A Deep Learning-Based Resource Allocator for Communication Networks With Dynamic User Utility Demands

Pourya Behmandpoor, *Graduate Student Member, IEEE*, Mark Eisen, *Member, IEEE*, Panagiotis Patrinos, *Member, IEEE*, and Marc Moonen, *Fellow, IEEE*

***Abstract*—Deep learning (DL)-based resource allocation (RA) has recently gained significant attention due to its performance efficiency. However, most related studies assume an ideal case where the number of users and their utility demands, e.g., data rate constraints, are fixed, and the designed DL-based RA scheme exploits a policy trained only for these fixed parameters. Consequently, computationally complex policy retraining is required whenever these parameters change. In this paper, we introduce a DL-based resource allocator (ALCOR) that allows users to adjust their utility demands freely, such as based on their application layer requirements. ALCOR employs deep neural networks (DNNs) as the policy in a time-sharing problem. The underlying optimization algorithm iteratively optimizes the on-off status of users to satisfy their utility demands in expectation. The policy performs unconstrained RA (URA)—RA without considering user utility demands—among active users to maximize the sum utility (SU) at each time instant. Depending on the chosen URA scheme, ALCOR can perform RA in either a centralized or distributed scenario. The derived convergence analyses provide theoretical guarantees for ALCOR's convergence, and numerical experiments corroborate its effectiveness compared to meta-learning and reinforcement learning approaches.**

***Index Terms*—Deep learning based resource allocation, dynamic data rate constraints, dynamic quality-of-service, utility demands, centralized and distributed resource allocation.**

Received 21 August 2024; revised 19 December 2024 and 16 May 2025; accepted 8 July 2025. Date of publication 6 August 2025; date of current version 22 December 2025. This work was supported in part by the ESAT Laboratory of KU Leuven in the frame of Research Project Fonds voor Wetenschappelijk Onderzoek-Vlaanderen (FWO) "User-Centric Distributed Signal Processing Algorithms for Next Generation Cell-Free Massive Multi-Input Multi-Output (MIMO)-Based Wireless Communication Networks" under Grant G0C0623N; in part by the Fonds de la Recherche Scientifique-FNRS and Fonds voor Wetenschappelijk Onderzoek-Vlaanderen Excellence of Science (EOS) "MUlti-SErvice WIreless NETworks (MUSE-WINET)" under Project 30452698; in part by FWO under Grant G081222N, Grant G033822N, and Grant G0A0920N; and in part by the Research Council KU Leuven C1 Project under Grant C14/24/103. The associate editor coordinating the review of this article and approving it for publication was Y. Zhou. *(Corresponding author: Pourya Behmandpoor.)*

Pourya Behmandpoor, Panagiotis Patrinos, and Marc Moonen are with the Department of Electrical Engineering (ESAT), STADIUS Center for Dynamical Systems, Signal Processing and Data Analytics, KU Leuven, 3000 Leuven, Belgium (e-mail: pourya.behmandpoor@esat.kuleuven.be; panos.patrinos@esat.kuleuven.be; marc.moonen@esat.kuleuven.be).

Mark Eisen is with the Applied Physics Laboratory, Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: mark.eisen@ieee.org).

This article has supplementary downloadable material available at https://doi.org/10.1109/TWC.2025.3592774, provided by the authors.

Digital Object Identifier 10.1109/TWC.2025.3592774

I. Introduction

RESOURCE allocation (RA) in communication systems has been an active research topic for decades [1], [2], [3], [4], resulting in the development of various RA schemes. Since the underlying optimization problem is nonconvex and possibly large-scale, conventional RA schemes typically exhibit slow convergence in large communication systems. Moreover, these schemes cannot be easily extended either to consider longer time horizons, e.g., to perform RA for an average of utilities over time, or to address the vast interconnection of users taking their various mutual effects into account. These limitations motivate recent advancements in deep learning (DL)-based RA, where a policy, such as a deep neural network (DNN), is typically dedicated to translating dynamic parameters of the communication system, e.g., channel coefficients, into optimal resources, e.g., the optimal transmit power for each user [5], [6].

Existing DL-based RA schemes in the literature incorporate various training strategies. The policies are either trained in a supervised manner [7], where RA solutions are available as labels, or in an unsupervised manner [8], [9], [10], [11], where a utility function, e.g., sum data rate of users, is considered the global reward function. Alternatively, policies can serve as decision-makers that monitor the state space, e.g., the set of channel coefficients, and choose actions within the action space, e.g., different transmit power levels. Such policies can be trained using reinforcement learning (RL) principles [12], [13], [14], [15]. The mentioned training strategies are mostly model-based; they assume a model for the reward, and the trainer optimizes the policy parameters using the calculated gradient of this model. An alternative training approach is model-free, where the gradient is approximated by various methods, such as policy gradient (REINFORCE [16]) or zeroth-order optimization, by measuring reward values [17], [18], [19]. This training can also be performed in real-time while the communication system is operating. Through real-time reward measurement, this approach can capture the full behavior of the system, including nonidealities such as nonlinearities in the modulator and demodulator or antenna setup, which are not typically captured by model-based approaches due to simplifications.

1536-1276 © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

In the inference step, existing DL-based RA schemes are employed in either a *centralized* or a *distributed* manner. In a centralized approach, a server is responsible for conducting the RA by gathering necessary information from all users and employing a centralized policy for RA [12], [13], [15], [19]. Distributed RA can serve as an alternative to centralized RA, enabling users to locally decide how to allocate shared communication resources without transmitting relevant information, such as channel state information, to a central server. Distributed RA is particularly beneficial in scenarios where a server with sufficient communication and computational capacity is unavailable within the network and where avoiding a single point of failure is critical. In this approach, message-passing occurs locally between neighboring users who interfere with each other the most, ensuring more efficient use of communication bandwidth [20], scalability (e.g., by using graph neural networks (GNNs) [19], [21], [22], [23], [24], [25], [26]), and a reduction in deployment delays and RA policy complexity [12], [13], [15], [27], [28]. These benefits come at the cost of limited local memory and computational resources, as well as more complex procedures to maintain the necessary synchrony between users during both training and inference steps. In fully connected networks, where each user can directly exchange messages with all other users, ensuring synchrony and identifying neighboring groups is more straightforward compared to networks with limited connectivity between users. In the distributed setting, the policies can be either identical among the users [13], [15] or different [12], [29]. Furthermore, the training step can also be classified as either centralized or distributed [12], [13], [15], [19], [26], [28], [30], [31].

DL-based RA can be categorized into *unconstrained* RA and *constrained* RA. In unconstrained RA, a global reward function is optimized with minimal constraints, such as simple box constraints that can be met by applying a projection through an appropriate output activation function. However, this approach cannot address more complex constraints, such as user utility (quality-of-service (QoS)) demands—e.g., user data rate constraints. This limitation means the RA is unable to allocate additional resources to users with higher demands. A simple workaround involves optimizing the weighted sum of utilities as the global reward function [11], [32], where the weights reflect users' utility demands. Conversely, in constrained RA, a global reward function is optimized while satisfying more complex constraints. During policy training, these constraints are typically managed either by including a penalization term in the reward function [8], [28], [33], [34], or by using the primal-dual optimization method, which directly addresses constrained optimization problems [18], [19], [23], [24], [26].

Constrained DL-based RA schemes in the literature utilize policies, such as DNNs, which are trained for a fixed set of RA constraints (e.g., utility demands). For example, in works such as [13], [28], [33], [34], RA, service migration, and radio access network slicing are addressed using RL for vehicular communication, where delay-sensitive and delay-tolerant, as RA constraints, are considered through a penalization term in the reward function. Although these works consider heterogeneous QoS and other RA constraints in their optimization problem to guide the policy toward meeting these constraints in dynamic RA, their trained policies cannot generalize to a new set of constraints—for example, when a user increases their data rate demand. In such RA schemes, policy retraining becomes necessary if the RA constraints change. This limitation undermines the applicability of existing DL-based RA schemes, as policy retraining is computationally complex, leading to delays in real-time deployment. One possible workaround is to incorporate dynamic constraints, such as utility demands, along with all other dynamic components of the communication system, into the policy input. In this case, the policy would be trained for various utility demands. However, generalizing the problem with a moderate amount of resources, such as the number of layers and neurons in DNN policies, is not practically feasible. This has already been experimentally validated in [11] and our simulations. This limitation has recently sparked research into developing resilient policies with the capability to generalize to new network conditions, such as new RA constraints. Works such as [35] in bandwidth and transmit power allocation, and [36] in beamforming, utilize transfer learning concepts [37] to fine-tune only part of the policy (e.g., the last few layers) to reduce the computational complexity of retraining. Furthermore, many studies such as [38], [39], [40], [41] in DL-based resource scheduling and task offloading, and [20], [26], [42] in DL-based RA, leverage meta-learning concepts [43], [44], [45], [46] to initialize a policy that can quickly adapt to varying network conditions with only a few training samples. In [20], a distributed RA method is proposed to maximize user energy efficiency while meeting their minimum SNR requirements to guarantee utility (QoS) demands. The RA policy is trained based on RL in a federated learning framework to allocate optimal subchannels and transmit power. Constraints are incorporated into the reward function by penalizing decisions that violate them, similar to the approach in [13] and [42] for V2V communication. Additionally, distributed transmit power allocation is investigated in [26], where GNNs are used to locally determine the optimal transmit power to increase the sum rate (SR). In line with [23], data rate constraints are satisfied (on average over time, as in our setting) by penalizing violations while following primal-dual updates. In all the aforementioned methods [20], [26], [38], [39], [40], [41], [42], constraint violations are penalized during training. Additionally, these methods leverage model-agnostic meta-learning (MAML) principles [43], where they first train an initial policy on a subset of constraints, and then adapt to new constraints (e.g., utility demands) by fine-tuning the entire policy with a few additional training samples.

Although meta-learning has shown promising results in rapidly adapting policies to new network conditions, it still involves policy fine-tuning, which may be computationally intensive for complex policies, such as large DNNs. Moreover, the initial policy training in meta-learning requires a more computationally demanding training procedure. The generalizability of the policy after fine-tuning also depends on the diversity and representativeness of the training samples, as well as the similarity of nework conditions [27], [43], [47],

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

which cannot always be easily guaranteed in RA. Among various network conditions, in this paper, we focus on user utility demands, allowing users to have dynamic utility demands (such as dynamic user data rate constraints) based on their real-time tasks. Existing DL-based RA methods, as well as DL-based resource scheduling and task offloading methods, require policy re-training or fine-tuning to address a new set of user utility demands. To the best of our knowledge, the proposed work is the first DL-based RA method that adapts to dynamic user utility demands without any policy retraining or fine-tuning.

Objective: In this paper,[1] the objective is to propose a DL-based RA scheme that can adapt, on average over time, to dynamic user utility demands. The novelty of this work lies in the fact that, unlike existing DL-based RA schemes, the adaptation to new sets of user utility demands does not require any retraining or fine-tuning of the involved policies, thereby conserving computational resources and improving speed.

Contributions: The contributions are listed as follows:

(i) A RA scheme is proposed, which involves time-sharing and DL-based RA among the users. At each time instant, the proposed RA scheme selects a subset of users to be activated and performs DL-based RA among these activated users to maximize their sum utility (SU). The proposed time-sharing algorithm is an optimization algorithm that iteratively controls the on-off status of users to guarantee their utility demands in expectation.

(ii) Convergence analyses are provided for the proposed RA scheme under standard assumptions, deriving a convergence rate of $\mathcal{O}(1/\sqrt{k})$, with k representing the iteration counter in the considered time-sharing algorithm.

(iii) Rigorous numerical experiments are conducted to assess the performance of the proposed RA scheme and its distributed variant against various benchmarks.

II. System Model

We consider N users (links) each equipped with a transmitter and a receiver. The direct channel between the transmitter and receiver of user i is denoted by h_{ii}, while the interference channel between the transmitter of user j and the receiver of user i is denoted by h_{ij}. All the channel coefficients define the full channel matrix $\boldsymbol{H} \in \mathbb{C}^{N\times N}$ with h_{ij} as its element in the ith row and jth column.

For the RA, we consider a central deep neural network (DNN) as the policy defined by

$$\phi(\boldsymbol{S}, \boldsymbol{\theta}) : \mathbb{R}^s \times \boldsymbol{\Theta} \to \mathbb{R}^r \tag{1}$$

with input $\boldsymbol{S}$ and policy parameter $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^n$. Here $\boldsymbol{S}$ is a random variable incorporating global or local measurements of users, which vary over time and is a function of parameters such as the global channel $\boldsymbol{H}$ and the users' state. Examples of elements in $\boldsymbol{S}$ could include channel coefficients, packet queue length awaiting transmission, or received interference power from neighbors (refer to Section IV for more details). The policy parameter $\boldsymbol{\theta}$ may contain DNN weights and biases, to be optimized during training. The policy output specifies resources (ideally to be optimal) such as transmit power, frequency band, timeslot, beamformer angle, etc. of all N users, with an r-dimensional output. Furthermore, each user i has a utility function $U_i(\boldsymbol{H}, \phi(\boldsymbol{S}, \boldsymbol{\theta}))$, which depends on the communication channel $\boldsymbol{H}$ and the resources allocated to all users by $\phi(\boldsymbol{S}, \boldsymbol{\theta})$. For instance, if the utility represents the user data rate, $U_i = R_i(\boldsymbol{H}, \boldsymbol{p})$, it depends on the channel matrix $\boldsymbol{H}$ and the transmit power of all users $\boldsymbol{p} = (p_1, \ldots, p_N)$ (due to interference between users), as set by the centralized policy output, i.e., $\boldsymbol{p} = \phi(\boldsymbol{S}, \boldsymbol{\theta})$. Refer to Section IV and (9) for further details. It is noted that the policy described in (1) is designed for centralized RA, where a single server performs RA. Distributed extensions will be discussed in Section III-E.

[1]This paper is an extension of the work presented in [48].

III. Proposed Method

A. Problem Statement

Before formally stating the problem, consider the user selection vector $\boldsymbol{\xi}$ defined as follows:

Definition 1 (User Selection Vector (USV) $\boldsymbol{\xi}$): Define the USV as $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_N) \in \Omega^{\xi} := \{0, 1\}^N$. The elements ξ_i are independently drawn from a Bernoulli distribution [49] with mean:

$$\kappa_i := \mathbb{E}\{\xi_i\} \in [0, 1], \quad \forall i \in [N]. \tag{2}$$

By the USV, user i is selected to be active if $\xi_i = 1$, otherwise it is switched off. Moreover, the USV Bernoulli distribution is denoted by $\mathcal{D}^{\xi}(\boldsymbol{\kappa})$ with $\boldsymbol{\kappa} := (\kappa_1, \cdots, \kappa_N) \in [0, 1]^N$.

Note that according to (2), the probability that user i is activated is κ_i. Moreover, the Bernoulli distribution, characterized by its binary variables, provides a straightforward and interpretable definition. Its first moment directly corresponds to the user activation probability and can be easily implemented by the server (in the centralized variant) or by individual users (in the distributed variant) by flipping a biased coin.

After defining the USV $\boldsymbol{\xi}$, we are interested in the following problems:

Time-sharing problem:

Find the optimal probabilities $\boldsymbol{\kappa} = (\kappa_1, \cdots, \kappa_N) \in [0, 1]^N$ for activating users at each time instant, based on $\boldsymbol{\xi} \sim \mathcal{D}^{\xi}(\boldsymbol{\kappa})$, such that:

- The utility demands of all users are satisfied in expectation (on average over time);

Resource allocation problem:

At each time instant, allocate resources among the users such that:

- The SU of active users is maximized for a given $\boldsymbol{\xi} \in \Omega^{\xi}$ and $\boldsymbol{H}$.

After optimizing the probabilities $\boldsymbol{\kappa}$, at each time instant, a random instance of the USV $\boldsymbol{\xi}$ defined in Definition 1 is drawn. Then, according to $\boldsymbol{\xi}$, some users are activated while others are deactivated. Subsequently, a URA is performed among the active users.

Before addressing the mentioned problems, the basic assumptions made throughout the paper are as follows:

Assumption 1 (Basic Assumptions):

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

(i) The dynamic parameters of the communication network, e.g., the channel $\boldsymbol{H}$, are ergodic stochastic variables [49], with possibly unknown distributions;
(ii) The utility demands $\boldsymbol{u}^{\min}$ are changing slower than the dynamic parameters of the communication network, and are assumed to be constant in time windows.

It is remarked that in Assumption 1, utility demands $\boldsymbol{u}^{\min}$ are not assumed to be fixed, which is a restrictive assumption in the literature that we aim to relax. Moreover, Assumption 1(i) regarding the dynamic parameters of the communication network is a standard assumption in the literature [13], [15], [24], [25]. Assumption 1(ii) is a mild assumption in practice since utility demands $\boldsymbol{u}^{\min}$ mostly reflect the demands originating from the application layer, e.g., following a task scheduling scheme based on currently running tasks. These demands are slower in nature, typically changing at a rate of seconds or more, compared to the channel dynamics in wireless communication networks, which typically change at a rate of milliseconds in fast-fading scenarios (refer to Section III-D for further explanation).

B. Unconstrained Resource Allocation (URA)

To address the DL-based URA for each $\boldsymbol{\xi}$ and $\boldsymbol{H}$, the RA policy ϕ defined in (1) is trained using the following unconstrained optimization:

$$\begin{aligned}&\underset{\boldsymbol{\theta}\in\mathbb{R}^n}{\text{maximize}} \sum_{i\in[N]} \mathbb{E}_{\boldsymbol{H}\sim\mathcal{D}^H,\boldsymbol{\kappa}\sim\mathcal{D}^\kappa,\xi\sim\mathcal{D}^\xi(\boldsymbol{\kappa})}\left[U_i^\theta(\boldsymbol{H}_\xi)\right],\\ &\text{where } U_i^\theta(\boldsymbol{H}_\xi) := U_i(\boldsymbol{H}_\xi,\phi(\boldsymbol{S}_\xi,\boldsymbol{\theta})),\end{aligned} \tag{3}$$

with $U_i(\boldsymbol{H}_\xi,\phi(\boldsymbol{S}_\xi,\boldsymbol{\theta}))$ as the utility of user i, explained in Section II. Here, $\boldsymbol{H}_\xi$ denotes the global channel for the subset of active users, which equals the channel matrix $\boldsymbol{H}$ with distribution $\mathcal{D}^H$ except for rows and columns corresponding to the zero elements of $\boldsymbol{\xi}$ (deactivated users), which are set to zero. Similarly, $\boldsymbol{S}_\xi$ contains measurements of the active users, and $\boldsymbol{\kappa} := (\kappa_1,\cdots,\kappa_N)$ with a uniform distribution $\mathcal{D}^\kappa$ in $[0,1]^N$. For simplicity, the policy input $\boldsymbol{S}_\xi$ is assumed to be a function of only the channel $\boldsymbol{H}$, so is U_i^θ.

In problem (3), the policy parameter is optimized to maximize the SU of users in expectation, serving as a global utility function. This problem can be addressed using first-order stochastic gradient descent (SGD) over batches of random samples. This leads to the following iterations with an initial value $\boldsymbol{\theta}^0$ and an iteration counter $\ell\in[L-1]$ with $L>0$:

$$\boldsymbol{\theta}^{\ell+1} = \boldsymbol{\theta}^\ell + \frac{\gamma^\ell}{B}\sum_{i\in[N]}\sum_{j=1}^{B}\nabla_{\boldsymbol{\theta}}\boldsymbol{U}_i^{\boldsymbol{\theta}^\ell}(\boldsymbol{H}_\xi^{\ell,j}). \tag{4}$$

In (4), sample averaging is performed over batches of size B with channels $\boldsymbol{H}_\xi^{\ell,j}$ and probabilities $\boldsymbol{\kappa}^{\ell,j}$ where $\boldsymbol{\xi}^{\ell,j}\sim\mathcal{D}^\xi(\boldsymbol{\kappa}^{\ell,j})$. Here, the superscript $\{\ell,j\}$ indicates the jth sample in the batch taken at iteration ℓ. The step size is also denoted by γ^ℓ. The capability of policy ϕ to generalize the URA problem for all possible $\boldsymbol{H},\boldsymbol{\xi}$, and $\boldsymbol{\kappa}$ heavily depends on the policy resources, such as the number of layers and perceptrons in each layer in the case of a DNN policy. Due to the limited memory and computational capacities, as well as the nonconvex nature of the problem, generalization is often suboptimal in practice.

The policy ϕ and the training procedure used to address the maximization problem in (3) depend on the URA method employed. For instance, existing URA methods in the literature, including those utilizing various training procedures other than (4), such as RL, can be incorporated. The proposed RA scheme is agnostic to the choice of the URA method (refer to Remark 1).

During the inference stage, the *trained* policy is utilized to perform URA among active users at each time instant as follows:

$$\begin{aligned}&r_i := [\phi(\boldsymbol{S}_\xi,\boldsymbol{\theta}^L)]_i \quad \text{if} \quad \xi_i = 1,\\ &r_i := 0 \quad \text{if} \quad \xi_i = 0,\end{aligned} \tag{5}$$

where $[\cdot]_i$ indicates the ith element of the vector and r_i defines the allocated resource for user i. The frequently used notations are summarized in Table I.

It is remarked that the term *unconstrained* in URA emphasizes that dynamic user utility demands—such as varying data rate or latency requirements—are not addressed in this stage of resource allocation. However, minimal constraints, such as simple box constraints, can still be handled within the URA. For example, a maximum transmit power limit can be enforced by applying a sigmoid activation function in the final layer of ϕ.

C. Time-Sharing

After training the policy $\phi(\cdot,\boldsymbol{\theta}^L)$ for URA, we proceed to formulate the time-sharing problem as defined in Section III-A. To do so, we first define the following function $F:\mathbb{R}^N\to\mathbb{R}^N$

$$\begin{aligned}F(\boldsymbol{\lambda}) &:= \mathbb{E}_{\boldsymbol{H},\boldsymbol{\xi}\sim\mathcal{D}^\xi(\boldsymbol{\kappa})}\left[\hat{F}(\boldsymbol{H}_\xi)\right],\\ \text{where}\quad \hat{F}(\boldsymbol{H}_\xi) &:= \boldsymbol{u}^{\min} - \boldsymbol{U}^{\boldsymbol{\theta}^L}(\boldsymbol{H}_\xi),\\ \kappa_i &= {}^{(1+\lambda_i)}\!/_{\max_\ell\{1+\lambda_\ell\}},\end{aligned} \tag{6}$$

with $\boldsymbol{\lambda}\in\mathbb{R}_+^N$, $\boldsymbol{\kappa}$ defined in Definition 1, and $\boldsymbol{U}^{\boldsymbol{\theta}^L} := (U_1^{\boldsymbol{\theta}^L},\cdots,U_N^{\boldsymbol{\theta}^L})$.

In (6), we aim to solve the following problem:

$$\text{find } \boldsymbol{\lambda}\in\mathbb{R}_+^N \text{ such that } F(\boldsymbol{\lambda})\le\boldsymbol{0}, \tag{7}$$

where $\boldsymbol{0}\in\mathbb{R}^N$ is a vector of all zeros and $\le$ is an element-wise operator. The equality $[F(\boldsymbol{\lambda})]_i = 0$ indicates that the expected utility of user i, $\mathbb{E}_{\boldsymbol{H},\boldsymbol{\xi}\sim\mathcal{D}^\xi(\boldsymbol{\kappa})}\left[U_i^{\boldsymbol{\theta}^L}(\boldsymbol{H}_\xi)\right]$, is equal to its utility demand $\boldsymbol{u}_i^{\min}$, while $[F(\boldsymbol{\lambda})]_i<0$ indicates that the expected utility of user i is greater than its demand. To guarantee inequality (7), we propose using $\boldsymbol{\lambda}$ to calculate the probabilities κ_i. This choice is motivated by the fact that κ_i remains nonzero for any choice of $\boldsymbol{\lambda}$ in (7), maintaining higher utility for the corresponding user, and yet, whenever required, κ_i can become arbitrarily small by choosing large λ_j by other users j. Thus, the coupling between users, which is due to interference in the communication system, can be captured by this definition of κ_i.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

TABLE I
LIST OF NOTATIONS

not.	Description	not.	Description
N	number of users	$\phi(\cdot,\boldsymbol{\theta})$	parameterized policy
$\boldsymbol{H}$	channel matrix	$U_i^{\theta}(\boldsymbol{H}_\xi)$	utility of user i
$\boldsymbol{X}_\xi$	$\boldsymbol{X}$ for activated users	r_i	resource allocated to user i
$\boldsymbol{\theta}$	policy parameter	$u_i^{\min}$	utility demand of user i
$\boldsymbol{\kappa}$	activation probabilities of users	$\hat{F}_i(\boldsymbol{H}_\xi)$	$u_i^{\min} - U_i^{\theta}(\boldsymbol{H}_\xi)$
$\boldsymbol{\xi}$	user selection vector ($\kappa_i = \mathbb{E}_\xi[\xi_i]$)	$F_i(\boldsymbol{\lambda})$	$\mathbb{E}_{H,\xi}\left[\hat{F}_i(\boldsymbol{H}_\xi)\right]$
$\boldsymbol{\lambda}$	decision vector, determining $\boldsymbol{\kappa}$	B	batch size

To estimate the expectation in (6), we define the following sample average:

$$\hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}^k) := \tfrac{1}{B}\textstyle\sum_{j=1}^{B} \hat{F}(\boldsymbol{H}_\xi^{k,j}), \tag{8}$$

where $\mathbb{H}_{\boldsymbol{\xi}}^k$ refers to the batch of samples, defined as $\mathbb{H}_{\boldsymbol{\xi}}^k := \{\boldsymbol{H}_\xi^{k,1}, \cdots, \boldsymbol{H}_\xi^{k,B}\}$ and $\boldsymbol{H}_\xi^{k,j}$ represents the jth sample in a batch of size B, taken at iteration k with USV $\boldsymbol{\xi}^k$. The algorithm for finding $\boldsymbol{\lambda}$ in (7) is outlined in the next subsection.

D. Algorithm

Algorithm 1 ALCOR

Input utility demands $\boldsymbol{u}^{\min}$, stepsizes $\gamma > 0$, $\alpha_k \in (0,1)$, batch size $B > 0$

Initialize $\bar{\boldsymbol{\lambda}}^{-1} = \bar{\boldsymbol{\lambda}}^0 \in \mathbb{R}^N$, $\boldsymbol{h}^{-1} \in \mathbb{R}^N$, $\bar{\boldsymbol{\kappa}}^0 \in \mathbb{R}^N$

Repeat $k = 0, 1, \ldots, K-1$

1: Sample a batch of $\bar{\boldsymbol{\xi}}^{k,j} \sim \mathcal{D}^{\xi}(\bar{\boldsymbol{\kappa}}^k)$ with $j \in [B]$

2: Perform URA (5) and form the average $\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k)$ in (8)

3: $\boldsymbol{h}^k := \bar{\boldsymbol{\lambda}}^k + \gamma\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k) + (1-\alpha_k)\left(\boldsymbol{h}^{k-1} - \bar{\boldsymbol{\lambda}}^{k-1} - \gamma\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k)\right)$

4: $\boldsymbol{\lambda}^k = \max\{\boldsymbol{0}, \boldsymbol{h}^k\}$

5: $\kappa_i^k = {(1+\lambda_i^k)}/{\max_\ell\{1+\lambda_\ell^k\}}, \quad \forall i \in [N]$

6: Sample a batch of $\boldsymbol{\xi}^{k,j} \sim \mathcal{D}^{\xi}(\boldsymbol{\kappa}^k)$ with $j \in [B]$

7: Perform URA (5) and form the average $\hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}^k)$ in (8)

8: $\bar{\boldsymbol{\lambda}}^{k+1} = \bar{\boldsymbol{\lambda}}^k - \alpha_k\left(\boldsymbol{h}^k - \boldsymbol{\lambda}^k - \gamma\hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}^k)\right)$

9: $\bar{\kappa}_i^{k+1} = {(1+\bar{\lambda}_i^{k+1})}/{\max_\ell\{1+\bar{\lambda}_\ell^{k+1}\}}, \quad \forall i \in [N]$

The proposed algorithm for the time-sharing problem (7) is summarized in Algorithm 1, which can be easily implemented using the mappings defined in (6) and (8) with a batch of channel samples $\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k := \{\bar{\boldsymbol{H}}_\xi^{k,1}, \cdots, \bar{\boldsymbol{H}}_\xi^{k,B}\}$. At each iteration k, in step 1, the algorithm activates a subset of users according to the USV instance $\bar{\boldsymbol{\xi}}^{k,j} \sim \mathcal{D}^{\xi}(\bar{\boldsymbol{\kappa}}^k)$, which depends on the most updated probabilities $\bar{\boldsymbol{\kappa}}^k := (\bar{\kappa}_1^k, \cdots, \bar{\kappa}_N^k)$. Then, in step 2, URA is performed among the activated users to maximize their SU. This procedure is repeated B times for each $j \in [B]$ to form a batch, allowing the calculation of the sample average (8) with the samples $\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k$. Following step 3, the current parameter $\boldsymbol{\lambda}^k$ is updated using either a fixed stepsize $\alpha_k = \alpha$ or a diminishing stepsize $\alpha_k = {\alpha_0}/{\sqrt{1+\tilde{\alpha}k}}$ with some $\alpha_0, \tilde{\alpha} > 0$, where Step 4 ensures the nonnegativity of updates, i.e., $\boldsymbol{\lambda}^k \in \mathbb{R}_+^N$. The same steps are repeated for a new batch of samples $\mathbb{H}_{\boldsymbol{\xi}}^k$ with $\boldsymbol{\xi}^{k,j} \sim \mathcal{D}^{\xi}(\boldsymbol{\kappa}^k)$, based on the probabilities $\boldsymbol{\kappa}^k$ set in step 5. Updates are performed following step 7 and 8. At the end, in step 9, the probabilities $\bar{\boldsymbol{\kappa}}^{k+1}$ are updated using the parameters $\bar{\boldsymbol{\lambda}}^{k+1}$. Note that in the implementation, $\bar{\kappa}_i^{k+1} = 0$ may be assigned whenever $\bar{\kappa}_i^{k+1} < 0$, which can occur due to negative values of $\bar{\lambda}_i^{k+1}$ during the initial iterations before the algorithm has converged. During the iterations, the utility demands are assumed to remain fixed within a time window of length T, which is sufficiently long to allow the algorithm to meet the utility demands on average over time, i.e., $K \ll T$. Moreover, once utility demands are updated, the algorithm restarts with a new input $\boldsymbol{u}^{\min}$.

It is noted that the two-step updates (step 4 and 8) can be seen as extragradient-like updates [50], [51], where during one iteration, the algorithm first generates an intermediate iterate $\boldsymbol{h}^k$ from a base iterate $\bar{\boldsymbol{\lambda}}^k$ and then completes the update by taking a step from the base iterate using the intermediate iterate. The time-sharing problem (7) can also be implemented by a simpler algorithm where a batch of samples $\boldsymbol{\xi}^{k,j} \sim \mathcal{D}^{\xi}(\boldsymbol{\kappa}^k)$ is employed, followed by URA among the activated users and calculating the average in (8). Then the parameter $\boldsymbol{\lambda}^k$ can be updated by

$$\boldsymbol{\lambda}^{k+1} = \max\left\{\boldsymbol{0}, \boldsymbol{\lambda}^k + \alpha_k\hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}^k)\right\}$$

at each iteration k. This update rule guarantees an increase in λ_i, and subsequently the probability κ_i, whenever the average utility of user i is below its demand $\boldsymbol{u}_i^{\min}$. However, there is no convergence guarantee for this simpler update rule in the optimization literature. Convergence analysis for Algorithm 1 is provided in the Appendix, where the time-sharing problem (7) is cast as an inclusion problem under the class of nonmonotone variational inequalities [52].

Remark 1 (Agnostic to URA Methods): The proposed RA scheme can employ policies as a closed box for URA. These policies can be centralized or distributed and can be trained and utilized by various DL methods, such as RL, GNN, etc. [8], [13], [22]. However, the scalability of the proposed method, the maximum number of users that ALCOR can accommodate for RA, and the communication overhead in the distributed variant depend on the employed URA method. Hence, suitable methods need to be selected based on the communication system of interest. Refer to Section IV-B for numerical studies on ALCOR's performance with various URA methods.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

ALCOR is limited to scenarios that can be formulated by (6) and (7). Specifically, ALCOR can address RA problems where increasing κ_i, the probability of user i being activated, leads to an increase in the user's utility and consequently satisfies its demand.

E. Distributed ALCOR

Distributed RA can serve as an alternative to centralized RA, enabling users to locally decide how to allocate shared communication resources, where a server with sufficient communication and computational capacity is unavailable within the network. Refer to Section I for further details on the benefits and limitations of distributed RA.

ALCOR can be readily extended to a distributed variant by employing distributed URA methods. In the distributed variant, each user has an individual policy $\phi_i(\cdot, \boldsymbol{\theta}_i)$ to perform distributed URA (see for example [12], [13], [15]). In this case, the policies are trained according to (3), with $U_i^{\theta}(\boldsymbol{H}_{\xi}) := U_i(\boldsymbol{H}_{\xi}, \boldsymbol{\phi}(\boldsymbol{S}_{\xi}, \boldsymbol{\theta}))$, where $\boldsymbol{\phi}(\boldsymbol{S}_{\xi}, \boldsymbol{\theta}) := (\phi_1(\boldsymbol{S}_{\xi,1}, \boldsymbol{\theta}_1), \cdots, \phi_N(\boldsymbol{S}_{\xi,N}, \boldsymbol{\theta}_N))$, $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N)$, and $\boldsymbol{S}_{\xi} = (\boldsymbol{S}_{\xi,1}, \ldots, \boldsymbol{S}_{\xi,N})$. Similar to the centralized case, due to the interference between users, the utility U_i^{θ} depends on the allocated resources of all users. However, in the distributed scenario, resources are defined by individual policies $\phi_i(\boldsymbol{S}_{\xi,i}, \boldsymbol{\theta}_i)$ rather than a centralized policy $\phi(\boldsymbol{S}_{\xi}, \boldsymbol{\theta})$. The individual policies consider individual local measurements $\boldsymbol{S}_{\xi,i}$ and their parameters may be different, i.e., $\boldsymbol{\theta}_i \neq \boldsymbol{\theta}_j$ for $i \neq j$ [12], or there may be a consensus among them, i.e., $\boldsymbol{\theta}_i = \boldsymbol{\theta}_j$ for $i \neq j$ [13], [15], depending on the employed distributed URA scheme. URA in (5) is performed with $r_i = \phi_i(\boldsymbol{S}_{\xi,i}, \boldsymbol{\theta}_i^L)$ if $\xi_i = 1$, and $r_i = 0$ otherwise. It is noteworthy that in distributed ALCOR, although the utility of user i, U_i^{θ}, depends on all individual policies, user i only measures U_i^{θ} without requiring other local parameters $\boldsymbol{\theta}_j, j \neq i$. Considering the definition of the mapping $\hat{F}$ in (6), each user i locally executes Algorithm 1 with local variables $\lambda_i^k, \bar{\lambda}_i^k, h_i^k$, and κ_i^k, utilizing local estimates $[\hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}^k)]_i$ and $[\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k)]_i$, where $[\cdot]_i$ indicates the ith element of the input vector. Refer to Section IV for an example.

Since ALCOR performs URA to allocate resources among activated users at each time instant, its performance depends on the employed URA method. Consequently, estimation errors in the URA policy input values can also affect overall performance, particularly in distributed scenarios, which are more prone to such errors due to the limited local computational capacities of users. To address this common issue in existing DL-based RA methods in the literature, [53] proposes an uncertainty injection algorithm during training, which enhances the generalizability of the policy against input estimation errors. As ALCOR is agnostic to the choice of URA method, incorporating such robust URA methods is feasible. Further investigation to improve URA generalizability within the framework of ALCOR can serve as a direction for future research.

To update κ_i^k in step 5, each user requires the normalization term $\max_{\ell}\{1 + \lambda_{\ell}^k\}$, which can be obtained by exchanging scalar values λ_{ℓ}^k among users via message-passing. This incurs an additional communication overhead in the distributed scenario. It is worth noting that obtaining local measurements $\boldsymbol{S}_{\xi,i}$ (see Section IV, the *distributed policy* paragraph, for an example of local measurements) also requires message-passing, which occurs at each time instant. In contrast, the scalar values λ_{ℓ}^k are communicated only after batches of samples ($2B$ time instants). Therefore, the processing latency and communication overhead incurred solely by the distributed ALCOR method are negligible compared to those of the employed distributed URA method [12], [13], [15], [19], [20], [27], [28]. Refer to Section IV for numerical communication overhead comparisons.

F. Convergence Study

The convergence rate of Algorithm 1 is provided in the following theorem. The formal convergence statement and its proof are presented in Appendix B.

Theorem 1 (Informal): Algorithm 1 converges with a rate of $\mathcal{O}(1/\sqrt{k})$, where k denotes the iteration number.

This theorem states that for a sufficiently large time window length T, ALCOR converges to an optimal time-sharing policy within the scope of the time window, and users can operate while meeting their current utility demands.

IV. Numerical Experiments

In this section, we evaluate the proposed RA scheme through numerical experiments across various communication scenarios. We address the power allocation problem where user utilities are defined as data rates. Specifically, the resources are the transmit powers $\boldsymbol{r} = \boldsymbol{p} = (p_1, \cdots, p_N) \in [0, p^{\max}]^N$, constrained by a maximum $p^{\max}$, and utilities are data rates, where $U_i = R_i$ and

$$R_i(\boldsymbol{H}, \boldsymbol{p}) := \log_2\left(1 + \frac{|h_{ii}|^2 p_i}{\sigma_n^2 + \sum_{j \neq i} |h_{ij}|^2 p_j}\right), \tag{9}$$

with σ_n^2 as the power of independent and identically distributed (IID) additive white Gaussian noise at the receivers. Two centralized and distributed URA schemes, adopted from [8], [15], are employed for policy training and performing URA among the activated users.

Centralized policy: A fully connected DNN is considered as the URA policy, following the structure outlined in [8]. Specifically, it consists of 4 layers with the number of neurons set to $\{400, 400, 200, 20\}$, unless specified otherwise. The policy input is the full channel matrix $\boldsymbol{H}$, denoted as $\boldsymbol{S} = \boldsymbol{H}$ in (1), and the output is the continuous transmit power of all users $\boldsymbol{p}$. With the considered policy, URA can be performed for a maximum number of 20 users, although the number of active users per time instant may be fewer. The activation function of hidden layers is the rectified linear unit (relu), while for the output layer, the *sigmoid* function is considered to ensure compliance with the transmit power box constraint. Batch normalization is also applied during training in all layers except the output layer. The policy structure is the same throughout the simulations unless specified otherwise.

Distributed policy: A fully connected deep neural network (DNN) is considered as the URA policy, comprising 4 layers

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

with the number of neurons set to $\{41, 100, 50, 1\}$. As shown in [15], all users utilize the same DNN, i.e., $\phi_1 = \ldots = \phi_N$. Each user collects local measurements to feed into its local policy and adjusts its continuous transmit power based on the policy output. The activation functions are consistent with the centralized case, and batch normalization is also employed. Define the following sets

$$\mathcal{I}_i^t := \{j \in [N], j \neq i \mid |h_{ij}^t|^2 p_j^t > \eta\sigma^2\},$$
$$\mathcal{O}_i^t := \{j \in [N], j \neq i \mid |h_{ji}^t|^2 p_i^t > \eta\sigma^2\},$$

for user i at time t. Here, $\mathcal{I}_i^t$ represents the set of users at time t causing interference to user i, where the interference power exceeds the threshold $\eta\sigma^2$. Conversely, $\mathcal{O}_i^t$ represents the set of users at time t receiving interference from user i exceeding the same threshold. In the simulations, the maximum cardinality of these sets is limited to 5 by selecting 5 most affected users in $\mathcal{O}_i^t$ and 5 most affecting users in $\mathcal{I}_i^t$, and the parameter η is fixed at 1. The local measurements of user i at time t, used as input for its policy ϕ_i, are obtained from [15] and are included here for completeness:

- Transmit power p_i^{t-1} and data rate R_i^{t-1};
- Direct channels $|h_{ii}^t|^2$ and $|h_{ii}^{t-1}|^2$;
- Received interference power from all users in two time instants: $\sum_{j=1,j\neq i}^N |h_{ij}^t|^2 p_j^{t-1} + \sigma^2, \sum_{j=1,j\neq i}^N |h_{ij}^{t-1}|^2 p_j^{t-2} + \sigma^2$;
- Received interference power from interfering neighbors in the set $\mathcal{I}_i$ in two timeslots: $\{|h_{ij}^t|^2 p_j^{t-1} \mid j \in \mathcal{I}_i^{t-1}\}, \{|h_{ij}^{t-1}|^2 p_j^{t-2} \mid j \in \mathcal{I}_i^{t-2}\}$;
- Data rate of the interfering neighbors: $\{R_j^{t-1} \mid j \in \mathcal{I}_i^{t-1}\}, \{R_j^{t-2} \mid j \in \mathcal{I}_i^{t-2}\}$;
- Normalized transmitted interference power to affected neighbors in the set $\mathcal{O}_i$ as $\left\{|h_{ji}^{t_i'}|^2 p_i^{t_i'} \left(\sum_{i=1,i\neq j}^N |h_{ji}^{t-1}|^2 p_i^{t-1} + \sigma^2\right)^{-1} \mid j \in \mathcal{O}_i^{t_i'}\right\}$, where t_i' is the last time when user i was active;
- Direct channel of interfered neighbors: $\{|h_{jj}^{t-1}|^2 \mid j \in \mathcal{O}_i^{t_i'}\}$;
- Data rate of interfered neighbors: $\{R_j^{t-1} \mid j \in \mathcal{O}_i^{t_i'}\}$.

The input set $\boldsymbol{S}_i$ for each user i comprises 41 elements, forming the local policy input. If the sets $\mathcal{I}_i$ and $\mathcal{O}_i$ contain fewer than 5 elements, zero-padding is applied at the respective policy input locations to ensure a fixed input size. The policy uses past measurements to accommodate channel instances that are correlated over time. However, initial simulations suggest that this correlation has negligible impact on the proposed RA scheme performance on the considered tasks. Therefore, we report the results with IID channel samples.

Policy training: In both centralized and distributed scenarios, policies are trained using an unsupervised approach, as discussed in [8], where the reward function during the training is (3). The distributions for $\boldsymbol{H}, \boldsymbol{\kappa}$, and $\boldsymbol{\xi}$ vary in different scenarios and are specified in the sequel.

Benchmarks:

(i) **Fixed DNN:** A DNN is considered in the centralized scenario for each user number N, where the utility demands $\boldsymbol{u}^{\text{fix}} \in \mathbb{R}^N$ are fixed and are to be satisfied in expectation. The detailed architecture of the DNNs will be mentioned subsequently for each N. The DNNs are trained in an unsupervised manner [8] using the reward function

$$\begin{aligned}&\text{reward}(\boldsymbol{\theta}, \boldsymbol{u}^{\text{fix}})\\ &= \sum\nolimits_{i=1}^N \mathbb{E}_{\boldsymbol{H}}\left[U_i^\theta(\boldsymbol{H})\right]\\ &\quad - L\sum\nolimits_{i=1}^N \max\{0, u_i^{\text{fix}} - \mathbb{E}_{\boldsymbol{H}}\left[U_i^\theta(\boldsymbol{H})\right]\},\end{aligned} \tag{10}$$

where constraint violations are penalized with $L = 100$. During training, expectations are approximated using batches of size 1000. Note that this reward function differs from the one in [8, eq. (14)], where the DNN is forced to satisfy the utility demands for every channel instance $\boldsymbol{H}$.

(ii) **WMMSE:** The conventional iterative URA scheme WMMSE [4] is considered for assessing the trained policies in URA and as a closed box URA scheme used within the proposed RA scheme.

(iii) **GP:** The iterative geometric programming (GP) RA scheme [3] is also considered, which is able to satisfy the utility demands for each channel instance. When comparing with this method, only channel instances feasible for the GP optimization problem are considered in the comparison.

(iv) **Meta-learning:** Existing works in the literature, such as DL-based RA methods in [20], [26], and [42], and DL-based resource scheduling and task offloading methods in [38], [39], [40], and [41], adapt to new network conditions (e.g., new user utility demands) by utilizing principles of meta-learning (refer to Section I for a literature review). Motivated by the success of meta-learning in these works, we consider this learning framework as another benchmark. In these works, constraint violations are penalized in the reward, and the MAML [43] principles are employed to initialize their policy, enabling quick adaptation to a new set of utility demands. For our specific RA problem, and in line with the mentioned works, we define the reward function as the one in (10) and initialize the policy using $\boldsymbol{\theta}$ optimized by:

$$\begin{aligned}&\underset{\boldsymbol{\theta}\in\mathbb{R}^n, \psi>0}{\text{maximize}}\ \mathbb{E}_{\boldsymbol{u}^{\min}}\left[\text{reward}(\boldsymbol{\theta}^+, \boldsymbol{u}^{\min})\right]\\ &\text{subject to}\ \boldsymbol{\theta}^+ := \boldsymbol{\theta} + \psi\nabla_{\boldsymbol{\theta}}\text{reward}(\boldsymbol{\theta}, \boldsymbol{u}^{\min}).\end{aligned} \tag{11}$$

where we use the same policy structure for a fair comparison, and $\boldsymbol{u}^{\min}$ is uniformly sampled from the interval $[0, 2.5]^N$. In the simulations, the expectation in (11) is approximated using exponential averaging with a decay factor of 0.7 over batches of size 1000 during training and 50 during inference. The optimization parameters $\boldsymbol{\theta}$ and ψ in (11) are optimized following the procedure outlined in [45] with step sizes of 10^{-4} and 5×10^{-6}, respectively.

(v) **Reinforcement Learning:** RL [13], [20], [28] is considered to address RA with dynamic user utility demands. Relevant simulation specifications are provided in Section IV-B.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

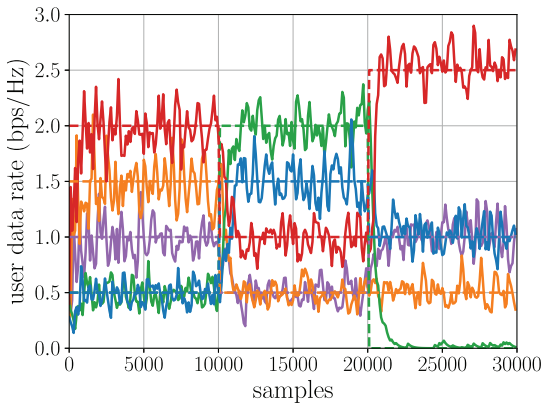


Fig. 1. Satisfying utility demands (on average) in a 5-user centralized scenario using ALCOR. Three time windows with different utility demands, $\boldsymbol{u}^{\min}$, are shown. Each solid line represents the instantaneous data rate of a user over time, and the dashed lines represent the dynamic utility demands of the corresponding colors.

In the simulations, the step sizes of Algorithm 1 are set as constants, with $\alpha_k = 0.9$ and suitable values of γ ranging from 1 to 10, depending on the specific experiment. Unless specified otherwise, the batch size is set to $B = 25$, and the utility demands $\boldsymbol{u}^{\min}$ are chosen such that the presentation is clear and the feasibility is guaranteed. In the rest of this section, two different channel scenarios are considered for performance evaluation.

A. Circularly Symmetric Channel

In this section, the channel coefficients are assumed to follow a circularly symmetric complex normal distribution, i.e. $h_{ij} \sim \mathcal{CN}(0, 1)$, $p^{\max} = 1$, and the SNR is set to 15 dB.

Adaptivity: In the first experiment, ALCOR's adaptivity is illustrated in Fig. 1. In a scenario with 5 users, the utility demands values are set as $\boldsymbol{u}^{\min} = (0.5, 0.5, 1, 1.5, 2)$, $\boldsymbol{u}^{\min} = (2, 1.5, 0.5, 0.5, 1)$, and $\boldsymbol{u}^{\min} = (0, 1, 1, 0.5, 2.5)$ in bps/Hz over three consecutive time windows. For clarity of presentation, the chosen utility demands are strict, ensuring that the average user data rates match these demands after convergence. The figure clearly shows that the proposed RA scheme can accurately adapt to the changing utility demands and meet them on average within a few iterations after each utility demand update. In the sequel, we focus on a single time window—starting from an update in utility demands—to study ALCOR's performance and compare it with other RA methods.

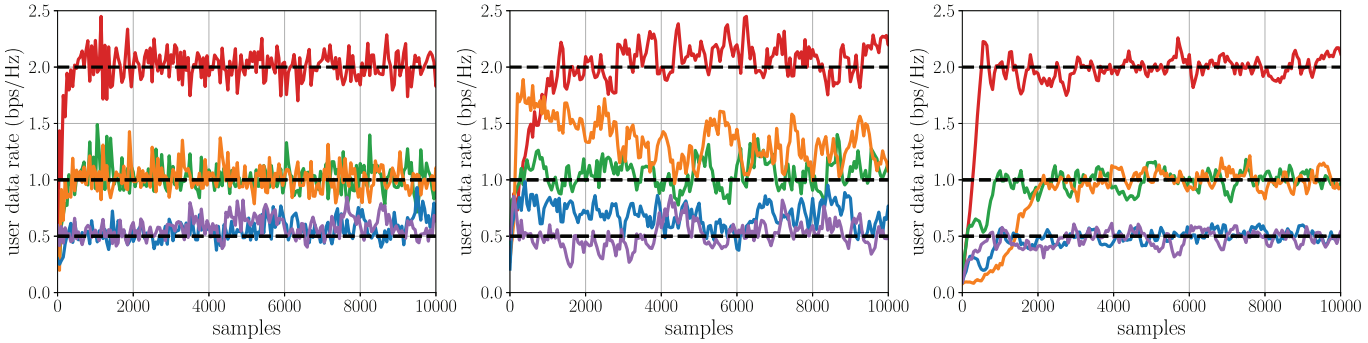
The performance of the proposed RA scheme in addressing dynamic utility demands is compared with the *vanilla learning* and meta-learning approaches. In both approaches, the RA policy is retrained in an unsupervised manner using the reward function in (10) for the new utility demands. In vanilla learning, the policy is initialized randomly, while in meta-learning, the policy is initialized using the optimization in (11), with a single update for $\boldsymbol{\theta}^+$ [45]. It should be noted that meta-learning requires more complex policy training following (11), which involves second-order (Hessian) optimization, as considered here, or first-order optimization, as in e.g., [26]. According to Fig. 2, ALCOR adapts to user utility demands with simpler updates following Algorithm 1, i.e., without fine-tuning the policy. Its adaptation is faster and more stable compared to the other approaches. Although the objective of the considered meta-learning approach is to find a policy capable of adapting to new utility demands with a single update—i.e., initializing the policy with the optimal $\boldsymbol{\theta}$ and then fine-tuning it with the optimal step size ψ optimized by (11)—it is evident that meta-learning cannot adapt to utility demands with a single update. Therefore, in our experiments, meta-learning adaptation continues with additional updates with a diminishing step size, in line with existing works on RA, e.g., [20], [26], [42]. Compared to vanilla learning, meta-learning satisfies utility demands more quickly due to its educated initialization. However, its slower adaptation compared to ALCOR can be attributed to the fact that RA tasks (with different utility demands) share limited high-level structures, necessitating more iterations for fine-tuning the policy. The complexity involved in meta-learning—both in training (initializing) the policy via (11) and in fine-tuning it when addressing new sets of utility demands—underscore the motivation for using ALCOR in the considered constrained DL-based RA task.

Unconstrained RA: In the next experiment, we assess the performance of the centralized and distributed policies, which are trained using (3), for URA as compared to WMMSE. Two training strategies are employed: 1) *Diverse training*: During training, $\boldsymbol{\kappa}$ is randomly and uniformly chosen from the set $\boldsymbol{\kappa} \in \{0.2 \times \mathbf{1}, 0.5 \times \mathbf{1}, \mathbf{1}\}$, where $\mathbf{1}$ is a vector of all ones. At each time instant, each user is independently switched on with a probability of κ_i during the training. 2) *Non-diverse training*: During training, $\boldsymbol{\kappa} = \mathbf{1}$, i.e., all users are always switched on.

The performance of the policies in URA is presented in Table II. Based on Table II, diverse training results in a slightly better generalization compared to non-diverse training. The performance of the policies in URA is also comparable to WMMSE, i.e., the policies maintain good performance in URA while making URA faster. The same performance is evident in [8]. As diverse and non-diverse training strategies are comparable in performance, for the remaining experiments, we employ non-diverse training, which is the standard training method in the URA literature.

Meeting utility demands: In the next experiment, the proposed RA scheme is compared against the GP method and the DNNs trained using the reward function (10) for fixed utility demands (fixed DNNs). As the GP method requires a feasible RA problem at each time instant, only channels feasible for the corresponding utility demands are considered. The feasibility is determined using the procedure outlined in [54]. Centralized and distributed policies are trained using both feasible and infeasible training samples with performance assessed under varying numbers of users in the communication system. The fixed DNNs have the following structures $\{100, 400, 200, 10\}$, $\{400, 400, 200, 20\}$, and $\{2500, 400, 400, 50\}$ for 10, 20, and 50 users, respectively. Table III reports the achieved SR of users while the utility demands are satisfied, on average. The user utility demands are reported in the caption where we use the notation $u^{\min}_{a:b}$ to refer to the utility demand of

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.



(a) *ALCOR* **(b)** *Meta-Learning* **(c)** *Vanilla Learning*

Fig. 2. Performance comparison of different RA schemes in a 5-user centralized scenario with a similar DNN structure of $\{25, 25, 25, 5\}$. Each color specifies the instantaneous data rate of a user over time, where the utility demands (data rate demands) $\boldsymbol{u}^{\min} = (0.5, 0.5, 1, 1, 2)$ are depicted by black dashed lines. While the proposed scheme (ALCOR) does not involve any DNN retraining, meta-learning and vanilla learning (re)train the DNN using the reward function of (10). The (re)training is performed with a diminishing step size of $\psi^\star/1+k$, where k is the epoch counter and $\psi^\star \approx 0.3$ is the optimized decision variable of (11). ALCOR demonstrates faster and more stable convergence.

TABLE II

URA PERFORMANCE COMPARISON BETWEEN DL-BASED POLICIES AND WMMSE FOR $N = 20$ USERS. AVERAGE SR IS REPORTED IN bps/Hz. CENTRALIZED AND DISTRIBUTED POLICIES ARE CONSIDERED WITH DIVERSE TRAINING (WHERE $\boldsymbol{\kappa}$ IS RANDOMLY SAMPLED DURING TRAINING) AND NON-DIVERSE TRAINING (WHERE $\boldsymbol{\kappa} = \mathbf{1}$ REMAINS CONSTANT DURING TRAINING) STRATEGIES. THESE POLICIES ARE ONLY RESPONSIBLE FOR UNCONSTRAINED RA AND CANNOT SATISFY USER UTILITY DEMANDS

$\boldsymbol{\kappa}$	cntr. (divs.)	cntr. (non-divs.)	dist. (divs.)	dist. (non-divs.)	WMMSE
0.25	5.88	5.63	5.82	5.45	5.88
0.5	6.38	6.30	6.31	6.09	6.8
0.75	6.56	6.56	6.52	6.35	7.29
1	6.71	6.72	6.66	6.62	7.72

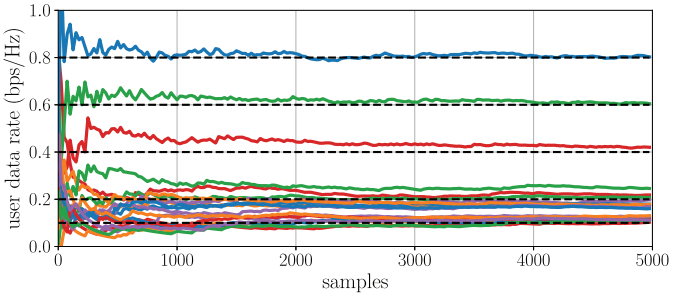
TABLE III

COMPARISON OF VARIOUS RA SCHEMES WITH DIFFERENT NUMBERS OF USERS AND UTILITY DEMANDS. AVERAGE SR IS REPORTED IN bps/Hz. ALL UTILITY DEMANDS ARE SATISFIED WITHOUT VIOLATION. THE UTILITY DEMANDS ARE ZERO EXCEPT FOR THE FOLLOWING USERS: $N = 10$: $\boldsymbol{u}^{\min}_{1:10} = 0.25$, $N = 20$: $\boldsymbol{u}^{\min}_{1:12} = 0.1$, $N = 50$: $\boldsymbol{u}^{\min}_{1:15} = 0.05$. THE FIXED DNNS ARE TRAINED USING THE REWARD FUNCTION (10) WITH THE UTILITY DEMANDS CONSIDERED FOR EACH N

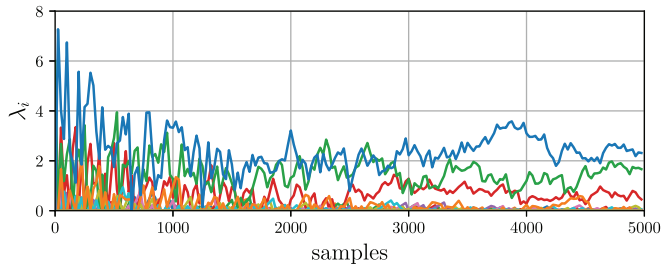
N	ALCOR (cntr.)	ALCOR (dist.)	fixed DNN	GP
10	5.41	4.37	6.82	3.86
20	6.56	6.37	6.73	3.81
50	6.67	5.95	6.16	4.73

users $i \in \{a, a+1, \cdots, b\}$. The results indicate that the proposed method outperforms the GP method in terms of SR. Furthermore, although the fixed DNNs, trained for fixed utility demands, achieve the highest SR, they can only satisfy the fixed utility demands for which they were trained. A notable advantage of the proposed method is that it is not limited to feasible channel instances, unlike methods such as the one proposed in [8]. Consequently, the proposed method does not require the computationally complex feasibility check during training. The utility demand violations are not reported in Table III, as they are equal to zero for all the reported RA schemes.

In the following, ALCOR is assessed under stricter utility demands. In this scenario, the problem is infeasible for most test samples, and the GP method is unable to provide any



(a) *Moving average of data rates. Each line represents a user's data rate moving average, and the dashed lines indicate the corresponding utility demands.*



(b) λ_i *values corresponding to the data rates in (a), with matching colors.*

Fig. 3. User utilities and the corresponding λ_i values generated by ALCOR for the distributed RA scenario reported in Table IV, with $N = 20$.

solution. The utility demand violation is reported as:

$$\text{viol.} := \max_{i \in [N]} \{\max\{0, u_i^{\min} - R_i\}/u_i^{\min}\} \times 100. \quad (12)$$

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
STRICTER UTILITY DEMANDS THAN THE ONES IN TABLE III. AVERAGE SR IS REPORTED IN bps/Hz AND VIOLATION IS CALCULATED BY (12). THE UTILITY DEMAND ARE ZERO EXCEPT FOR THE FOLLOWING USERS: $N = 10$: $\boldsymbol{u}^{\min}_{1:10} = 0.1$ EXCEPT $u^{\min}_5 = 0.6, u^{\min}_7 = 0.8, u^{\min}_9 = 1$, $N = 20$: $\boldsymbol{u}^{\min}_{1:10} = 0.1$ EXCEPT $u^{\min}_5 = 0.4, u^{\min}_7 = 0.6, u^{\min}_{12} = 0.2, u^{\min}_{19} = 0.8$, $N = 50$: $\boldsymbol{u}^{\min}_{20:30} = 0.3$. THE FIXED DNNS ARE TRAINED WITH THE UTILITY DEMANDS REPORTED IN TABLE III

	ALCOR (cntr.)		ALCOR (dist.)		fixed DNN	
N	SR	viol.	SR	viol.	SR	viol.
10	3.13	0.61	3.85	0.01	6.35	37.71
20	3.97	0.00	5.35	0.00	6.58	60.45
50	6.09	1.37	4.75	2.84	6.17	64.14

Additionally, the fixed DNNs are the same as those in Table III. The results are summarized in Table IV. According to the results, the proposed RA scheme can accurately satisfy dynamic utility demands while maintaining a high SR. As expected, the fixed DNNs cannot meet the new set of utility demands, and they require retraining of their policies. The users' average data rates are also plotted in Fig. 3 for the distributed case with $N = 20$ (see the supplementary material for additional plots). It is evident that the user data rates rapidly converge to meet the demands upon any change. The values $\boldsymbol{\lambda}$ are also plotted, showing higher values for users with greater demands. This finding aligns with what is expected from (6) and (7), where higher values of λ_i increase the probability κ_i of user i being active, thereby pushing its expected utility toward higher values.

Regarding the communication overhead incurred during the distributed RA, the URA policy adopted from [15] requires sharing 25 variables per channel update per user. Considering a channel coherence time of 10 ms and 32-bit floating-point precision per value, the communication overhead (without compression) is roughly 78 kbps per user. To obtain the normalization terms in step 5, ALCOR needs to share (or update upon any change) the normalization term $\max_\ell\{1+\lambda^k_\ell\}$ every $2B$ time instants among the users. Our experiments show an average rate of 67 bps per user, which is negligible compared to the overhead incurred by the employed distributed URA and other existing RA methods [12], [13], [15], [19], [20], [27], [28].

B. Multi-Link per Cell Channel

In this section, the proposed RA scheme is assessed considering a communication system consisting of 7 cells. Each cell contains a number of transmitters and receivers, with transmitters positioned at the center and receivers uniformly distributed within the cell. The minimum and maximum allowed distances between the transmitters and the receivers are set to $r = 50$ and $R = 500$ meters, respectively. The carrier frequency is set to 6 GHz, and the sub-channel bandwidth is 1 MHz. The noise power, σ^2_n, is set to -114 dBm, and the maximum transmit power is $p^{\max} = 38$ dBm. The large-scale fading component, α_{ji}, which models the path loss and shadowing between transmitter i and receiver j, is determined by $\alpha_{ji} = 120.9 + 37.6\log_{10}(d_{ji}) + \zeta_{ji}$ dB, where d_{ji} denotes the distance measured in kilometers, and $\zeta_{ji} \sim \mathcal{N}(0, \sigma^2_s)$ represents the shadowing with a standard deviation of $\sigma_s = 8$. The small-scale fading component follows IID circularly symmetric Gaussian distribution, i.e., $g_{ij} \sim \mathcal{CN}(0, 1)$. Hence, the channel coefficients are derived as $|h_{ij}|^2 = |g_{ij}|^2/10^{\alpha_{ij}/10}$. The small-scale fading component changes every 10 ms and the large-scale fading component changes 10 times slower. Both the centralized and distributed DNNs are trained using possibly infeasible samples in the training stage. The structures of the DNNs are the same as those described in Section IV-A.

In this section, we assess ALCOR with a broader range of resource types and resource constraints. Hence, in addition to power allocation, the task of sub-channel assignment is added to the RA problem, where each user has access to only one sub-channel among 5 possible sub-channels at each time instant [13], [20], [28], [42]. Each user i has a packet arrival rate following a Poisson distribution with parameter ν_i, with packets waiting in a buffer queue for transmission [28]. Packets have 4000 bits each. Users are either delay-sensitive, where the queue must have a maximum length, or delay-tolerant, where the queue needs only to be stable without a maximum length threshold. We consider a queue to be stable as long as its length does not increase over three consecutive batches. The average packet arrival rate of each user is randomly selected via $\nu_i \in [50, 400]$ packets per second. We consider 20 users, among which 10 are delay-sensitive with varying latency constraints randomly selected from $[20, 80]$ ms. For a delay-sensitive user i, the corresponding element in $\hat{F}(\boldsymbol{H}_\xi)$ in (6) is defined as $q_i - d^{\max}_i$, where q_i is the queue length and $d^{\max}_i$ is its threshold, depending on the user's latency constraint. For a delay-tolerant user i, this formulation changes to $c_i - c^{\max}_i$, where c_i is a counter indicating the number of consecutive batches with increases in queue size, and $c^{\max}_i = 3$ is its threshold.

To also study the impact of different URA schemes on ALCOR's accuracy and convergence in the mentioned communication scenario, the performance is reported while employing three different URA policies: 1) the centralized URA policy considered in the previous subsection; 2) a GNN-based distributed URA policy; and 3) a trivial URA policy (maximum power) that allocates $p^{\max}$ and a random sub-channel to active users. The architecture of the centralized URA policy is updated to have 4 layers, with the number of neurons set to $\{400, 512, 256, 120\}$, with two last layers. The first of the last two layers has 20 neurons with sigmoid activation functions for power allocation $p_i \in [0, p^{\max}], \forall i$, and the second is a categorical layer with 5 neurons and a softmax activation function for each user for sub-channel assignment. The GNN-based distributed URA follows the principles of aggregation GNNs proposed in [19]. For each user policy i at time instant t, the policy input is defined by $\boldsymbol{\mathcal{H}}^t_i := (y^{1,t}_i, \cdots, y^{K,t}_i) \in \mathbb{R}^K$, where $y^{k,t}_i$ is the ith element of the vector $\boldsymbol{y}^{k,t} := \left[\prod_{t'=0}^{k-1} |\tilde{\boldsymbol{H}}^{t-t'^T}|\right] \mathbf{1} \in \mathbb{R}^N$. Here, $K = 8$ defines the range of neighbors (hops) that each user considers in its input, $\tilde{\boldsymbol{H}}$ is the channel matrix $\boldsymbol{H}$ where the channel coefficients associated with the $m = 5$ most affected receivers

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

TABLE V
PERFORMANCE OF ALCOR EMPLOYING VARIOUS URA SCHEMES FOR $N = 20$, COMPARED WITH RL-BASED RA

	ALCOR with various URA schemes			
	centralized	GNN-based dist.	max power	RL-based RA
average SR (bps/Hz)	31.7	29.4	16.2	32.6
average latency (ms)	64.1	77.1	567.8	55.2
latency violation (%)	1.1	2.3	90.1	36.1
convergence (ms)	1134.9	962.4	-	4307.5

by each user's transmitter are preserved and all other elements are set to zero, $\mathbf{1} \in \mathbb{R}^N$ represents a vector of all ones, and $|\cdot|$ is the pointwise absolute value operator. A fully connected DNN with the structure $\{K, 256, 128, 1+5\}$ neurons is considered for each user, with one neuron in the last layer for power allocation and a categorical layer of 5 neurons for sub-channel assignment.

ALCOR's performance is compared to RL-based RA, where a centralized policy is trained using REINFORCE [16], [19], with the observation state space $\mathcal{O} = \{\boldsymbol{H}, \boldsymbol{u}^{\min}, \boldsymbol{q}\}$, where $\boldsymbol{q}$ is a vector containing the queue lengths of all users. The action space $\mathcal{A}$ includes $N + 5 \times N$ values, similar to the output dimension of the centralized URA, and accounts for all users' transmit powers and sub-channels. The discounted reward $\sum_{t=0}^{\infty} \sum_{i=1}^{N} 0.9^t r_i^t$ is considered, where the reward for user i at time instant t is set equal to the user's data rate, $r_i^t = R_i^t$, if all the user's constraints are met, and $r_i^t = 0$ otherwise (refer to [20] for further details). Based on the state and action space dimensions mentioned above, a DNN with the structure $\{440, 512, 256, 120\}$ is used in this simulation.

The performance of the above-mentioned RA methods is summarized in Table V, where ALCOR is evaluated separately with three different URA methods and compared against the RL-based RA. The comparison is made in terms of SR averaged over time, average latency, latency violation—defined as the percentage of time during which latency constraints are violated (i.e., queue lengths exceed the threshold) after convergence—and convergence rate—defined as the average time required for the method to satisfy violated constraints following any update in the latency constraints of delay-sensitive users. According to the results, it is evident that ALCOR's performance is highly dependent on the employed URA scheme. Specifically, for the traffic load of $\nu_i \in [50, 400]$—which represents a moderate to high traffic load in the considered communication setup—the trivial maximum power URA in Table V is unable to satisfy user latency constraints, resulting in unstable queue sizes without convergence. Queues that do not converge under this policy are excluded from the reported average latency. On the other hand, the centralized URA policy achieves high average SR and low latency while properly meeting the constraints, and it converges quickly to new constraints. The GNN-based distributed policy exhibits comparable performance. It is noted that GNN-based RA methods demonstrate higher scalability compared to the centralized approaches, thanks to the considered graph representation of the communication network [19]. Although RL-based RA maintains stable queues and lower latencies with higher average SR values, it fails to satisfy all dynamic latency constraints and demonstrates a lower convergence rate. This reduced performance can be attributed to the higher complexity the policy must learn—namely, both resource allocation and constraint satisfaction. In contrast, ALCOR offloads the task of constraint satisfaction from the policy to an iterative process, allowing the policy to focus entirely on URA among the active users.

The employed GNN-based RA requires sharing $K \times m = 40$ variables per channel update per user. With 32-bit floating-point precision and a 10 ms channel coherence time, the communication overhead incurred by the GNN-based RA is 125 kbps per user. As mentioned in the previous subsection, the communication overhead required for sharing the normalization term is a comparably negligible value of 67 bps per user, according to the experiments.

V. CONCLUSION AND FUTURE DIRECTIONS

In this paper, a DL-based RA scheme has been proposed that utilizes DNNs as URA policies within an iterative optimization algorithm to meet dynamic user utility demands in expectation. The optimization algorithm addresses a time-sharing problem by optimizing the on-off status of users. In parallel, URA policies are responsible for performing unconstrained RA among active users at each time instant to maximize their SU. The proposed approach is agnostic to URA schemes. Consequently, depending on the employed URA policy, the proposed RA scheme can be implemented in either a centralized or distributed scenario. Convergence analyses have been provided, relying on variational inequalities, to establish convergence guarantees.

The proposed user selection procedure, based on the Bernoulli distribution, can be extended to account for user channel conditions. In this extension, users with the best channel conditions are activated while ensuring that their activation probabilities equal κ_i, as optimized by the algorithm. Extending ALCOR to scenarios where each user has multiple dynamic constraints would also be beneficial for many real-world applications. Additionally, exploring the application of ALCOR to other RA settings—such as computational power allocation, service migration, network slicing, and simultaneous wireless information and power transfer—under appropriate dynamic constraints, represents a compelling direction for future research.

APPENDIX A

In this Appendix, we present the notation, basic assumptions, and a lemma that are required for the convergence analysis.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

A. Notation

The distance from $\boldsymbol{\lambda} \in \mathbb{R}^N$ to a set $\mathcal{X} \subseteq \mathbb{R}^N$ is given by $\operatorname{dist}(\boldsymbol{\lambda}, \mathcal{X}) := \min_{\boldsymbol{v} \in \mathcal{X}} \|\boldsymbol{v} - \boldsymbol{\lambda}\|$. An operator $F : \mathbb{R}^N \rightrightarrows \mathbb{R}^n$ maps each point $\boldsymbol{\lambda} \in \mathbb{R}^N$ to a set $F(\boldsymbol{\lambda}) \subseteq \mathbb{R}^n$. The graph of operator F is denoted by $\operatorname{gph} F := \{(\boldsymbol{\lambda}, \boldsymbol{v}) \in \mathbb{R}^N \times \mathbb{R}^n \mid \boldsymbol{v} \in F(\boldsymbol{\lambda})\}$. The set of zeros is defined by $\operatorname{zer} F := \{\boldsymbol{\lambda} \in \mathbb{R}^N \mid 0 \in F(\boldsymbol{\lambda})\}$. Operator F is ρ-strongly monotone with some $\rho \geq 0$ if $\langle \boldsymbol{v} - \boldsymbol{v}', \boldsymbol{\lambda} - \boldsymbol{\lambda}' \rangle \geq \rho \|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|^2$ for all $(\boldsymbol{\lambda}, \boldsymbol{v}), (\boldsymbol{\lambda}', \boldsymbol{v}') \in \operatorname{gph} F$. The operator is called monotone if $\rho = 0$. The deviation between two distributions $\mathcal{D}$ and $\mathcal{D}'$ is measured using the Wasserstein-1 distance:

$$W_1(\mathcal{D}, \mathcal{D}') = \sup_{g \in \operatorname{lip}_1} \{\mathbb{E}_{x \sim \mathcal{D}}[g(x)] - \mathbb{E}_{y \sim \mathcal{D}'}[g(y)]\}, \quad (13)$$

where lip_1 denotes the set of 1-Lipschitz continuous functions $g : \mathbb{R}^N \to \mathbb{R}^N$. During the proof, we use the Young's inequality with $a, b \in \mathbb{R}^n$ and $e > 0$: $\|a - b\|^2 \leq (1 + e)\|a\|^2 + (1 + 1/e)\|b\|^2$.

B. Preliminary Assumptions and Lemmas

The following assumptions are required for the convergence:

Assumption 2 (Feasibility of Inclusion Problem (7)): The inclusion problem (7) is feasible, namely, there exists at least one $\boldsymbol{\lambda}^\star \in \mathbb{R}^N$ such that $F(\boldsymbol{\lambda}^\star) \leq \mathbf{0}$.

In Algorithm 1, ALCOR utilizes a DL model within an iterative optimization algorithm. The DL model is integrated into the mapping F within the inclusion problem (7). The feasibility of the problem then depends on several factors, including the utility demands $\boldsymbol{u}^{\min}$, the capability of the DL model to generalize the URA problem, the quality of training in Section III-B, and the characteristics of the communication network, such as the quality of communication channels. Hence, the communication network requires to have mechanisms to ensure this feasibility (refer to the supplementary material for further details).

Assumption 3 (Assumption on the Expected Utility): The expected utility $\mathbb{E}_{\boldsymbol{H}}\left[\boldsymbol{U}^{\boldsymbol{\theta}^L}(\boldsymbol{H}_\xi)\right]$ is Lipschitz continuous in $\boldsymbol{\xi}$. Namely, there is a $\beta > 0$ such that for all $\boldsymbol{\xi}, \boldsymbol{\xi}'$:

$$\left\|\mathbb{E}_{\boldsymbol{H}}\left[\boldsymbol{U}^{\boldsymbol{\theta}^L}(\boldsymbol{H}_\xi)\right] - \mathbb{E}_{\boldsymbol{H}}\left[\boldsymbol{U}^{\boldsymbol{\theta}^L}(\boldsymbol{H}_{\xi'})\right]\right\| \leq \beta \|\boldsymbol{\xi} - \boldsymbol{\xi}'\|.$$

This assumption ensures that when the USV changes, the change in the expected utilities is bounded. This is easily verifiable for many utilities e.g., data rates where the resource is the transmit power.

To continue with the convergence analysis, we recast the problem (7) to the following inclusion problem:

$$\text{find } \boldsymbol{\lambda} \in \mathbb{R}^N \text{ such that } \mathbf{0} \in T(\boldsymbol{\lambda}) := F(\boldsymbol{\lambda}) + A(\boldsymbol{\lambda}), \quad (14)$$

with $A(\boldsymbol{\lambda}) := \partial\varphi(\boldsymbol{\lambda})$ as the subdifferential of the indicator function φ defined as $\varphi(\boldsymbol{\lambda}) = 0$ if $\boldsymbol{\lambda} \in \mathbb{R}^N_+$ and $\varphi(\boldsymbol{\lambda}) = -\infty$ otherwise. The inclusion problem (14) ensures finding a variable $\boldsymbol{\lambda}$ such that $F(\boldsymbol{\lambda}) \leq \mathbf{0}$. This is achieved under two conditions: 1) $\mathbf{0} \in T(\boldsymbol{\lambda})$ with $[F(\boldsymbol{\lambda})]_i = 0$ and $[A(\boldsymbol{\lambda})]_i = 0$ for some $i \in [N]$. 2) $\mathbf{0} \in T(\boldsymbol{\lambda})$ with $[F(\boldsymbol{\lambda})]_i < 0$ and $[A(\boldsymbol{\lambda})]_i > 0$ for some $i \in [N]$. The second case is possible thanks to the definition of κ_i allowing κ_i to be nonzero when $\lambda_i = 0$, thereby permitting $[F(\boldsymbol{\lambda})]_i < 0$. It is remarked that, by the definition of A, the case where $0 \in T(\boldsymbol{\lambda})$ with $[F(\boldsymbol{\lambda})]_i > 0$ and $[A(\boldsymbol{\lambda})]_i < 0$ never occurs. The necessary assumptions on the mappings of inclusion problem (14) are as follows:

Assumption 4 (Assumptions on the Mappings $F, \hat{F}$):

(i) The mapping $F : \mathbb{R}^N \to \mathbb{R}^N$ is Lipschitz continuous with the constant $L_F \in [0, +\infty)$: $\|F(\boldsymbol{\lambda}) - F(\boldsymbol{\lambda}')\| \leq L_F \|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|, \ \forall \boldsymbol{\lambda}, \boldsymbol{\lambda}' \in \mathbb{R}^N_+$;

(ii) Weak Minty variational inequality (WMVI) holds, i.e., there exists a nonempty set $\mathcal{S}^\star \subseteq \operatorname{zer} T$ such that for all $\boldsymbol{\lambda}^\star \in \mathcal{S}^\star$ and some $\rho \in (-1/2L_F, +\infty)$

$$\langle \boldsymbol{v}, \boldsymbol{\lambda} - \boldsymbol{\lambda}^\star \rangle \geq \rho \|\boldsymbol{v}\|^2, \quad \text{for all } (\boldsymbol{\lambda}, \boldsymbol{v}) \in \operatorname{gph} T;$$

(iii) The stochastic oracle has a bounded variance: $\mathbb{E}_{\boldsymbol{H}, \boldsymbol{\xi} \sim \mathcal{D}^\xi(\boldsymbol{\kappa})}\left[\|\hat{F}(\boldsymbol{H}_\xi) - F(\boldsymbol{\lambda})\|^2\right] \leq \sigma^2, \forall \boldsymbol{\lambda} \in \mathbb{R}^N_+$.

Assumption 4(i) assumes that mapping F in (7) and (14) is Lipschitz continuous. This assumption can be readily fulfilled by taking Assumption 3 along with the structure of distribution $\mathcal{D}^\xi$ in Definition 1. The following lemma investigates this assumption:

Lemma 1 (Sufficient Conditions for Assumption 4(i)): Take Assumption 3, then Assumption 4(i) holds with $L_F = \beta\omega$, where $\omega > 0$ is the Lipschitz constant of the distribution $\mathcal{D}^\xi$ (see (15) for definition).

Proof: This proof is closely following the proof in [55, lem. 2.1] and is presented here for completeness. By the Bernoulli distribution $\mathcal{D}^\xi$ associated to USVs $\boldsymbol{\xi}$ in Definition 1, and also the distribution distance measure defined in (13), the smoothness notion can be extended to the distributions: a constant $L_{\mathcal{D}^\xi} > 0$ exists such that

$$W_1(\mathcal{D}^\xi(\boldsymbol{\kappa}), \mathcal{D}^\xi(\boldsymbol{\kappa}')) \leq L_{\mathcal{D}^\xi} \|\boldsymbol{\kappa} - \boldsymbol{\kappa}'\|. \quad (15)$$

Moreover, as in (7) $\boldsymbol{\kappa}$ is a continuous function of $\boldsymbol{\lambda}$, a constant $L_\kappa > 0$ exists such that

$$\|\boldsymbol{\kappa}(\boldsymbol{\lambda}) - \boldsymbol{\kappa}(\boldsymbol{\lambda}')\| \leq L_\kappa \|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|.$$

Hence, the composition is also Lipschitz continuous:

$$W_1(\mathcal{D}^\xi(\boldsymbol{\kappa}(\boldsymbol{\lambda})), \mathcal{D}^\xi(\boldsymbol{\kappa}(\boldsymbol{\lambda}'))) \leq \omega \|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|,$$

where $\omega := L_\kappa L_{\mathcal{D}^\xi}$ is defined.

Define $g(\boldsymbol{\xi}) := \langle \boldsymbol{v}, \mathbb{E}_{\boldsymbol{H}}\left[\boldsymbol{U}^{\boldsymbol{\theta}^\star}(\boldsymbol{H}_\xi)\right]\rangle$ with a vector $\boldsymbol{v}$ where $\|\boldsymbol{v}\| \leq 1$. By Assumption 3, $g(\boldsymbol{\xi})$ is β-Lipschitz continuous in $\boldsymbol{\xi}$, since:

$$\begin{aligned}\|g(\boldsymbol{\xi}) - g(\boldsymbol{\xi}')\| &\leq \|\boldsymbol{v}\| \left\|\mathbb{E}_{\boldsymbol{H}}\left[\boldsymbol{U}^{\boldsymbol{\theta}^\star}(\boldsymbol{H}_\xi)\right] - \mathbb{E}_{\boldsymbol{H}}\left[\boldsymbol{U}^{\boldsymbol{\theta}^\star}(\boldsymbol{H}_{\xi'})\right]\right\| \\ &\leq \beta \|\boldsymbol{\xi} - \boldsymbol{\xi}'\|,\end{aligned}$$

hence:

$$\begin{aligned}\langle \boldsymbol{v}, F(\boldsymbol{\lambda}') - F(\boldsymbol{\lambda}) \rangle &= \mathbb{E}_{\boldsymbol{\xi} \sim \mathcal{D}^\xi(\boldsymbol{\kappa})}[g(\boldsymbol{\xi})] - \mathbb{E}_{\boldsymbol{\xi}' \sim \mathcal{D}^\xi(\boldsymbol{\kappa}')}[g(\boldsymbol{\xi}')] \\ &\leq \beta W_1(\mathcal{D}^\xi(\boldsymbol{\kappa}), \mathcal{D}^\xi(\boldsymbol{\kappa}')),\end{aligned}$$

where in the equality the definition of F in (7) is considered, and the inequality is due to (13) considering $g \in \operatorname{lip}_\beta$. Taking $v = (F(\boldsymbol{\lambda}') - F(\boldsymbol{\lambda}))/\|F(\boldsymbol{\lambda}') - F(\boldsymbol{\lambda})\|$ and then using (15) completes the proof with $L_F = \beta L_\kappa L_{\mathcal{D}^\xi}$.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

It is remarked that there is no control over the constants β and $L_{\mathcal{D}^\xi}$ and they are determined by the communication network and the Bernoulli distribution, respectively. However, the constant L_κ is adjustable by modifying c in the function $\kappa_i = {}^{(c+\lambda_i)}/_{\max_\ell\{c+\lambda_\ell\}}$ in (7). ■

The class of nonmonotone mappings for which convergence analyses are possible is characterized by Assumption 4(ii), where nonmonotonicity can be captured thanks to the possibly negative values of ρ. The mapping F defined in (7) may not be (strongly) monotone, since $\langle \boldsymbol{v}_1-\boldsymbol{v}_2, \boldsymbol{\lambda}_1-\boldsymbol{\lambda}_2\rangle \ngeq \rho\left\|\boldsymbol{\lambda}_1-\boldsymbol{\lambda}_2\right\|^2$ with a $\rho \geq 0$ for all $(\boldsymbol{\lambda}_1, \boldsymbol{v}_1), (\boldsymbol{\lambda}_2, \boldsymbol{v}_2) \in \operatorname{gph} T$. This relation can be interpreted as follows: increasing λ_i for user i results in increasing the probability of user i being activated, which consequently increases its expected utility and decreases $[F(\boldsymbol{\lambda})]_i$ due to the definition. Therefore, the relation cannot be guaranteed with a $\rho \geq 0$, whereas Assumption 4(ii), with the possibility of $\rho < 0$, readily captures the relation. It is worth noting that this assumption may be considered restrictive, since ρ is lowerbounded by $-{}^1/_{2L_F}$, a value that depends on $L_\kappa, L_{\mathcal{D}^\xi}$, and β according to Lemma 1. Nevertheless, extensive numerical studies in Section IV have not witnessed divergence as long as Assumption 2 holds (see the supplementary material where Assumption 2 does not hold). Finally, Assumption 4(iii) is a standard assumption in stochastic optimization literature, e.g., [51], [56].

APPENDIX B

With the necessary assumptions considered and the recast problem in (14), the next theorem demonstrates the convergence of the iterations generated by Algorithm 1.

Theorem 2 (Subsequential Convergence): Suppose that Assumption 1 to 4 hold. Moreover, take the stepsize sequence $(2 = k \in \mathbb{N}, 3 =)^{[}_{\alpha_k} k = 0][K] \in (0,1)$ and $\gamma \in (0, {}^1/_{L_F})$, the batch size sequence $(2 = k \in \mathbb{N}, 3 =)^{[}_{B_k} k = 0][K] \in \mathbb{N}$, and suppose

$$\bar{\mu} := 2\frac{\rho}{\gamma} + \frac{1-\sqrt{\alpha^{\max}}}{1+\sqrt{\alpha^{\max}}} - \alpha^{\max} - 2\bar{\alpha}\gamma^2 L_F^2 \mathcal{A} > 0,$$

where $\alpha^{\max} := \max_k\{\alpha_k\}$, $\bar{\alpha} := \max_k\{{}^{\alpha_k}/_{\alpha_{k+1}}\}$, and $\mathcal{A} := 3\left(\frac{1}{\sqrt{\alpha^{\max}}(1-\gamma L_F)^2} + \frac{1-\sqrt{\alpha^{\max}}}{\sqrt{\alpha^{\max}}}\right)$. Then, the iterates $\boldsymbol{\lambda}^k$ generated by Algorithm 1 hold the following estimate

$$\begin{aligned}\min_{k\in\{0,\cdots,K\}} \mathbb{E}\left[\operatorname{dist}(\mathbf{0}, T(\boldsymbol{\lambda}^k))^2\right] &\leq \frac{\left\|\bar{\boldsymbol{\lambda}}^0 - \boldsymbol{\lambda}^\star\right\|^2 + \mathcal{A}\left\|\boldsymbol{h}^{-1} - \bar{\boldsymbol{\lambda}}^{-1} + \gamma F(\bar{\boldsymbol{\lambda}}^{-1})\right\|^2}{\gamma^2\bar{\mu}\sum_{\ell=0}^K \alpha_\ell} \\ &+ \frac{\sum_{k=0}^K \left\{\alpha_k^2 Q^{\sigma^2}/_{B_k} + 2\bar{\alpha}\alpha_k\gamma^4 L_F^2 \mathcal{A}^{\sigma^2}/_{B_k}\right\}}{\gamma^2\bar{\mu}\sum_{\ell=0}^K \alpha_\ell}.\end{aligned} \tag{16}$$

with $Q := \gamma^2\mathcal{A} + \gamma^2\alpha^{\max}(\mathcal{A}/3 + \gamma^2)$.

Proof: Define the following operators:

$$\begin{aligned} H(\boldsymbol{x}) &:= \boldsymbol{x} + \gamma F(\boldsymbol{x}), \\ \bar{H}(\boldsymbol{x}) &:= \boldsymbol{x} + \gamma\hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}), \quad \boldsymbol{\xi} \sim \mathcal{D}^\xi(\boldsymbol{\kappa}(\boldsymbol{x})), \end{aligned} \tag{17}$$

and the filtration:

$$\mathcal{F}_k := \operatorname{filtration}\{\mathbb{H}_{\boldsymbol{\xi}}^0, \bar{\mathbb{H}}_{\boldsymbol{\xi}}^0, \cdots, \mathbb{H}_{\boldsymbol{\xi}}^{k-1}, \bar{\mathbb{H}}_{\boldsymbol{\xi}}^{k-1}\},$$

where includes all the randomness involved up to iteration k. Using the notations of Algorithm 1 in (17), $\bar{H}(\boldsymbol{\lambda}^k) = \boldsymbol{\lambda}^k + \gamma\hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}^k)$ and $\bar{H}(\bar{\boldsymbol{\lambda}}^k) = \bar{\boldsymbol{\lambda}}^k + \gamma\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k)$. Take the Lyapunov function [51]:

$$\begin{aligned}\mathcal{U}_{k+1} := &\left\|\bar{\boldsymbol{\lambda}}^{k+1} - \boldsymbol{\lambda}^\star\right\|^2 \\ &+ \mathcal{A}_{k+1}\left\|\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k)\right\|^2 + \mathcal{B}_{k+1}\left\|\bar{\boldsymbol{\lambda}}^{k+1} - \bar{\boldsymbol{\lambda}}^k\right\|^2,\end{aligned} \tag{18}$$

with some $\mathcal{A}_{k+1} > 0$ and $\mathcal{B}_{k+1} > 0$ that will be defined in the sequel. Expanding the first term results in,

$$\begin{aligned}&\left\|\bar{\boldsymbol{\lambda}}^{k+1} - \boldsymbol{\lambda}^\star\right\|^2 \\ &= \left\|\bar{\boldsymbol{\lambda}}^k - \boldsymbol{\lambda}^\star\right\|^2 \\ &\quad - 2\alpha_k\langle \boldsymbol{h}^k - \bar{H}(\boldsymbol{\lambda}^k), \bar{\boldsymbol{\lambda}}^k - \boldsymbol{\lambda}^\star\rangle + \alpha_k^2\left\|\boldsymbol{h}^k - \bar{H}(\boldsymbol{\lambda}^k)\right\|^2,\end{aligned} \tag{19}$$

due to step 8 in Algorithm 1. We continue by upperbounding the terms in (18) and (19). The term $\left\|\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k)\right\|^2$ is upperbounded as follows:

$$\begin{aligned}&\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k) \\ &= \gamma\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k) - \gamma F(\bar{\boldsymbol{\lambda}}^k) + (1-\alpha_k)\left(\boldsymbol{h}^{k-1} - \bar{\boldsymbol{\lambda}}^{k-1} - \gamma\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k)\right), \\ &\left\|\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k)\right\|^2 \\ &= (1-\alpha_k)^2\left\|\boldsymbol{h}^{k-1} - \bar{\boldsymbol{\lambda}}^{k-1} - \gamma F(\bar{\boldsymbol{\lambda}}^k)\right\|^2 \\ &\quad + \left\|\alpha_k\left(\gamma F(\bar{\boldsymbol{\lambda}}^k) - \gamma\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k)\right)\right\|^2 \\ &\quad + 2(1-\alpha_k)\langle \boldsymbol{h}^{k-1} - \bar{\boldsymbol{\lambda}}^{k-1} - \gamma F(\bar{\boldsymbol{\lambda}}^k), \\ &\quad \alpha_k(\gamma\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k) - \gamma F(\bar{\boldsymbol{\lambda}}^k))\rangle,\end{aligned} \tag{20}$$

where the first equality is due to the definition of H in (17) and $\boldsymbol{h}^k$ in Algorithm 1. The second equality is also derived by adding and subtracting $(1-\alpha_k)\gamma F(\bar{\boldsymbol{\lambda}}^k)$. By taking the expectation conditioned on $\mathcal{F}_k$, the first term in the vector inner product is deterministic, while the second term vanishes due to unbiasedness of $\hat{\bar{F}}$ in (6) and (8). Hence, the last term is zero in expectation. Hence,

$$\begin{aligned}&\mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k)\right\|^2 | \mathcal{F}_k\right] \\ &= (1-\alpha_k)^2\left\|\boldsymbol{h}^{k-1} - \bar{\boldsymbol{\lambda}}^{k-1} - \gamma F(\bar{\boldsymbol{\lambda}}^k)\right\|^2 \\ &\quad + \alpha_k^2\gamma^2\mathbb{E}\left[\left\|\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k) - F(\bar{\boldsymbol{\lambda}}^k)\right\|^2 | \mathcal{F}_k\right] \\ &\leq (1+e_k)(1-\alpha_k)^2\left\|\boldsymbol{h}^{k-1} - \bar{\boldsymbol{\lambda}}^{k-1} - \gamma F(\bar{\boldsymbol{\lambda}}^{k-1})\right\|^2 \\ &\quad + (1+{}^1/_{e_k})(1-\alpha_k)^2\left\|\gamma F(\bar{\boldsymbol{\lambda}}^k) - \gamma F(\bar{\boldsymbol{\lambda}}^{k-1})\right\|^2 + \gamma^2\alpha_k^2\sigma_k^2 \\ &\leq U_1\left\|\boldsymbol{h}^{k-1} - \bar{\boldsymbol{\lambda}}^{k-1} - \gamma F(\bar{\boldsymbol{\lambda}}^{k-1})\right\|^2\end{aligned}$$

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

$$+ U_2 \left\|\bar{\boldsymbol{\lambda}}^k - \bar{\boldsymbol{\lambda}}^{k-1}\right\|^2 + U_3 \tag{21}$$

with

$$U_1^k := (1+e_k)(1-\alpha_k)^2, \quad U_2^k := (1+{}^1\!/_{e_k})(1-\alpha_k)^2 L_F^2 \gamma^2,$$
$$U_3^k := \gamma^2 \alpha_k^2 \sigma_k^2, \text{ and } \sigma_k^2 := {}^{\sigma^2}\!/_{B_k},$$

where the first inequality is derived by adding and subtracting $\gamma F(\bar{\boldsymbol{\lambda}}^{k-1})$ along with the Young's inequality with a sequence of $e_k > 0$. In addition, Assumption 4(iii) is invoked for a minibatch of size B_k to have

$$\mathbb{E}\left[\left\|\hat{\bar{F}}(\bar{\mathbb{H}}_{\boldsymbol{\xi}}^k) - F(\bar{\boldsymbol{\lambda}}^k)\right\|^2 |\mathcal{F}_k\right]$$
$$= \mathbb{E}\left[\left\|\frac{1}{B_k}\sum\nolimits_{j=1}^{B_k} \hat{F}(\boldsymbol{H}_{\xi}^j) - F(\bar{\boldsymbol{\lambda}}^k)\right\|^2 |\mathcal{F}_k\right] \leq {}^{\sigma^2}\!/_{B_k}. \tag{22}$$

To bound the last term in (19),

$$\alpha_k^2 \mathbb{E}\left[\left\|\boldsymbol{h}^k - \bar{H}(\boldsymbol{\lambda}^k)\right\|^2 |\mathcal{F}_k\right]$$
$$= \alpha_k^2 \left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 + \alpha_k^2 \gamma^2 \mathbb{E}\left[\left\|F(\boldsymbol{\lambda}^k) - \hat{\bar{F}}(\mathbb{H}_{\boldsymbol{\xi}}^k)\right\|^2 |\mathcal{F}_k\right]$$
$$\leq \alpha_k^2 \left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 + \alpha_k^2 \gamma^2 \sigma_k^2, \tag{23}$$

where the first equality is by adding and subtracting $\gamma F(\boldsymbol{\lambda}^k)$ and the fact that $\hat{\bar{F}}$ is unbiased due to (6) and (8). The last inequality is also due to Assumption 4(iii).

Similarly, the last term in (18) can be bounded by

$$\mathbb{E}\left[\left\|\bar{\boldsymbol{\lambda}}^{k+1} - \bar{\boldsymbol{\lambda}}^k\right\|^2 |\mathcal{F}_k\right] = \alpha_k^2 \mathbb{E}\left[\left\|\boldsymbol{h}^k - \bar{H}(\boldsymbol{\lambda}^k)\right\|^2 |\mathcal{F}_k\right]$$
$$\leq \alpha_k^2 \left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 + \alpha_k^2 \gamma^2 \sigma_k^2, \tag{24}$$

due to step 8 of Algorithm 1 and (23).

To bound the second term in the rhs of (19), we refer to the following lemma:

Lemma 2 (Bounding of $-2\alpha_k \langle \boldsymbol{h}^k - \bar{H}(\boldsymbol{\lambda}^k), \bar{\boldsymbol{\lambda}}^k - \boldsymbol{\lambda}^\star\rangle$ [51 eq. (E.7) and eq. (E.9)]): Take assumptions Assumption 4(i) and 4(ii). Then the following bound holds:

$$-2\alpha_k \mathbb{E}\left[\langle \boldsymbol{h}^k - \bar{H}(\boldsymbol{\lambda}^k), \bar{\boldsymbol{\lambda}}^k - \boldsymbol{\lambda}^\star\rangle |\mathcal{F}_k\right]$$
$$\leq \alpha_k \left(\epsilon_1 + \frac{1}{\epsilon_2}\Delta\right) \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k)\right\|^2 |\mathcal{F}_k\right]$$
$$- \left(\frac{\alpha_k}{1+\epsilon_2}\Delta + 2\alpha_k \frac{\rho}{\gamma}\right) \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 |\mathcal{F}_k\right],$$

with $\Delta := 1 - \frac{1}{\epsilon_1(1-\gamma L_F)^2} \geq 0$, and some positive ϵ_1 and ϵ_2.

Define the following

$$\mu_k := 2\frac{\rho}{\gamma} + \frac{\Delta}{1+\epsilon_2} - \alpha_k(1+\mathcal{B}_{k+1})$$
$$X_1^k := \mathcal{A}_{k+1} + \alpha_k\left(\epsilon_1 + \frac{\Delta}{\epsilon_2}\right)$$
$$X_2^k := \alpha_k^2 \gamma^2 \sigma_k^2 (1+\mathcal{B}_{k+1}). \tag{25}$$

Putting bounds derived in (23), (24), and Lemma 2 into the Lyapunov function (18) results in

$$\mathbb{E}[\mathcal{U}_{k+1}|\mathcal{F}_k] \leq \left\|\bar{\boldsymbol{\lambda}}^k - \boldsymbol{\lambda}^\star\right\|^2 - \alpha_k \mu_k \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 |\mathcal{F}_k\right]$$
$$+ X_1^k \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k)\right\|^2 |\mathcal{F}_k\right] + X_2^k. \tag{26}$$

Considering the bound for $\left\|\boldsymbol{h}^k - H(\bar{\boldsymbol{\lambda}}^k)\right\|^2$ derived in (21), the following is concluded:

$$\mathbb{E}\mathcal{U}_{k+1}[\mathcal{F}_k] - \mathcal{U}_k \leq -\alpha_k \mu_k \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 |\mathcal{F}_k\right]$$
$$+ \left(U_1^k X_1^k - \mathcal{A}_k\right)\left\|\boldsymbol{h}^{k-1} - H(\bar{\boldsymbol{\lambda}}^{k-1})\right\|^2$$
$$+ \left(U_2^k X_1^k - \mathcal{B}_k\right)\left\|\bar{\boldsymbol{\lambda}}^k - \bar{\boldsymbol{\lambda}}^{k-1}\right\|^2$$
$$+ U_3^k X_1^k + X_2^k, \tag{27}$$

with U_1^k, U_2^k and U_3^k defined in (21). To guarantee descent in (27), we need to satisfy the bounds $\epsilon_1 > 0, \epsilon_2 > 0$, and $\Delta \geq 0$ defined in Lemma 2, along with $\mu_k > 0$, $U_1^k X_1^k - \mathcal{A}_k \leq 0$, and $U_2^k X_1^k - \mathcal{B}_k \leq 0$. Hence, with $\alpha^{\max} := \max_k\{\alpha_k\}$, the parameters are set as follows:

- $\Delta \geq 0 \implies \epsilon_1 \geq {}^1\!/_{(1-\gamma L_F)^2}$, set: $\epsilon_1 = {}^1\!/_{\sqrt{\alpha^{\max}}(1-\gamma L_F)^2}$ $\implies \Delta = 1 - \sqrt{\alpha^{\max}}$
- $U_1^k X_1^k - \mathcal{A}_k \leq 0 \longrightarrow$ take: $\mathcal{A}_k = \mathcal{A} \longrightarrow$
$$(1-\alpha_k)^2(1+e_k)\mathcal{A} + (1-\alpha_k)^2(1+e_k)\alpha_k\left(\epsilon_1 + \frac{\Delta}{\epsilon_2}\right) - \mathcal{A}$$
$$\leq -\alpha_k \mathcal{A} + e_k(1-\alpha_k)\mathcal{A} + (1-\alpha_k)^2(1+e_k)\alpha_k\left(\epsilon_1 + \frac{\Delta}{\epsilon_2}\right)$$
$$\leq -\alpha_k(1 - {}^{e_k}\!/_{\alpha_k})\mathcal{A} + (1+e_k)\alpha_k\left(\epsilon_1 + \frac{\Delta}{\epsilon_2}\right) \leq 0$$
$$\implies \mathcal{A} \geq \frac{(1+\nu)(\epsilon_1 + {}^{\Delta}\!/_{\epsilon_2})}{1-\nu}, \text{ with } \nu := {}^{e_k}\!/_{\alpha_k} \in (0,1),$$
set: $\epsilon_2 = \sqrt{\alpha^{\max}} \implies$
$$\mathcal{A} = \frac{1+\nu}{1-\nu}\left(\frac{1}{\sqrt{\alpha^{\max}}(1-\gamma L_F)^2} + \frac{1-\sqrt{\alpha^{\max}}}{\sqrt{\alpha^{\max}}}\right)$$
- $U_2^k X_1^k - \mathcal{B}_k \leq 0 \implies$ take: $\mathcal{B}_k = \frac{U_2^k}{U_1^k}\mathcal{A} = \frac{1}{e_k}\gamma^2 L_F^2 \mathcal{A}$
- $\mu_k > 0 \longrightarrow \mu_k = 2\frac{\rho}{\gamma} + \frac{\Delta}{1+\epsilon_2} - \alpha_k(1+\mathcal{B}_{k+1})$
$$= 2\frac{\rho}{\gamma} + \frac{\Delta}{1+\epsilon_2} - \alpha_k - \frac{\alpha_k}{e_{k+1}}\gamma^2 L_F^2 \mathcal{A}$$
$$\geq 2\frac{\rho}{\gamma} + \frac{1-\sqrt{\alpha^{\max}}}{1+\sqrt{\alpha^{\max}}} - \alpha^{\max} - \frac{\bar{\alpha}}{\nu}\gamma^2 L_F^2 \mathcal{A} =: \bar{\mu} \geq 0$$
with $\bar{\alpha} := \max_k\{{}^{\alpha_k}\!/_{\alpha_{k+1}}\}$, (28)

where in the first bullet, ϵ_1 is chosen due to $\alpha^{\max} \in (0,1)$. In the second bullet, the first inequality is due to $(1-\alpha_k)^2 < 1-\alpha_k$, and the second inequality is due to $(1-\alpha_k)^2 < 1$, and $(1-\alpha_k) < 1$. The third bullet uses the relation of $U_2^k X_1^k - \mathcal{B}_k$ with $U_1^k X_1^k - \mathcal{A}_k$. In the last bullet, $\alpha^{\max} \geq \alpha_k$ is considered

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

due to the diminishing stepsize assumption. The last term in (27) can also be upperbounded as

$$U_3^k X_1^k + X_2^k \leq \alpha_k^2 \sigma_k^2 Q + \bar{\alpha}\alpha_k \sigma_k^2 \gamma^{4L_F^2 \mathcal{A}/\nu},$$

where $Q := \gamma^2 \mathcal{A} + \gamma^2 \alpha^{\max}(\epsilon_1 + {}^{\Delta}/_{\epsilon_2}) + \gamma^2$, $\alpha^{\max} \geq \alpha_k$ and the definition of ν is considered. Putting the above inequalities back into (27) results in

$$\mathbb{E}[\mathcal{U}_{k+1}|\mathcal{F}_k] - \mathcal{U}_k \leq -\alpha_k \mu_k \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 |\mathcal{F}_k\right] + \alpha_k^2 \sigma_k^2 Q + \bar{\alpha}\alpha_k \sigma_k^2 \gamma^{4L_F^2 \mathcal{A}/\nu}. \quad (29)$$

Taking the total expectation, rearranging, and telescoping results in

$$\bar{\mu} \sum_{k=0}^{K} \alpha_k \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2\right] \leq \mathcal{U}_0 - \mathbb{E}[\mathcal{U}_{K+1}] + \textstyle\sum_{k=0}^{K}\{\alpha_k^2 \sigma_k^2 Q + \bar{\alpha}\alpha_k \sigma_k^2 \gamma^{4L_F^2 \mathcal{A}/\nu}\}$$

where μ_k is lowerbounded by $\bar{\mu}$, defined in (28). Omitting $\mathbb{E}[\mathcal{U}_{K+1}] > 0$ from rhs and dividing both sides by $\bar{\mu}\sum_{k=0}^{K}\alpha_k$ gives

$$\sum_{k=0}^{K} \frac{\alpha_k}{\sum_{\ell=0}^{K}\alpha_\ell} \mathbb{E}\left[\left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2\right] \leq \frac{\left\|\bar{\boldsymbol{\lambda}}^0 - \boldsymbol{\lambda}^\star\right\|^2 + \mathcal{A}\left\|\boldsymbol{h}^{-1} - \bar{\boldsymbol{\lambda}}^{-1} + \gamma F(\bar{\boldsymbol{\lambda}}^{-1})\right\|^2}{\bar{\mu}\sum_{\ell=0}^{K}\alpha_\ell} + \frac{\sum_{k=0}^{K}\left\{\alpha_k^2\sigma_k^2 Q + \bar{\alpha}\alpha_k\sigma_k^2\gamma^{4L_F^2\mathcal{A}/\nu}\right\}}{\bar{\mu}\sum_{\ell=0}^{K}\alpha_\ell}, \quad (30)$$

where initializations $\bar{\boldsymbol{\lambda}}^{-1} = \bar{\boldsymbol{\lambda}}^0$ are considered in $\mathcal{U}_0$. Moreover,

$$\left\|\boldsymbol{h}^k - H(\boldsymbol{\lambda}^k)\right\|^2 = \left\|\gamma T(\boldsymbol{\lambda}^k)\right\|^2 \geq \operatorname{dist}(\mathbf{0}, \gamma T(\boldsymbol{\lambda}^k))^2,$$

where $\boldsymbol{h}^k \in \boldsymbol{\lambda}^k + \gamma A(\boldsymbol{\lambda}^k)$ is used due to step 4 of Algorithm 1, as the step imposes the update $\boldsymbol{\lambda}^k = (\mathrm{id} + \gamma A)^{-1}\boldsymbol{h}^k = \max\{\mathbf{0}, \boldsymbol{h}^k\}$, where id is the identity function and $(\mathrm{id}+\gamma A)^{-1}$ is the inverse of the mapping $\mathrm{id} + \gamma A$ which solves $\boldsymbol{\lambda}^k = \arg\min_{\boldsymbol{u}}\left\{\varphi(\boldsymbol{u}) + {}^1/_{2\gamma}\left\|\boldsymbol{u} - \boldsymbol{h}^k\right\|^2\right\}$ with φ defined in (14). The lhs of (30) can also be lowerbounded by the $\min\{\cdot\}$ operator as it is a weighted sum. Hence,

$$\min_{k\in\{0,\cdots,K\}} \mathbb{E}\left[\operatorname{dist}(\mathbf{0}, T(\boldsymbol{\lambda}^k))^2\right] \leq \frac{\left\|\bar{\boldsymbol{\lambda}}^0 - \boldsymbol{\lambda}^\star\right\|^2 + \mathcal{A}\left\|\boldsymbol{h}^{-1} - \bar{\boldsymbol{\lambda}}^{-1} + \gamma F(\bar{\boldsymbol{\lambda}}^{-1})\right\|^2}{\gamma^2\bar{\mu}\sum_{\ell=0}^{K}\alpha_\ell} + \frac{\sum_{k=0}^{K}\left\{\alpha_k^2\sigma_k^2 Q + \bar{\alpha}\alpha_k\sigma_k^2\gamma^{4L_F^2\mathcal{A}/\nu}\right\}}{\gamma^2\bar{\mu}\sum_{\ell=0}^{K}\alpha_\ell}.$$

Setting $\nu = 0.5$, and using (22) completes the proof. ■

Following the extragradient-like updates as in Algorithm 1, the same estimate for $\mathbb{E}\left[\operatorname{dist}(\mathbf{0}, T(\boldsymbol{\lambda}^k))^2\right]$—indicating $\mathbf{0} \in T(\boldsymbol{\lambda}^k)$ in (14) for the class of nonmonotone mappings—has been presented in [51], except for the last term on the rhs of (16) with α_k in the numerator rather than α_k^2, indicating that an increasing batch size is necessary. This difference is due to the independence of the stochastic oracle $\hat{F}$ from the decision variables $\boldsymbol{\lambda}$ and $\bar{\boldsymbol{\lambda}}$. Moreover, the condition type on $\bar{\mu}$ in Theorem 2 can also be found in [51], and it can be controlled by adjusting $\alpha^{\max}$ as well as L_F. It should be mentioned that the Lipschitz constant L_F can be chosen arbitrarily small (see Lemma 1).

The estimate in (16) indicates the possibility of achieving a diminishing rhs, and consequently, convergence to a fixed point, by selecting suitable stepsizes α_k and batch sizes B_k. The following remark highlights the convergence rate achieved through the standard choice of these parameters [56]:

Remark 2: Consider either of the following two scenarios, with appropriate stepsize α_k and Lipschitz constant L_F (see Lemma 1) that satisfies $\bar{\mu} > 0$:

- Take fixed stepsize $\alpha_k = \alpha$ and fixed batch size $B_k = \sqrt{K}$;
- Take diminishing stepsize $\alpha_k = \frac{\alpha_0}{\sqrt{1+\tilde{\alpha}k}}$ and the increasing batch size $B_k = 1 + \sqrt{k}$, with some positive α_0 and $\tilde{\alpha}$.

Then, the convergence rate is

$$\min_{k\in\{0,\cdots,K\}} \mathbb{E}\left[\operatorname{dist}(0, T(\boldsymbol{\lambda}^k))^2\right] \leq \mathcal{O}({}^1/\sqrt{K}).$$

□

REFERENCES

[1] V. P. Mhatre, K. Papagiannaki, and F. Baccelli, "Interference mitigation through power control in high density 802.11 WLANs," in *Proc. 26th IEEE Int. Conf. Comput. Commun.*, May 2007, pp. 535–543.

[2] R. Cendrillon, J. Huang, M. Chiang, and M. Moonen, "Autonomous spectrum balancing for digital subscriber lines," *IEEE Trans. Signal Process.*, vol. 55, no. 8, pp. 4241–4257, Aug. 2007.

[3] M. Chiang, C. W. Tan, D. P. Palomar, D. O'Neill, and D. Julian, "Power control by geometric programming," *IEEE Trans. Wireless Commun.*, vol. 6, no. 7, pp. 2640–2651, Jul. 2007.

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

[5] F. Zhou, G. Lu, M. Wen, Y.-C. Liang, Z. Chu, and Y. Wang, "Dynamic spectrum management via machine learning: State of the art, taxonomy, challenges, and open research issues," *IEEE Netw.*, vol. 33, no. 4, pp. 54–62, Jul. 2019.

[6] W. Tong and G. Y. Li, "Nine challenges in artificial intelligence and wireless communications for 6G," *IEEE Wireless Commun.*, vol. 29, no. 4, pp. 140–145, Aug. 2022.

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

[8] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards optimal power control via ensembling deep neural networks," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1760–1776, Mar. 2020.

[9] W. Lee, "Resource allocation for multi-channel underlay cognitive radio network based on deep neural network," *IEEE Commun. Lett.*, vol. 22, no. 9, pp. 1942–1945, Sep. 2018.

[10] L. Liang, H. Ye, G. Yu, and G. Y. Li, "Deep-learning-based wireless resource allocation with application to vehicular networks," *Proc. IEEE*, vol. 108, no. 2, pp. 341–356, Feb. 2020.

[11] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1248–1261, Jun. 2019.

[12] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

[13] H. Ye, G. Y. Li, and B. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3163–3173, Apr. 2019.

[14] C. Guo, Z. Li, L. Liang, and G. Y. Li, "Reinforcement learning-based power control for reliable mission-critical wireless transmission," *IEEE Internet Things J.*, vol. 10, no. 23, pp. 20868–20883, Dec. 2023.

[15] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, Oct. 2019.

[16] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 12, Nov. 1999, pp. 1057–1063.

[17] M. Eisen, C. Zhang, L. F. O. Chamon, D. D. Lee, and A. Ribeiro, "Learning optimal resource allocations in wireless systems," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2775–2790, May 2019.

[18] D. S. Kalogerias, M. Eisen, G. J. Pappas, and A. Ribeiro, "Model-free learning of optimal ergodic policies in wireless systems," *IEEE Trans. Signal Process.*, vol. 68, pp. 6272–6286, 2020.

[19] Z. Wang, M. Eisen, and A. Ribeiro, "Learning decentralized wireless resource allocations with graph neural networks," *IEEE Trans. Signal Process.*, vol. 70, pp. 1850–1863, 2022.

[20] Z. Ji, Z. Qin, and X. Tao, "Meta federated reinforcement learning for distributed resource allocation," *IEEE Trans. Wireless Commun.*, vol. 23, no. 7, pp. 7865–7876, Jul. 2024.

[21] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[22] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 101–115, Jan. 2021.

[23] N. NaderiAlizadeh, M. Eisen, and A. Ribeiro, "Learning resilient radio resource management policies with graph neural networks," *IEEE Trans. Signal Process.*, vol. 71, pp. 995–1009, 2023.

[24] M. Eisen and A. Ribeiro, "Optimal wireless resource allocation with random edge graph neural networks," *IEEE Trans. Signal Process.*, vol. 68, pp. 2977–2991, 2020.

[25] N. NaderiAlizadeh, M. Eisen, and A. Ribeiro, "State-augmented learnable algorithms for resource management in wireless networks," *IEEE Trans. Signal Process.*, vol. 70, pp. 5898–5912, 2022.

[26] K. Huang, L. Liang, X. Yi, H. Ye, S. Jin, and G. Y. Li, "Meta-learning empowered graph neural networks for radio resource management," *IEEE Trans. Commun.*, early access, Mar. 17, 2025, doi: 10.1109/TCOMM.2025.3552302.

[27] H. T. Nguyen, M. T. Nguyen, H. T. Do, H. T. Hua, and C. V. Nguyen, "DRL-based intelligent resource allocation for diverse QoS in 5G and toward 6G vehicular networks: A comprehensive survey," *Wireless Commun. Mobile Comput.*, vol. 2021, no. 1, Jan. 2021, Art. no. 5051328.

[28] J. Tian, Q. Liu, H. Zhang, and D. Wu, "Multiagent deep-reinforcement-learning-based resource allocation for heterogeneous QoS guarantees for vehicular networks," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1683–1695, Feb. 2022.

[29] P. Behmandpoor, M. Moonen, and P. Patrinos, "Asynchronous message-passing and zeroth-order optimization based distributed learning with a use-case in resource allocation in communication networks," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 10, pp. 916–931, 2024.

[30] P. Behmandpoor, P. Patrinos, and M. Moonen, "Model-free decentralized training for deep learning based resource allocation in communication networks," in *Proc. 31st Eur. Signal Process. Conf. (EUSIPCO)*, Sep. 2023, pp. 1494–1498.

[31] P. Behmandpoor, P. Patrinos, and M. Moonen, "Federated learning based resource allocation for wireless communication networks," in *Proc. IEEE 30th Eur. Signal Process. Conf. (EUSIPCO)*, Nov. 2022, pp. 1656–1660.

[32] P. Behmandpoor, J. Verdyck, and M. Moonen, "Deep learning-based cross-layer resource allocation for wired communication systems," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 4120–4124.

[33] Y. Peng, L. Liu, Y. Zhou, J. Shi, and J. Li, "Deep reinforcement learning-based dynamic service migration in vehicular networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

[34] W. Wu et al., "Dynamic RAN slicing for service-oriented vehicular networks via constrained learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2076–2089, Jul. 2021.

[35] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for radio resource allocation with diverse quality-of-service requirements in 5G," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2309–2324, Apr. 2021.

[36] Y. Yuan, G. Zheng, K.-K. Wong, B. Ottersten, and Z.-Q. Luo, "Transfer learning and meta learning-based fast downlink beamforming adaptation," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1742–1755, Mar. 2021.

[37] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2009.

[38] Y. Yuan, L. Lei, T. X. Vu, Z. Chang, S. Chatzinotas, and S. Sun, "Adapting to dynamic LEO-B5G systems: Meta-critic learning based efficient resource scheduling," *IEEE Trans. Wireless Commun.*, vol. 21, no. 11, pp. 9582–9595, Nov. 2022.

[39] G. Qu, H. Wu, R. Li, and P. Jiao, "DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 3, pp. 3448–3459, Sep. 2021.

[40] J. Wang, J. Hu, G. Min, A. Y. Zomaya, and N. Georgalas, "Fast adaptive task offloading in edge computing based on meta reinforcement learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 1, pp. 242–253, Jan. 2021.

[41] L. Huang, L. Zhang, S. Yang, L. P. Qian, and Y. Wu, "Meta-learning based dynamic computation task offloading for mobile edge computing networks," *IEEE Commun. Lett.*, vol. 25, no. 5, pp. 1568–1572, May 2021.

[42] Y. Yuan, G. Zheng, K.-K. Wong, and K. B. Letaief, "Meta-reinforcement learning based resource allocation for dynamic V2X communications," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 8964–8977, Sep. 2021.

[43] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, Aug. 2017, pp. 1126–1135.

[44] A. Vettoruzzo, M.-R. Bouguelia, J. Vanschoren, T. Rögnvaldsson, and K. Santosh, "Advances and challenges in meta-learning: A technical review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 7, pp. 4763–4779, Jul. 2024.

[45] A. Nagabandi et al., "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," 2018, *arXiv:1803.11347*.

[46] K. Li and J. Malik, "Learning to optimize," 2016, *arXiv:1606.01885*.

[47] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5149–5169, Sep. 2022.

[48] P. Behmandpoor, P. Patrinos, and M. Moonen, "Learning-based resource allocation with dynamic data rate constraints," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 4088–4092.

[49] F. Edition, A. Papoulis, and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*. New York, NY, USA: McGraw-Hill, 2002.

[50] Y.-G. Hsieh, F. Iutzeler, J. Malick, and P. Mertikopoulos, "Explore aggressively, update conservatively: Stochastic extragradient methods with variable stepsize scaling," in *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 16223–16234.

[51] T. Pethick, O. Fercoq, P. Latafat, P. Patrinos, and V. Cevher, "Solving stochastic weak minty variational inequalities without increasing batch size," 2023, *arXiv:2302.09029*.

[52] J. Diakonikolas, C. Daskalakis, and M. I. Jordan, "Efficient methods for structured nonconvex-nonconcave min-max optimization," in *Proc. Int. Conf. Artif. Intell. Statist.*, Jan. 2020, pp. 2746–2754.

[53] W. Cui and W. Yu, "Uncertainty injection: A deep learning method for robust optimization," *IEEE Trans. Wireless Commun.*, vol. 22, no. 11, pp. 7201–7213, Nov. 2023.

[54] L. P. Qian, Y. J. Zhang, and J. Huang, "MAPEL: Achieving global optimality for a non-convex wireless power control problem," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, pp. 1553–1563, Mar. 2009.

[55] D. Drusvyatskiy and L. Xiao, "Stochastic optimization with decision-dependent distributions," *Math. Oper. Res.*, vol. 48, no. 2, pp. 954–998, May 2023.

[56] S. Ghadimi, G. Lan, and H. Zhang, "Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization," *Math. Program.*, vol. 155, nos. 1–2, pp. 267–305, Jan. 2016.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.

Pourya Behmandpoor (Graduate Student Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from Iran University of Science and Technology (IUST), Tehran, Iran, and the Ph.D. degree in electrical engineering from KU Leuven, Belgium. He recently joined Vrije Universiteit Brussel (VUB), Belgium, as a Postdoctoral Researcher. His research interests include learned compression, distributed machine learning, large-scale optimization, and signal processing, with applications in federated learning and cooperative perception in multi-agent systems, sensor networks, and wireless communication networks.

Mark Eisen (Member, IEEE) received the master's degree in statistics and the Ph.D. degree in electrical engineering from the University of Pennsylvania in 2019. He is currently working as a Staff Scientist at the Applied Physics Laboratory, Johns Hopkins University. He was previously a Research Scientist with Intel Labs, Hillsboro, OR, USA. His research interests include networked control systems, statistical optimization, reinforcement learning, and wireless communications. In 2022, he received the Best Paper Award at the IEEE International Conference on Factory Communication Systems.

Panagiotis (Panos) Patrinos (Member, IEEE) received the M.Eng. degree in chemical engineering, the M.S. degree in applied mathematics, and the Ph.D. degree in control and optimization from the National Technical University of Athens, Athens, Greece, in 2003, 2005, and 2010, respectively. In 2014, he was a Visiting Professor with Stanford University. After his Ph.D., he held post-doctoral positions with the University of Trento and IMT Lucca, Italy, where he became an Assistant Professor in 2012. He is currently an Associate Professor with the Department of Electrical Engineering (ESAT), KU Leuven, Leuven, Belgium. His research interests include the interface of optimization, machine learning, and control. His current research focuses on fundamental theoretical and algorithmic aspects within these three fields, aimed at addressing challenges across diverse engineering disciplines and these include including autonomous systems and robotics, signal processing, energy, and communication systems.

Marc Moonen (Fellow, IEEE) is currently a Full Professor at the Electrical Engineering Department of KU Leuven, where he is heading a research team working in the area of numerical algorithms and signal processing for digital communications, wireless communications, DSL, and audio signal processing.

Dr. Moonen was a Board Member of Directors of European Association for Signal Processing (EURASIP) from 2000 to 2016. He was a member of the IEEE Signal Processing Society Awards Board from 2008 to 2010, a member of the IEEE Signal Processing Society Technical Committee on Signal Processing for Communications from 2005 to 2007 and from 2008 to 2010, and a member of the IEEE Signal Processing Society Fellow Selection Committee from 2014 to 2016. He has been a member of the editorial board of *Signal Processing*, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, *IEEE Signal Processing Magazine*, *Integration-the VLSI Journal*, *EURASIP Journal on Wireless Communications and Networking*, and *EURASIP Journal on Advances in Signal Processing*. He was a fellow of EURASIP in 2018. He was the 1997 Laureate of Belgium Royal Academy of Science. He received the 1994 KU Leuven Research Council Award, the 1997 Alcatel Bell Belgium Award (with Piet Vandaele), the 2004 Alcatel Bell Belgium Award (with Raphael Cendrillon), the CELTIC Excellence Award 2008 for EU/CELTIC Project "BANITS" (with project partners). He received the Journal Best Paper Awards from IEEE TRANSACTIONS ON SIGNAL PROCESSING (with Geert Leus and Daniele Giacobello) and from *Signal Processing* (Elsevier) (with Simon Doclo), and the Conference Best Paper Awards at IEEE ICASSP (with Ann Spriet), IEEE ICC (with R. Moraes, P. Tsiaflakis, and Jeroen Van Dyck), IEEE-IWAENC (with Ann Spriet and Simon Doclo), and at IEEE-WIC Symp (with Geert Van Meerbergen). He was the Chairperson of the EURASIP Fellow Selection Committee from 2019 to 2021. He was the Editor-in-Chief of *EURASIP Journal on Applied Signal Processing* from 2003 to 2005 and an Area Editor of Feature Articles in *IEEE Signal Processing Magazine* from 2012 to 2014. He was the IEEE Signal Processing Society Vice President-Publications from 2021 to 2023 and the Chairperson of the IEEE Benelux Signal Processing Chapter from 1998 to 2002. He was the President of EURASIP from 2007 to 2008 and from 2011 to 2012.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on February 24,2026 at 14:26:01 UTC from IEEE Xplore. Restrictions apply.