A Deep Reinforcement Learning Approach for Optimal Power Flow

Heng Liang
Department of Information Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong
lh021@ie.cuhk.edu.hk

Shuai Mao
Department of Mechanical and Automation
The Chinese University of Hong Kong
Shatin, Hong Kong
1155160617@link.cuhk.edu.hk

Abstract

The increasing penetration of distributed energy resources features the future power networks with highly fluctuations, posing the urgent needs for fast controlling of the grids. To improve the solving speed and return an optimal setpoint for the operating of power network in a short timescale, we propose a deep reinforcement learning control method to solve the optimal power flow (OPF) problem. The design of our power environment will be elaborated. We compare the performances of four SOTA RL algorithms including TD3, PPO, SAC, DDPG in our environment. Consequently, TD3 has the best performance. Then, we compare the TD3 agent with a traditional gradient-based control scheme and the result shows that our RL controller can provide a safer setpoint for power system. The code is available at https://github.com/henrycliang/RL_OPF.

1 Introduction

Optimal power flow (OPF) is a powerful tool to determine the best operating point for the power system under the constraints of physical laws. Given the fixed substation voltages and multiple bus power consumptions, OPF seeks to minimizing some certain costs, e.g. generation cost, power loss, by adjusting the power injections in generation buses while not to violate the voltage bounds and the thermal limits of transmission lines. However, the growing size of power networks and increasing penetrations of renewable energy generation (e.g. rooftop photovoltaics, wind power) and controllable loads (e.g. electric vehicles) make it become a challenging problem. They feature the the properties of highly active, fast-changing and distributed, posing the requirements for fast and scalable OPF solvers.

The OPF problem is known to be hard to solve because of the nonconvexity and nonlinearity introduced by bus injection model (BIM) and brand flow model (BFM)(11) and there are mainly two trajectories to handle it. On the one hand, they convexify the problem through the techniques of relaxation (7) and then off-the-shelf convex optimization solver can be applied to its convex equivalents such as SDP relaxation and SOCP relaxation. This method relies on the assumption that the relaxation is exact even though it holds in most of cases in tree networks(12). Otherwise, it only gives a lower bound on the optimal solution. On the other hand, they simplify the power equations through the linear approximation(8). It assumes the line loss are small and can be ignored, leading to the absence of terms associated with line currents and of course the non-convex terms. However, this method expose the power networks at the risk of voltage violation since it optimistically estimates voltages if lacking of feedbacks from the real power system. The author in (10) reveals that interactions with real power system can improve the accuracy of the linear approximation while not sacrifice the speed and scalability of the solvers. This motivates us to reformulate the OPF problem by incorporating feedbacks, or namely, rewards from the power system, hence, a deep reinforcement learning solver can be developed based on it.

This paper is organized as follows: we firstly review related works in the next part. Section II describes the OPF problem and then proposes a reinforcement learning solution strategy. Section III presents the results of our experiments. Section IV concludes this paper.

1.1 Related Work

Varied works have been done to tackle the OPF problem. The authors in (1) firstly formulates the OPF problem as SDP programming and the SOCP programming is developed in (7). These methods relax the rank one constraints for SDP and equality constraints for SOCP and the global optimal can be obtained if the assumption of exact relaxation holds. (12) reveals the conditions that the relaxation is exact in tree networks. However, these methods are centralized, needing heavy computations and communications, which are not efficient for large scale power network. (15) and (14) decompose the solver from centralized to distributed by using ADMM, every bus solves its local problem through neighboring communications. (9) proposes an online gradient method for the OPF problem, making the solving of this non-convex problem traceable. (6) extends the online gradient methods to distributed implementations by leveraging the linear model and feedbacks from the real power network. It relies on the centralized coordinator to receive and distribute the information for local buses. All the aboves solve the problem from the perspectives of optimization and can be classified as traditional methods.

Recently, the success of deep learning extends to power system. The author in (13) utilizes deep neural network to solve the DC OPF problem. And the worse-case guarantees are analysed in (17). (20) and (21) further leverage reinforcement learning approaches to solve the OPF problem. The rewards are designed as the consequence of satisfaction of the physical laws and the optimality of the feasible points, further improving the solving of AC OPF problem.

2 Problem Definition

In this section, we formulate the OPF problem and then propose a deep reinforcement learning solution strategy to solve the OPF problem.

2.1 Problem Formulation

Consider a single-phase equivalent power distribution network as a directed *tree* graph $\mathcal{T} := \{\mathcal{N}^+, \mathcal{E}\}$, where $\mathcal{N}^+$ index all nodes in the network. The set $\mathcal{E}$ collects the ordered pairs of nodes representing the lines in the network. We define the directions of lines pointing away from the root, e.g., in line (i, j), node i closer to the root than node j. Let p_i and q_i denote the net active and reactive power injections (i.e., power supply (p_{g_i}, q_{g_i}) minus consumption (p_{d_i}, q_{d_i})) and v_i denotes the squared voltage magnitude at each node $i \in \mathcal{N}^+$. Let ℓ_{ij} denotes the squared current magnitude, $z_{ij} = r_{ij} + \boldsymbol{i}x_{ij}$ denote the series impedance, and P_{ij} and Q_{ij} denote the sending-end active and reactive power on each line $(i, j) \in \mathcal{E}$. We adopt the distribution power flow model in (2):

$$P_{ij} = -p_j + \sum_{k:(j,k)\in\mathcal{E}} P_{jk} + r_{ij}\ell_{ij},\ \forall j \in \mathcal{N}^+ \tag{1a}$$

$$Q_{ij} = -q_j + \sum_{k:(j,k)\in\mathcal{E}} Q_{jk} + x_{ij}\ell_{ij},\ \forall j \in \mathcal{N}^+ \tag{1b}$$

$$v_j = v_i - 2\left(r_{ij}P_{ij} + x_{ij}Q_{ij}\right) + \left|z_{ij}\right|^2 \ell_{ij}, \forall (i,j) \in \mathcal{E} \tag{1c}$$

$$\ell_{ij}v_i = P_{ij}^2 + Q_{ij}^2,\ \forall (i,j) \in \mathcal{E} \tag{1d}$$

The equations (1) models the physical laws of the power system, where (1a-1b) are the power balance equations and (1c) is the Ohm's law. The quadratic terms in (1d) are introduced by angle relaxation of BFM (7). Optimal power flow seeks to minimize some certain costs subjected to equations (1) and under the defined power injection and voltage bounds. it can be assuming that the power injection as:

$$\mathcal{Y}_i = \left\{(p_i, q_i) \mid \underline{p}_i \leqslant p_i \leqslant \bar{p}_i, \underline{q}_i \leqslant q_i \leqslant \bar{q}_i\right\},\ \forall i \in \mathcal{N} \tag{2}$$

Then the optimal power flow problem can be formulated as:

$$\min_{\boldsymbol{p},\boldsymbol{q}} \sum_{i\in\mathcal{N}} f_i\left(p_i, q_i\right) \tag{3a}$$

$$\text{s.t. } (1a), (1b), (1c), (1d) \tag{3b}$$

$$\underline{v_i} \leqslant v_i \leqslant \overline{v_i},\ \forall i \in \mathcal{N} \tag{3c}$$

$$(p_i, q_i) \in \mathcal{Y}_i,\ \forall i \in \mathcal{N} \tag{3d}$$

where $f_i\left(p_i, q_i\right)$ is a cost function associated with bus i and is assumed to be convex in operating of power system (18) (e.g. to maximize the user's utility $f_i = (p_i - p_i^{ref})^2 + (q_i - q_i^{ref})^2$, where p_i^{ref} and q_i^{ref} are the nominal real and reactive power of user i).

The OPF problem (3) is hard to solved because of the non-convexity of (1d). And even the traditional OPF algorithm may be failed if the convex relaxation of (1d) is not exact (15). Hence in next part, we propose a deep reinforcement learning method to seek for an optimal operating point for the power system, namely, solve the OPF problem.

2.2 Power Environment for Reinforcement Learning

In this part, we demonstrate the design of our power environment. We suppose a deep reinforcement learning (DRL) agent governs the power injections into the power system. This agent takes power network's state (e.g. power injection, voltage) as input, which is in fact a vector, then outputs an action for every active bus's system operator to adjust its power injections. The real power system modeled by power equations (1a-1c) is simulated with OpenDSS software platform (`https://www.epri.com/pages/sa/opendss`). With above assumptions, we create our power environment in OpenAI Gym (5). The detail designs are elaborated below.

We modified the state and action spaces designed in (20), the state spaces to be inputted into the agent are defined as:

$$state = [p_i, q_i, v_i, \forall i \in \mathcal{N}] \tag{4}$$

The action spaces are designed as the incremental adjustment of power injections, or namely, the step-wise adjustment of generator outputs:

$$action = [\Delta p_1, ..., \Delta p_N, \Delta q_1, ..., \Delta q_N] \tag{5}$$

Hence, the next step of power injections are computed using equation (6). The next state in (4), such as all bus voltages $(v_i, \forall i \in \mathcal{N})$, is obtained through measurements from the real power network simulators. This is a feedback-based control manner and is widely adopted to compensate for modeling errors (4; 3; 19).

$$p_i = p_i + \Delta p_i, q_i = q_i + \Delta q_i,\ \forall i \in \mathcal{N} \tag{6}$$

Under the above design of state and action spaces, there may be several cases when the DRL agent takes an action given a state. 1) The action taken by the agent may not satisfy power injection region (3d), which leads to an unfeasible state that the generators cannot provide such active and reactive power to the network. We set done for this case to terminate an episode. 2) Although the action produces a result satisfying (3d), the state violates the voltage bounds (3c). We also set done for this case to terminate an episode; 3) The actions taken by the agent lead to a feasible result but might not be optimal. We design the reward as (7) for the feasible solutions and the reward will increase as the objective function f_i reduces.

$$reward = \lambda - \sum_{i\in\mathcal{N}} f_i(p_i, q_i), \quad \text{for feasible } (p, q) \tag{7}$$

where λ is a system hyper-parameter, we choose it as the estimated lower bound of the OPF problem (3). Since the optimal solution is usually located in the boundary of constraints (3c,3d), the done cases 1) and 2) are in fact reaching the optimal solution then we can terminate this episode, otherwise it will result in a large negative reward under the design of our reward function. Besides, we also set done if the episode reward (7) is positive, because the positive reward means reaching the estimated lower bound of OPF problem. The interaction of our agent with our designed power environment is illustrated in Fig.2. There is a maximum episode length to result in terminating an episode if the agent cannot reach the done conditions within an episode.

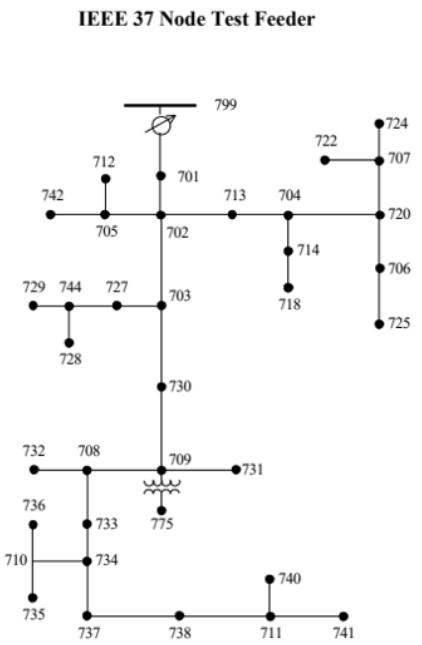

Figure 1: IEEE-37 node test feeder from California. Node 799 is the source bus with fixed voltages. Some buses are equipped with controllable devices such as generators.

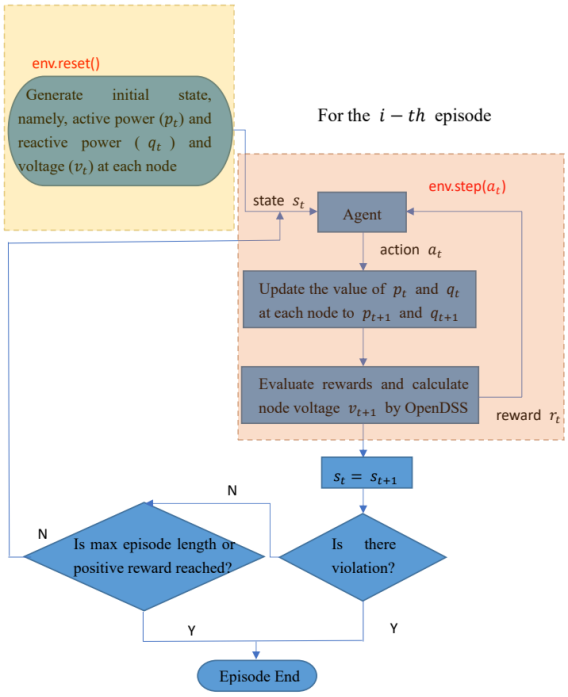

Figure 2: The interaction of our power environment. The env.reset() initializes state before an episode, the env.step() takes the actions and compute reward, then verify the done conditions in blue boxes. Only when no done condition meet, then continue the next time step.

2.3 TD3 Algorithm

We use Twin Delayed Deep Deterministic Policy Gradient (TD3) (16) as the algorithm to update our policy. TD3 is further improved on the basis of DDPG, and the basic framework is the same as DDPG, which can mitigate the overestimation of Q-value. There are two Q-functions, Q_{ϕ_1} and Q_{ϕ_2}, TD3 learns simultaneously by mean square Bellman error minimization, which results in a deterministic policy μ_θ to directly output the deterministic action that maximizes $Q_\phi(s,a)$.

The target actions are obtained:

$$a'(s') = clip(\mu_{\theta_{targ}}(s' + clip(\epsilon, -c, c), a_{low}, a_{high}), \epsilon \sim \mathcal{N}(0, \sigma) \tag{8}$$

where $\mu_{\theta_{targ}}$ is the target policy; ϵ is the noise.

Since the parameters of both Q-functions are initialized randomly, so the outputs of these two functions are different, from which we choose smaller one as a target value:

$$y(r, s', d) = r + \gamma(1-d) \min_{i=1,2} Q_{\phi_{i,targ}}(s', a'(s')) \tag{9}$$

and then both functions will be trained by regressing to the above target value $y(r, s', d)$:

$$L(\phi_1, D) = \underset{s,a,r,s',d\sim_D}{E}[(Q_{\phi_1}(s,a) - y(r,s',d))^2] \tag{10}$$

$$L(\phi_2, D) = \underset{s,a,r,s',d\sim_D}{E}[(Q_{\phi_2}(s,a) - y(r,s',d))^2] \tag{11}$$

Assuming that the minimum value selected by the Q-target network is Q_{ϕ_1}, the policy is updated by maximizing Q_{ϕ_1}:

$$\max_\theta \underset{s\sim_D}{E}[Q_\phi(s, \mu_\theta(s)))] \tag{12}$$

On training, we solve the above problem and can obtain the policy μ_θ, which maximizes the expectation of rewards. On testing, the optimal action is given by the trained policy μ_θ, which is in fact the output of Actor network.

3 Experiments

3.1 Experiment Setup

We collect a real IEEE-37 bus test feeder in California, with a 4.8 kV operating voltage from `https://cmte.ieee.org/pes-testfeeders/resources/`. Its topology is showed in Fig.1. The source bus is located in node 799 with assumption that the voltage of source bus is fixed. Some of nodes in IEEE-37 bus test feeder are active nodes equipped controllable generators to provide both active and reactive power injections into the network, while some aren't. We first transfer it to its single phase equivalent and build this real power network in the simulator OpenDSS according to the default configurations in IEEE-37 bus test feeder, as what we do in (10).

We first run TD3 that will be compared with other three SOTA algorithms consisting of SAC, PPO, DDPG in our power environment and use the pre-trained agent to find an optimal setpoint for the operating of power system. Then we compare the TD3 agent with a traditional gradient-based control scheme developed in (10). All of these experiments are showed below.

3.2 Results

We have run four SOTA algorithms including TD3, SAC, PPO, DDPG. For fair comparison, we train these four agents with same hyper-parameters such as 5×10^5 time steps as well as the same learning rate of 1e-3. Then, we plot the smoothed episode rewards with respect to episode steps in Fig.3. As we can see, the TD3 converges quickly in about 1000 episodes. It reaches the positive rewards in some episode steps, which corresponds to the estimated lower bound of the OPF problem.

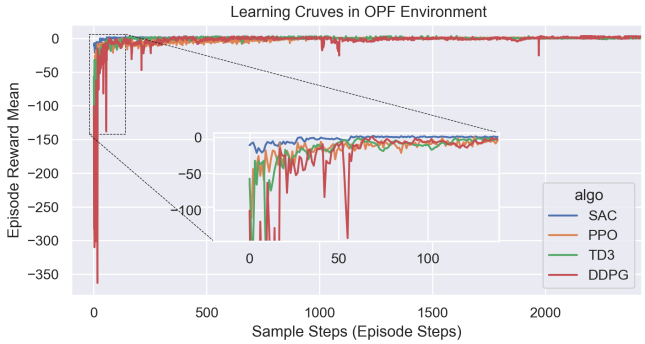

Figure 3: The learning curve of TD3, PPO, SAC and DDPG

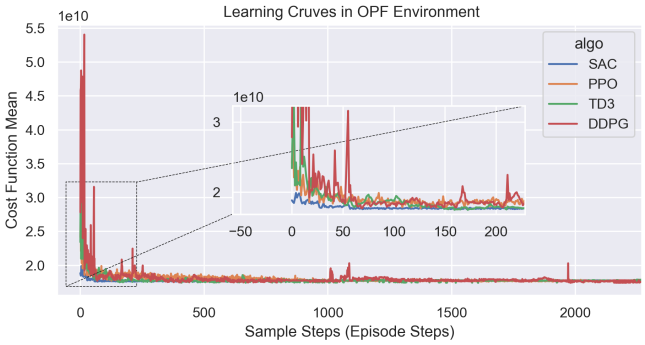

Figure 4: The cost function during TD3, PPO, SAC and DDPG training.

We further plot the smoothed cost function during training in Fig.4. The setting of estimated lower bound in our experiment is 1.78×10^{10}, and in fact it can be further tuned smaller to the cases that the positive reward can't reach. But we find the cost function doesn't reduce any more and observe severe oscillations in these cases.

We evaluate these four agents and have tested 100 episodes for each agent. The result can be found in Table 1 where we can find TD3 has the best performance with the largest reward and smallest cost.

Table 1: Testing Result

Agent	Mean Reward	Cost	Episodes
TD3	**1.5947**	**1.7640**	100
PPO	1.2712	1.7672	100
DDPG	1.0424	1.7695	100
SAC	0.2434	1.7775	100

We also plot the optimal setpoint produced by RL control and traditional gradient-based control schemes. Fig.5 depicts the voltages distributions. The green dots are the voltages under the default configurations, without any control. There are severe voltage violations under the default setting. Orange dots are the voltages under traditional control. Although it lifts up all node voltages within the voltage bounds (0.95 p.u. to 1.05 p.u.), most of node voltages still located in the boundary, and has the potential to go into the under-voltage cases. Red dots are the results produced by our RL control scheme. It successfully lifts up all node voltages above the voltage lower bound (0.95 p.u.). Most of nodes are located in a safe operating region.

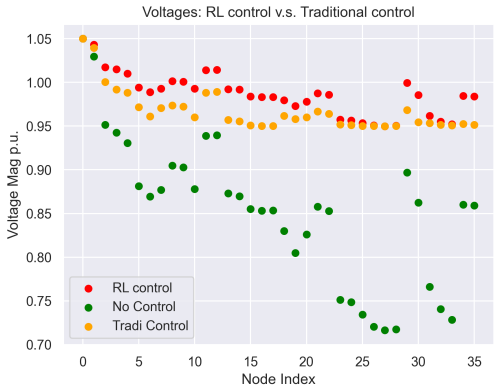

Figure 5: The voltages in IEEE-37 node network. No Control means nominal power injections; Tradi Control means the result by the method in (10); RL Control means our proposed method.

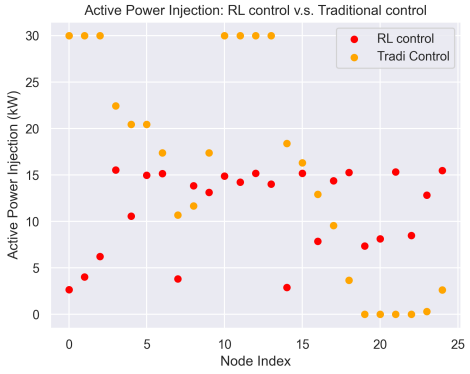

Figure 6: Active power injection

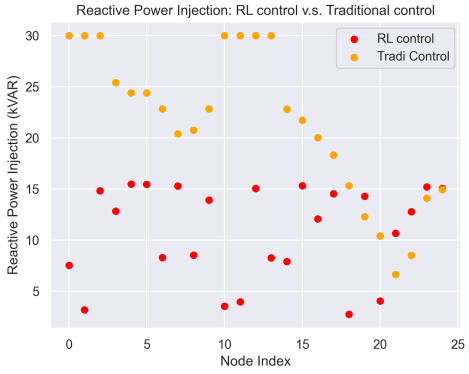

Figure 7: Reactive power injection

We further plot the active and reactive power injections in the nodes with controllable devices in Fig.6 and Fig.7. The available power injection defined in (2) is 0-30 kW (kVAR). Our RL control scheme produce a smoothed setpoint for all users, while the traditional control scheme returns an extreme setpoint with 0 kW (kVAR) and 30 kW (kVAR) for some users.

4 Conclusion

In this paper, we firstly create a power environment based on OpenAI Gym, and then apply the SOTA RL algorithms consisting of TD3, SAC, PPO, DDPG to train our agent. By comparison, TD3 agent can achieve relatively good performance with the largest reward and smallest cost. Then, compared with the traditional gradient-based control method, the RL control scheme can achieve a safer setpoint for the operating of power system.

References

[1] X. Bai and H. Wei. A semidefinite programming method with graph partitioning technique for optimal power flow problems. *International Journal of Electrical Power & Energy Systems*, 33(7):1309–1314, 2011.

[2] M. Baran and F. F. Wu. Optimal sizing of capacitors placed on a radial distribution system. *IEEE Transactions on Power Delivery*, 4(1):735–743, 1989.

[3] A. Bernstein and E. Dall'Anese. Real-time feedback-based optimization of distribution grids: A unified approach. *IEEE Transactions on Control of Network Systems*, 6(3):1197–1209, 2019.

[4] S. Bolognani, R. Carli, G. Cavraro, and S. Zampieri. Distributed reactive power feedback control for voltage regulation and loss minimization. *IEEE Transactions on Automatic Control*, 60(4):966–981, 2014.

[5] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[6] E. Dall'Anese and A. Simonetto. Optimal power flow pursuit. *IEEE Transactions on Smart Grid*, 9(2):942–952, 2018.

[7] M. Farivar and S. H. Low. Branch flow model: Relaxations and convexification—part i. *IEEE Transactions on Power Systems*, 28(3):2554–2564, 2013.

[8] L. Gan and S. H. Low. Convex relaxations and linear approximation for optimal power flow in multiphase radial networks. In *2014 Power Systems Computation Conference*, pages 1–9. IEEE, 2014.

[9] L. Gan and S. H. Low. An online gradient algorithm for optimal power flow on radial networks. *IEEE Journal on Selected Areas in Communications*, 34(3):625–638, 2016.

[10] H. Liang, X. Zhou, and C. Zhao. Hierarchical optimal power flow with improved gradient evaluation. In *In Preprint*, 2021.

[11] S. H. Low. Convex relaxation of optimal power flow—part i: Formulations and equivalence. *IEEE Transactions on Control of Network Systems*, 1(1):15–27, 2014.

[12] S. H. Low. Convex relaxation of optimal power flow—part ii: Exactness. *IEEE Transactions on Control of Network Systems*, 1(2):177–189, 2014.

[13] X. Pan, T. Zhao, and M. Chen. Deepopf: Deep neural network for dc optimal power flow. In *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–6. IEEE, 2019.

[14] Q. Peng and S. H. Low. Distributed algorithm for optimal power flow on an unbalanced radial network. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 6915–6920. IEEE, 2015.

[15] Q. Peng and S. H. Low. Distributed optimal power flow algorithm for radial networks, i: Balanced single phase case. *IEEE Transactions on Smart Grid*, 9(1):111–121, 2016.

[16] L. G. H. N. Silver, D. Deterministic policy gradient algorithms. *arXiv preprint*, pages pp. 387–395, 2014.

[17] A. Venzke, G. Qu, S. Low, and S. Chatzivasileiadis. Learning optimal power flow: Worst-case guarantees for neural networks. In *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–7. IEEE, 2020.

[18] C. Zhao, E. Dall-Anese, and S. H. Low. Optimal power flow in multiphase radial networks with delta connections. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2017.

[19] X. Zhou, Z. Liu, C. Zhao, and L. Chen. Accelerated voltage regulation in multi-phase distribution networks based on hierarchical distributed algorithm. *IEEE Transactions on Power Systems*, 35(3):2047–2058, 2019.

[20] Y. Zhou, W.-J. Lee, R. Diao, and D. Shi. Deep reinforcement learning based real-time ac optimal power flow considering uncertainties. *Journal of Modern Power Systems and Clean Energy*, 2021.

[21] Y. Zhou, B. Zhang, C. Xu, T. Lan, R. Diao, D. Shi, Z. Wang, and W.-J. Lee. A data-driven method for fast ac optimal power flow solutions via deep reinforcement learning. *Journal of Modern Power Systems and Clean Energy*, 8(6):1128–1139, 2020.