Deep Reinforcement Learning Based Joint Downlink Beamforming and RIS Configuration in RIS-aided MU-MISO Systems Under Hardware Impairments and Imperfect CSI

Baturay Saglam*, Student Member, IEEE, Doga Gurgunoglu†, Student Member, IEEE, Suleyman S. Kozat*, Senior Member, IEEE
*Department of Electrical and Electronics Engineering, Bilkent University, Ankara 06800, Turkey, †Division of Decision and Control Systems, KTH Royal Institute of Technology, Stockholm 100 44, Sweden

Abstract—We investigate the joint transmit beamforming and reconfigurable intelligent surface (RIS) configuration problem to maximize the sum downlink rate of a RIS-aided cellular multiuser multiple input single output (MU-MISO) system under imperfect channel state information (CSI) and hardware impairments by considering a practical phase-dependent RIS amplitude model. To this end, we present a novel deep reinforcement learning (DRL) framework and compare its performance against a vanilla DRL agent under two scenarios: the golden standard where the base station (BS) knows the channel and the phase-dependent RIS amplitude model perfectly, and the mismatch scenario where the BS has imperfect CSI and assumes ideal RIS reflections. Our numerical results show that the introduced framework substantially outperforms the vanilla DRL agent under mismatch and approaches the golden standard.

Index Terms—reconfigurable intelligent surface, sum rate, multiuser multiple input single output, hardware impairment, phase-dependent amplitude, deep reinforcement learning

I. INTRODUCTION

Reconfigurable intelligent surface (RIS) is one of the emerging technologies explored for the next generation of wireless communication systems [1]. An RIS consists of numerous reflecting elements with sub-wavelength spacing whose impedances are adjusted to induce the desired phase shifts on the incident wave before it is reflected, allowing a multipath interference of desired nature to be created at the receiver [2]. However, depending on the circuit implementation and the choice of meta-materials, the incident wave is attenuated depending on the desired phase shifts at the individual elements, modeled as the phase-dependent reflection amplitude model [1], which causes significant performance losses [2].

Considering the non-linear model in [1] renders the already complex optimization-based approaches impractical [3]. An alternative to such approaches, deep reinforcement learning (DRL), has become a widely-studied machine learning (ML) approach for RIS-aided wireless systems, such as non-orthogonal multiple access (NOMA) downlink systems [4], millimeter wave communications [5], vehicular communications and trajectory optimization [6]–[8], and the transmit beamforming and phase shifts design [6], [9]–[15]. While the prior applications of DRL to RIS-aided systems assumed ideal reflections, a DRL application considering RIS hardware impairments does not exist to the best of our knowledge.

In this paper, we study the joint design of transmit beamforming and phase shifts for RIS-aided multiuser multiple input single output (MU-MISO) systems through a DRL approach to maximize the sum downlink rate of the users under the phase-dependent amplitude model. Our contributions in this manuscript can be summarized as follows: (i) We present a set of novel modifications for the application of DRL to RIS-aided systems, which address two critical aspects of non-episodic tasks, e.g., the joint design of transmit beamforming and phase shifts, that prior approaches have passed over. (ii) To the best of our knowledge, we devise the first DRL-based approach for the phase-dependent reflection amplitude model in RIS-aided MU-MISO systems to provide an alternative ML-based framework for the suboptimal iterative algorithms proposed in [1]. The presented method obtains near-optimal results by maximizing the actual sum downlink rate corresponding to phase-dependent RIS reflections while the agent assumes them to be ideal. (iii) To ensure reproducibility and support further research on DRL-based RIS systems, we provide our source code and results in the GitHub repository\(^1\).

II. SYSTEM MODEL

We consider the downlink of a narrow-band RIS-aided MU-MISO system consisting of \( K \) single-antenna users, \( M \) base station (BS) antennas, and \( L \) RIS elements. The transmit beamforming matrix \( \mathbf{G} \in \mathbb{C}^{M \times K} \) maps \( K \) data streams denoted by \( \mathbf{x} \in \mathbb{C}^{K \times 1} \) for \( K \) users onto \( M \) transmit antennas. The line-of-sight channel between the BS and RIS is denoted by \( \mathbf{H} \in \mathbb{C}^{L \times M} \), and the diagonal reflection matrix of the RIS is denoted by \( \mathbf{\Phi} \triangleq \text{diag}(\phi_1, \ldots, \phi_L) \in \mathbb{C}^{L \times L} \). The channel

\(^1\)https://github.com/baturaysaglam/RIS-MISO-PDA-Deep-Reinforcement-Learning
between the RIS and the \( k \)th user is denoted by \( h_k \). In the following subsections, we explain two different environment models that we consider: the first being the true environment model of the system with phase-dependent RIS amplitude and perfect CSI, and the second being the mismatch environment with ideal reflection assumption and imperfect CSI.

### A. True Environment Model

The received signal at user \( k \) can be expressed as:
\[
z_k = h_k^T \Phi H G x + w_k, \tag{1}\]
where the complex scalars \( z_k \) and \( w_k \) denote the received signal and the additive receiver noise at the \( k \)th user, respectively, and we assume that \( w_k \sim \mathcal{CN}(0, \sigma_w^2) \) for all \( k \). The RIS follows the phase-dependent amplitude model in [1], with entries \( \phi_l = \beta(\varphi_l) e^{j\varphi_l} \) for \( \varphi \in [0, 2\pi) \), resulting in:
\[
\beta(\varphi_l) = (1 - \beta_{\text{min}}) \left( \frac{\sin(\varphi_l - \mu) + 1}{2} \right) + \beta_{\text{min}}, \tag{2}\]
where \( \beta_{\text{min}} \in [0, 1], \mu \geq 0, \) and \( \kappa \geq 0 \) are constants that depend on the hardware implementation of the RIS. In the golden standard scenario, the BS knows the individual cascaded channels to each user, denoted by:
\[
D_k \triangleq \text{diag}(h_k) \in \mathbb{C}^{L \times M}, \quad \forall k = 1, \ldots, K. \tag{3}\]
Hence (1) can be rewritten as:
\[
z_k = \phi^T D_k G x + w_k, \tag{4}\]
where \( \phi \in \mathbb{C}^{L \times 1} \) denotes the column vector consisting of the diagonal entries of \( \Phi \).

### B. Mismatch Environment Model

In this simplified model, the RIS reflections are assumed to be lossless, i.e., \( \hat{\phi} \triangleq [e^{j\varphi_1}, \ldots, e^{j\varphi_L}]^T \). Moreover, the agent has access to only an imperfect estimate of the cascaded channels, namely:
\[
\hat{D}_k \triangleq \Phi h_k + E_k, \quad \forall k = 1, \ldots, K, \tag{5}\]
where \( E_k \in \mathbb{C}^{L \times M} \) denotes the channel estimation error matrix of the cascaded channel of each user, with independent and identically distributed (i.i.d.) entries \( e_{l,m}^{(k)} \sim \mathcal{CN}(0, \sigma_e^2) \).

### C. Problem Formulation

1) **The Golden Standard Objective:** Our emphasis is to utilize the DRL agent to maximize the sum downlink rate in the system, which is denoted as:
\[
R_{\Sigma} \triangleq \sum_{k=1}^{K} \log \left( 1 + \frac{\| \phi^T \Phi h_k G x \|^2}{\sum_{l \neq k} \| \phi^T \Phi h_l G x \|^2 + \sigma_w^2} \right). \tag{6}\]
The BS aims to maximize (6) by adjusting \( G \) and \( \phi \). Under the transmission power constraint \( P_l \) and the domain restriction of phase shifts, the optimization problem is expressed as:
\[
\begin{align*}
\text{maximize} & \quad R_{\Sigma} \\
\text{subject to} & \quad \varphi_l \in [0, 2\pi), \quad \forall l = 1, \ldots, L, \\
& \quad \text{tr}(GG^H) \leq P_l. \tag{7}\end{align*}
\]
where \( \phi \) depends on \( \varphi_1, \ldots, \varphi_L \) and \( \beta(\varphi_l) \) according to (2) when the BS agent is aware of the true environment model.

2) **The Mismatch Objective:** The optimization problem to be solved in the mismatch scenario is defined as:
\[
\tilde{R}_{\Sigma} \triangleq \sum_{k=1}^{K} \log \left( 1 + \frac{\| \hat{\phi}^T \hat{D}_k G x \|^2}{\sum_{l \neq k} \| \hat{\phi}^T \hat{D}_l G x \|^2 + \sigma_w^2} \right). \tag{8}\]
Consequently, the BS agent considers the following optimization problem:
\[
\begin{align*}
\text{maximize} & \quad \tilde{R}_{\Sigma} \\
\text{subject to} & \quad \varphi_l \in [0, 2\pi), \quad \forall l = 1, \ldots, L, \\
& \quad \text{tr}(GG^H) \leq P_l. \tag{9}\end{align*}
\]
While (7) and (9) have the same form, the performance is evaluated according to the true objective function \( R_{\Sigma} \) in both cases. Therefore, we investigate the impact of a misspecified environment on the BS agent for the system performance.

### III. DEEP REINFORCEMENT LEARNING FRAMEWORK

#### A. Overview

At each discrete time step \( t \), the agent observes a state \( s \in S \) and takes an action \( a \in A \), and observes a next state \( s' \in S \) and receives a reward \( r \), where \( S \) and \( A \) are the state and action spaces, respectively. In fully observable environments, the reinforcement learning (RL) problem is usually represented by a finite Markov Decision Process, a tuple \( (S, A, P, \gamma) \), where \( P \) is the transition dynamics such that \( s', r \sim P(s,a) \) and \( \gamma \in [0,1] \) is a constant discount factor. The objective in RL is to find an optimal policy \( \pi \) that maximizes the value defined by \( V_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i+1} \), where the discount factor \( \gamma \) prioritizes the short term rewards. The policy of an agent is regarded as stochastic if it maps states to action probabilities \( \pi : S \rightarrow p(A) \), or deterministic if it maps states to unique actions \( \pi : S \rightarrow A \). The performance of a policy is assessed under the action-value function (Q-function or critic) that represents \( V_t \) while following the policy \( \pi \) after acting \( a \) in state \( s \): \( Q^t(s,a) = \mathbb{E}_r[s_{t+1}, \ldots] \gamma^t r_{t+i+1} \), where \( \gamma^t \) is the action selected by the policy on the observed next state \( s' \).

In deep RL, the critic is approximated by a deep neural network \( Q_\theta \) with parameters \( \theta \), i.e., the Deep Q-learning algorithm [17]. Given a transition tuple \( t = (s, a, r, s') \), the Q-network is trained by minimizing a loss \( J(\theta) \) on the temporal-difference (TD) error \( \delta \) corresponding to \( Q_\theta \) [18], the difference between the output of \( Q_\theta \) and learning target \( y \):

\[
\begin{align*}
y & \triangleq r + \gamma Q_\theta(s', a'), \\
\delta & \triangleq y - Q_\theta(s, a); \\
\theta & \leftarrow \theta - \eta \nabla J(\theta), \tag{12}\end{align*}
\]
where \( J(\theta) = |\delta|^2 \), \( \nabla J(\theta) \) is the gradient of the loss \( J(\theta) \) with respect to \( \theta \), and \( \eta \) is the learning rate. The target \( y \) in (10) utilizes a separate target network with parameters \( \theta' \) that
maintains stability and fixed objective in learning the optimal Q-function [17]. The target parameters are updated to copy the parameters \( \theta \) after a number of learning steps.

### B. The Soft Actor-Critic Algorithm

In contrast to the prior work that employed suboptimal DRL algorithms, e.g., [4], [9], [10], [13], we leverage a state-of-the-art DRL algorithm, Soft Actor-Critic (SAC) [19], which outperforms its counterparts in the majority of the well-known DRL benchmarks [19]. The Soft Actor-Critic (SAC) algorithm is actor-critic, operating in continuous action spaces, hence it employs a separate actor network to choose actions on the observed states since the maximum \( \max_a Q(s, a) \) to select actions is intractable due to an infinite number of possible actions. It is also an off-policy method; hence, it stores transitions (or experiences) collected during exploration steps into a buffer, the experience replay memory [20], and samples them in each learning step to train the actor and critic networks. Contrarily, on-policy algorithms immediately use the collected transitions for training and discard them immediately. While an on-policy algorithm is also applicable, as in [6], SAC also outperforms the competing on-policy algorithms in several DRL benchmarks [19]. Furthermore, our initial simulations showed that no actor-critic algorithm could converge for the problem of interest except SAC, although we performed intensive hyper-parameter tuning.

A SAC agent maintains three networks: two Q-networks and a single stochastic policy network (or actor network), each being a multi-layer perceptron (MLP). The reason behind using two Q-networks is to reduce the overestimation of Q-value estimates [21]. The Q-networks take the states provided by the environment and actions produced by the actor network as inputs and produce Q-value estimates, which are scalar values. Given the actor network \( \pi_\psi \) parameterized by \( \psi \), the Q-networks are jointly trained in the SAC algorithm as:

\[
\hat{y} = r + \gamma \min_{i=1,2} Q_{\theta_i}(s', a') - \alpha \log(\pi_{\psi}(|s'|)), \quad (13)
\]

\[
J(\theta_i) = \frac{1}{N} \| \hat{y} - Q_{\theta_i}(s, a) \|^2_2, \quad (14)
\]

\[
\theta_i \leftarrow \theta_i - \eta \nabla_{\theta_i} J(\theta_i), \quad (15)
\]

where \((s, a, r, s')_{i=1}^{N}\) is the mini-batch of transitions sampled from the experience replay buffer and \(N\) being the mini-batch size, \(\theta_i\) are the parameters corresponding to the \(i\)th Q-network, \(\alpha\) is the entropy regularization term, and \(\| \cdot \|_2\) is the L2 norm. Note that we denote state and action vectors by \( s \) and \( a \), respectively, while \( s' \) represents the batch of state and action vectors, respectively.

Similarly, the policy network takes state vectors from the environment as inputs and produces numerical action vectors. The loss for the policy network in the SAC algorithm is expressed by:

\[
J(\psi) = \frac{1}{N} \sum_{i=1}^{N} \alpha \log \pi_{\psi}(\hat{a}_i|s_i) - \min_{j=1,2} Q_{\theta_j}(s_i, \hat{a}_i)|_{a \sim \pi_{\psi}(.|s)} . \quad (16)
\]

Then, the policy gradient \( \nabla_{\psi} J(\psi) \) is computed by the stochastic policy gradient algorithm [19], which does not have an explicit form, and parameters are updated through gradient ascent:

\[
\psi \leftarrow \psi + \eta \nabla_{\psi} J(\psi). \quad (17)
\]

Lastly, the entropy regularization term \( \alpha \), a hyper-parameter, allows controlling the exploration, with higher \( \alpha \) corresponding to more exploration, i.e., trying different actions, outcomes of which are not well-known by the agent. Moreover, a DRL algorithm with a deterministic policy could also be employed, however, it requires an exploration noise sampled from a random variable added to actions, independent of the policy and action selection. In contrast, entropy regularization in SAC diversifies the agent’s action selection by considering the policy’s current knowledge, i.e., (16). Therefore, we believe that an actor-critic algorithm with a stochastic policy can solve the challenging transmit beamforming and phase shifts design more effectively.

### C. Construction of the Environment

In RL, environments are usually distinguished into episodic and non-episodic tasks. An episode can terminate when a terminal condition is met, e.g., the fall of a walking humanoid or reaching a pre-specified number of time steps. Non-episodic (continuing) tasks, in contrast, do not have a certain ending point, and the agent roams the environment on an infinite horizon. Since the BS performs beamforming and configures RIS elements continuously, it is evident that this is a continuing task by nature. While it is possible to specify a set of terminal conditions such as a sum rate threshold or a maximum number of steps as in [9], it might introduce bias and mislead the learning agents [22]. Therefore, we adopt the continuing task framework to develop our approach.

1) **Action**: The policy network outputs the flattened concatenation of \( G \) and \( \phi \) as the action vector. However, neural networks cannot process complex numbers; therefore, the action network produces the real and imaginary parts separately and constructs \( G \) and \( \phi \). To satisfy the transmit power and phase domain constraints, i.e., (7) and (9), the agent normalizes the output of actions. Consequently, an action vector consists of \( 2MK + 2L \) elements.

2) **State**: The state vector consists of transmission and reception powers for each user, the previous action, and the cascaded channel matrices \( D_k \) or their estimates (\( \hat{D}_k \)) for \( k = 1, \ldots, K \), depending on whether the BS has perfect or imperfect CSI. Similarly, these matrices are flattened, and the number of elements is doubled due to the real and imaginary parts except for powers. We consider the transmission powers allocated to each data stream at the BS and the reception power at each user. Consequently, we obtain \( 2K \) power-related entries. \( 2KLM \) entries come from the cascaded channel estimates for each user, and \( 2MK + 2L \) entries come from the previous action vector, resulting in a \( 2KLM + 2MK + 2L + 2K \)-dimensional state vector. Furthermore, the correlation between state dimensions degrades the performance of learning RL.
agents [22]. Hence, we whiten state vectors after each environment step. Lastly, the initial state of the training still requires the action in the previous step. Thus, we initialize \( G \) as an identity matrix and \( \phi \) as a vector of ones to constitute the initial environment state.

3) Reward: At every time step, the reward is determined by the sum downlink rate expressed by either (6) or (8), depending on the considered objective function.

IV. METHODOLOGY

A. Adapting the Deep Reinforcement Learning Framework to Non-Episodic Tasks

While \( \gamma < 1 \) prioritizes short-term rewards, the agent must be equally concerned with instantaneous and future rewards since the reward (sum rate) should always be kept maximum [22]. Therefore, we set \( \gamma = 1 \). Moreover, agents must carefully remember past actions’ outcomes to compute future actions, which cannot be achieved by learning only from instant rewards [22]. To solve this, the DRL framework should be adapted to non-episodic tasks by considering the average reward. Therefore, the reward at the current step used to train the agent is modified as follows:

\[
\tilde{r} \equiv r - \bar{r},
\]

where \( r \) is the instantaneous reward computed by the environment at the current state and \( \bar{r} \) is the average of the rewards collected until the current state. An alternative explanation for the average reward is that recall the definition of \( r + Q_\theta(s, a) \), the estimate of the value \( Q_\theta(s, a) \) when \( \gamma = 1 \), which corresponds to the rewards that the agent will collect until the terminal state is observed. However, there is no terminal state in continuing tasks; thus, the sum of collected rewards could go to infinity. The average reward overcomes this by restraining the estimation value. Hence, the agent should learn only from \( \tilde{r} \) instead of \( r \) or \( \bar{r} \).

B. Maximizing the True Sum Rate Under the Mismatch Environment Model

To maximize (6) by considering the objective (8), we leverage a recent work proposed for the exploration of continuous action spaces, the Deep Intrinsically Motivated Exploration (DISCOVER) algorithm [23]. Motivated by the animal psychological systems, DISCOVER utilizes a separate explorer network \( \xi_\omega \) with parameters \( \omega \) that represents a deterministic exploration policy, the objective of which is to perturb the actions selected by the policy so that the prediction error by the Q-network is constantly maximized. Consequently, this leads agents to state-action spaces where Q-value prediction is difficult, allowing them to correct the prediction error of unknown or less selected actions.

However, we do not directly use the DISCOVER algorithm as the SAC agent already explores the action space by utilizing a stochastic policy with the entropy parameter, i.e., the term \( \alpha \) in (13) and (16). Rather, we slightly modify the DISCOVER algorithm such that the explorer network predicts \( \beta(\phi_l) \) for \( l = 1, \ldots, L \) on the observed states:

\[
\xi_\omega(s) = \left[ \begin{array}{c} 1 & \ldots & 1 & \beta_1 & \beta_2 & \ldots & \beta_L & \beta_L \end{array} \right]^T, \tag{19}
\]

where \( \beta_l \in [\beta_{\text{min}}, 1] \) and the number of ones in the latter equation is the number of elements included by \( G \) to the action vector since the transmit beamforming produced by the agent does not affect the RIS reflection loss in the phase-dependent amplitude model. In addition, there are two entries for each \( \beta_l \) to scale both the real and imaginary parts of \( \phi \). Thus, the prediction of \( \xi_\omega(s) \) is used to perturb only the phase part of the actions:

\[
a_\beta = a \odot \lambda \cdot \xi_\omega(s) \implies \hat{\phi}_\beta \triangleq [\beta_1 e^{j\phi_1} \ldots \beta_L e^{j\phi_L}]^T, \tag{20}
\]

where \( \odot \) is the Hadamard product and the \( \lambda \in (0, 1] \) parameter is employed to restrict the explorer network not to perturb the actions chosen by the actor detrimentally, similar to DISCOVER. The environment takes \( a_\beta \) from the agent and computes the next state and reward with respect to \( a_\beta \).

Therefore, for every update step, the perturbed actions are sampled from the experience replay buffer instead of the raw ones produced by the policy network. Accordingly, the losses for the Q- and actor networks are modified as follows:

\[
\bar{\nabla}_\beta J_\beta(\phi_l) \triangleq \text{min}_{i=1,2} ||\hat{\beta} - Q_\theta_l(s, a_\beta)||_2^2,
\]

\[
J_\beta(\theta_l) \triangleq J_\beta(\psi) \triangleq \frac{1}{N} \sum_{j=1}^N \alpha \log \pi_\phi(\hat{a}_j|s_j) - \text{min}_{j=1,2} Q_\theta_l(s_j, \hat{a}_j), \tag{23}
\]

where \( \hat{a}_\beta, i = \beta_i \odot \lambda \cdot \xi_\omega \). Notice that a target explorer network also perturbs the next action \( a' \), as per DISCOVER. Also, we use \( \gamma = 1 \) and average reward in (21). Adhering to DISCOVER, the explorer network is optimized such that the sum of absolute TD errors corresponding to both Q-networks is maximized:

\[
\bar{\delta}_\beta(s, a \odot \lambda \cdot \xi_\omega(s)) \triangleq \frac{1}{N} ||\hat{\beta} - Q_\theta_l(s, a \odot \lambda \cdot \xi_\omega(s))||_2^2, \tag{24}
\]

\[
J(\omega) = \bar{\delta}_\beta(s, a \odot \lambda \cdot \xi_\omega(s)) + \bar{\delta}_\beta(s, a \odot \lambda \cdot \xi_\omega(s)). \tag{25}
\]

The deterministic exploration policy gradient is then computed by the Deterministic Policy Gradient algorithm [24] and used to perform gradient ascent over the TD errors defined by (24):

\[
\nabla_\omega J(\omega) = \sum_{i=1}^2 \mathbb{E}[\nabla_\omega \xi_\omega(s)] \nabla_\omega J(\omega), \tag{26}
\]

\[
\omega \leftarrow \omega + \eta \nabla_\omega J(\omega). \tag{27}
\]

This forms our framework to solve the downlink RIS-aided MU-MISO system under the phase-dependent amplitude model. Overall, the explorer network predicts \( \beta(\phi_l) \) and scales the actions selected by the policy by its estimation \( \beta_l \), and the scaled action is fed to the environment. Notice that the reward
Thus, the agent observes the effect of the explorer network’s value predictions and jointly learning what action the policy where they experience significant uncertainty about the Q-model by considering the current knowledge of the Q-networks considering the artificial \( \hat{\beta} \). Since now

By maximizing the TD error, the exploration policy further

\[ \beta \]

A. Experimental setup

To test the effectiveness of the introduced approach, we compare \( \beta \)-Space Exploration against a vanilla SAC agent under two scenarios: the golden standard and the mismatch. In the golden standard case, the agent knows the true environment model and is trained by computing rewards according to (6). In the mismatch case, however, the BS tries to solve (9) and computes its rewards using (8) in a mismatch environment. Similarly, the SAC agent combined with \( \beta \)-Space Exploration tries to solve (9) under the mismatch scenario. In our experiments, we report the instant rewards, i.e., the term \( r \) in (18), provided by the true environment according to (6) for all three scenarios. We performed our experiments by fixing \( M = K = 4 \), and using \( \beta_{\text{min}} = \{0.3, 0.6\} \) and \( L = \{16, 64\} \). We performed these convergence experiments by fixing \( P_t = 30 \) dBm. We also tested the convergence points for three scenarios under \( P_t = \{5, 10, 15, 20, 25, 30\} \) dBm.

The SAC algorithm implementation follows the structure outlined in the original paper [19]. We performed an extensive hyper-parameter tuning starting from the hyper-parameter setting provided by [19]. The tuned hyper-parameter setting is outlined in Table I along with the chosen environment parameter values. Different from [19], we also include entropy tuning, as shown by the authors of SAC in [27] to improve the algorithm’s overall performance. Entropy tuning is an automated gradient-based temperature tuning method that adjusts the entropy term \( \alpha \) over the visited states to match a target

| Hyper-Parameter | Value |
|-----------------|-------|
| \# of hidden layers\(^1\) | 2 |
| \# units in each hidden layer\(^1\) | 256 |
| Hidden layers activation\(^1\) | ReLU |
| Final layer activation (Q-networks) | Linear |
| Final layer activation (actor, explorer) | tanh |
| Learning rate \( \eta \)\(^−1\) | 10\(^{−3}\) |
| Weight decay\(^1\) | None |
| Weight initialization\(^1\) | Xavier uniform [25] |
| Bias initialization\(^1\) | constant |
| Optimizer\(^1\) | Adam [26] |
| Total time steps per training | 20000 |
| Experience replay buffer size | 20000 |
| Experience replay sampling method | uniform |
| Mini-batch size | 16 |
| Discount term \( \gamma \) | 1 |
| Learning rate for target networks \( \tau \)\(^†\) | 10\(^{−3}\) |
| Network update interval\(^†\) after each environment step |
| Initial \( \alpha \) | 0.2 |
| Entropy target \( \beta \)-Space Exploration \( \lambda \) | 0.3 |

\( \mu \)

\( \nu \)

Channel noise variance \( \sigma_n \)

AWGN channel variance \( \sigma_w \)

Channel matrix initialization (Rayleigh)

\( \mathcal{CN}(0, 1) \)

\(^1\) Applies to all networks

\(^†\) Environment hyper-parameter
TABLE II: Average of last 1000 instant rewards achieved by the SAC agents, computed according to (6), over 10 trials of different transmission powers are provided in Fig. 1d. From our standards [28]. Similarly, the converged sum rates across different transmission powers are provided in Fig. 1d. From our standards [28].

B. Results and Conclusions

The learning curves in Fig. 1 depict the instant rewards achieved by the agents computed according to (6), averaged over ten random seeds. Moreover, we report the converged sum rates in Table II, being the average of the last 1000 instant rewards over ten trials as per the DRL benchmarking standards [28]. Similarly, the converged sum rates across different transmission powers are provided in Fig. 1d. From our evaluation results, we infer that \(\beta\)-Space Exploration attains near-optimal results in all of the settings tested. Specifically, when \(\beta_{\text{min}} = 0.3\), the SAC agent under the mismatch environment shows considerably worse performance than the golden standard, and the resulting performance approaches the golden standard when \(\beta_{\text{min}}\) is increased to 0.6. This is expected since the interval for possible RIS loss factors shrinks as \(\beta(\varphi_t) \in [\beta_{\text{min}}, 1]\). However, our method is not affected by the \(\beta_{\text{min}}\) value and exhibits a robust performance, achieving high sum downlink rates slightly lower than the golden standards for each value of \(\beta_{\text{min}}\). Furthermore, our algorithm regards no issues with the convergence rate, that is, learning curves are practically parallel to the golden standard, which implies that \(\beta\)-Space Exploration can learn how the actions selected by the policy affect the loss in the RIS reflections, i.e., resulting \(\beta(\varphi_t)\), in a negligible amount of time compared to the total training duration, and maximize the sum rate accordingly.

When the number of RIS elements \(L\) is increased to 64, the sum rate acquired by the golden standard increases significantly due to the higher number of RIS elements whose characteristics are known by the BS agent. At the same time, the BS agent learning under mismatch attains notably poor per-

---

**TABLE II: Average of last 1000 instant rewards achieved by the SAC agents, computed according to (6), over 10 trials of 20000 time steps. ± captures a 95% confidence interval over the trials. The performance increase denotes the percentage of mean sum rate improvement obtained by \(\beta\)-Space Exploration over the vanilla SAC agent in the mismatch environment with respect to the difference between the golden standard and mismatch scenarios.**

| Setting                  | Golden Standard | Mismatch | \(\beta\)-Space Exploration | Performance Increase |
|--------------------------|-----------------|----------|-----------------------------|----------------------|
| \(\beta_{\text{min}} = 0.3, P_t = 30\) dBm, \(K = 4, M = 4, L = 16\) | 8.16 ± 0.92     | 6.37 ± 0.76 | 7.88 ± 0.56                 | 84%                  |
| \(\beta_{\text{min}} = 0.6, P_t = 30\) dBm, \(K = 4, M = 4, L = 16\) | 8.17 ± 0.83     | 7.43 ± 0.80 | 7.92 ± 0.70                 | 66%                  |
| \(\beta_{\text{min}} = 0.6, P_t = 30\) dBm, \(K = 4, M = 4, L = 64\) | 8.71 ± 0.84     | 7.00 ± 0.57 | 8.70 ± 0.94                 | 99%                  |
| \(P_t = 5\) dBm         | 5.17 ± 0.41     | 4.72 ± 0.38 | 5.10 ± 0.36                 | 84%                  |
| \(P_t = 10\) dBm        | 6.89 ± 0.53     | 6.08 ± 0.56 | 6.69 ± 0.52                 | 75%                  |
| \(P_t = 15\) dBm        | 8.44 ± 0.72     | 7.46 ± 0.85 | 8.34 ± 0.91                 | 90%                  |
| \(P_t = 20\) dBm        | 8.94 ± 0.68     | 7.87 ± 0.99 | 8.74 ± 0.90                 | 81%                  |
| \(P_t = 25\) dBm        | 9.30 ± 0.97     | 8.19 ± 0.85 | 9.22 ± 0.72                 | 93%                  |
| \(P_t = 30\) dBm        | 9.40 ± 0.95     | 8.55 ± 0.93 | 9.29 ± 0.82                 | 87%                  |

---

Fig. 1: Learning curves for the tested settings. The shaded region represents a 95% confidence interval over 10 random seeds. A sliding window of size 25 smooths the curves for visual clarity.

value rather than an initial one. We also linearly decay the exploration regularization term \(\lambda\) such that it becomes zero at the end of the training since large \(\lambda\) values in the final steps of the training may degrade the performance of a SAC agent that learned to control the environment sufficiently well. Furthermore, our simulations follow the well-known DRL benchmarking standards [28], that is, each experiment runs over ten random seeds for a fair comparison with the baselines since the variance due to the environment stochasticity or stochasticity in the learning, e.g., the initial state of the environment, random weight initialization, is a significant concern in DRL [28]. Precise experimental setup and implementation can be found in the code of our repository.

formance due to additional RIS elements whose characteristics are misspecified. In contrast, our method can successfully reach the level where the golden standard converges after a slight delay, with the highest performance increase recorded, as shown in Table II. Therefore, we deduce that our algorithm is also generalizable to a variety of RIS settings. Finally, the performance gain offered by β-Space Exploration over the vanilla SAC agent under the mismatch environment is consistent across the selected interval of transmission power. Additionally, the resulting confidence intervals of our algorithm are usually tighter than the ones corresponding to the golden standard, suggesting that our framework improves credibly over the baseline due to the structure of the introduced method rather than unintended consequences or any exhaustive hyper-parameter tuning. Consequently, our empirical studies imply that the introduced framework is a preferable way of overcoming the challenges introduced by the hardware impairments in the RIS since it can achieve near-golden standard performance under imperfect CSI and phase-dependent amplitude model [1], which is realistic and generalizable to real-world problems.

REFERENCES

[1] S. Abeywickrama, R. Zhang, Q. Wu, and C. Yuen, “Intelligent reflecting surface: Practical phase shift model and beamforming optimization,” IEEE Transactions on Communications, vol. 68, no. 9, pp. 5849–5863, 2020.

[2] C. Ozturk, M. F. Keskin, H. Wymeersch, and S. Gezici, “On the impact of hardware impairments on RIS-aided localization,” in ICC 2022 - IEEE International Conference on Communications, 2022, pp. 2846–2851.

[3] J. Wang, W. Tang, Y. Han, S. Jin, X. Li, C.-K. Wen, Q. Cheng, and T. J. Cui, “Interplay between RIS and AI in wireless communications: Fundamentals, architectures, applications, and open research problems,” IEEE Journal on Selected Areas in Communications, vol. 39, no. 8, pp. 2271–2288, 2021.

[4] Z. Yang, Y. Liu, Y. Chen, and N. Al-Dhahir, “Machine learning for user partitioning and phase shifters design in RIS-aided NOMA networks,” IEEE Transactions on Communications, vol. 69, no. 11, pp. 7414–7428, 2021.

[5] Q. Zhang, W. Saad, and M. Bennis, “Millimeter wave communications with an intelligent reflector: Performance optimization and distributional reinforcement learning,” IEEE Transactions on Wireless Communications, vol. 21, no. 3, pp. 1836–1850, 2022.

[6] M. Samir, M. Elhattab, C. Assi, S. Sharafeddine, and A. Ghrayeb, “Optimizing age of information through aerial reconfigurable intelligent surfaces: A deep reinforcement learning approach,” IEEE Transactions on Vehicular Technology, vol. 70, no. 4, pp. 3978–3983, 2021.

[7] A. Al-Hilo, M. Samir, M. Elhattab, C. Assi, and S. Sharafeddine, “Reconfigurable intelligent surface enabled vehicular communication: Joint user scheduling and passive beamforming,” IEEE Transactions on Vehicular Technology, vol. 71, no. 3, pp. 2333–2345, 2022.

[8] X. Liu, Y. Liu, and Y. Chen, “Machine learning empowered trajectory and passive beamforming design in uav-ris wireless networks,” IEEE Journal on Selected Areas in Communications, vol. 39, no. 7, pp. 2042–2055, 2021.

[9] 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, 2020.

[10] K. Feng, Q. Wang, X. Li, and C.-K. Wen, “Deep reinforcement learning based intelligent reflecting surface optimization for MISO communications systems,” IEEE Wireless Communications Letters, vol. 9, no. 5, pp. 745–749, 2020.

[11] X. Liu, Y. Liu, Y. Chen, and H. V. Poor, “RIS enhanced massive non-orthogonal multiple access networks: Deployment and passive beamforming design,” IEEE Journal on Selected Areas in Communications, vol. 39, no. 4, pp. 1057–1071, 2021.