Constrained Reinforcement Learning for Predictive Control in Real-Time Stochastic Dynamic Optimal Power Flow

Tong Wu, *Member, IEEE*, Anna Scaglione, *Fellow, IEEE*, and Daniel Arnold, *Member, IEEE*

***Abstract*—Deep Reinforcement Learning (DRL) has emerged as a favored approach for resolving control challenges in power systems. Traditional DRL guides the agent through exploration of numerous policies, each embedded within a neural network (NN), aiming to maximize the associated reward function. However, this approach can lead to infeasible solutions that violate physical constraints such as power flow equations, voltage limits, and dynamic constraints. Ensuring these constraints are met is crucial in power systems, as they are a safety critical infrastructure. To address this issue, existing DRL algorithms remedy the problem by projecting the actions onto the feasible set, which can result in suboptimal solutions. This article introduces a pioneering primal-dual approach to learn optimal constrained DRL policies specifically for predictive control in real-time stochastic dynamic optimal power flow. The focus is on controlling power generations and battery outputs while ensuring compliance with critical constraints. We also prove the convergence of the critic and actor networks. Our case studies, based on IEEE standard systems, underscore the preeminence of the proposed approach in identifying near-optimal actions for various states while concurrently adhering to safety constraints.**

***Index Terms*—Constrained reinforcement learning, stochastic dynamic optimal power flow control.**

NOMENCLATURE

Sets

$\mathcal{N}$	The set of all buses, having a cardinality of N.
$\mathcal{G}$	The set of all buses equipped with G generators.
$\mathcal{G}_s$ ($\mathcal{G}_n$)	The set of slack (non-slack) buses fitted with generation capabilities.
$\mathcal{B}$	The set of all Battery Energy Storage Systems (BESSs), with a cardinality of B.

Manuscript received 15 March 2023; revised 7 July 2023 and 10 October 2023; accepted 15 October 2023. Date of publication 20 October 2023; date of current version 19 April 2024. This work was supported in part by National Science Foundation (NSF) under Grant NSF ECCS 2210012, and in part by the Cybersecurity, Energy Security, and Emergency Response (CESER), Risk Management and Tools and Technologies (RMT) Program of the U.S. Department of Energy through Supervisory Parameter Adjustment for Distribution Energy Storage (SPADES) Project under Grant DE-AC02-05CH11231. An earlier version of this paper was presented at PESGM 2023 in [doi: 10.48550/arXiv.2302.10382]. Paper no. TPWRS-00368-2023. *(Corresponding author: Anna Scaglione.)*

Tong Wu and Anna Scaglione are with the Department of Electrical and Computer Engineering, Cornell Tech, Cornell University, New York City, NY 10044 USA (e-mail: tw385@cornell.edu; as337@cornell.edu).

Daniel Arnold is with the Energy Technologies Area, E O Lawrence Berkeley National Laboratory Berkeley, CA 94720 USA (e-mail: dbarnold@lbl.gov).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TPWRS.2023.3326121.

Digital Object Identifier 10.1109/TPWRS.2023.3326121

Operators

$\boldsymbol{A}^T, \boldsymbol{A}^H$	The transpose and Hermitian of matrix $\boldsymbol{A}$.
$D(\boldsymbol{A})$	The vector of the diagonal elements of $\boldsymbol{A}$.
$diag(a)$	A diagonal matrix with diagonal entries from a.
$\boldsymbol{A} \circ \boldsymbol{B}$	Hadamard product (entry by entry product).
$(\boldsymbol{A})^*$	Conjugate of a complex vector or matrix.

Abbreviations

Cplx-STGCN	Complex-valued Spatio-temporal graph convolutional neural networks.
BESS	Battery Energy Storage System.
SDOPF	Stochastic Dynamic Optimal Power Flow.
CRL	Constrained Reinforcement Learning.
DRL	Deep Reinforcement Learning.

I. INTRODUCTION

A. Background and Motivation

THE power grid is a complex, dynamic network composed of interconnected components that can be influenced by numerous factors, including fluctuations in demand, changes in energy resource availability, and the operation of power plants and control assets (e.g. frequency control and voltage regulation) [2]. The increasing penetration of renewable and decentralized energy resources (DER) poses significant operational challenges for power networks operators, because of the need to manage their dynamic behavior. At the same time, the widespread deployment of advanced measurement technologies such as Phasor Measurements Units (PMUs) in the bulk system, and the Advanced Metering Infrastructure (AMI), in distribution systems, provides new opportunities to leverage the data for real-time power network control [3], rather than relying only on local control loops to respond to the grid state.

From an operational perspective, in the presence of the uncertainty not only of demand but also of DER generation, the challenge of optimal control of dynamic devices such as battery energy storage systems (BESSs), is being addressed through the formulation of stochastic dynamic optimal power flow (SDOPF) methods. These methods dispatch generation resources and select BESSs charging or discharging periods accounting for the future impact of real-time decision-making, to ensure efficient and reliable operations [4]. In fact, a SDOPF formulation solves the general problem of how to optimally dispatch generation and operating storage units across a network to meet net electric load within a time-horizon economically, accounting for the

dynamic constraints of the electric power supply sources [5]. However, its implementation in real-time is challenging due to the unpredictable nature of DER and demand of electric power, the dynamic constraints of generation and storage, and the computational complexity of the SDOPF problem.

Deep reinforcement learning (DRL) has gained significant attention for its potential to learn SDOPF policies in dynamic grid control applications, such as BESS management and EV charging control with dynamic constraints. By training the algorithm offline on real-world scenarios, several studies (see e.g. [3], [6], [7]) demostrate that DRL policies for decision making under uncertainty are a promising solution for optimizing the real-time operation of BESSs [3]. Examples of DRL applications for BESS management include a distributed operation strategy using double deep Q-learning for community BESS in microgrids [6]. In [7], an optimal strategy for electric vehicle (EV) charging was developed in a distribution network through the use of reinforcement learning, taking into account the dynamic constraints of the state of charge (SOC) of the EVs.

B. Related Works

In our review of the prior art we will highlight research in the areas of stochastic dynamic optimal power flow (OPF) (as it is particularly vast, our review is not comprehensive) learning-methods to solve OPF formulations, as well as the literature on constrained reinforcement learning.

1) Stochastic Dynamic OPF: A stochastic OPF formulation was first introduced in [8] to solve the optimal dispatch problem with uncertainties in power systems. [9] were key in developing a dynamic stochastic optimal power flow control model that incorporated wide-area measurements for smart grids. [10] further innovated by extending this model to address the unique challenges of intermittent renewable energy generation. In a different setting, [11] showcased the utility of dynamic stochastic optimal power flow in residential energy systems with their work on smart homes with PEV energy storage. Lastly, [12] made strides by introducing a hybrid approach, combining stochastic and deterministic methods in their multiperiod DC optimal power flow model. However, these iterative optimization methods have very high computational complexity, limiting their promise for real-time control.

2) Learning-Based OPF: Recently, learning-based approaches for solving OPF problems are trending upward. In a nutshell, the idea behind these algorithms is to leverage the universal approximation capabilities of DNNs to learn the policy mapping load input onto OPF solutions in the class of function representable with the DNN [13], [14]. Once the training is completed, one can pass the network loads to the trained DNN and instantly obtain a quality solution for the OPF. A key difficulty for applying DNN to solve AC-OPF problems lies in the fact that the solutions may not satisfy the physical and operational constraints that make the solution feasible. To address this problem, [13] includes a regularization term in the DNN reward objective that penalizes solutions that are AC-OPF infeasible. In [14], instead, a small-scale mapping method was proposed to recover the feasible results. The unsupervised idea is to learn the solution in an unsupervised manner, minimizing the cost directly [15], [16]. [15] considered both the penalty function and mapping function for both equality and inequality constraints. In [16], a piece-wise penalty function based on the log-barrier is considered to enforce constraints. However, these learning-based methods cannot consider dynamic constraints and how the current actions affect the future.

3) Constrained Reinforcement Learning: DRL methodologies have shown significant promise in tackling complex stochastic nonlinear dynamic control problems by aiming to maximize not only immediate but also future rewards from control actions. However, it is critical to note that DRL policies can occasionally render decisions that are infeasible due to violations of power flow equations and SOC limits. In light of this, Constrained Reinforcement Learning (CRL) has emerged as a pivotal approach for addressing constrained sequential decision-making issues within safety-critical systems, particularly by employing techniques such as Lagrangian relaxation to navigate through CRL problems (see review [17]). Further to the aforementioned resources, [18], [19] utilize a Natural Policy Gradient Primal-Dual, providing assurances for convergence towards a fixed point. [20] introduces an exploration into chance-constrained reinforcement learning via primal-dual methods. Additionally, [21] offers an upper confidence primal-dual algorithm and substantiates upper boundaries for both regret and constraint violation. Similar work on CRL application in OPF consider that operational constraints are satisfied by a novel convex safety layer based on the penalty convex-concave procedure [22]. In [23], a Lagrangian based DRL is considered to optimize OPF function. However, this method is hard to scale in the presence of multi-stage dynamic constraints because its design is too simple to handle equality constraints. Instead of training CRL that can generate the feasible actions directly, [24] considers the feasibility projection that maps the actions onto the convex feasible sets.

C. Contributions and Organization

The above CRL techniques predominantly focus on aggregate constraints, wherein the cumulative total of a constrained variable from the initial to the present time step is bound within a specified limit. Conversely, within the context of multi-stage stochastic dynamic OPF, each time step necessitates adherence to both power-flow and dynamic constraints. The main contribution of this article is summarized as follows:

- We propose a training framework for CRL that ensures the actions selected by the policy are feasible at each time step. Specifically, we modify the twin delayed deep deterministic policy gradient algorithm (TD3) [25] to optimize the control of power generation and BESS charging and discharging actions in a multi-stage SDOPF problem.
- Our proposed approach adopts predictive control, which implies that the demands and renewable energy are unknown to us. Nevertheless, the optimization technique needs to be aware of the demands and renewable energy in order to solve the optimization procedure and yield actions.
- We use the augmented Lagrangian method to solve the constrained SDOPF and update the dual variables of the modified TD3 using primal-dual methods.

- We introduce a complex-valued spatio-temporal graph convolutional neural network (Cplx-STGCN) for the actor to capture the spatiotemporal correlation of the environment states.
- We prove the convergence of critic networks and, under mild assumptions, the convergence of the augmented Lagrangian actor networks.

The article is structured as follows: Section II discusses the SDOPF problem, while Section III introduces a constrained reinforcement learning method. Section IV presents the complex-valued graph convolutional policy function, and Section V analyzes its convergence. Implementation and a specific case study of the proposed method are shared in Section VI. Section VII validates the approach via experimental simulations, and conclusions are drawn in Section VIII.

II. Problem Formulation

The problem solved in this article is an instance of the following a multi-stage stochastic optimal control formulation:

$$\min_{\pi(\boldsymbol{x}_{t-1})} \mathbb{E}_{\boldsymbol{d},\pi(\cdot)}\left[\sum_{t=\tau}^{\tau+T-1} \ell_t(\boldsymbol{x}_t,\boldsymbol{a}_t,\boldsymbol{d}_t)\right] \tag{1a}$$

$$\boldsymbol{x}_t = f_t(\boldsymbol{x}_{t-1},\boldsymbol{a}_{t-1},\boldsymbol{d}_{t-1}), \tag{1b}$$

$$\boldsymbol{a}_t = \pi(\boldsymbol{x}_{t-1}),\ (\boldsymbol{x}_t,\boldsymbol{a}_t)\in\chi_t, \tag{1c}$$

where the state vector at time t is denoted by $\boldsymbol{x}_t$, while $\boldsymbol{a}_t$ represents a control vector at the same time point, encompassing all controllable devices within power grids. The vector $\boldsymbol{d}_t$ indicates environmental observations, such as demands and renewable energy inputs. The cost function is represented by $\ell_t(\boldsymbol{x}_t,\boldsymbol{a}_t,\boldsymbol{d}_t)$, and χ_t stands for network and device bound constraints. The system dynamics function, f_t, models the internal dynamics and various temporal interdependencies of grid assets, like the SOC for BESSs. Furthermore, $\pi(\boldsymbol{a}_t|\boldsymbol{x}_t)$ denotes the randomized policy.

Concerning the SDOPF, it represents a standard multi-stage stochastic optimal control problem, aiming to achieve economic dispatch of power flows by effectively controlling power generations and BESSs. Specifically, $\boldsymbol{x}_t = [\boldsymbol{v}_t;\boldsymbol{soc}_t]^\top$ incorporates voltage angles, $\boldsymbol{v}_t$, and the vector of SOCs, $\boldsymbol{soc}_t$, for all batteries within the system. Note that $\forall i\in\mathcal{N}/\mathcal{B}, [\boldsymbol{soc}_t]_i = 0$ and $\forall i\in\mathcal{B}$, $[\boldsymbol{soc}_t]_i$ is the state of charge of that BESS, and thus $\boldsymbol{soc}_t$ has the same dimension with $\boldsymbol{v}_t$. The control vector $\boldsymbol{a}_t = [\boldsymbol{g}_t^p;\boldsymbol{g}_t^q;\boldsymbol{p}_{ch,t};\boldsymbol{p}_{dis,t}]^\top$ includes active power generation $\boldsymbol{g}_t^p = [g_{1,t}^p,\ldots,g_{G,t}^p]^\top$, reactive power generations $\boldsymbol{g}_t^q = [g_{1,t}^q,\ldots,g_{G,t}^q]^\top$ and $\boldsymbol{p}_{dis,t} = [p_{dis,1,t},\ldots,p_{dis,B,t}]^\top$ and $\boldsymbol{p}_{ch,t} = [p_{ch,1,t},\ldots,p_{ch,B,t}]^\top$ denote the charge and discharge rates of the BESSs.

A. Objectives

The objective of the SDOPF formulation, when incorporating BESSs, entails components f_{ess} and f_{cost}. Specifically, f_{cost} represents the fuel costs, detailed as follows:

$$f_{cost,t} = \sum_{i\in\mathcal{G}}\left(a_i g_{i,t}^2 + b_i g_{i,t} + c_i\right), \tag{2}$$

where a_i, b_i, and c_i are positive. f_{ess} denotes power loss from battery charging and discharging.

$$f_{ess,t} = \sum_{i\in\mathcal{B}}(1-\eta_{ch,i})p_{ch,i,t} + \left(\frac{1}{\eta_{dis,i}} - 1\right)p_{dis,i,t} \tag{3}$$

where $\eta_{ch,i}$ and $\eta_{dis,i}$ denote the charging and discharging efficiency respectively, and $p_{ch,i,t}$ and $p_{dis,i,t}$ denote the charging and discharging power of the i^{th} BESS. The dynamics of the SOC across each period are described as follows:

$$soc_{i,t} = soc_{i,t-1} + \frac{\Delta t}{E_{cap}}\left(\eta_{ch}\, p_{ch,i,t} - \frac{1}{\eta_{dis}}\, p_{dis,i,t}\right), i\in\mathcal{B} \tag{4}$$

where Δt represents the duration of each decision period and E_{cap} is the energy capacity of the BESS. The real power values of $p_{dis,i,t}$ and $p_{ch,i,t}$ are subjected to the following constraints:

$$\begin{aligned} 0 \le p_{ch,i,t} \le P_{rated}^{ch}, i\in\mathcal{B} \\ 0 \le p_{dis,i,t} \le P_{rated}^{dis}, i\in\mathcal{B} \end{aligned} \tag{5}$$

where P_{rated}^{ch} and P_{rated}^{dis} denote the limits for charging and discharging rates, respectively. The reward for an action is derived as the complement of the objectives in (1a), which is intended to be maximized:

$$r_t = \sum_{t=\tau}^{\tau+T-1} -\ell_t(\boldsymbol{x}_t,\boldsymbol{a}_t,\xi_t) = \sum_{t=\tau}^{\tau+T-1}\left(-f_{cost,t} - f_{ess,t}\right). \tag{6}$$

B. Stochastic Dynamic OPF

In this work, we utilize the AC power flow to uphold the power-flow constraints. The formulation for the multi-stage SDOPF problem is as follows:

$$\min_{\pi(\boldsymbol{a}_t|\boldsymbol{v}_{t-1},\boldsymbol{soc}_{t-1})} \mathbb{E}_{\boldsymbol{d}}\left[\sum_{t=\tau}^{\tau+T-1}\ell_t(\boldsymbol{x}_t,\boldsymbol{a}_t,\boldsymbol{d}_t)\right] \tag{7a}$$

$$\mathbf{M}_b\boldsymbol{p}_{dis,t} - \mathbf{M}_b\boldsymbol{p}_{ch,t} + \mathbf{M}_g\boldsymbol{g}_t^p - \boldsymbol{d}_t^p = \Re\{D(\boldsymbol{v}_t\boldsymbol{v}_t^H\mathbf{Y}^H)\}, \tag{7b}$$

$$\mathbf{M}_g\boldsymbol{g}_t^q - \boldsymbol{d}_t^q = \Im\{D(\boldsymbol{v}_t\boldsymbol{v}_t^H\mathbf{Y}^H)\}, \tag{7c}$$

$$\underline{\boldsymbol{g}}^p \le \boldsymbol{g}_t^p \le \overline{\boldsymbol{g}}^p,\ \underline{\boldsymbol{g}}^q \le \boldsymbol{g}_t^q \le \overline{\boldsymbol{g}}^q,\ \underline{\boldsymbol{v}} \le |\boldsymbol{v}| \le \overline{\boldsymbol{v}} \tag{7d}$$

$$0 \le \boldsymbol{p}_{ch,t} \le \boldsymbol{p}_{rated}^{ch},\ 0 \le \boldsymbol{p}_{dis,t} \le \boldsymbol{p}_{rated}^{dis}, \tag{7e}$$

$$|(\mathbf{C}_f\boldsymbol{v}_t)\circ(\mathbf{Y}_f^*\boldsymbol{v}_t^*)| \le s_{\max}, |(\mathbf{C}_{to}\boldsymbol{v}_t)\circ(\mathbf{Y}_{to}^*\boldsymbol{v}_t^*)| \le s_{\max} \tag{7f}$$

$$\boldsymbol{soc}_{\min} \le \boldsymbol{soc}_t \le \boldsymbol{soc}_{\max}, \forall t\in[\tau,\tau+T-1] \tag{7g}$$

$$\boldsymbol{soc}_t = \boldsymbol{soc}_{t-1} + \frac{\Delta t}{E_{cap}}\left(\eta_{ch}\boldsymbol{p}_{ch,t} - \frac{\boldsymbol{p}_{dis,t}}{\eta_{dis}}\right), \tag{7h}$$

where the active power demand vector is $\boldsymbol{d}_t^p = [d_{1,t}^p,\ldots,d_{N,t}^p]^\top$, the reactive power demand vector is $\boldsymbol{d}_t^q = [d_{1,t}^q,\ldots,d_{N,t}^q]^\top$, $\mathbf{Y}$

is the admittance matrix, and $\mathbf{Y}_f$ and $\mathbf{Y}_{to}$ denote the branch admittance matrices corresponding to the 'from bus' and the 'to bus' respectively, and $v_t = [v_{1,t}, \ldots, v_{N,t}]^\top$ is the grid state in the AC power flow, i.e. $v_t = |v_t| \circ e^{\mathrm{j}\boldsymbol{\theta}_t}$, $v_{n,t} = |v_{n,t}|e^{\mathrm{j}\theta_{n,t}}$. Let $\mathbf{C}_f$ and $\mathbf{C}_{to}$ represent the connection matrices for line and the 'from buses' and the 'to buses', respectively. The complex power injection is given by $s = (v \circ i^*) = D(v(i)^H)$, where $v \in \mathcal{C}^N$ represents the voltage phasor vector, and $i \in \mathcal{C}^N$ symbolizes the current phasor vector. $(\cdot)^*$ denotes the conjugate of the complex vector or matrix and $\circ$ denotes the Hadamard product (element-wise product). The function $D(\cdot)$ signifies the vector of diagonal elements from a matrix, while $(i)^H$ and $(i)^*$ represent the Hermitian and the conjugate of the vector i, respectively. The current phasor vector can be further expressed as $i = \mathbf{Y}v$, thereby allowing s to be rewritten as $D(vv^H\mathbf{Y}^H)$. Therefore, the active and reactive power injections are expressed as $\Re(s) = \Re\{D(vv^H\mathbf{Y}^H)\}$ and $\Im(s) = \Im\{D(vv^H\mathbf{Y}^H)\}$. $s_{\max}$ denotes the vector of long-term rating limits for each branch. Let $\mathbf{M}_g$ be the $\{0,1\}^{N\times G}$ matrix that maps the generation vector g_t^p (where $g_t^p \in \mathbb{R}^{|\mathcal{G}|}$) to $\mathbb{R}^N$, as follows:

$$[\mathbf{M}_g g^p]_i = 0, [\mathbf{M}_g g^q]_i = 0, \quad \forall i \in \mathcal{N} \setminus \mathcal{G}$$
$$[\mathbf{M}_g g^p]_i = g_j^p, [\mathbf{M}_g g^q]_i = g_j^q, \quad \forall i \in \mathcal{G},\ \forall j \in [1, \ldots, G]. \tag{8}$$

Likewise, $\mathbf{M}_b$ represents the matrix that maps vectors $p_{ch,t}$ and $p_{dis,t}$ across the entire network, inserting zeroes in buses without batteries. The feasible set of constraints (7b)–(7g) is symbolized by χ_t. The method proposed to solve (7) is detailed in next.

III. Constrained Reinforcement Learning

In this section, we enhance actor-critic policy gradient methods to incorporate the immediate constraints as articulated in (7). We specifically utilize the continuous reinforcement learning framework, TD3 [25], which employs policy gradients to optimize actions. This methodology is akin to the gradient descent strategy used in convex optimization. Despite the action space being infinite, our method proves effective in recognizing actions that approximate the optimal solution.

A. Actor-Critic Method

The basic policy gradient method, which operates as an actor-only mechanism, generally encounters issues of high variance and lethargic learning when adjusting parameters for the approximated policy function [26]. The actor-critic method mitigates these drawbacks, amending policy function parameters guided by an approximate value function, known as *the critic*. In Fig. 1, the actor, symbolized by the policy function π_ϕ and parameterized by ϕ, selects actions, while the critic, represented as a state-value function Q_ξ and parameterized by ξ, assesses the actor's decisions.

1) Forecasting Action: The action tuple for the multi-stage SDOPF at time t is denoted as follows:

$$\hat{a}_t = [\hat{g}_t^p; \hat{g}_t^q; \hat{p}_{ch,t}; \hat{p}_{dis,t}]^\top,$$
$$\boldsymbol{A}_t = [\hat{a}_t, \ldots, \hat{a}_{t+T-1}]^\top, \tag{9}$$

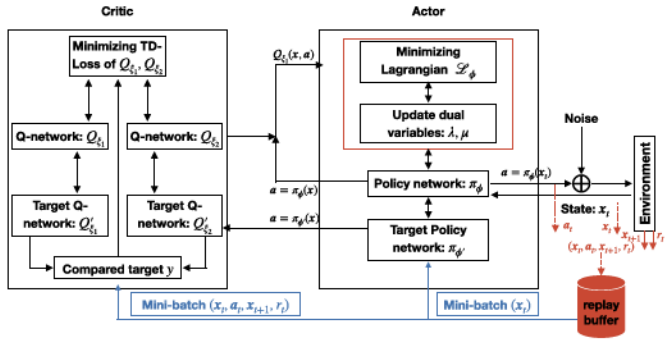


Fig. 1. Actor-critic architecture.

where $\hat{a}_t$ represents the NN-normalized control action, with elements $\hat{g}_t^p, \hat{g}_t^q, \hat{p}_{ch,t}$, and $\hat{p}_{dis,t}$ normalized to [0, 1] due to the NN's sigmoid activation. $\boldsymbol{A}_t$ denotes the vector of impending control actions, i.e., $\hat{a}_t, \ldots, \hat{a}_{t+T-1}$, utilizing a sliding window policy. Only $\hat{a}_t$ is applied to the environment, ensuring the satisfaction of constraints χ_t and the feasibility of future actions $\hat{a}_{t+1}, \ldots, \hat{a}_{t+T-1}$.

2) Voltage Magnitudes: The policy controls the future power injections, which are implicitly related to the voltage magnitudes through the power flow equations. In order to consider the constraint $\underline{v} \le |v| \le \overline{v}$, we utilize an independent neural network to solve for the voltage magnitudes for a given action to ensure that they are are equal to the ground-truth and within the bound $[\underline{v}, \overline{v}]$. We define the prediction network as $|\hat{v}| = P_\omega(x)$, where $|\hat{v}|$ is defined as the normalized versions of $|v|$ in the range $[0, 1]$.

3) Critic Design: Q-learning employs temporal difference learning to derive the value function, utilizing foundational principles by [27] and the pivotal Bellman equation [28]. This equation intimately connects the value attributed to a current state-action pair, $(x, \boldsymbol{A})$, with that of its forthcoming counterpart, $(x', \boldsymbol{A}')$:

$$Q_\xi(x, \boldsymbol{A}) = r + \gamma \mathbb{E}[Q_\xi(x', \boldsymbol{A}')],\ \boldsymbol{A}' = \pi_\phi(x') \tag{10}$$

where γ denotes the discount coefficient for impending rewards, and for expansive state spaces, the value might be approximated using a differential function estimator $Q_\xi(x, \boldsymbol{A})$, characterized by parameters ξ. In the context of deep Q-learning, the network undergoes updates utilizing temporal difference learning, employing a critic network $Q_\xi(x, \boldsymbol{A})$ to sustain a consistent objective y across numerous updates.

$$y = r + \gamma Q_\xi(x, \boldsymbol{A}),\ \boldsymbol{A} = \pi_\phi(\boldsymbol{A}|x), \tag{11}$$

where the actions are derived from a target actor network π_ϕ and r is defined in (6).

a) Target Networks: Utilizing *target networks* enhances stability in deep reinforcement learning by mitigating approximation errors [29]. As depicted in Fig. 2, two target networks, $Q_{\xi'_1}$ and $Q_{\xi'_2}$, and two critic networks, Q_{ξ_1} and Q_{ξ_2}, are employed. The Clipped Double DQN approach [29] employs target networks to select the minimal value between two estimates:

$$y = r + \gamma \min_{i=1,2} Q_{\xi'_i}(x, \boldsymbol{A}), \tag{12}$$

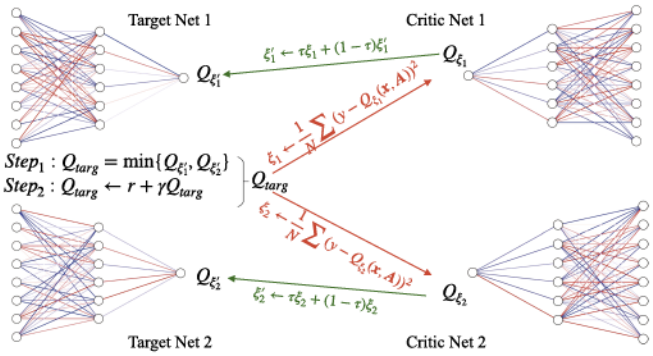


Fig. 2. Critic design.

where the value target y avoids introducing extra overestimation relative to the conventional Q-learning target, enhancing the critic networks' estimation accuracy.

b) Critic Networks: Upon obtaining the target Q values, critic networks adjust their parameters by:

$$\xi_1 \leftarrow \arg\min_{\xi_1} \frac{1}{N}\sum (y - Q_{\xi_1}(\boldsymbol{x}, \boldsymbol{A}))^2,$$
$$\xi_2 \leftarrow \arg\min_{\xi_2} \frac{1}{N}\sum (y - Q_{\xi_2}(\boldsymbol{x}, \boldsymbol{A}))^2 \tag{13}$$

Given batch size N and target values y from (12), the critic networks undergo an update. Subsequently, target networks' weights are updated at each time step by a factor τ, derived from the critic networks:

$$\xi_1' \leftarrow \tau\xi_1 + (1-\tau)\xi_1', \ \xi_2' \leftarrow \tau\xi_2 + (1-\tau)\xi_2'. \tag{14}$$

where ξ_1 and ξ_2 represent critic network parameters per (13), and ξ_1' and ξ_2' symbolize target network parameters according to (11), both target and critic networks iteratively undergo mutual updates.

4) Constrained Actor Design: After defining the critic, we proceed to establish the actor network and introduce its constrained action space. Typically, we train the action network to maximize the critic network, i.e.

$$\phi \leftarrow \arg\max_{\phi} Q_{\xi_1}(\boldsymbol{x}_{t-1}, \pi_\phi(\boldsymbol{x}_{t-1})). \tag{15}$$

where ϕ denotes the action network parameters. We can utilize either Q_{ξ_1} or Q_{ξ_2} to guide $\pi_\phi(\cdot)$ in updating ϕ. An action $\boldsymbol{A}_t = \pi_\phi(\boldsymbol{x}_{t-1})$ is deemed feasible if it satisfies all constraints, χ_t. Thus, π_ϕ is derived by maximizing the critic network while upholding χ_t:

$$\max_{\phi} Q_{\xi_1}(\boldsymbol{x}_{t-1}, \pi_\phi(\boldsymbol{x}_{t-1})) \ \text{s.t.}\ \boldsymbol{A}_t \in \chi_t. \tag{16}$$

where $\boldsymbol{A}_t = \pi_\phi(\boldsymbol{x}_{t-1})$.

B. Primal-Dual Constrained RL Framework

The actor network $\pi_\phi(\cdot)$ involves $g_t^p, g_t^q, p_{ch,t}, p_{dis,t}$, which are linearly constrained in (7b)–(7h). To write the linear equality constraints in a compact way, we consider

$$\boldsymbol{L}\pi_\phi(\boldsymbol{x}_{t-1}) = \boldsymbol{b} \tag{17}$$

Likewise, the linear inequality constraints are given by

$$\boldsymbol{K}\pi_\phi(\boldsymbol{x}_{t-1}) \preceq \boldsymbol{c}, \ \text{or}\ [\boldsymbol{K}\pi_\phi(\boldsymbol{x}_{t-1}) - \boldsymbol{c}]_+ = 0 \tag{18}$$

where $[a]_+ = \max\{a, 0\}$. We can summarize the optimization problem of the constrained reinforcement learning as

$$\min_{\phi} \ -Q_{\xi_1}(\boldsymbol{x}_{t-1}, \pi_\phi(\boldsymbol{x}_{t-1}))$$
$$\text{s.t.}\ \boldsymbol{L}\pi_\phi(\boldsymbol{x}_{t-1}) - \boldsymbol{b} = \boldsymbol{0}, \ \boldsymbol{K}\pi_\phi(\boldsymbol{x}_{t-1}) - \boldsymbol{c} \preceq \boldsymbol{0} \tag{19}$$

where Q_{ξ_1} is a non-convex, non-differentiable and Lipschitz continuous function. Both $\boldsymbol{L}\pi_\phi(\boldsymbol{x}) - \boldsymbol{b} = \boldsymbol{0}, \boldsymbol{K}\pi_\phi(\boldsymbol{x}) - \boldsymbol{c} \preceq \boldsymbol{0}$ are linear, but $\pi_\phi(\boldsymbol{x})$ is non-convex, non-differentiable and Lipschitz continuous function. The algorithm entails solving:

1) the Primal Problem by SGD for T loops:

$$\min_{\phi} \mathcal{L}_\phi^\alpha = \min_{\phi} -Q_{\xi_1}(\boldsymbol{x}_{t-1}, \pi_\phi(\boldsymbol{x}_{t-1})) + \boldsymbol{\lambda}^\top[\pi(\boldsymbol{x}_{t-1}) - \boldsymbol{d}_t - \boldsymbol{L}\boldsymbol{\vartheta}] + \boldsymbol{\mu}^\top[\boldsymbol{K}\pi_\phi(\boldsymbol{x}_{t-1}) - \boldsymbol{c}]_+ + \frac{\alpha_I^\top}{2}$$
$$\|\pi(\boldsymbol{x}_{t-1}) - \boldsymbol{d}_t - \boldsymbol{L}\boldsymbol{\vartheta}\|_2^2 + \frac{\alpha_F^\top}{2}\|[\boldsymbol{K}\pi_\phi(\boldsymbol{x}_{t-1}) - \boldsymbol{c}]_+\|_2^2$$

2) the Dual Problem:

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \alpha_\lambda[\pi(\boldsymbol{x}_{t-1}) - \boldsymbol{d}_t - \boldsymbol{L}\boldsymbol{\vartheta}]$$
$$\boldsymbol{\mu}^{k+1} = \boldsymbol{\mu}^k + \alpha_\mu[\boldsymbol{K}\pi_\phi(\boldsymbol{x}_{t-1}) - \boldsymbol{c}]_+$$

The main goal of primal-dual process is to reduce a complex optimization problem into smaller, easier-to-solve subproblems that involve the use of duality (i.e. the augmented Lagrangian), and ensure that the constraints are satisfied. At the most basic level, duality makes it easier to solve with a gradient descent methodology constrained optimization by using the dual function maximization to update the dual variable of the Lagrange multiplier, and then doing a gradient descent on the primal problem using the current multiplier update. In our case the primal problem is that of optimizing the neural network model that represents the policy with an objective that is the Lagrangian of the problem, and we use the dual variable update to enforce the power-flow constraints. Since the only situation in which the dual variable converges is when the constraints are met, the policy function will not adopt infeasible actions. More specifically:

- *Primal Update:* This step involves updating the primal variables while keeping the dual variables fixed. In this stage, we solve the subproblem, which typically has a simpler structure and can be solved independently.
- *Dual Update:* After the primal variables are updated, the dual variables are then updated. This update is done by taking into account the discrepancies (or residuals) between the current primal variables and the constraints. If the constraints are not satisfied (i.e., the residuals are not zero), the dual variables are adjusted to enforce these constraints in the next primal update. This update is usually simple and straightforward.

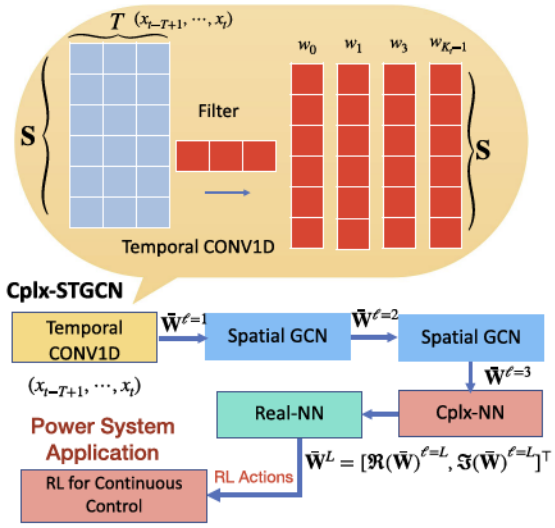


Fig. 3. Architecture of complex-valued spatio-temporal graph convolutional neural network.

This iterative process is repeated until the residuals become small enough (below a predefined threshold), indicating that a good approximation to the solution of the constrained optimization problem has been found.

IV. Cplx-GCN-Based Actor Networks

A reinforcement learning algorithm continuously interacts with the environment, which provides the time-series of the system states. The physics of the grid implies that the grid state variables correlation is a function of the grid topological and electrical characteristics. It has been amply documented at this point in time that the best way to leverage the knowledge of the underlying grid is to use graph convoltional neural networks.

In this work, we consider a graph signal $x = [v; soc]^\top \in \mathbb{C}^{2|\mathcal{N}|}$, where each entry $[v]_i$ and $[soc]_i$ represent the voltage phasor and the state of charge at bus $i \in \mathcal{N}$, respectively. The set $\mathcal{N}_i$ denotes the nodes connected to node i. The graph shift operator (GSO) $\mathbf{S} \in \mathbb{R}^{|\mathcal{N}|\times|\mathcal{N}|}$ linearly combines values of the graph signal's neighbors. Operations such as filtering, transformation, and prediction are closely related to the GSO. In this work, we focus on complex symmetric GSOs, meaning $\mathbf{S} = \mathbf{S}^\top$. This is relevant for our power grid application, where $\mathbf{S} = \mathbf{Y}$ [30].

As shown in Fig. 3, the temporal convolutional layer contains a 1-D CNN with a width-T kernel with K_t output channels. In this work, we consider the input state $x_t \in \mathbb{C}^{|\mathcal{N}|\times 2}$ with two channels, i.e., v_t and soc_t. The convolutional kernel $\mathbf{\Gamma} \in \mathbb{C}^{2T\times K_t}$ is designed to map the input $\mathbf{X} \in \mathbb{C}^{|\mathcal{N}|\times 2T}$ into a output graph signal with C_t channels $\bar{\mathbf{X}} \in \mathbb{C}^{|\mathcal{N}|\times K_t}$. Therefore, we define the temporal convolution as,

$$\bar{\mathbf{X}} = \mathbf{\Gamma} *_{\mathcal{T}} \mathbf{X}, \tag{20}$$

where each column of $[\bar{\mathbf{X}}]_\tau$ is defined as $\bar{x}_\tau, \tau = 0, 1, \ldots, K_t - 1$. After the temporal convolutional layer, we are ready to put $\bar{\mathbf{X}}$ into the spatial layer. Based on [31], we can design the following transfer functions and neuron:

$$\mathbb{H}(\mathbf{S}, z) = \sum_{t=0}^{K_t-1}\sum_{k=0}^{K-1} h_{k,t}\mathbf{S}^k z^{-t},$$

$$\bar{w}_t = \sigma[w_t] = \sigma\left[\sum_{k=0}^{K-1}\sum_{\tau=0}^{K_t-1} h_{k,\tau}\mathbf{S}^k x_{t-\tau}\right]. \tag{21}$$

where $h_{k,t}$ represents a trainable parameter that, in this context, is a scalar. Accordingly, the graph signal $\bar{w}_t$ from the spatial feature extraction layer (see Fig. 3) is:

$$\bar{w}_t = \mathrm{CReLU}\left[\sum_{k=0}^{K-1}\sum_{\tau=0}^{K_t-1} h_{k,\tau}\mathbf{S}^k \bar{x}_{t-\tau}\right], \tag{22}$$

By combining the temporal and spatial convolutions at each layer, the multiple output channels of the Complex-valued Spatio-Temporal Graph Convolutional neural Network (Cplx-STGCN) layer ($\ell = 1$) are expressed as

$$\bar{\mathbf{W}}_{t,\ell=1} = \mathrm{CReLU}(\mathbf{H} *_{\mathcal{G}} (\mathbf{\Gamma} *_{\mathcal{T}} \mathbf{X}_t)), \tag{23}$$

where $\mathbf{H}$ and $\mathbf{\Gamma}$ are the trainable parameters. We denote (23) by the feature extraction layer.

In the following, we refer to Complex ReLU (namely CReLU) as the simple complex activation that applies separate ReLUs on both of the real and the imaginary part of a neuron, i.e:

$$\mathrm{CReLU}(w) = \mathrm{ReLU}(\Re(w)) + \mathrm{j}\,\mathrm{ReLU}(\Im(w)). \tag{24}$$

Spatio-Temporal GCN are a special case of *multiple features GCN*. Specifically, let $\mathbf{X} = [x^1, \ldots, x^F]$ and let us refer to the multiple channel outputs as $\mathbf{W} = [w^1, \ldots, w^G]$, where F is the number of input features and G is the number of output channels. A layer of multiple features GCN operates as follows:

$$\bar{\mathbf{W}} = \sigma[\mathbf{W}] = \sigma\left[\sum_{k=0}^{K-1} \mathbf{S}^k \times \mathbf{X} \times \mathbf{H}_k\right] = \mathrm{CReLU}(\mathbf{H} *_{\mathcal{G}} \mathbf{X}), \tag{25}$$

where these matrices include $G \times F$ coefficient matrix $\mathbf{H}_k$ with entries $[\mathbf{H}_k]_{fg} = h_k^{fg}$, and $\mathbf{H} *_{\mathcal{G}}$ defines the notion of graph convolution operator based on the concept of spectral graph convolution. In summary, the policy function π_ϕ can be expressed by

$$\pi_\phi = \begin{cases} \bar{\mathbf{W}}_{\ell=1} = \mathrm{CReLU}(\mathbf{H} *_{\mathcal{G}} (\mathbf{\Gamma} *_{\mathcal{T}} \mathbf{X})) \\ \bar{\mathbf{W}}_{\ell+1} = \mathrm{CReLU}\left(\Theta_\ell^{cplx} * \bar{\mathbf{W}}_\ell\right), \ 1 \le \ell \le L-1 \\ a = \mathrm{sigmoid}\left(\Theta_L^{re} * \begin{bmatrix}\Re(\bar{\mathbf{W}}_L) \\ \Im(\bar{\mathbf{W}}_L)\end{bmatrix}\right). \end{cases} \tag{26}$$

where Θ_ℓ^{cplx} represents the complex-valued trainable weight matrix and Θ_L^{re} represents the real-valued trainable weight matrix, and thus $\phi = \{\mathbf{H}, \mathbf{\Gamma}, \Theta_\ell^{cplx}, \Theta_L^{re} | \forall\, 1 \ge \ell \ge L-1\}$. Therefore, the primal optimization in (19) utilizes the stochastic gradient descents to update ϕ of $\pi(\phi)$ to minimize $-Q_{\xi_1}(x', \pi_\phi(x))$ subject to some constraints.

V. Convergence Analysis

A. Convergence of Value Functions

We first focus on the convergence of the value function:

Theorem 1: We make the following assumptions:

1) Each state action pair is sampled an infinite number of times.
2) The Markov decision process is finite.
3) $\gamma \in [0, 1)$.
4) Q values are stored in a lookup table.
5) Q_{ξ_1} and Q_{ξ_2} receive an infinite number of updates.
6) The learning rates satisfy $\eta_t \in [0, 1], \sum_t \eta_t = \infty, \sum_t (\eta_t)^2 < \infty$ with probability 1 and $\forall (\boldsymbol{x}, \boldsymbol{a}) \neq (\boldsymbol{x}_t, \boldsymbol{a}_t), \eta_t = 0$.
7) $\forall r, \mathrm{Var}[r] < \infty$

Then constrained TD3 will converge to the optimal value function Q^*, as defined by the Bellman optimality equation, with probability 1.

The primal-and-dual update only applies to the actor function instead of the policy function. Therefore, the proof is shown in Section A of Supplementary Material [25] applied to the value function of the proposed CRL. In Theorem 1, $Q_{\xi_1}(\boldsymbol{x}_t, \boldsymbol{a}_t)$ converges to $Q^*(\boldsymbol{x}_t, \boldsymbol{a}_t)$.

B. Convergence of Actor Functions

In the following, we analyze the convergence of the actor network after the critic network converges.

It is obvious that $\pi_\phi(\boldsymbol{x})$ is non-convex, non-differentiable due to the activation function, but subderivative. Recall that we have the primal-and-dual method for the actor network:

Assumption V.1: The primal update for (27) can always find the local optimal solution ϕ^* for a $T \gg 1$:

$$\begin{aligned}\phi^* = \arg\min_\phi & -Q^*(\boldsymbol{x}', \pi_\phi(\boldsymbol{x})) + \boldsymbol{\lambda}^\top [\boldsymbol{L}\pi_\phi(\boldsymbol{x}) - \boldsymbol{b}] \\ & + \boldsymbol{\mu}^\top [\boldsymbol{K}\pi_\phi(\boldsymbol{x}) - \boldsymbol{c}]_+ + \frac{\alpha_\lambda}{2} \left\| [\boldsymbol{L}\pi_\phi(\boldsymbol{x}) - \boldsymbol{b}] \right\|_2^2 \\ & + \frac{\alpha_\mu}{2} \left\| [\boldsymbol{K}\pi_\phi(\boldsymbol{x}) - \boldsymbol{c}]_+ \right\|_2^2 \end{aligned} \tag{27}$$

This assumption was already proved by in [32].

Assumption V.2: We define the unaugmented Lagrangian $\mathcal{L}_\phi$ has a saddle point $(\phi^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$.

Definition 1: We define the equality residual $\boldsymbol{r}_\lambda = \boldsymbol{L}\pi_\phi(\boldsymbol{x}) - \boldsymbol{b}$ and the inequality residual $\boldsymbol{r}_\mu = [\boldsymbol{K}\pi_\phi(\boldsymbol{x}) - \boldsymbol{c}]_+$.

Theorem 2: Let $(\phi^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ be a saddle point for $\mathcal{L}_\phi^\alpha$, and define

$$V^k = \frac{1}{\alpha_\lambda} \left\| \boldsymbol{\lambda}^k - \boldsymbol{\lambda}^* \right\|_2^2 + \frac{1}{\alpha_\mu} \left\| \boldsymbol{\mu}^k - \boldsymbol{\mu}^* \right\|_2^2 \tag{28}$$

V^k decreases as

$$V^{k+1} \leq V^k - \alpha_\lambda \left\| \boldsymbol{r}_\lambda^{k+1} \right\|_2^2 - \alpha_\mu \left\| \boldsymbol{r}_\mu^{k+1} \right\|_2^2. \tag{29}$$

Because $V^k \leq V^0$, it follows that $\boldsymbol{\lambda}^k$ and $\boldsymbol{\mu}^k$ are bounded. Iterating the inequality above gives that

$$\alpha_\lambda \sum_{k=0}^{\infty} \left\| \boldsymbol{r}_\lambda^{k+1} \right\|_2^2 + \alpha_\mu \sum_{k=0}^{\infty} \left\| \boldsymbol{r}_\mu^{k+1} \right\|_2^2 \leq V^0 \tag{30}$$

which means that $\boldsymbol{r}_\lambda^{k+1} \to 0$ and $\boldsymbol{r}_\mu^{k+1} \to 0$ as $k \to \infty$.

With Assumption V.1 and Theorem 2, we can conclude that both primal and dual updates converge to a saddle point.

VI. Case Study: Primal-Dual CRL Implementation for SDOPF

We implement the proposed algorithm ((27) and (20)) for the multi-stage stochastic dynamic optimal power flow problem (7). Then, we summarize the detailed steps of the primal and dual updates in Algorithm 1.

A. Primal-Dual SDOPF Formulation

With the above primal-dual framework, we aim to train the constrained policy function $\pi_\phi(\cdot)$ for SDOPF. In particular, we define power generations $\boldsymbol{g}_t^p, \boldsymbol{g}_t^q$, as well as BESS charging power $\boldsymbol{p}_{ch,t}$ and discharging power $\boldsymbol{p}_{dis,t}$ through actions $\boldsymbol{A}_t = [\hat{\boldsymbol{a}}_t, \ldots, \hat{\boldsymbol{a}}_{t+T-1}]$.

$$\begin{aligned} & \hat{\boldsymbol{a}}_t \triangleq [\pi_\phi(\boldsymbol{x}_{t-1})]_t, \hat{\boldsymbol{a}}_t \triangleq [\hat{\boldsymbol{g}}_t^p; \hat{\boldsymbol{g}}_t^q; \hat{\boldsymbol{p}}_{ch,t}; \hat{\boldsymbol{p}}_{dis,t}]^\top, \hat{\boldsymbol{g}}_t^p \triangleq \hat{\boldsymbol{a}}_{t,1}, \\ & \hat{\boldsymbol{g}}_t^q \triangleq \hat{\boldsymbol{a}}_{t,2}, \hat{\boldsymbol{p}}_{ch,t} \triangleq \hat{\boldsymbol{a}}_{t,3}, \hat{\boldsymbol{p}}_{dis,t} \triangleq \hat{\boldsymbol{a}}_{t,4}, \boldsymbol{g}_t^p \triangleq (1 - \hat{\boldsymbol{g}}_t^p)\underline{\boldsymbol{g}}^p + \hat{\boldsymbol{g}}_t^p \overline{\boldsymbol{g}}^p, \\ & \boldsymbol{g}_t^q \triangleq (1 - \hat{\boldsymbol{g}}_t^q)\underline{\boldsymbol{g}}^q + \hat{\boldsymbol{g}}_t^q \overline{\boldsymbol{g}}^q, \boldsymbol{p}_{ch,t} \triangleq \hat{\boldsymbol{p}}_{ch,t} \boldsymbol{p}_{rated}^{ch}, \boldsymbol{p}_{dis,t} \\ & \quad \triangleq \hat{\boldsymbol{p}}_{dis,t} \boldsymbol{p}_{rated}^{dis}. \end{aligned} \tag{31}$$

We also have the predicted voltage magnitudes $|\boldsymbol{v}|$ as:

$$|\hat{\boldsymbol{v}}_t| = [P_\omega(\boldsymbol{x}_{t-1})]_t, |\boldsymbol{v}_t^p| = (1 - |\hat{\boldsymbol{v}}_t|)\underline{\boldsymbol{v}} + |\hat{\boldsymbol{v}}_t| \overline{\boldsymbol{v}}, \tag{32}$$

where $|\boldsymbol{v}_t^p|$ is constrained to be equal to the ground-truth one, i.e., $|\boldsymbol{v}_t|$. Therefore, we replace $|\boldsymbol{v}_t|$ in (7b) and (7c) with $|\boldsymbol{v}_t^p|$, with an additional constraint:

$$|\boldsymbol{v}_t^p| - |\boldsymbol{v}_t| = 0. \tag{33}$$

We introduce dual variables $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ in relation to (7), alongside the augmented penalty parameters $\boldsymbol{\alpha}$:

$$\boldsymbol{\lambda}_t = \begin{bmatrix} \boldsymbol{\lambda}_{1,t} \\ \boldsymbol{\lambda}_{2,t} \\ \vdots \\ \boldsymbol{\lambda}_{4,t} \end{bmatrix}, \boldsymbol{\mu}_t = \begin{bmatrix} \boldsymbol{\mu}_{1,t} \\ \boldsymbol{\mu}_{2,t} \\ \vdots \\ \boldsymbol{\mu}_{6,t} \end{bmatrix}, \boldsymbol{\alpha}_\lambda = \begin{bmatrix} \boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \\ \vdots \\ \boldsymbol{\alpha}_4 \end{bmatrix}, \boldsymbol{\alpha}_\mu = \begin{bmatrix} \boldsymbol{\alpha}_5 \\ \boldsymbol{\alpha}_6 \\ \vdots \\ \boldsymbol{\alpha}_{10} \end{bmatrix}, \tag{34}$$

Furthermore, we reformulate the equality and inequality constraints of (7) in a more compact manner:

$$\boldsymbol{r}_{\lambda,t} = \begin{bmatrix} \mathbf{M}_b \boldsymbol{p}_{dis,t} - \mathbf{M}_b \boldsymbol{p}_{ch,t} + \mathbf{M}_g \boldsymbol{g}_t^p - \boldsymbol{d}_t^p - \Re\{D(\boldsymbol{v}_t^p (\boldsymbol{v}_t)^H \mathbf{Y}^H)\} \\ \mathbf{M}_g \boldsymbol{g}_t^q - \boldsymbol{d}_t^q - \Im\{D(\boldsymbol{v}_t^p (\boldsymbol{v}_t)^H \mathbf{Y}^H)\} \\ \boldsymbol{soc}_t - \boldsymbol{soc}_{t-1} + \frac{\Delta t}{E_{cap}} \left(\eta_{ch} \boldsymbol{p}_{ch,t} - \frac{\boldsymbol{p}_{dis,t}}{\eta_{dis}} \right) \\ |\boldsymbol{v}_t^p| - |\boldsymbol{v}_t| \end{bmatrix} \tag{35}$$

$$r_{\mu,t} = \begin{bmatrix} soc_{t-1} - \frac{\Delta t\left(\eta_{ch} p_{ch,t} - \frac{p_{dis,t}}{\eta_{dis}}\right)}{E_{cap}} - soc_{\max} \\ soc_{\min} + \frac{\Delta t\left(\eta_{ch} p_{ch,t} - \frac{p_{dis,t}}{\eta_{dis}}\right)}{E_{cap}} - soc_{t-1} \\ |v_t^p| - \overline{v} \\ \underline{v} - |v_t^p| \\ |(\mathbf{C}_f v_t) \circ (\mathbf{Y}_f^* v_t^*)| - s_{\max} \\ |(\mathbf{C}_{to} v_t) \circ (\mathbf{Y}_{to}^* v_t^*)| - s_{\max} \end{bmatrix}_+ \tag{36}$$

With the definition of (31), the augmented Lagrangian is:

$$\min_{\phi} \mathcal{L}_\phi = -Q_{x_i'}(x_t, \pi_\phi(x_{t-1})) + \sum_{t=\tau+1}^{\tau+T} \left(\lambda_t^\top r_{\lambda,t} + \mu_t^\top r_{\mu,t} + \left\| \begin{bmatrix} \text{diag}(\alpha_\lambda) & 0 \\ 0 & \text{diag}(\alpha_\mu) \end{bmatrix} \begin{bmatrix} r_{\lambda,t} \\ r_{\mu,t} \end{bmatrix} \right\|_2^2 \right) \tag{37}$$

where λ_t and μ_t are the dual variable vectors, and α_λ and α_μ are positive scalars that penalize the augmented terms. The above problem is different to the optimization problem in (7) due to the following facts:

1) In (35), we substitute g^p, g^q, p_{dis}, and p_{ch} with their associated constraints, $\underline{g}^p \le g_t^p \le \overline{g}^p$, $\underline{g}^q \le g_t^q \le \overline{g}^q$, $0 \le p_{ch,t} \le p_{rated}^{ch}$, and $0 \le p_{dis,t} \le p_{rated}^{dis}$, using (31). $\pi_\phi(x_{t-1})$ is neural network output, sigmoid-constrained to [0, 1].
2) In (7), soc serves as the state variables for solving. Conversely, in (36), soc_i are the provided training samples, used to confine the actions $\pi_\phi(x_{t-1})$.
3) For the voltage magnitude bound, i.e., $|v|$, we need to predict it by a independent GCN given the states x_t, i.e., $|v^p|$. We also need to make a constraint for the predicted voltage magnitude equal to the ground-truth one, i.e., $|v^p| = |v|$. The output of the independent GCN, $|v^p|$, serves as a conduit linking the ground-truth $|v|$ with the power flow equations. This linkage ensures that $|v|$ complies with the power flow equations while remaining proximate to $|v^p|$, which consistently lies within the feasible bounds $[\underline{v}, \overline{v}]$.
4) For instance, when the actions g_t^p and g_t^q result in voltage magnitudes $|v|$ that exceed the feasible bounds $[\underline{v}, \overline{v}]$, the residual $r_{\lambda,t}$ will exhibit significant values for both power flow equations and $|v^p| - |v|$. By minimizing this residual using the policy gradient, actions g_t^p and g_t^q are adjusted to maintain $|v|$ within the stipulated range.

The primal-dual update involves two steps: first, minimizing the Lagrangian function, and then maximizing the dual function as follows:

$$\begin{aligned} \lambda_t^{k+1} &\leftarrow \lambda_t^k + \text{diag}(\alpha_\lambda) r_{\lambda,t}, \ \forall t \in [\tau, \tau+T] \\ \mu_t^{k+1} &\leftarrow \mu_t^k + \text{diag}(\alpha_\mu) r_{\mu,t}, \ \forall t \in [\tau, \tau+T] \end{aligned} \tag{38}$$

where λ_t^{k+1} and μ_t^{k+1} are iteratively updated based on batch samples. Our iterative process combines primal and dual updates to optimize ϕ for $\pi_\phi(\cdot)$ while maintaining the feasibility of both equality and inequality constraints.

Algorithm 1: Constrained Reinforcement Learning for Multi-Stage SDOPF.

1 Initialize critic networks Q_{ξ_1}, Q_{ξ_2} and actor network π_ϕ with random parameters ξ_1, ξ_2, and ϕ;
2 Set target networks $\xi_1' = \xi_1$, $\xi_2' = \xi_2$, $\phi' = \phi$;
3 Initialize replay buffer $\mathcal{B}$, state x_0, and define update periods pu and du;
4 **for** $t = 1 : T$ **do**
  /* This the sampling processing */
5 Select action $A_t = \pi_\phi(x_{t-1})$;
6 Observe reward r_t using Eq. (6);
7 Acquire new state $x_t = \textbf{env}(A_t)$;
8 Store transition (x_{t-1}, A_t, r_t, x_t) in $\mathcal{B}$;
  /* Training processing */
9 Sample a mini-batch of N transitions $\{(x_{n-1}, A_n, r_n, x_n) | n = 1, \cdots, N\}$ from $\mathcal{B}$;
10 $y \leftarrow r_t + \gamma \min_{i=1,2} Q_{\xi_i'}(x_n, \pi_\phi(x_n))$;
11 Update critics: $\xi_1/\xi_2 \leftarrow \arg\min_{\xi_1/\xi_2} \frac{1}{N} \sum (y - Q_{\xi_1/\xi_2}(x_{n-1}, A_n))^2$;
12 **if** *t mod pu* **then**
13 Utilize the deterministic policy gradient to update ϕ: $\phi \leftarrow \phi - \eta \nabla \mathcal{L}_\phi(x_{n-1}, x_n)$, where η represents the learning rate and $\mathcal{L}_\phi$ is articulated in Eq. (37);
14 Update target networks by (14);
15 **if** *t mod du* **then**
16 Update the dual variables by (38).
17 **Function** ***env*** (A_{t+1})
18 Select the initial action, namely, $a_t = [g_t^p; g_t^q; p_{ch,t}; p_{dis,t}]^\top$, from $A_t = [a_t, \ldots, a_{t'}]^\top$, with $t' = t + T - 1$;
19 **for** $i \in \mathcal{B}$ **do**
20 **if** $0 < soc_{i,t} < 1$ **then**
21 $soc_{i,t+1} \leftarrow soc_{i,t} + \frac{\Delta t}{E_{cap}}\left(\eta_{ch} p_{ch,i,t} - \frac{p_{dis,i,t}}{\eta_{dis}}\right)$;
22 **else if** $soc_{i,t} = 1$ **then**
23 $p_{ch,i,t} \leftarrow 0$, $soc_{i,t+1} \leftarrow soc_{i,t} - \frac{\Delta t}{E_{cap}} \frac{p_{dis,i,t}}{\eta_{dis}}$;
24 **else if** $soc_{i,t} = 0$ **then**
25 $p_{dis,i,t} \leftarrow 0$, $soc_{i,t+1} \leftarrow soc_{i,t} + \frac{\Delta t \eta_{ch} p_{ch,i,t}}{E_{cap}}$;
26 Derive v_t by solving Eqs. (7b) and (7c), with a_t fixed;
27 Return $\mathbf{X}_t = [x_t, x_{t-1}, \cdots, x_{t-T+1}]^\top$, where $x_t = [v_t, soc_t]$;

B. Constrained Reinforcement Learning Algorithm

The training process of the constrained reinforcement learning algorithm, described above, is summarized in Algorithm 1. In steps 1–3, we initialize the parameters of double critic networks Q_{ξ_1}, Q_{ξ_2}, double target networks $Q_{\xi_1'}$, $Q_{\xi_2'}$, and the actor networks π_ϕ. Steps 5–8 represent the process of data sampling and storing transition tuple (x_{t-1}, A_t, r_t, x_t). In Steps 9–11, we update the critic networks, and we update the actor network in Steps 12–13 and the dual variables in Steps 15–16. In Steps 19–25, we update the state of charge soc that should be projected within [0, 1]. In Step 26, we fix $g_t^p, g_t^q, p_{ch,t}, p_{dis,t}$ and solve power flow equations to obtain v_t. Specifically, we fixed the power generations on the non-slack buses, i.e. $g_{i,t}^p$ and $g_{i,t}^q$ $\forall i \in \mathcal{G}_n$, and keep the voltage magnitude and angle on the slack bus 1 p.u. and 0, respectively. Then, we utilize the Newton's method to solve v_t.

After training the GCN-policy function, we are ready to implement the GCN policy to forecast the control actions, i.e., $\hat{g}_t^p, \hat{g}_t^q, \hat{p}_{ch,t}, \hat{p}_{dis,t}$, by feeding the previous state measurements x_{t-1}. Then, the active and reactive power generations g_t^p, g_t^q, and the battery charging and discharging powers $p_{ch,t}, p_{dis,t}$ are utilized to control the power systems in real time.

VII. Experimental Results

A. Experimental Setup

Experiments were conducted on the IEEE 14-bus and 30-bus systems, each with two BESSs. The IEEE 14-bus system has BESSs located at Bus 9, while the IEEE 30-bus system has BESSs at Buses 13 and 22. These BESSs have a capacity E_{cap} of 1000 MWh and charge/discharge efficiencies (η_{ch} and η_{dis}) of 0.98. We've set the system to reset renewable energy and demand every hour, using the hourly demand data from the NREL Wind Integration Toolkit for training. The time step is determined to be 18 seconds. This calculation is based on the reinforcement learning system's design to update 200 times and then reset the environment once to accommodate the new demand and renewable energy (60 minutes/200 updates $=$ 18 seconds per update). The SOC is bounded in $[0, 1]$. For training of the constrained DRL we relied on PyTorch and used realistic demand profiles from the Texasgrid. We consider one wind power generator bus in IEEE 14-bus system, and three of them in IEEE 30-bus system, and six of them in the IEEE 118-bus system, simulated using the sample power profiles that were collected by NREL Wind [33]. The training and testing phases for the proposed CRL is conducted across different operating points. Once the policy function is trained, the CRL method can predict optimal control actions for diverse operating points without retraining during the testing process.

We developed and trained all proposed CRL architectures using PyTorch 1.10.0 across all scenarios. We executed all algorithms on a 64-bit Windows operating system, powered by a 2.6 GHz Six-Core Intel Core i7 processor. The system is equipped with a total of 16 GB RAM and an NVIDIA GeForce RTX 2060 graphics card. The training process for the IEEE 14-bus system takes around 1.6 hours to complete for $4 * 10^5$ search iterations, in comparison to the IEEE 30-bus system and the IEEE 118-bus system, which require up to four hours and eight hours respectively for the same number of scenarios. Once the policy training is completed, the testing phase becomes substantially quicker. Specifically, the IEEE 14-bus system is able to process 3000 scenarios in less than 73.5 seconds. On the other hand, the IEEE 30-bus system handles the same amount of scenarios within a maximum time span of 118.7 seconds, whereas the IEEE 118-bus system requires up to 173.2 seconds. We evaluate the performance of the proposed CRL algorithm against both the stochastic OPF (labeled as "STC" in Fig. 4) method presented in [34] and the model predictive control (MPC) based on their optimality and implementation time. As depicted in Fig. 4, even after considering the cumulative computational times of both the offline and online phases, the time required for 3000 time-series operating points–approximately 17 hours–by the stochastic OPF is notably ten times longer than that of the CRL, which takes around 1.6 hours. Take note that the dual update for training does not require gradient descents, enabling rapid computation. Therefore, when compared to unconstrained DRL utilizing the same structure and datasets, the training time for an equivalent epoch remains substantially similar. Besides, we independently assess the constraints on voltage magnitude, a strategy inspired by our prior work with Cplx-GCN, which demonstrated remarkable accuracy in estimating voltage magnitudes [31]. The MSE for voltage magnitudes, when RL converges, are as follows: 8.41×10^{-5} for the IEEE 14-bus system, 8.96×10^{-5} for the IEEE 30-bus system, and 9.54×10^{-5} for the IEEE 118-bus system.

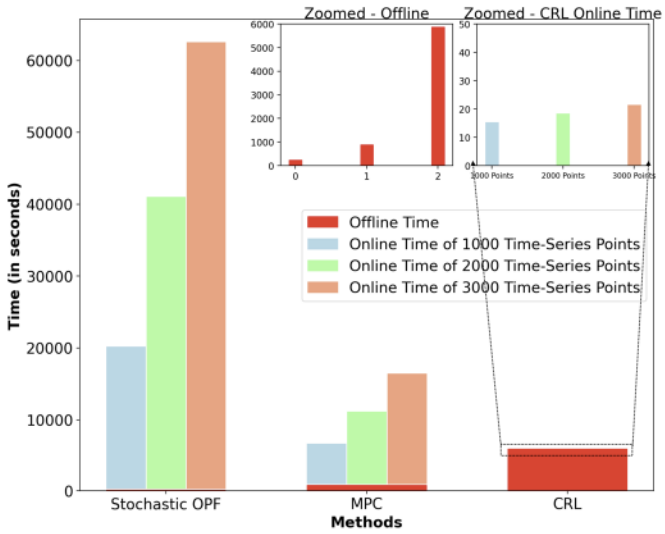
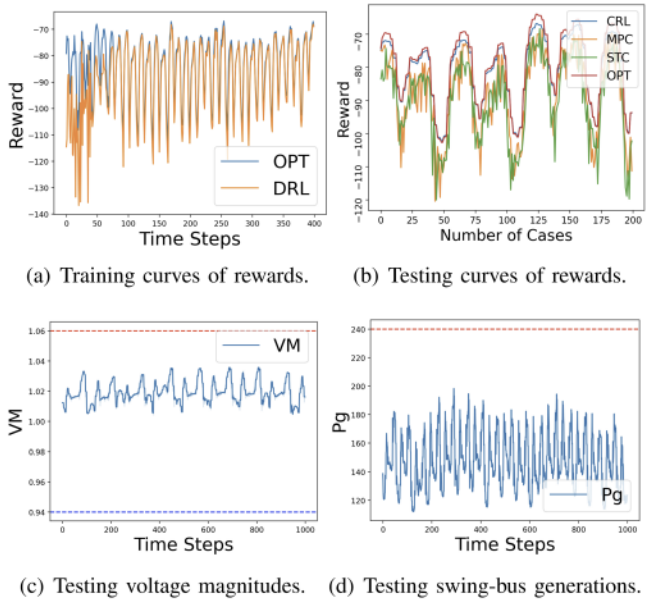


Fig. 4. Illustration of Total Computation time of the proposed CRL (Training + Testing), MPC time, and Stochastic OPF with scenarios.

The design of the actor networks has been meticulously planned as follows. The feed-forward network architecture for the IEEE 14-bus, IEEE 30-bus, and IEEE 118-bus systems consists of a single layer of cplx-STGCN for feature extraction, followed by a layer of Complex-valued Neural Network (cplx-NN), and then a layer of Real-valued Neural Network (real-NN) for output generation. For the cplx-STGCN layer, the output channel is set at 10, and the STGCN operates at an order of $K = 5$. The cplx-NN layer comprises 512 neurons, while the real-NN layer is equipped with 1024 neurons. The critic network, in contrast, has a comparatively straightforward task - to regress the long-term discounted reward. Consequently, it only requires a relatively simple design, consisting of a three-layer real-NN with each layer housing 256 neurons.

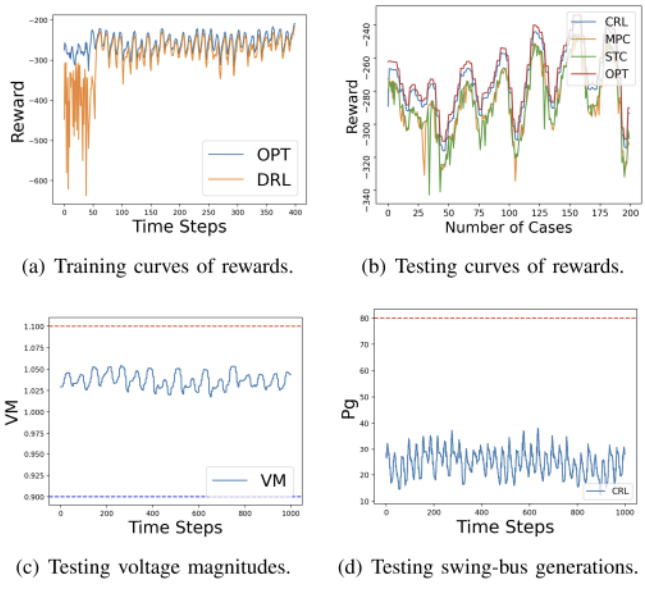
The reinforcement learning setting is defined as follows: the buffer size is 500, the discount factor for the reward is 0.99, the rate at which the target network updates is 0.005, and the frequency of policy updates with delay is once per 2 iterations. Both network parameters are optimized using Adam with a learning rate of 10^{-3}. The networks are trained after each time step using a mini-batch of 100 transitions, sampled uniformly from the replay buffer, which stores the entire history of transition tuples (x_{t-1}, A_t, r_t, x_t). It should be noted that the actor networks in this design incorporate two independent



(a) Training curves of rewards. (b) Testing curves of rewards.

(c) Testing voltage magnitudes. (d) Testing swing-bus generations.

Fig. 5. (a) and (b) depict the training and testing curves of the proposed CRL, along with the associated optimal rewards determined by the optimization method, denoted as "OPT". (c) and (d) showcase the testing curves corresponding to the average voltage magnitudes and swing-bus generations within the IEEE 14-bus system.



(a) Training curves of rewards. (b) Testing curves of rewards.

(c) Testing voltage magnitudes. (d) Testing swing-bus generations.

Fig. 6. Training and testing trajectories for the proposed CRL are illustrated in figures (a) and (b), accompanied by the optimal rewards calculated through the optimization method, labelled as "OPT". Figures (c) and (d) display the testing curves related to the mean voltage magnitudes and swing-bus productions within the IEEE 30-bus framework.

GCN, one for active power generation and another for reactive power generation. This design has the potential to improve the performance of the proposed CRL system by reducing the action spaces for the GCN models. The baseline method pertains to the identical optimization problem as that of CRL, specifically, Problem (7). This issue is addressed utilizing readily available solvers, including interior point methods [35].

B. Baseline Methods

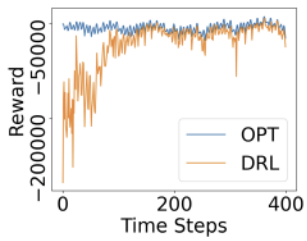
Before evaluating our proposed approach, we define several baseline methods for comparison. First of all, we consider two well-known DRL methods with Deep Q-Network (DQN) and Deep Deterministic Policy Gradient (DDPG). Secondly, we compare the GCN-policy function with the fully connected neural networks (FNN), the convolutional neural networks (CNN) and Graph neural networks (GNN). Thirdly, we compare the proposed method with the optimization method knowing the future information. We also extend the existing learning-based OPF methods, i.e., the penalty method [13] and DC3 [15], to the reinforcement learning setting and compare them with the proposed algorithm. The percentage optimality gap, "%gap", is calculated as $\%\,\text{gap} = 100 \times \frac{z^{\text{UB}} - z^{\text{LB}}}{z^{\text{UB}}}$, where z^{UB} is the objective cost of an OPF feasible solution derived from a relaxation, and z^{LB} is the optimal objective cost of this relaxation.

C. Learning Curves and Optimal Curves

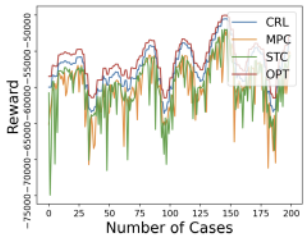
1) IEEE 14-Bus System: The learning curves in Fig. 5(a) show the rewards, represented on the y-axis, at each time step on the x-axis. Our results are compared with an oracle solution that knows the future in the time horizon and performs the constrained optimization. The results indicate that our learning policy is very close to the optimal curves, demonstrating the effectiveness of our method in forecasting optimal actions without future information. The average gap between our DRL approach and the optimization method is only 2.52%. Upon completion of the training, the policy function is subjected to a test involving unseen scenarios, encompassing 2×10^4 samples. The ensuing results, which are based on a selection of 200 sample outcomes for graphical representation, are depicted in Fig. 5(b). These results demonstrate that the policy tends to select actions that approach optimality, with a relatively small optimality gap of merely 2.25%. However, the Stochastic OPF and MPC have larger gap with the optimality.

Applying the policy trained through the constrained DRL, we also tested the performance injecting new demand profiles in the future samples, to check if the solutions are feasibile. Both the voltage magnitudes and power generations are feasible. We observe that infeasible actions are likely to happen if the training does not include dual updates. For example, when the values of $g_{i,t}, i \in \mathcal{G}_n$ of the (non-slack buses) predicted by the RL is very small, the values of $g_{i,t}, i \in \mathcal{G}_s$ in the slack bus will violate the upper bound constraint. Furthermore, the constraints of voltage magnitudes are easily violated if the reactive power injections are either too large or insufficient. These facts are illustrated in Fig. 5(c) and 5(d), showing the swing-bus generation $g_{i,t}, i \in \mathcal{G}_s$ and the average voltage magnitude $\frac{1}{N}\sum_{i\in\mathcal{N}} |v_t|$, demonstrating that the proposed CRL is always feasible.

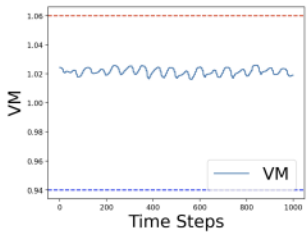
2) Analysis of IEEE 30-Bus and 118-Bus Systems: In our investigation into the CRL implementation, we present the learning curves for both the IEEE 30-bus and 118-bus systems, shown in Figs. 6(a) and 7(a), respectively. For the 30-bus system with BESSs at Bus 13 and Bus 22, the average optimal gap



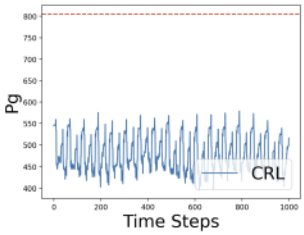
(a) Training curves of rewards.



(b) Testing curves of rewards.

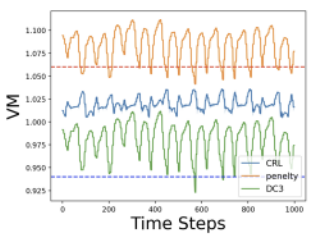


(c) Testing voltage magnitudes.

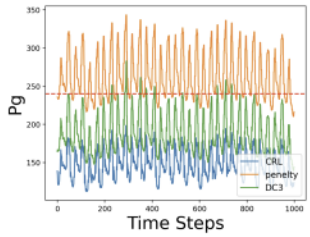


(d) Testing swing-bus generations.

Fig. 7. Training and testing trajectories for the proposed CRL are illustrated in figures (a) and (b), accompanied by the optimal rewards calculated through the optimization method, labelled as "OPT". Figures (c) and (d) display the testing curves related to the mean voltage magnitudes and swing-bus productions within the IEEE 118-bus framework.

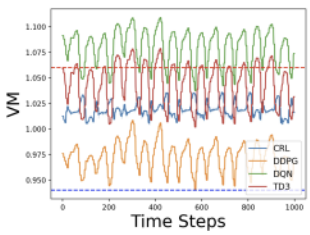


(a) Testing voltage magnitudes.

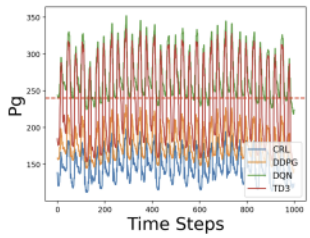


(b) Testing wing-bus generations.

Fig. 8. Comparison of the penalty method, the DC3 method and the proposed CRL in the IEEE 14-bus system.



(a) Testing voltage magnitudes.



(b) Testing wing-bus generations.

Fig. 9. Comparison of DDPG, DQN, TD3 and the proposed CRL in the IEEE 14-bus system.

between our proposed algorithm and the optimization method with future information is 3.39%. In the case of the 118-bus system, equipped with batteries at Buses 5, 23, 41, 58, 75, 96, and 114, the learning curves depict a reduction in the average optimal gap, resulting in a 4.05% difference. Post-training, we evaluated the policy functions against unseen scenarios using 2×10^4 samples for the 30-bus and a similar set for the 118-bus. Selected results from a pool of 200 samples for each system are presented in Figs. 6(b) and 7(b). These results illustrate that the policies lean towards selecting near-optimal actions without depending on future information. The implementation outcomes further underscore the superior performance of the proposed CRL in comparison to both the Stochastic OPF and MPC. To further validate our approach, we assessed the feasibility of generation and voltage magnitudes. The results for the 30-bus system are highlighted in Fig. 6(c) and 6(d), while the 118-bus system's results are showcased in Figs. 7(c) and 7(d). Across both systems, our data emphasizes consistent feasibility, with the larger 118-bus system achieving a 100% feasibility rate.

D. Feasibility Comparison

We compared the proposed primal-and-dual CRL with two baselines, namely, the penalty method [13] and DC3 [15]. Typically, the penalty method includes a violation penalty as a rectified linear unit function included in the reward. For example, the violation penalty for voltage magnitude is given by

$$r_{vm} = -\left([|v| - \overline{v}]_+ + [\underline{v} - |v|]_+\right). \tag{39}$$

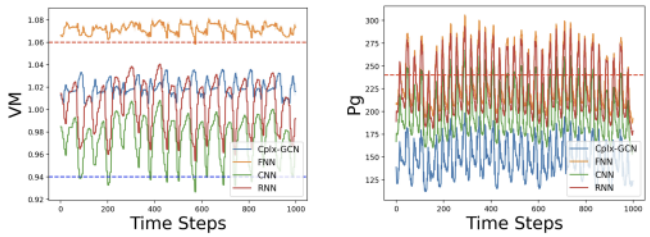
The DC3 method considers the equality and inequality constraints as $\|\cdot\|_2^2$ for the objectives, i.e.,

$$\min_{\phi} -Q^*(x', \pi_\phi(x)) + \frac{\alpha_\lambda}{2}\|[L\pi_\phi(x) - b]\|_2^2 + \frac{\alpha_\mu}{2}\|[K\pi_\phi(x) - c]_+\|_2^2,$$

where these norm terms are included in the policy objectives without the primal-and-dual update process. Fig. 8(a) and 8(b) show that the proposed constrained DRL can ensure 100% feasibility, whereas the traditional DRL has only 80.78% feasibility rate. The penalty method provides 13.50% feasibility rate and the DC3 method has 96.17% feasibility rate for the voltage magnitudes, and 29.46% feasibility rate and 98.56% feasibility rate for the swing-bus generations. This indicates that the key element to enforce the 100% feasibility lies in the dual updates.

E. Comparison of Reinforcement Learning

We also compare the proposed CRL with the existing deep reinforcement learning methods, i.e., DQN, DDPG and TD3 without the primal-and-dual updates, considering the IEEE-14 bus system. The proposed CRL method showcases higher rewards, i.e., 2.25%, whereas TD3 converges to 5.04%, DQN converges to 7.33% and DDPG converges to 8.10%. The testing voltage magnitude curves and testing power generation curves are in Fig. 9(a) and 9(b). In particular, TD3, DQN and DDPG have 52.39%, 14.95% and 100% power generation feasibility rate, and 70.89%, 17.72% and 99.27% voltage magnitude feasibility rate, while the proposed CRL has both 100% power generation



(a) Testing voltage magnitudes. (b) Testing wing-bus generations.

Fig. 10. Comparison of FNN, CNN, RNN and Cplx-GCN policies in the IEEE 14-bus system.

feasibility rate and 100% voltage magnitude feasibility rate. Furthermore, the proposed method can control voltage magnitude profiles with less variations around 1 p.u.

F. Comparison of Policy Neural Networks

We run a set of numerical experiments to compare the Cplx-STGCN policy neural networks with different architectures for the policy neural networks, namely a Fully connected NN (FNN), a convolutional NN (CNN), and a Recursive NN (RNN). The test average optimal gaps of FNN, CNN and RNN are 4.99%, more than double compared to the Cplx-STGCN (the gap is 2.25%); this experiment clearly illustrates the ability of Cplx-STGCN to better capture spatio-temporal features of the state in the policy function. In Fig. 10(a) and 10(b), FNN, CNN and RNN have 65.41%, 89.45% and 63.37% power generation feasibility rate, and 1.09%, 92.60% and 100% voltage magnitude feasibility rate. In contrast, the proposed CRL is always feasible.

VIII. Conclusion

In this study, we introduced a constrained reinforcement learning method, utilizing prime-dual decomposition and double-Q learning for updating policy and critic networks in multi-stage SDOPF problems. We further proved the convergence of the proposed CRL under mild assumptions. Numerical results reveal that actions selected via our policy closely approximate those of a future-aware oracle OPF, ensuring 100% feasibility. Compared with other RL methods, the proposed CRL achieves the higher reward. Compared with other neural networks, the proposed Cplx-STGCN policy function has better performance in extracting the spatio-temporal features for voltage phasors. Besides, the feasible rate of the proposed CRL is higher than other constrained neural network methodologies. For future work, we aim to explore distributed multi-agent reinforcement learning to manage the challenge of controlling a significantly larger network.

Appendix

The proof of Theorem 2 is provided as follows.

Proof: Since $(\phi^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ is a saddle point for unaugmented Lagrangian$\mathcal{L}_\phi$, we have

$$\mathcal{L}_\phi(\phi^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \leq \mathcal{L}_\phi(\phi^{k+1}, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \tag{40}$$

Using $\boldsymbol{L}\pi_{\phi^*}(\boldsymbol{x}) - \boldsymbol{b} = \boldsymbol{0}$ and $[\boldsymbol{K}\pi_{\phi^*}(\boldsymbol{x}) - \boldsymbol{c}]_+ = \boldsymbol{0}$, the left sides is $p^* = -Q^*(\boldsymbol{x}, \pi_{\phi^*}(\boldsymbol{x}))$. With $p^{k+1} = -Q^*(\boldsymbol{x}, \pi_{\phi^{k+1}}(\boldsymbol{x}))$, this can be written as

$$p^* \leq p^{k+1} + \boldsymbol{\lambda}^{*\top}[\boldsymbol{L}\pi_\phi(\boldsymbol{x}) - \boldsymbol{b}] + \boldsymbol{\mu}^{*\top}[\boldsymbol{K}\pi_\phi(\boldsymbol{x}) - \boldsymbol{c}]_+ \tag{41}$$

We can derive the first key inequality:

$$p^* \leq p^{k+1} + \boldsymbol{\lambda}^{*\top}\boldsymbol{r}_\lambda^{k+1} + \boldsymbol{\mu}^{*\top}\boldsymbol{r}_\mu^{k+1} \tag{42}$$

By definition, ϕ^{k+1} minimizes $\mathcal{L}_\phi^\alpha(\phi, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)$. The optimality condition is

$$\begin{aligned}
&0 \in \partial(-Q^*(\pi_{\phi^{k+1}}(\boldsymbol{x}))\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) + \boldsymbol{L}^\top\boldsymbol{\lambda}^k\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) + \alpha_\lambda\boldsymbol{L}^\top \\
&(\boldsymbol{L}\pi_{\phi^{k+1}}(\boldsymbol{x}) - \boldsymbol{b})\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) + \boldsymbol{K}^\top\boldsymbol{D}^{(k+1)}\boldsymbol{\mu}^k\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) \\
&+ \alpha_\mu\boldsymbol{K}^\top\boldsymbol{D}^{(k+1)}(\boldsymbol{K}\pi_{\phi^{k+1}}(\boldsymbol{x}) - \boldsymbol{c})_+\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) \\
&= \partial(-Q^*(\pi_{\phi^{k+1}}(\boldsymbol{x}))\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) + \boldsymbol{L}^\top\left(\boldsymbol{\lambda}^k + \alpha_\lambda(\boldsymbol{L}\pi_{\phi^{k+1}}(\boldsymbol{x})\right. \\
&\left.-\boldsymbol{b})\right)\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) + \boldsymbol{K}^\top\boldsymbol{D}^{(k+1)}\left(\boldsymbol{\mu}^k + \alpha_\mu(\boldsymbol{K}\pi_{\phi^{k+1}}(\boldsymbol{x})\right. \\
&\left.-\boldsymbol{c})_+\right)\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) \\
&= \partial(-Q^*(\pi_{\phi^{k+1}}(\boldsymbol{x}))\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) + \boldsymbol{L}^\top\boldsymbol{\lambda}^{k+1}\partial\pi_{\phi^{k+1}}(\boldsymbol{x}) \\
&+ \boldsymbol{K}^\top\boldsymbol{D}^{(k+1)}\boldsymbol{\mu}^{k+1}\partial\pi_{\phi^{k+1}}(\boldsymbol{x})
\end{aligned} \tag{43}$$

where $\boldsymbol{D}^{(k+1)}$ is a diagonal matrix

$$\boldsymbol{D}_{ii}^{(k+1)} = \begin{cases} 1 & [\boldsymbol{K}]_i^\top[\pi_{\phi^{k+1}}(\boldsymbol{x})]_i > \boldsymbol{c}_i \\ 0 & [\boldsymbol{K}]_i^\top[\pi_{\phi^{k+1}}(\boldsymbol{x})]_i \leq \boldsymbol{c}_i \end{cases} \tag{44}$$

Due to (43), we have

$$\begin{aligned}
0 \in \partial_{\phi^{k+1}}\big(&-Q^*(\pi_{\phi^{k+1}}(\boldsymbol{x})) + (\boldsymbol{\lambda}^{k+1})^\top\boldsymbol{L}\pi_{\phi^{k+1}}(\boldsymbol{x}) \\
&+ (\boldsymbol{\mu}^{k+1})^\top\boldsymbol{D}^{(k+1)}\boldsymbol{K}\pi_{\phi^{k+1}}(\boldsymbol{x})\big),
\end{aligned} \tag{45}$$

which implies that ϕ^{k+1} minimizes:

$$(-Q^*(\pi_\phi(\boldsymbol{x})) + (\boldsymbol{\lambda}^{k+1})^\top\boldsymbol{L}\pi_\phi(\boldsymbol{x}) + (\boldsymbol{\mu}^{k+1})^\top\boldsymbol{D}^{(k+1)}\boldsymbol{K}\pi_\phi(\boldsymbol{x}). \tag{46}$$

It follows

$$\begin{aligned}
&(-Q^*(\pi_{\phi^{k+1}}(\boldsymbol{x}))) + (\boldsymbol{\lambda}^{k+1})^\top\boldsymbol{L}\pi_{\phi^{k+1}}(\boldsymbol{x}) \\
&\quad + (\boldsymbol{\mu}^{k+1})^\top\boldsymbol{D}^{(k+1)}\boldsymbol{K}\pi_{\phi^{k+1}}(\boldsymbol{x}) \leq (-Q^*(\pi_{\phi^*}(\boldsymbol{x}))) \\
&\quad + \boldsymbol{\lambda}^{k+1}\boldsymbol{L}\pi_{\phi^*}(\boldsymbol{x}) + \boldsymbol{\mu}^{k+1}\boldsymbol{D}^{(k+1)}\boldsymbol{K}\pi_{\phi^*}(\boldsymbol{x})
\end{aligned} \tag{47}$$

Using $\boldsymbol{L}\pi_{\phi^*}(\boldsymbol{x}) = \boldsymbol{b}$ and $\boldsymbol{K}\pi_{\phi^*}(\boldsymbol{x}) \preceq \boldsymbol{c}$, we can obtain the second key inequation:

$$\begin{aligned}
&p^{k+1} - p^* \\
&\leq -(\boldsymbol{\lambda}^{k+1})^\top\boldsymbol{r}_\lambda^{k+1} - (\boldsymbol{\mu}^{k+1})^\top\boldsymbol{D}^{(k+1)}\boldsymbol{K}(\pi_{\phi^{k+1}}(\boldsymbol{x}) - \pi_{\phi^*}(\boldsymbol{x})) \\
&\leq -(\boldsymbol{\lambda}^{k+1})^\top\boldsymbol{r}_\lambda^{k+1} - (\boldsymbol{\mu}^{k+1})^\top\boldsymbol{D}^{(k+1)}(\boldsymbol{K}\pi_{\phi^{k+1}}(\boldsymbol{x}) - \boldsymbol{c}) \\
&= -(\boldsymbol{\lambda}^{k+1})^\top\boldsymbol{r}_\lambda^{k+1} - (\boldsymbol{\mu}^{k+1})^\top(\boldsymbol{K}\pi_{\phi^{k+1}}(\boldsymbol{x}) - \boldsymbol{c})_+ \\
&= -(\boldsymbol{\lambda}^{k+1})^\top\boldsymbol{r}_\lambda^{k+1} - (\boldsymbol{\mu}^{k+1})^\top\boldsymbol{r}_\mu^{k+1}
\end{aligned} \tag{48}$$

Adding (42) and (48), regrouping terms, and multiplying through by 2 gives

$$2(\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^*)^\top \boldsymbol{r}_\lambda^{k+1}+2(\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^*)^\top \boldsymbol{r}_\mu^{k+1}\le 0 \tag{49}$$

We begin by rewriting the first term. Substituting $\boldsymbol{\lambda}^{k+1}=\boldsymbol{\lambda}^k+\alpha_\lambda \boldsymbol{r}_\lambda^{k+1}$ and $\boldsymbol{\mu}^{k+1}=\boldsymbol{\mu}^k+\alpha_\mu \boldsymbol{r}_\mu^{k+1}$:

$$\begin{aligned}&2(\boldsymbol{\lambda}^k-\boldsymbol{\lambda}^*)^\top \boldsymbol{r}_\lambda^{k+1}+\alpha_\lambda\left\|\boldsymbol{r}_\lambda^{k+1}\right\|_2^2+\alpha_\lambda\left\|\boldsymbol{r}_\lambda^{k+1}\right\|_2^2\\&+2(\boldsymbol{\mu}^k-\boldsymbol{\mu}^*)^\top \boldsymbol{r}_\mu^{k+1}+\alpha_\mu\left\|\boldsymbol{r}_\mu^{k+1}\right\|_2^2+\alpha_\mu\left\|\boldsymbol{r}_\mu^{k+1}\right\|_2^2\end{aligned} \tag{50}$$

and substituting $\boldsymbol{r}_\lambda^{k+1}=\frac{1}{\alpha_\lambda}(\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^k)$ and $\boldsymbol{r}_\mu^{k+1}=\frac{1}{\alpha_\mu}(\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^k)$ in the first two terms gives

$$\begin{aligned}&\frac{2}{\alpha_\lambda}(\boldsymbol{\lambda}^k-\boldsymbol{\lambda}^*)^\top(\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^k)+\frac{1}{\alpha_\lambda}\left\|\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^k\right\|_2^2\\&+\alpha_\lambda\left\|\boldsymbol{r}_\lambda^{k+1}\right\|_2^2+\frac{2}{\alpha_\mu}(\boldsymbol{\mu}^k-\boldsymbol{\mu}^*)^\top(\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^k)\\&+\frac{1}{\alpha_\mu}\left\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^k\right\|_2^2+\alpha_\mu\left\|\boldsymbol{r}_\mu^{k+1}\right\|_2^2\end{aligned} \tag{51}$$

Since $\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^k=(\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^*)-(\boldsymbol{\lambda}^k-\boldsymbol{\lambda}^*)$ and $\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^k=(\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^*)-(\boldsymbol{\mu}^k-\boldsymbol{\mu}^*)$, this can be written as

$$\begin{aligned}&\frac{1}{\alpha_\lambda}\left(\left\|\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^*\right\|_2^2-\left\|\boldsymbol{\lambda}^k-\boldsymbol{\lambda}^*\right\|_2^2\right)+\alpha_\lambda\left\|\boldsymbol{r}_\lambda^{k+1}\right\|_2^2\\&+\frac{1}{\alpha_\mu}\left(\left\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^*\right\|_2^2-\left\|\boldsymbol{\mu}^k-\boldsymbol{\mu}^*\right\|_2^2\right)+\alpha_\mu\left\|\boldsymbol{r}_\mu^{k+1}\right\|_2^2.\end{aligned} \tag{52}$$

Therefore, we can obtain

$$V^{k+1}\le V^k-\alpha_\lambda\left\|\boldsymbol{r}_\lambda^{k+1}\right\|_2^2-\alpha_\mu\left\|\boldsymbol{r}_\mu^{k+1}\right\|_2^2 \tag{53}$$

This states that V^k decreases in each iteration by an amount that only depends on the norm of the residual. Because $V^k\le V^0$, it follows that $\boldsymbol{\lambda}^k$ and $\boldsymbol{\mu}^k$ are bounded. Iterating the inequality above gives that

$$\alpha_\lambda\sum_{k=0}^{\infty}\left\|\boldsymbol{r}_\lambda^{k+1}\right\|_2^2+\alpha_\mu\sum_{k=0}^{\infty}\left\|\boldsymbol{r}_\mu^{k+1}\right\|_2^2\le V^0 \tag{54}$$

which means that $\boldsymbol{r}_\lambda^{k+1}\to 0$ and $\boldsymbol{r}_\mu^{k+1}\to 0$ as $k\to\infty$.

This completes the proof. ■

Together with Assumption A.1 and Lemma 1, both the primal and dual variables converge into a saddle point $(\phi^*,\boldsymbol{\lambda}^*,\boldsymbol{\mu}^*)$.

ACKNOWLEDGMENT

Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors of this work.

REFERENCES

[1] T. Wu, A. Scaglione, and D. Arnold, "Constrained reinforcement learning for stochastic dynamic optimal power flow control," in *Proc. IEEE Power Energy Gen. Meeting*, 2023, pp. 1–5.

[2] J. Machowski, Z. Lubosny, J. W. Bialek, and J. R. Bumby, *Power System Dynamics: Stability and Control*. Hoboken, NJ, USA: Wiley, 2020.

[3] X. Chen, G. Qu, Y. Tang, S. Low, and N. Li, "Reinforcement learning for selective key applications in power systems: Recent advances and future challenges," *IEEE Trans. Smart Grid*, vol. 13, no. 4, pp. 2935–2958, Jul. 2022.

[4] S. Gill, I. Kockar, and G. W. Ault, "Dynamic optimal power flow for active distribution networks," *IEEE Trans. Power Syst.*, vol. 29, no. 1, pp. 121–131, Jan. 2014.

[5] Y. Guo, K. Baker, E. Dall'Anese, Z. Hu, and T. H. Summers, "Data-based distributionally robust stochastic optimal power flow–Part I: Methodologies," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1483–1492, Mar. 2019.

[6] V.-H. Bui, A. Hussain, and H.-M. Kim, "Double deep Q-learning-based distributed operation of battery energy storage system considering uncertainties," *IEEE Trans. Smart Grid*, vol. 11, no. 1, pp. 457–469, Jan. 2020.

[7] T. Ding, Z. Zeng, J. Bai, B. Qin, Y. Yang, and M. Shahidehpour, "Optimal electric vehicle charging strategy with Markov decision process and reinforcement learning technique," *IEEE Trans. Ind. Appl.*, vol. 56, no. 5, pp. 5811–5823, Sep./Oct. 2020.

[8] T. Yong and R. H. Lasseter, "Stochastic optimal power flow: Formulation and solution," in *Proc. IEEE Power Eng. Soc. Summer Meeting*, 2000, pp. 237–242.

[9] J. Liang, G. K. Venayagamoorthy, and R. G. Harley, "Wide-area measurement based dynamic stochastic optimal power flow control for smart grids with high variability and uncertainty," *IEEE Trans. Smart Grid*, vol. 3, no. 1, pp. 59–69, Mar. 2012.

[10] J. Liang, D. D. Molina, G. K. Venayagamoorthy, and R. G. Harley, "Two-level dynamic stochastic optimal power flow control for power systems with intermittent renewable generation," *IEEE Trans. Power Syst.*, vol. 28, no. 3, pp. 2670–2678, Aug. 2013.

[11] X. Wu, X. Hu, X. Yin, and S. J. Moura, "Stochastic optimal energy management of smart home with PEV energy storage," *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 2065–2075, May 2018.

[12] O. Mégel, J. L. Mathieu, and G. Andersson, "Hybrid stochastic-deterministic multiperiod DC optimal power flow," *IEEE Trans. Power Syst.*, vol. 32, no. 5, pp. 3934–3945, Sep. 2017.

[13] X. Pan, M. Chen, T. Zhao, and S. H. Low, "DeepOPF: A feasibility-optimized deep neural network approach for AC optimal power flow problems," *IEEE Syst. J.*, vol. 17, no. 1, pp. 673–683, Mar. 2023.

[14] T. Wu, Y.-J. A. Zhang, and S. Wang, "Deep learning to optimize: Security-constrained unit commitment with uncertain wind power generation and BESSs," *IEEE Trans. Sustain. Energy*, vol. 13, no. 1, pp. 231–240, Jan. 2022.

[15] P. L. Donti, D. Rolnick, and J. Z. Kolter, "DC3: A learning method for optimization with hard constraints," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–17.

[16] D. Owerko, F. Gama, and A. Ribeiro, "Unsupervised optimal power flow using graph neural networks," 2022, *arXiv:2210.09277*.

[17] Y. Liu, A. Halev, and X. Liu, "Policy learning with constraints in model-free reinforcement learning: A survey," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 4508–4515.

[18] D. Ding, K. Zhang, J. Duan, T. Başar, and M. R. Jovanović, "Convergence and sample complexity of natural policy gradient primal-dual methods for constrained MDPs," 2022, *arXiv:2206.02346*.

[19] D. Ding, K. Zhang, T. Basar, and M. Jovanovic, "Natural policy gradient primal-dual method for constrained Markov decision processes," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 8378–8390.

[20] S. Paternain, M. Calvo-Fullana, L. F. O. Chamon, and A. Ribeiro, "Learning safe policies via primal-dual methods," in *Proc. IEEE 58th Conf. Decis. Control*, 2019, pp. 6491–6497.

[21] S. Qiu, X. Wei, Z. Yang, J. Ye, and Z. Wang, "Upper confidence primal-dual reinforcement learning for CMDP with adversarial loss," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 15277–15287.

[22] A. R. Sayed, C. Wang, H. I. Anis, and T. Bi, "Feasibility constrained online calculation for real-time optimal power flow: A convex constrained deep reinforcement learning approach," *IEEE Trans. Power Syst.*, vol. 38, no. 6, pp. 5215–5227, 2023, doi: 10.1109/TPWRS.2022.3220799.

[23] Z. Yan and Y. Xu, "Real-time optimal power flow: A Lagrangian based deep reinforcement learning approach," *IEEE Trans. Power Syst.*, vol. 35, no. 4, pp. 3270–3273, Jul. 2020.

[24] M. M. Hosseini and M. Parvania, "On the feasibility guarantees of deep reinforcement learning solutions for distribution system operation," *IEEE Trans. Smart Grid*, vol. 14, no. 2, pp. 954–964, Mar. 2023.

[25] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596. [Online]. Available: https://proceedings.mlr.press/v80/fujimoto18a/fujimoto18a-supp.pdf

[26] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[27] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, 1988.
[28] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.
[29] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 2094–2100.
[30] R. Ramakrishna and A. Scaglione, "Grid-graph signal processing (grid-GSP): A graph signal processing framework for the power grid," *IEEE Trans. Signal Process.*, vol. 69, pp. 2725–2739, 2021.
[31] T. Wu, A. Scaglione, and D. Arnold, "Complex-value spatio-temporal graph convolutional neural networks and its applications to electric power systems AI," 2022, *arXiv:2208.08485*.
[32] F. Zou, L. Shen, Z. Jie, W. Zhang, and W. Liu, "A sufficient condition for convergences of adam and RMSProp," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11119–11127.
[33] C. Draxl, A. Clifton, B.-M. Hodge, and J. McCaa, "The wind integration national dataset (WIND) toolkit," *Appl. Energy*, vol. 151, pp. 355–366, 2015.
[34] B. Analui and A. Scaglione, "A dynamic multistage stochastic unit commitment formulation for intraday markets," *IEEE Trans. Power Syst.*, vol. 33, no. 4, pp. 3653–3663, Jul. 2018.
[35] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 12–19, Feb. 2011.