Deep Unfolded Superposition Coding Optimization for Two-Hop NOMA MANETs

Tomer Alter and Nir Shlezinger

***Abstract*—Mobile ad hoc network (MANET) is an attractive technology for tactical communications. When combined with non-orthogonal multiple access (NOMA), it can support a large number of devices over flexible topologies in a spectral efficient manner. However, to harness the benefits of NOMA MANETs, one should determine the transmission protocol, particularly the superposition code, which involves a lengthy optimization procedure that has to be repeated when the topology changes. In this work, we tackle this challenge by proposing an optimizer for rapidly tuning superposition codes for two-hop NOMA MANETs. This is achieved using the emerging *deep unfolding* methodology, leveraging data to boost reliable settings with a fixed and limited latency. We unfold optimization based on a small number of projected gradient steps applied to the minimal rate objective, combine it with an ensemble to cope with non-convexity and use machine learning tools to tune the hyperparameters of the optimizers in an unsupervised manner. Our numerical results demonstrate that the proposed method enables the rapid setting of high-rate superposition codes for various channels.**

***Index terms*— NOMA, MANET, deep unfolding.**

I. Introduction

Military tactical communications are subject to strict demands in terms of connectivity, robustness, mobility, security, and covertness [1]. In order to meet these requirements, tactical networks often deviate from the uplink/downlink star topology operation of conventional wireless communications, e.g., cellular networks. Among the leading approaches for realizing dynamic and flexible tactical networks is the mobile ad hoc network (MANET) technology [2], where communication links are established during operation, possibly on demand and over multi-hop routes, allowing the communicating entities to collaborate in conveying the desired messages.

A significant challenge in multi-user communications, which is even more substantial in MANETs than conventional topologies, is the treatment of cross interference. Such interference naturally arises from the fact that multiple users share the same temporal and spectral channel resources. The common strategy boosts orthogonality among the communicating entities, either via division of the spectral and temporal resources or alternatively via media access protocols designed to prevent collisions. Recent years have witnessed a growing interest in transiting from orthogonality-based communications to non-orthogonal multiple access (NOMA) solutions [3], [4]. NOMA techniques allow users to simultaneously share the wireless channel resources for supporting heterogeneous end-devices, albeit this inevitably imposes interference.

Various receive methods have been proposed for enabling symbol detection in the presence of interference [5]. In non-orthogonal broadcast channels (BCs), where a set of superimposed messages are transmitted to different users over a shared channel, the successive interference cancellation (SIC) algorithm has been shown to be particularly suitable. This is a benefit of its ability to approach the achievable rate region of such channels when combined with superposition coding [3], [4], whilst its complexity only grows linearly with the number of users. Similar interference handling approaches were also proposed for multiple access channels (MACs), see [6], [7].

The theoretical benefits of NOMA are widely studied in the context of conventional uplink/downlink wireless communications, based on BC/MAC modeling, respectively. Moreover, recent studies have identified that these gains can also be harnessed in MANETs [8]. Nonetheless, harnessing these gains requires the setting of the superposition code for each channel realization, which often varies rapidly, i.e., the setting has to be repeated frequently. This procedure involves lengthy optimization even for conventional topologies, with recent studies proposing optimization methods [9] and the usage of machine learning architectures [10]. However, the setting of superposition codes is notably more complicated in MANETs, motivating the design of efficient optimization techniques for rapidly and reliably configuring NOMA MANETs.

In this work, we propose Unfolded PGDNet, which rapidly translates channel state information (CSI) into superposition code for two-hop NOMA-MANETs. We derive our method by formulating the superposition code design as a non-convex constrained optimization problem aiming at maximizing the minimal rate in the network. We then leverage emerging learn-to-optimize methodologies [11], and particularly deep unfolding [12], [13], to learn from data how to enable projected gradient steps to rapidly recover a suitable configuration. We then cope with the non-convexity of the problem by proposing an ensemble method, which identifies multiple solutions and selects the one with the best minimal rate. Our numerical results demonstrate that the proposed Unfolded PGDNet rapidly produces solid superposition code configurations, reducing the latency compared with conventional optimization by factors varying from $10\times$ to over $80\times$.

The rest of this work is organized as follows: Section II describes the system model; the proposed learn-to-optimize algorithm for configuring NOMA-MANETs is detailed in Section III, and is numerically evaluated in Section IV. Finally, Section V provides concluding remarks.

II. System Model

A. Communication System Model

We consider a two-hop scalar MANET with a single transmitter, M relaying nodes, and N receiver nodes (end-users). There are no direct links between the transmitter and the

The authors are with the School of ECE, Ben-Gurion University of the Negev (e-mail: tomeralt@post.bgu.ac.il; nirshl@bgu.ac.il).

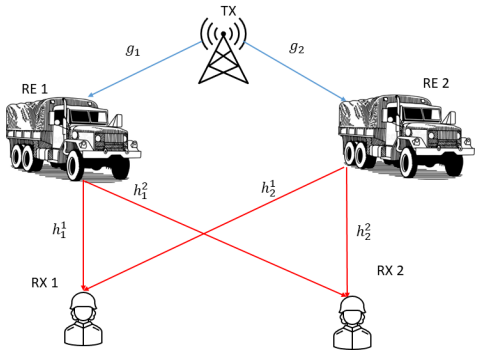

Fig. 1. Two-hop MANET with $M = 2$ relays and $N = 2$ receivers.

receivers, and communication is achieved only via the relays. An example topology with $M = N = 2$ is illustrated in Fig. 1.

The communicating devices employ *power domain NOMA* [4]. Accordingly, letting s_n denote the signal intended for the nth user, $n \in \{1, \ldots, N\} \triangleq \mathcal{N}$, the transmitter generates a linear combination of $\{s_n\}$ via *superposition coding* weighted by the non-negative power coefficients $\{\varphi_n\}_{n\in\mathcal{N}}$. Letting g_m denote channel coefficient between the transmitter and the ith relay, $i \in \{1, \ldots, M\} \triangleq \mathcal{M}$, the signal received by the mth relay is given by

$$y_m^{\rm re} = g_m \sum_{n=1}^{N} \varphi_n s_n + w_m, \tag{1}$$

where w_m is additive white Gaussian noise (AWGN) with variance $\sigma_{\rm re}^2$.

The relays operate in a *decode-and-forward* manner [14, Ch. 16], aiming to recover $\{s_n\}$ from their corresponding channel output (1). Then, $\{s_n\}$ are relayed to the receivers with each relay employing a dedicated superposition code, where we use $\{p_{m,n}\}_{n\in\mathcal{N}}$ to denote the power coefficients of the mth relay. Letting $h_m^{(i)}$ be the channel from the mth relay to the ith receiver, the signal received at the ith receiver is obtained as

$$y_i^{\rm rx} = \sum_{m=1}^{M} h_m^{(i)} \sum_{n=1}^{N} p_{m,n} s_n + v_i, \tag{2}$$

where v_i is AWGN with variance $\sigma_{\rm rx}^2$. The variables defined above are summarized in Table I. We assume full CSI, i.e., the channel coefficients are known.

Decoding at both the relays and the end-users is carried out using SIC, whose combination with superposition coding is known to approach the capacity region of single-hop NOMA systems [4]. SIC operates in an iterative fashion: first the signal with the highest signal-to-interference-and-noise ratio (SINR) is being decoded while treating all the other signals as noise. Then, it is subtracted from the superimposed message and the signal with the second largest SINR in the same fashion. This procedure is carried out over all N messages at the relays (which decode all messages for decode-and-forward), while the ith end-user terminates SIC when decoding s_i.

TABLE I
LIST OF VARIABLES AND PARAMETERS

Symbol	Definition
v_i, w_m	i.i.d AWGN noise signals
g_m	transmitter - mth relay channel
$h_m^{(i)}$	mth relay - ith receiver channel
s_n	signal intended for nth receiver
φ_n	transmitter's power allocation to nth signal
$p_{m,n}$	mth relay's power allocation to nth signal

B. Problem Formulation

The communication model for the considered NOMA-MANET is formulated in Subsection II-A for arbitrary superposition code, dictated by the power coefficients $\{\varphi_n\}$ and $\{p_{m,n}\}$. However, the exact setting of these coefficients notably affects the achievable communication rate in the MANET. In particular, by (1) the nth message is recoverable by the mth relay (with arbitrarily small error probability) if it is coded with a rate (in bits per channel use), not larger than [15, Ch. 6]

$$R_{m,n}^{\rm re} = \log_2\left(1 + \frac{|g_m|^2\varphi_n^2}{|g_m|^2 \sum\limits_{i\in\mathcal{N}/n:\varphi_i\geq\varphi_n} \varphi_i^2 + \sigma_{\rm re}^2}\right). \tag{3}$$

The message also has to be recoverable by the lth user, where in particular the lth user recovers all messages preceding and including its intended message via SIC. By (2), the nth message can be recovered by the $l$the user when its rate is not larger than

$$R_{l,n}^{\rm rx} = \log_2\left(1 + \frac{\tilde{g}_n^{(l)}}{\sum\limits_{i\in\mathcal{N}/n:\tilde{g}_i^{(l)}\geq\tilde{g}_n^{(l)}} \tilde{g}_i^{(l)} + \sigma_{\rm rx}^2}\right), \tag{4}$$

where we defined

$$\tilde{g}_n^{(i)} \triangleq \left|\sum_{m=1}^{M} h_m^{(i)} p_{m,n}\right|^2, \qquad n, i \in \mathcal{N}. \tag{5}$$

The rate expressions in (3) and (4) depends on the superposition coding coefficients, written henceforth as the $M+1\times N$ matrix $\boldsymbol{P}$, whose entries are given by $[\boldsymbol{P}]_{m,n} = p_{m,n}$ for $m \in \mathcal{M}$ and $[\boldsymbol{P}]_{M+1,n} = \varphi_n$. In particular, the rate supported for the nth message is given by

$$R_n(\boldsymbol{P}) = \min\left\{\min_{l\in\mathcal{N}:\tilde{g}_n^{(l)}\geq\tilde{g}_l^{(l)}} R_{l,n}^{\rm rx}, \min_{m\in\mathcal{M}} R_{m,n}^{\rm re}\right\}. \tag{6}$$

The minimization of the first term in (6) over indices (n, l) satisfying $\tilde{g}_n^{(l)} \geq \tilde{g}_l^{(l)}$ (which includes $n = l$, i.e., the intended message) follows the SIC operation at the end-users, namely, from the fact that the end-users only recover messages with SINR not larger than their intended message.

Our aim is to set the superposition coding coefficients $\boldsymbol{P}$ to maximize the *minimal rate*, i.e., the throughput one can guarantee for all end-users, being a key performance measure

in tactical networks. Without loss of generality, we assume a unit power constraint on each of the transmitting devices, i.e., $\boldsymbol{P} \in \mathcal{P}$, where $\mathcal{P}$ is defined as

$$\mathcal{P} = \left\{ \boldsymbol{P} \in [0,1]^{M+1\times N} : \left\| [\boldsymbol{P}]_{m,:} \right\|_2 = 1; \forall m \right\}. \quad (7)$$

Accordingly, our goal is to map the CSI, i.e., knowledge of $\boldsymbol{h} \triangleq \left\{ \{g_m\}, \{h_m^{(i)}\} \right\}$ and $\boldsymbol{\sigma} = \{\sigma_{\rm rx}, \sigma_{\rm re}\}$, into

$$\boldsymbol{P}^\star = \arg\max_{\boldsymbol{P}\in\mathcal{P}} \min_{n\in\mathcal{N}} R_n(\boldsymbol{P}). \quad (8)$$

The superposition code design problem formulated in (8) gives rise to two main challenges:

- C1 The optimization problem is non-convex, thus recovering the optimal $\boldsymbol{P}^\star$. Moreover, even finding a good set of $\boldsymbol{P}$ (in the sense of maximizing the minimal rate) is likely to involve lengthy iterative solvers.
- C2 The superposition code has to be set anew each time the CSI changes, thus the design has to be carried out rapidly, e.g., within a small and fixed number of iterative steps.

To facilitate tackling (8) while coping with C1-C2, we assume access to T past channel realizations that are available offline, i.e., when designing the solver. Such data can be obtained from past measurements and is denoted by

$$\mathcal{D} \triangleq \{\boldsymbol{h}_t, \boldsymbol{\sigma}_t\}_{t=1}^T. \quad (9)$$

III. Unfolded Superposition Coding Optimization

In this section, we present our proposed data-aided optimizer for the rapid superposition code design. To gradually derive our method, we first formulate the projected gradient descent (PGD) steps for tackling (8) in Subsection III-A. While PGD is suitable for convex problems [16, Ch. 9] and typically requires many iterations to converge, and is thus not tailored to tackle C1-C2, it is used as our basis optimizer due to its simplicity and interpretability. Then, in Subsection III-B we convert PGD into PGDNet, leveraging the data (9) to learn to rapidly optimize via deep unfolding methodology [12], [13] (thus tackling C2), and extend it to cope with C1 using an ensemble of PGDNets. We provide a discussion in Subsection III-C.

A. Projected Gradient Descent for NOMA MANETs

Problem (8) represents constrained maximization with $(N+1)\cdot M$ optimization variables. While the problem is non-convex, one can possibly identify a suitable setting the superposition code parameters $\boldsymbol{P}$ by applying iterative PGD steps starting from some suitable initialization $\boldsymbol{P}^{(0)}$. The resulting iterative procedure at iteration index k is given by

$$\boldsymbol{P}^{(k+1)} = \Pi_{\mathcal{P}}\left(\boldsymbol{P}^{(k)} + \mu^{(k)} \nabla_{\boldsymbol{P}} \min_{n\in\mathcal{N}} R_n\big(\boldsymbol{P}^{(k)}\big) \right), \quad (10)$$

where $\Pi_{\mathcal{P}}(\cdot)$ denotes the projection operator onto $\mathcal{P}$. As the projection operator onto the simplex typically involves additional iterative procedures [17], we approximate it using the following operator (which guarantees that the projection of any matrix with at least one positive entry per column indeed lies in $\mathcal{P}$ by (7)):

$$\tilde{\boldsymbol{P}} = \Pi_{\mathcal{P}}(\boldsymbol{P}) \Leftrightarrow [\tilde{\boldsymbol{P}}]_{m,:} = \frac{1}{\left\| [\boldsymbol{P}]_{m,:}^+ \right\|_2} [\boldsymbol{P}]_{m,:}^+, \quad (11)$$

where $(\cdot)^+$ is the positive part operator, i.e., $a^+ \triangleq \max(0,a)$, applied element-wise.

To implement PGD, the gradients of the objective function are required. These are stated in the following lemma.

Lemma 1. *For a given $n \in \mathcal{N}$, let $m_1 \in \mathcal{M}$ and $l_1 \in \mathcal{N}$ be the indices holding $R^{\rm re}_{m_1,n} = \min_{m\in\mathcal{M}} R^{\rm re}_{m,n}$ and $R^{\rm rx}_{l_1,n} = \min_{l\in\mathcal{N}:\tilde{g}_n^{(l)} \geq \tilde{g}_l^{(l)}} R^{\rm rx}_{l,n}$, respectively. Then, for each $t \in \mathcal{N}$ and $s \in \mathcal{M}$ it holds that*

$$\frac{\partial R_n}{\partial \varphi_t} = \begin{cases} \frac{\partial R^{\rm re}_{m_1,n}}{\partial \varphi_t} & R^{\rm re}_{m_1,n} < R^{\rm rx}_{l_1,n} \\ 0 & \text{else} \end{cases}, \quad (12a)$$

$$\frac{\partial R_n}{\partial p_{s,t}} = \begin{cases} 0 & R^{\rm re}_{m_1,n} < R^{\rm rx}_{l_1,n} \\ \frac{\partial R^{\rm rx}_{l_1,n}}{\partial p_{s,t}} & \text{else} \end{cases}, \quad (12b)$$

where the gradients of $R^{\rm re}_{m,n}, R^{\rm rx}_{l,n}$ are given in (13).

Proof: Recall that for any two functions $f_1(x), f_2(x)$ it holds that $\frac{d}{dx}\min(f_1(x), f_2(x)) = \frac{d}{dx} f_1(x)$ for every x such that $f_1(x) < f_2(x)$. Consequently, computing the objective gradients boils down to (12) while (13) is obtained by taking the gradients of (3) and (5). ∎

The resulting optimization method with K iterations is summarized as Algorithm 1.

Algorithm 1: $\mathbf{PGD}_K(\cdot; \boldsymbol{P}^{(0)}, \boldsymbol{\mu})$

Init: Iterations K; Step sizes $\boldsymbol{\mu} = \{\mu^{(k)}\}_{k=0}^{K-1}$; Initial guess $\boldsymbol{P}^{(0)}$

Input: CSI $\boldsymbol{h}, \boldsymbol{\sigma}$

for $k = 0, 1, \ldots K$ **do**

  Calculate $\nabla_{\boldsymbol{P}} \min_{n\in\mathcal{N}} R_n\big(\boldsymbol{P}\big)$ using (12);

  Update $\boldsymbol{P}^{(k+1)}$ via (10) ;

return $\boldsymbol{P}^{(K)}$

B. Unfolded PGDNet

Algorithm 1 optimizes the power allocation for a given channel realization. However, its convergence speed highly depends on the hyperparameters, i.e., the step sizes $\boldsymbol{\mu}$, and the resulting rate can be notably affected by the initial guess $\boldsymbol{P}^{(0)}$ due to the non-convexity. We thus propose a data-aided implementation of PGD, termed *Unfolded PGDNet*. We derive our method based on the following steps: We first tackle, C2 converting PGD into a fixed-latency discriminative trainable architecture [18] via deep unfolding methodology [13], and propose dedicated training algorithm. Then, we extend it into an ensemble of multiple parallel models that converge to different local optima to cope with C1.

1) PGDNet Architecture: Our solution is based on deep unfolding [13], which converts iterative optimizers with a

$$\frac{\partial R_{m,n}^{\mathrm{re}}}{\partial \varphi_t} = \frac{1}{\ln 2} \begin{cases} 0 & \varphi_t > \varphi_n \\ \dfrac{2|g_m|^2 \varphi_n}{|g_m|^2 \sum\limits_{i\in\mathcal{N}/n:\varphi_i>\varphi_n} \varphi_i^2 + \sigma_{\mathrm{re}}^2} & \varphi_t = \varphi_n \\ \dfrac{-2|g_m|^4 \varphi_n^2 \varphi_t}{|g_m|^2 \sum\limits_{i\in\mathcal{N}/n:\varphi_i\geq\varphi_n} \varphi_i^2 + \sigma_{\mathrm{re}}^2} \dfrac{1}{|g_m|^2 \sum\limits_{i\in\mathcal{N}/n:\varphi_i>\varphi_n} \varphi_i^2 + \sigma_{\mathrm{re}}^2} & \varphi_t < \varphi_n \end{cases} \tag{13a}$$

$$\frac{\partial R_{l,n}^{\mathrm{rx}}}{\partial p_{s,t}} = \frac{1}{\ln 2} \begin{cases} 0 & \tilde{g}_t^{(l)} > \tilde{g}_n^{(l)} \\ \dfrac{2|h_s^{(l)}|^2 p_{s,t}}{\sum\limits_{i\in\mathcal{N}/n:\tilde{g}_i^{(l)}>\tilde{g}_n^{(l)}} \tilde{g}_i^{(l)} + \sigma_{\mathrm{rx}}^2} & \tilde{g}_t^{(l)} = \tilde{g}_n^{(l)} \\ \dfrac{-2|h_s^{(l)}|^2 p_{s,t} \tilde{g}_n^{(l)}}{\sum\limits_{i\in\mathcal{N}/n:\tilde{g}_i^{(l)}\geq\tilde{g}_n^{(l)}} \tilde{g}_i^{(l)} + \sigma_{\mathrm{rx}}^2} \dfrac{1}{\sum\limits_{i\in\mathcal{N}/n:\tilde{g}_i^{(l)}>\tilde{g}_n^{(l)}} \tilde{g}_i^{(l)} + \sigma_{\mathrm{rx}}^2} & \tilde{g}_t^{(l)} < \tilde{g}_n^{(l)} \end{cases} \tag{13b}$$

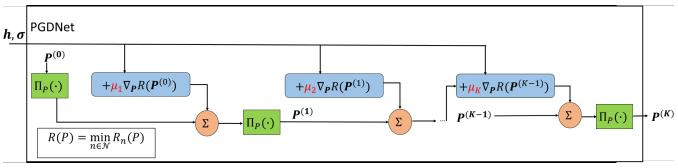

Fig. 2. PGDNet architecture; trainable parameters are marked in red.

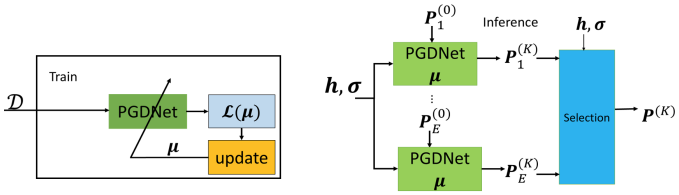

Fig. 3. Unfolded PGDNet in training (left) and inference (right).

predetermined number of iterations into machine learning models. The proposed PGDNet applies PGD (Algorithm 1) with a predefined (and small) number of iterations K. By doing that, we guarantee an a-priori known, and considerably low, run-time. While the accuracy of first-order optimizers such as PGD is usually invariant of hyperparameters setting when allowed to run until convergence (as long as the step-sizes are sufficiently small) [16], its performance is largely affected by these settings when there is a fixed number of iterations.

In PGDNet, we treat the hyperparameters of the iterative optimizer, i.e., $\boldsymbol{\mu}$, as trainable parameters. The resulting architecture can be viewed as a form of a deep neural network (DNN) with K layers: Each layer of index k implements a single PGD iteration, and has a single trainable parameter, which is the step-size μ_k. The resulting architecture is illustrated in Fig. 2. As in Algorithm 1, we denote its operation for CSI $\boldsymbol{h}, \boldsymbol{\sigma}$ with hyperparameters $\boldsymbol{\mu}$ and initial guess $\boldsymbol{P}^{(0)}$ as $\mathbf{PGD}_K(\boldsymbol{h}, \boldsymbol{\sigma}; \boldsymbol{P}^{(0)}, \boldsymbol{\mu})$.

2) Training PGDNet: The training procedure aims to set the hyperparameters $\boldsymbol{\mu}$ to make PGDNet maximize the minimal rate for the channel available in the data set $\mathcal{D}$. As in [19], we also exploit the interpretability of the trainable architecture to facilitate and regularize the training procedure. Specifically, we account for the fact that the features exchanged between layers correspond to superposition coding coefficients, i.e., $\boldsymbol{P}^{(k)}$ for the kth layer, and encourage the model to learn gradually improved settings. Accordingly, by writing $\mathbf{PGD}_K^{(k)}(\cdot;\cdot)$ as the output of the kth iteration of PGDNet, i.e., $\boldsymbol{P}^{(k)}$, we set the loss function to

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\mu}) = \frac{-1}{|\mathcal{D}|} \sum_{(\boldsymbol{h},\boldsymbol{\sigma})\in\mathcal{D}} \sum_{k=1}^{K} \log_2(1+k) \times \min_{n\in\mathcal{N}} R_n \left(\mathbf{PGD}_K^{(k)}(\boldsymbol{h}, \boldsymbol{\sigma}; \boldsymbol{P}^{(0)}, \boldsymbol{\mu}) \right). \tag{14}$$

Note that (14) enables *unsupervised learning*, i.e., there is no need to provide 'ground truth' power allocation. This stems from the task being an optimization problem, for which one can evaluate each setting of the optimization variables.

The learn-to-optimize method, summarized as Algorithm 2, uses data to tune $\boldsymbol{\mu}$ based on (14). We initialize $\boldsymbol{\mu}$ before the training process with a fixed step size with which PGD converges (given sufficient iterations). Then, we exploit the differentiability of the gradients in (12) and the projection operator in (11) to tune $\boldsymbol{\mu}$ via conventional deep learning tools (where in Algorithm 2 we employ mini-batch stochastic gradient descent for learning). At the end of the training process, the learned vector $\boldsymbol{\mu}$ is used as a hyperparameter for a rapid finding of the best power allocation for a given channel via K iterations of Algorithm 1.

Algorithm 2: Training PGDNet

Init: Set $\boldsymbol{\mu}$ as fixed step sizes;
Set learning rate η, batches Q, and epochs.
Input: Training set $\mathcal{D}$
, Initial guess $\boldsymbol{P}^{(0)}$
for $i = 1, 2, \ldots,$ epochs **do**
  Randomly divide $\mathcal{D}$ into Q batches $\{\mathcal{D}_q\}_{q=1}^{Q}$
  for $q = 1, \ldots, Q$ **do**
    Compute average loss $\mathcal{L}_{\mathcal{D}_q}(\boldsymbol{\mu})$ using (14);
    Update $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} - \eta \nabla_{\boldsymbol{\mu}} \mathcal{L}(\boldsymbol{\mu})$
return $\boldsymbol{\mu}$

3) PGDNet Ensemble: PGDNet uses data to make PGD operate most efficiently within K iterations (where K is small). However, by C1 it is still expected to converge to a

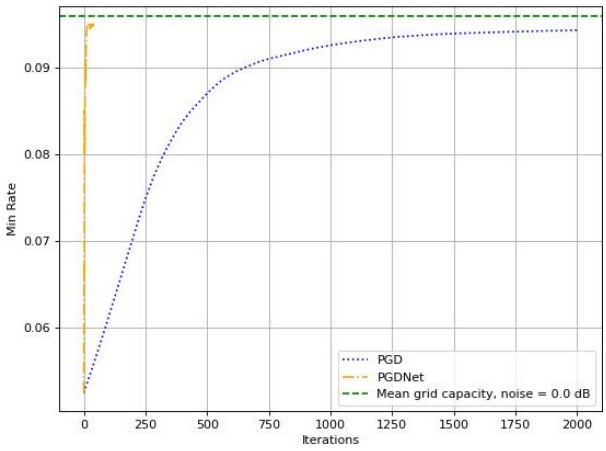

Fig. 4. Min rate per PGD iteration averaged over 200 channels

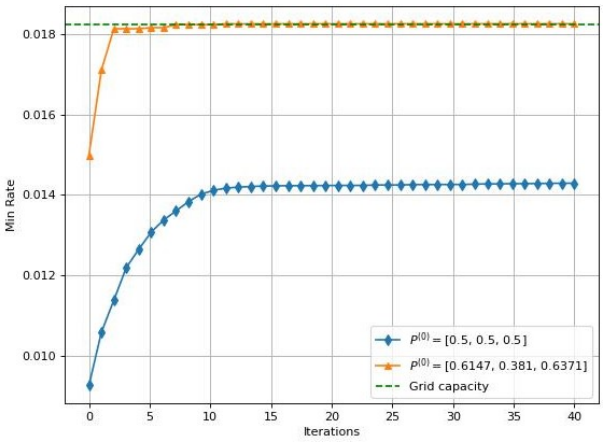

Fig. 5. Min rate per PGD iteration for a single channel realization

power allocation in the proximity of the initial guess $\boldsymbol{P}^{(0)}$, which is not necessarily the most suitable one. Nonetheless, by setting K to be relatively small, one can apply PGDNet $E > 1$ times, each time with a different $\boldsymbol{P}^{(0)}$ (thus accounting for C1), while still inferring sufficiently fast to cope with C2.

The resulting power allocation procedure thus utilizes an ensemble model [20] comprised of E PGDNet algorithms. Specifically, we train a single PGDNet model for a given initial guess denoted $\boldsymbol{P}_1^{(0)}$, and learn the hyperparameters $\boldsymbol{\mu}$. Then, in runtime, given a channel realization $\boldsymbol{h}, \boldsymbol{\sigma}$, we apply PGDNet with these hyperparameters E, using different initial guesses denoted $\boldsymbol{P}_1^{(0)}, \ldots, \boldsymbol{P}_E^{(0)}$, as illustrated in Fig. 3. To select the superposition code, we leverage the fact that we can evaluate each suggested setting using the optimization objective in (8), and select the power allocation which gives the maximal min-rate among all the iterations of the ensemble models, i.e.,

$$\max_{k\in 1,\ldots K} \max_{e\in 1,\ldots E} \min_{n\in\mathcal{N}} R_n \left(\mathbf{PGD}_K^{(k)}(\boldsymbol{h}, \boldsymbol{\sigma}; \boldsymbol{P}_e^{(0)}, \boldsymbol{\mu}) \right). \quad (15)$$

C. Discussion

Our proposed unfolded PGDNet is designed to design superposition codes for a given NOMA-MANET while coping with the non-convexity of the problem (C1) and the need to design rapidly (C2). The former is tackled by employing an ensemble of convex optimizers and selecting the best candidate setting computed, while rapid operation of each optimizer is achieved by using learning techniques to optimize the optimzer via deep unfolding. While classic approaches to improve convergence in terms of a number of iterations, e.g., line search [16, Ch. 9], come at the cost of increased latency due to additional processing carried out at each iteration, our unfolded method does not induce any excessive latency during inference compared with conventional PGD with fixed pre-determined step sizes. This makes our approach highly suitable for facilitating the rapid tuning of superposition codes.

The formulation of Unfolded PGDNet is invariant of the initial guesses $\boldsymbol{P}_1^{(0)}, \ldots, \boldsymbol{P}_E^{(0)}$. In our numerical study reported in Section IV we used equal power allocation for $\boldsymbol{P}_1^{(0)}$ and randomized the remaining guesses uniformly over $\mathcal{P}$. This setting was empirically shown to be beneficial in terms of rapidly finding high min-rate superposition codes. Nonetheless, one can explore different settings of the initial guesses, which we leave for future investigation. Moreover, our derivation is tailored for a two-hop scalar MANETs; its extension to arbitrary mesh MANETs with multivariate signaling, e.g., multi-antenna systems, is left for future study.

IV. Numerical Evaluations

In this section, we numerically evaluate Unfolded PGDNET[1]. We simulate a two-hop MANET with $N = 2$ receivers and $M = 2$ relays, as in Fig. 1. The channels obey Rayleigh fading, i.e., the channel gains are randomized from a unit variance Gaussian distribution, with $|\mathcal{D}| = 1000$ realizations used for learning, and 200 realizations for the test.

We employ PGDNet with $K = 40$ iterations, trained over 100 epochs with batch size $|\mathcal{D}_q| = 100$ using Adam. We use uniform allocation for the initial guess used in training, i.e., $[\boldsymbol{P}_1^{(0)}]_{m,n} = \frac{1}{\sqrt{N}}$. We compare PGDNet with the conventional PGD with fixed step sizes (manually tuned to achieve consistent convergence). Since the considered setup has only $(N-1)(1+M) = 3$ optimization variables, we also compute the maximal min-rate by an exhaustive search over all allocations in the unit cube with a grid of resolution of 10^{-2}. The outcome of this exhaustive search, which is not feasible for larger M, N, is termed *Grid capacity* and serves as an upper bound on the min-rate of the considered optimizers.

Fig, 4 reports the min-rate of Unfolded PGDNet with a single $E = 1$ model compared with classic PGD versus the iteration number, averaged over all 200 channels, for the noise level of $\sigma_{\rm rx}^2 = \sigma_{\rm re}^2 = 0$ dB. We clearly observe in Fig. 4 that our learn-to-optimize method allows PGDNet to systematically approach the grid capacity with approximately $82\times$ fewer iterations compared to conventional PGD (which requires on average about 3300 iterations to converge). It is noted though that PGDNet with $E = 1$ is still within some margin from the grid capacity itself due to the non-convex nature of the problem. This arises as some of the channels, starting from $\boldsymbol{P}_1^{(0)}$ do not allow to reach the grid capacity. However, when

[1]The source code used in our empirical study along with the hyperparameters is available at https://github.com/AlterTomer/Deep-Unfolded-PGD

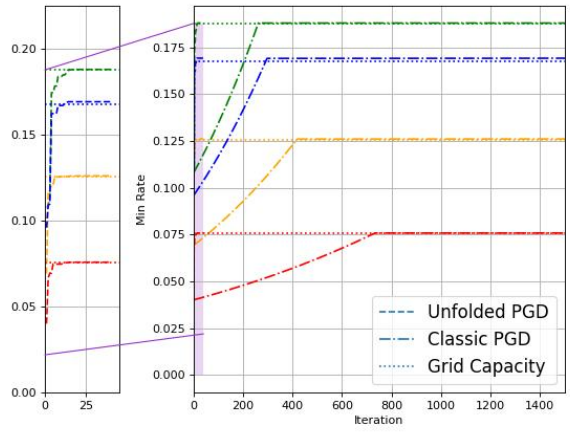

Fig. 6. Min-rate vs. iteration for four different channels

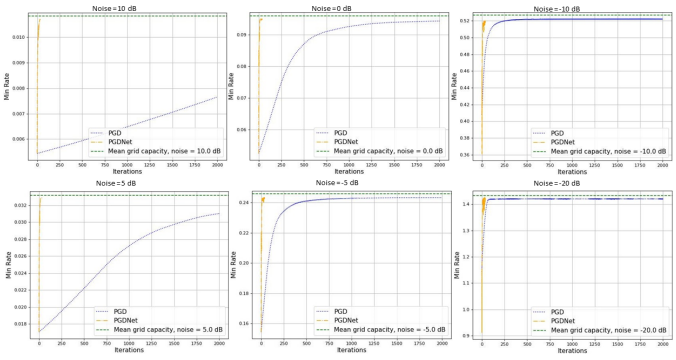

Fig. 7. Min-rate vs. iteration averaged over 200 channels in different SNRs

using an alternative initialization, a PGDNet trained with $\boldsymbol{P}_1^{(0)}$ can in fact rapidly achieve the grid capacity when starting from a different initial guess, as demonstrated in Fig. 5, confirming the need for our ensemble-based design.

Figs. 4 and 5 indicate the validity of the main ingredients of Unfolded PGDNet, i.e., unfolding for rapid operation and ensemble for coping with non-convexity. We thus proceed by evaluating the overall Unfolded PGDNet using an ensemble of $E = 6$ models, where the initial points not used in training, i.e., $\boldsymbol{P}_2^{(0)}, \ldots, \boldsymbol{P}_E^{(0)}$, are randomized uniformly over $\mathcal{P}$. The achieved sum-rate versus iterations for four random channel realizations are reported in Fig. 6, while Fig. 7 depicts the min-rate averaged over 200 channels for different noise levels (though PGDNet is only trained for 0 dB noises). In Fig. 6 we can clearly observe that for the same channels, the unfolded PGD achieves grid capacity much faster than the classic PGD. The reduction in latency achieved by Unfolded PGDNet here is of a factor of approximately $13.8\times$ ($K \times E = 240$ iterations vs. 3317 required on average for PGD). We also observe in Fig. 7 that the gains of Unfolded PGDNet become more dominant in noisier settings, demonstrating the ability of the proposed methodology to notably facilitate optimizing superposition codes for NOMA MANETs in challenging channel conditions.

V. Conclusion

In this paper we proposed Unfolded PGDNet for tuning superposition coding in two-hop NOMA MANETs. Unfolded PGDNet leverages data operate reliably within a fixed and small number of iterations while being trainable in an unsupervised manner. We cope with the non-convexity of the problem by ensembling multiple models. Our numerical results show that Unfolded PGDNet rapidly sets superposition codes that approach the maximal achievable min-rate.

References

[1] S. H. Cha, M. Shin, J.-H. Ham, and M. Y. Chung, "Robust mobility management scheme in tactical communication networks," *IEEE Access*, vol. 6, pp. 15 468–15 479, 2018.

[2] J. L. Burbank, P. F. Chimento, B. K. Haberman, and W. T. Kasch, "Key challenges of military tactical networking and the elusive promise of MANET technology," *IEEE Commun. Mag.*, vol. 44, no. 11, pp. 39–45, 2006.

[3] Y. Liu, Z. Qin, M. Elkashlan, Z. Ding, A. Nallanathan, and L. Hanzo, "Nonorthogonal multiple access for 5G and beyond," *Proc. IEEE*, vol. 105, no. 12, pp. 2347–2381, 2017.

[4] Z. Ding, X. Lei, G. K. Karagiannidis, R. Schober, J. Yuan, and V. K. Bhargava, "A survey on non-orthogonal multiple access for 5G networks: Research challenges and future trends," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 10, pp. 2181–2195, 2017.

[5] J. G. Andrews, "Interference cancellation for cellular systems: a contemporary overview," *IEEE Wireless Commun.*, vol. 12, no. 2, pp. 19–29, 2005.

[6] Y. Gao, B. Xia, K. Xiao, Z. Chen, X. Li, and S. Zhang, "Theoretical analysis of the dynamic decode ordering SIC receiver for uplink NOMA systems," *IEEE Commun. Lett.*, vol. 21, no. 10, pp. 2246–2249, 2017.

[7] N. Shlezinger, R. Fu, and Y. C. Eldar, "DeepSIC: Deep soft interference cancellation for multiuser MIMO detection," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1349–1362, 2021.

[8] M. Jain, N. Sharma, A. Gupta, D. Rawal, and P. Garg, "Performance analysis of NOMA assisted mobile ad hoc networks for sustainable future radio access," *IEEE Sustain. Comput.*, vol. 6, no. 2, pp. 347–357, 2020.

[9] A. Amer, S. Hoteit, and J. Ben-Othman, "Resource allocation for enabled-network-slicing in cooperative NOMA-based systems with underlay D2D communications," in *IEEE International Conference on Communications (ICC)*, 2023.

[10] A. Chowdhury, G. Verma, C. Rao, A. Swami, and S. Segarra, "ML-aided power allocation for tactical MIMO," in *IEEE Military Communications Conference (MILCOM)*, 2021, pp. 273–278.

[11] T. Chen, X. Chen, W. Chen, Z. Wang, H. Heaton, J. Liu, and W. Yin, "Learning to optimize: A primer and a benchmark," *The Journal of Machine Learning Research*, vol. 23, no. 1, pp. 8562–8620, 2022.

[12] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *Proc. IEEE*, vol. 111, no. 5, pp. 465–499, 2023.

[13] N. Shlezinger, Y. C. Eldar, and S. P. Boyd, "Model-based deep learning: On the intersection of deep learning and optimization," *IEEE Access*, vol. 10, pp. 115 384–115 398, 2022.

[14] A. El Gamal and Y.-H. Kim, *Network information theory*. Cambridge university press, 2011.

[15] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[16] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[17] W. Wang and M. A. Carreira-Perpinán, "Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application," *arXiv preprint arXiv:1309.1541*, 2013.

[18] N. Shlezinger and T. Routtenberg, "Discriminative and generative learning for linear estimation of random signals [lecture notes]," *IEEE Signal Process. Mag.*, 2023, early access.

[19] O. Lavi and N. Shlezinger, "Learn to rapidly and robustly optimize hybrid precoding," *arXiv preprint arXiv:2301.00369*, 2023.

[20] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.