ADMM-SLPNet: A Model-Driven Deep Learning Framework for Symbol-Level Precoding

Junwen Yang, *Graduate Student Member, IEEE,* Ang Li, *Senior Member, IEEE,* Xuewen Liao, *Member, IEEE,* and Christos Masouros, *Senior Member, IEEE*

***Abstract*—Constructive interference (CI)-based symbol-level precoding (SLP) is an emerging downlink transmission technique for multi-antenna communications systems, and its low-complexity implementations are of practical importance. In this paper, we propose an interpretable model-driven deep learning framework to accelerate the processing of SLP. Specifically, the network topology is carefully designed by unrolling a parallelizable algorithm based on the proximal Jacobian alternating direction method of multipliers (PJ-ADMM), attaining parallel and distributed architecture. Moreover, the parameters of the iterative PJ-ADMM algorithm are untied to parameterize the network. By incorporating the problem-domain knowledge into the loss function, an unsupervised learning strategy is further proposed to discriminatively train the learnable parameters using unlabeled training data. Simulation results demonstrate significant efficiency improvement of the proposed ADMM-SLPNet over benchmark schemes.**

***Index Terms*—Deep learning, deep unfolding, algorithm unrolling, model-driven, symbol-level precoding, ADMM.**

I. Introduction

In a multi-user multi-antenna wireless communications system with accessible channel state information (CSI), the noiseless received signal in each symbol slot at the receiver can be exactly designed prior to transmission. Such processing precodes the data symbols onto transmit antennas once per symbol slot, known as symbol-level precoding (SLP) [1]–[3]. The most prominent feature provided by SLP has been recognized as its ability in interference management, where all interference is meant to be exploited rather than suppressed to make it constructive to the desired signal and thus beneficial to correct detection [1]. To this end, constructive interference (CI)-based SLP is also referred to as interference exploitation precoding. Compared to the currently popular interference mitigation precoding [4], namely the conventional block-level precoding (BLP) that utilizes CSI only to design a linear precoder, whose precoding matrix is hence updated every channel coherence interval, SLP that operates on a symbol level has been shown to exhibit enhanced performance under a variety of design criteria [1], for example, the power minimization (PM) SLP subject to instantaneous signal-to-interference-plus-noise ratio (SINR) constraints and the max-min SINR balancing (SB) SLP with power budget. This is because the nonlinear precoder provides more flexible and accurate manipulations in signal processing than the linear ones. Nevertheless, it seems that the state-of-the-art SLP is not in full compliance with the philosophy of pragmatism in realistic systems owing to the high complexity of solving the corresponding constrained nonlinear optimization problem on a symbol-by-symbol basis.

To address the computational issue of SLP, there have been numerous studies, e.g., the efficient gradient projection algorithm (EGPA) [1], the closed-form suboptimal solution [5], the iterative closed-form algorithm [6], [7] for SB-SLP, and the CI-BLP [8]. Recently, the separability of the PM-SLP problem has been revealed in [9], which has also developed several parallel methods to speed up the SLP process. As a step further, [10] has proven an explicit duality between the PM-SLP and SB-SLP problems, which enables solving the SB-SLP problem leveraging the parallel methods proposed in [9] based on the proximal Jacobian alternating direction method of multipliers (PJ-ADMM). The efficiency of the PJ-ADMM-based SLP is fundamentally affected by the parameters therein, while determining the parameters for fast convergence is a challenging and time-consuming task.

In this paper, our goal is to accelerate the processing of the PJ-ADMM-based SLP using the model-driven deep learning methodology. The idea of unfolding a model-based iterative algorithm into a deep network has been proposed in sparse coding [11]. This ideal and the discriminative parameter learning approach have been named deep unfolding in [12] and applied to Markov random fields. In [13], the ADMM algorithm for compressed sensing has been mapped to a deep architecture dubbed ADMM-Net via deep unfolding. The same technique has been used to unfold a proximal interior-

J. Yang, A. Li are with the School of Information and Communications Engineering, Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China (e-mail: jwyang@stu.xjtu.edu.cn; ang.li.2020@xjtu.edu.cn).

X. Liao is with the School of Information and Communications Engineering, Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China, and also with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China (e-mail: yeplos@mail.xjtu.edu.cn).

C. Masouros is with the Department of Electronic and Electrical Engineering, University College London, London WC1E 7JE, U.K. (e-mail: c.masouros@ucl.ac.uk).

The work of Ang Li was supported in part by the Young Elite Scientists Sponsorship Program by CIC (Grant No. 2021QNRC001), in part by the National Natural Science Foundation of China under Grant 62101422, and in part by the Science and Technology Program of Shaanxi Province under Grant 2021KWZ-01. The work of Xuewen Liao was supported in part by the National Key Research and Development Project of China under Grant 2019YFB2101600; in part by the Science and Technology Program of Shaanxi Province under Grant 2021GXLH-Z-038; and in part by the Open Research Fund of the National Mobile Communications Research Laboratory, Southeast University, under Grant 2020D12. *(Corresponding author: Xuewen Liao and Ang Li.)*

point method (IPM) for image restoration in [14], which has been further employed in PM-SLP to attain an unsupervised deep neural network [15]. Instead of unfolding an existing optimization algorithm, [16] uses customized loss function and lambda layers with expert knowledge to tackle the SB-SLP problem. Motivated by these works, we propose an unsupervised model-driven deep learning framework for SLP named ADMM-SLPNet. Specifically, we map the PJ-ADMM algorithm to a data flow graph and design the topology of the proposed ADMM-SLPNet by unrolling the PJ-ADMM-based SLP, retaining the interpretable feature of the model-based algorithm rather than treating the network as an end-to-end black box. We then untie the parameters of PJ-ADMM across iterations to parameterize each layer of the ADMM-SLPNet. To discriminatively train the untied learnable parameters, we propose an unsupervised learning strategy by incorporating the problem-domain knowledge into the loss function. In addition, the parallel and distributed feature of the PJ-ADMM is inherited by the proposed network. Simulation results demonstrate the superiority of the ADMM-SLPNet in terms of processing speed over the original PJ-ADMM SLP algorithm and other benchmark schemes.

The remainder of this paper is organized as follows. Section II introduces the system model and the concept of CI, then formulates the PM-SLP problem. Section III briefly reviews the iterative PJ-ADMM algorithm, followed by the detailed design of the proposed ADMM-SLPNet, which includes the network topology based on deep unfolding, the proposed unsupervised training strategy, and the computational cost analysis. Section IV provides the simulation results. Section V concludes the paper.

Notations: $(\cdot)^T$ and $(\cdot)^{-1}$ denote the transpose and inverse operators, respectively. $|\cdot|$ represents the absolute value of a real-valued scalar or the modulus of a complex-valued scalar. $\|\cdot\|$ denotes the Euclidean norm of a vector or the spectral norm of a matrix. $\Re\{\cdot\}$ and $\Im\{\cdot\}$ respectively denote the real/imaginary parts of a complex-valued input. $\mathbf{1}$ and $\mathbf{I}$ represents the all-ones vector and the identity matrix of appropriate dimensions, respectively.

II. Preliminaries

A. System Model

Consider the downlink transmission of a multi-user multiple-input single-output (MU-MISO) system, where the base station (BS) equipped with N_t antennas serves K single-antenna users in the same time-frequency resource block. In each symbol slot, the relationship between the transmit signal $\tilde{\mathbf{x}}$ and the received signal of the k-th user $\tilde{y}_k$ can be expressed as

$$\tilde{y}_k = \tilde{\mathbf{h}}_k^T \tilde{\mathbf{x}} + \tilde{z}_k, \tag{1}$$

where $\tilde{\mathbf{h}}_k \in \mathbb{C}^{N_t}$ denotes the quasi-static Rayleigh flat-fading channel vector between the BS and the k-th user, and $\tilde{z}_k \sim \mathcal{CN}(0, \sigma_k^2)$ is the complex-valued additive white Gaussian noise at the k-th user. At the BS side, SLP is employed to map the data symbol vector $\tilde{\mathbf{s}} \triangleq [\tilde{s}_1, \cdots, \tilde{s}_K]^T \in \mathbb{C}^K$ to the transmit signal $\tilde{\mathbf{x}}$.

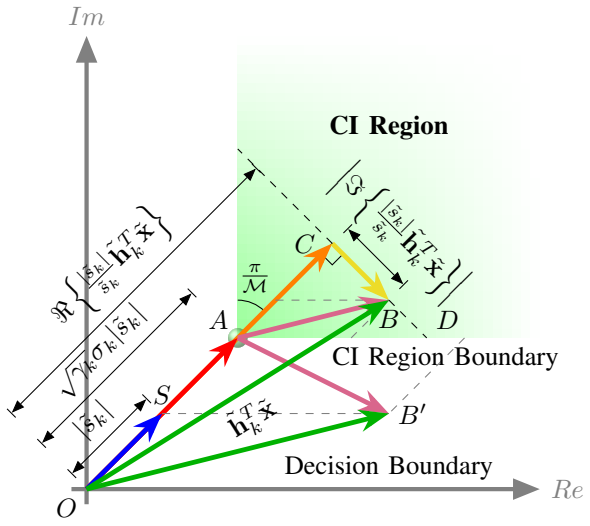

Fig. 1. The first quadrant of the QPSK constellation.

B. Constructive Interference

When CI is attained via SLP, there is no interference destructive to the desired signal. Therefore the quality of service metric is shifted from received average SINR to instantaneous SINR, which can be written as

$$\mathrm{SINR}_k \triangleq \frac{\left|\tilde{\mathbf{h}}_k^T \tilde{\mathbf{x}}\right|^2}{\sigma_k^2}. \tag{2}$$

To illustrate the concept of CI, without loss of generality, we take the first quadrant of the quadrature phase-shift-keying (QPSK) constellation, whose modulation order is $\mathcal{M} = 4$, as an example and depict the geometric interpretation of CI in Fig. 1. $\overrightarrow{OS} = \tilde{s}_k$, $\overrightarrow{OA} = \sqrt{\gamma_k}\sigma_k\tilde{s}_k$, $\overrightarrow{OB} = \tilde{\mathbf{h}}_k^T\tilde{\mathbf{x}}$ respectively denote the symbol of interest, the nominal constellation point, the noiseless received signal of the k-th user, where γ_k is a prescribed SINR threshold. One important observation is that when $\overrightarrow{OB}$ lies in the green-shaded CI region, the instantaneous SINR is guaranteed to be no less than $\gamma_k |\tilde{s}_k|^2$. On the contrary, if the noiseless received signal falls out of the CI region, see $\overrightarrow{OB'} = \overrightarrow{OA} + \overrightarrow{AB'}$ in Fig. 1, the nominal constellation point $\overrightarrow{OA}$ is imposed with a destructive signal $\overrightarrow{AB'}$. The instantaneous SINR is accordingly less than $\gamma_k |\tilde{s}_k|^2$. Therefore the design criteria of SLP ought to specify the location of $\overrightarrow{OB}$ in the CI region. For a clear illustration, we orthogonally decompose $\overrightarrow{OB}$ along $\overrightarrow{OA}$, which results in that $\overrightarrow{OB} = \overrightarrow{OC} + \overrightarrow{CB}$. Furthermore, if $\overrightarrow{OB}$ locates in the CI region, we have $\left|\overrightarrow{CD}\right| \geq \left|\overrightarrow{CB}\right|$, where D denotes the intersection of $\overrightarrow{CB}$ and its nearest CI region boundary. From the geometric relationships illustrated in Fig. 1, the CI constraints that guarantee the noiseless received signal located in the CI region can be expressed as

$$\Re\left\{\hat{\mathbf{h}}_k^T \tilde{\mathbf{x}}\right\} - \frac{\left|\Im\left\{\hat{\mathbf{h}}_k^T \tilde{\mathbf{x}}\right\}\right|}{\tan\frac{\pi}{\mathcal{M}}} \geq \sqrt{\gamma_k}\sigma_k, \forall k, \tag{3}$$

where $\hat{\mathbf{h}}_k^T \triangleq \frac{\tilde{\mathbf{h}}_k^T}{\tilde{s}_k}$.

C. Problem Formulation

With the CSI and information on data symbols, SLP dedicated to design the transmit signal under certain criteria. The PM-SLP problem on $\tilde{\mathbf{x}}$ that minimizes the total transmit power subject to CI constraints has the following mathematical description [1]:

$$\begin{aligned} \min_{\tilde{\mathbf{x}}} \quad & \|\tilde{\mathbf{x}}\|^2 \\ \text{s.t.}\ & \Re\left\{\hat{\mathbf{h}}_k^T\tilde{\mathbf{x}}\right\} - \frac{\left|\Im\left\{\hat{\mathbf{h}}_k^T\tilde{\mathbf{x}}\right\}\right|}{\tan\frac{\pi}{\mathcal{M}}} \geq \sqrt{\gamma_k}\sigma_k,\ \forall k. \end{aligned} \tag{4}$$

The above quadratic programming problem is convex and can be solved via off-the-shelf solvers. But most standard solvers, e.g., SeDuMi and SDPT3, are based on the high-complexity IPM. To alleviate the computational burden, a number of algorithms were proposed, e.g., the EGPA [1] and the suboptimal closed-form solution [5].

Based on the separability of the PM-SLP problem shown in [9], the PM-SLP problem (4) can be equivalently transformed to the following separable real-valued problem [9]:

$$\begin{aligned} \min_{\{\mathbf{x}_i\}} \quad & \sum_{i=1}^{N} \|\mathbf{x}_i\|^2 \\ \text{s.t.}\ & \sum_{i=1}^{N} \mathbf{A}_i\mathbf{x}_i \succeq \mathbf{b}, \end{aligned} \tag{5}$$

where

$$\mathbf{x} \triangleq \begin{bmatrix} \Re\{\tilde{\mathbf{x}}\} \\ \Im\{\tilde{\mathbf{x}}\} \end{bmatrix} \in \mathbb{R}^{2N_t}, \tag{6a}$$

$$\mathbf{A} \triangleq \left[\bar{\mathbf{A}}_1^T, \cdots, \bar{\mathbf{A}}_K^T\right]^T \in \mathbb{R}^{2K\times 2N_t}, \tag{6b}$$

$$\bar{\mathbf{A}}_k \triangleq \mathbf{T}\mathbf{S}_k\mathbf{H}_k, \tag{6c}$$

$$\mathbf{T} \triangleq \begin{bmatrix} 1 & -\frac{1}{\tan\frac{\pi}{\mathcal{M}}} \\ 1 & \frac{1}{\tan\frac{\pi}{\mathcal{M}}} \end{bmatrix} \in \mathbb{R}^{2\times 2}, \tag{6d}$$

$$\mathbf{S}_k \triangleq \begin{bmatrix} \Re\{\tilde{s}_k^{-1}\} & -\Im\{\tilde{s}_k^{-1}\} \\ \Im\{\tilde{s}_k^{-1}\} & \Re\{\tilde{s}_k^{-1}\} \end{bmatrix} \in \mathbb{R}^{2\times 2}, \tag{6e}$$

$$\mathbf{H}_k \triangleq \begin{bmatrix} \Re\left\{\tilde{\mathbf{h}}_k^T\right\} & -\Im\left\{\tilde{\mathbf{h}}_k^T\right\} \\ \Im\left\{\tilde{\mathbf{h}}_k^T\right\} & \Re\left\{\tilde{\mathbf{h}}_k^T\right\} \end{bmatrix} \in \mathbb{R}^{2\times 2N_t}, \tag{6f}$$

$$\mathbf{b} \triangleq \left[\mathbf{b}_1^T, \cdots, \mathbf{b}_K^T\right]^T \in \mathbb{R}^{2K}, \tag{6g}$$

$$\mathbf{b}_k \triangleq \sqrt{\gamma_k}\sigma_k\mathbf{1} \in \mathbb{R}^2. \tag{6h}$$

$\mathbf{x}_i \in \mathbb{R}^{n_i}$ and $\mathbf{A}_i \in \mathbb{R}^{2K\times n_i}$ respectively denote the i-th blocks of $\mathbf{x}$ and $\mathbf{A}$, where $\sum_{i=1}^{N} n_i = 2N_t$, and N denotes the number of blocks. Equivalently, we have $\mathbf{x}_i = \mathbf{E}_i^T\mathbf{x}$ and $\mathbf{A}_i = \mathbf{A}\mathbf{E}_i$, where $\mathbf{E}_i \in \mathbb{R}^{2N_t\times n_i}$, and each column of $\{\mathbf{E}_i\}$ is uniquely picked from the columns of the $2N_t \times 2N_t$ identity matrix. The next section will briefly review the PJ-ADMM-based SLP that can solve the above separable problem in parallel, followed by the proposition of the ADMM-SLPNet.

III. ADMM-SLPNet

A. Algorithm Review and Motivation for ADMM-SLPNet

After rearranging the original PM-SLP problem into the sum of multiple blocks, all blocks of the transmit signal can be updated in a parallel manner by leveraging PJ-ADMM, which is outlined in Algorithm 1 [9]. The detailed derivations of the iterative PJ-ADMM algorithm are omitted due to space limitations. In the iterative algorithm design, the penalty term is used to regularize the objective function, such that the CI constraints are penalized into the augmented Lagrangian function. The rationale behind this is that a CI-constrained SLP optimization problem can be rewritten as an easy-to-handle penalized problem that is less constrained or entirely unconstrained by incorporating the CI constraints into the objective function. The penalty parameter determines the severity of the penalty for the CI constraint violations. It also serves as a step size in the update of the Lagrangian multiplier. If the penalty parameter is sufficiently large, the constrained PM-SLP problem is well-approximated by the augmented Lagrangian function, therefore we can obtain the optimal solution to the original PM-SLP problem. The proximal terms are adopted to reduce the effect of instability of the transmit signal by limiting the distance between two consecutive iterations, where the proximal parameters determine the size of the proximity. The damping parameter is added to smooth the update of the Lagrangian multiplier.

It can be inferred from the above that a larger value of the penalty parameter and the damping parameter, or a smaller value of the proximal parameters, can enlarge the step size in each iteration. The convergence rate is therefore fundamentally affected by the aforementioned parameters. Although we have an intuition on the mechanism of the interaction between the parameters and convergence performance, there is unfortunately no general rule working on the adjustment of the parameters in ADMM for convergence acceleration. The commonly used experiential strategy in practice is monotonously increasing or decreasing the parameters from one iteration to the next. Despite its incremental performance improvement over constant parameters, this strategy is limited by the tied parameters and is lack of concrete theoretical foundations.

Unlike the model-driven method that develops the iterative algorithm to solve the well-modelled problem, the data-driven deep learning method treats the task as an end-to-end black box and trains the black box using a huge volume of labelled data. The network architecture and hyper-parameter design, as well as the labelled data acquisition, are however the bottlenecks of the data-driven method due to unexplainabiltity. The dilemma between model-driven and data-driven methods has not been resolved until the emergence of the model-driven deep learning method [11]–[13]. In the following, we will present such a framework for PM-SLP, i.e., the ADMM-SLPNet.[1]

B. Network Topology

The iterative PJ-ADMM algorithm tackles the PM-SLP problem by recursively updating the transmit signal. This procedure can be unrolled to a hierarchical machine via

[1]The proposed framework can also be used to solve the SB-SLP problem through the duality between the PM-SLP problem considered in this paper and the SB-SLP problem [10].

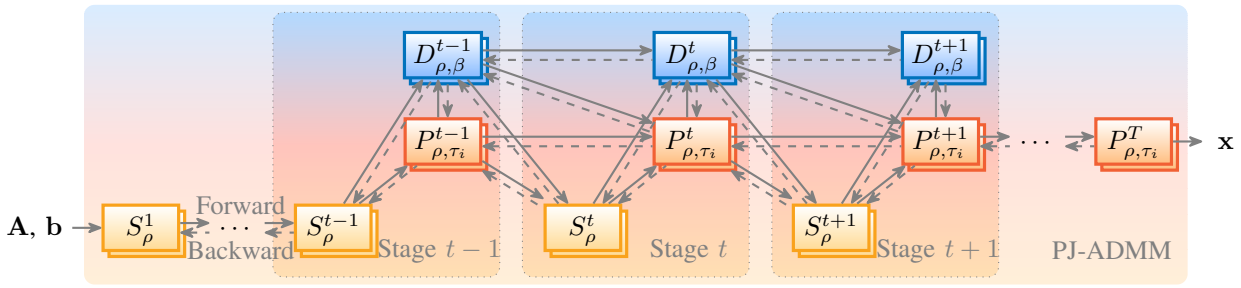

Fig. 2. The data flow graph of the iterative PJ-ADMM algorithm for PM-SLP.

Algorithm 1 Iterative PJ-ADMM Algorithm for the PM-SLP Problem (5) [9]

Input: $\mathbf{A}$, $\mathbf{b}$, ρ, τ_i, β
Output: $\mathbf{x}$

1: Initialize $\mathbf{x}_i^0$ $(i = 1, \cdots, N)$, and $\boldsymbol{\lambda}^0$;
2: Set $t \leftarrow 0$;
3: **repeat**
4: Update $\mathbf{c}^{t+1}$ by

$$\mathbf{c}^{t+1} = \max\left\{\sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i^t - \mathbf{b} - \frac{\boldsymbol{\lambda}^t}{\rho}, \mathbf{0}\right\}; \tag{7}$$

5: Update $\mathbf{x}_i^{t+1}$ for $i = 1, \cdots, N$ in parallel by:
6: **for** $i = 1, \cdots, N$ **do**
7: Update $\mathbf{x}_i^{t+1}$ by

$$\mathbf{x}_i^{t+1} = \left(2\mathbf{I} + \rho \mathbf{A}_i^T \mathbf{A}_i + \tau_i \mathbf{I}\right)^{-1} \left[\tau_i \mathbf{x}_i^t + \rho \mathbf{A}_i^T \left(-\sum_{j \neq i}^{N} \mathbf{A}_j \mathbf{x}_j^t + \mathbf{b} + \mathbf{c}^{t+1} + \frac{\boldsymbol{\lambda}^t}{\rho}\right)\right], \forall i; \tag{8}$$

8: **end for**
9: Update $\boldsymbol{\lambda}^{t+1}$ by

$$\boldsymbol{\lambda}^{t+1} = \boldsymbol{\lambda}^t + \rho\beta\left(-\sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i^{t+1} + \mathbf{b} + \mathbf{c}^{t+1}\right); \tag{9}$$

10: Set $t \leftarrow t + 1$;
11: **until** Convergence.

deep unfolding. To begin with, we unfold the operations in Algorithm 1 as a data flow graph using the concept of the directed acyclic graph. Each PJ-ADMM iteration is mapped to one stage in the data flow graph as shown in Fig. 2, where each stage comprises of three layers, i.e., the slack variable update layer, the primal variable update layer, and the dual variable update layer. Each layer consists of one node and directed edges that interconnect nodes, where the former corresponds to one operation in PJ-ADMM iteration, and the latter corresponds to data flows. Once the CSI and data symbols enter the data flow graph and flow over the layers, it will eventually output the desired transmit signal. The proposed ADMM-SLPNet can be obtained by treating the data flow graph as a learnable network. In order to train the network, we introduce trainable parameters to the unfolded network, which include the penalty parameter ρ in the slack variable update layer, the proximal parameters $\{\tau_i\}$ and penalty parameter ρ in the primal variable update layer, the damping parameter β and the penalty parameter ρ in the dual variable update layer. In such a way, the parameters in the iterative PJ-ADMM algorithm are untied to parameterize the unfolded network, which means that the value of the same learnable parameter can vary in different layers/stages. As opposed to the iterative algorithm, in which the parameters are either constant or monotonously changing, the unfolded network is more flexible thus powerful.

1) Slack Variable Update Layer: This layer updates the non-negative slack variable following the element-wise maximizing operation in (7). The output of this layer is given by

$$S_\rho^{t+1} : \mathbf{c}^{t+1} = \max\left\{\sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i^t - \mathbf{b} - \frac{\boldsymbol{\lambda}^t}{\rho}, \mathbf{0}\right\}. \tag{10}$$

2) Primal Variable Update Layer: This layer generates all N blocks of the transmit signal in parallel following the operation in (8). Given the slack variable and dual variable, this layer outputs the transmit signal by

$$P_{\rho,\tau_i}^{t+1} : \mathbf{x}_i^{t+1} = \left(2\mathbf{I} + \rho \mathbf{A}_i^T \mathbf{A}_i + \tau_i \mathbf{I}\right)^{-1} \left[\tau_i \mathbf{x}_i^t + \rho \mathbf{A}_i^T \left(-\sum_{j \neq i}^{N} \mathbf{A}_j \mathbf{x}_j^t + \mathbf{b} + \mathbf{c}^{t+1} + \frac{\boldsymbol{\lambda}^t}{\rho}\right)\right], \forall i. \tag{11}$$

3) Dual Variable Update Layer: This layer updates the Lagrangian multiplier by the gradient update procedure in (9). The updated dual variable can be expressed as

$$D_{\rho,\beta}^{t+1} : \boldsymbol{\lambda}^{t+1} = \boldsymbol{\lambda}^t + \rho\beta\left(-\sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i^{t+1} + \mathbf{b} + \mathbf{c}^{t+1}\right). \tag{12}$$

In the forward pass, CSI and data symbols flow over the network one after another stage and generate the transmit signal at the last stage. In the backward pass, the gradient of the loss function with respect to the variables and parameters in each stage can be computed inversely from the

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3301241

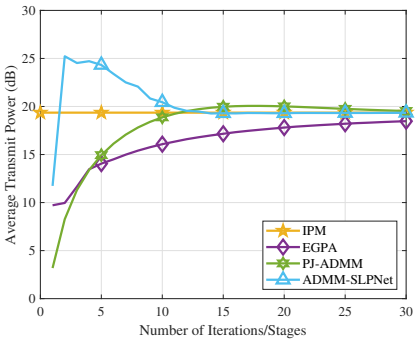

Fig. 3. Average transmit power versus number of iterations/stages, $N_t = 12$, $K = 12$, $N = 3$, γ = 12 dB, $N_c = 100$, $N_s = 20$, QPSK.

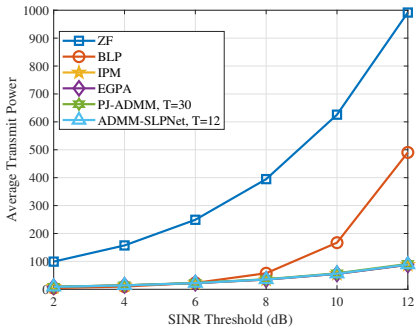

Fig. 4. Average transmit power versus SINR threshold, $N_t = 12$, $K = 12$, $N = 3$, $N_c = 100$, $N_s = 20$, QPSK.

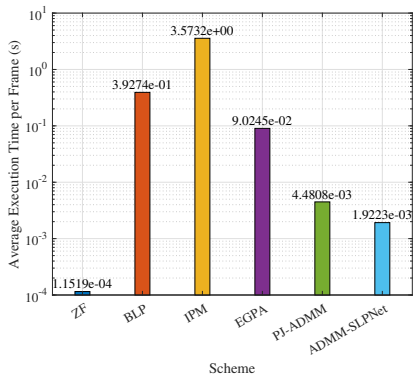

Fig. 5. Average execution time per frame, $N_t = 12$, $K = 12$, $N = 3$, γ = 12 dB, $N_c = 100$, $N_s = 20$, QPSK.

last stage to the first one using the chain rule. Consequently, the parameters can be trained based on the obtained gradient information. After designing the ADMM-SLPNet architecture and introducing learnable parameters, we will discriminatively train the network in the next subsection.

C. Unsupervised Training

The learnable parameters of the ADMM-SLPNet are typically trained by minimizing the loss function using the error back-propagation method, through which the value of the parameters in different layers can be discriminatively learned from the training dataset. Instead of choosing the supervised learning strategy in a wide range of existing deep unfolding methods to minimize the distance between the network output and the ground-truth solution, we propose an unsupervised learning approach for ADMM-SLPNet. The rational behind this is that the unsupervised learning not only discards data labelling but also has better generalization performance than supervised learning.

The loss function for the unsupervised learning is closely related to the objective function of the PM-SLP problem. It is observed that the updated primal variable in (8) may fall out of the feasible region defined by the CI constraints, which means that the feasibility of the transmit signal is not guaranteed. Furthermore, if we choose the objective function in (5) as the loss function of the network, then the transmit signal will go to zero, which leads to an infeasible solution that minimizes the loss function but badly violates the CI constraints. To circumvent this problem, we propose to project the network output in (8) onto its feasible region, i.e., the CI regions. The projected signal can be written as

$$\mathbf{x}_p\left(\Theta\right) \triangleq \mathbf{x}\left(\Theta\right) + \mathbf{A}^{-1} \max\left\{\mathbf{b} - \sum_{i=1}^{N} \mathbf{A}_i \mathbf{x}_i\left(\Theta\right), \mathbf{0}\right\}, \quad (13)$$

where $\mathbf{x}\left(\Theta\right)$ is the network output based on the parameter set $\Theta \triangleq \{\rho, \tau_i, \beta\}$. Then we choose the symbol-level transmit power of the projected signal as the loss function, which is given by

$$\mathbf{L}_\Theta \triangleq \left\|\mathbf{x}_p\left(\Theta\right)\right\|^2. \quad (14)$$

When the training data flow over the ADMM-SLPNet, there will be a transmit signal generated by the network. In the training phase, the gradient of the loss function with respect to the learnable parameters in each layer will be back-propagated along the network to train the parameters without the ground-truth transmit signal. It is worth noting that we compute and propagate the subgradient for the slack variable update layer containing element-wise maximizing operation.

So far, we have designed the ADMM-SLPNet by unrolling the iterative PJ-ADMM algorithm and training the network parameters by the proposed unsupervised training strategy. Along these lines, the unfolded network is trained by minimizing the loss function with respect to the parameters. The parameters are untied across layers, thus attaining a more powerful architecture. The next subsection will provide a brief analysis of the computational cost associated with the network.

D. Computational Cost

We evaluate the computational cost of the ADMM-SLPNet based on the number of real-valued multiplications. To reduce computational overhead, we pre-compute and store $\left\{\mathbf{A}_i^T \mathbf{A}_i\right\}$ for reusing in each iteration. Within one iteration, $\left\{\mathbf{A}_i \mathbf{x}_i\right\}$ and $\frac{\boldsymbol{\lambda}}{\rho}$ are also only computed once and stored. The computational complexity of the proposed ADMM-SLPNet includes two parts: the stage-independent part for $\left\{\mathbf{A}_i^T \mathbf{A}_i\right\}$, and the stage-dependent part, where the second part needs to be counted once per stage. The first part requires $8K^2N_t/N$ real-valued multiplications for each block. For the second part, the computation of $\left\{\mathbf{A}_i \mathbf{x}_i\right\}$ requires $4KN_t/N$ real-valued multiplications per block. Besides, (10), (11), and (12) require $2K$, $2N_t/N + 8N_t^3/N^3 + 2N_t/N + 2K + 4KN_t/N$, and $2K + 1$ real-valued multiplications for each block per stage, respectively. The total computational cost is approximately $8K^2N_t/N + T(8N_t^3/N^3 + 8KN_t/N + 4N_t/N + 6K + 1)$ real-valued multiplications for each block, where T denotes the number of stages.

IV. Numerical Results

We consider a single-cell downlink MU-MISO system employing QPSK signaling, where the channel vector $\tilde{\mathbf{h}}_k$ is assumed time varying as independent and identical distributed

(i.i.d.) $\mathcal{CN}(0, \mathbf{1})$. All K users have the same instantaneous SINR threshold and identical unit noise variance, i.e., $\gamma_k = \gamma$, $\sigma_k = 1$. Each transmission frame is assumed to composed of N_s = 20 symbol slots, within which the channel is static. The ADMM-SLPNet is trained using Adam optimizer with a dataset composed of 5000 random training samples, where $N_t = 12$, $K = 12$. The number of training epochs, batch size, and learning rate are set to 1000, 1, and 0.1, respectively. The benchmark schemes include the ZF precoding, the conventional interference mitigation PM-BLP [17], the IPM for PM-SLP implemented by CVX [18], the EGPA for PM-SLP [1], and the iterative PJ-ADMM algorithm for PM-SLP in Algorithm 1 [9]. We partition the original problem into $N = 3$ subproblems. The parameters for the iterative PJ-ADMM algorithm are carefully chosen from a lot of candidates, i.e., $\rho = 0.4$, $\beta = 1$, and $\tau_i = 0.8\rho\|\mathbf{A}_i\|^2$. The results in this section are averaged over $N_c = 100$ random channel realizations, i.e., 2000 symbol slots.

Fig. 3 demonstrates the convergence behavior of the proposed ADMM-SLPNet, where the average transmit power of the considered schemes is depicted as a function of the number of iterations/stages. It can be seen that the ADMM-SLPNet and PJ-ADMM require respectively $T = 12$ stages and $T = 30$ iterations to approximate the optimal average transmit power obtained by IPM. The model-driven deep learning framework shows significant convergence superiority over the corresponding iterative algorithm.

Fig. 4 provides comparisons of the average transmit power at various SINR thresholds. The number of stages for the ADMM-SLPNet is $T = 12$, and the number of iterations for the PJ-ADMM is $T = 30$, as obtained in Fig. 3. We can observe that the ADMM-SLPNet enables attaining the SLP gain at all SINR thresholds with fewer iterations/stages than the PJ-ADMM. The results in Fig. 4 validate the effectiveness of the proposed model-driven deep learning framework as well as the unsupervised learning approach for SLP.

Fig. 5 presents the average execution time per frame of the considered schemes in Fig. 4. The execution time for the ADMM-SLPNet and PJ-ADMM is averaged over $N = 3$. It is shown that the ADMM-SLPNet has the lowest execution time over the 4 compared SLP schemes. The average execution time of the ADMM-SLPNet with $T = 12$ stages is approximately 0.05%, 2.13%, 42.90%, and 4.89% of those of the IPM, EGPA, PJ-ADMM with $T = 30$, and BLP, respectively.

V. Conclusion

This paper developed the ADMM-SLPNet, a model-driven deep learning framework defined over the iterative PJ-ADMM SLP algorithm. Leveraging the deep unfolding to unroll the PJ-ADMM framework, we incorporated the problem-domain knowledge into the network topology and the loss function, thereby attaining interpretability for the ADMM-SLPNet. Besides, the domain knowledge-aided loss function is irrelevant to the optimal solutions to the training dataset, i.e., the data label. Moreover, the network was trained using an unsupervised learning approach with the proposed loss function, which eliminates the need for data labeling. Numerical results demonstrated the superiority of the proposed model-driven deep learning framework to achieve a better performance faster over the iterative algorithm.

References

[1] C. Masouros and G. Zheng, "Exploiting known interference as green signal power for downlink beamforming optimization," *IEEE Trans. Signal Process.*, vol. 63, no. 14, pp. 3628–3640, Jul. 2015.

[2] M. Alodeh, S. Chatzinotas, and B. Ottersten, "Constructive multiuser interference in symbol level precoding for the MISO downlink channel," *IEEE Trans. Signal Process.*, vol. 63, no. 9, pp. 2239–2252, May 2015.

[3] A. Li, D. Spano, J. Krivochiza, S. Domouchtsidis, C. G. Tsinos, C. Masouros, S. Chatzinotas, Y. Li, B. Vucetic, and B. Ottersten, "A tutorial on interference exploitation via symbol-level precoding: Overview, state-of-the-art and future directions," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 796–839, Secondquarter 2020.

[4] A. Wiesel, Y. Eldar, and S. Shamai, "Linear precoding via conic optimization for fixed MIMO receivers," *IEEE Trans. Signal Process.*, vol. 54, no. 1, pp. 161–176, Jan. 2006.

[5] A. Haqiqatnejad, F. Kayhan, and B. Ottersten, "Power minimizer symbol-level precoding: A closed-form suboptimal solution," *IEEE Signal Process. Lett.*, vol. 25, no. 11, pp. 1730–1734, Nov. 2018.

[6] A. Li and C. Masouros, "Interference exploitation precoding made practical: Optimal closed-form solutions for PSK modulations," *IEEE Trans. Wireless Commun.*, vol. 17, no. 11, pp. 7661–7676, Nov. 2018.

[7] A. Li, C. Masouros, B. Vucetic, Y. Li, and A. L. Swindlehurst, "Interference exploitation precoding for multi-level modulations: Closed-form solutions," *IEEE Trans. Commun.*, vol. 69, no. 1, pp. 291–308, Jan. 2021.

[8] A. Li, C. Shen, X. Liao, C. Masouros, and A. Lee Swindlehurst, "Practical interference exploitation precoding without symbol-by-symbol optimization: A block-level approach," *IEEE Trans. Wireless Commun.*, vol. 22, no. 6, pp. 3982–3996, Jun. 2023.

[9] J. Yang, A. Li, X. Liao, and C. Masouros, "Low complexity SLP: An inversion-free, parallelizable ADMM approach," *arXiv preprint arXiv:2209.12369*, Sep. 2022. [Online]. Available: https://arxiv.org/abs/2209.12369

[10] ——, "Speeding-up symbol-level precoding using separable and dual optimizations," *arXiv preprint arXiv:2211.14818*, Nov. 2022. [Online]. Available: https://arxiv.org/abs/2211.14818

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

[12] J. R. Hershey, J. L. Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," *arXiv preprint arXiv:1409.2574*, Sep. 2014. [Online]. Available: https://arxiv.org/abs/1409.2574

[13] Y. Yang, J. Sun, H. Li, and Z. Xu, "Deep ADMM-Net for compressive sensing MRI," in *Proc. Adv. Neural Inform. Process. Syst.*, vol. 29, 2016, pp. 10–18.

[14] C. Bertocchi, E. Chouzenoux, M.-C. Corbineau, J.-C. Pesquet, and M. Prato, "Deep unfolding of a proximal interior point method for image restoration," *Inverse Problems*, vol. 36, no. 3, p. 034005, Feb. 2020.

[15] A. Mohammad, C. Masouros, and Y. Andreopoulos, "An unsupervised deep unfolding framework for robust symbol level precoding," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 1075–1090, 2023.

[16] Z. Lei, X. Liao, Z. Gao, and A. Li, "CI-NN: A model-driven deep learning-based constructive interference precoding scheme," *IEEE Commun. Lett.*, vol. 25, no. 6, pp. 1896–1900, Jun. 2021.

[17] E. Björnson, M. Bengtsson, and B. Ottersten, "Optimal multiuser transmit beamforming: A difficult problem with a simple solution structure [lecture notes]," *IEEE Signal Process. Mag.*, vol. 31, no. 4, pp. 142–148, Jul. 2014.

[18] M. Grant and S. Boyd, *CVX: MATLAB software for disciplined convex programming, version 2.1*, Mar. 2014. [Online]. Available: http://cvxr.com/cvx