A Learning-only Method for Multi-Cell Multi-User MIMO Sum Rate Maximization

Qingyu Song[†], Juncheng Wang[‡], Jingzong Li[§], Guochen Liu[¶], Hong Xu[†]
[†]The Chinese University of Hong Kong, [‡]Hong Kong Baptist University, [§]City University of Hong Kong, [¶]Huawei

IEEE INFOCOM 2024 - IEEE Conference on Computer Communications | 979-8-3503-8350-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/INFOCOM52122.2024.10621282

***Abstract*—Solving the sum rate maximization problem for interference reduction in multi-cell multi-user multiple-input multiple-output (MIMO) wireless communication systems has been investigated for a decade. Several machine learning-assisted methods have been proposed under conventional sum rate maximization frameworks, such as the Weighted Minimum Mean Square Error (WMMSE) framework. However, existing learning-assisted methods suffer from a deficiency in parallelization, and their performance is intrinsically bounded by WMMSE. In contrast, we propose a structural learning-only framework from the abstraction of WMMSE. Our proposed framework increases the solvability of the original MIMO sum rate maximization problem by dimension expansion via a unitary learnable parameter matrix to create an equivalent problem in a higher dimension. We then propose a structural solution updating method to solve the higher dimensional problem, utilizing neural networks to generate the learnable matrix-multiplication parameters. We show that the proposed structural learning framework achieves lower complexity than WMMSE thanks to its parallel implementation. Simulation results under practical communication network settings demonstrate that our proposed learning-only framework achieves up to 98% optimality over state-of-the-art algorithms while providing up to 47$\times$ acceleration in various scenarios.**

I. Introduction

Multi-input and multi-output (MIMO) is a fundamental technique in current and future wireless communication systems [1], where multiple base stations (BSs) and multiple user equipment (UEs) transmit and receive data signals from each other with multiple antennas. In downlink data transmission, where signals are transmitted from the BSs to the UEs, one UE may receive signals transmitted to other UEs, causing inter-UE interference. A classical way to reduce such interference is to solve a sum rate maximization problem under a power budget, which requires beamforming design and transmit power allocation to maximize the total throughput of the UEs. However, the MIMO sum rate maximization problem has been demonstrated to be non-convex and NP-hard [1]. In the past decade, many conventional algorithms have been proposed for MIMO sum rate maximization, such as the zero-forcing precoding [2] and the gradient projection method [3].

In terms of optimality, an iterative algorithm known as the Weighted Minimum Mean Square Error (WMMSE) algorithm [4] is widely regarded as the state-of-the-art (SOTA) method. The WMMSE algorithm formulates an equivalent dual problem of the original MIMO sum rate maximization problem, by introducing two auxiliary beamforming variables. It then applies the standard block coordinate descent (BCD) method to iteratively update the auxiliary beamforming variables and a power allocation variable. However, WMMSE suffers from low efficiency as it generally requires a large number of iterations to converge, especially when the numbers of users and antennas are large [5]. Moreover, it is hard to accelerate WMMSE using modern parallel-computing hardware such as GPU since WMMSE requires serial variable updates.

This work is supported in part by funding from Huawei (CUHK #7010691).

Learning to optimize (L2O) has emerged as a promising solution to various optimization problems, by leveraging modern machine learning methods to improve the solutions' efficiency and optimality. L2O can improve efficiency since many matrix-multiplication-based methods, such as neural networks, are hardware-acceleratable with GPU or TPU. Furthermore, some studies on L2O theoretically show their optimality by approximating sophisticated matrix operations with learnable matrices [6]. Existing L2O works on MIMO sum rate maximization can be categorized as the *learning-assisted* approach and the *learning-only* approach.

The learning-assisted approach utilizes learning-based methods to approximate the time-consuming operations of WMMSE. For instance, Schynol and Pesavento [5] approximate the beamforming update steps in WMMSE with graph convolutional neural networks. Motivated by Taylor's expansion of matrix inversion, Hu et al. [7] employ two learnable parameter matrices to approximate each matrix inversion in WMMSE, which reduces the computational complexity by matrix multiplication. Both methods set the number of iterations in WMMSE as a pre-defined constant. Their numerical results achieve more than 90% optimality of WMMSE and demonstrate salient speedup, especially for large-scale problems with many users and antennas. Nonetheless, the learning-assisted methods inherit the same serial solving process of WMMSE (or other conventional optimization algorithms). Thus, the deficiency in parallelization still remains. In terms of optimality, their performance is intrinsically upper bounded by WMMSE.

The learning-only approach does not follow any existing algorithms' frameworks. It uses a black-box model to approximate the solutions directly. Various neural network techniques, such as vanilla deep neural network (DNN) [8] and graph neural networks (GNNs) [9, 10], have been employed to solve the MIMO sum rate maximization problem. In [8], distance-based metrics, such as mean-squared error and cosine similarity, are employed to directly approximate the solutions generated by a given conventional sum rate maximization algorithm. In [9, 10], negative sum rates are constructed as the loss function to achieve maximization. These learning-only methods enable parallel implementation, achieving hundreds to thousands of times acceleration with GPU [8, 9, 10].

However, all the above learning-only methods tackle a degenerated version of the original sum rate maximization problem, named the power allocation problem without beamforming design. Furthermore, these works focus on the single-input single-output (SISO) scenario where the BS and the UE are equipped with only one antenna. In this case, the decision variable space degenerates from a large complex number matrix space to a small real number scalar space. Applying their solutions to the original matrix-based optimization problem can lead to significant performance deterioration. For instance, in our experiments, the method proposed in [10] only achieves less than 20% optimality to WMMSE for MIMO sum rate maximization.

In this work, we take advantage of both the learning-assisted and learning-only approaches. We fix the number of solution iterations to reduce the run time similar to the learning-assisted models in [9, 10, 11]. Meanwhile, we propose a learning-only structural framework and a solution updating method based on the key insight drawn from the zero-forcing scheme [2].

The main contributions of this work are as follows:

- We propose an efficient structural learning framework to solve the MIMO sum rate maximization problem. Inspired by the WMMSE algorithm that constructs an equivalent dual problem by introducing two auxiliary parameter matrices, we introduce a learnable unitary matrix to construct an equivalent higher dimensional problem of the original sum rate maximization problem. Such a learnable unitary matrix improves the solvability of the original problem in high-dimensional space. After obtaining a higher dimensional precoding solution, we use the learnable unitary matrix to project it back to the original solution space.
- We propose a learning-only structural solution updating method to iteratively update our precoding solutions. We abstract the WMMSE algorithm updates [4] into two parameter matrices and construct a left and right matrix-multiplication-based solution updating module. Moreover, we propose a learning-based model named, Encoder-Decoder model, to efficiently learn the parameter matrices. We extract key features from the objective of the MIMO sum rate maximization problem, and use a message-passing neural network (MPNN) to coordinate and aggregate the features of all UEs. Our proposed solution updating method enables parallelization and achieves significantly lower computational complexity than WMMSE.
- Simulation results under practical communication network settings show that our proposed learning-only method achieves up to 98% optimality while significantly improves the computational efficiency with up to 47$\times$ speedup compared to the WMMSE algorithm. To the best of our knowledge, our proposed learning-only method is the first to achieve comparable optimality performance and much better acceleration than the current SOTA learning-assisted methods for MIMO sum rate maximization.

The rest of this paper is organized as follows. In Section II, we present the related work. Section III describes the system model and the MIMO sum rate maximization problem. In Section IV, we present our structural learning framework. Then, we discuss our structural solution updating method with computational complexity analysis in Section V. Simulation results are presented in Section VI, followed by concluding remarks in Section VII.

Notations: The Hermitian transpose, inverse, trace, and determinant of a matrix $\mathbf{A}$ are denoted by $\mathbf{A}^H$, $\mathbf{A}^{-1}$, $\mathrm{Tr}(\mathbf{A})$, and $\det(\mathbf{A})$, respectively. The Hadamard product between two matrices $\mathbf{A}$ and $\mathbf{B}$ is denoted by $\mathbf{A} \odot \mathbf{B}$. A positive semi-definite matrix is denoted as $\mathbf{A} \succeq \mathbf{0}$. The notation $\mathbf{I}$ denotes an identity matrix. For a vector $\mathbf{g}$, $\mathbf{g} \sim \mathcal{CN}(\mathbf{0}, \sigma^2\mathbf{I})$ means that $\mathbf{g}$ is a circular complex Gaussian random vector with mean $\mathbf{0}$ and variance $\sigma^2\mathbf{I}$. The complex space is denoted as $\mathbb{C}$.

II. Related Work

A. Conventional Frameworks

The studies of sum rate maximization problem have kept a decade. In the conventional optimization category, two flourishing methods are zero-forcing [2] and WMMSE [4]. The zero-forcing method utilizes singular value decomposition to construct orthogonal vectors to interference channel state information (CSI) matrices as solutions, and the optimality is achievable when there are enough BS antennas. The WMMSE algorithm directly constructs an equivalent weighted sum-MSE minimization problem with an MMSE receiver $\mathbf{U}$ and weight matrix $\mathbf{W}$ for each decision variable $\mathbf{V}$ [4]. The problem is convex in each of $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{V}$ [4]. Then the BCD method is sufficient to find a stationary solution [4].

B. Learning-Only Approach

Deep learning methods have emerged as a promising approach in recent years. One of the earliest approaches in 2018 [8] uses vanilla deep neural networks to solve a degenerated SISO power allocation problem. This approach has achieved over 90% optimality of the WMMSE algorithm while significantly reducing the execution time by hundreds of times. The model uses a black box architecture with an L_2-norm loss function to approximate the solutions of the WMMSE algorithm by training. For the same power allocation problem, several machine learning techniques are employed, such as assembling DNN and unsupervised learning [12], GNN with on heterogeneous graph [13], message passing neural networks with channel [9, 10], and reinforcement learning algorithms like Deep Deterministic Policy Gradient method and Deep Q-learning Network [14]. Notably, Shen et al. [9] demonstrate that the permutation equivalent GNN yields a distributed message passing algorithm, which is theoretically representable of the WMMSE algorithm. However, in our practice, such sophisticated neural network designs perform poorly on MIMO sum rate maximization.

C. Learning-Assisted Approach

Learning-assisted unfolding methods for solving matrix space optimization problems have been proposed in [5, 7, 11]. These works follow a consistent workflow that utilizes machine learning techniques to replace some components of a given algorithm to improve efficiency. For example, using recurrent neural networks to approximate the step size and gradient vectors in gradient projection algorithm [11], using multi-layer perceptions to approximate the Taylor decomposition of matrix inverse in WMMSE [7], and using graph convolutional neural networks to approximate the auxiliary variable updating in WMMSE [5]. The SOTA work in [5] can achieve 100% optimality and several times accelerations over WMMSE. However, its efficiency is still limited by WMMSE since solution variables are serially updated one after another.

III. MIMO Sum Rate Maximization

We consider a multi-cell multi-user system with B BSs, each equipped with N_t transmitting antennas. Each BS serves U UEs, each is equipped with N_r receiving antennas. Let $\mathcal{B} = \{1, \dots, B\}$ and $\mathcal{U} = \{1, \dots, U\}$. The CSI between UE u and BS b is denoted by matrix $\mathbf{H}_{b,u} \in \mathbb{C}^{N_r \times N_t}, b \in \mathcal{B}, u \in \mathcal{U}$, and the corresponding precoding matrix (decision variable) is denoted by $\mathbf{V}_{b,u} \in \mathbb{C}^{N_t \times N_r}$. Denote the signal sent from BS b to UE u as $\mathbf{s}_{b,u} \in \mathbb{C}^{N_r \times 1}$. The received signal at UE u served by BS b is given by

$$\mathbf{H}_{b,u}\mathbf{V}_{b,u}\mathbf{s}_{b,u} + \underbrace{\sum_{\tilde{u}\in\mathcal{U},\tilde{u}\neq u} \mathbf{H}_{b,u}\mathbf{V}_{b,\tilde{u}}\mathbf{s}_{b,\tilde{u}}}_{\text{intra-cell interference}} + \underbrace{\sum_{\tilde{b}\in\mathcal{B},\tilde{b}\neq b}\sum_{\tilde{u}\in\mathcal{U}} \mathbf{H}_{\tilde{b},u}\mathbf{V}_{\tilde{b},\tilde{u}}\mathbf{s}_{\tilde{b},\tilde{u}}}_{\text{inter-cell interference}} + n_{b,u}, \tag{1}$$

where the second term is the intra-cell interference caused by signals sent from BS b to the other UEs served by BS b, the third term is the inter-cell interference caused by signals sent from all other BSs $\tilde{b} \neq b$ to all other UEs, and $\mathbf{n}_{b,u} \in \mathcal{CN}\left(0, \sigma_{i_k}^2 \mathbf{I}\right)$ is the white Gaussian noise [4].

Our goal is to design the global precoding matrix $\mathbf{V} = \{\mathbf{V}_{b,u} \in \mathbb{C}^{N_t \times N_r}\}$ based on the global CSI $\mathbf{H} = \{\mathbf{H}_{b,u} \in \mathbb{C}^{N_r \times N_t}\}$, to maximize the sum data rate of all UEs, subject to per-BS maximum transmit power limits $P \in \mathbf{R}$. This leads to the following multi-cell multi-user MIMO sum rate maximization problem, given by

$$\begin{aligned}\mathbf{P}: \quad \max_{\mathbf{V}} \quad & \sum_{u\in\mathcal{U}}\sum_{b\in\mathcal{B}} \log_2 \det \Bigg(\mathbf{I} + \left(\mathbf{H}_{b,u}\mathbf{V}_{b,u}\right)\left(\mathbf{H}_{b,u}\mathbf{V}_{b,u}\right)^H \\ & \Bigg(\sum_{\substack{\tilde{b}\in\mathcal{B},\tilde{u}\in\mathcal{U}\\(\tilde{b},\tilde{u})\neq(b,u)}} \left(\mathbf{H}_{\tilde{b},u}\mathbf{V}_{\tilde{b},\tilde{u}}\right)\left(\mathbf{H}_{\tilde{b},u}\mathbf{V}_{\tilde{b},\tilde{u}}\right)^H + \sigma_{b,u}^2\mathbf{I}\Bigg)^{-1}\Bigg) \\ \text{s.t.} \quad & \sum_{u\in\mathcal{U}} \operatorname{Tr}\left(\mathbf{V}_{b,u}\mathbf{V}_{b,u}^H\right) \leq P, \quad b \in \mathcal{B}\end{aligned} \tag{2}$$

problem $\mathbf{P}$ is known to be non-convex, and NP-hard [1].

IV. Structural Learning Framework

In this section, we propose a general structural learning framework to solve the MIMO sum rate maximization problem $\mathbf{P}$. We first construct a higher dimensional problem by matrix multiplications on a learnable unitary matrix. Such a learnable unitary matrix improves the solvability of the original MIMO sum rate maximization problem $\mathbf{P}$ by introducing more learnable parameters in the framework. After obtaining the solution in higher dimensional space (to be discussed in Section V), we utilize the learnable unitary matrix to project it back to the original solution space.

A. Higher-Dimensional Feasibility-Equivalent Problem

When the total number of receiving antennas is fixed, the zero-forcing scheme tells that if more transmitting antennas are available, the MIMO sum rate maximization problem is more solvable since the BSs have more degrees of freedom to cancel interference. Inspired by this, we first construct a higher dimensional problem to the original MIMO sum rate maximization problem $\mathbf{P}$ by increasing the number of transmitting antennas at each BS $N_t' \geq N_t$. Denote $\{\mathbf{V}_{b,u}' \in \mathbb{C}^{N_t' \times N_r}\}$ as the new precoding variables in the higher dimensional space. Let $\mathbf{D} \in \mathbb{C}^{N_t \times N_t'}$ be a learnable unitary matrix with $\mathbf{D}\mathbf{D}^H = \mathbf{I}$. The following lemma shows that for any feasible precoding solution $\{\mathbf{V}_{b,u}'\}$ to $\mathbf{P}$ in the higher dimensional space, $\{\mathbf{D}\mathbf{V}_{b,u}'\}$ is a feasible precoding solution to $\mathbf{P}$ in the original space.

Lemma 1. *If* $\{\mathbf{V}_{b,u}', u \in \mathcal{U}, b \in \mathcal{B}\}$ *is a feasible solution to* $\boldsymbol{P}$*, i.e.,* $\sum_{u\in\mathcal{U}} \operatorname{tr}\left(\mathbf{V}_{b,u}'{\mathbf{V}_{b,u}'}^H\right) \leq P$*, then* $\{\mathbf{V}_{b,u} = \mathbf{D}\mathbf{V}_{b,u}' \in \mathbb{C}^{N_t \times N_r}\}$ *is also a feasible solution to* $\boldsymbol{P}$*.*

Proof: From the trace cyclic property and the von Neumann's trace inequality [15], we have

$$\begin{aligned}&\sum_{u\in\mathcal{U}} \operatorname{tr}\left(\mathbf{V}_{b,u}\mathbf{V}_{b,u}^H\right) = \sum_{u\in\mathcal{U}} \operatorname{tr}\left(\mathbf{D}\mathbf{V}_{b,u}'\mathbf{V}_{b,u}'^H\mathbf{D}^H\right) \\ =&\sum_{u\in\mathcal{U}} \operatorname{tr}\left(\mathbf{V}_{b,u}'\mathbf{V}_{b,u}'^H\mathbf{D}^H\mathbf{D}\right) \leq \sum_{u\in\mathcal{U}} \operatorname{tr}\left(\mathbf{V}_{b,u}'\mathbf{V}_{b,u}'^H\right)\operatorname{tr}\left(\mathbf{D}^H\mathbf{D}\right) \\ =&\sum_{u\in\mathcal{U}} \operatorname{tr}\left(\mathbf{V}_{b,u}'\mathbf{V}_{b,u}'^H\right)\operatorname{tr}\left(\mathbf{D}\mathbf{D}^H\right) = \sum_{u\in\mathcal{U}} \operatorname{tr}\left(\mathbf{V}_{b,u}'\mathbf{V}_{b,u}'^H\right) \leq P.\end{aligned}$$

■

Utilizing the results in Lemma 1, instead of directly finding the precoding solution to $\mathbf{P}$, we introduce more learnable parameters in our structural solution update framework to find a precoding solution to $\mathbf{P}$ in a higher dimensional space. Note that the learning ability of our framework also improves as the number of learnable parameters increases.

With the equivalent solution construction method, we propose a general learning framework. As shown in Figure 1, start from the original problem $\mathbf{P}$ to achieve initialization, we utilize the learnable matrix $\mathbf{D}$ to construct a new higher dimensional problem, named *Dimension Expansion*. We use a structural solution updating method (to be introduced in Section V) to solve the new problem. Then we project each iteration's solutions back to the original space of $\mathbf{P}$ with $\mathbf{D}$.

B. Dimension Expansion and Shrink

We construct a virtual new problem by increasing the dimensionality of CSI $\{\mathbf{H}'_{b,u}\}$ via the learnable unitary matrix $\mathbf{D}$. As introduced in Section IV-A, we obtain the solution of $\mathbf{P}$ in its original solution space by multiplying the learnable unitary matrix $\mathbf{D}$ with the higher dimensional solution $\{\mathbf{V}'_{b,u}\}$, i.e., $\{\mathbf{D}\mathbf{V}'_{b,u}\}$.

In the objective function of $\mathbf{P}$, every $\mathbf{V}_{b,u}$ is multiplied with a channel matrix, for example $\mathbf{H}_{b,u}\mathbf{D}\mathbf{V}'_{b,u}$. Have another look at $\mathbf{H}_{b,u}\mathbf{D}\mathbf{V}'_{b,u}$, we combine $\mathbf{H}_{b,u}\mathbf{D}$ and regard it as a new channel matrix, denoted as $\mathbf{H}'_{b,u}$. Hence, $\mathbf{D}$ increases the dimensionality of $\mathbf{P}$ and gets a higher dimensional problem. Denote the new global CSI as $\{\mathbf{H}'_{b,u}\}$, we calculate $\{\mathbf{H}'_{b,u}\}$ by

$$\mathbf{H}'_{b,u} = \mathbf{H}_{b,u}\mathbf{D}, u \in \mathcal{U}, b \in \mathcal{B}. \tag{3}$$

For the initialization of the new problem, we propose to initialize from the feasible solution of the original problem $\mathbf{P}$. Denote the initial solution of the new problem as $\{\mathbf{V}^{0'}_{b,u}\}$. Suppose we have an initial solution $\{\mathbf{V}^{0}_{b,u}\}$ to $\mathbf{P}$, we utilize the learnable unitary matrix $\mathbf{D}$ to expand it to a higher dimensional space by

$$\mathbf{V}^{0'}_{b,u} = \mathbf{D}^H\mathbf{V}^{0}_{b,u}, u \in \mathcal{U}, b \in \mathcal{B}. \tag{4}$$

There are many strategies to generate an initial solution to the original problem $\mathbf{P}$, such as the normalized maximum-ratio combining method [5] and the zero-forcing scheme [2]. Then, we construct a higher dimensional problem with $\{\mathbf{H}'\}$ and $\{\mathbf{V}^{0\prime}\}$. However, since $\mathbf{D}^H\mathbf{D}$ is not unitary by nature, the calculation will violate the power constraints. We add a normalization process to protect the violated solutions to the boundary of the transmit power constraints of $\mathbf{P}$ by

$$\mathbf{V}_{b,u} = \frac{\mathbf{V}_{b,u}}{\sum_{u\in\mathcal{U}}\sqrt{\mathrm{tr}(\mathbf{V}_{b,u}\mathbf{V}^H_{b,u})}} \cdot \sqrt{P}, u \in \mathcal{U}, b \in \mathcal{B}. \tag{5}$$

After solving the problem in higher dimensional space, we apply our proposed feasible solution construction method in Section IV-A to construct a feasible solution to $\mathbf{P}$ by

$$\mathbf{V}_{b,u} = \mathbf{D}\mathbf{V}'_{b,u}, u \in \mathcal{U}, b \in \mathcal{B}. \tag{6}$$

C. Solving Problem in Higher Dimensional Space

After the Dimension Expansion, we solve an equivalent higher dimensional problem. Since our learnable matrix $\mathbf{D}$ should be trained with a specific algorithm, any differentiable methods can be employed. Both conventional algorithms and learning-based are applicable. For example, a learning-assisted WMMSE algorithm named GCNWMMSE [5] is a feasible learning-based method. Non-differentiable methods, such as the zero-forcing scheme [2], can not be applied in our framework since the desired gradient value at singular vectors with zero singular values reaches infinity.

In the next section, we propose a structural solution updating method with higher efficiency than WMMSE. We utilize our proposed method as the update module to improve the solution in higher dimensional space for T iteration. In an iteration t, we propose an Encoder-Decoder model that takes the CSI $\mathbf{H}'$ and solution of the last iteration $\mathbf{V}'_{t-1}$ ($\mathbf{V}'_0$ for first iteration) as the input to generate two learnable parameter matrices $\mathbf{W}_L$ and $\mathbf{W}_R$. The Encoder-Decoder model will be introduced later. We update the solution as $\mathbf{V}^t = \mathbf{W}^t_L \cdot \mathbf{V}^{t-1} \cdot \mathbf{W}^t_R$. After each iteration, we apply the normalization method to keep the solution feasible.

V. Structural Solution Updating Method

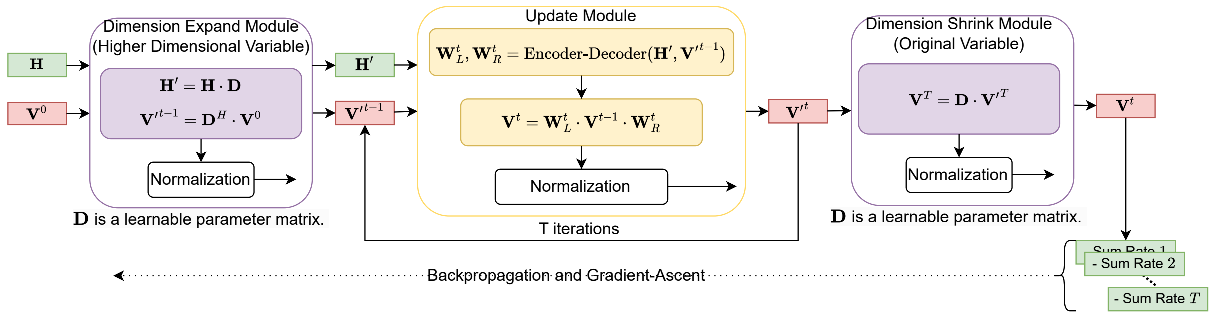
In this section, we present details of our structural solution updating method, which introduces two learnable parameter matrices to approximate the solution to $\mathbf{P}$. Our encoder-decoder extracts key features from the objective of $\mathbf{P}$ to train a MPNN for updating the two learnable parameter matrices. We further show that the proposed structural solution updating method achieves a lower computational complexity than WMMSE after parallelization.

A. Left and Right Learnable Parameter Matrices

Liu et al. [6] derive a solution framework for gradient descent-based LASSO regression problem from the first-order optimal condition on convergence. Then, a simple learnable parameter matrix generated by a long short-term memory (LSTM) model is employed to approximate sophisticated matrix operations. Our structural solution updating framework differentiates the one in [6] in several aspects. First, our framework introduces two tailored (left and right) learnable parameter matrices inspired by the WMMSE updates for solving $\mathbf{P}$. Second, we efficiently extract the key features from the objective function of $\mathbf{P}$. Third, we use a more powerful Encoder-Decoder model to update our learnable parameter matrices.

We first introduce the classic WMMSE solution updates, which inspire us to adopt two learnable parameter matrices to approximate the solution to $\mathbf{P}$. From Theorem 1 in [4], we can derive the following equivalent problem of $\mathbf{P}$ by introducing two auxiliary matrices $\mathbf{U} = \{\mathbf{U}_{b,u}\}$ and $\mathbf{W} = \{\mathbf{W}_{b,u} \succeq \mathbf{0}\}$:

$$\begin{aligned}\min_{\mathbf{W},\mathbf{U},\mathbf{V}} \quad & \sum_{u\in\mathcal{U}}\sum_{b\in\mathcal{B}} \left(\mathrm{Tr}\left(\mathbf{W}_{b,u}\mathbf{E}_{b,u}\right) - \log\det\left(\mathbf{W}_{b,u}\right)\right) \\ \text{s.t.} \quad & \sum_{u\in\mathcal{U}} \mathrm{Tr}\left(\mathbf{V}_{b,u}\mathbf{V}^H_{b,u}\right) \le P, \quad b \in \mathcal{B}\end{aligned}$$



Dimension Expand Module (Higher Dimensional Variable)
$\mathbf{H}$
$\mathbf{V}^0$
$\mathbf{H}' = \mathbf{H} \cdot \mathbf{D}$
$\mathbf{V}'^{t-1} = \mathbf{D}^H \cdot \mathbf{V}^0$
Normalization
$\mathbf{D}$ is a learnable parameter matrix.
$\mathbf{H}'$
$\mathbf{V}'^{t-1}$
Update Module
$\mathbf{W}_L^t, \mathbf{W}_R^t = \text{Encoder-Decoder}(\mathbf{H}', \mathbf{V}'^{t-1})$
$\mathbf{V}^t = \mathbf{W}_L^t \cdot \mathbf{V}^{t-1} \cdot \mathbf{W}_R^t$
Normalization
T iterations
$\mathbf{V}'^t$
Dimension Shrink Module (Original Variable)
$\mathbf{V}^T = \mathbf{D} \cdot \mathbf{V}'^T$
Normalization
$\mathbf{D}$ is a learnable parameter matrix.
$\mathbf{V}^t$
- Sum Rate 1
- Sum Rate 2
- Sum Rate T
Backpropagation and Gradient-Ascent

Fig. 1: Training workflow of our proposed model.

$$\underbrace{\Big(\sum_{(b,u)} \mathbf{H}_{b,u}^H \mathbf{U}_{b,u} \mathbf{W}_{b,u} \mathbf{U}_{b,u}^H \mathbf{H}_{b,u} + \mu_k^* \mathbf{I}\Big)^{-1} \mathbf{H}_{b,u}^H \Big(\sum_{(b,u)} \mathbf{H}_{b,u} \mathbf{V}_{b,u} \mathbf{V}_{b,u}^H \mathbf{H}_{b,u}^H + \sigma_{b,u}^2 \mathbf{I}\Big)^{-1} \mathbf{H}_{b,u}}_{①} \mathbf{V}_{b,u} \underbrace{\mathbf{W}_{b,u}}_{②}, \quad u \in \mathcal{U}, b \in \mathcal{B}. \tag{10}$$

where

$$\mathbf{E}_{b,u} = \left(\mathbf{I} - \mathbf{U}_{b,u}^H \mathbf{H}_{b,u} \mathbf{V}_{b,u}\right)\left(\mathbf{I} - \mathbf{U}_{b,u}^H \mathbf{H}_{b,u} \mathbf{V}_{b,u}\right)^H + \sum_{\substack{\tilde{b} \in \mathcal{B}, \tilde{u} \in \mathcal{U} \\ (\tilde{b},\tilde{u}) \neq (b,u)}} \mathbf{U}_{b,u} \mathbf{H}_{\tilde{b},u} \mathbf{V}_{\tilde{b},\tilde{u}} \mathbf{V}_{\tilde{b},\tilde{u}}^H \mathbf{H}_{\tilde{b},u}^H \mathbf{U}_{b,u}^H + \sigma_{b,u}^2 \mathbf{U}_{b,u}^H \mathbf{U}_{b,u}.$$

Then, $\{\mathbf{U}_{b,u}\}$, $\{\mathbf{W}_{b,u}\}$, and $\{\mathbf{V}_{b,u}\}$ are sequentially updated using the BCD method as follows:

$$\mathbf{U}_{b,u} = \Big(\sum_{(b,u)} \mathbf{H}_{b,u} \mathbf{V}_{b,u} \mathbf{V}_{b,u}^H \mathbf{H}_{b,u}^H + \sigma_{b,u}^2 \mathbf{I}\Big)^{-1} \mathbf{H}_{b,u} \mathbf{V}_{b,u}, \tag{7}$$

$$\mathbf{W}_{b,u} = (\mathbf{I} - \mathbf{U}_{b,u}^H \mathbf{H}_{b,u} \mathbf{V}_{b,u})^{-1}, \tag{8}$$

$$\mathbf{V}_{b,u} = \Big(\sum_{(b,u)} \mathbf{H}_{b,u}^H \mathbf{U}_{b,u} \mathbf{W}_{b,u} \mathbf{U}_{b,u}^H \mathbf{H}_{b,u} + \mu_k^* \mathbf{I}\Big)^{-1} \mathbf{H}_{b,u}^H \mathbf{U}_{b,u} \mathbf{W}_{b,u}, \tag{9}$$

where μ_k^* is the optimal solution of the Lagrangian multiplier.

Note that the BCD method sequentially updates $\mathbf{U}_{b,u}$, $\mathbf{W}_{b,u}$, and $\mathbf{V}_{b,u}$ in each iteration. In contrast, our solution directly learns and updates $\mathbf{V}_{b,u}$ in parallel among UEs to save the run time. Substitute $\mathbf{U}_{b,u}$ in (7) and $\mathbf{W}_{b,u}$ in (8) to the equation of $\mathbf{V}_{b,u}$ in (9), we have a complete formulation of $\mathbf{V}_{b,u}$-update in equation (10). Theorem 1 in [6] shows that matrix operations are representable by a single parameter matrix as a preconditioner with some constraints. Hence, we utilize a left learnable parameter matrix $\mathbf{W}_L$ to represent the operations in formula ① before $\mathbf{V}_{b,u}$ and another right learnable parameter matrix $\mathbf{W}_R$ after $\mathbf{V}_{b,u}$ to represent the operations in formula ②. We then have a new learnable $\mathbf{V}_{b,u}$-update formulation $\mathbf{W}_L \mathbf{V}_{b,u} \mathbf{W}_R$. Other algorithms are also representable with such formulation [6]. For example, we can update $\mathbf{V}_{b,u}$ via gradient descent by letting $\mathbf{W}_L = (\mathbf{I} + \alpha \nabla f(\mathbf{V}_{b,u}) \mathbf{V}_{b,u}^+)$ and $\mathbf{W}_R = \mathbf{I}$, where α is a gradient descent step size, $\nabla f(\mathbf{V}_{b,u})$ is the gradient of $\mathbf{V}_{b,u}$ in $\mathbf{P}$, and $\mathbf{V}_{b,u}^+$ is the pseudoinverse of $\mathbf{V}_{b,u}$.

Although WMMSE is guaranteed by the classic convergence theory of the BCD method to converge to a stationary point of the original sum rate maximization problem $\mathbf{P}$, its iterative and serial updates are computationally costly. Modern parallel computing toolkits, such as CUDA [16], support matrix operation accelerations and can execute independent calculations in parallel [17]. However, in WMMSE, $\mathbf{U}_{b,u}$, $\mathbf{W}_{b,u}$, and $\mathbf{V}_{b,u}$ are serially updated. We propose an efficient learning-based Encoder-Decoder model in the next section.

B. Encoder-Decoder Model

Several efficient learning-only architectures using modern neural networks have been proposed [9, 10, 13]. For instance, Shen et al. [10] propose a communication-based feature transmission method among the UEs and the BSs using several MPNNs [18]. However, it requires massive communications on high dimensional feature vectors and redundant CSI $\{\mathbf{H}\}$ among UEs. We employ the Encoder-Decoder structure to achieve an efficient model with fewer communications, where we only transmit $\{\mathbf{V}\}$ before encoding and the feature vectors with refined information after encoding in each iteration. Moreover, we apply an extra communication to share $\{\mathbf{H}\}$ among UEs.

The overall architecture of our proposed Encoder-Decoder model is shown in Figure 2. In the following, we introduce the detailed workflow from the perspective of one UE (b,u). The encoder first takes the input features from the objective function in $\mathbf{P}$ and outputs a feature vector with integrated information from the $\mathbf{V}$ and $\mathbf{HV}$ models (to be introduced in Section V-C). Then, the decoder exchanges the feature vector with an MPNN and combines the feature of UE (b,u) and the features of all other UEs. After that, we use two linear models to generate $\mathbf{W}_L$ and $\mathbf{W}_R$ separately.

Note that the Encoder-Decoder model is constructed as a multi-agent system with message passing and aggregation, where the solution updates of different UEs are executed in parallel. We further analyze the computational complexity of our solution and compare it with WMMSE in Section V-D.

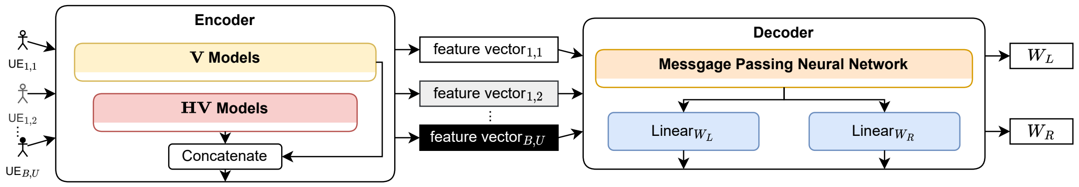
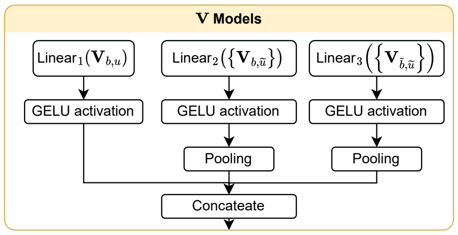
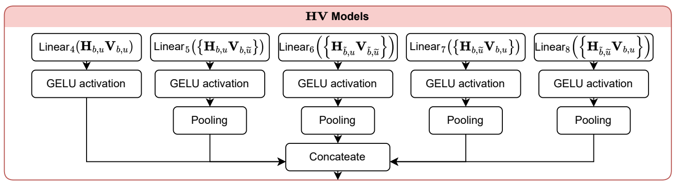


Fig. 2: $\mathbf{W}_L$, $\mathbf{W}_R$ generation by Encoder-Decoder model.



$\mathbf{V}$ Models
Linear$_1(\mathbf{V}_{b,u})$
Linear$_2(\{\mathbf{V}_{b,\tilde{u}}\})$
Linear$_3(\{\mathbf{V}_{\tilde{b},\tilde{u}}\})$
GELU activation
GELU activation
GELU activation
Pooling
Pooling
Concateate

Fig. 3: $\mathbf{V}$ models' architecture.



$\mathbf{HV}$ Models
Linear$_4(\mathbf{H}_{b,u}\mathbf{V}_{b,u})$
Linear$_5(\{\mathbf{H}_{b,u}\mathbf{V}_{b,\tilde{u}}\})$
Linear$_6(\{\mathbf{H}_{\tilde{b},u}\mathbf{V}_{\tilde{b},\tilde{u}}\})$
Linear$_7(\{\mathbf{H}_{b,\tilde{u}}\mathbf{V}_{b,u}\})$
Linear$_8(\{\mathbf{H}_{\tilde{b},\tilde{u}}\mathbf{V}_{b,u}\})$
GELU activation
Pooling
Concateate

Fig. 4: $\mathbf{HV}$ models' architecture.

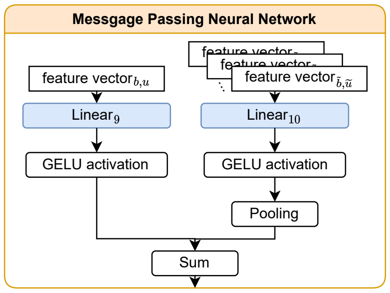


Fig. 5: Message Passing Neural Network's architecture.

C. Feature Extraction Module Design

Our encoder is under a hierarchical architecture, employing several linear models to encode the raw channel and precoding information. Two kinds of features are selected from the objective function of $\mathbf{P}$, i.e., the $\mathbf{V}$ feature and the $\mathbf{HV}$ feature. We eliminate the $\{(\mathbf{HV})(\mathbf{HV})^H\}$ feature from $\mathbf{P}$ to improve efficiency. The output dimensions of encoding models are set as multiples of a fixed integer of the corresponding inputs. For example, the output dimensions for each $\mathbf{HV}$ feature module $4N_t^2$ is four times its input dimension in a flattened vector shape. The expanding ratio is set as two for the $\mathbf{V}$ feature module, i.e., the output dimension $2N_tN_r$ is two times its input dimension N_tN_r.

Note that the output feature vectors are the concatenation of the two feature modules' output vectors. Such an encoding process enables parallel feature extraction among all UEs. The output of the encoder is feature vectors for all UEs. In the following, we introduce the two feature module designs and the MPNN architecture from the perspective of a single UE (b, u).

a) $\mathbf{V}$ *Feature Module:* As in shown Figure 3, we choose three types of $\mathbf{V}$:

- Precoding $\mathbf{V}_{b,u}$, the desired precoding at BS b to serve UE u.
- Intra-cell precoding $\{\mathbf{V}_{b,\tilde{u}}|\tilde{u} \in \mathcal{U}; \tilde{u} \neq u\}$, the precoding at BS b to other UEs $\tilde{u} \in \mathcal{U}$.
- Inter-cell precoding $\{\mathbf{V}_{\tilde{b},\tilde{u}}|\tilde{u} \in \mathcal{U}, \tilde{b} \in \mathcal{B}, \tilde{b} \neq b\}$, the precoding at all other BSs $\tilde{b} \in \mathcal{B}$ to serve all other UEs $(\tilde{b}, \tilde{u}) \in \mathcal{B}, \mathcal{U}$.

b) $\mathbf{HV}$ *Feature Module:* We take a similar workflow to select features for $\mathbf{HV}$, as shown in Figure 4. We consider two characteristics of each UE, i.e., a selfish UE that only considers information related to the data rate itself and a moral UE that further considers its effect on other UEs. In total, there are five types of $\mathbf{HV}$:

- $\mathbf{H}_{b,u}\mathbf{V}_{b,u}$, the desired signals transmitted from BS b to serve UE u.
- Intra-cell interference $\{\mathbf{H}_{b,\tilde{u}}\mathbf{V}_{b,\tilde{u}}|\tilde{u} \in \mathcal{U}, \tilde{u} \neq u\}$, the signal transmitted from BS b to other UEs $\tilde{u} \in \mathcal{U}$.
- Inter-cell interference $\{\mathbf{H}_{\tilde{b},u}\mathbf{V}_{\tilde{b},\tilde{u}}|\tilde{u} \in \mathcal{U}, \tilde{b} \in \mathcal{B}, \tilde{b} \neq b\}$, the signal transmitted from all other BSs $\tilde{b} \in \mathcal{B}$ to serve all other UEs $(\tilde{b}, \tilde{u}) \in \mathcal{B}, \mathcal{U}$.
- Interference to intra-cell UEs $\{\mathbf{H}_{b,\tilde{u}}\mathbf{V}_{b,u}|\tilde{u} \in \mathcal{U}, \tilde{u} \neq u\}$, the signal transmitted from BS b to UE u that interfering other UEs $\tilde{u} \in \mathcal{U}$.
- Interference to inter-cell UEs $\{\mathbf{H}_{\tilde{b},\tilde{u}}\mathbf{V}_{b,u}|\tilde{u} \in \mathcal{U}, \tilde{u} \neq u\}$, the signal transmitted from BS b to UE u that interfering other UEs $\tilde{u} \in \mathcal{U}$ at BSs $\tilde{b} \in \mathcal{B}$.

c) MPNN Structure: In the encoder, we mix all related information of one UE in $\mathbf{P}$ with the above two feature extraction models. However, in the MIMO sum rate maximization problem $\mathbf{P}$, all UEs compete with each other for beamforming and power resources to achieve a better throughput. An explicit coordinative strategy is necessary for the precoding solution to converge. On the contrary, the selfish single-user water-filing method without coordination fails to converge [19].

We facilitate coordination by utilizing an MPNN [18]. Our MPNN architecture is illustrated in Figure 5. From the perspective of a single UE, the MPNN collects the feature vectors from all other UEs, i.e., both intra-cell UEs and inter-cell UEs. Then, it aggregates the features with a mean-pooling method. Moreover, we utilize two fully connected layers with a GELU activation function to enhance such feature aggregation by learning. In the output layer, we construct two linear models to take the feature vectors of the encoder output as input and generate two vectors, respectively. We reshape the output vectors to generate the parameters of our proposed workflow, i.e., left matrix $\mathbf{W}_L$ and right matrix $\mathbf{W}_R$.

D. Computational Complexity Analysis

In this section, we demonstrate the efficiency of our proposed structural solution updating method by computational complexity analysis. We show that our method achieves lower computational complexity than WMMSE after parallelization.

As introduced in Section V-B, our structural solution updating method is based on neural networks. Therefore, we analyze the computational complexity of matrix multiplications and activation functions. As introduced above, our framework processes all the UEs' features in parallel. Such parallel computing reduces the time-consuming of multiple UEs to a unit computational complexity of one UE [17].

For the GELU activation function $\mathrm{GELU}(\mathbf{V}) = \mathbf{V} * \Phi(\mathbf{V})$, where Φ is the Cumulative Distribution Function of Gaussian Distribution [20], its empirical complexity is $\mathcal{O}(1)$. Counting the number of utilization in three modules in Figure 3, 4, and 5, the overall complexity is $\mathcal{O}(10)$. Then, we calculate the matrix multiplication complexities, including the $\mathbf{HV}$ calculation in feature pre-processing, all matrix multiplications in neural networks, and the calculation of $\mathbf{W}_L\mathbf{V}\mathbf{W}_R$. The results are listed in Table I.

We directly calculate the complexities of $\mathbf{HV}$ calculation in pre-processing period and complexities of $\mathbf{W}_L\mathbf{V}\mathbf{W}_R$ in the end of each iteration as $\mathcal{O}(N_tN_r^2)$ and $\mathcal{O}(N_t^2N_r + N_tN_r^2)$ respectively. Since processing multiple UEs within the neural network model is also concurrently executed, we count the complexity of one execution after such parallelization. For example, Linear 2 in $\mathbf{V}$ model of Figure 3 encodes the intra-cell UEs' $\{\mathbf{V}_{b,\tilde{u}}\}$ in parallel. The time-consuming is equivalent to encoding one $\mathbf{V}_{b,\tilde{u}}$.

As introduced in Section V-C, for the $\mathbf{V}$ feature module and $\mathbf{HV}$ feature module, we set the output dimensions to be integer multiples of their input dimensions. We denote the integers for $\mathbf{V}$ and $\mathbf{HV}$ as d_1 and d_2. For linear models 1, 2, and 3 in the $\mathbf{V}$ model, the input data is the flattened feature vector of $\mathbf{V}$. The overall complexity is $\mathcal{O}(3d_1N_t^2N_r^2)$. For five linear models in the $\mathbf{HV}$ model, the input data is the flattened $\mathbf{HV}$. The overall complexity is $\mathcal{O}(5d_2N_r^4)$.

In the MPNN module, we take the concatenated feature vector of all outputs of $\mathbf{V}$ model and $\mathbf{HV}$ model and generate a feature vector with the same dimension as output. Notably, we have two linear models within the MPNN. The overall time complexity is $\mathcal{O}(18d_1^2N_t^2N_r^2 + 60d_1d_2N_tN_r^3 + 50d_2^2N_tN_r^4)$.

By summing up all results and merging similar terms in Table I, we get our proposed model's overall time complexity of one iteration as

$$\begin{aligned}&\mathcal{O}\big((3d_1N_r)N_t^3 + (18d_1^2N_r^2 + 5d_2N_r^2 + 3d_1N_r^2 + N_r)N_t^2 \\ &\quad + (60d_1d_2N_r^3 + 3d_1N_r^3 + 2N_r^2)N_t + (50d_2^2 + 10d_2)N_r^4\big).\end{aligned} \tag{10}$$

From [4], we get WMMSE's complexity of one iteration as

$$\mathcal{O}\big((BU)^2N_t^3 + (BU)^2N_rN_t^2 + (BU)^2N_r^2N_t + (BU)N_r^3\big). \tag{11}$$

We focus on the N_t terms and compare the coefficients of different terms. Our model achieves a constant time complexity increase with only respect to the number of receiving antennas. However, the time complexity of WMMSE exponentially increases with the number of BS and UE. Take the example of 2 BSs, 8 UEs, $N_t = 8$, and $N_r = 2$, the WMMSE has a $\mathcal{O}(172160)$ complexity while our model achieves $\mathcal{O}(64212)$ complexity with $d_1 = 2, d_2 = 4$. Our model achieves 2.7$\times$ speedup in this case. Considering a large-scale problem with 2 BSs, 32 UEs, $N_t = 32$, and $N_r = 2$, the speedup will increase to 92$\times$ with $d_1 = 2, d_2 = 16$.

VI. Simulation

In this section, we numerically evaluate our proposed structural learning framework in section IV and our proposed solution updating method in section V. We conduct experiments on Python 3.9 and PyTorch 1.10.0 on Ubuntu 18.04 with 128GB memory, Intel Xeon Gold 5320 CPU @ 2.2 GHz, and one NVIDIA RTX 3090 GPU.

A. Simulation Configurations

We construct three scenarios of problem $\mathbf{P}$ with different numbers of UEs U and different numbers of transmitting antennas N_t, including small scale, moderate scale, and large scale. The detailed settings are shown in Table II. We split the datasets with a ratio of 8:1:1 for training, evaluation, and testing. The evaluation dataset is for high-parameter tuning. We evaluate performance on the testing set.

To generate data (CSI $\{\mathbf{H}_{b,u}\}$ and $\sigma_{b,u}$ in $\mathbf{P}$), we follow the configurations outlined in the latest WMMSE paper [21]. We first generate the coordinates of BSs in a cellular network. We set the first BS at the origin as the central BS. Then, we set the following BSs at the center of cellulars around the central BS. We consider two scenarios with distinct inter-cell interference levels based on inter-BS distances. We set the inter-BS distance for the small inter-cell interference scenario as 2.8 km [21]. For the scenario with large inter-cell interference, we set it as 0.5 km. We randomly sample U coordinates within the $[0.1, 0.3]$ km range of each base station as the positions of user equipment.

We follow [21] to calculate $\{\mathbf{H}_{b,u}\}$. We randomly sample UE coordinates within the range of 0.1 km to 0.3 km to the BS that serves it. We apply a path loss model upon circularly-symmetric standard complex Gaussian distribution to generate $\{\mathbf{H}_{b,u}\}$. Then, we calculate the distance between UE (b,u) to BS b as $\omega_{b,u}$ and path loss attenuation is calculated by $G_{b,u} = 128.1 + 37.6\log_{10}(\omega_{b,u})$[dB]. The path loss is linearly applied by $10^{-G_{b,u}/20} \odot \mathbf{H}_{u,b}$.

We set the noise power for each BS to be correlated with target CSI $\mathbf{H}_{b,u}, u \in \mathcal{U}$ with a constant signal-to-noise value (SNR). The noise power on each BS b is calculated by $\sigma_b^2 = 10^{\frac{1}{U}\sum_u \log_{10}\frac{1}{N_r}\|\mathbf{H}_{b,u}\|_F^2} \times 10^{-\frac{\mathrm{SNR}}{10}}$ [21], where the SNR is set as 5[dB]. The maximum transmit power limit in $\mathbf{P}$ is set as $P = 10$[W].

TABLE I: Layer wise matrix multiplication complexities of one iteration in our structural solution updating method.

Module & Operation	Input Dimension	Output Dimension	Time Complexity
$\mathbf{HV}$	–	–	$\mathcal{O}(N_t N_r^2)$
$\mathbf{V}$ Linear 1 2 3	$N_t * N_r$	$N_t * N_r * d_1$	$\mathcal{O}(3d_1 N_t^2 N_r^2)$
$\mathbf{HV}$ Linear 1 2 3 4 5	$N_r * N_r$	$N_r * N_r * d_2$	$\mathcal{O}(5d_2 N_r^4)$
MPNN Module	$N_t * N_r * d_1 * 3 + N_r^2 * d_2 * 5$	$N_t * N_r * d_1 * 3 + N_r^2 * d_2 * 5$	$\mathcal{O}(18d_1^2 N_t^2 N_r^2 + 60d_1 d_2 N_t N_r^3 + 50d_2^2 N_r^4)$
Linear $\mathbf{W}_L$	$N_t * N_r * d_1 * 3 + N_r^2 * d_2 * 5$	$N_t * N_t$	$\mathcal{O}(3d_1 N_t^3 N_r + 5d_2 N_t^2 N_r^2)$
Linear $\mathbf{W}_R$	$N_t * N_r * d_1 * 3 + N_r^2 * d_2 * 5$	$N_r * N_r$	$\mathcal{O}(3d_1 N_t N_r^3 + 5d_2 N_r^4)$
$\mathbf{W}_L \cdot \mathbf{V} \cdot \mathbf{W}_R$	–	–	$\mathcal{O}(N_t^2 N_r + N_t N_r^2)$

TABLE II: Scenarios Settings

Properties	Small Scale	Moderate Scale	Large Scale
B	2	2	2
U	4	8	16
N_t	8	16	32
N_r	2	2	2
Data Size	100,000	120,000	180,000

B. Implementations

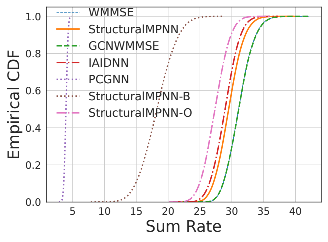
Since the CSI matrices $\{\mathbf{H}_{b,u}\}$ and the precoding matrices $\{\mathbf{V}_{b,u}\}$ in problem $\mathbf{P}$ are in complex domain. We implement complex-number neural networks based on an open-source code base developed in [22]. We implement two linear models for each complex-number tensor for its real and imaginary parts. We follow the complex number algebra to manipulate the outputs, feeding the input tensor's real and imaginary parts into the 'real' model and summing up their outputs as real parts. We use a similar method to generate the output's imaginary part by the 'imaginary' model. In our structural framework, we utilize one structural solution updating model to update $\{\mathbf{V}_{b,u}\}$ for five iterations.

In our structural solution updating model, we incorporate the constant parameter σ into $\mathbf{H}$ to avoid superfluous inputs in neural networks. Based on the noise power configuration in section VI-A, the UEs from the same BS have an identical noise power value. We use this value to normalize each $\mathbf{H}_{b,u}$ on the same base station by $\mathbf{H}_{b,u}/\sigma_b$. This manipulation keeps the same objective value in $\mathbf{P}$ and slightly increases the value of $\mathbf{H}_{b,u}$ to avoid numeric overflow.

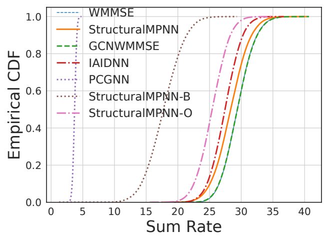
We evaluate the performances of following methods:

1) **StructualMPNN**, our proposed model with the structural framework and structural solution updating method. We train our model for 400 epochs. We set different values for the multiplier d_1 in the $\mathbf{V}$ feature module and the multiplier d_2 in the $\mathbf{HV}$ feature module. They are set as 2 and 4 for small-scale and moderate-scale scenarios and 2 and 16 for the large-scale scenario. We run five iterations and share parameters among them.
2) WMMSE [4], SOTA conventional optimization algorithm. We set the stopping threshold to 0.01 and the max iterations to 50. We use a zero-forcing scheme and identical power allocation method to get initial $\{\mathbf{V}_{b,u}\}$.
3) GCNWMMSE [5], SOTA learning-assisted WMMSE. It utilizes graph convolutional networks (GCN) to approximate weight $\{\mathbf{W}_{b,u}\}$ calculation and precoding $\{\mathbf{V}_{b,u}\}$ calculation. However, the official implementation only utilizes GCN to approximate the inversion in $\{\mathbf{V}_{b,u}\}$, which is almost WMMSE.
4) IAIDNN [7], learning-assisted WMMSE. It utilizes learnable parameter matrix to approximate matrix inversion.
5) PCGNN [10], an MPNN-based learning-only method. Note that the original model is only for SISO power allocation. We implement a MIMO version model following the HetGNN model in [10]. We add an extra message-passing flow from UEs to BSs. The overall message-passing flow is UEs $\rightarrow$ BSs $\rightarrow$ UEs $\rightarrow$ UEs.
6) StructualMPNN-B. We replace our proposed left-right matrix-multiplication-based solution updating method with a black-box neural network.
7) StructualMPNN-O. We remove our proposed learnable unitary matrix, where we directly solve the original problem $\mathbf{P}$ with our structural solution updating method.

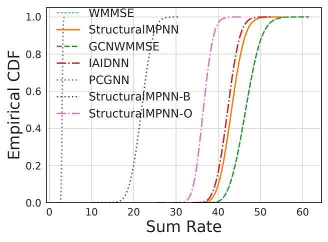
We implement GCNWMMSE and IAIDNN with 32-bit float numbers and train them for 100 epochs, which spends a similar time cost to training our model for 400 epochs. Both models update $\{\mathbf{V}_{b,u}\}$ for seven iterations and use different parameters in different iterations. We set the loss function as the summation of sum rates in all iterations. The mini-batches for training our model in three scales are 256, 512, and 512. The basic training configurations are in Table III.



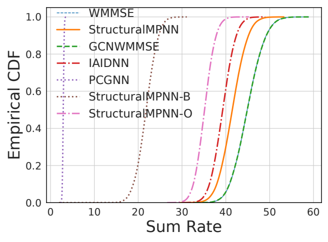
(a) Small, Small Interference



(b) Small, Large Interference



(c) Small, Small Interference



(d) Small, Large Interference

Fig. 6: Empirical cumulative distribution function of sum rate over WMMSE.

TABLE III: Training Configurations

Item	**Value**
Optimizer	AdamW [23], momentum: [0.9, 0.999]
Initial Learning Rate	0.001
Learning Rate Decay	Cosine, minimal: 0.00001
Weight Decay (L_1-norm) Rate	0.01

TABLE IV: Sum rate over WMMSE.

Models	Small		Moderate		Large	
StructuralMPNN	98.6%	96.9%	96.9%	95.7%	93.0%	91.5%
GCNWMMSE[5]	100.4%	100.5%	100.2%	100.5%	100.2%	100.3%
IAIDNN[7]	93.2%	94.0%	91.2%	87.9%	92.0%	89.7%
PCGNN[10]	12.7%	12.7%	6.5%	6.5%	-	
StructualMPNN-B	58.7%	60.7%	46.9%	48.7%	-	
StructualMPNN-O	88.3%	86.8%	78.8%	78.6%	-	

C. *Overall Optimality and Efficiency Comparison*

We first compare the sum rate yielded by our method with WMMSE and other baselines. We plot the empirical cumulative distribution functions of the sum rates on the testing sets. As shown in Figure 6, our methods are close to WMMSE. We further collect the sum rates of different methods over the sum rates of WMMSE. As listed in Table IV, in small-scale scenarios, our methods achieve up to 98% optimality of WMMSE. Although our method is slightly worse than GCNWMMSE on accuracy, its efficiency performance shown below significantly outperforms all baselines. Moreover, the deficiencies in StructuralMPNN-B and StructuralMPNN-O demonstrate the effectiveness of our design. The learning-only PCGNN for SISO fails to solve the MIMO problem.

To evaluate the empirical efficiency of our proposed method, we quantify the execution time of serially solving one hundred randomly generated problems $\mathbf{P}$ with different CSI $\{\mathbf{H}\}$. We collect the time costs of solving the problems on CPU and GPU and normalize them by the time costs of WMMSE to get the relative acceleration over WMMSE. The results on CPU and GPU are shown in Table V and Table VI. Our proposed method outperforms GCNWMMSE and IAIDNN, achieving 47× acceleration on GPU. Although learning-assisted algorithms achieve 100% optimality of WMMSE, they suffer low efficiencies on small-scale problems, whose performance deteriorates on GPU.

TABLE V: Speedup over WMMSE on CPU.

Scale:	Small		Moderate		Large	
Interference:	Small	Large	Small	Large	Small	Large
StructuralMPNN	**3.16**	**4.21**	**4.5**	**4.2**	**6.62**	**6.83**
GCNWMMSE[5]	0.76	1.04	1.16	1.07	3.27	3.33
IAIDNN[7]	1.01	1.39	1.47	1.36	4.04	4.12

TABLE VI: Speedup over WMMSE on GPU.

Scale:	Small		Moderate		Large	
Interference:	Small	Large	Small	Large	Small	Large
StructuralMPNN	**3.11**	**4.26**	**8.45**	**7.77**	**46.82**	**47.85**
GCNWMMSE[5]	0.39	0.55	0.61	0.57	1.66	1.71
IAIDNN[7]	0.57	0.8	0.82	0.77	2.22	2.19

We further compare our proposed method with the truncated WMMSE algorithm. We run WMMSE for six iterations. The sum rates achieved by our method outperform WMMSE by 3.7% in small scale and 2.1% in moderate scale. Time costs are shown in Figure 7. Our method achieves around 1.1× acceleration on CPU and around 1.5× acceleration on GPU.

We further evaluate our proposed learnable unitary matrix in Section IV-A over other algorithms. We add our proposed learnable unitary parameter matrix to the IAIDNN [7]. We jointly train our learnable unitary matrix with IAIDNN. The sum rate curve of the evaluation set during training is shown in Figure 8, where our structural learning framework leads to better convergence than the original IAIDNN.

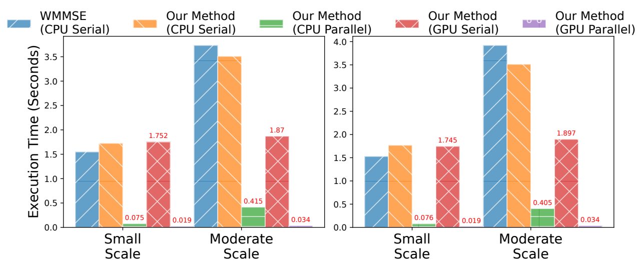


Fig. 7: Execution time (seconds) on small scale and moderate scale scenarios. 'Serial' represents serial execution. 'Parallel' represents parallel execution.

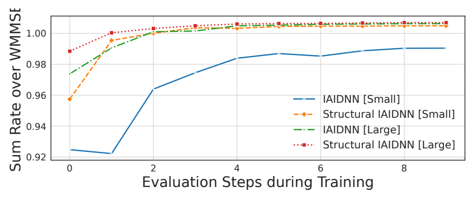


Fig. 8: Sum rate curve of IAIDNN [7] with our learnable unitary matrix.

VII. Conclusion

In this paper, we propose a structured learning-only framework to solve the multi-cell multi-user MIMO sum rate maximization problem. We propose a learnable unitary matrix to construct an equivalent higher-dimensional problem by learning. The learnable unitary matrix increases the solvability of the sum rate maximization problem. Our experimental results demonstrate that a learning-assisted WMMSE algorithm [5] can be improved to get faster convergence in training. Moreover, we propose another structural solution-updating method, where we make an abstraction of WMMSE [4] and utilize two learnable matrices to update each precoding in the left-right matrix-multiplication form. We further propose an Encoder-Decoder neural network model to generate such two learnable parameter matrices by learning. We propose several models to encode different features from the objective function and utilize an MPNN to achieve UE coordination. We demonstrate that our model is more efficient than the solution updating of WMMSE. We evaluate our proposed methods under various communication network settings. Our experimental results show up to 98% optimality of WMMSE and up to 47× acceleration over WMMSE.

References

[1] Z.-Q. Luo and S. Zhang, "Dynamic Spectrum Management: Complexity and Duality," *IEEE J. Sel. Areas Commun.*, vol. 2, no. 1, pp. 57–73, Feb. 2008.

[2] X. Gao, O. Edfors, F. Rusek, and F. Tufvesson, "Linear Pre-Coding Performance in Measured Very-Large MIMO Channels," in *2011 IEEE Vehicular Technology Conference (VTC Fall)*, 2011.

[3] Y.-F. Liu, Y.-H. Dai, and Z.-Q. Luo, "Coordinated Beamforming for MISO Interference Channel: Complexity Analysis and Efficient Algorithms," *IEEE Trans. Signal Process.*, vol. 59, no. 3, pp. 1142–1157, Mar. 2011.

[4] Q. Shi, M. Razaviyayn, Z. Q. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Trans. Wireless Commun.*, vol. 59, no. 9, pp. 4331–4340, Sep. 2011.

[5] L. Schynol and M. Pesavento, "Coordinated Sum-Rate Maximization in Multicell MU-MIMO With Deep Unrolling," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1120–1134, Apr. 2023.

[6] J. Liu, X. Chen, Z. Wang, W. Yin, and H. Cai, "Towards Constituting Mathematical Structures for Learning to Optimize," in *ICML*, 2023.

[7] Q. Hu, Y. Cai, Q. Shi, K. Xu, G. Yu, and Z. Ding, "Iterative Algorithm Induced Deep-Unfolding Neural Networks: Precoding Design for Multiuser MIMO Systems," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1394–1410, Feb. 2021.

[8] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5438–5453, Oct. 2018.

[9] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph Neural Networks for Scalable Radio Resource Management: Architecture Design and Theoretical Analysis," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 101–115, Jan. 2021.

[10] Y. Shen, J. Zhang, S. Song, and K. B. Letaief, "Graph Neural Networks for Wireless Communications: From Theory to Practice," *IEEE Trans. Wireless Commun.*, vol. 22, no. 5, pp. 3554–3569, May 2023.

[11] M. Zhu, T.-H. Chang, and M. Hong, "Learning to beamform in heterogeneous massive MIMO networks," *IEEE Trans. Wireless Commun.*, vol. 22, no. 7, pp. 4901–4915, July 2023.

[12] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards optimal power control via ensembling deep neural networks," *IEEE Trans. Comput.*, vol. 68, no. 3, pp. 1760–1776, Mar. 2019.

[13] J. Guo and C. Yang, "Learning power allocation for multi-cell-multi-user systems with heterogeneous graph neural networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 2, pp. 884–897, Feb. 2021.

[14] Y. Zhao, I. G. Niemegeers, and S. M. De Groot, "Dynamic Power Allocation for Cell-Free Massive MIMO: Deep Reinforcement Learning Methods," *IEEE Access*, vol. 9, pp. 102 953–102 965, 2021.

[15] L. Mirsky, "A trace inequality of John von Neumann," *Monatshefte für Mathematik*, vol. 79, pp. 303–306, Dec. 1975.

[16] M. Fatica, "CUDA toolkit and libraries," in *2008 IEEE Hot Chips 20 Symposium (HCS)*, 2008.

[17] Y. Ding, L. Zhu, Z. Jia, G. Pekhimenko, and S. Han, "Ios: Inter-operator scheduler for cnn acceleration," in *Proceedings of Machine Learning and Systems*, 2021.

[18] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *ICML*, 2017.

[19] K.-K. Wong, G. Liu, W. Cun, W. Zhang, M. Zhao, and Z. Zheng, "Truly Distributed Multicell Multi-Band Multiuser MIMO by Synergizing Game Theory and Deep Learning," *IEEE Access*, vol. 9, pp. 30 347–30 358, 2021.

[20] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[21] X. Zhao, S. Lu, Q. Shi, and Z.-Q. Luo, "Rethinking WMMSE: Can Its Complexity Scale Linearly With the Number of BS Antennas?" *IEEE Trans. Signal Process.*, vol. 71, pp. 433–446, 2023.

[22] S. M. Popoff, "complexpytorch," 2022. [Online]. Available: https://github.com/wavefrontshaping/complexPyTorch

[23] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.