Meta-Critic Reinforcement Learning for Intelligent Omnidirectional Surface Assisted Multi-User Communications

Qinpei Luo, *Student Member, IEEE*, Boya Di, *Member, IEEE* and Zhu Han, *Fellow, IEEE*

***Abstract*—With the 5G systems being highly developed, the urge of the next generation networks is increasingly necessary, which demands extremely high data rate and low latency. As an emerging technology capable of reflecting and refracting the incident signals on both sides simultaneously, recently the intelligent omnidirectional surface (IOS) has been used to enhance the capacity of wireless networks. However, it is challenging to design an IOS-enabled beamforming scheme that can respond quickly in a varying mobile environment due to its high complexity. In this paper, we aim to maximize the sum rate in an IOS-aided multi-user system given dynamically changing channel states and user mobility. A novel meta-critic reinforcement learning framework named meta-critic deep deterministic policy gradient algorithm is proposed to design the IOS-enabled beamforming scheme. We propose a meta-critic network that can recognize the environment change and automatically performs the self-renewal of the learning model. A stochastic explore-and-reload procedure is also tailored to reduce the high-dimensional action space problem. Simulation results demonstrate that our proposed method outperforms other benchmarks including the state-of-the-art reinforcement learning method in both achievable sum rate and convergence speed.**

***Index Terms*—Intelligent omni-surface, Beamforming, Meta-learning, Dynamic environments, 6G.**

I. Introduction

Having been rolled out in many countries, the scale of the fifth-generation (5G) communication system has already reached a high standard, urging the academia and industry to shift their attention towards the next generation, i.e., the 6G system [2]. Compared to 5G, stringent requirements have been raised to achieve extremely high data rates even in dynamic environments [3]. To meet such requirements, intelligent metasurfaces have emerged as a promising enabling technology owing to its potential to boost the system performance in terms of spectrum efficiency as well as cell coverage [4] in the 6G networks have emerged as a promising technology. One typical metasurface, intelligent omni-surface (IOS), has recently attracted great attention for its ability of simultaneous signal reflection and refraction towards target directions to serve users on both sides [5] by configuring the phase shifts of its elements.

Despite the capability of IOS to enhance the data rate, the numerous number of IOS elements also brings a heavy computational burden since the phase shifts of all IOS elements need to be optimized [6]. This problem can degrade the system performance such as achievable data rate, especially in a dynamic environment that requires a fast response of IOS phase shifts configuration. Therefore, it is critical to develop an efficient beamforming scheme that is capable of adapting to the varying propagation environment, users' positions, the number of users, etc., for a large number of IOS elements.

In the literature, reinforcement learning have served as mathematical tools to adapt to unpredictable environments. Existing works mainly consider static environments and small-scale RIS elements. In both [7] and [8], the deep reinforcement learning (RL) method was introduced to improve the sum rate and energy efficiency, respectively. Recently, transfer learning [9] and meta learning [10] have emerged as promising methods to reduce the data that need to be collected, thus faster adapting to the varying environment compared to traditional methods. In [11], the authors design a four-layer neural network. By transferring the weights of two layers in the pre-trained model to different target domains divided by the number of RIS elements, their method minimizes transmit power with fast convergence. The work [12] introduces the Expectation Maximization based meta-learning method and shows a trade-off between its performance and efficiency.

However, most existing RL-related works [7], [8], [13], [14] consider a simplified setting where the channel state information (CSI) and locations of users are static across time. Once the environment changes, the RL model needs to be retrained from scratch to update the IOS beamformer, otherwise the solution is out-of-date and the performance is degraded. Though there are some initial transfer learning or meta-learning based works [11], [12] that transfer the features learned from one environment to another, they have not considered the time-varying dynamic environment where CSI, location, and the number of users in different time slots are relevant. Consequently, the proposed approaches may not fit well when it comes to practice. Besides, most of the works consider a relatively small scale of intelligent surface elements. The scalability remains unguaranteed when model complexity increases as the number of RIS elements grows.

Unlike the existing works, we aim to develop an efficient beamforming scheme to address the following practical concerns:

- *Q1*: How to adapt to the dynamic case where the channel information, user positions, and the number of users vary with time?
- *Q2*: How to search for a solution efficiently within a complex solution space brought by a large-scale IOS?

This article was presented in part at the IEEE Vehicular Technology Conference (VTC2023Spring), in June 2023 [1].

Qinpei Luo is with School of Electronics Engineering and Computer Science, Peking University, Beijing, China (email: luoqinpei@pku.edu.cn)

Boya Di is with State Key Laboratory of Advanced Optical Communication Systems and Networks, School of Electronics, Peking University, Beijing, China (email: diboya@pku.edu.cn)

Zhu Han is with Electrical and Computer Engineering Department, University of Houston, TX, USA (email: hanzhu22@gmail.com)

It is not trivial to solve the above issues. *First*, the high dimension of the phase shift optimization problem makes the traditional learning methods infeasible as the continuous action space is too large. Therefore, it is necessary to develop a beamforming scheme that is robust against the size of IOS with respect to the sum rate of the system. *Second*, for a dynamic environment, the CSI, locations and numbers of users can vary with time. It is thus necessary to consider the dynamic characteristics of the environment and design a beamforming approach that converges fast with fewer data to obtain an up-to-date IOS configuration.

In this paper, we design a novel IOS beamforming scheme based on the idea of meta learning [10], which mainly focuses on the development of algorithms that can automatically learn how to learn, i.e., extracting the common patterns or correlations across the tasks. Our main idea is to develop the actor-critic architecture for reinforcement learning by replacing the conventional critic with a novel meta-critic that can extract common features of all tasks from different environments. In summary, we contribute to state-of-the-art research in the following ways by addressing the above challenges.

- We first model the sum rate maximization problem in a dynamic environment as a Markov decision process (MDP) to simultaneously depict the configuration of IOS and time-related channels. Based on this, we propose a meta-critic deep deterministic policy gradient (MC-DDPG) scheme for the IOS-based beamforming given dynamic channel states and moving users. Stemming from the meta learning [15], a novel meta-critic is designed which serves as an automotive tool for fast real-time model parameter generation in new environments by learning from multiple scenario-specific tasks.
- We design an *Explore and Reload* procedure in the training process of our model, in which we set an exploring factor to add randomness in the solution searching process and decay it while training to achieve convergence. Therefore, it makes our proposed method more robust and easier to converge to a solution even when there are numerous IOS elements. Besides, benefiting from the feature extraction ability of the tailored meta-learning network structure, only a small amount of cascaded channel information between the transmitter and users is required for training, thereby significantly saving the pretraining overhead.
- Simulation results show that given a small amount of channel information, our proposed MC-DDPG outperforms the traditional RL method and an iterative algorithm in terms of both the sum rate and the convergence speed in dynamic environments, where CSI, positions and the number of users vary with time. The robustness of the MC-DDPG scheme given different IOS sizes is also verified.

The rest of this paper is organized as follows. Sections II and III present the system model and problem formulation, respectively. In Section IV, we propose our tailored MC-DDPG algorithm to solve the sum rate maximization problem. Simulation results are shown in Section V. Finally, we draw the conclusions in Section VI.

II. System Model

In this section, we first describe the IOS-assisted multi-user communication system and then present the detailed IOS and channel models.

A. Scenario Description

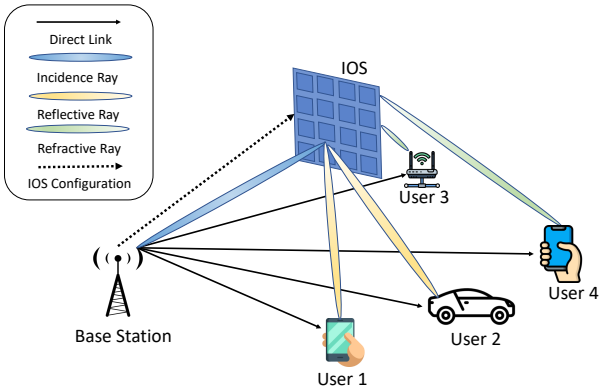

Fig. 1. System model of the IOS-assisted multi-user system

As shown in Fig. 1, we consider a downlink multi-user MISO wireless communication system with a M-antenna base station (BS) serving K users with a single antenna. Because of the shadowing effect and unexpected fading of propagation paths, the Line-of-Sight (LoS) channel between the BS and users is often unstable and suffers from severe fading [16]. To enhance the capacity of the system, an IOS consisting of N elements is deployed to reflect and refract the transmit signals simultaneously towards both sides of the IOS. By configuring the phase shifts of IOS elements, we are able to generate wave beams with heterogeneous directions.

Based on the scenario described above, we consider a dynamic communication environment in that the CSI, location and number of users may vary with time. However, due to the large scale of IOS elements, it might degrade the speed of adapting to the rapidly changing environment if we update the IOS configuration after collecting a large amount of CSI. As a result, we need an approach that can adjust the phase shifts of IOS quickly with only a small amount of CSI fed back from the environment.

B. Intelligent Omni-Surface Model

Different from the reflecting type meta-surface (a.k.a. RIS) [17], IOS can manipulate electromagnetic waves with a dual function of simultaneously reflecting and refracting signals. By controlling the biased voltages of each IOS element, the reflective and refractive waves can be oriented to users. Each element is sub-wavelength and is capable of performing 2^b possible phase shifts to reflect and refract the incident ray [18]. The dual function of IOS can be expressed by [5]

$$\Delta^t = \sqrt{\frac{1}{1+\epsilon}}, \Delta^r = \sqrt{\frac{\epsilon}{1+\epsilon}}, \tag{1}$$

in which ϵ is the refraction-reflection ratio, i.e., the ratio of the refractive signal to the reflective signal. Δ^t and Δ^r are the

energy spilt for the reflected and refracted signals, respectively. The response Γ_n of the n-th IOS element can be given by

$$\Gamma_n = \Delta^n q_n, \tag{2}$$

$$q_n = \sqrt{G_n F_A F_D \delta_w \delta_h |\gamma_n|^2} e^{-j\theta_n}, \tag{3}$$

in which G_n is the antenna gain of the n-th IOS element. γ_n refers to the power ratio of the reflective or refractive signal. F_A and F_D are the normalized power radiation patterns of the incident signal and reflective/refractive signal. δ_w and δ_h denote the width and height of each IOS element, respectively. θ_n represents the phase shifts of IOS, which can be further written as θ_n^t and θ_n^r indicating reflection/refraction phase shift. On top of that, because of limitations of hardware design, the two phase shifts of the same IOS element are coupled with each other

$$\theta_n^t - \theta_n^r = C, \tag{4}$$

where C is a constant determined by the structure of the metasurface.

C. Channel Model

The direct channel between the BS and K users can be denoted as $\mathbf{H}_{BU} \in \mathbb{C}^{K\times M}$. The BS-IOS link and the IOS-user link can be denoted by $\mathbf{H}_{BI} \in \mathbb{C}^{N\times M}$ and $\mathbf{H}_{IU} \in \mathbb{C}^{K\times N}$, respectively. According to the Saleh-Valenzuela model [19], the channel matrices can be expressed by

$$\mathbf{H}_{BI} = \sqrt{S_1}\boldsymbol{A}_I \boldsymbol{\Sigma}_{BI} \boldsymbol{D}_B^H, \tag{5}$$

$$\mathbf{H}_{IU,k} = \sqrt{S_{2,k}}\boldsymbol{A}_{IU,k} \boldsymbol{\Sigma}_{IU,k} \boldsymbol{D}_{I,k}^H, \tag{6}$$

$$\mathbf{H}_{BU,k} = \sqrt{S_{3,k}}\boldsymbol{A}_{BU,k} \boldsymbol{\Sigma}_{BU,k} \boldsymbol{D}_{B,k}^H, \tag{7}$$

where $\boldsymbol{D}_B, \boldsymbol{D}_{IU,k}$ and $\boldsymbol{D}_{BU,k}$ represent the transmit steering matrices, while $\boldsymbol{A}_I, \boldsymbol{A}_{IU,k}$ and $\boldsymbol{A}_{BU,k}$ denote the receive steering matrices. The i-th columns of each $\boldsymbol{D}$ are channel steering vectors, which can be expressed by $\boldsymbol{f}(N,\theta) = \frac{1}{\sqrt{N}}[1, e^{j\pi\theta}, \ldots, e^{j(N-1)\pi\theta}]^H$, where N is the number of antennas and θ is the angle-of-arrival (AoA) or angle-of-departure (AoD). The matrices are set as $\boldsymbol{\Sigma}_{BI} = diag(\sqrt{\frac{N^2 N_b}{I_1}}[\lambda_{BI,1}, \ldots, \lambda_{BI,I_1}])$, $\boldsymbol{\Sigma}_{IU} = diag(\sqrt{\frac{N^2 N_u}{I_2}}[\lambda_{IU,1}, \ldots, \lambda_{IU,I_2}])$ and $\boldsymbol{\Sigma}_{BU} = diag(\sqrt{\frac{N_b N_u}{I_3}}[\lambda_{BU,1}, \ldots, \lambda_{BU,I_3}])$, where I_1, I_2 and I_3 are the numbers of links of each channel. For the i-th link, $\lambda_{BI,i}, \lambda_{IU,i}$ and $\lambda_{BU,i}$ denote the channel gains. We assume that each channel of $\mathbf{H}_{BU}, \mathbf{H}_{BI}, \mathbf{H}_{IU}$ consists two components, the LoS and Non-Line-of-Sight (NLoS) channels, respectively.

For users in the reflective or refractive zone of the IOS, the LoS component of the equivalent channel from the BS to user k can be given as

$$\mathbf{H}_k^{LoS} = \Delta^u \mathbf{H}_{IU,k} \boldsymbol{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}, k \in \mathcal{K}_u, \tag{8}$$

in which $\boldsymbol{\Theta} \in \mathbb{C}^{N\times N} = diag([e^{j\theta_1}, \cdots, e^{j\theta_N}])$, $[e^{j\theta_1}, \cdots, e^{j\theta_N}]$ being the phase shift configuration of IOS elements. $\mathcal{K}_u$ refers to the set of users, and $u \in \{r, t\}$ refers to the reflective and refractive users, respectively.

We assume that the equivalent channel of each user follows the Rician distribution [20] with a factor K^R, i.e.,

$$\mathbf{H}_k = \sqrt{\frac{K^R}{K^R+1}}\mathbf{H}_k^{LOS} + \sqrt{\frac{1}{K^R}}\mathbf{H}_k^{NLOS}. \tag{9}$$

Such a channel model can be further modeled as a finite-state Markov channel [21]. Specifically, we fix the LOS component and discretize the NLoS channel $\boldsymbol{H}^{NLOS}$ into L levels, i.e., $\mathcal{H} = \mathbf{H}_1, \ldots, \mathbf{H}_L$. The AoAs and AoDs of the NLOS channel on each level are random. The transition probability matrix is defined as

$$\mathbf{P} = \begin{bmatrix} p_{1,1} & \cdots & p_{1,L} \\ \vdots & \ddots & \vdots \\ p_{L,1} & \cdots & p_{L,L} \end{bmatrix}, \tag{10}$$

where the transition probability $p_{l,l'}$ can be written as

$$p_{l,l'} = Prob[\mathbf{H}_{t+1} = \mathbf{H}_{l'} | \mathbf{H}_t = \mathbf{H}_l], \mathbf{H}_l, \mathbf{H}_{l'} \in \mathcal{H}. \tag{11}$$

The equation above indicates that given the channel state $\mathbf{H}_t = \mathbf{H}_l$ at time slot t, $p_{l,l'}$ refers to the probability of channel state at the next time slot $\mathbf{H}_{t+1}$ transiting from $\mathbf{H}_l$ to $\mathbf{H}_{l'}$. Without loss of generality, we generate $\mathbf{P}$ randomly to depict the time-varying NLoS channel.

III. Problem Formulation

In this section, we will first formulate the sum rate maximization problem, and then explain why and how we reformulate it into a MDP to develop a reinforcement learning method.

A. Sum Rate Maximization Problem

To better depict the influence brought by dynamic environments, we consider the sum rate maximization problem in T time slots, each of which has a duration of ΔT. The received signal of user k in time slot t can be written as

$$y_{k,t} = (\Delta \mathbf{H}_{IU,k} \boldsymbol{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}) \sum_{j=1}^{K} \boldsymbol{V}_{j,t} x_j + n_{k,t}, \tag{12}$$

where Δ can be Δ^r or Δ^t determined by the type of the user. $\boldsymbol{V}_{j,t}$ refers to the digital beamforming vector from the BS to the j-th user. x_k denotes the symbol BS sends to user j. $n_{k,t}$ represent Gaussian noise which follows $N(0, \sigma_{k,t}^2)$. The Signal to Interference plus Noise Ratio of user k can be expressed as

$$\gamma_{k,t} = \frac{|(\Delta \mathbf{H}_{IU,k} \boldsymbol{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}) \boldsymbol{V}_{k,t} x_k|^2}{|(\Delta \mathbf{H}_{IU,k} \boldsymbol{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}) \sum_{j=1,\neq k}^{K} \boldsymbol{V}_{j,t} x_j|^2 + \sigma_{k,t}^2}, \tag{13}$$

from which we can express user k's data rate as,

$$R_{k,t} = |\Delta T log(1 + \gamma_{k,t})|. \tag{14}$$

The sum rate maximization problem can be formulated as

$$\begin{aligned} \mathbf{P1}: & \max_{\boldsymbol{V}_t, \boldsymbol{\Theta}_t} \sum_{t=1}^{T} \sum_{k=1}^{K} R_{k,t}, \\ & s.t.\ Tr(\boldsymbol{V}_t^H \boldsymbol{V}_t) \leq P_T, t = 1, \cdots, T, \\ & \forall \theta_n \in \boldsymbol{\Theta}_t, \theta_n^t - \theta_n^r = c, \end{aligned} \tag{15}$$

where P_T refers to the total transmit power. Given $\mathbf{\Theta}_t$, through zero-force (ZF) beamforming[1] and water-filling algorithm [24], we can get a sub-optimal solution of $\boldsymbol{V}_t$ directly. Let $\mathbf{H}$ denote the equivalent channel between the BS and all users, then the beamformer can be given by

$$\boldsymbol{V}_D = \mathbf{H}^H(\mathbf{H}\mathbf{H}^H)^{-1}\boldsymbol{P}^{\frac{1}{2}}, \tag{16}$$

where $\boldsymbol{P} = diag([p_1, p_2, \ldots, p_M])$ represents transmit power on each antenna. The optimal power allocation is solved by water-filling [25] as

$$p_k = \frac{1}{\nu_k} max\left\{\frac{1}{\mu} - \nu_k\sigma^2, 0\right\}, \tag{17}$$

where ν_k is the k-th diagonal element of $(\mathbf{H}^H(\mathbf{H}\mathbf{H}^H)^{-1})^H\mathbf{H}^H(\mathbf{H}\mathbf{H}^H)^{-1}$, and μ is set for normalization, which makes p_k satisfy $\sum_{k=1}^{M} p_k = P_T$. In the next subsection, we will show the reason why we reformulate such a problem into a MDP.

B. Motivation of MDP

For such an optimization problem in a time-varying communication environment, it can be reformulated into a MDP for the following three reasons.

1) The communication environment that we consider is dynamic like the channel state and user's number, location, and the number of users, and these characteristics are time-related. Thus we choose to reformulate it into a MDP to better depict how these characteristics vary with time.
2) Secondly, in the practical application of IOS-assisted communication, we are expected to configure the IOS shifts in real-time. Once we get the CSI, we adjust the IOS phase shifts, which can be easily modeled as a MDP.
3) Thirdly, introducing a machine learning method can improve the efficiency of searching for solutions compared to traditional iteration-based approaches. But if we want to adopt a learning method to solve this problem, one unavoidable difficulty is that the labeled dataset is missed. In this case, unsupervised learning which does not require labeled datasets is more suitable. However, traditional unsupervised learning like Principle Component Analysis (PCA) [26] or clustering [27] cannot solve it. That is the main motivation why we try to reformulate $\mathbf{P1}$ into a MDP to apply RL.

Next, we will illustrate the MDP reformulation of the optimization problem in detail.

C. MDP Reformulation

For a fixed digital beamforming scheme such as ZF [24] or MMSE [22], we can rewrite $\mathbf{P1}$ as

$$\begin{aligned} &\mathbf{P2}: \max_{\mathbf{\Theta}_t} \sum_{t=1}^{T}\sum_{k=1}^{K} R_{k,t}. \\ &\forall\theta_n \in \mathbf{\Theta}_t, \theta_n^t - \theta_n^r = c, \end{aligned} \tag{18}$$

[1]Other digital beamforming methods like Minimum Mean Squared Error (MMSE) [22], Grid of Beams and Eigen Based Beamforming [23] can also be adopted towards this problem.

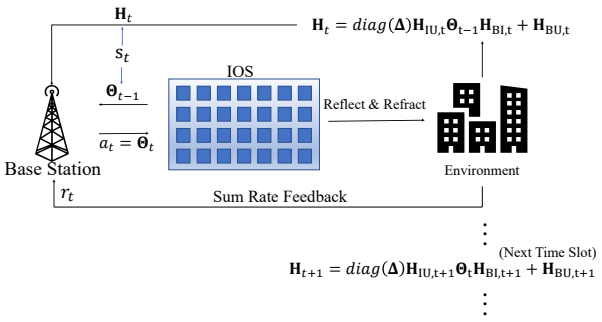

Fig. 2. MDP Process of IOS configuration

Given the time-varying characteristics of channels, we then reformulate $\mathbf{P2}$ as a MDP consisting of the following components.

1) **Action**: The action in the MDP is the configuration of phases of all IOS elements, defined by

$$a_t = \mathbf{\Theta}_t, \forall\theta_t \in \mathbf{\Theta}_t, \theta_t \in (-\pi, \pi). \tag{19}$$

2) **State**: The state in the MDP refers to the channel states and the IOS phase shift matrix configuration. The channel state is measured by the equivalent channel between the BS and users, i.e.,

$$\mathbf{H} = diag(\mathbf{\Delta})\mathbf{H}_{IU}\mathbf{\Theta}\mathbf{H}_{BI} + \mathbf{H}_{BU}, \tag{20}$$

where $\mathbf{\Delta} = [\Delta_1, \cdots, \Delta_K]$. The IOS phase shift configuration in each time slot $t-1$ is given by $\mathbf{\Theta}_{t-1}$, with $\mathbf{\Theta}_0 = [0, \cdots, 0]$ being the initial configuration. Then the state of the MDP can be defined by

$$s_t = \{\mathbf{H}_t, \mathbf{\Theta}_{t-1}\}. \tag{21}$$

3) **Reward**: The reward of the MDP is consistent with the objective value of $\mathbf{P2}$, i.e., the sum rate of all users in time slot t. To avoid the high variation issue brought by the high value of the reward [28], we multiply the sum rate by a coefficient η, and the reward can be expressed by

$$r_t = \eta\sum_{k=1}^{K} R_{k,t}. \tag{22}$$

Then the accumulated reward at step t is given by $\overline{r_t} = \sum_{t'=t}^{T}\delta^{t'-t}r_{t'}$, where $\delta \in [0,1]$ is the discount factor. We remark that the target of reinforcement learning is identical to the optimization problem, which is given as the following proposition:

Proposition 1. *If $\delta = 1$, then the objective of the reinforcement learning over the designed MDP, i.e., reward maximization, is equivalent to the target of problem $\mathbf{P2}$.*

Proof: Please see Appendix A. □

The whole process is shown in Fig. 2. At time slot t, the BS acquires the equivalent channel information $\mathbf{H}_t$ via the pilot signals and records the IOS configuration $\mathbf{\Theta}_{t-1}$ at the previous time slot $t-1$. It then performs the ZF or MMSE method [24] to determine the digital beamforming vector $\boldsymbol{V}_t$.

Then it determines the action a_t, i.e., the IOS phase shifts $\mathbf{\Theta}_t$ at current time slot t, to maximize its expected reward in (22). The BS then transmits the signals to users and obtains the reward of the current time slot t, i.e., the sum rate of all users. The purpose of the whole process is to maximize performance over a period, in which the environment keeps changing from one time slot to another.

Remark 1. *As defined in the state of MDP, we only need to acquire the equivalent channel information from BS to users, i.e.,* $\mathbf{H}_t$*. That is to say, our proposed approach does not require specific two-hop channel information of* $\mathbf{H}_{IU}$*,* $\mathbf{H}_{BI}$ *and* $\mathbf{H}_{BU}$*.*

IV. MC-DDPG Algorithm Design

In this section, we first explain our motivation for using the meta-critic learning method, the introduce the proposed MC-DDPG framework to solve the sum rate maximization problem, as well as illustrate its design with more details.

A. Motivation of Meta-Critic Reinforcement Learning

Different reinforcement learning methods have been utilized in RIS-enabled beamforming to configure the phase shifts of intelligent surface elements as in [13]. However, in real-world dynamic settings, they may face the following challenges.

1) *Difficulty to obtain datasets*: To train a reliable IOS beamforming scheme, we need to collect a sufficiently large amount of CSI from the environment and sum rate feedback from users once the environment changes, which consumes too much time and power.
2) *Age of configuration*: In the dynamic scenario, the communication environment varies with time. Thus, when the settings of the task change, traditional RL methods may take a long time to be re-trained and converge to a sufficiently good IOS configuration solution, which may be out-of-date as the environment changes rapidly.

To deal with these challenges, we aim to train a learning model capable of "learning to learn". That is to say, different from traditional RL methods which can only learn from and perform well on a single task, our proposed method can automatically identify the task and update its model quickly to converge with fewer data collected, as long as it is pre-trained well on multiple tasks. In that way, the model can learn more proficiently on a given new task.

To establish such a framework, we employ actor-critic as the essential structure. The two components, actor and critic, approximately fit into the role of "learning" and "learning to learn", respectively. The actor gives action according to the current state, which is then evaluated by the critic to give a Q-value fed back to the actor, thus it can learn to make better choices [29]. We design the meta-critic to extract features from all the different tasks and learn how to evaluate the actions of different actors on different tasks. The meta-critic can be used subsequently to guide the action of the actor to perform on a newly-given task[2].

[2]In fact, this process can be viewed as the relationship between teacher and students. We want to train a teacher well enough so that he/she is able to instruct students to improve their performance.

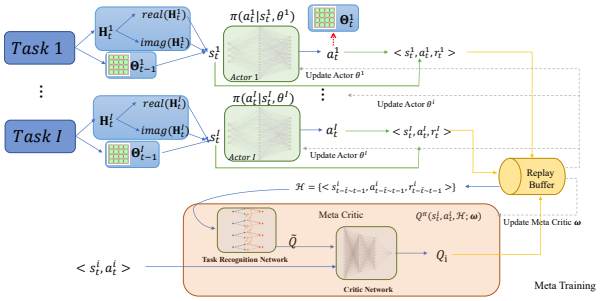

Fig. 3a. Framework of MC-DDPG: Meta Learning Phase

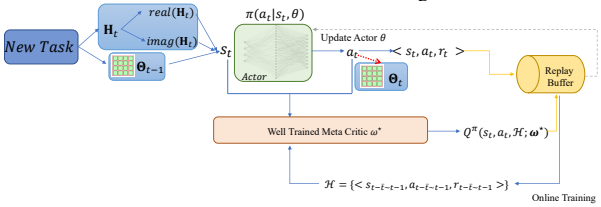

Fig. 3b. Framework of MC-DDPG: Online Learning Phase

B. MC-DDPG Algorithm Framework

A hierarchical design is shown in Figs. 3a and 3b, consisting of the meta-learning phase and online learning phase. Unlike the traditional actor-critic framework [29] where each actor is paired with a critic, we design a meta-critic to perform as the aggregate critic replacing critics of all single actors. The experience from all actors will be sent to one meta critic for updating, which is the key to "learn to learn". Below we first present the key components of the MC-DDPG framework.

1) Task: The blue block refers to the task of RL, which denotes a process of the BS maximizing the sum rates of all users in a specific wireless environment. For different tasks, the channel states and locations of users are various.

2) Actor: The green block in Fig. 3a and Fig. 3b represents the actor of the RL method. Each actor corresponds with a specific task. It receives the state information from task i as defined in (21), and outputs correspondent action from learned *policy*. As the equivalent channel matrix $\mathbf{H}$ is of complex value, we first divide it into real and imaginary parts, which are combined together with the IOS configuration $\mathbf{\Theta}$ to be fed into the policy network illustrated in detail in Section IV-C.

3) Meta Critic: The meta-critic can be divided into two parts, i.e., a task recognition network and a critic network. Given a specific task, the task recognition network extracts the history information from the replay buffer and generates the task-recognition Q-value that represents the feature of the task, i.e., the characteristics of the wireless environment. The task-recognition Q-value is subsequently sent to the critic network together with the state-action information of this task. The critic network outputs a task-specific Q-value to iteratively update the actor networks [15], which can be viewed as the evaluation of the IOS configuration in the current wireless environment.

Meta Learning Process Description: The meta-learning phase can be described as follows: First, for each task i, its current state s_t^i is fed to actor i. Then the actor refers to its learned policy $\pi(a|s_t^i)$ to generate an action a_t^i. Task i executes

the action and receives the reward r_t^i from the environment, then the state-action-reward information $< s_t^i, a_t^i, r_t^i >$ will be stored in the replay buffer. Meanwhile, the meta critic collects the history information of task i, i.e., $\mathcal{H}_t^i$[3], from the replay buffer together with the state-action pair $< s_t^i, a_t^i >$ to give a task-specific Q-value which can further update the actor networks. The meta-critic is updated by the trajectories of all tasks in the replay buffer deployed in the BS, which will be discussed in detail in Section IV-C.

In the online learning phase, for a newly-coming real-time task, the critic is kept static. Therefore, it is directly used to evaluate the action of the actor and update it, while the update of the actor network is just the same as that of the meta-learning phase.

C. Detailed Design of MC-DDPG

In this subsection, we first explain the essential parts of our proposed MC-DDPG algorithm, including the tailored design of the meta-critic and actors. Then the *Explore and Reload* procedure designed to assist in the solution searching is also introduced. Finally, the analysis of complexity for our proposed method is given.

1) Tailored Description of Meta Critic and Actors: We apply the TD3 structure [30] to design our meta-critic such that two Q-networks are introduced for accurate Q-value estimation. We use two neural networks (NN) with weights $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2$ as the Q-networks to parameterize the meta critic, and each actor of task i is also modeled as an NN policy $\pi(a|s_t^i, \boldsymbol{\theta}^i)$ with weights $\boldsymbol{\theta}^i$.

In the dynamic case, the distribution of trajectories, i.e., the time series of the state-action pairs $\{< s_1, a_1 >, \cdots, < s_t, a_t >, \cdots\}$, from different tasks may deviate. Thus, to extract aggregate features from different wireless environments, we cannot directly use the structure of the traditional critic network. Instead, we are supposed to design a critic that is capable of identifying different environments by collecting the history of time-related series of them, including CSI, IOS configurations, and sum rates. For that purpose, since each task refers to a specific wireless environment, we introduce the Long-Short Term Memory (LSTM) networks [31] as a *task recognition* network that outputs the task recognition Q-value. Following the interaction process in Fig. 2, we define the task i's history $\mathcal{H}_{[u\sim v]}^i$ as a segment of tuples of state, action, and reward from step u to step v, i.e., $\mathcal{H}_{u\sim v}^i = \{s_u^i, a_u^i, r_u^i, \ldots, s_v^i, a_v^i, r_v^i\}$. For simplicity, we directly use the most recent $\bar{t}$ examples as the input of the task recognition network, i.e., $\mathcal{H}_t^i = \mathcal{H}_{t-\bar{t}\sim t-1}^i = \{s_{t-\bar{t}}^i, a_{t-\bar{t}}^i, r_{t-\bar{t}}^i, s_{t-1}^i, a_{t-1}^i, r_{t-1}^i\}$.

We adopt a four-layer full connected network (FNN) to learn the features from the current state-action pairs and the task recognition Q-value from LSTM. The meta-critic first recognizes the coming task with a task-recognition Q-value as the output, and then evaluates the configuration of IOS in a specific environment by giving a task-specific Q-value. We desire to pre-train a meta-critic that is able to instruct the actor to rapidly adapt to any new tasks, described as below

$$\tilde{Q}_k = f_{LSTM}(\mathcal{H}_{[t-\bar{t},t-1]}^i; \boldsymbol{\omega}_k^{LSTM}), k = 1, 2, \tag{23}$$

$$Q_k(s_t^i, a_t^i, \mathcal{H}_{[t-\bar{t},t-1]}^i; \boldsymbol{\omega}_k) = f_{FNN}(s_t^i, a_t^i, \tilde{Q}_k; \boldsymbol{\omega}_k^{FNN}), k = 1, 2, \tag{24}$$

where the $\tilde{Q}$ denotes the task recognition Q-value, and the Q-value in (24) represents the task-specific Q-value.

For the actor, to avoid the overfitting problem of the complex network, we use FNN as our policy network. It takes current state s_t as input and outputs a deterministic action

$$\pi(a|s_t^i; \boldsymbol{\theta}^i) = f_{FNN}(s_t^i; \boldsymbol{\theta}^i). \tag{25}$$

2) Loss Function: To train the policy and value networks, we first define the loss functions of the actor and critic networks, respectively, then minimize them based on the backpropagation method. For the meta critic, we use temporal difference (TD) error of all tasks as the loss function [32]. It depicts the difference between the estimated and real Q-value, by minimizing which we can get a critic network that can better evaluate the current configuration of IOS.

$$\begin{aligned} L(\boldsymbol{\omega}_k) = \frac{1}{I}\sum_{i=1}^{I} \mathbb{E}_{\pi(\boldsymbol{\theta}^i)}[Q(s_t^i, a_t^i, \mathcal{H}_{[t-\bar{t},t-1]}^i; \boldsymbol{\omega}_k) - \\ (r_t + \gamma \min_k Q(s_{t+1}^i, a_{t+1}^i, \mathcal{H}_{[t-\bar{t}+1,t]}^i; \boldsymbol{\omega}_k))]^2, k = 1, 2, \end{aligned} \tag{26}$$

where I refers to the number of tasks. TD error in (26) represents the similarity between estimated Q-value $Q(s_t^i, a_t^i, \mathcal{H}_{[t-\bar{t},t-1]}^i; \boldsymbol{\omega}_k)$ and target Q-value $(r_t + \gamma min_{k=1,2} Q(s_{t+1}^i, a_{t+1}^i, \mathcal{H}_{[t-\bar{t}+1,t]}^i; \boldsymbol{\omega}_k))$ of two critic networks, and so its minimization can help meta critic better estimate Q-value of all tasks. As for the actor of each task, the loss function can be represented by the negative Q-value

$$J(\boldsymbol{\theta}^i) = \mathbb{E}_{\pi(\boldsymbol{\theta}^i)}[-Q(s_t^i, a_t^i, \mathcal{H}_{[t-\bar{t},t-1]}^i; \boldsymbol{\omega}_1)], \tag{27}$$

minimization of which is equivalent to maximizing the expected accumulated reward in (22).

3) Parameter Update: The parameters update can be expressed by

$$\boldsymbol{\omega}_{t+1} = \boldsymbol{\omega}_t - \rho \nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}), \tag{28}$$

$$\boldsymbol{\theta}_{t+1}^i = \boldsymbol{\theta}_t^i - \rho \nabla_{\boldsymbol{\theta}^i} L(\boldsymbol{\theta}^i), \tag{29}$$

where the gradient in (28) and (29) can be given by [29]

$$\begin{aligned} \nabla_{\boldsymbol{\omega_k}} L(\boldsymbol{\omega_k}) = \frac{1}{I}\sum_{i=1}^{I} [2L(\boldsymbol{\omega_k}) \nabla_{\boldsymbol{\omega_k}} (Q(s_t^i, a_t^i, \mathcal{H}_{[t-\bar{t},t-1]}^i; \boldsymbol{\omega_k}) \\ -\gamma \min_{k=1,2} Q(s_{t+1}^i, a_{t+1}^i, \mathcal{H}_{[t-\bar{t}+1,t]}^i; \boldsymbol{\omega}_k))], \end{aligned} \tag{30}$$

$$\nabla_{\boldsymbol{\theta}^i} J(\boldsymbol{\theta}^i) = -Q(s_t^i, a_t^i, \mathcal{H}^i; \boldsymbol{\omega}_k)) \nabla_{\boldsymbol{\theta}^i} \log \pi(a|s_t^i, \boldsymbol{\theta}^i) \tag{31}$$

The target networks can be soft-updated [33] as below:

$$\boldsymbol{\omega}_k' \leftarrow \tau \boldsymbol{\omega}_k + (1-\tau)\boldsymbol{\omega}_k', k \in \{1, 2\}, \tag{32}$$

$$\boldsymbol{\theta}^{i\prime} \leftarrow \tau \boldsymbol{\theta}^i + (1-\tau)\boldsymbol{\theta}^{i\prime}. \tag{33}$$

4) Explore & Reload procedure for IOS configuration: Note that in our system we assume IOS with numerous

[3]The definition of $\mathcal{H}_t^i$ can be referred to following Section IV-C1.

elements, thus our considered problem $\mathbf{P2}$ has a high-dimensional action space, making the convergence speed a main concern. In this case, it is rather hard to set the learning rate. Thus, we design a stochastic *Explore and Reload* procedure where an exploration noise e is introduced to enhance the randomness of the action, i.e., the choice of phase shift of each IOS element, thereby avoiding the accumulated deviation error from the optimal point.

$$a_t = \pi(a|s_t) + e, \tag{34}$$

where $e \sim \mathcal{N}(0, \epsilon)$, ϵ refers to the exploration factor. Before training, we initialize ϵ as ϵ_0. Then in each training episode, we record the maximum reward $\mathcal{R}_{max}$ and the corresponding model of policy π^{max}. We also set two thresholds Th_{reward} and Th_{eps}. For the sake of convergence, we set ϵ to be exponentially decaying as the number of episodes grows until convergence. If the current reward declines beyond Th_{reward} compared to $\mathcal{R}_{max}$, i.e.,

$$\mathcal{R}_{max} - \mathcal{R}_{current} > Th_{reward} \tag{35}$$

or $\mathcal{R}_{max}$ has not been updated for Th_{eps} episodes, the actor reloads the best recorded model π^{max} and resets the exploration noise $\epsilon = \epsilon_0$ to restart the exploration.

D. Algorithm Description

We summarize the proposed MC-DDPG algorithm in Alg. 1, which includes two phases: meta-training and online learning. At the beginning of meta-training, we first initialize the network parameters and replay buffer. For each episode, we select I learning tasks and initialize them (Line 5), then conduct $MaxStep$ steps. In one task at each step, we select an action by policy to set the IOS configuration and add exploration noise on it before sending it to IOS with a reward and the next state fed back (Line 8). The transition tuple is stored in the replay buffer in Line 9. Lines 10-11 are the process of updating the meta critic, while Lines 12-13 aim to update the parameters of the actor policy network of task i with some delay. Line 15 denotes the model reloading process we described in Section IV-C4. The output of the meta training is a well-trained meta critic $\boldsymbol{\omega}^\star$ (Line 16).

In the online learning phase with a newly coming task, we directly use well-trained meta critic $\boldsymbol{\omega}^\star$ to estimate Q-value, and thus, only the actor needs to be trained. We first initialize the policy network and replay buffer (Line 20). The actor network determines the current IOS configuration and stores the tuple $< s_t, s_{t+1}, a_t, r_t >$ in the buffer as shown in Lines 8-9. The actor is then updated and reloaded (Lines 26-28). The output of online learning is the trained policy of actor $\boldsymbol{\theta}^\star$ (Line 29), which determines the IOS configuration scheme. The convergence analysis of proposed MC-DDPG is given below:

Proposition 2: The MC-DDPG algorithm can converge guaranteed by the decaying exploration noise and learning rate.

Proof: Please see Appendix B. □

Algorithm 1 MC-DDPG Algorithm for IOS-assisted Multi-user Communication

1: **Meta Training Phase:**
2: **input:** Multiple task samples from different wireless environments.
3: **Initialize:** (For each task i) Critic Networks $Q_{\boldsymbol{\omega}_1}, Q_{\boldsymbol{\omega}_2}$, and actor network π_{θ^i} with parameters $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \boldsymbol{\theta}^i$; Target Networks $\boldsymbol{\omega}_1' \leftarrow \boldsymbol{\omega}_1, \boldsymbol{\omega}_2' \leftarrow \boldsymbol{\omega}_2, \boldsymbol{\theta}^{i\prime} \leftarrow \boldsymbol{\theta}^i$;Replay Buffer $\mathcal{B}^i$;
4: **for** eps in range(*MaxEpisode*) **do**
5: Sample I tasks and initialize states $s_0^1, \cdots, s_0^I$ with initial channel information and default IOS configurations.
6: **for** t in range(*MaxStep*) **do**
7: **for** each task i **do**
8: Configure IOS by (34) and get reward r_t^i and next state s_{t+1}^i.
9: Store the transition tuple into replay buffer $\mathcal{B}^i$.
10: Sample a batch from the replay buffer $\mathcal{B}^i$.
11: Update Meta Critic by (26) and (28).
12: Update $\boldsymbol{\theta}^i$ by (27) and (29) with delay.
13: Update target networks by (32) and (33) with delay.
14: **for** each task i **do**
15: Follow the procedure described in Section IV-C4.
16: **Output:** Well-trained meta critic $\boldsymbol{\omega}^\star$.
17: ———————————————————
18: **Online Training Phase:**
19: **input:**A new task from a new wireless environment; Well-trained meta critic $\boldsymbol{\omega}^\star$.
20: **Initialize:** Policy network $\boldsymbol{\theta}_0$; Replay Buffer $\mathcal{B}$;
21: **for** eps in range(*MaxEpisode*) **do**
22: Initialize system state s_0 with initial channel information and default IOS configurations.
23: **for** t in range(*MaxStep*) **do**
24: Configure IOS by (34) and get reward r_t and next state s_{t+1}.
25: Store transition tuple into replay buffer $\mathcal{B}$.
26: Sample a batch from the replay buffer $\mathcal{B}$.
27: Update $\boldsymbol{\theta}$ by (27) and (29).
28: Follow the procedure described in Section IV-C4.
29: **Output:** The trained policy of actor $\boldsymbol{\theta}^\star$.

E. Computation Complexity Analysis of MC-DDPG

The computing process of MC-DDPG can be divided into two parts: the actor and the meta-critic. For the actor whose policy is estimated by a FNN, we define $n_{a,v}$ as the number of neurons in the hidden layer v. As the state and action space is $2MK$ and N respectively, the time complexity of actor network is $\mathcal{O}(2MKn_{a,1}+\sum_{v=1}^{V_3-1} n_{a,v}n_{a,v+1}+Nn_{a,V_3})$, where V_3 refers to the number of the hidden layers of FNN.

Concerning the meta-critic, we first consider the complexity of the LSTM network, which is determined by the number of memory cells and the size of each layer. We assume that the length of the time series is $\bar{t}$, which is also the number of memory cells. Meanwhile, we denote $n_{l,v}$ as the size of layer v, while V_1 refers to the number of hidden layers. Then the computation complexity of LSTM can be expressed by $\mathcal{O}(4\bar{t}[(2MK+N+1)n_{l,1}+\sum_{v=1}^{V_1-1} n_{l,v}n_{l,v+1}])$ [34]. As for the critic network using FNN, we follow the analysis of the

actor network and represent it as $\mathcal{O}((2MK+N+1)n_{c,1}+\sum_{c=1}^{V_2-1} n_{c,v}n_{c,v+1}+n_{c,V_2})$, where V_2 and $n_{c,v}$ refer to the number of hidden layers in critic network and the size of each layer v respectively.

According to the above analysis, we now can give the sum-up computation complexity of MC-DDPG as $\mathcal{O}(\alpha_1(MK)+\alpha_2 N+\alpha_3)$, in which $\alpha_1 = 2(n_{a,1}+4\bar{t}n_{l,1}+n_{c,1}), \alpha_2 = 4\bar{t}n_{l,1}+n_{c,1}+n_{a,V_3}, \alpha_3 = 4\bar{t}(n_{l,1}+\sum_{v=1}^{V_1-1} n_{l,v}n_{l,v+1})+n_{c,1}+\sum_{v=1}^{V_3-1} n_{a,v}n_{a,v+1}+\sum_{c=1}^{V_2-1} n_{c,v}n_{c,v+1}+n_{c,V_2}$. It can be seen that once the structure and parameters of networks are determined, the complexity of MC-DDPG increases linearly with the problem size of $\mathbf{P2}$, i.e., MK and N.

V. Simulation Results

In this section, we evaluate the proposed MC-DDPG approach in dynamic settings. The performance of MC-DDPG is compared to two benchmark algorithms including a state-of-the-art RL method and a traditional optimization method.

A. Simulation Setup

Major parameters of the simulation are summed up in Table I. We assume that the task is updated[4] every 300 episodes, each of which consists of 20 time slots. We compare our proposed scheme with two benchmarks, in each of which the whole algorithm needs to be initialized and performed again for any newly-coming task.

1) Twin delayed deep deterministic policy gradient (TD3), which is a state-of-the-art RL algorithm [30] without the meta critic. It introduces two Q-networks as the critic for better estimation of Q-value, which help it to outperform other RL based methods on many traditional RL tasks.
2) Zero-Force Exhausting (ZF Exhaust), where the digital beamforming is based on the ZF method, and the IOS phase shift optimization is performed via the exhaustion method with discretized phase shifts of IOS elements [35].

TABLE I
Simulation Parameters

Parameter	Value
Total number of users K	$2 \sim 4$
Number of antennas on BS M	8
Number of IOS elements	{64, 100, 256, 1024}
Time limitation T	20 time slots
Duration of time slot Φ	0.01 s
Transmit Power of Antenna P_T	23 dBm (200 mW)
Carrier frequency f_c	5.9 GHz
Noise power spectral density	-93 dBm/Hz
Distribution of stochastic policy for exploration	Gaussian
Initial exploration factor ϵ_0	$1e^{-3}$
Learning rate ρ	$1e^{-5}$
Batch size	128
Replay Buffer size	10000
Discount Factor γ	0.99
Soft update factor τ	0.005
Scenario update interval	300
Software Platform	Intel® Core™ i5-9300H CPU @ 2.40GHz NVIDIA GeForce GTX 1650 Python 3.8.5 with Pytorch 1.12.1+cuda 11.6

B. Performance Evaluation

1) Dynamic Channel State:

[4]Each task correspondent to different channel states and locations of users.

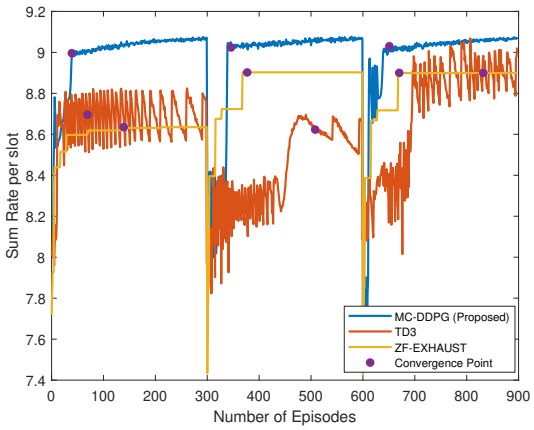

Fig. 5. Sum rate performance with respect to the varying channel states

In Fig. 5, we evaluate how the performance of the proposed algorithm varies with the channel states. Specifically, we update the transition probability matrix in (10) every 300 episodes[5], i.e., the task is also updated periodically. As shown in Fig. 5, the proposed MC-DDPG converges within 50 episodes to provide a better performance within 50 episodes and achieves a higher sum rate compared to the two benchmarks. This shows that the proposed scheme can efficiently adapt to rapid environment changes. It can be also shown that there exists a trade-off between the convergence speed and the achievable sum rate of our proposed method.

2) Varying User's locations:

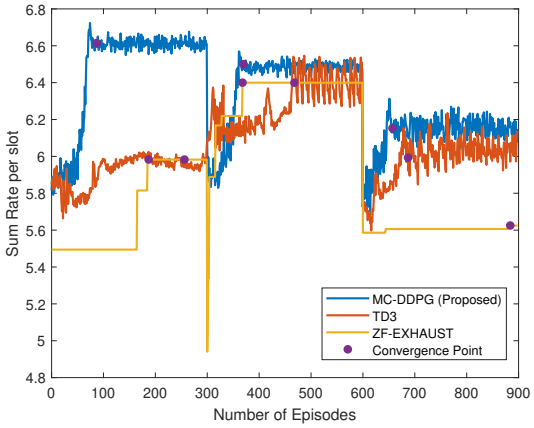

Fig. 6. Sum rate performance with respect to varying users' loactions

In Fig. 6, we evaluate the performance of the proposed scheme considering mobile users. It is clearly shown that the achievable sum rate of MC-DDPG is higher compared to the two benchmarks, and its time for convergence is significantly shorter. We can also observe a significant degradation of ZF-Exhaust as it is subject to random initialization of IOS phase shifts thus the solution can be harder to search if the starting point deviates from the optimal too much.

3) Speed of Users:

In this simulation, we assume that for each task between time slots, the users keep moving. For simplicity, we set them to move in the same direction, like the x-axis or y-axis.

[5]This setting is to guarantee the convergence of the ZF Exhaust scheme.

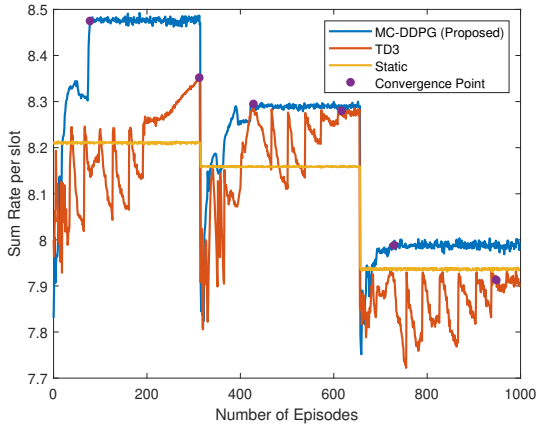

Fig. 7. Sum rate performance v.s Speed of User

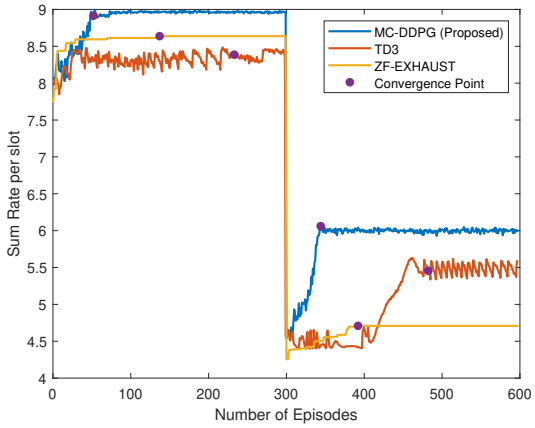

Fig. 8. Sum rate performance concerning Number of Users

Each task corresponds to a specific speed for all users, which means in each time slot the user's locations are changed by a constant. In this case, we can not use Zero-Forcing Exhausting as the benchmark as the users' locations are dynamic within each task. Thus, another benchmark based on static IOS beamforming is introduced. We get the initial CSI at the beginning of the first task, then use the ZF-Exhaust method to search for a solution and keep it static throughout the whole process.

As shown in Fig. 7, the speed of the three tasks increases from left to right, which explains why the performance of all methods drops. With faster speed, the location of the user is more heterogeneous, thus it is harder to find a stable solution. Although on average, it takes a longer time for MC-DDPG to converge compared to other settings, our proposed method still shows faster convergence and better performance than the other two benchmarks, which verifies the robustness for varying speeds of each user.

We also want to remark that the duration of each time slot Φ, is expected to be at least the same as the interval of interactions[6],

$$\Phi \geq \Delta t, \tag{36}$$

where

$$\Delta t = RTT + t_{proc}, \tag{37}$$

which indicates that Δt depends on the time of processing t_{proc} and the roundtrip time (RTT). With our device, Φ and Δt are both set to 0.01s. And in the simulation of Fig. 7, the distances the users move in each time slot are 0.1m, 0.5m, and 0.8m respectively. Thus the corresponding speeds are 10m/s, 50m/s, and 80m/s when it comes to reality.

4) Entrance/Departure of Users:

In Fig. 8, we consider a dynamic case where users may leave the cell coverage area from time to time. In this case, the input size of meta critic is not consistent as state s_t has $\mathrm{H}_t \in \mathbb{C}^{K\times M}$ component which varies with the number of users. Without loss of generality, We assume that the maximum number of users is 4. When $K < 4$, we use zero-padding to reform the input state s_t to train the meta critic.

Fig. 8 shows how the sum rate varies when users leave the coverage of the BS. For the case where two out of four users leave, the MC-DDPG can converge very fast compared to other benchmarks, meanwhile providing better performance. This verifies the robustness of our proposed MC-DDPG against a fast-varying environment.

[6]It refers to the time from one reception of CSI and giving action to the next reception.

C. Influence of the Number of IOS Elements

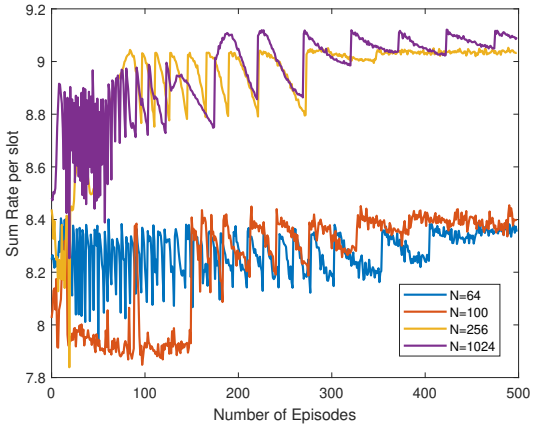

Fig. 9a. Convergence performance of MC-DDPG given different numbers of IOS elements

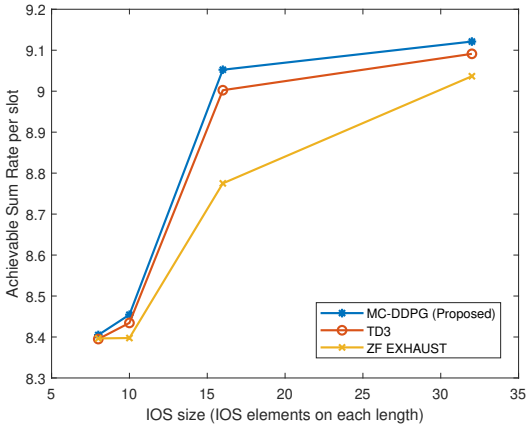

Fig. 9b. Achievable sum rate v.s. the number of IOS elements

We simulate the proposed MC-DDPG algorithm with a single task under a different number of IOS elements. The simulation results are shown as Fig. 9a and Fig. 9b. The curves

interact with each other because the solution space is much wider with more IOS elements, and it brings more vacillation and exploration at first, accompanied by a possible lower performance. But eventually, IOS with more elements achieves higher performance, which implies the effectiveness of the proposed meta-critic method against the large-scale IOS-assisted communication system. It also proves the meaningfulness of increasing the number of IOS elements and the introduction of model reloading that eases the instability problem.

VI. Conclusion

In this paper, we considered an IOS-assisted communication system in dynamic environments, for which we proposed a MC-DDPG beamforming scheme for sum rate maximization given the limited channel information. By designing and training a meta critic, the proposed scheme can adapt to the dynamic environment such as the heterogeneous channel states, user positions/velocity, and the number of users. Simulation results show that the online MC-DDPG algorithm achieves a faster convergence speed and a higher sum rate compared to the benchmarks. Three conclusions can be drawn below. *First*, the designed meta-critic significantly enhances the IOS-assisted multi-user communications against the user mobility and the dynamic channel states. *Second*, there exists a trade-off between the convergence speed of the proposed MC-DDPG and the achievable sum rate. *Third*, our proposed method is robust against different numbers of IOS elements with respect to the sum rate of the IOS-aided communications system.

Appendix A
Proof of Proposition 1

The objective of the task is to obtain a policy $\pi(a|s_t)$ to maximize the accumulated reward at each step t, which is equivalent to maximizing the episode expected reward $\sum_{t=1}^{T}\delta^t r_t$ [29]. We just insert the expression of r_t in (22) into it and let $\delta=1$, then we can get

$$\sum_{t=1}^{T}\delta^t r_t=\eta\sum_{t=1}^{T}\sum_{k=1}^{K}R_{k,t}. \tag{38}$$

As we have already defined $a_t=\mathbf{\Theta}_t$, and η is just a constant, the target of the reinforcement learning task is equivalent to that of $\mathbf{P2}$.

Appendix B
Proof of Proposition 2

First, we introduce a lemma that is proved by Singh et al. [36].

Lemma 1: Consider a stochastic process $(\alpha_t,\Delta_t,F_t), t\geq 0$, where $\alpha_t,\Delta_t,F_t: X\rightarrow\mathcal{R}$ satisfy the equations

$$\Delta_{t+1}(x)=(1-\alpha_t(x))\Delta_t(x)+\alpha_t(x)F_t(x), x\in X, t=0,1,2,\cdots \tag{39}$$

Let P_t be a sequence of increasing σ-fields such that α_0 and Δ_0 are P_0-measurable and α_t,Δ_t and F_{t-1} are P_t-measurable, $t=1,2,\ldots$. Assuming that the following hold:

1. The set X is finite.
2. $0\leq\alpha_t(x)\leq 1, \sum_t\alpha_t(x)=\infty, \sum_t\alpha_t^2(x)<\infty$ with probability 1.
3. $\|E\{F_t(\cdot)|P_t\}\|_W\leq\kappa\|\Delta_t\|_W+c_t$, where $\kappa\in[0,1)$ and c_t converges to zero with probability 1.
4. $Var\{F_t(x)|P_t\}\leq K(1+\|\Delta_t\|_W)^2$, where K is some constant.

Where $\|\cdot\|_W$ denotes the maximum norm. Then, Δ_t converges to zero with probability 1.

The following proof is based on the proof finished by Fujimoto et al. [30]. According to the expression of action-selection (34) of MC-DDPG, as the exploration noise e is set to be exponentially decaying with the growth of the number of episodes until convergence, it converges to zero with probability 1. Thus, we only need to consider the policy $\pi(a|s_t)$, which is determined by each $\boldsymbol{\theta}_i$. Note that by (31), the update of $\boldsymbol{\theta}_i$ depends on the Q-value function of the current state and action, i.e., $Q(s_t^i,a_t^i,\mathcal{H};\boldsymbol{w}_1)$. That is to say if for each i, $Q(s_t^i,a_t^i,\mathcal{H}_t^i;\boldsymbol{w}_1)$ converges to the optimal value function $\hat{Q}_i$, the performance of MC-DDPG is guaranteed to converge.

For each task i, we set Q_t^i as the Q-value of this task at step t, and

$$P_t^i=\{Q_0^i(\boldsymbol{w}_1),Q_0^i(\boldsymbol{w}_1),s_0^i,a_0^i,\mathcal{H}_0^i,\rho_0,r_1^i,s_1^i,a_1^i,\cdots,s_t^i,a_t^i\}, \tag{40}$$

where ρ_t denotes the learning rate at step t. Applying Lemma 1, we let $X=S\times H\times A, \Delta_t^i=Q_t^i(\boldsymbol{w}_1)-\hat{Q}_i, \alpha_t=\rho_t$, where S, H and A refer to the state space, history, and action space respectively. Let $\hat{a}^i=\text{argmax}_a Q(s_{t+1}^i,a|\boldsymbol{w}_1)$, then we have

$$\begin{aligned}
&\Delta_{t+1}(s_t^i,a_t^i,\mathcal{H}_t^i)=(1-\rho_t(s_t^i,a_t^i))(Q(s_t^i,a,\mathcal{H}_t^i|\boldsymbol{w}_1)-\hat{Q}_i(s_t^i,a_t^i,\mathcal{H}_t^i))\\
&+\rho_t(s_t^i,a_t^i)(r_t^i+\gamma\min(Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_1),Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_2))\\
&-\hat{Q}_i(s_t^i,a_t^i,\mathcal{H}_t^i))\\
&=(1-\rho_t(s_t^i,a_t^i))\Delta_t(s_t^i,a_t^i,\mathcal{H}_t^i)+\rho_t(s_t^i,a_t^i)F_t(s_t^i,a_t^i,\mathcal{H}_t^i),
\end{aligned} \tag{41}$$

where

$$\begin{aligned}
&F_t(s_t^i,a_t^i,\mathcal{H}_t^i)=\\
&r_t^i+\gamma\min(Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_1),Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_2))\\
&-\hat{Q}_i(s_t^i,a_t^i,\mathcal{H}_t^i))\\
&=r_t^i+\gamma\min(Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_1),Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_2))\\
&-\hat{Q}_i(s_t^i,a_t^i,\mathcal{H}_t^i))+\gamma Q_t(s_t^i,\hat{a}^i,\mathcal{H}_t^i|\boldsymbol{w}_1)-\gamma Q_t(s_t^i,\hat{a}^i,\mathcal{H}_t^i|\boldsymbol{w}_1)\\
&=F_t^Q(s_t^i,a_t^i,\mathcal{H}_t^i)+c_t,
\end{aligned} \tag{42}$$

in which $F_t^Q=r_t^i+\gamma Q_t(s_t^i,\hat{a}^i,\mathcal{H}_t^i|\boldsymbol{w}_1)-\hat{Q}_i(s_t^i,a_t^i,\mathcal{H}_t^i)$ is identical with the traditional Deep Q-learning that use only one network and $c_t=\gamma\min(Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_1),Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_2))-\gamma Q_t(s_t^i,\hat{a}^i,\mathcal{H}_t^i|\boldsymbol{w}_1)$.

Let $Q'=r_t^i+\gamma\min(Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_1),Q(s_{t+1}^i,\hat{a}^i,\mathcal{H}_{t+1}^i|\boldsymbol{w}_2))$. It can be shown that $Q(\boldsymbol{w}_1)$

$$\begin{aligned}
&Q_{t+1}(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_1)-Q_{t+1}(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_2)\\
&=Q_t(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_1)+\rho(s_t^i,a_t^i)(Q'-Q_t(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_1))\\
&-Q_t(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_2)-\rho(s_t^i,a_t^i)(Q'-Q_t(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_2))\\
&=(1-\rho(s_t^i,a_t^i))(Q_t(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_1)-Q_t(s_t^i,a_t^i,\mathcal{H}_t^i|\boldsymbol{w}_2)).
\end{aligned} \tag{43}$$

Thus $Q(\boldsymbol{w}_1) - Q(\boldsymbol{w}_2)$ converges to zero, which indicates that c_t also converges to zero. The assumptions of Lemma 1 can be examined as follows.

1. The MDP defined in our scenario is finite, and so is the space of history, which verifies assumption 1.
2. The policy we set and decay the learning rate meets the requirement of assumption 2, as $\alpha_t = \rho_t$.
3. $\|E\{F_t(\cdot)|P_t\}\|_W = \left\|E\{F_t^Q|P_t\}\right\|_W + c_t$, we have already shown that c_t converges to zero. According to the basic characteristic of bellman functions and Q-learning, $E[F_t^Q|P_t] \leq \gamma \|\Delta_t\|$, thus assumption 3 holds.
4. $Var[r(s,a)] < \infty, \forall s, a$, which guarantees that assumption 4 holds.

This shows that for each i, $Q(s_t^i, a_t^i, \mathcal{H}_t^i; \boldsymbol{w}_1)$ converges to $\hat{Q}_i$ as Δ_t converges to zero with probability 1, which proves the convergence of MC-DDPG.

REFERENCES

[1] Q. Luo, B. Di, and Z. Han, "Meta-critic reinforcement learning for ios-assisted multi-user communications in dynamic environments (accepted)," in *2023 IEEE 97th Vehicular Technology Conference: (VTC2023-Spring)*, Florence, Italy, pp. 1–6, Jun. 2023.

[2] W. Jiang, B. Han, M. A. Habibi, and H. D. Schotten, "The Road Towards 6G: A Comprehensive Survey," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 334–366, Feb. 2021.

[3] M. Z. Chowdhury, M. Shahjalal, S. Ahmed, and Y. M. Jang, "6G Wireless Communication Systems: Applications, Requirements, Technologies, Challenges, and Research Directions," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 957–975, Jul. 2020.

[4] S. Zeng, H. Zhang, B. Di, Z. Han, and L. Song, "Reconfigurable Intelligent Surface (RIS) Assisted Wireless Coverage Extension: RIS Orientation and Location Optimization," *IEEE Communications Letters*, vol. 25, no. 1, pp. 269–273, Sept. 2021.

[5] H. Zhang, S. Zeng, B. Di, Y. Tan, M. Di Renzo, M. Debbah, Z. Han, H. V. Poor, and L. Song, "Intelligent omni-surfaces for full-dimensional wireless communications: Principles, technology, and implementation," *IEEE Communications Magazine*, vol. 60, no. 2, pp. 39–45, Feb. 2022.

[6] X. Ma, Z. Chen, W. Chen, Z. Li, Y. Chi, C. Han, and S. Li, "Joint channel estimation and data rate maximization for intelligent reflecting surface assisted terahertz MIMO communication systems," *IEEE Access*, vol. 8, pp. 99 565–99 581, May 2020.

[7] C. Huang, R. Mo, and C. Yuen, "Reconfigurable intelligent surface assisted multiuser MISO systems exploiting deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 8, pp. 1839–1850, Aug. 2020.

[8] G. Lee, M. Jung, A. T. Z. Kasgari, W. Saad, and M. Bennis, "Deep reinforcement learning for energy-efficient networking with reconfigurable intelligent surfaces," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland, Jun. 2020.

[9] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, Dec. 2016.

[10] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial intelligence review*, vol. 18, pp. 77–95, Jun. 2002.

[11] Y. Ge and J. Fan, "Beamforming Optimization for Intelligent Reflecting Surface Assisted MISO: A Deep Transfer Learning Approach," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3902–3907, Mar. 2021.

[12] M. Jung and W. Saad, "Meta-Learning for 6G Communication Networks with Reconfigurable Intelligent Surfaces," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada, pp. 8082–8086, Jun. 2021.

[13] K. Feng, Q. Wang, X. Li, and C.-K. Wen, "Deep reinforcement learning based intelligent reflecting surface optimization for MISO communication systems," *IEEE Communications Letters*, vol. 9, no. 5, pp. 745–749, May 2020.

[14] Y. Zhang and H. Xu, "Two-Stage Online Reinforcement Learning based Distributed Optimal Resource Allocation for Multiple RIS-assisted Mobile Ad-Hoc Network," in *International Conference on Computing, Networking and Communications (ICNC)*, Honolulu, HI, USA, pp. 563–567, Feb. 2023.

[15] F. Sung, L. Zhang, T. Xiang, T. Hospedales, and Y. Yang, "Learning to learn: Meta-critic networks for sample efficient learning," *arXiv preprint arXiv:1706.09529*, 2017.

[16] B. Sklar, "Rayleigh fading channels in mobile digital communication systems. I. Characterization," *IEEE Communications Magazine*, vol. 35, no. 7, pp. 90–100, Jul. 1997.

[17] Q. Wu, S. Zhang, B. Zheng, C. You, and R. Zhang, "Intelligent Reflecting Surface-Aided Wireless Communications: A Tutorial," *IEEE Transactions on Communications*, vol. 69, no. 5, pp. 3313–3351, May 2021.

[18] S. Zeng, H. Zhang, B. Di, Y. Tan, Z. Han, H. V. Poor, and L. Song, "Reconfigurable intelligent surfaces in 6G: Reflective, transmissive, or both?" *IEEE Communications Letters*, vol. 25, no. 6, pp. 2063–2067, Jun. 2021.

[19] R. W. Heath, N. Gonzalez-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 3, pp. 436–453, Feb. 2016.

[20] W. Wang and W. Zhang, "Intelligent Reflecting Surface Configurations for Smart Radio Using Deep Reinforcement Learning," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 8, pp. 2335–2346, Jun. 2022.

[21] H. S. Wang and N. Moayeri, "Finite-state Markov channel-a useful model for radio communication channels," *IEEE Transactions on Vehicular Technology*, vol. 44, no. 1, pp. 163–171, Feb. 1995.

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

[23] G. Barb, M. Otesteanu, F. Alexa, and A. Ghiulai, "Digital beamforming techniques for future communications systems," in *12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, Porto, Portugal, Jul. 2020.

[24] B. Di, H. Zhang, L. Song, Y. Li, Z. Han, and H. V. Poor, "Hybrid beamforming for reconfigurable intelligent surface based multi-user communications: Achievable rates with limited discrete phase shifts," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 8, pp. 1809–1822, Jun. 2020.

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

[26] X. Ma, J. Zhang, Y. Zhang, Z. Ma, and Y. Zhang, "A PCA-based modeling method for wireless MIMO channel," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Atlanta, GA, USA, pp. 874–879, May 2017.

[27] G. Gupta and M. Younis, "Fault-tolerant clustering of wireless sensor networks," in *IEEE Wireless Communications and Networking (WCNC)*, New Orleans, LA, USA, vol. 3, pp. 1579–1584 vol.3, Mar. 2003.

[28] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *arXiv preprint arXiv:1710.11248*, 2017.

[29] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 5, pp. 834–846, Sept.-Oct. 1983.

[30] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning (ICML)*, Stockholm, Sweden, pp. 1587–1596, Jul. 2018.

[31] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural computation*, vol. 31, no. 7, pp. 1235–1270, Jul. 2019.

[32] M. Sewak, *Deep reinforcement learning*. Springer, 2019.

[33] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv:1509.02971*, 2015.

[34] S. Zhang, Y. Wu, T. Che, Z. Lin, R. Memisevic, R. R. Salakhutdinov, and Y. Bengio, "Architectural complexity measures of recurrent neural networks," in *Advances in Neural Information Processing Systems*, vol. 29, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds. [Online]. Available: https://proceedings.neurips.cc/paper/2016/file/860320be12a1c050cd7731794e231bd3-Paper.pdf

[35] Y. Zhang, B. Di, H. Zhang, Z. Han, H. V. Poor, and L. Song, "Meta-Wall: Intelligent Omni-Surfaces Aided Multi-Cell MIMO Communications," *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, pp. 7026–7039, Sept. 2022.

[36] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári, "Convergence results for single-step on-policy reinforcement-learning algorithms," *Machine learning*, vol. 38, pp. 287–308, Mar. 2000.