Received 18 October 2021; revised 9 December 2021; accepted 28 December 2021. Date of publication 31 December 2021; date of current version 4 February 2022.

Digital Object Identifier 10.1109/OJCOMS.2021.3139858

Matrix-Inverse-Free Deep Unfolding of the Weighted MMSE Beamforming Algorithm

LISSY PELLACO (Graduate Student Member, IEEE), MATS BENGTSSON (Senior Member, IEEE), AND JOAKIM JALDÉN (Senior Member, IEEE)

Division of Information Science and Engineering, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 100 44 Stockholm, Sweden

CORRESPONDING AUTHOR: L. PELLACO (e-mail: pellaco@kth.se)

This work was supported by the European Research Council Project AGNOSTIC under Grant 742648. Part of this work has been presented at ICASSP 2021.

ABSTRACT Downlink beamforming is a key technology for cellular networks. However, computing beamformers that maximize the weighted sum rate (WSR) subject to a power constraint is an NP-hard problem. The popular weighted minimum mean square error (WMMSE) algorithm converges to a local optimum but still exhibits considerable complexity. In order to address this trade-off between complexity and performance, we propose to apply deep unfolding to the WMMSE algorithm for a MU-MISO downlink channel. The main idea consists of mapping a fixed number of iterations of the WMMSE into trainable neural network layers. However, the formulation of the WMMSE algorithm, as provided in Shi *et al.*, involves matrix inversions, eigendecompositions, and bisection searches. These operations are hard to implement as standard network layers. Therefore, we present a variant of the WMMSE algorithm i) that circumvents these operations by applying a projected gradient descent and ii) that, as a result, involves only operations that can be efficiently computed in parallel on hardware platforms designed for deep learning. We demonstrate that our variant of the WMMSE algorithm convergences to a stationary point of the WSR maximization problem and we accelerate its convergence by incorporating Nesterov acceleration and a generalization thereof as learnable structures. By means of simulations, we show that the proposed network architecture i) performs on par with the WMMSE algorithm truncated to the same number of iterations, yet at a lower complexity, and ii) generalizes well to changes in the channel distribution.

INDEX TERMS Deep unfolding, downlink beamforming, iterative optimization algorithm, weighted MMSE algorithm, neural network.

I. INTRODUCTION

DOWNLINK beamforming is a pivotal technology in the fourth and fifth generation cellular communication systems [1]. It leverages the use of multiple antennas to achieve an improved spectral efficiency to meet the demanding performance requirements expected from the system [2]. A common approach to downlink beamforming is to find transmit beamformers that maximize the weighted sum rate (WSR) under a total transmit power constraint. However, the WSR maximization problem is known to be NP-hard [3], [4]. Algorithms that find the optimal solution exist [5]–[7], but the high computational complexity and the consequent latency and power consumption that they exhibit negate the advantages of beamforming. Therefore, it is common to resort to suboptimal solutions that mitigate the computational complexity at the expense of performance. On one hand, simple heuristics with closed form solutions have been proposed, such as zero forcing beamforming and its regularized form [8], maximum ratio transmission [9], matched filtering, and Wiener filtering precoders [10]. These heuristics reduce the computational load and the power consumption, but limit the achieved performance. On the other hand, there exist iterative algorithms, based on convex approximations [11]–[14] or on tractable alternative formulations [15]–[19] of the original problem, that converge to a local optimum. They are more onerous in terms of computational load and power requirements, but achieve a higher WSR. A popular approach that belongs to this class of

iterative algorithms is the weighted minimum mean square error (WMMSE) algorithm [16]. It converges to a local optimum of the WSR function and has gained popularity due to the resulting performance. However, it still exhibits relatively high computational complexity. This has fostered the development of lower-complexity approximations at the cost of a degraded performance [20], [21].

The complexity versus performance trade-off is of considerable importance for cellular networks. Base stations must comply with stringent cost and power specifications, respond to fast changing channel conditions, and accommodate latency requirements of newly 5G-supported real-time applications, from industrial automation to remote medical care. We address the complexity versus performance trade-off by applying a machine-learning-based technique, called *deep unfolding*, to the WMMSE algorithm [16] for a multi-user multiple-input single-output (MU-MISO) downlink channel. In particular, we propose an unfolding of the WMMSE algorithm that only relies on matrix-vector multiplications and scalar nonlinearities that can be efficiently computed in parallel on hardware platforms designed for deep learning tasks. This is accomplished by removing matrix inversions, eigendecompositions, and bisection searches from the iterative updates of the WMMSE algorithm.

A. RELEVANT PRIOR WORK

Inspired by recent advances in machine learning for physical layer applications [22]–[28], machine-learning-based solutions that address the complexity versus performance trade-off have been proposed for downlink beamforming [29]–[31] and, in particular, for the WMMSE algorithm [32]–[34]. The common underlying idea consists of replacing the well-performing, yet expensive and high-latency, iterative algorithms with neural networks. These approaches [32]–[34] are based on end-to-end learning, i.e., neural networks take as input the wireless channel and directly predict the beamformer weights. In this case, the complexity and latency constraints translate into architectural constraints. Therefore, the network can be designed to be compliant with the power and complexity requirements at the expense of bounding its performance. However, the end-to-end approach still presents the issues i) of selecting the proper network architecture, as the search space remains large in spite of the architectural constraints, and ii) of lack of explainability, as the network behaves as a black box.

Lately, a different machine learning approach that leverages domain knowledge has enabled substantial progress for iterative algorithms, like the WMMSE. This approach takes the name *deep unfolding* and was pioneered by Gregor and LeCun in 2010 [35]. The key idea consists of i) mapping the iterations of an optimization algorithm into learnable neural network layers whose structure replicates the iterations of the original algorithm, ii) concatenating a fixed number of layers such that the computational complexity and latency of the inference process are compliant with the requirements, and iii) training learnable parameters of the network in order to achieve the best possible performance given the fixed architecture.

By automatically incorporating expert knowledge in the learning approach, deep unfolding significantly mitigates the problems of architecture selection and of explainability, which are typical of the end-to-end approach, as both the structure and the behavior of the network are largely determined by the underlying algorithm. Clearly, the challenge of selecting trainable parameters remains, but, because the structure of the network mimics the structure of the algorithm itself, the choice is naturally guided by domain knowledge and usually a small subset of parameters is selected to be trainable. As a result, the number of trainable parameters is significantly lower than for the end-to-end learning approach.

Given the advantages of deep unfolding over the end-to-end learning approach, its application to the iterative WMMSE algorithm seems natural. However, as mentioned in [32], its direct application is not straightforward because the WMMSE algorithm involves matrix inversions, eigendecompositions, and bisection searches. These operations are hard to implement as standard network layers. While the authors of [32] fall back on the end-to-end approach for this reason, recent work [36], [37] has shown that it is indeed possible to actualize the benefits of deep unfolding for the WMMSE algorithm. In our previous paper [36], we propose a variant of the WMMSE algorithm, called *unfoldable WMMSE*, which replaces the hard-to-unfold matrix inversions, eigendecompositions, and bisection searches of the original algorithm with operations readily amenable to deep unfolding. In [37], the authors propose a network architecture, called *iterative algorithm induced deep-unfolding neural network* (IAIDNN), given by unfolding a fixed number of iterations of a reformulation of the WMMSE algorithm. This reformulation also avoids eigendecompositions and bisection searches and approximates the matrix inversions (up to the penultimate iteration) by trainable operations structured according to the first-order Taylor expansion of the inverse matrix. The matrix inversions in the last network layer and in the zero forcing initialization step are however computed explicitly.

As previously mentioned, matrix inversions are hard to implement as standard network layers, but do not prevent the application of deep unfolding per se. Given differentiability with respect to the trainable parameters, any operation can in principle be inserted in a network architecture and incorporated into the back-propagation. While matrix inversions along with gradient calculations needed for back-propagation are, for example, supported in software packages, such as Tensorflow [38], their implementation includes steps that are hard to parallelize and thus hard to accelerate using hardware for deep learning heavily reliant on parallelization.

Contrasting the IAIDNN with the unfoldable WMMSE algorithm proposed by us, our solution i) is truly matrix-inverse-free, which brings substantial benefits in terms of hardware implementation, ii) presents significantly fewer learnable parameters, as the dimension of the trainable

parameter space does not scale with the problem dimension, and iii) achieves a higher WSR in the fully loaded scenario, i.e., with an equal number of users and transmit antennas at the base station (see Section VIII-C). In fact, in the fully loaded scenario, the specific trainable structure adopted in the IAIDNN struggles to properly approximate the matrix inversion operation, as also recognized in [37, Section VI-B]. Consequently, for this particular scenario, the authors propose an improved version of the IAIDNN [37, Section VI-E], which, however, foregoes the matrix inverse approximation and replaces it with a step that explicitly includes a matrix inversion. Nevertheless, it must be mentioned that the IAIDNN, despite suffering from performance degradation in the fully loaded scenario unlike our proposed approach, is directly applicable to the more general multi-user multiple-input multiple-output (MU-MIMO) scenario, whereas our network is specifically designed for the MU-MISO case.

Finally, in [39], the authors also unfold the WMMSE algorithm and map it to a network architecture, but restrict their attention to the much simpler single-input single-output (SISO) case, which does not present the problematic matrix inversions, eigendecompositions, and bisection searches that have hampered the application of deep unfolding. Nevertheless, [39] confirms the relevance of deep unfolding for the WMMSE algorithm and the growing interest in this area.

B. CONTRIBUTIONS

This paper significantly extends our conference paper [36], which presents the *unfoldable WMMSE* algorithm. This variant of the WMMSE employs a truncated projected gradient descent (PGD) that involves only operations conformant with the structure of standard neural networks. We set the number of PGD steps and the total number of iterations of the unfoldable WMMSE algorithm based on the complexity and latency requirements. We build a neural network, which we refer to as *deep unfolded WMMSE*, that replicates the fixed number of algorithm iterations and we select the PGD step sizes to be the trainable parameters. In relation to [36], the contributions of this paper are listed below.

- We provide a proof of convergence of the unfoldable WMMSE algorithm to a stationary point of the WSR maximization problem, as established in Theorem 1, and hence demonstrate that the truncated PGD does not nullify the convergence guarantees of the original WMMSE.
- We incorporate in the deep unfolded WMMSE an acceleration technique to boost the convergence of the truncated PGD, while still maintaining a matrix-inverse-free network. We consider both Nesterov acceleration [40], a well-known approach already adopted in deep unfolded architectures [41]–[43], and a generalization thereof, inspired by [44]. These acceleration schemes confer additional degrees of freedom to the network by introducing momentum parameters, which we optimize jointly with the PGD step sizes. This considerably improves performance.
- We perform extensive numerical experiments and show that the trainable acceleration technique provides a performance boost. We show that the (accelerated) deep unfolded WMMSE i) competes with the WMMSE truncated to the same number of iterations with significant computational benefits and ii) outperforms the IAIDNN [37] in a fully loaded scenario and performs on par with it in a lightly loaded scenario. We study through numerical experiments the generalizability of the (accelerated) deep unfolded WMMSE and illustrate its robustness to changes in the channel distribution, i.e., to domain shift.

We believe that the contributions of this paper are not restricted to the WMMSE algorithm only. The idea of resorting to iterative first-order matrix-inverse-free methods to replace steps hard to unfold and parallelize is general and can be applied to a wide range of algorithms in communications and signal processing. In particular, the absence of hard-to-parallelize operations makes the resulting algorithms capable of gaining performance improvements when implemented on hardware optimized for deep learning.

For reproducibility, the code to generate the results in this paper is available in the public GitHub repository at [45].

C. NOTATION

We adopt the following notation. Bold uppercase and bold lowercase letters indicate matrices and column vectors, respectively. $\mathbf{0}$ is a column vector of zeros and $\boldsymbol{I}_M$ is the $M \times M$ identity matrix. $\boldsymbol{I}$ is used if the size of the matrix is clear from the context. $(\cdot)^T$ and $(\cdot)^H$ denote the transpose and the Hermitian transpose of a vector or matrix. $\|\cdot\|$ denotes the 2-norm of a vector or matrix. $\|\cdot\|_{\mathrm{F}}$, $(\cdot)^{-1}$, $\mathtt{Tr}(\cdot)$, $\mathtt{diag}(\cdot)$, and $\sigma_{\max}(\cdot)$ denote the Frobenius norm, the inverse, the trace, the diagonal elements, and the largest singular value of a matrix, respectively. For symmetric matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$, $\boldsymbol{X} \preceq \boldsymbol{Y}$ indicates that $\boldsymbol{Y} - \boldsymbol{X}$ is positive semidefinite. $\Re(\cdot)$ and $\Im(\cdot)$ indicate the real part and the imaginary part and $\mathbb{R}^M$ and $\mathbb{C}^M$ indicate the M-dimensional real and complex spaces, respectively. $\mathbb{E}_y(x)$ indicates the expected value of x computed with respect to the probability distribution of y. $\nabla_x f$ is the gradient of f with respect to x (when clear from context, x is omitted). $\mathcal{CN}(\mu, \sigma^2)$ indicates the complex Gaussian distribution with mean μ and variance σ^2 and analogous notation follows for the multivariate complex Gaussian distribution. $\mathcal{U}(a, b)$ indicates the uniform distribution between a and b.

II. PROBLEM FORMULATION AND SYSTEM MODEL

We consider a multi-user multiple-input single-output (MU-MISO) interference downlink channel. The base station, equipped with M transmit antennas, sends independent data symbols to N single-antenna users. Let $x_i \sim \mathcal{CN}(0, 1)$ be the transmitted data symbol to user i and let $\boldsymbol{h}_i \in \mathbb{C}^M$ be

the channel between the base station and user i. With linear beamforming, the signal at user i is

$$y_i = \boldsymbol{h}_i^H \boldsymbol{v}_i x_i + \sum_{j=1, j\neq i}^{N} \boldsymbol{h}_i^H \boldsymbol{v}_j x_j + n_i, \tag{1}$$

where $\boldsymbol{v}_i \in \mathbb{C}^M$ is the transmit beamformer for user i and $n_i \sim \mathcal{CN}(0, \sigma^2)$ is independent additive white Gaussian noise with power σ^2. The rate of user i is

$$R_i = \log_2 (1 + \mathrm{SINR}_i), \tag{2}$$

where

$$\mathrm{SINR}_i = \frac{|\boldsymbol{h}_i^H \boldsymbol{v}_i|^2}{\sum_{j=1, j\neq i}^{N} |\boldsymbol{h}_i^H \boldsymbol{v}_j|^2 + \sigma^2} \tag{3}$$

is the signal-to-interference-plus-noise-ratio (SINR) of user i. We seek to maximize the weighted sum rate (WSR) subject to a total transmit power constraint, i.e.,

$$\max_{\boldsymbol{V}} \quad \sum_{i=1}^{N} \alpha_i \log_2 (1 + \mathrm{SINR}_i) \tag{4a}$$

$$\text{s.t.} \quad \mathtt{Tr}\left(\boldsymbol{V}\boldsymbol{V}^H\right) \leq P, \tag{4b}$$

where $\boldsymbol{V} \triangleq [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_N]^T$, $\alpha_i > 0$ is the priority of user i (assumed to be known) and P is the maximum transmit power. We assume the base station has perfect channel knowledge and define $\boldsymbol{H} \triangleq [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_N]^T$.

Unlike the quality of service (QoS) problem formulation, in which the transmit beamformers must be selected such that each user reaches a certain QoS [46], problem (4) is guaranteed to always be feasible.

III. WMMSE ALGORITHM

Problem (4) is non-convex and has been shown to be NP-hard [3], [4]. The WMMSE algorithm finds a local optimum of (4) by applying block coordinate descent [47] to

$$\min_{\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V}} \quad \sum_{i=1}^{N} \alpha_i \left(w_i e_i - \log_2 w_i\right) \tag{5a}$$

$$\text{s.t.} \quad \mathtt{Tr}\left(\boldsymbol{V}\boldsymbol{V}^H\right) \leq P, \tag{5b}$$

which has the same optimal $\boldsymbol{V}$ as (4) when

$$\begin{aligned} e_i &= \mathbb{E}_{\boldsymbol{x}, n_i}\left\{\left|\hat{x}_i - x_i\right|^2\right\} \\ &= \sum_{j=1}^{N} |u_i \boldsymbol{h}_i^H \boldsymbol{v}_j|^2 - 2u_i \boldsymbol{h}_i^H \boldsymbol{v}_i + \sigma^2 |u_i|^2 + 1, \end{aligned} \tag{6}$$

where $\boldsymbol{x} \triangleq [x_1, x_2, \ldots, x_N]^T$, $\boldsymbol{x}$ and n_i are assumed to be independent, $\hat{x}_i = u_i y_i$ is the estimated data symbol at the receiver of user i, $u_i \in \mathbb{C}$ is the receiver gain of user i, $\boldsymbol{u} \triangleq [u_1, u_2, \ldots, u_N]^T$, w_i is the weight of user i, and $\boldsymbol{w} \triangleq [w_1, w_2, \ldots, w_N]^T$. Problem (5) is jointly non-convex over $(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V})$, but it is convex in each individual optimization variable $\boldsymbol{u}$, $\boldsymbol{w}$, and $\boldsymbol{V}$. Therefore, by iteratively optimizing over one variable while keeping the others fixed, a local optimum of (5) can be found. This procedure gives the following sequential updates:

$$u_i = \frac{\boldsymbol{h}_i^H \boldsymbol{v}_i}{\sum_{j=1}^{N} |\boldsymbol{h}_i^H \boldsymbol{v}_j|^2 + \sigma^2} \quad \text{for } i = 1, \ldots, N, \tag{7a}$$

$$w_i = \frac{\sum_{j=1}^{N} |\boldsymbol{h}_i^H \boldsymbol{v}_j|^2 + \sigma^2}{\sum_{j=1, j\neq i}^{N} |\boldsymbol{h}_i^H \boldsymbol{v}_j|^2 + \sigma^2} \quad \text{for } i = 1, \ldots, N, \tag{7b}$$

$$\boldsymbol{v}_i = \alpha_i u_i w_i (\boldsymbol{A} + \mu \boldsymbol{I})^{-1} \boldsymbol{h}_i \quad \text{for } i = 1, \ldots, N, \tag{7c}$$

where

$$\boldsymbol{A} \triangleq \sum_{i=1}^{N} \alpha_i w_i |u_i|^2 \boldsymbol{h}_i \boldsymbol{h}_i^H, \tag{8}$$

and $\mu \geq 0$ is a Langrange multiplier chosen such that the power constraint is satisfied. If $\mu = 0$ does not satisfy the power constraint, then the optimal $\boldsymbol{V}$ must satisfy the power constraint with equality. Hence, μ can be found by solving

$$\mathtt{Tr}\left(\boldsymbol{V}\boldsymbol{V}^H\right) = P. \tag{9}$$

As shown in [16], this leads to the equation

$$\sum_{j=1}^{M} \frac{\mathtt{diag}(\boldsymbol{\Phi})_j}{\left(\mathtt{diag}(\boldsymbol{\Lambda})_j + \mu\right)^2} = P, \tag{10}$$

where $\mathtt{diag}(\boldsymbol{X})_j$ indicates the j^{th} diagonal element of a matrix $\boldsymbol{X}$, $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^H$ is the eigendecomposition of $\boldsymbol{A}$, and $\boldsymbol{\Phi} = \boldsymbol{U}^H(\sum_{i=1}^{N} \alpha_i^2 w_i^2 |u_i|^2 \boldsymbol{h}_i \boldsymbol{h}_i^H)\boldsymbol{U}$. The left hand-side of (10) is monotonically decreasing in μ, therefore μ can be found by a bisection search [48] with the starting points

$$\mu_{\text{low}} = \sqrt{\frac{1}{P}\sum_{j=1}^{M} \mathtt{diag}(\boldsymbol{\Phi})_j} \quad \text{and} \quad \mu_{\text{high}} = 0,$$

where μ_{low} and μ_{high} are such that the left-hand side of (10) is smaller and greater than P, respectively.

To summarize, $\boldsymbol{V}$ is first initialized such that the power constraint is satisfied, then $\boldsymbol{u}$, $\boldsymbol{w}$, and $\boldsymbol{V}$ are iteratively updated according to (7) until a convergence criterion is met. For more details on the WMMSE algorithm, we refer the reader to [16].

IV. UNFOLDABLE WMMSE ALGORITHM

We aim to apply deep unfolding to the WMMSE algorithm. Specifically, we aim to build network layers replicating the update equations of the WMMSE algorithm (7). However, the update equation of $\boldsymbol{V}$ (7c) entails a matrix inversion, an eigendecomposition, and a bisection search, i.e., operations that are complicated to represent as standard neural network layers. Therefore, we propose to replace equation (7c) with an alternative update rule that can be readily encoded as a network layer.

In the WMMSE algorithm, update equation (7c) is obtained as the solution with the method of Lagrange

multipliers [49] to the following partial optimization problem in $\boldsymbol{V}$

$$\min_{\boldsymbol{V}} \sum_{i=1}^{N} \alpha_i \left(w_i e_i - \log_2 w_i \right) \tag{11a}$$

$$\text{s.t. } \mathtt{Tr}\left(\boldsymbol{V}\boldsymbol{V}^H \right) \leq P, \tag{11b}$$

where w_i is the weight of user i and $e_i = e_i(\boldsymbol{V})$ is defined in (6). We observe that i) the cost function is convex (quadratic) and differentiable in $\boldsymbol{V}$ and that ii) the constraint is a convex set. Therefore, problem (11) can be alternatively solved with the projected gradient descent (PGD) approach [47]. PGD is a first-order method. Thus, it requires only gradient information and function values. At each iteration, the optimization variable is updated by i) taking a step in the descent direction defined by the negative gradient of the cost function and ii) projecting the update onto the feasible set determined by the constraint.

We define $f(\boldsymbol{V}) \triangleq \sum_{i=1}^{N} \alpha_i (w_i e_i - \log_2 w_i)$ as our cost function and $\mathcal{C} = \{\boldsymbol{V} | \mathtt{Tr}(\boldsymbol{V}\boldsymbol{V}^H) \leq P\}$ as the power constraint set. The k^{th} PGD update is given by

$$\widetilde{\boldsymbol{V}}^k = \boldsymbol{V}^{k-1} - \gamma \nabla f\left(\boldsymbol{V}^{k-1} \right), \tag{12a}$$

$$\boldsymbol{V}^k = \Pi_{\mathcal{C}} \left\{ \widetilde{\boldsymbol{V}}^k \right\}, \tag{12b}$$

where $\nabla f(\boldsymbol{V}^k) = [\nabla f(\boldsymbol{v}_1^k), \nabla f(\boldsymbol{v}_2^k), \ldots, \nabla f(\boldsymbol{v}_N^k)]^T$, where $\nabla f(\boldsymbol{v}_i^k) = -2\alpha_i w_i u_i \boldsymbol{h}_i + 2\boldsymbol{A}\boldsymbol{v}_i^k$, where $\boldsymbol{A}$ is defined in (8), where γ is the step size, and where $\Pi_{\mathcal{C}}\{\boldsymbol{V}\} = \min_{\boldsymbol{Z} \in \mathcal{C}} \|\boldsymbol{V} - \boldsymbol{Z}\|_{\mathrm{F}}$. In particular,

$$\Pi_{\mathcal{C}}\{\boldsymbol{V}\} = \begin{cases} \boldsymbol{V}, & \text{if } \mathtt{Tr}(\boldsymbol{V}\boldsymbol{V}^H) \leq P \\ \frac{\boldsymbol{V}}{\|\boldsymbol{V}\|_{\mathrm{F}}} \sqrt{P}, & \text{otherwise.} \end{cases} \tag{13}$$

In this way, we substitute the matrix inversion, the eigendecomposition, and the bisection search with simple operations, differentiable almost everywhere, that can be easily formulated as standard neural network layers. We refer to this variant formulation of the WMMSE as *unfoldable WMMSE* to stress its suitability for deep unfolding.

To summarize, we initialize $\boldsymbol{V}$ such that the power constraint is satisfied, then we sequentially compute (7a), (7b), and K (inner) PGD steps (12), for a total of L (outer) iterations. Thus, a fixed number of operations is performed, resulting in a deterministic data flow, predetermined execution time, and fixed and known computational complexity.

In [16], Shi *et al.* prove that the iterates of the WMMSE algorithm converge to a stationary point of (5). In the unfoldable WMMSE algorithm, we replace the update equation of $\boldsymbol{V}$ (7c), which solves problem (11) optimally, with a sequence of K PGD steps, which only reduces the objective function in (11). Therefore, a natural question that arises relates to the convergence property of the unfoldable WMMSE algorithm, namely whether the proposed suboptimal update rule for $\boldsymbol{V}$ invalidates the convergence guarantees stated by Shi *et al.* In the following theorem, we establish that the unfoldable WMMSE algorithm retains the same convergence property as the WMMSE algorithm (compare with [16, Th. 3]).

Theorem 1: Any limit point $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \overline{\boldsymbol{V}})$ of the iterates generated by the unfoldable WMMSE algorithm as $L \to \infty$ for a fixed number $K > 0$ of PGD steps of size $0 < \gamma \leq \frac{2\sigma^4}{(\lambda\sigma^2 + P\lambda^2)} \cdot \frac{1}{\bar{\alpha}}$, where $\lambda = \sigma_{\max}(\boldsymbol{H}^H\boldsymbol{H})$ and $\bar{\alpha} = \max_i \alpha_i$, is a stationary point of (5) and the corresponding $\overline{\boldsymbol{V}}$ is a stationary point of (4). Conversely, if $\overline{\boldsymbol{V}}$ is a stationary point of (4), then the point $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \overline{\boldsymbol{V}})$, with $\bar{u}_i$ and $\bar{w}_i$ defined by (7a) and (7b), respectively, is a stationary point of (5).

We relegate the proof to the Appendix, but note that it is evident that if we let $K \to \infty$ the unfoldable WMMSE algorithm exhibits the same convergence guarantees of the WMMSE as $L \to \infty$ because the PGD is known to converge to the optimum solution of (11). However, the point of the theorem is to state that even for a finite number K of PGD steps the unfoldable WMMSE retains the converge property of the WMMSE algorithm established in [16].

A final question of interest is how to choose K and L in practice. While we have not carried out an extensive analysis of this question, it is clear that the complexity largely depends on the product of K and L, i.e., the complexity that can be afforded places a limit on KL. While this typically allows for several configurations of K and L, we find empirically in the numerical results section that choosing K and L of the same order is generally a good choice. In other words, it makes sense to not solely rely on either inner or outer optimization steps.

V. DEEP UNFOLDED WMMSE

It is clear that the performance of the unfoldable WMMSE is penalized by the computational constraint that we impose when setting the number of iterations and the number of PGD steps per iteration. In this computationally-restricted environment, the actual performance of the unfoldable WMMSE depends significantly on the choice of the step sizes, as they determine the behavior and the convergence speed of the sequence of PGD steps. Therefore, we propose to optimize the step sizes by means of deep learning, i.e., by making them trainable parameters of a deep learning architecture that mimics the unfoldable WMMSE algorithm. Specifically, we map each iteration of the unfoldable WMMSE to a neural network layer and we concatenate L layers, such that passing through the network is equivalent to executing L iterations of the unfoldable WMMSE. We refer to this neural network architecture as *deep unfolded WMMSE*. We collectively denote the step sizes as $\boldsymbol{\Gamma} = [\boldsymbol{\gamma}^1, \boldsymbol{\gamma}^2, \ldots, \boldsymbol{\gamma}^L]$, where $\boldsymbol{\gamma}^l = [\gamma^{l,1}, \gamma^{l,2}, \ldots, \gamma^{l,K}]$ and where $\gamma^{l,k}$ is the step size used at the k^{th} PGD step in the l^{th} iteration. In the following, we keep this notation, i.e., we use the superscripts $(\cdot)^{l,k}$ to indicate the k^{th} PGD step in the l^{th} iteration. We omit one of the two superscripts only when it is clear from context. Fig. 1 depicts the deep unfolded WMMSE architecture.

The overall goal is to find the transmit beamformer that maximizes the WSR. Therefore, as the optimal beamformer is unknown, a natural choice of loss function for network

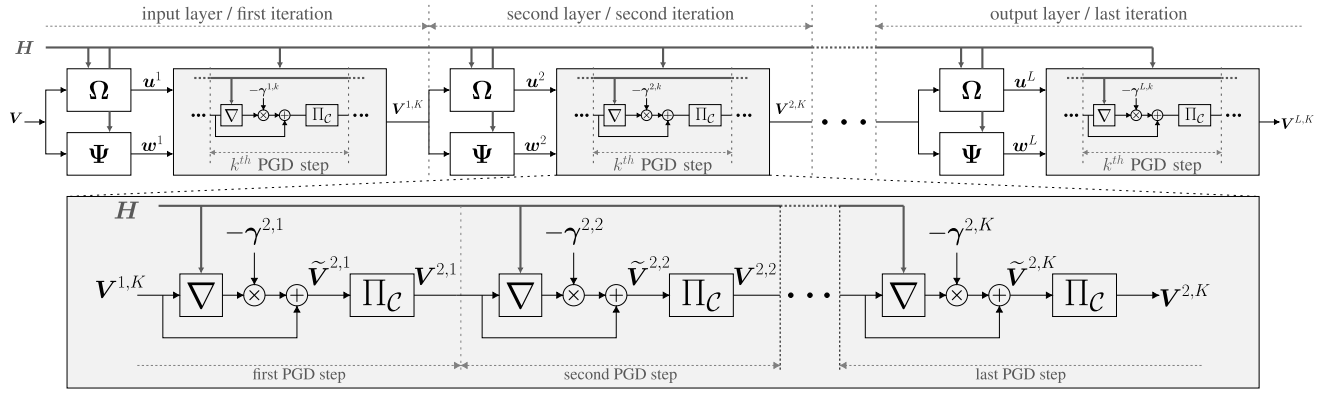

FIGURE 1. Network architecture of the deep unfolded WMMSE. It is given by L iterations of the unfoldable WMMSE algorithm. The superscripts $(\cdot)^{l,k}$ indicate the k^{th} PGD step in the l^{th} layer/iteration. Each layer consists of the update equation of u (7a), denoted by Ω, of the update equation of w (7b), denoted by Ψ, and of K PGD steps, as depicted in the gray box. The step sizes of the truncated PGD sequence are the trainable parameters. ∇ and $\Pi_{\mathcal{C}}$ denote the gradient and the projection operations in (12), respectively.

training consists of directly maximizing the WSR achieved with the transmit beamformer given as output by the network, as in [50]. Note that this, as a byproduct, also makes the training unsupervised. However, in order to avoid possible complications that might occur during training, such as vanishing gradients and saturation of hidden units [51], we also include in the loss function the WSR achieved by using the transmit beamformers given as output by all the other layers. This training recommendation is inspired by the concept of auxiliary classifiers in GoogLeNet [52] and was adopted in [53]. Thus, we adopt the following loss function

$$\mathcal{L}(\boldsymbol{\Gamma}) = -\frac{1}{N_{\mathrm{s}}} \sum_{n=1}^{N_{\mathrm{s}}} \sum_{l=1}^{L} f_{\mathrm{WSR}}\Big(\boldsymbol{H}_n, \boldsymbol{V}_n^l(\boldsymbol{H}_n; \boldsymbol{\Gamma})\Big), \tag{14}$$

where N_{s} is the size of the training set and $f_{\mathrm{WSR}}(\boldsymbol{H}_n, \boldsymbol{V}_n^l(\boldsymbol{H}_n; \boldsymbol{\Gamma}))$ indicates the WSR achieved with the n^{th} channel realization, drawn i.i.d. from the distribution of $\boldsymbol{H}$, and with the transmit beamformer given as output by the l^{th} layer of the neural network with $\boldsymbol{\Gamma}$ as trainable parameters. By adopting this unsupervised approach, we avoid the time- and resource-consuming process of generating labeled data for the training phase [54]. The loss function $\mathcal{L}(\boldsymbol{\Gamma})$ is continuous and differentiable almost everywhere with respect to $\boldsymbol{\Gamma}$. Hence, we can use gradient-based optimization methods [35] and back-propagation for training. In particular, we minimize $\mathcal{L}(\boldsymbol{\Gamma})$ by applying the Adam optimizer [55], which is a variant of stochastic gradient descent.

The structure of the PGD update is chosen to yield steps toward the optimal solution of problem (11). However, by adopting the loss function in (14), we do not train the step sizes such that the truncated PGD can approximate as closely as possible the update equation of $\boldsymbol{V}$ (7c) in the WMMSE, which solves optimally (11). Instead, we train the step sizes such that the deep unfolded WMMSE can reach the highest possible WSR within the fixed network architecture and complexity. As will be shown in the numerical results section, this allows the deep unfolded WMMSE to often outperform, for a fixed number of iterations, the WMMSE algorithm truncated to the same number of iterations. This is similar to observations made in [39].

VI. ACCELERATION

As will be shown in the numerical results section, there are cases in which the exact optimization of problem (11), as carried out in the original WMMSE algorithm, is beneficial and the deep unfolded WMMSE struggles to compete with the WMMSE truncated to the same number of iterations. A natural remedy to this consists of solving problem (11) more accurately. This can be trivially achieved by increasing the number of PGD steps, but this comes at the cost of a higher computational load. Alternatively, we can also boost the convergence of the PGD toward the optimum through an acceleration technique. Therefore, we propose to incorporate in the layer structure of the deep unfolded WMMSE an acceleration scheme and employ it as a learnable structure. In particular, we consider Nesterov acceleration [40], a well-known approach to speed up the convergence of first-order methods, and a generalization thereof, inspired by [44]. These acceleration schemes come at a minimal computational overhead with respect to increasing the number of PGD steps and have the benefit of conferring an increased flexibility to the trainable network by providing additional degrees of freedom. In the standard formulation, the k^{th} Nesterov-accelerated PGD update is

$$\widetilde{\boldsymbol{V}}^k = \boldsymbol{V}^{k-1} + \theta \bar{\boldsymbol{V}}^{k-1} - \gamma \nabla f\Big(\boldsymbol{V}^{k-1} + \theta \bar{\boldsymbol{V}}^{k-1}\Big), \tag{15a}$$

$$\boldsymbol{V}^k = \Pi_{\mathcal{C}}\Big\{\widetilde{\boldsymbol{V}}^k\Big\} \tag{15b}$$

where $\bar{\boldsymbol{V}}^k = \boldsymbol{V}^k - \boldsymbol{V}^{k-1}$, θ is the momentum parameter, and f and ∇f are defined as in the PGD update (12). However, in order to grant more flexibility, we can also reformulate the standard Nesterov acceleration by adding an extra momentum parameter ξ that specifically regulates the location at which the gradient is computed [44], i.e.,

$$\widetilde{\boldsymbol{V}}^k = \boldsymbol{V}^{k-1} + \theta \bar{\boldsymbol{V}}^{k-1} - \gamma \nabla f\Big(\boldsymbol{V}^{k-1} + \xi \bar{\boldsymbol{V}}^{k-1}\Big), \tag{16a}$$

$$\boldsymbol{V}^k = \Pi_{\mathcal{C}}\left\{\widetilde{\boldsymbol{V}}^k\right\}. \tag{16b}$$

We refer to this scheme, which includes standard Nesterov acceleration by setting $\xi = \theta$, as *super Nesterov* acceleration. We can also recover the standard PGD formulation (12) by setting the momentum parameters to zero, a case for which convergence is again established by Theorem 1.

We treat the acceleration scheme as a learnable structure and we propose to optimize the momentum parameters, collectively denoted as $\boldsymbol{\Theta}$ and $\boldsymbol{\Xi}$, jointly with the step sizes $\boldsymbol{\Gamma}$. For training, we adopt the same loss function specified in (14), which is now a function of the step sizes and of the momentum parameters, i.e., $\mathcal{L}(\boldsymbol{\Gamma}, \boldsymbol{\Theta}, \boldsymbol{\Xi})$. It is continuous and jointly differentiable almost everywhere over $(\boldsymbol{\Gamma}, \boldsymbol{\Theta}, \boldsymbol{\Xi})$. Therefore, we employ the Adam optimizer and back-propagation also in this case.

VII. COMPLEXITY ANALYSIS

In this section, we analyze the computational complexity of the unfoldable WMMSE algorithm and of the original WMMSE algorithm. To facilitate the comparison, we do not consider the cost of updating $\boldsymbol{u}$ (7a) and $\boldsymbol{w}$ (7b), as these steps are common to the two algorithms and their contribution to the total complexity is lower with respect to the update of $\boldsymbol{V}$. Further, in order to simplify the derivations below, we only consider complex-valued multiplications.

At each iteration of the WMMSE algorithm we need to compute the following.

- *The Matrix $\boldsymbol{A}$:* It requires N vector-matrix multiplications, each with a cost of M^2. If $N = M$, as we assume in the numerical results section, the complexity is M^3.
- *The Regularized Inverse of $\boldsymbol{A}$:* It can be computed by first executing the LU decomposition of $\boldsymbol{A}$ and then by solving the associated linear systems. The complexity is $2M^3$ [56].
- *The Eigenvalues and Eigenvectors of $\boldsymbol{A}$:* It can be done with the tridiagonal QR iteration. It is an iterative algorithm and on average it costs approximately $6M^3$. However, it requires the input matrix to be tridiagonal. The conversion of matrix $\boldsymbol{A}$ into this form costs $\frac{8}{3}M^3$. Therefore, the total complexity is $\frac{26}{3}M^3$ [57].
- *The Diagonal Elements of the Matrix $\boldsymbol{\Phi}$ in (10):* It requires N scalar-matrix multiplications, each with a complexity of M^2, and one $M \times M$ matrix-matrix multiplication, with a complexity of M^3. The right multiplication with $\boldsymbol{U}$ can be obtained with M^2 multiplications as only the diagonal elements are needed. If $N = M$, as we assume in the numerical results section, the total complexity is $2M^3$.
- *The Update of $\boldsymbol{V}$ (7c):* It requires N vector-matrix multiplications, each with a cost of M^2. If $N = M$, as we assume in the numerical results section, the complexity becomes M^3.

Therefore, the overall cost per iteration of the WMMSE algorithm is $\frac{44}{3}M^3 + \mathcal{O}(M^2)$, where $\mathcal{O}(M^2)$ accounts for the lower complexity terms from the matrix inverse and eigendecomposition operations, from the scalar multiplications in the update of $\boldsymbol{V}$, and from the computation of the diagonal elements of matrix $\boldsymbol{\Phi}$.

At each iteration of the unfoldable WMMSE algorithm we need to compute the following.

- *The Matrix $\boldsymbol{A}$:* It requires N vector-matrix multiplications, each with a cost of M^2. If $N = M$, as we assume in the numerical results section, the complexity is M^3.
- *The Update of $\boldsymbol{V}$:* It requires N vector-matrix multiplications, each with a cost of M^2, for each PGD step. If $N = M$, as we assume in the numerical results section, the complexity becomes KM^3.

Therefore, the overall cost per iteration of the unfoldable WMMSE algorithm is $(K+1)M^3 + \mathcal{O}(M^2)$. The same complexity analysis holds for the accelerated version given that equations (15) and (16) only entail $\mathcal{O}(M^2)$ cost terms.

It follows that for $K < \frac{41}{3}$, as in our simulations, the cost of the unfoldable WMMSE is reduced with respect to the cost of the WMMSE algorithm. However, this comparison is not entirely fair because it hides the fact that all the computations involved in the unfoldable WMMSE algorithm are highly parallelizable and thus suitable for implementation on standardized hardware optimized for deep learning. Conversely, the computations specific to the original WMMSE algorithm, such as matrix inversion, eigendecomposition, and the bisection search, are sequential in nature and typically implemented via iterative procedures [57]. Therefore, it must be highlighted that the main computational benefit of the proposed matrix-inverse-free unfoldable WMMSE algorithm is not to reduce the multiplicative term in the dominant complexity term M^3. The main benefit is to make the computations more easily parallelizable. In fact, the unfoldable WMMSE can benefit from M parallel computation paths of $\mathcal{O}(M^2)$ cost, while the original WMMSE, because of the sequential nature of the steps therein, cannot leverage parallel computations.

VIII. NUMERICAL RESULTS

A. SETUP

The (accelerated) deep unfolded WMMSE algorithm was implemented in Python 3.6.8 with Tensorflow 1.13.1 and the original WMMSE algorithm was implemented in Python 3.6.8. Full code to reproduce all the results in the paper, including dataset generation, is available in the public GitHub repository at [45].

It should be noted that the (accelerated) deep unfolded WMMSE can be readily implemented with basic neural network operations. The projection (13) can in fact be equivalently implemented without conditional statements as

$$\Pi_{\mathcal{C}}\{\boldsymbol{V}\} = \frac{\boldsymbol{V}\sqrt{P}}{\phi\left(\|\boldsymbol{V}\|_{\mathrm{F}} - \sqrt{P}\right) + \sqrt{P}},$$

where $\phi(x) = \max(0, x)$ is the rectified linear unit (ReLU). In the simulations, we adopt the following setup:

- We set $N = M$ unless otherwise stated and $\alpha_i = 1$ for $i = 1, \ldots, N$.
- As in [15], [17], [34], we initialize the WMMSE and the (accelerated) deep unfolded WMMSE with the matched filter, i.e., $\boldsymbol{V} = a\boldsymbol{H}$, where $a \in \mathbb{R}$ is chosen such that the power constraint is satisfied with equality.
- In the accelerated deep unfolded WMMSE, for the first PGD iteration, we set $\boldsymbol{V}^{l,-1} = \boldsymbol{0}$ for $l = 1, \ldots, L$.
- In the WMMSE, we end the bisection search when (10) is satisfied up to an error of 10^{-4}. Unless a fixed number of iterations is considered, we assume the WMMSE reaches convergence when the increment in WSR at the next iteration is less than or equal to 10^{-4} bits per channel use.
- In the deep unfolded WMMSE, unless otherwise stated, we initialize the step sizes to one, i.e., $\gamma^{l,k} = 1$ for $k = 1, \ldots, K$ and $l = 1, \ldots, L$. In the accelerated deep unfolded WMMSE, we initialize the step sizes to the learnt values obtained by training the deep unfolded WMMSE and we initialize the momentum parameters to zero, i.e., $\theta^{l,k} = 0$ and $\xi^{l,k} = 0$ for $k = 1, \ldots, K$ and $l = 1, \ldots, L$.
- We set the learning rate of the Adam optimizer to 10^{-3}.
- For each combination of M, L, K, and P/σ^2 we train a new network, unless otherwise stated.
- The training and the test sets consist of channel realizations drawn i.i.d. from $\mathcal{CN}(\boldsymbol{0}, \boldsymbol{I})$ unless otherwise stated. We enlarge the size of the training set, from $4 \cdot 10^6$ to 10^7, for increasing values of M, L, K, and P/σ^2 and we set the batch size to 10^2. The results are averaged over 10^5 channel realizations.
- As available deep learning tools do not support complex numbers, we adopt the equivalent real-valued representation given by $u' = [\Re(u),\ \Im(u)]$, $w' = [\Re(w),\ \Im(w)]$, $\boldsymbol{v}' = [\Re(\boldsymbol{v})^T,\ \Im(\boldsymbol{v})^T]^T$, and

$$\boldsymbol{h}' = \begin{bmatrix} \Re(\boldsymbol{h}) & -\Im(\boldsymbol{h}) \\ \Im(\boldsymbol{h}) & \Re(\boldsymbol{h}) \end{bmatrix}.$$

We have verified the robustness of training of the neural network by re-running the simulation set multiple times. The results are consistent across different runs and are consistent with the performance metrics reported in the following sections, even if each metric refers to a single training of the neural network.

B. PERFORMANCE

In this section, we compare i) the performance of the deep unfolded WMMSE, ii) the performance of the accelerated deep unfolded WMMSE, both with Nesterov and super Nesterov acceleration, and iii) the performance of the original WMMSE truncated to the same number of outer iterations. We use the zero forcing (ZF) and its regularized form (RZF) as baseline solutions [8], [20].

Fig. 2 shows the performance of the different approaches as the number of iterations L varies, with $P/\sigma^2 = 10$ dB and with $K = 4$ and $K = 8$, i.e., with 4 and 8 PGD steps in each layer of the (accelerated) deep unfolded WMMSE. As can be noted, the deep unfolded WMMSE with $K = 4$ surpasses the truncated WMMSE with a margin that decreases as the number of iterations increases, reaching, for $L = 6$, the 99.2 percent of the WSR achieved by the WMMSE at convergence. Although very close to convergence, by extending the PGD sequence to $K = 8$, i.e., by further moving toward the optimal solution of (11), we realize a small gain in performance. However, as we increased the number of PGD steps, we empirically observed that the neural network converged with difficulty to a good optimum [51]. We addressed this complication by progressively adding a single step size from $K = 4$ to $K = 8$ and by jointly training the newly added step size with the pretrained step sizes, initialized to their optimized values. From Fig. 2 we also observe that the accelerated deep unfolded WMMSE variants outperform the non-accelerated alternatives, with a small margin that decreases as the number of iterations increases. In particular, super Nesterov acceleration surpasses Nesterov acceleration, both in case of $K = 4$ and $K = 8$, even though the gain is modest. It can also be noted that the ZF and the RZF solutions (reported in the caption to avoid cluttering the plot) fail to compete with the considered approaches.

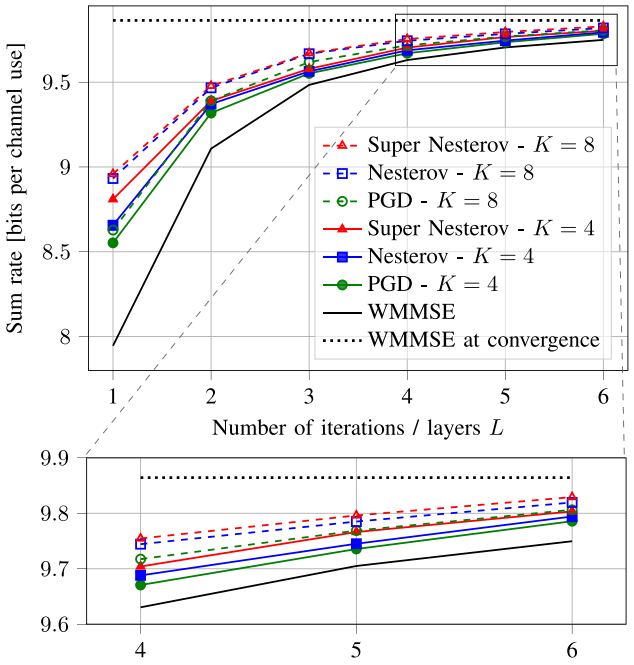

FIGURE 2. WSR obtained with $N = M = 4$ and $P/\sigma^2 = 10$ dB. The WSR, in bits per channel use, achieved by the WMMSE at convergence is 9.864, by the ZF is 5.173, and by the RZF is 8.28.

Fig. 3 presents the same performance comparison of Fig. 2, but in case of higher P/σ^2, i.e., 20 dB. This case is of greater interest as the deep unfolded WMMSE for $L = 6$ and $K = 4$ only reaches 92.35 percent of the WSR achieved at convergence by the WMMSE. The margin of improvement is thus larger and in fact the increment in WSR achieved i) by extending the sequence of PGD steps to $K = 8$ and

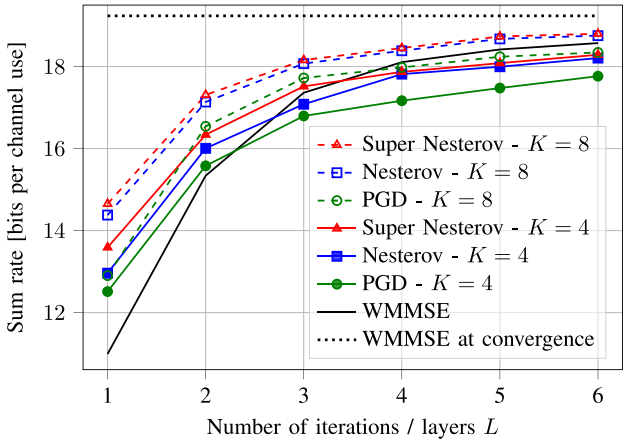

FIGURE 3. WSR obtained with $N = M = 4$ and $P/\sigma^2 = 20$ dB. The WSR, in bits per channel use, achieved by the WMMSE at convergence is 19.237, by the ZF is 14.924, and by the RZF is 16.351.

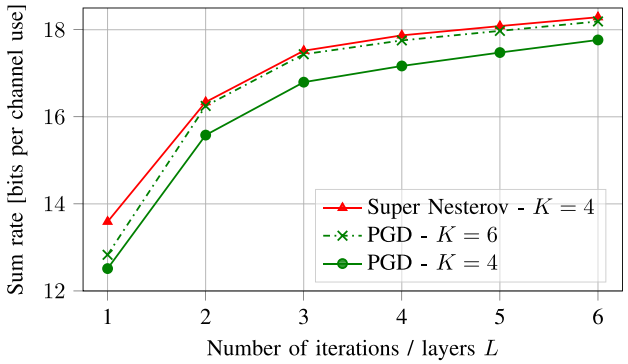

FIGURE 4. WSR obtained with $N = M = 4$ and $P/\sigma^2 = 20$ dB.

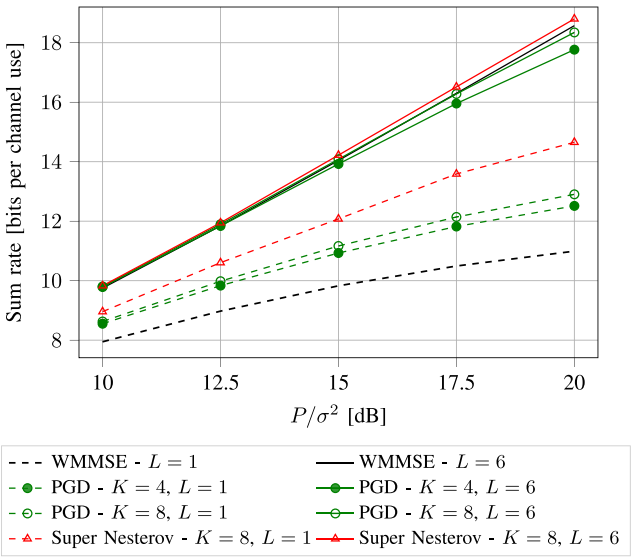

FIGURE 5. WSR obtained with $N = M = 4$.

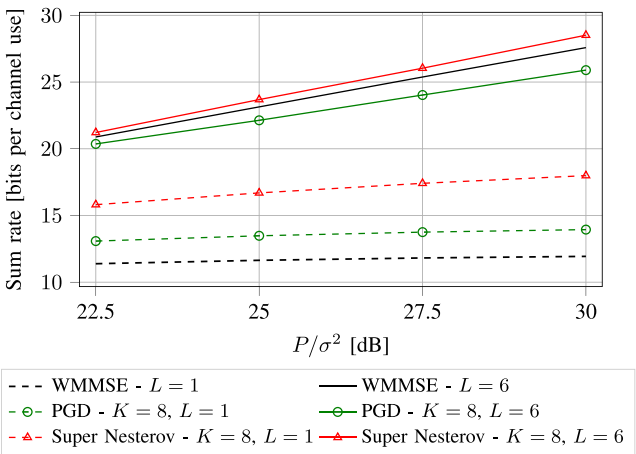

FIGURE 6. WSR obtained with $N = M = 4$.

ii) by incorporating a learnable acceleration scheme is more significant. In particular, it can be noted that for $L \geq 4$ the accelerated deep unfolded WMMSE with $K = 8$ is the only approach that outperforms the truncated WMMSE, achieving, for $L = 6$, 97.73 percent of the WSR attained at convergence by the WMMSE. Finally, with $P/\sigma^2 = 20$ dB, the ZF and the RZF have a better performance, but still fail to compete with the proposed approaches for $L > 2$.

Fig. 4 highlights the benefit of employing a learnable acceleration scheme over extending the sequence of PGD steps. As can be seen, super Nesterov acceleration with $K = 4$ performs comparably to the non-accelerated alternative with $K = 6$, yet at a lower complexity as the computational overhead of the acceleration scheme is minimal with respect to extending the sequence of PGD steps.

Fig. 5 shows the WSR attained by the various approaches for P/σ^2 in the range from 10 dB to 20 dB. The trends observed for 10 and 20 dB are confirmed for intermediate P/σ^2 values as well. Figure 6 shows the WSR attained for $P/\sigma^2 > 20$ dB by the (accelerated) deep unfolded WMMSE with $K = 8$ and by the truncated WMMSE. In this case, we consider only $K = 8$ because, as observed before, as P/σ^2 increases it is beneficial to employ more PGD steps in each layer. It can be seen that even at higher P/σ^2 conditions, the proposed approach continues to perform well.

So far we have reported the results in terms of *average* WSR over the test set. However, the performance of the WMMSE and of the (accelerated) deep unfolded WMMSE can vary over different channel realizations. In order to investigate this, we look at i) the empirical WSR distribution achieved by the truncated WMMSE and by our best performing method, i.e., super Nesterov acceleration with $K = 8$, and at ii) the empirical distribution of the corresponding difference in WSR computed channel by channel. As can be seen in Fig. 7, the empirical WSR distributions of the two methods closely match. Moreover, the narrowness of the difference distribution indicates that for any given channel realization the accelerated deep unfolded WMMSE performs similarly to the truncated WMMSE, sometimes even surpasses it for the given number of iterations. With $P/\sigma^2 = 20$ dB, as shown by Fig. 8, the overall match between the two

TABLE 1. WSR obtained in various scenarios.

P/σ^2 [dB]	10		20	
(M, N)	(4,4)	(8,4)	(4,4)	(8,4)
WMMSE at convergence [bits per channel use]	9.864	14.833	19.237	27.198
Super Nesterov $K = 8, L = 6$	99.64 %	98.58 % $\xrightarrow{\text{training}}$ 99.97 %	97.73 %	94.73 % $\xrightarrow{\text{training}}$ 99.89 %
IAIDNN, $L = 6$	95.79 %	99.99 %	94.04 %	99.97 %

WSR obtained by the different approaches, expressed as percentages of the WSR achieved by the WMMSE at convergence. In case of $(M = 8,\ N = 4)$, we first evaluate the performance of the accelerated deep unfolded WMMSE trained for $(M = 4,\ N = 4)$ (WSR reported on the left-hand side of the arrow) and then we continue training it for the specific $(M = 8,\ N = 4)$ scenario (WSR reported on the right-hand side of the arrow).

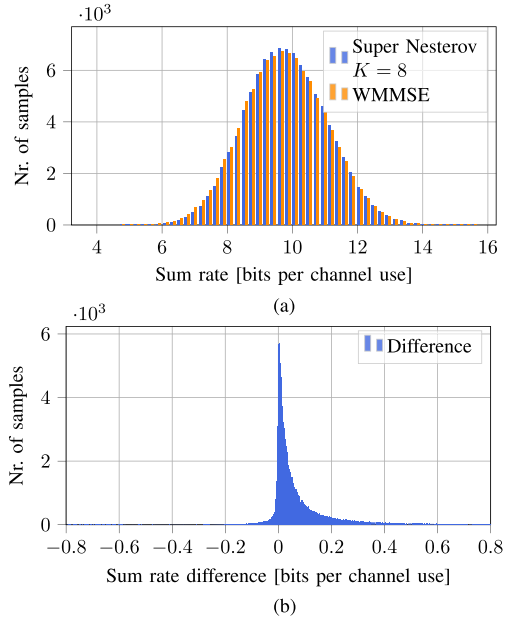

FIGURE 7. Distributions of the WSR achieved by the accelerated deep unfolded WMMSE and the truncated WMMSE (a) and the corresponding WSR difference (b), where positive difference implies higher WSR for the accelerated deep unfolded WMMSE. The results are obtained for $N = M = 4$, $L = 6$, $K = 8$, and $P/\sigma^2 = 10$ dB.

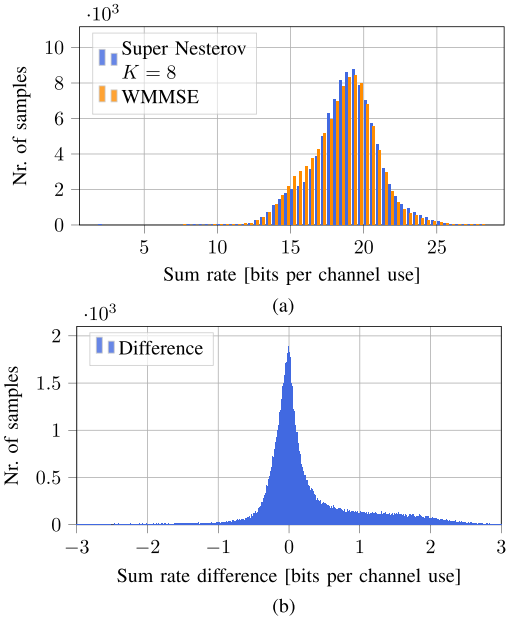

FIGURE 8. Distributions of the WSR achieved by the accelerated deep unfolded WMMSE and the truncated WMMSE (a) and the corresponding WSR difference (b), where positive difference implies higher WSR for the accelerated deep unfolded WMMSE. The results are obtained for $N = M = 4$, $L = 6$, $K = 8$, and $P/\sigma^2 = 20$ dB.

WSR distributions is good as well, although the difference distribution presents longer tails.

C. COMPARISON TO IAIDNN

As the IAIDNN [37] is also based on the idea of deep unfolding, we provide a detailed comparison between our approach and the IAIDNN in order to highlight the differences. To this end, we compare the performance of our best performing method, i.e., the deep unfolded WMMSE with super Nesterov acceleration with $K = 8$, against the performance of the IAIDNN [37], in the fully loaded ($M = N$) and in the lightly loaded ($M > N$) scenarios. For reference, we also consider the performance of the unaccelerated deep unfolded WMMSE with $K = 4$ and of the original WMMSE truncated to the same number of iterations.

For the IAIDNN, we use the authors's code publicly available at [58], we set the batch size to 10^2, and we train the network until convergence. To initialize the IAIDNN, the authors adopt zero forcing initialization, which however includes a matrix inversion. This initialization is not technically necessary, as it can be replaced with the matched filter initialization, which is matrix-inverse-free. The matched filter initialization does not cause performance degradation as far as we can tell. Nevertheless, for comparison, we report the IAIDNN results obtained with the zero forcing initialization, as originally proposed by the authors [37].

Fig. 9 (a) shows the performance of the various approaches as the number of iterations L increases, with $P/\sigma^2 = 10$ dB and in case of $(M = 4, N = 4)$, i.e., in the fully loaded scenario. As can be seen, the performance of all the approaches improves with L. In particular, the accelerated deep unfolded WMMSE and the IAIDNN reach, respectively, for $L = 6$, 99.64 percent and 95.79 percent of the WSR achieved at convergence by the WMMSE. Although satisfactory, the performance of the IAIDNN lies slightly below the performance of our proposed approach and fails to surpass the truncated WMMSE. This result was expected, as the authors acknowledge in [37, Secs. VI-B and VI-E] that in the fully loaded scenario the performance of the IAIDNN is largely penalized by the adopted matrix inverse approximation. Such approximation is indeed more suitable to the lightly loaded scenarios ($M > N$), i.e., when the individual

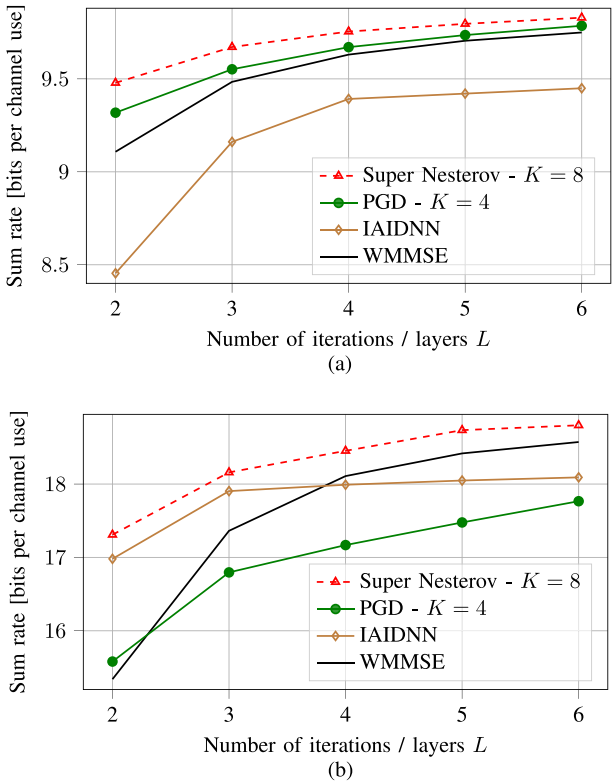

FIGURE 9. WSR obtained with $N = M = 4$, in case of $P/\sigma^2 = 10$ dB (a) and of $P/\sigma^2 = 20$ dB (b).

user channels are more likely to be orthogonal. This can be seen in Table 1, where the IAIDNN exhibits excellent performance in the $(M = 8, N = 4)$ scenario.

Moreover, the significantly larger trainable space of the IAIDNN with respect to the accelerated deep unfolded WMMSE makes training more complicated in practice. The total number of trainable parameters for the IAIDNN is $7LN+(L-1)N(3M^2+M)$, whereas the total number of trainable parameters for the accelerated deep unfolded WMMSE with super Nesterov is $3LK$ (it reduces to $2LK$ with Nesterov acceleration and to LK without acceleration). As can be seen, the dimension of the trainable space of our approach conveniently does not depend on the problem dimension, i.e., the number of users N and the number of base station antennas M.

Fig. 9 (b) shows the same performance comparison as Fig. 9 (a), but in case of $P/\sigma^2 = 20$ dB. As can be seen, the same overall trend is confirmed, with the exception that here the IAIDNN outperforms i) the truncated WMMSE up to $L = 3$ and ii) the unaccelerated deep unfolded WMMSE with $K = 4$, which, as we already observed, in case of $P/\sigma^2 = 20$ dB yields mediocre performance. In this case, the accelerated deep unfolded WMMSE and the IAIDNN reach, respectively, for $L = 6$, 97.73 percent and 94.04 percent of the WSR achieved at convergence by the WMMSE.

Finally, Table 1 reports the performance of the various approaches as percentages of the WSR achieved by the WMMSE at convergence. It reports the results, for $L = 6$, for the fully loaded scenario $(M = 4, N = 4)$ and for the lightly loaded scenario $(M = 8, N = 4)$. In the fully loaded scenario, as already shown in Fig. 9, the IAIDNN reaches a satisfactory performance but fails to surpass the truncated WMMSE. In the lightly loaded scenario, instead, the IAIDNN has a boost in performance and for $L = 6$ achieves 99.99 percent of the WSR at convergence in case of $P/\sigma^2 = 10$ dB and 99.97 percent in case of $P/\sigma^2 = 20$ dB. With regards to the accelerated deep unfolded WMMSE, in the fully loaded scenarios, it surpasses the truncated WMMSE and, in the lightly loaded scenario, it yields performance on par with the IAIDNN. In particular, in the $(M = 8, N = 4)$ scenario, we consider i) the performance given by the accelerated deep unfolded WMMSE trained for $(M = 4, N = 4)$, i.e., trained on a different channel distribution, and ii) the performance achieved by the same network with additional training for the specific case of $(M = 8, N = 4)$. The re-trained network achieves a WSR comparable to the IAIDNN for $L = 6$, whereas the network tailored to the $(M = 4, N = 4)$ case has a lower performance, as expected. However, even without retraining the accelerated deep unfolded WMMSE to the specific channel distribution, it achieves 98.58 percent of the WSR at convergence for $P/\sigma^2 = 10$ dB and 94.73 percent for $P/\sigma^2 = 20$ dB, respectively, showing good generalizabilty. The generalization capability of the (accelerated) deep unfolded WMMSE is further discussed in the following section.

D. NETWORK GENERALIZATION CAPABILITIES

In the previous sections, we mostly presented results obtained by testing the deep unfolded WMMSE on channel realizations drawn from the same channel distribution used to generate the training set. A common criticism to neural-network-based solutions is that they have poor generalization capabilities and thus suffer from severe performance degradation in practical scenarios, in which there is a mismatch between the training and test distributions. To investigate the robustness of the (accelerated) deep unfolded WMMSE to changes in the channel distribution, we first consider a scenario in which each user experiences a different path loss. To simulate this, we model the channel as $\tilde{\boldsymbol{h}}_i = \sqrt{d_i}\boldsymbol{h}_i$, where $\boldsymbol{h}_i \sim \mathcal{CN}(\boldsymbol{0}, \boldsymbol{I}_M)$ represents the small scale fading component and $d_i \sim \mathcal{U}(-5, 5)$ dB represents the path loss experienced by user i. However, we do not retrain the (accelerated) deep unfolded WMMSE on channel realizations with different path losses. We test the network previously trained for the nominal case, i.e., with $d_i = 0$ dB $\forall i$. Figs. 10 and 11 show the empirical WSR distribution and the empirical difference distribution for the cases of $P/\sigma^2 = 10$ dB and $P/\sigma^2 = 20$ dB, respectively. As can be seen, for $P/\sigma^2 = 10$ dB, the (accelerated) deep unfolded WMMSE attains a WSR profile on par with the truncated WMMSE. For $P/\sigma^2 = 20$ dB we observe corner cases

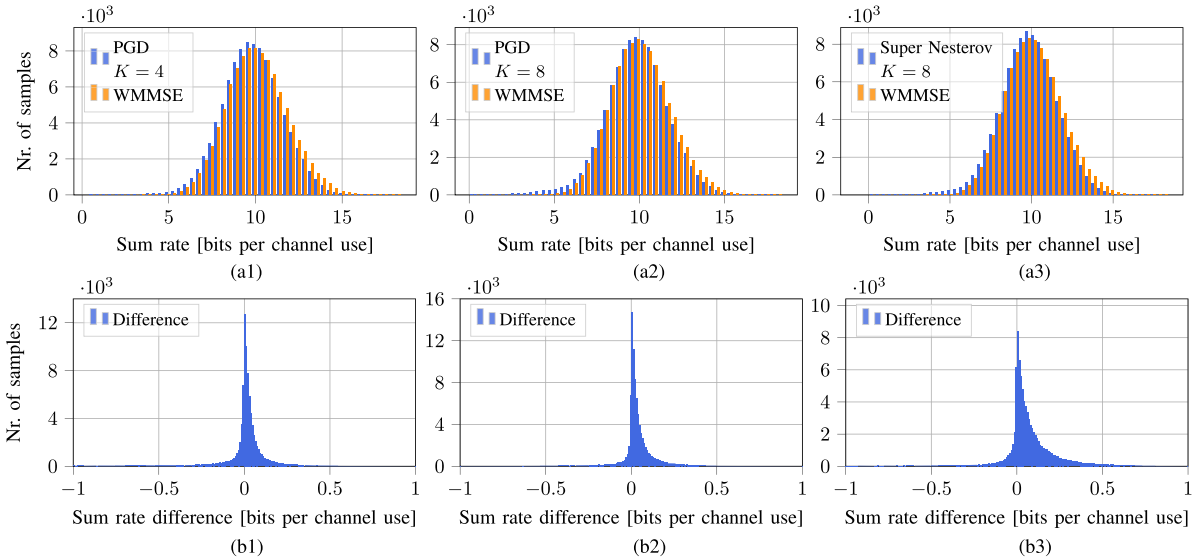

FIGURE 10. Distributions of the WSR achieved by the deep unfolded WMMSE and the truncated WMMSE (a1) - (a3) and the corresponding WSR difference (b1) - (b3), where positive difference implies higher WSR for the deep unfolded WMMSE. The results are obtained for $N = M = 4$, $L = 6$, $P/\sigma^2 = 10$ dB and a path loss component drawn from $\mathcal{U}(-5,5)$ dB for each user. The deep unfolded WMMSE was trained for the nominal case of $P/\sigma^2 = 10$ dB, i.e., with a path loss of 0 dB for each user. The average WSR, expressed in bits per channel use, achieved by the truncated WMMSE is 10.078, while the average WSR achieved by the deep unfolded WMMSE is 9.845 (a1), 9.866 (a2), and 9.879 (a3). For reference, the WSR achieved by the WMMSE at convergence is 10.255, by the ZF is 4.83, and by the RZF is 8.352.

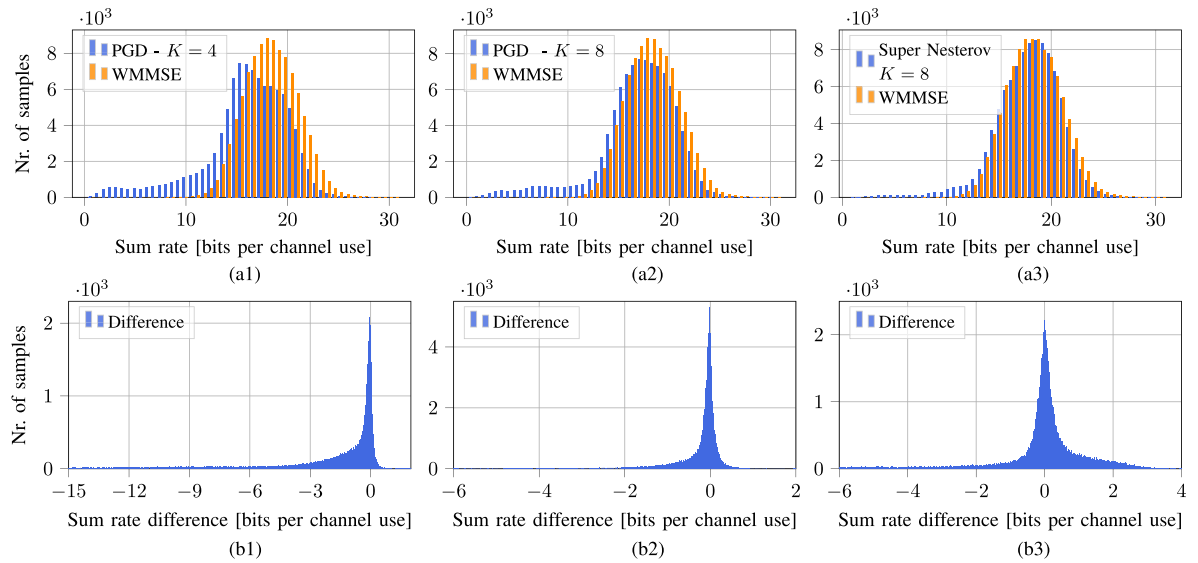

FIGURE 11. Distributions of the WSR achieved by the deep unfolded WMMSE and the truncated WMMSE (a1)–(a3) and the corresponding WSR difference (b1)–(b3), where positive difference implies higher WSR for the deep unfolded WMMSE. The results are obtained for $N = M = 4$, $L = 6$, $P/\sigma^2 = 20$ dB and a path loss component drawn from $\mathcal{U}(-5,5)$ dB for each user. The deep unfolded WMMSE was trained for the nominal case of $P/\sigma^2 = 20$ dB, i.e., with a path loss of 0 dB for each user. The average WSR, expressed in bits per channel use, achieved by the truncated WMMSE is 18.345, while the average WSR achieved by the deep unfolded WMMSE is 16.035 (a1), 16.984 (a2), and 17.866 (a3). For reference, the average WSR achieved by the WMMSE at convergence is 19.308, by the ZF is 14.263, and by the RZF is 16.005.

in which the (accelerated) deep unfolded WMMSE struggles to compete with the truncated WMMSE. Nevertheless, we can also observe in the difference distribution (b3) of Fig. 11 a significant percentage of channel realizations for which the accelerated deep unfolded WMMSE delivers a higher WSR with respect to the truncated WMMSE.

We also investigate the robustness to a mismatch in the number of antennas at the base station between training and testing. In particular, we test the (accelerated) deep unfolded WMMSE, previously trained for $(M = 4, N = 4)$, over the case of $(M = 8, N = 8)$. Fig. 12 shows the resulting empirical WSR and difference distributions. Despite the model and distribution mismatch, the deep unfolded WMMSE and its accelerated variant show good generalization capabilities leading to performances comparable to the truncated WMMSE, with the exception of a low-probability tail. This confirms what we already observed in Table 1, namely that our approach trained on the fully loaded scenario $(M = 4, N = 4)$ generalizes well to

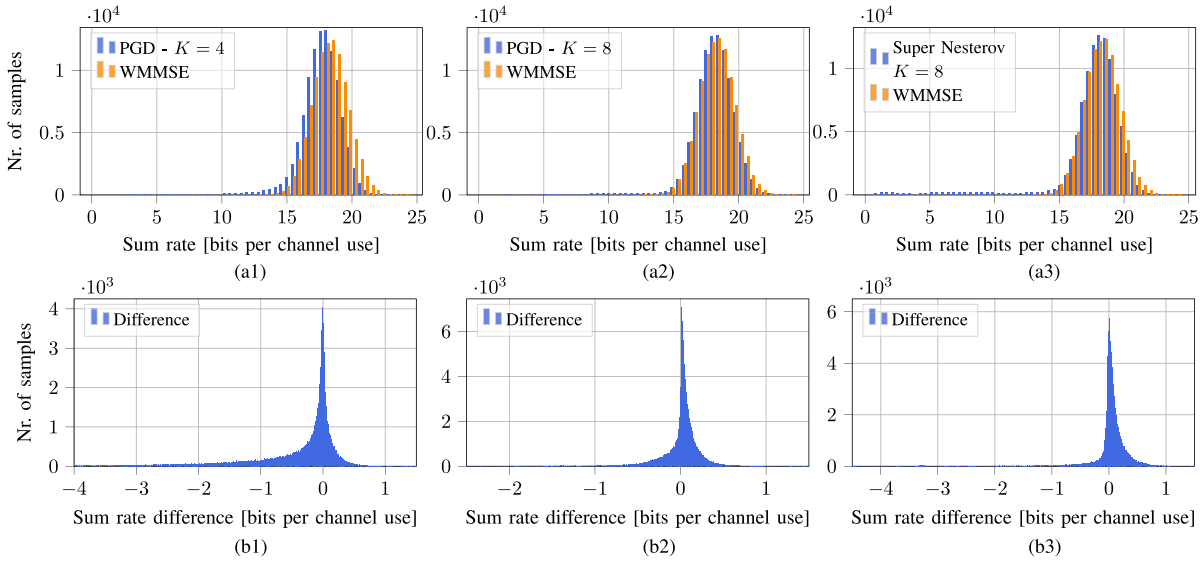

FIGURE 12. Distributions of the WSR achieved by the deep unfolded WMMSE and the truncated WMMSE (a1)–(a3) and the corresponding WSR difference (b1)–(b3), where positive difference implies higher WSR for the deep unfolded WMMSE. The results are obtained for $N = M = 8$, $L = 6$, and $P/\sigma^2 = 10$ dB, while the deep unfolded WMMSE was trained for $N = M = 4$. The average WSR, expressed in bits per channel use, achieved by the truncated WMMSE is 18.293, while the average WSR achieved by the deep unfolded WMMSE is 17.667 (a1), 18.089 (a2), and 17.628 (a3). For reference, the average WSR achieved by the WMMSE at convergence is 19.603, by the ZF is 6.338, and by the RZF is 15.845.

the lightly loaded scenario ($M = 8, N = 4$). Also, it must be mentioned that the end-to-end approaches in the literature [32]–[34] and the IAIDNN [37] are in general not trivially extendable to scenarios with a larger number of antennas. In fact, as the number of transmit antenna increases, the size of the input to the network increases as well, typically leading to a larger number of trainable parameters. Therefore, it is not obvious how to translate the network trained for a given number of antennas to settings with more antennas, except by retraining it. Conversely, our (accelerated) deep unfolded WMMSE is directly applicable to scenarios with a different number of transmit antennas. Clearly, if the number of antennas increases, the size of the input to the network increases as well, implying that the structure of the (accelerated) deep unfolded WMMSE needs to be changed accordingly. However, we do not need to retrain it because the number of trainable parameters remains unchanged, as it does not depend on the number of antennas (see Section VIII-C). Therefore, we can directly transfer the parameters already trained for a different number of antennas to the new network.

IX. CONCLUSION AND FUTURE WORK

From a theoretical standpoint, we provided the matrix-inverse-free unfoldable WMMSE algorithm along with a proof of convergence to a stationary point of the WSR maximization problem. From a practical standpoint, through extensive numerical results, we showed that the (accelerated) deep unfolded WMMSE successfully addresses the complexity versus performance trade-off both in the fully and in the lightly loaded scenarios and exhibits robustness to changes in the channel distribution. The proposed (accelerated) deep unfolded WMMSE benefits from improved explainability and fewer trainable parameters with respect to conventional end-to-end solutions and because it is truly matrix-inverse-free it can fully exploit the potential of standardized and parallelized hardware designed for deep learning.

We explicitly considered the case of a single base station serving multiple single-antenna users, but our approach can be easily extended to the case of multiple base stations by changing the power constraint accordingly. Moreover, our approach is also extendable to more complicated scenarios, e.g., reconfigurable intelligent surface (RIS) aided MU-MISO downlink channels, where the WMMSE algorithm has been used as a subcomponent of an iterative algorithm [59] or as an integrated part of the joint optimization of transmit beamformers and RIS phase shifts [60]. In such cases, although it is possible to apply our approach, the resulting performance would have to be investigated. Such an investigation is outside the scope of this paper, but we leave it as an interesting topic for future work.

Another relevant research direction to explore is the extension of our approach to the MU-MIMO scenario, in which both receive and transmit beamformers are considered. This extension is however not straightforward because the receive processing introduces one matrix inverse in the update of $\boldsymbol{u}$ (7a) and one matrix inverse in the update of $\boldsymbol{w}$ (7b). While the matrix inverse in the update of $\boldsymbol{u}$ can be handled via a gradient descent approach, similarly to how we handled the matrix inverse in the update of $\boldsymbol{V}$ for the MU-MISO case, the matrix inverse introduced for $\boldsymbol{w}$ presents more challenges. Hence, a different methodology than the one presented in the paper needs to be applied. Therefore, we leave the MU-MIMO as a topic for future research. Nevertheless, we believe that our approach, i.e., resorting

to matrix-inverse-free iterative first-order methods, generalizes to a wide range of algorithms that otherwise would be hard to unfold and would not benefit from efficient parallel implementation.

APPENDIX

In this Appendix, we prove Theorem 1 which builds on Lemma 1 and Lemma 2, given below. We first introduce some notation and give the definition of a stationary point in Proposition 1.

In case of a real-valued function f of complex variable $\boldsymbol{x}$, we indicate with f' the equivalent function of the real-valued representation of $\boldsymbol{x}$, denoted by $\boldsymbol{x}'$ and given by the concatenation of $\Re(\boldsymbol{x})$ and $\Im(\boldsymbol{x})$. The properties of f, like continuous differentiability and convexity, hold both over the set of complex variables $\mathcal{X}$ and the set of their equivalent real-valued representation, denoted by $\mathcal{X}'$. Let $\bar{\alpha}$ be the largest user priority in (4), i.e.,

$$\bar{\alpha} = \max_i \alpha_i \tag{17}$$

and let λ be the largest singular value of matrix $\boldsymbol{H}^H\boldsymbol{H}$, i.e.,

$$\lambda = \sigma_{\max}\left(\boldsymbol{H}^H\boldsymbol{H}\right) = \left\|\boldsymbol{H}^H\boldsymbol{H}\right\|. \tag{18}$$

Consequently, we have that $\|\boldsymbol{h}_i\|^2 \leq \lambda \ \ \forall i$.

The relevance of stationary points is made clear by the next proposition, which we restate from [47, Proposition 2.2.1].

Proposition 1 (Necessary Optimality Condition): If $\bar{\boldsymbol{x}}'$ is a local minimum of a continuously differentiable function f' over a convex set $\mathcal{X}'$, then

$$\nabla f'\left(\bar{\boldsymbol{x}}'\right)^T\left(\boldsymbol{x}' - \bar{\boldsymbol{x}}'\right) \geq 0 \ \forall \boldsymbol{x}' \in \mathcal{X}'. \tag{19}$$

Any point $\bar{\boldsymbol{x}}'$ satisfying (19) is referred to as a stationary point.

Before proving that the iterates of the unfoldable WMMSE algorithm converge to a stationary point, we establish in Lemma 1 that there exists a limit point.

Lemma 1: The sequence of iterates generated by the unfoldable WMMSE algorithm has at least one limit point.

Proof: To prove the statement we show that the iterates $\{(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V})^l\}$ generated by the unfoldable WMMSE are confined to a compact set. From constraint (5b) we observe that $\boldsymbol{V}$ is confined to a compact set by construction. From equation (7a) we observe that

$$|u_i| \leq \frac{\left|\boldsymbol{h}_i^H\boldsymbol{v}_i\right|}{\left|\boldsymbol{h}_i^H\boldsymbol{v}_i\right|^2 + \sigma^2} \leq \max_{\xi \geq 0} \frac{\xi}{\xi^2 + \sigma^2} = \frac{1}{2\sigma} \ \ \forall i. \tag{20}$$

From equation (7b) we observe that $\boldsymbol{w}$ is real-valued and, by using (18) and constraint (5b), we have that

$$w_i \leq 1 + \frac{|\boldsymbol{h}_i^H\boldsymbol{v}_i|^2}{\sigma^2} \leq 1 + \frac{P\lambda}{\sigma^2} \ \ \forall i. \tag{21}$$

Therefore

$$1 \leq w_i \leq 1 + \frac{P\lambda}{\sigma^2} \ \ \forall i. \tag{22}$$

The sequences $\{\boldsymbol{u}^l\}$, $\{\boldsymbol{w}^l\}$, and $\{\boldsymbol{V}^l\}$ are hence all confined to compact sets and this implies that the sequence $\{(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V})^l\}$ is confined to a compact set as well. By Bolzano-Weierstrass theorem [61], we conclude that there exists a subsequence $\{(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V})^{l_j}\}$ converging to a limit point $\{(\bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \bar{\boldsymbol{V}})\}$. ■

Now, to proceed further, we need the following result on the cost function of the partial optimization problem over $\boldsymbol{V}$ (11). To avoid cluttering the notation, we first rewrite (5) as

$$\min_{\boldsymbol{u},\boldsymbol{w},\boldsymbol{V}} \ \ \varphi(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V}) \tag{23a}$$

$$\text{s.t.} \ \ \boldsymbol{V} \in \mathcal{C}, \tag{23b}$$

where $\varphi(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V}) \triangleq \sum_{i=1}^N \alpha_i(w_i e_i - \log_2 w_i)$ and where $\mathcal{C} = \{\boldsymbol{V} | \mathrm{Tr}(\boldsymbol{V}\boldsymbol{V}^H) \leq P\}$.

Lemma 2: $\varphi(\boldsymbol{u}^l, \boldsymbol{w}^l, \boldsymbol{V})$, where l is the iteration index and u_i^l and w_i^l are defined by (7a) and (7b), respectively, is an L-smooth function over $\boldsymbol{V}$ with $L \leq \frac{\bar{\alpha}(\sigma^2\lambda + P\lambda^2)}{2\sigma^4}$ for any l and arbitrary fixed u_i^l and w_i^l.

Proof: We define $\varphi^l(\boldsymbol{V}) \triangleq \varphi(\boldsymbol{u}^l, \boldsymbol{w}^l, \boldsymbol{V})$ and recall that

$$\nabla\varphi^l(\boldsymbol{V}) = \left[\nabla\varphi_1^l(\boldsymbol{v}_1), \nabla\varphi_2^l(\boldsymbol{v}_2), \ldots, \nabla\varphi_N^l(\boldsymbol{v}_N)\right]^T, \tag{24}$$

where $\nabla\varphi_i^l(\boldsymbol{v}_i) = -2\alpha_i w_i^l u_i^l \boldsymbol{h}_i + 2\boldsymbol{A}^l\boldsymbol{v}_i$ and

$$\boldsymbol{A}^l = \sum_{i=1}^N \alpha_i w_i^l |u_i^l|^2 \boldsymbol{h}_i\boldsymbol{h}_i^H. \tag{25}$$

By using (17), (20), and (22) and by noting that $\boldsymbol{H}^H\boldsymbol{H} = \sum_{i=1}^N \boldsymbol{h}_i\boldsymbol{h}_i^H$ we have

$$\boldsymbol{A}^l \preceq \frac{\bar{\alpha}(\sigma^2 + P\lambda)}{4\sigma^4}\boldsymbol{H}^H\boldsymbol{H} \ \ \forall l. \tag{26}$$

Therefore

$$\left\|2\boldsymbol{A}^l\right\| \leq \frac{\bar{\alpha}\left(\sigma^2\lambda + P\lambda^2\right)}{2\sigma^4} \ \ \forall l, \tag{27}$$

where we have used that $\|\boldsymbol{H}^H\boldsymbol{H}\| = \lambda$. We define

$$P_A \triangleq \frac{\bar{\alpha}(\sigma^2\lambda + P\lambda^2)}{2\sigma^4}. \tag{28}$$

We can now derive the following upper bound on the L-smooth constant of $\varphi^l(\boldsymbol{V})$ for any l and feasible $\dot{\boldsymbol{v}}$, $\ddot{\boldsymbol{v}}$

$$\left\|\nabla\varphi_i^l(\dot{\boldsymbol{v}}) - \nabla\varphi_i^l(\ddot{\boldsymbol{v}})\right\| = \left\|2\boldsymbol{A}^l(\dot{\boldsymbol{v}} - \ddot{\boldsymbol{v}})\right\| \tag{29a}$$

$$\leq P_{\mathrm{A}}\|\dot{\boldsymbol{v}} - \ddot{\boldsymbol{v}}\| \ \forall \, i, l, \dot{\boldsymbol{v}}, \ddot{\boldsymbol{v}}. \tag{29b}$$

and thus

$$\left\|\nabla\varphi^l(\dot{\boldsymbol{V}}) - \nabla\varphi^l(\ddot{\boldsymbol{V}})\right\|_{\mathrm{F}} \leq P_{\mathrm{A}}\left\|\dot{\boldsymbol{V}} - \ddot{\boldsymbol{V}}\right\|_{\mathrm{F}} \forall \, l, \dot{\boldsymbol{V}}, \ddot{\boldsymbol{V}}. \tag{30}$$

■

Now that we have in place Lemma 1 and Lemma 2 we can give the proof of Theorem 1.

Proof: Let $\boldsymbol{x}$ denote the joint variable $(\boldsymbol{u}, \boldsymbol{w}, \boldsymbol{V})$. Let $\boldsymbol{z}_u^l = (\boldsymbol{u}^{l+1}, \boldsymbol{w}^l, \boldsymbol{V}^l)$, let $\boldsymbol{z}_w^l = (\boldsymbol{u}^{l+1}, \boldsymbol{w}^{l+1}, \boldsymbol{V}^l)$, and let $\boldsymbol{z}_V^l = (\boldsymbol{u}^{l+1}, \boldsymbol{w}^{l+1}, \boldsymbol{V}^{l+1})$, where l is the iteration index.

At any iteration of the unfoldable WMMSE algorithm, we decrease the value of φ in (23). Specifically,

$$\varphi(z_u^l) = \min_{\boldsymbol{\xi}} \varphi(\boldsymbol{\xi}, \boldsymbol{w}^l, \boldsymbol{V}^l) \tag{31}$$

and

$$\varphi\left(z_w^l\right) = \min_{\boldsymbol{\xi}} \varphi\left(\boldsymbol{u}^{l+1}, \boldsymbol{\xi}, \boldsymbol{V}^l\right), \tag{32}$$

imply

$$\varphi\left(z_V^{l-1}\right) \geq \varphi\left(z_u^l\right) \geq \varphi\left(z_w^l\right). \tag{33}$$

As we apply K PGD steps of size $\gamma \leq \frac{2\sigma^4}{\bar{\alpha}(\sigma^2\lambda + P\lambda^2)}$ to

$$\min_{\boldsymbol{\xi}} \ \varphi\left(\boldsymbol{u}^{l+1}, \boldsymbol{w}^{l+1}, \boldsymbol{\xi}\right) \tag{34a}$$

$$\text{s.t.} \ \ \mathrm{Tr}\left(\boldsymbol{\xi}\boldsymbol{\xi}^H\right) \leq P, \tag{34b}$$

where $\varphi(\boldsymbol{u}^{l+1}, \boldsymbol{w}^{l+1}, \boldsymbol{V})$ is L-smooth with $L \leq \frac{\bar{\alpha}(\sigma^2\lambda + P\lambda^2)}{2\sigma^4}$, as stated in Lemma 2, it follows [62] that

$$\varphi\left(\boldsymbol{u}^{l+1}, \boldsymbol{w}^{l+1}, \boldsymbol{V}^{l,k}\right) \geq \varphi\left(\boldsymbol{u}^{l+1}, \boldsymbol{w}^{l+1}, \boldsymbol{V}^{l,k+1}\right), \tag{35}$$

where $\boldsymbol{V}^{l,k}$ is the k^{th} PGD update at the l^{th} iteration. This implies that

$$\varphi\left(z_w^l\right) \geq \varphi\left(z_V^l\right). \tag{36}$$

Combining (33) and (36) yields

$$\varphi\left(z_V^{l-1}\right) \geq \varphi\left(z_u^l\right) \geq \varphi\left(z_w^l\right) \geq \varphi\left(z_V^l\right) \ \forall l. \tag{37}$$

Let $\bar{\boldsymbol{x}} = (\bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \overline{\boldsymbol{V}})$ be a limit point of $\{\boldsymbol{x}^l\}$, as established in Lemma 1. This implies that the sequences $\{\boldsymbol{u}^l\}$, $\{\boldsymbol{w}^l\}$, and $\{\boldsymbol{V}^l\}$ have as limit points $\bar{\boldsymbol{u}}$, $\bar{\boldsymbol{w}}$, and $\overline{\boldsymbol{V}}$, respectively. In order to establish that $\bar{\boldsymbol{x}}$ is a stationary point of (5), we need to show first that $\bar{\boldsymbol{u}}$, $\bar{\boldsymbol{w}}$ and $\overline{\boldsymbol{V}}$ are stationary points of (31), (32), and (34), respectively. For $\boldsymbol{u}$ and $\boldsymbol{w}$ we can verbatim follow the proof of convergence of the block coordinate descent algorithm given by Bertsekas in [47, Proposition 2.7.1] because we solve (31) and (32) optimally, i.e., we can restrict attention to a subsequence $\{l_j\}$ such that

$$\varphi\left(z_u^{l_j}\right) \leq \varphi\left(\boldsymbol{u}, \boldsymbol{w}^{l_j}, \boldsymbol{V}^{l_j}\right) \ \ \forall \boldsymbol{u} \text{ and } \forall j \geq 1 \tag{38}$$

$$\varphi\left(z_w^{l_j}\right) \leq \varphi\left(\boldsymbol{u}^{l_j+1}, \boldsymbol{w}, \boldsymbol{V}^{l_j}\right) \ \ \forall \boldsymbol{w} \text{ and } \forall j \geq 1. \tag{39}$$

Taking the limit for $j \to \infty$, we have that

$$\varphi(\bar{\boldsymbol{x}}) \leq \varphi\left(\boldsymbol{u}, \bar{\boldsymbol{w}}, \overline{\boldsymbol{V}}\right) \ \ \forall \boldsymbol{u}, \tag{40}$$

$$\varphi(\bar{\boldsymbol{x}}) \leq \varphi\left(\bar{\boldsymbol{u}}, \boldsymbol{w}, \overline{\boldsymbol{V}}\right) \ \ \forall \boldsymbol{w}. \tag{41}$$

Since $\varphi(\boldsymbol{u}, \bar{\boldsymbol{w}}, \overline{\boldsymbol{V}})$ and $\varphi(\bar{\boldsymbol{u}}, \boldsymbol{w}, \overline{\boldsymbol{V}})$ are both differentiable in $\boldsymbol{u}$ and $\boldsymbol{w}$, respectively, and the optima are obtained in the interior of the domain, the gradient must vanish at $\bar{\boldsymbol{x}}$, i.e.,

$$\nabla_{\boldsymbol{u}'}\varphi'\left(\bar{\boldsymbol{x}}'\right) = \boldsymbol{0}, \tag{42}$$

$$\nabla_{\boldsymbol{w}'}\varphi'\left(\bar{\boldsymbol{x}}'\right) = \boldsymbol{0}. \tag{43}$$

Thus, $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{w}}$ are stationary points of (31) and (32), respectively. It remains to prove that $\overline{\boldsymbol{V}}$ is a stationary point of (34). Let $g_{\mathcal{C}}(\boldsymbol{V}; \boldsymbol{u}^l, \boldsymbol{w}^l)$ be the gradient mapping defined by

$$g_{\mathcal{C}}\left(\boldsymbol{V}; \boldsymbol{u}^l, \boldsymbol{w}^l\right) = \frac{1}{\gamma}\left(\boldsymbol{V} - \Pi_{\mathcal{C}}\left(\boldsymbol{V} - \gamma\nabla\varphi\left(\boldsymbol{u}^l, \boldsymbol{w}^l, \boldsymbol{V}\right)\right)\right), \tag{44}$$

where $\Pi_{\mathcal{C}}$ is defined in (13). Recall that $\overline{\boldsymbol{V}}$ is a limit point of the sequence $\{\boldsymbol{V}^l\}$, which satisfies

$$\varphi\left(\boldsymbol{u}^l, \boldsymbol{w}^l, \boldsymbol{V}^l\right) \leq \varphi\left(\boldsymbol{u}^l, \boldsymbol{w}^l, \boldsymbol{V}^{l-1}\right) \ \ \forall l. \tag{45}$$

We restrict attention to a subsequence $\{l_j\}$ such that

$$\varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \overline{\boldsymbol{V}}\right) \leq \varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \boldsymbol{V}^{l_j}\right) \ \ \forall j \geq 1. \tag{46}$$

We consider here only a single PGD step, but the following argument can be easily extended to the case of K PGD steps, where K is an arbitrary finite number. Let $\widetilde{\boldsymbol{V}}$ be the next PGD iterate, starting from $\overline{\boldsymbol{V}}$. By using Lemma 2 and by considering that $\varphi(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \boldsymbol{V})$ is convex in $\boldsymbol{V}$, we have from [62, Corollary 2.2.1] that for any $\gamma \leq \frac{2\sigma^4}{\bar{\alpha}(\sigma^2\lambda + P\lambda^2)}$

$$\varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \widetilde{\boldsymbol{V}}\right) \leq \varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \overline{\boldsymbol{V}}\right) - \frac{\gamma\left\|g_{\mathcal{C}}\left(\overline{\boldsymbol{V}}; \boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}\right)\right\|^2}{2}. \tag{47}$$

Therefore, it must hold that $g_{\mathcal{C}}(\overline{\boldsymbol{V}}; \boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}) = 0$ or otherwise the assumption that $\overline{\boldsymbol{V}}$ is a limit point cannot be true as we would have that

$$\varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \widetilde{\boldsymbol{V}}\right) < \varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \overline{\boldsymbol{V}}\right). \tag{48}$$

Further, $g_{\mathcal{C}}(\overline{\boldsymbol{V}}; \boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}) = 0$ implies that

$$\overline{\boldsymbol{V}} = \Pi_{\mathcal{C}}(\overline{\boldsymbol{V}} - \gamma\nabla\varphi(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \overline{\boldsymbol{V}})), \tag{49}$$

which, in turn, implies that $\overline{\boldsymbol{V}}$ is a fixed point of the PGD iterations. Therefore $\overline{\boldsymbol{V}}$ has to be an optimal point of $\varphi(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \boldsymbol{V})$ over $\mathcal{C}$ because for convex and L-smooth functions the PGD is known to converge to an optimal point [62]. Hence, we can state that

$$\varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \overline{\boldsymbol{V}}\right) \leq \varphi\left(\boldsymbol{u}^{l_j}, \boldsymbol{w}^{l_j}, \boldsymbol{V}\right) \ \ \forall \boldsymbol{V} \in \mathcal{C}, \ \forall j \geq 1, \tag{50}$$

and taking the limit $j \to \infty$, we have that

$$\varphi(\bar{\boldsymbol{x}}) \leq \varphi(\bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \boldsymbol{V}) \ \ \forall \boldsymbol{V} \in \mathcal{C}. \tag{51}$$

From Proposition 1 it follows that

$$\nabla_{\boldsymbol{V}'}\varphi'(\bar{\boldsymbol{x}}')^T\left(\boldsymbol{V}' - \overline{\boldsymbol{V}}'\right) \geq 0 \ \ \forall \boldsymbol{V}' \in \mathcal{C}'. \tag{52}$$

Combining inequalities (42), (43), and (52), we conclude that $\bar{\boldsymbol{x}}$ is a stationary point of (5), i.e.,

$$\nabla\varphi'(\bar{\boldsymbol{x}}')^T(\boldsymbol{x}' - \bar{\boldsymbol{x}}') \geq 0 \ \ \forall \boldsymbol{x}' \in \mathcal{C}'. \tag{53}$$

It remains to show that $\overline{\boldsymbol{V}}$ is a stationary point of (4) if and only if $\bar{\boldsymbol{x}} = (\bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \overline{\boldsymbol{V}})$ is a stationary point of (5) for some $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{w}}$. To this end, we can verbatim follow the second part of the proof given by Shi *et al.* in [16, Th. 3], as the relation between problems (4) and (5) does not depend on the specific algorithm used to solve (5). ■

REFERENCES

[1] P. von Butovitsch *et al.*, "Advanced antenna systems for 5G networks (white paper)," Stockholm, Sweden, Ericsson, White Paper, 2020.

[2] "5G systems enabling the transformation of industry and society," Ericsson, Stockholm, Sweden, White Paper, 2017.

[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] Z.-Q. Luo and S. Zhang, "Dynamic spectrum management: Complexity and duality," *IEEE J. Sel. Topics Signal Process.*, vol. 2, no. 1, pp. 57–73, Feb. 2008.

[5] S. K. Joshi, P. C. Weeraddana, M. Codreanu, and M. Latva-Aho, "Weighted sum-rate maximization for MISO downlink cellular networks via branch and bound," *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 2090–2095, Apr. 2012.

[6] L. Liu, R. Zhang, and K.-C. Chua, "Achieving global optimality for weighted sum-rate maximization in the K-user Gaussian interference channel with multiple antennas," *IEEE Trans. Wireless Commun.*, vol. 11, no. 5, pp. 1933–1945, May 2012.

[7] E. Björnson, G. Zheng, M. Bengtsson, and B. Ottersten, "Robust monotonic optimization framework for multicell MISO systems," *IEEE Trans. Signal Process.*, vol. 60, no. 5, pp. 2508–2523, May 2012.

[8] P. Zetterberg and B. Ottersten, "The spectrum efficiency of a base station antenna array system for spatially selective transmission," *IEEE Trans. Veh. Technol.*, vol. 44, no. 3, pp. 651–660, Aug. 1995.

[9] T. K. Y. Lo, "Maximum ratio transmission," *IEEE Trans. Commun.*, vol. 47, no. 10, pp. 1458–1461, Oct. 1999.

[10] M. Joham, W. Utschick, and J. A. Nossek, "Linear transmit processing in MIMO communications systems," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2700–2712, Aug. 2005.

[11] C. T. K. Ng and H. Huang, "Linear precoding in cooperative MIMO cellular networks with limited coordination clusters," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 9, pp. 1446–1454, Dec. 2010.

[12] L.-N. Tran, M. F. Hanif, A. Tolli, and M. Juntti, "Fast converging algorithm for weighted sum rate maximization in multicell MISO downlink," *IEEE Signal Process. Lett.*, vol. 19, no. 12, pp. 872–875, Dec. 2012.

[13] M. G. Kibria, H. Murata, and S. Yoshida, "Coordinated linear precoding in downlink multicell MU-MISO OFDMA networks," in *Proc. IEEE 78th Veh. Technol. Conf. (VTC Fall)*, Las Vegas, NV, USA, 2013, pp. 1–5.

[14] D. H. N. Nguyen and T. Le-Ngoc, "Sum-rate maximization in the multicell MIMO multiple-access channel with interference coordination," *IEEE Trans. Wireless Commun.*, vol. 13, no. 1, pp. 6–48, Jan. 2014.

[15] R. Brandt and M. Bengtsson, "Fast-convergent distributed coordinated precoding for TDD multicell MIMO systems," in *Proc. IEEE 6th Int. Workshop Comput. Adv. Multi-Sens. Adapt. Process. (CAMSAP)*, Cancun, Mexico, Dec. 2015, pp. 457–460.

[16] 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. Signal Process.*, vol. 59, no. 9, pp. 4331–4340, Sep. 2011.

[17] S. S. Christensen, R. Agarwal, E. De Carvalho, and J. M. Cioffi, "Weighted sum-rate maximization using weighted MMSE for MIMO-BC beamforming design," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 4792–4799, Dec. 2008.

[18] D. A. Schmidt, C. Shi, R. A. Berry, M. L. Honig, and W. Utschick, "Minimum mean squared error interference alignment," in *Proc. Conf. Rec. 43rd Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, USA, 2009, pp. 1106–1110.

[19] H. Ghauch, T. Kim, M. Bengtsson, and M. Skoglund, "Sum-rate maximization in sub-28-GHz millimeter-wave MIMO interfering networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 7, pp. 1649–1662, Jul. 2017.

[20] D. H. N. Nguyen and T. Le-Ngoc, "MMSE precoding for multiuser MISO downlink transmission with non-homogeneous user SNR conditions," *EURASIP J. Adv. Signal Process.*, vol. 2014, Jun. 2014, Art. no. 85.

[21] F. Sun and E. de Carvalho, "Weighted MMSE beamforming design for weighted sum-rate maximization in coordinated multi-cell MIMO systems," in *Proc. IEEE Veh. Technol. Conf.*, 2012, pp. 1–5.

[22] Z. Qin, H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep learning in physical layer communications," *IEEE Wireless Commun.*, vol. 26, no. 2, pp. 93–99, Apr. 2019.

[23] S. Dörner, S. Cammerer, J. Hoydis, and S. T. Brink, "Deep learning based communication over the air," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 132–143, Feb. 2018.

[24] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2224–2287, 3rd Quart., 2019.

[25] D. Gündüz, P. de Kerret, N. D. Sidiropoulos, D. Gesbert, C. R. Murthy, and M. van der Schaar, "Machine learning in the air," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2184–2199, Oct. 2019.

[26] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[27] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3039–3071, 4th Quart., 2019.

[28] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Oct. 2019.

[29] H. Huang, Y. Song, J. Yang, G. Gui, and F. Adachi, "Deep-learning-based millimeter-wave massive MIMO for hybrid precoding," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3027–3032, Mar. 2019.

[30] J. Zhang, W. Xia, M. You, G. Zheng, S. Lambotharan, and K.-K. Wong, "Deep learning enabled optimization of downlink beamforming under per-antenna power constraints: Algorithms and experimental demonstration," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3738–3752, Jun. 2020.

[31] T. Lin and Y. Zhu, "Beamforming design for large-scale antenna arrays using deep learning," *IEEE Wireless Commun. Lett.*, vol. 9, no. 1, pp. 103–107, Jan. 2020.

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

[33] W. Xia, G. Zheng, Y. Zhu, J. Zhang, J. Wang, and A. P. Petropulu, "A deep learning framework for optimization of MISO downlink beamforming," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1866–1880, Mar. 2020.

[34] H. Huang, W. Xia, J. Xiong, J. Yang, G. Zheng, and X. Zhu, "Unsupervised learning-based fast beamforming design for downlink MIMO," *IEEE Access*, vol. 7, pp. 7599–7605, 2019.

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

[36] L. Pellaco, M. Bengtsson, and J. Jaldén, "Deep weighted MMSE downlink beamforming," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Toronto, ON, Canada, 2021, pp. 4915–4919.

[37] 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.

[38] M. Abadi *et al.* "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems." 2015. [Online]. Available: http://tensorflow.org/

[39] A. Chowdhury, G. Verma, C. Rao, A. Swami, and S. Segarra, "Efficient power allocation using graph neural networks and deep algorithm unfolding," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Toronto, ON, Canada, 2021, pp. 4725–4729.

[40] Y. Nesterov, "A method for solving a convex programming problem with convergence rate $O(1/k^2)$," *Soviet Mathematics Doklady*, vol. 27, no. 2, pp. 372–367, 1983.

[41] S. A. H. Hosseini, B. Yaman, S. Moeller, M. Hong, and M. Akçakaya, "Dense recurrent neural networks for accelerated MRI: History-cognizant unrolling of optimization algorithms," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 6, pp. 1280–1291, Oct. 2020.

[42] Y. Li *et al.*, "End-to-end video compressive sensing using Anderson-accelerated unrolled networks," in *Proc. IEEE Int. Conf. Comput. Photogr. (ICCP)*, St. Louis, MO, USA, 2020, pp. 1–12.

[43] P. del Aguila Pla, V. Saxena, and J. Jaldén, "Spotnet-learned iterations for cell detection in image-based immunoassays," in *Proc. IEEE 16th Int. Symp. Biomed. Imag. (ISBI)*, Venice, Italy, 2019, pp. 1023–1027.
[44] G. Nakerst, J. Brennan, and M. Haque, "Gradient descent with momentum—To accelerate or to super-accelerate?" Jan. 2020, *arXiv:2001.06472*.
[45] L. Pellaco. "WMMSE-Deep-Unfolding." 2021. [Online]. Available: https://github.com/lpkg/WMMSE-deep-unfolding
[46] A. B. Gershman, N. D. Sidiropoulos, S. Shahbazpanahi, M. Bengtsson, and B. Ottersten, "Convex optimization-based beamforming," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 62–75, May 2010.
[47] D. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA, USA: Athena Sci., 1999.
[48] R. L. Burden and J. D. Faires, *Numerical Analysis*, 4th ed. Boston, MA, USA: PWS-Kent Publ. Company, 1989.
[49] D. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods* (Optimization and Neural Computation Series), 1st ed. Belmont, MA, USA: Athena Sci., 1996.
[50] H. Hojatian, J. Nadal, J.-F. Frigon, and F. Leduc-Primeau, "Unsupervised deep learning for massive MIMO hybrid beamforming," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7086–7099, Nov. 2021.
[51] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Stat.*, 2010, pp. 249–256.
[52] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 1–9.
[53] W. Samek, T. Wiegand, and K.-R. Müller, 'Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," Aug. 2017, *arXiv: 1708.08296*.
[54] E. Björnson and P. Giselsson, "Two applications of deep learning in the physical layer of communication systems [lecture notes]," *IEEE Signal Process. Mag.*, vol. 37, no. 5, pp. 134–140, Sep. 2020.
[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, San Diego, CA, USA, May 2015, pp. 1–14.
[56] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: Johns Hopkins Univ. Press, 1996.
[57] J. W. Demmel, *Applied Numerical Linear Algebra*. Philadelphia, PA, USA: SIAM, Jan. 1997.
[58] "DeepUnfolding_WMMSE." [Online]. Available: https://github.com/hqyyqh888/DeepUnfolding_WMMSE (Accessed: Sep. 2021).
[59] H. Guo, Y.-C. Liang, J. Chen, and E. G. Larsson, "Weighted sum-rate maximization for reconfigurable intelligent surface aided wireless networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3064–3076, May 2020.
[60] M. Zhang, L. Tan, K. Huang, and L. You, "On the trade-off between energy efficiency and spectral efficiency in RIS-aided multi-user MISO downlink," *Electronics*, vol. 10, no. 11, p. 1307, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/11/1307
[61] R. Bartle and D. Sherbert, *Introduction to Real Analysis*, 4th ed. Hoboken, NJ, USA: Wiley, 2011.
[62] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, 1st ed. Boston, MA, USA: Springer Publ. Company, Incorp., 2004.