Accelerated Deep Reinforcement Learning for Uplink Power Control in a Dynamic Cell-Free Massive MIMO Network

Charmae Franchesca Mendoza, *Graduate Student Member, IEEE*, Megumi Kaneko, *Senior Member, IEEE*, Markus Rupp, *Fellow, IEEE*, and Stefan Schwarz, *Senior Member, IEEE*

***Abstract*—We investigate the deep reinforcement learning (DRL) framework for uplink power control in a cell-free massive multiple-input, multiple-output (MIMO) network. Although DRL does not require prior sets of training data as opposed to supervised or unsupervised machine learning approaches, existing methods suffer from substantial convergence time, which is prohibitive in a highly dynamic or large-scale mobile environment. To address this crucial issue, we propose a DRL framework that capitalizes on prioritized sampling to speed up the learning process, thereby enabling rapid adaptation to the variations of the wireless environment. The proposed method is not only tailored to user mobility, but also to network variations due to device activation and deactivation. Numerical results demonstrate the effectiveness of our proposed algorithm, as it exhibits near-optimal performance, outperforming the benchmark schemes in terms of the guaranteed rate and total power consumption, with much faster convergence.**

***Index Terms*—Deep reinforcement learning, prioritized sampling, cell-free massive MIMO, power control.**

I. Introduction

CELL-FREE massive multiple-input, multiple-output (MIMO) combines the benefits of massive MIMO and distributed systems [1], and is therefore considered a key enabler for 6G [2]. It consists of a large number of distributed access points (APs) or antennas jointly serving a smaller number of user equipments (UEs) using the same time-frequency resources. All APs are controlled by a centralized processing unit (CPU) for coordination and synchronization.

Power control in cell-free massive MIMO has been widely studied, as interference can severely degrade the network performance. However, most of the existing works, such as [1], [3], [4], utilize traditional optimization techniques that require the channel state information (CSI), which may be challenging to obtain in practice. Machine learning has recently gained popularity with its ability to solve various resource allocation problems with much lower computational complexity while achieving good performance [5], [6]. In particular, deep reinforcement learning (DRL) has garnered much interests, as it does not require any prior knowledge in terms of training datasets that are not always available, since it relies on reward feedbacks that are usually inherent in target wireless systems [7]. In [8], it was employed for power assignment in uplink cell-free massive MIMO considering both sum rate and user fairness. A DRL-based algorithm for downlink power allocation was presented in [9] assuming different optimization objectives, shown to outperform deep learning and conventional optimization techniques. However, these works only considered a static scenario that is rather unrealistic and does not take advantage of the full potential of DRL. In [10], DRL was utilized for uplink power control to maximize the network sum rate while satisfying individual user rate constraints. While both static and mobile users were considered, it did not tackle one of the challenges of DRL, which is convergence speed. Specifically, the DRL system must be re-trained whenever the wireless environment changes. A slow convergence implies that by the time this re-training is finished, the environment is likely to have changed again, making decisions to be outdated. To speed up the training process, prioritized sampling was used in [11], where, in contrast to uniform sampling, certain experiences that are regarded as more important are selected more often. This technique was utilized in [12] to solve a power allocation problem for balancing SINR maximization and power minimization. However, their investigation only considered a fully static, point-to-point scenario.

In this letter, we design a DRL-based framework for uplink power control in cell-free massive MIMO, which, unlike previous studies, not only considers both static and mobile users, but also another layer of dynamicity pertaining to the activation and deactivation of devices fluctuating over time, allowing us to integrate more realistic scenarios. For instance, Internet of Things (IoT) devices have limited battery power, and thus, go into a sleep mode to prolong their battery life. However, existing algorithms require precisely knowing the UEs' ON/OFF patterns that are difficult to predict, as they depend on the battery state of the individual devices. This motivates us to design a method that is capable of learning such dynamic behaviors on the go. Our proposed model-free DRL method is specially crafted such that the power control decision is solely based on the user rate feedback, hence neither the CSI nor the device activation pattern is required to be known in advance. Additionally, we augment the vanilla deep deterministic policy gradient (DDPG) DRL algorithm [13] with prioritized sampling to ensure that the system is able to quickly adapt to the changes in the wireless environment, as in the case of a live network. The prioritization

Manuscript received 20 February 2024; accepted 3 April 2024. Date of publication 12 April 2024; date of current version 11 June 2024. This work was supported in part by the NII-TU Wien MoU Grant funded by NII; in part by the Grants-in-Aid for Scientific Research from the Ministry of Education, Science, Sports, and Culture of Japan under Grant 22KK0156 and Grant 20H00592; in part by the Austrian Federal Ministry for Labour and Economy; in part by the National Foundation for Research, Technology and Development; and in part by the Christian Doppler Research Association. The associate editor coordinating the review of this article and approving it for publication was J. He. *(Corresponding author: Charmae Franchesca Mendoza.)*

Charmae Franchesca Mendoza, Markus Rupp, and Stefan Schwarz are with the Christian Doppler Laboratory for Digital Twin Assisted AI for Sustainable Radio Access Networks, Institute of Telecommunications, Technische Universität Wien, 1040 Vienna, Austria (e-mail: charmae.mendoza@tuwien.ac.at; markus.rupp@tuwien.ac.at; stefan.schwarz@tuwien.ac.at).

Megumi Kaneko is with the Information Systems Architecture Science Research Division, National Institute of Informatics, Tokyo 101-8430, Japan (e-mail: megkaneko@nii.ac.jp).

Digital Object Identifier 10.1109/LWC.2024.3387839

is dictated by the temporal-difference (TD) error [7], which is automatically calculated when updating DDPG. Therefore, it does not incur additional computational complexity. Our main contributions are summarized as follows.

1) We first provide the mathematical formulation of the target uplink power control problem, aiming at maximizing the guaranteed rate of a cell-free massive MIMO network.
2) We describe the proposed DDPG-based method, which, unlike existing schemes, is specifically designed to adapt to various dynamics of the wireless environment in an online fashion, such as user mobility and variations of UE activation patterns over time. This is realized by capitalizing on TD-error-based prioritized sampling.
3) The effectiveness of the proposed method is shown through simulation results under different scenarios, where it outperforms the benchmark schemes in terms of convergence speed, guaranteed rate and power consumption.

II. System Model

We consider an uplink cell-free massive MIMO network with M single-antenna APs and K single-antenna UEs, such that $M \gg K$. The set of all APs is denoted by $\mathcal{M} = \{1, \ldots, M\}$, and the set of all UEs by $\mathcal{K} = \{1, \ldots, K\}$. The channel between AP m and UE k is given by $h_{k,m} = \sqrt{g_{k,m}}\widetilde{h}_{k,m}$. Large-scale fading coefficient $g_{k,m}$ follows a distance-dependent path loss model with lognormally-distributed shadow fading, and small-scale fading coefficient $\widetilde{h}_{k,m} \sim \mathcal{CN}(0,1)$ is independent and identically distributed (i.i.d.). We consider the block-fading channel model. Each coherence block contains $\tau_c = \tau_p + \tau_u + \tau_d$ samples, of which τ_p are for uplink training, τ_u are for uplink data, and τ_d are for downlink data transmission.

All UEs first transmit their pilot sequences simultaneously. The pilot sequence of UE k is denoted by $\boldsymbol{\phi}_k \in \mathbb{C}^{\tau_p \times 1}$. The received $(\tau_p \times 1)$-pilot signal at AP m is,

$$\mathbf{y}_m = \sum_{k\in\mathcal{K}} \sqrt{\tau_p \rho_p} h_{k,m} \boldsymbol{\phi}_k + \mathbf{n}_m, \tag{1}$$

where ρ_p is the pilot transmit power, and $\mathbf{n}_m$ is the noise vector with i.i.d. elements following $\mathcal{CN}(0, \sigma_n^2)$. AP m estimates the channel by projecting the received pilot signal onto $\boldsymbol{\phi}_k^H$,

$$\hat{y}_{k,m} = \boldsymbol{\phi}_k^H \mathbf{y}_m = \sum_{k'\in\mathcal{V}_k} \sqrt{\tau_p \rho_p} h_{k',m} + \boldsymbol{\phi}_k^H \mathbf{n}_m, \tag{2}$$

where $\mathcal{V}_k$ denotes the set of users utilizing the same pilot sequence $\boldsymbol{\phi}_k$. The minimum mean-squared error (MMSE) channel estimate is then [1], [10],

$$\hat{h}_{k,m} = \frac{\sqrt{\tau_p \rho_p} g_{k,m}}{\sum_{k'\in\mathcal{V}_k} \tau_p \rho_p g_{k',m} + \sigma_n^2} \hat{y}_{k,m} = c_{k,m}\, \hat{y}_{k,m}, \tag{3}$$

with $\hat{h}_{k,m} \sim \mathcal{CN}(0,\, V_{k,m})$, $V_{k,m} = \frac{\tau_p \rho_p g_{k,m}^2}{\sum_{k'\in\mathcal{V}_k} \tau_p \rho_p g_{k',m} + \sigma_n^2}$.

During uplink data transmission, all UEs transmit their data simultaneously. UE k sends its symbol x_k, with $\mathbb{E}\{|x_k|^2\} = 1$. The received signal at AP m is,

$$y_m^{(u)} = \sum_{k\in\mathcal{K}} h_{k,m} \sqrt{\rho_k} x_k + n_m^{(u)}, \tag{4}$$

where ρ_k is the uplink transmit power of UE k, and $n_m^{(u)}$ is the noise at AP m. Each AP utilizes its local channel estimate $\hat{h}_{k,m}$ to get $\hat{h}_{k,m}^* y_m^{(u)}$, which is then sent to the CPU for data detection. The received signal at the CPU is,

$$z_k^{(u)} = \underbrace{\sum_{m\in\mathcal{M}} \hat{h}_{k,m}^* h_{k,m} \sqrt{\rho_k} x_k}_{\text{desired signal}} + \underbrace{\sum_{m\in\mathcal{M}} \sum_{\substack{k'\in\mathcal{K} \\ k'\neq k}} \hat{h}_{k,m}^* h_{k',m} \sqrt{\rho_{k'}} x_{k'}}_{\text{inter-user interference}} + \underbrace{\sum_{m\in\mathcal{M}} \hat{h}_{k,m}^* n_m^{(u)}}_{\text{noise}}. \tag{5}$$

The signal-to-interference-plus-noise ratio (SINR) of UE k in (6), shown at the bottom of the page, is obtained through the power of the three terms in (5), as in [1], [10]. The rate of UE k in bits per second (bps) is,

$$u_k = B\left(1 - \frac{\tau_p}{\tau_c}\right) \log_2(1 + \text{SINR}_k), \tag{7}$$

where B is the system bandwidth in Hertz (Hz).

III. Problem Formulation

We hereby focus on determining the uplink transmit power for all UEs, with the goal of maximizing the minimum user rate over the cell-free massive MIMO network. The optimization problem is formulated as,

$$\max_{\{\rho_k\}} \quad \min_{k\in\mathcal{K}} u_k \tag{8a}$$

$$\text{subject to} \quad 0 \leq \rho_k \leq \rho_{\max}, \forall k \in \mathcal{K}, \tag{8b}$$

where (8a) represents our objective, and (8b) defines the valid range for the power values, with $\rho_{\max}$ being the maximum transmit power. The problem is equivalent to maximizing the minimum user SINR, which, in principle, can be solved using conventional optimization algorithms, including those described in [1], [14]. However, as argued in Section I, such approaches require the exact knowledge of active and deactivated devices, which is impractical in the case of IoT applications, especially if these changes occur rapidly. Therefore, we next design a DRL scheme that only relies on the user performance.

IV. Proposed DRL-Based Uplink Power Control With Prioritized Sampling

In this section, we present our proposed DRL framework for uplink power control, designed to cope with device activation and deactivation, and to accelerate convergence. The CPU serves as the DRL agent, and the environment consists of the users and APs, as depicted in Fig. 1.

$$\text{SINR}_k = \frac{\rho_k \left(\sum_{m\in\mathcal{M}} V_{k,m}\right)^2 + \sum_{m\in\mathcal{M}} \rho_k V_{k,m} g_{k,m}}{\tau_p \sum_{k'\in\mathcal{V}_k} \rho_{k'} \rho_p \left(\sum_{m\in\mathcal{M}} c_{k,m} g_{k',m}\right)^2 + \sum_{\substack{k'\in\mathcal{K} \\ k'\neq k}} \rho_{k'} \sum_{m\in\mathcal{M}} g_{k',m} V_{k,m} + \sum_{m\in\mathcal{M}} V_{k,m} \sigma_n^2} \tag{6}$$

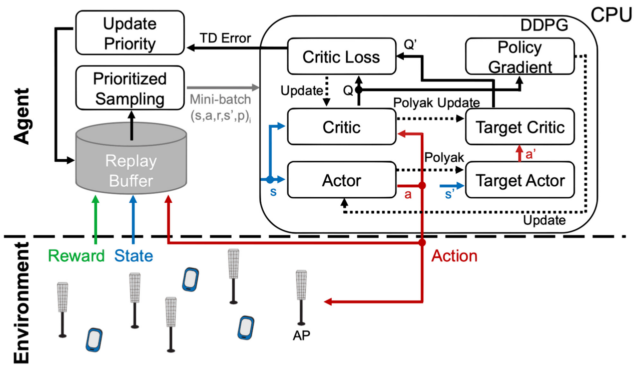


Fig. 1. Agent-environment scenario of the DDPG-based framework with prioritized sampling.

A. DRL Components

1) State: The actual ON/OFF state of UE k at time t is denoted by $d_k^{(t)} \in \{0,1\}$. The UEs may switch from active to inactive mode (and vice versa), which we refer to as UE toggling, with parameters T_{tog} and K_{tog}. The UE toggling period T_{tog} specifies the number of episodes over which the ON/OFF state of the UEs is assumed to be constant. The number of UEs that switch every T_{tog} is indicated by K_{tog}. The activation pattern is, however, unknown at the CPU. It estimates this information based on whether the APs have received any signal from a specific device. It considers the UE inactive if no associated signal, such as UE rate feedback, has been detected. This is reflected in the state vector describing the environment,

$$\mathbf{s}^{(t)} = \left[b_1^{(t)}, \ldots, b_K^{(t)}, u_1^{(t-1)}, \ldots, u_K^{(t-1)}\right], \tag{9}$$

where $b_k^{(t)} \in \{0,1\}$ indicates whether UE k is estimated to be active or not at time t, such that $b_k^{(t)} = \hat{d}_k^{(t)}$, and $u_k^{(t-1)}$ is the rate of UE k at time $t-1$.

2) Action: Based on the current state observation, the agent decides the uplink transmit power of all K UEs denoted by,

$$\mathbf{a}^{(t)} = \left[\rho_1^{(t)}, \ldots, \rho_K^{(t)}\right]. \tag{10}$$

It assigns zero power to UEs estimated to be inactive, while allocating non-zero power within range $\xi \le \rho_k^{(t)} \le \rho_{\max}$ to the active users, with $0 < \xi \ll 1$.

3) Reward: The agent receives a corresponding reward,

$$r^{(t+1)} = \min_{k' \in \mathcal{K}_{\text{ON}}^{(t)}} u_{k'}^{(t)}, \tag{11}$$

where $\mathcal{K}_{\text{ON}}^{(t)} = \{k \in \mathcal{K} | b_k^{(t)} = 1\}$ is the set of active UEs.

B. Proposed Algorithm

Algorithm 1 outlines the procedure of the proposed method, which we detail below. We leverage the uniform sampling-based DDPG DRL algorithm in [13], as it was shown to best handle the continuous state and action spaces at stake.

Step 1: We first initialize the four deep neural networks (DNNs) of DDPG: primary actor (θ^{μ}), primary critic (θ^{Q}), target actor ($\theta^{\mu'}$), and target critic ($\theta^{Q'}$). We also initialize the prioritized sampling parameters, and the noise for action exploration (Lines 1 to 4).

Step 2: The CPU observes the current ON/OFF UE states and user performance. It decides the uplink transmit powers,

Algorithm 1 DDPG with pER for Power Control

```
1:  Initialize the primary actor $\mu(\mathbf{s}|\theta^{\mu})$ and critic $Q(\mathbf{s}, \mathbf{a}|\theta^{Q})$ with
    weights $\theta^{\mu}$ and $\theta^{Q}$. Initialize the target networks $\mu'$ and $Q'$ with
    weights $\theta^{\mu'} \leftarrow \theta^{\mu}$ and $\theta^{Q'} \leftarrow \theta^{Q}$.
2:  Initialize the pER parameters $\beta \leftarrow \beta_{\text{start}}$ and $p_{\max} \leftarrow 0$.
3:  for episode $= 0, \ldots, E-1$ do
4:      Initialize a random process $\mathcal{N}$ for action exploration.
5:      Initialize state $\mathbf{s}^{(0)}$.
6:      for timestep $t = 0, \ldots, T-1$ do
7:          Observe the current state $\mathbf{s}^{(t)}$.
8:          Select and apply action $\mathbf{a}^{(t)} \leftarrow \mu(\mathbf{s}^{(t)}|\theta^{\mu}) + \mathcal{N}^{(t)}$.
9:          Observe the reward $r^{(t+1)}$ and next state $\mathbf{s}^{(t+1)}$.
10:         Store experience $(\mathbf{s}^{(t)}, \mathbf{a}^{(t)}, r^{(t+1)}, \mathbf{s}^{(t+1)}, p_{\max})$ in $\mathcal{B}$.
11:         Compute the probabilities (12) of the experiences.
12:         Compute their weights (13).
13:         Sample a random mini-batch of $X$ experiences from $\mathcal{B}$ based
            on the calculated probabilities.
14:         Update the critic by minimizing the loss (14).
15:         Update the priorities (15) of the samples.
16:         Update $p_{\max} \leftarrow \max_{j \in \{0,\ldots,\text{len}(\mathcal{B})\}} p_j$.
17:         Update the actor by maximizing the gradient (16).
18:         Update the target networks using Polyak averaging: $\theta^{\mu'} \leftarrow$
            $\tau_{\text{pol}}\theta^{\mu} + (1-\tau_{\text{pol}})\theta^{\mu'}$ and $\theta^{Q'} \leftarrow \tau_{\text{pol}}\theta^{Q} + (1-\tau_{\text{pol}})\theta^{Q'}$,
            with $\tau_{\text{pol}} \in (0,1)$.
19:         Update $\beta$ as in (17).
20:     end for
21: end for
```

and consequently, obtains a reward that is the guaranteed rate that we aim to maximize. It then estimates the latest activation state of the UEs for the next time step, as described in Section IV-A. If UE k' is estimated to be inactive, the corresponding elements $b_{k'}^{(t+1)}$ and $u_{k'}^{(t)}$ in $\mathbf{s}^{(t+1)}$ would be 0 (Lines 5 to 9).

Step 3: Unlike the vanilla DDPG algorithm that uniformly samples the experiences used to update its DNN parameters, we propose to augment it with prioritized sampling or prioritized experience replay (pER) [11]. When the agent saves its new experience i in its replay buffer $\mathcal{B}$, we attach a priority information p_i, forming the modified experience tuple $(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_i', p_i)$. A new experience is always assigned the current maximum priority $p_{\max}$ for it to be sampled at least once (Line 10).

Step 4: The priority values are converted to probabilities $P(i) \in [0,1]$ as,

$$P(i) = \frac{p_i^{\alpha}}{\sum_{j=0}^{\text{len}(\mathcal{B})-1} p_j^{\alpha}}, \tag{12}$$

where $\text{len}(\mathcal{B})$ denotes the current length of the buffer (Line 11). The prioritization factor $\alpha \in [0,1]$ controls how much we rely on the prioritization, with $\alpha = 0$ for uniform sampling.

Step 5: The pER mechanism introduces a distribution bias, which we correct by assigning importance-sampling weights to the samples,

$$w_i = (\text{len}(\mathcal{B}) \cdot P(i))^{-\beta}. \tag{13}$$

We define a correction parameter $\beta \in [0,1]$, such that $\beta = 0$ corresponds to the case where no correction is made. The experiences with high priority are likely to be oversampled by pER. We counter this by assigning them lower weights to lessen their impact (Line 12).

Step 6: We sample a mini-batch of X experiences based on the calculated probabilities. For each sample, we compute the TD error $\delta_i = y_{\text{Targ},i} - Q(\mathbf{s}_i, \mathbf{a}_i|\theta^Q)$. The target networks are used to calculate the updated Q-value $y^{(t)}_{\text{Targ}} = r^{(t+1)} + \gamma Q'(\mathbf{s}^{(t+1)}, \mu'(\mathbf{s}^{(t+1)}|\theta^{\mu'})|\theta^{Q'})$ based on the Bellman equation [7]. The primary critic network is updated by minimizing the weighted loss between the updated and current Q-values,

$$L_{\text{pER}} = \frac{1}{X}\sum_{i=0}^{X-1} w_i \Big(y_{\text{Targ},i} - Q(\mathbf{s}_i, \mathbf{a}_i|\theta^Q)\Big)^2, \qquad (14)$$

over the X sampled experiences using gradient descent (Lines 13 to 14).

Step 7: The priorities are calculated based on the TD errors as,

$$p_i = |\delta_i| + \epsilon, \qquad (15)$$

where $\epsilon > 0$ is a small number to avoid dividing by 0 in (12). Experiences with larger δ_i are assigned a higher priority, so that they are more likely to be sampled, resulting to more chances of minimizing the TD error. Based on the newly calculated TD errors, we update the priorities of the sampled experiences, and, subsequently, the current $p_{\max}$ (Lines 15 to 16).

Step 8: The primary actor network is updated by taking the following derivative and applying gradient ascent [13],

$$\nabla_{\theta^\mu} Q \approx \frac{1}{X}\sum_{i=0}^{X-1} \nabla_{\mathbf{a}} Q\Big(\mathbf{s}, \mathbf{a}|\theta^Q\Big)|_{\mathbf{s}=\mathbf{s_i}, \mathbf{a}=\mu(\mathbf{s_i})} \nabla_{\theta^\mu}\mu\Big(\mathbf{s}|\theta^\mu\Big)|_{\mathbf{s}=\mathbf{s_i}}. \qquad (16)$$

The target networks are updated using Polyak averaging (Lines 17 to 18).

Step 9: We anneal β during training as,

$$\beta^{(t+1)} = \beta^{(t)} + (\beta_{\text{end}} - \beta_{\text{start}}) {N_{\text{ts}}}^{-1}. \qquad (17)$$

We initialize β to β_{start}, and increase its value until it reaches β_{end} over N_{ts} time steps (Line 19).

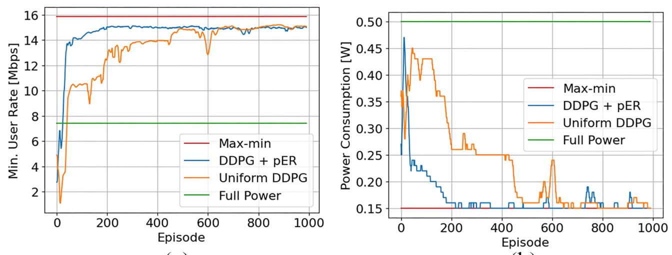
V. Numerical Evaluations

A. Simulation Scenario

We consider an uplink cell-free massive MIMO network with $M = 30$ single-antenna APs and $K = 10$ single-antenna UEs, all of which are uniformly distributed over a 500×500 m^2 area. The simulation parameters are listed in Table I. For both the actor and critic networks, we employ a fully connected DNN with two hidden layers having 64 neurons each. We use three benchmark schemes to evaluate the performance of our proposed framework: (a) *Uniform DDPG* – vanilla DDPG algorithm utilizing uniform sampling, (b) *Full power* – each UE transmits with $\rho_{\max}$, and (c) *Max-min* – Problem (8) is solved as in [1]. Moreover, we test our system considering different mobility scenarios, where we combine static and mobile users with dynamic UE activation patterns. In the latter case, the users randomly select a direction (left, right, up, or down) and a speed from 0 to 1 m/s uniformly at each time step. For our proposed framework, *DDPG + pER*, we experimented with different prioritization factor α values to determine the most appropriate one for our problem. We achieved the best performance for $\alpha = 0.5$, and thus, we use this value in the sequel.

TABLE I
Simulation Parameters

Parameter	Value
Carrier frequency	1.9 GHz
Bandwidth B	20 MHz
Path loss exponent	2
Shadow fading standard deviation	8 dB
Noise figure	9 dB
Pilot transmit power ρ_p	0.1 W
Maximum transmit power $\rho_{\max}$	0.1 W
Buffer size $\lvert\mathcal{B}\rvert$	1e6
Batch size	100
Learning rate for actor, critic	0.001
Discount rate γ	0.9
Polyak factor τ_{pol}	0.005
Prioritization factor α	0.5
pER correction parameter β_{start}, β_{end}	0.4, 1



(a) (b)

Fig. 2. Performance comparison for the fully static scenario.

B. Fully Static / No UE Toggling

We first consider a fully static scenario, where the ON/OFF states of the non-mobile users stay constant throughout the learning process. Fig. 2(a) shows the evolution of the minimum user rate. The proposed scheme converges to 15 Mbps, while *Uniform DDPG* achieves the same rate 300 episodes later. Compared to *Max-min* that provides the upper bound, our framework behaves close to optimal, with a difference of only 0.9 Mbps or 5.66%. Note that *Max-min* requires the UE activation pattern to be known in advance for solving Problem (8) with traditional optimization techniques. However, in practice, this information is not available to the CPU, and is also hard to predict. We highlight that, by estimating the ON/OFF states and relying solely on the UE rate feedback, our system is able to adapt to the current state of the environment without this knowledge, as shown in Fig. 2. The *Full power* baseline performs worst due to the increased inter-user interference.

The total transmit power consumption is depicted in Fig. 2(b). For both sampling configurations, we observe that power consumption reduces as training progresses. This suggests that the agent is able to recognize the ON/OFF patterns of UEs, allowing it to select better actions or power values as it further interacts with the environment. It is worth noting that power reduction is implicitly accounted for in the reward definition, as maximizing the guaranteed rate requires minimizing inter-user interference. With prioritized sampling, power consumption noticeably reaches that of *Max-min* faster compared to *Uniform DDPG*. The *Full power* benchmark consumes the most power.

C. Static Users With UE Toggling

We next consider the case of non-mobile UEs that switch from active to inactive mode (and vice versa) with $T_{\text{tog}} = 500$ and $K_{\text{tog}} = 0.1K$. The UE toggling then happens at episode 500, which explains the sudden "activity" around this

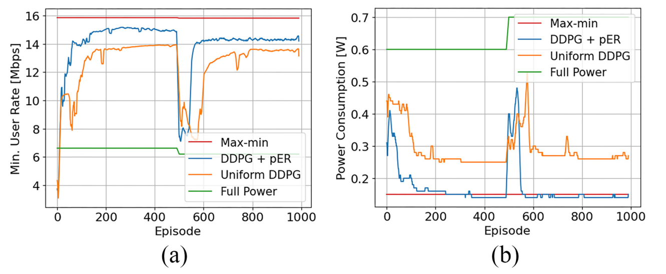


Fig. 3. Performance comparison for the static, $T_{\text{tog}} = 500$, $K_{\text{tog}} = 0.1K$ scenario.

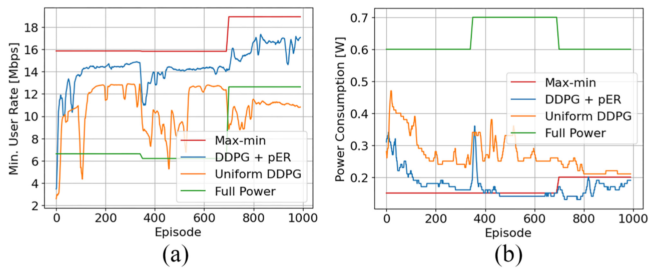


Fig. 4. Performance comparison for the static, $T_{\text{tog}} = 350$, $K_{\text{tog}} = 0.1K$ scenario.

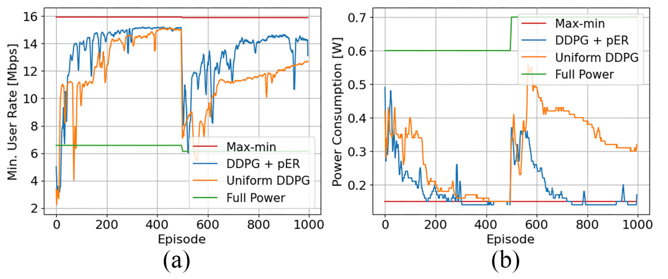


Fig. 5. Performance comparison for the mobile, $T_{\text{tog}} = 500$, $K_{\text{tog}} = 0.1K$ scenario.

area in Fig. 3. After which, we observe that the prioritization helps the DRL system to recover from the environment change faster, with the agent already finding a solution at episode 600 in Fig. 3(a). In contrast, *Uniform DDPG* is only able to do so 200 episodes later. Compared to the fully static scenario, we achieve not only accelerated convergence, but also better performance (7.4% rate increase) with prioritized sampling. Similarly, our framework consumes less power than *Uniform DDPG* in Fig. 3(b), while performing close to *Max-min*.

We now allow the UE toggling to happen more frequently by setting T_{tog} to 350 in Fig. 4. When the first toggle happens at episode 350, the proposed scheme is able to quickly reach its newly converged value of 14.5 Mbps at episode 450 in Fig. 4(a). On the other hand, *Uniform DDPG* settles for a rate 11.72% lower only 100 episodes later. The second toggle occurs at episode 700, in which case the minimum user rate is expected to increase as indicated by the *Max-min* and *Full power* baselines. With prioritization, the agent is able to adapt to this environment change, characterized by the corresponding increase in the guaranteed rate. In contrast, *Uniform DDPG* likely needs more time to do so.

D. Mobile Users With UE Toggling

We now consider the case of mobile UEs with $T_{\text{tog}} = 500$ and $K_{\text{tog}} = 0.1K$ in Fig. 5. After the toggle at episode 500, we observe that it now takes more time for both DDPG-based systems to reach convergence. Specifically, compared to the static scenario in Section V-C, this happens 100 episodes later for our proposed scheme, while *Uniform DDPG* has yet to do so even at episode 1000. In this case, the agent has to deal with the additional user mobility that impacts the power or action selection, on top of having to detect the environment change caused by the UE toggling. Nevertheless, we still benefit from the prioritized sampling that achieves significant performance gain in terms of convergence speed, rate and power consumption, while approaching the optimal performance of *Max-min*.

VI. Conclusion

We have proposed a novel DRL framework for power control in uplink cell-free massive MIMO, designed to handle device activation and deactivation combined with user mobility, without requiring any prior knowledge at the CPU. To ensure that the system can quickly adapt to the dynamics of a practical wireless environment, we have exploited a TD-error-based prioritization that accelerates the learning process. Numerical results have shown that the proposed algorithm achieves faster convergence and enhanced performance compared to the baseline schemes. As future work, we aim at investigating different variants of prioritized sampling, and extending the proposed framework towards a multi-agent system for distributed learning.

Acknowledgment

The authors acknowledge TU Wien Bibliothek for financial support through its Open Access Funding Programme.

References

[1] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, “Cell-free massive MIMO versus small cells,” *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1834–1850, Mar. 2017.
[2] C.-X. Wang et al., “On the road to 6G: Visions, requirements, key technologies, and testbeds,” *IEEE Commun. Surveys Tuts.*, vol. 25, no. 2, pp. 905–974, 2nd Quart., 2023.
[3] H. Q. Ngo et al., “On the total energy efficiency of cell-free massive MIMO,” *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 1, pp. 25–39, Mar. 2018.
[4] T. Van Chien, E. Björnson, and E. G. Larsson, “Joint power allocation and load balancing optimization for energy-efficient cell-free massive MIMO networks,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6798–6812, Oct. 2020.
[5] C. D’Andrea, A. Zappone, S. Buzzi, and M. Debbah, “Uplink power control in cell-free massive MIMO via deep learning,” in *Proc. IEEE CAMSAP*, Le Gosier, Guadeloupe, 2019, pp. 554–558.
[6] M. Zaher, Ö. T. Demir, E. Björnson, and M. Petrova, “Learning-based downlink power allocation in cell-free massive MIMO systems,” *IEEE Trans. Wireless Commun.*, vol. 22, no. 1, pp. 174–188, Jan. 2023.
[7] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.
[8] M. Rahmani et al., “Deep reinforcement learning-based power allocation in uplink cell-free massive MIMO,” in *Proc. IEEE WCNC*, Austin, Texas, USA, 2022, pp. 459–464.
[9] L. Luo et al., “Downlink power control for cell-free massive MIMO with deep reinforcement learning,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6772–6777, Jun. 2022.
[10] X. Zhang, M. Kaneko, V. An Le, and Y. Ji, “Deep reinforcement learning-based uplink power control in cell-free massive MIMO,” in *Proc. IEEE CCNC*, Las Vegas, NV, USA, 2023, pp. 567–572.
[11] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, “Prioritized experience replay,” in *Proc. ICLR*, 2016, pp. 1–21.
[12] S. Zhou, Y. Cheng, X. Lei, and H. Duan, “Deep deterministic policy gradient with prioritized sampling for power control,” *IEEE Access*, vol. 8, pp. 194240–194250, 2020.
[13] T. P. Lillicrap et al., “Continuous control with deep reinforcement learning,” in *Proc. ICLR*, 2016, pp. 1–14.
[14] M. Bashar et al., “On the uplink max–min SINR of cell-free massive MIMO systems,” *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2021–2036, Apr. 2019.