values of system parameters as a function of the M-MIMO transmission SNR, defined as:

$$\text{SNR} = \frac{\mathbb{E}\{\|\mathbf{Hs}\|^2\}}{\mathbb{E}\{\|\mathbf{n}\|^2\}} = \frac{\gamma}{\sigma_n^2 N_r}. \tag{43}$$

B. STATIONARY CHANNEL RESULTS

We begin by comparing the convergence behavior of LcgNet and PrLcgNet offering insights into their effectiveness. Specifically, in Fig. 4, we plot the average NMSE of these detectors over the training dataset as the number of layers increases, and this for different SNR levels. For reference we also show the average NMSE of the CG and LMMSE detectors[5] At an SNR of 30 dB in Fig. 4(a), PrLcgNet achieves an average NMSE of -31dB with only 7 layers, while LcgNet

[5]For the CG detector, the number of layers corresponds to the iteration number while for LMMSE, there is no iteration or layer involved and, accordingly, the error level is constant.



(a) SNR = 30 dB (b) SNR = 20 dB (c) SNR = 10 dB (d) SNR = 5 dB

FIGURE 4. Average NMSE vs. layers at different SNRs.

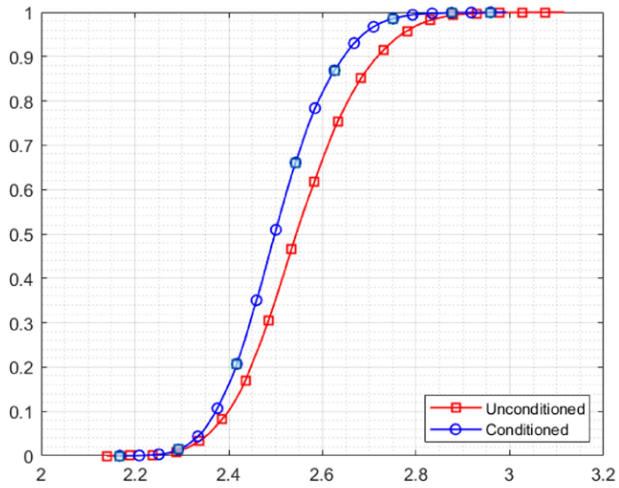


FIGURE 5. CDF of the eigenvalue spread of the filter matrix and preconditioned filter matrix (SNR = 30 dB).

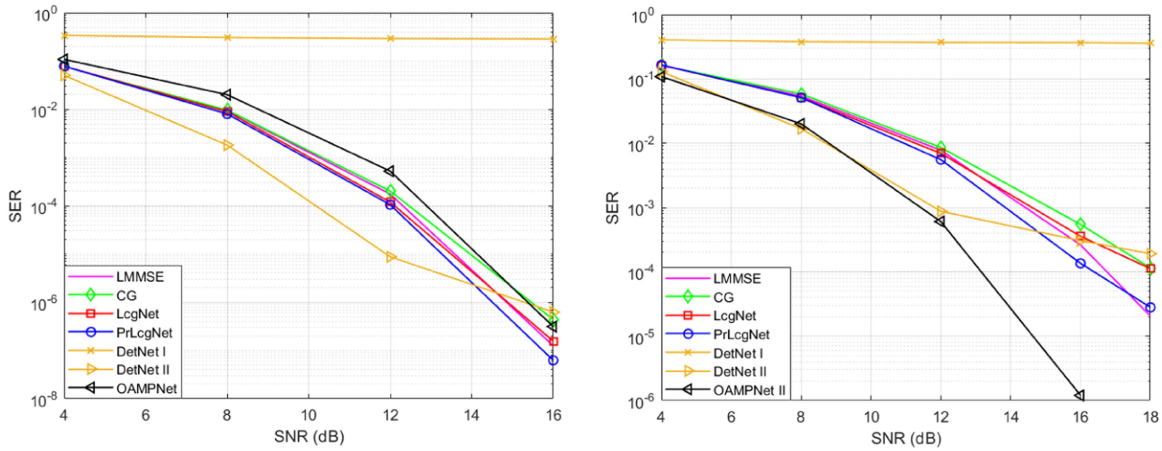
requires 13 layers to achieve similar results. Similarly, at 20 dB Fig. 4(b), PrLcgNet demonstrates an average NMSE of -23dB with 7 layers, whereas LcgNet requires 15 layers for comparable performance. Lower SNR levels of 10 dB and 5 dB presented in Fig. 4(c) and (d), further establish PrLcgNet's superiority, as it attains lower average NMSE values with fewer layers. In Fig. 5, we present the Cumulative Distribution Function (CDF) of the eigenvalue distribution for both the original filter matrix $\mathbf{A}$ used in LcgNet and the preconditioned filter matrix $\mathbf{M}^{-1}\mathbf{A}$ used in PrLcgNet. It is evident that the eigenvalues of the preconditioned matrix are more clustered compared to those of the original filter matrix. This is consistent with the observation of improved convergence behavior in PrLcgNet compared to LcgNet.

Next, we assess the SER performance of PrLcgNet, comparing it with several other detectors for reference. The detectors under evaluation include:

- *PrLcgNet:* 10-layer PrLcgNet model.
- *LcgNet:* 10-layer LcgNet model.
- *CG:* 10-iteration CG algorithm.
- *LMMSE:* LMMSE detector as described in (5).
- *DetNet I:* 10-layer DetNet [14] model with similar per-layer detection complexity as PrLcgNet.
- *DetNet II:* 10-layer DetNet model with the per-layer complexity outlined in [14].
- *OAMPNET:* 1-layer OAMPNet model [15].

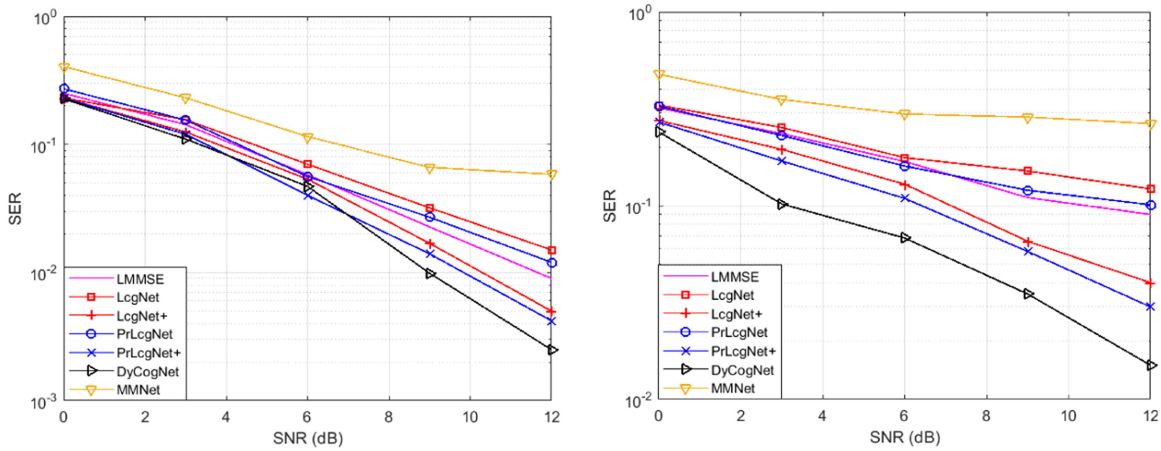
We present the resulting SER curves in Fig. 6, covering a range of SNR levels. Notably, PrLcgNet consistently performs on par with the LMMSE detector while requiring lower computational complexity. We also note that both LcgNet and PrLcgNet outperform a 10-iteration CG detector which shares a similar order of complexity.

Comparing PrLcgNet with DetNet, the results show that at equal per-layer detection complexity, PrLcgNet (and all the other considered detectors) significantly outperforms the DetNet-I detector, which employs a considerably larger parameter count of 62,890, in contrast to PrLcgNet's 640 parameters. Enhancing DetNet to its original detection complexity, i.e., DetNet II, does result in a performance boost, albeit at a cost of increased complexity, approximately fourfold that of PrLcgNet, alongside a substantial parameter count of 986,890. At low to moderate SNR, DetNet II exhibits superior performance in both uncorrelated and correlated scenarios; however, with increasing SNR levels, its performance levels off and the situation reverses. Regarding OAMPNet, which employs fewer parameters than the other detectors (2 per layer), its performance depends on the level of spatial correlation: in the case of uncorrelated fading as shown in Fig. 6(a), it is surpassed by all other detectors (except DetNet I), while in correlated scenarios in Fig. 6(b), it achieves the best performance. However, this advantage comes at the cost of a significant increase in complexity. Specifically, the required matrix inversion per layer in OAMPNet incurs a



(a) Rayleigh fading channel (b) Spatially correlated channel

FIGURE 6. SER vs. SNR for various detectors under different stationary channel conditions.



(a) maximum user speed of 30km/h (b) maximum user speed of 60km/h

FIGURE 7. SER vs. SNR for Gauss-Markov block fading scenario.

computational cost of $\mathcal{O}(N_r^3)$, which becomes prohibitive for M-MIMO systems where N_r is typically very large.

C. NON-STATIONARY CHANNEL RESULTS

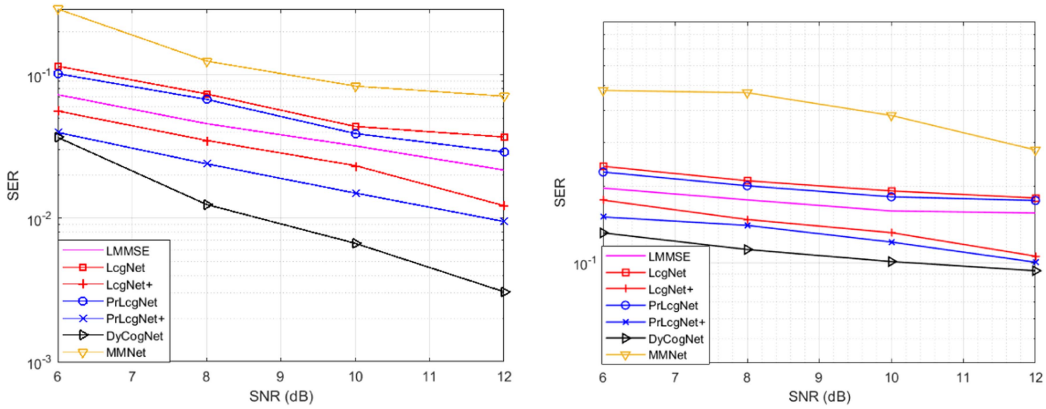
Next, we assess the SER performance of DyCoGNet in non-staationary channel scenarios, comparing it with other detectors for reference. The detectors under evaluation include:

- *LMMSE:* LMMSE detector as described in (5).
- *PrLcgNet:* 10-layer PrLcgNet model trained exclusively on pilot data before initiating data detection.
- *PrLcgNet+:* 10-layer PrLcgNet model which integrates online self-supervised learning.
- *LcgNet:* 10-layer LcgNet model trained exclusively on pilot data before initiating data detection.
- *LcgNet+:* 10-layer LcgNet model that incorporates online self-supervised learning.
- *DyCoGNet:* 10-layer PrLcgNet model, combining both online self-supervised learning and meta-learning.
- *MMNet:* 10-layer MMNet model [17] trained exclusively on pilot data before initiating data detection.

1) GAUSS-MARKOV BLOCK FADING CHANNEL

We configure the simulations with $T_p = 16$ pilot blocks and $T_d = 50$ data blocks, with a block length of $T = 209$ symbols. In effect, this corresponds to uses 330 message bits encoded through a Reed-Solomon (RS) [19,15] encoder for each of the N_t antennas. Two user speeds, 30 km/h and 60 km/h, are considered, along with a center frequency of 5 GHz.

Fig. 7(a) and (b) provide comparisons of average SER performance under time-varying channel conditions over 10 trial runs for user speeds 30 km/h and 60 km/h respectively. At lower speed in Fig. 7(a), it is observed that DyCoGNet exhibits superior performance. The advantage of meta-learning is evident as DyCoGNet surpasses PrLcgNet+, which lacks this component. PrLcgNet+ shows an advantage over LcgNet+, underscoring the effectiveness of its



(a) maximum Doppler spread $f_D = 10$ Hz (b) maximum Doppler spread $f_D = 20$ Hz

FIGURE 8. SER vs. SNR for TDL-A channel scenario for different maximum Doppler spreads.

preconditioning approach. Both models perform better than their counterparts that train only on pilot signals, emphasizing the benefits of online self-supervised learning. The LMMSE detector, while slightly superior to both PrLcgNet and LcgNet, shows a diminishing performance advantage as SNR increases. MMNet, despite its large parameter count (41,600 compared to DyCoGNet's 640), is the least performing detector due to its lack of adaptation to changing channel conditions in the data transmission phase, as it relies solely on information from outdated pilot signals. At a higher speed in Fig. 7(b), all detectors experience a significant drop in SER performance due to more substantial channel variation. Nonetheless, similar performance trends persist as detectors incorporating self-supervised online learning still maintain superior performance compared to those that do not, effectively handling outdated channel information through pilot free online re-training.

2) M-MIMO TDL-A CHANNEL

For this scenario, we set $T_p = 15$ pilot blocks and $T_d = 45$ data blocks, with each block comprising $T = 114$ symbols equivalent to 180 message bits encoded via a RS [19,15] encoder for each antenna. The delay spread is set to 30 ns, and we explore two distinct mobility scenarios: a low-mobility scenario corresponding to a maximum Doppler spread $f_D = 10$ Hz and a high-mobility scenario with $f_D = 20$ Hz.

In Fig. 8(a), where $f_D = 10$ Hz, DyCoGNet consistently outperforms all other detectors. Significant performance improvements are evident for both LcgNet+ and PrLcgNet+ when compared to their base detectors without self-supervised online adaptation. MMNet shows the poorest performance among all compared detectors for reasons explained above.

Moving to the high-mobility scenario characterized by $f_D = 20$ Hz, the overall performance trend remains largely consistent, with DyCoGNet outperforming all other detectors. However, there is a significant degradation in the overall performance of detectors due to the rapid obsolescence of channel information. Interestingly, we observe an error floor, reminiscent of the effects observed in differential phase shift keying systems in high Doppler scenarios [39, Chapter 6]. We also note that in these and several other experiments with different rates of channel variations, characterized by different f_D values, conventional methods such as LMMSE were not observed to outperform the online learning-based methods at any point.

Table 2 provides a detailed comparison of computational complexities for all detectors, considering training, testing, and (where applicable) online-adaptation (where applicable), in terms of key system parameters. From the table, it can be observed that the DyCoGNet detector incurs an additional cost of I_η/F iterations per block in its online adaptation phase due to the meta-learning component. However, due to obtaining good initializations for the training phase via meta-learning, the overall number of training iterations can potentially be reduced. Another point to note is that the MMNet has a training and testing complexity that is quadratic in N_r which can be prohibitive in M-MIMO systems where N_r is typically very large.

Table 3 provides a comparison of the average training and testing runtimes for the detectors under consideration. Notably, MMNet incurs the largest training and testing times due to its relatively large parameter count. DyCoGNet has a per-iteration training time that is larger than that of PrLcgNet+ and LcgNet+ due to the meta-learning component, however this gap can be reduced by reducing the frequency of meta-learning albeit at the cost of reduced SER performance. The simulations were conducted using an Intel Xeon Processor running at 2.20 GHz and 12 GB RAM.

D. IMPERFECT CSI

Finally, we assess the robustness of DyCoGNet under conditions of imperfect CSI. In typical wireless communication systems, the receiver generally operates with imperfect CSI

TABLE 2. Detector Complexity in Non-Stationary Scenarios per Block

Detectors	Training Complexity	Testing Complexity	Online Adaptation
DyCoGNet	$\mathcal{O}\left(I_\theta LTN_t^2\right)$	$\mathcal{O}(LTN_t^2)$	$\mathcal{O}\left(\left(I_\theta + \frac{I_\eta}{F}\right) LTN_t^2\right)$
PrLcgNet+/LcgNet+	$\mathcal{O}\left(I_\theta LTN_t^2\right)$	$\mathcal{O}(LTN_t^2)$	$\mathcal{O}\left(I_\theta LTN_t^2\right)$
PrLcgNet/LcgNet	$\mathcal{O}\left(I_\theta LTN_t^2\right)$	$\mathcal{O}(LTN_t^2)$	N/A
MMNET	$\mathcal{O}\left(I_\theta LTN_r^2\right)$	$\mathcal{O}(LTN_r^2)$	N/A
LMMSE	N/A	$\mathcal{O}(TN_t^3)$	N/A

TABLE 3. Average Training and Testing Runtimes of Detectors per Block of 100 Symbols

Detectors	Training time per iteration (sec)	Testing time (sec)
DyCoGNet	0.048	0.016
PrLcgNet/+	0.023	0.016
LcgNet/+	0.021	0.012
MMNet	0.333	0.149
LMMSE	N/A	0.011

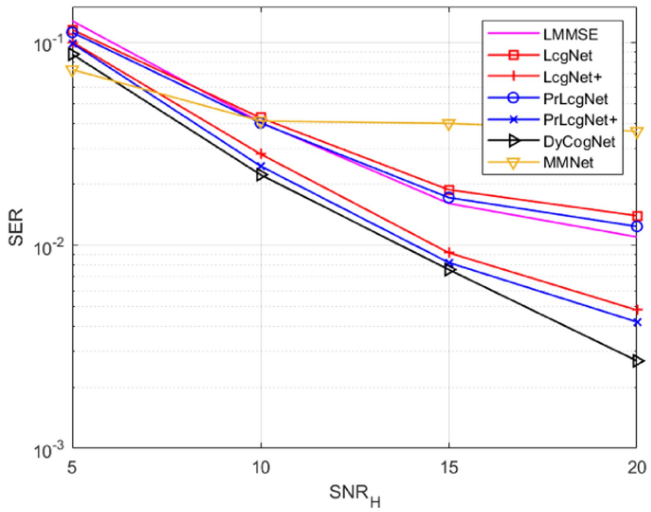


FIGURE 9. SER vs. SNR_H for Gauss-Markov block fading scenario with imperfect CSI.

and must rely on estimated channel matrices, which inherently include errors. To simulate the effects of imperfect CSI on detector performance, we hence introduce an additive error term to the true M-MIMO channel matrix, mathematically represented as:

$$\tilde{\mathbf{H}} = \mathbf{H} + \mathbf{E} \tag{44}$$

where $\tilde{\mathbf{H}}$ denotes the noisy estimate of the actual channel matrix $\mathbf{H}$, and $\mathbf{E} \sim \mathcal{CN}(0, \sigma_e^2 \mathbf{I})$ represents the estimation error. We quantify the quality of channel estimation using the following M-MIMO channel estimation SNR:

$$\text{SNR}_\text{H} = \frac{\mathbb{E}\left[\|\mathbf{H}\|_F^2\right]}{\mathbb{E}\left[\|\mathbf{E}\|_F^2\right]}. \tag{45}$$

The SER performance of the various detectors under study as a function of SNR_H is illustrated in Fig. 9. Here, the Gauss-Markov fading scenario is assumed with the same parameter settings as in Fig. 7(a) and the transmission SNR (43) set to 12 dB. As can be seen from Fig. 9, there is a general degradation in the SER performance of all detectors as the size of the estimation errors increases, corresponding to a decrease in the SNR_H metric. Nevertheless, DyCoGNet consistently outperforms the other benchmark detectors over a wide range of SNR_H.

VI. CONCLUSION

In this study, we addressed the challenge of symbol detection in time-varying M-MIMO systems. While learning-based methods have shown promise in stationary scenarios, their adaptation to non-stationary conditions has remained a significant challenge due to the need for parameter adaptation. To overcome this challenge, we introduced a series of innovations building upon the foundation of the LcgNet M-MIMO detector. First we introduced PrLcgNet, an enhancement of LcgNet achieved by incorporating a preconditioner in the LcgNet architecture. This addition not only improved convergence but also reduced complexity, all while maintaining performance superiority over LcgNet, as confirmed through extensive simulations. Subsequently, we introduced DyCoGNet, an extension of PrLcgNet purpose-built for time-varying M-MIMO scenarios. DyCoGNet uses FEC-aided self-supervised learning for autonomous adaptation. Furthermore, it leverages meta-learning to enable swift adaption to time-varying channel conditions. Our simulations demonstrated that PrLcgNet achieved faster convergence and comparable performance to LcgNet in stationary scenarios and DyCoGNet, its extension for dynamic contexts, significantly improved SER performance compared to baseline methods without meta-learning and online self-supervised learning.

REFERENCES

[1] T. Olutayo and B. Champagne, "Learned preconditioned conjugate gradient descent for massive MIMO detection," in *Proc. IEEE Latin- Amer. Conf. Commun.*, 2022, pp. 1–6.
[2] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, May/Jun. 2020.
[3] 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.
[4] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1639–1642, Jul. 1999.

[5] E. G. Larsson, "MIMO detection methods: How they work," *IEEE Signal Process. Mag.*, vol. 26, no. 3, pp. 91–95, May 2009.
[6] S. Yang and L. Hanzo, "Fifty years of MIMO detection: The road to large-scale MIMOs," *IEEE Commun. Surv. Tuts.*, vol. 17, no. 4, pp. 1941–1988, Fourthquarter 2015.
[7] L. V. Nguyen, N. T. Nguyen, N. H. Tran, M. Juntti, A. L. Swindlehurst, and D. H. N. Nguyen, "Leveraging deep neural networks for massive MIMO data detection," *IEEE Wireless Commun.*, vol. 30, no. 1, pp. 174–180, Feb. 2023.
[8] L. Li, H. Hou, and W. Meng, "Convolutional-neural-network-based detection algorithm for uplink multiuser massive MIMO systems," *IEEE Access*, vol. 8, pp. 64250–64265, 2020.
[9] J. Xu, L. Li, L. Zheng, and L. Liu, "Detect to learn: Structure learning with attention and decision feedback for MIMO-OFDM receive processing," *IEEE Trans. Commun.*, vol. 72, no. 1, pp. 146–161, Jan. 2024.
[10] N. Shlezinger, R. Fu, and Y. C. Eldar, "DeepSIC: Deep soft interference cancellation for multiuser MIMO detection," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1349–1362, Feb. 2021.
[11] A. Balatsoukas-Stimming and C. Studer, "Deep unfolding for communications systems: A survey and some new directions," in *Proc. IEEE Int. Workshop Signal Process. Syst.*, 2019, pp. 266–271.
[12] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *Proc. IEEE*, vol. 111, no. 5, pp. 465–499, May 2023.
[13] J. R. Hershey, J. L. Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," 2014, *arXiv:1409.2574*.
[14] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun.*, 2017, pp. 1–5.
[15] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "A model-driven deep learning network for MIMO detection," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2018, pp. 584–588.
[16] Y. Wei, M.-M. Zhao, M. Hong, M.-J. Zhao, and M. Lei, "Learned conjugate gradient descent network for massive MIMO detection," *IEEE Trans. Signal Process.*, vol. 68, pp. 6336–6349, 2020.
[17] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive neural signal detection for massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5635–5648, Aug. 2020.
[18] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, "ViterbiNet: A deep learning based Viterbi algorithm for symbol detection," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3319–3331, May 2020.
[19] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, "Data-driven factor graphs for deep symbol detection," in *Proc. IEEE Int. Symp. Inf. Theory*, 2020, pp. 2682–2687.
[20] S. Park, O. Simeone, and J. Kang, "Meta-learning to communicate: Fast end-to-end training for fading channels," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 5075–5079.
[21] Z. Zhang, N. Wang, H. Wu, C. Tang, and R. Li, "MR-DRO: A fast and efficient task offloading algorithm in heterogeneous edge/cloud computing environments," *IEEE Internet Things*, vol. 10, no. 4, pp. 3165–3178, Feb. 2023.
[22] S. Park, H. Jang, O. Simeone, and J. Kang, "Learning to demodulate from few pilots via offline and online meta-learning," *IEEE Trans. Signal Process.*, vol. 69, pp. 226–239, 2021.
[23] ETSI 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," Tech. Rep. TR 38.901 version 14.0.0 Release 14, 2017.
[24] M. Biguesh and A. B. Gershman, "Training-based MIMO channel estimation: A study of estimator tradeoffs and optimal training signals," *IEEE Trans. Signal Process.*, vol. 54, no. 3, pp. 884–893, Mar. 2006.
[25] S. Willhammar, J. Flordelis, L. Van der Perre, and F. Tufvesson, "Channel hardening in massive MIMO–A measurement based analysis," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun.*, 2018, pp. 1–5.
[26] E. Telatar, "Capacity of multi-antenna Gaussian channels," *Eur. Trans. Telecommun.*, vol. 10, no. 6, pp. 585–595, 1999.
[27] J. Nocedal and S. J. Wright, *Numerical Optimization*. Berlin, Germany: Springer, 1999.
[28] K. Chen, *Matrix Preconditioning Techniques and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[29] J. Shewchuk et al., "An introduction to the conjugate gradient method without the agonizing pain," Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU-CS-94-125, Aug. 1994.
[30] J. Wang and G. Meng, "SAOR preconditioned conjugate gradient method," in *Proc. IEEE Workshop Intell. Inf. Technol. Appl.*, 2007, pp. 331–334.
[31] J. Jin, Y. Xue, Y.-L. Ueng, X. You, and C. Zhang, "A split preconditioned conjugate gradient method for massive MIMO detection," in *Proc. IEEE Int. Workshop Signal Process. Syst.*, 2017, pp. 1–6.
[32] H. Dag and A. Semlyen, "A new preconditioned conjugate gradient power flow," *IEEE Trans. Power Syst.*, vol. 18, no. 4, pp. 1248–1255, Nov. 2003.
[33] B. M. Hochwald and T. L. Marzetta, "Unitary space-time modulation for multiple-antenna communications in Rayleigh flat fading," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 543–564, Mar. 2000.
[34] T. Raviv, S. Park, O. Simeone, Y. C. Eldar, and N. Shlezinger, "Online meta-learning for hybrid model-based deep receivers," *IEEE Trans. Wireless Commun.*, vol. 22, no. 10, pp. 6415–6531, Oct. 2023.
[35] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.
[36] J.-P. Kermoal, L. Schumacher, K. I. Pedersen, P. E. Mogensen, and F. Frederiksen, "A stochastic MIMO radio channel model with experimental validation," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 6, pp. 1211–1226, Aug. 2002.
[37] S. L. Loyka, "Channel capacity of MIMO architecture using the exponential correlation matrix," *IEEE Commun. Lett.*, vol. 5, no. 9, pp. 369–371, Sep. 2001.
[38] W. Chen and R. Zhang, "Kalman-filter channel estimator for OFDM systems in time and frequency-selective fading environment," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2004, pp. 337–380.
[39] A. Goldsmith, *Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.