Fully Decentralized Reinforcement Learning-based Control of Photovoltaics in Distribution Grids for Joint Provision of Real and Reactive Power

Rayan El Helou, Student Member, IEEE, Dileep Kalathil, Member, IEEE, and Le Xie, Senior Member, IEEE

Abstract—In this paper, we introduce a new framework to address the problem of voltage regulation in unbalanced distribution grids with deep photovoltaic penetration. Both real and reactive power setpoints are explicitly controlled at each solar panel smart inverter, and the objective is to simultaneously minimize system-wide voltage deviation and maximize solar power output. We formulate the problem as a Markov decision process (MDP) with continuous action spaces and use proximal policy optimization (PPO), a reinforcement learning (RL)-based approach, to solve it, without the need for any forecast or explicit knowledge of network topology or line parameters. By representing the system in a quasi-steady state manner, and by carefully formulating the MDP, we reduce the complexity of the problem and allow for fully decentralized (communication-free) policies, all of which make the trained policies much more practical and interpretable. Numerical simulations on a 240-node unbalanced distribution grid, based off of a real network in Midwest U.S., are used to validate the proposed framework and RL approach.

Index Terms—Unbalanced distribution grids, photovoltaic (PV) inverters, voltage regulation, reinforcement learning (RL).

I. INTRODUCTION

PHOTOVOLTAIC (PV) smart inverter technology introduced in recent years enable solar panels to act as distributed energy resources (DERs) that can provide bidirectional reactive power support to electric power grid operations [11][12]. This support can be used to regulate local and system-wide voltages in distributed grids, and IEEE Standard 1547-2018 provides requirements on the use of such support. Voltage regulation is critical for network safety, both at the transmission and distribution levels.

In the distribution grid, voltage regulation is usually controlled through either discrete switching (e.g. tap transformers, capacitor banks) or continuous set points (e.g. PV inverters). Two paradigms of control and information structure are proposed in the literature to address the voltage regulation problem. On one hand, there are solutions which assume complete or partial knowledge of system parameters and topology (e.g. [5][11]), and on the other, there are those which are purely data-driven and rely on little to no knowledge of a system model, (e.g. this paper and [12][18]). In either case, voltage regulation can be posed as a Markov decision process (MDP). However, in the first case, control schemes are adopted based on the assumed system models, while in the second case, reinforcement learning (RL) approaches are used to bypass the need to model the system. In [12], for example, Batch RL is adopted to solve optimal setting of voltage regulation transformers, where a virtual transitions generator is used to allow the RL agent to collect close-to-real samples, for learning, without jeopardizing real-time operation. In [17], Deep RL is used to optimize reactive power support over two timescales: one for discrete capacitor configuration and the other for continuous inverter setpoints.

Such methods are inherently limited by physical constraints on reactive power support which are conventionally assumed to be uncontrollable. Here, we discuss the flexibility of such constraints, and the value in relaxing them. Conventional practices of maximum power-point tracking (MPPT) have been state of the art, wherein each PV inverter is designed to extract the maximum real/active power from the solar panel. However, with growing number of PV panels in the distribution grid, it becomes important to fully investigate the benefits and costs of always absorbing the maximum real power from the sun into the grid in real-time. By absorbing less real power, for instance, there is more room for reactive power support. We illustrate a set of scenarios where instead of injecting all of the solar power into the network, it might rather be better to save or store the power and to inject it at a later time. Even in the absence of a storage system, under deep enough photovoltaic penetration, we find that it might surprisingly be better to draw only parts of the available power, in order to avoid over voltage, if there’s insufficient reactive power resources available.

The key contributions of this paper are summarized as follows: (1) to propose a decentralized control policy architecture that can be shown to train as well as, or better than, a centralized policy architecture in a continuous action space setting using reinforcement learning (RL), and (2) to propose a parametrized reward function which enables the user to dictate the balance between maximization of real power injection from solar panels and minimization of voltage deviations from nominal. The RL agent observes voltages in the network and incrementally changes real and reactive power setpoints, similar to a integral droop controller (e.g. [19]), but does not rely on any knowledge of network topology or line parameters, and is fully decentralized, requiring minimal communication infrastructures for practical implementation.

The remainder of the paper is organized as follows. In Section II the voltage regulation objective with joint real and reactive power compensation is formulated. In Section III we
provide a general review of Markov Decision Processes, and one specific to our problem in Section IV, with modifications to simplify the task. In Section V, centralized and decentralized policy architectures are proposed, and they are evaluated in Section VI with numerical simulations.

II. PRELIMINARIES

A. Three-phase Unbalanced Distribution Grid Model

Consider a three-phase balanced distribution grid with a single substation bus that acts as the sole point of connection to a bulk power grid. Let \( \mathcal{N} := \{0, 1, \ldots, N\} \) uniquely identify the set of buses in the network (one integer per bus), and zero is reserved for the substation bus. Similarly, let \( \mathcal{L} \subset \mathcal{N} \times \mathcal{N} \) uniquely identify the set of lines in the network, such that buses \( i \) and \( j \) are connected if and only if \((i, j) \in \mathcal{L} \). For convention, \((i, i) \in \mathcal{L} \ \forall i \in \mathcal{N} \).

The set of algebraic power flow equations that govern this three-phase balanced (single-phase equivalent) network are:

\[
P_i - iQ_i = \bar{V}_i^* \sum_{j \in \mathcal{N}} y_{ij} \bar{V}_j \quad \forall i \in \mathcal{N} \tag{1}
\]

where \( P_i \) and \( Q_i \) are the net injection of real and reactive power, respectively, at bus \( i \) (from bus to grid). \( \bar{V}_i \) is the complex phasor voltage at the same bus, and \( y_{ij} \), also complex, is the element in the \( i^{th} \) row and \( j^{th} \) column of the network’s admittance matrix. \( i := \sqrt{-1} \).

To model a distribution grid which is not three-phase balanced, or unbalanced for short, we may simply replace bus indices with phase indices in Eq. (1), and \( \mathcal{N} \) with the set of all phases. This allows us to easily generalize over two-phase and single-phase buses.

B. Voltage Regulation through both real and Reactive Power Compensation

To regulate voltages across the distribution grid, we propose a framework for joint real and reactive power control of PV inverter setpoints. The control objective is to track desired voltage levels while not wasting solar power in the process.

1) Voltage Measurement: Throughout this paper, \( V_i \) refers to the positive sequence voltage magnitude at bus \( i \). It is this voltage which we seek to regulate at each bus, with a desired setpoint of 1.0 p.u.. At the substation bus, \( V_0 \) is fixed at 1.0 p.u. as it is modeled as an ideal voltage source.

2) Active and Reactive Power Setpoints: Let \( P_i^c \) and \( Q_i^c \) be the total real and reactive power, respectively, injected by the PV inverters at bus \( i \). These are the decision variables (superscript \( c \) for control), and we let this injected power be evenly distributed across all phases per bus.

Each PV inverter has an apparent power capacity, \( S_i \), which limits \( P_i^c \) and \( Q_i^c \) as follows:

\[
(P_i^c)^2 + (Q_i^c)^2 \leq S_i^2
\]

\[
P_i^c \leq P_{i,max} \leq 0.9S_i \tag{2b}
\]

where \( P_{i,max} \) is the maximum amount of real power that can be drawn from the solar panel at a given moment in time. It changes during the day due to exogenous environmental factors, hence the superscript. The upper bound on this quantity is 0.9\( S_i \) since each inverter in the network is assumed to obey standard IEEE 1547-2018.

Strictly speaking, if \( P_i^c(t) \) is the actual real power injected by the inverter at time \( t \), and \( P_{i,ref}(t) \) is the setpoint, then those two cannot be equal at the same time. There is a small time delay (~10 ms, or less than one 60 Hz cycle) between when the setpoint is assigned and when the actual quantity tracks it. We let both the discrete time step and the tracking time be 10 ms. This allows us to treat the system as a quasi-steady state system, as illustrated in Fig. 1.

![Fig. 1: Illustration of quasi-steady state behavior: 1) It takes \( 9 \) time steps for the setpoint to be reflected in the actual injection, and 2) \( V(t+1) \) is algebraically tied to \( P_{i,ref}(t) \), not to \( P_{i,ref}(t+1) \).](image)

3) Voltage Control Objective: Consider that only some of the buses, \( \mathcal{C} \subset \mathcal{N} \), in the network are equipped with controllable smart inverters, and that voltage deviation is considered only at those buses. The voltage control objective is formulated as follows:

\[
\max_{P^c, Q^c} \mathbb{E} \left\{ \sum_{i=0}^{T} \sum_{c \in \mathcal{C}} R_{V_i(t)} + \mu_i R_{P_i^c(t)} \right\} \tag{3a}
\]

s.t. \( P_i(t) = P_i^c(t) - P_i^{l}(t) \)
\( Q_i(t) = Q_i^c(t) - Q_i^{l}(t) \)
Eq. (1), i.e. power flow, \( \forall t \)
Eq. (2), i.e. inverter constraints, \( \forall t \)
\( R_{V_i(t)} := \frac{1}{0.05} \min \{ \delta - |V_i(t)|, 0 \} \) \tag{3f}
\( R_{P_i^c(t)} := \frac{P_i^c(t)}{0.9S_i} \) \tag{3g}

where superscript \( l \) denotes load consumption. Unlike with \( P^c \) and \( Q^c \) (controllable inverter), \( P^l \) and \( Q^l \) (uncontrollable load) are generally not evenly distributed across all individual phases.

Terms \( R_{V_i} \) and \( R_{P_i^c} \) (\( R \) for reward) in the Eq. (3f,3g) of the objective are justified as follows. Voltage deviation (from nominal 1.0 p.u.) at each bus is considered acceptable if it is kept within some user-defined \( \delta \). Deviations greater than this are assigned negative rewards, as depicted in Fig. 2 to signify an undesirable voltage profile. As for real power, we seek to extract as much of it as possible from the solar panel, physically bounded by 0.9\( S_i \) (see Eq. (2b)), so we assign positive reward \( (\mu_i \geq 0 \ \forall i) \) to more power drawn from the solar panels. The term \( \mu \) acts as a balancing term here,
between voltage deviation minimization and solar production maximization, considering the fact that over-injection of power leads to over-voltage.

The following assumptions are made about variables which are not explicitly controlled:

- Neither network topology nor line parameters are used by the controller at any time during training or execution.
- No load or solar forecasting is made available to the controller, neither upon training nor during execution.
- Net load at a control bus is measured by the controller before supplying a setpoint to the solar panel inverter.

III. MARKOV DECISION PROCESSES AND PROXIMAL POLICY OPTIMIZATION

In this section, we review MDP’s and the general goal of RL algorithms. We also review a specific RL algorithm, called PPO, which alleviates the curse of dimensionality associated with continuous actions.

A. Markov Decision Processes and Reinforcement Learning

MDP’s can be used to model discrete time stochastic control problems. We adopt a simplified definition of MDP’s based on finite-time processes (with episode length of T), with no discount factor, and rewards dependant only on states (not on actions). An MDP can be defined as a four-tuple \((S, A, P, R)\), where \(S\) is the state space and \(A\) is the action space. \(P(s'|s,a)\) is the probability of transitioning from state \(s\) to \(s'\) upon taking action \(a\), and \(R(s)\) is the reward collected at this transition. Note: \(P\) and \(V\) in this section denote probability and value, not real power and voltage.

Reward function, \(R\), is usually designed in such a way that selecting control policies which maximize expected cumulative rewards yields desired system performance. In practical control applications, states represent physical quantities, and we seek to steer the system towards better states. A value function, \(V : S \rightarrow \mathbb{R}\), is used to quantify how well it is to be at a given state, and is defined as the expected sum of rewards as follows:

\[
V(s) := \mathbb{E}^s \left[ \sum_{t=0}^{T-1} R(s_t) \mid s_0 = s \right]
\]

where \(E^s\) refers to expectation over states visited assuming the best control policy is adopted. Thus, the best control policy, \(\pi^* : S \rightarrow A\) must satisfy the following equation:

\[
\pi^*(s) = \arg\max_{a \in A} \left[ R(s) + \sum_{s' \in S} P(s'|s,a)V(s') \right]
\]

also known as the Bellman equation.

Reinforcement learning (RL) is a method for learning \(\pi^*\), the optimal control policy, without the need to know transition kernel \(P\). This can be challenging in a context where \(S\) and \(A\) are not discrete or are very large if discretized to suit our needs. In the following section, we review a relatively recent development in the field of RL, called Proximal Policy Optimization, which addresses the continuity in state and action spaces.

B. Proximal Policy Optimization

PPO is an RL algorithm developed by a team at OpenAI [20] which has proven successful in a broad range of tasks, such as robotics control and sophisticated video games like Dota 2. We later demonstrate the use of this algorithm in our voltage regulation problem. We refer the reader to [20] for more details on PPO, but here is a summary of this algorithm.

PPO is the successor of TRPO [21], and they both use an advantage estimate \(\hat{A}_t\), calculated at the end of each episode, to quantify how well a policy performs over each state in that episode, relative to some baseline performance. In its simplest form, it is the difference between actual returns and expected returns. That is,

\[
\hat{A}_t = G_t - \hat{V}(s_t)
\]

where \(\hat{V}\) is referred to as the baseline value function and \(G_t\), known as the return or rewards-to-go is the sum of actual rewards collected from \(t\) onwards. A neural network is usually used to define baseline \(\hat{V}\) and its parameters are updated every batch of episodes by comparing value estimates with actual returns.

At any point in training using either PPO or TRPO, there’s a current policy \(\pi_\theta\) and an old one \(\pi_{\theta_{old}}\), from last training iteration, where \(\theta\) denotes policy parameters. If \(\dot{A}_t > 0\), then \(\pi_\theta\) is considered to have exceeded expectations, making it more desirable to move away from \(\pi_{\theta_{old}}\) towards \(\pi_\theta\). This is captured by the following surrogate function, used in TRPO, to be maximized at every training iteration:

\[
L(\theta) = \mathbb{E}_t \left[ \frac{\pi_\theta(s_t,a_t)\dot{A}_t}{\pi_{\theta_{old}}(s_t,a_t)\dot{A}_t} \right]
\]

where \(\pi_\theta(s, a)\) is shorthand for probability of taking action \(a\) given state \(s\) under policy \(\pi_\theta\).

Using auto-differentiation libraries like PyTorch [22], we can maximize this quantity over parameters \(\theta\), but one more modification is needed to form the PPO surrogate function.
Lack of constraints on the surrogate in Eq. (7) leads to unstable updates in $\theta$, so clipping (or saturation) is introduced [20]:

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[ \min\{\ast, \hat{A}_t, \text{clip}(\ast, 1 - \varepsilon, 1 + \varepsilon, \hat{A}_t)\} \right]$$

where $\ast := \frac{\pi_0(s_t, a_t)}{\pi_{\theta,a}(s_t, a_t)}$.

Policy parameters $\theta$ are updated to maximize $L^{CLIP}()$, and $\varepsilon$ is a small positive number ($\approx 0.2$).

IV. VOLTAGE REGULATION FORMALIZED AS AN MDP

An objective was formulated in Section II-B3 where the decision variables are $\Delta P^C$ and $\Delta Q^C$, the setpoints for the real and reactive power injections at the PV smart inverters, and $C$ is the subset of buses in the grid at which those setpoints can be controlled. In this section, we present a reformulation of the same objective but as an MDP, defined by the tuple $(S, A, P, R)$, like in Section III-A.

A. Incremental Control

Given voltage measurements at every bus in $C$, what policy do we adopt to determine $\Delta P^C$ and $\Delta Q^C$? Here are two approaches: 1) Directly determine optimal real and reactive power setpoints, i.e. $(\Delta P^C, \Delta Q^C)$, by algebraically tying to voltage, or 2) change setpoints incrementally, i.e. $(\Delta P^C, \Delta Q^C)$, similar to an integral controller.

The first approach requires the design and memorization of a highly non-linear function that is likely dependant on system operating conditions. Due to the lack of tracking in this approach, forecasting would be required to respond to different operating conditions. The second approach, on the other hand, enables tracking a desired state, within resource limits of course, and it remains to demonstrate the convergence and stability of this approach in time domain. Reference [11] guides the design of an integral controller for a distribution grid with known line parameters, with guarantees on stability assuming reactive power is within limits. In that paper, it is shown that a simple communication-free (fully decentralized) linear control can achieve this; however, only reactive power was controlled there, not real power.

In this paper, we further explore this second approach by using RL to drop knowledge of system parameters, while maintaining decentralized control.

B. Problem Formulation as an MDP

The voltage regulation problem is formulated as an MDP by defining the tuple $(S, A, P, R)$ as follows.

1) State space: Let $S \subseteq \mathbb{R}^{2n}$ be the set of real power injections and voltage measurement at all controllable buses ($n = |C|$). For convenience, each state $s \in S$ is defined as an affine transformation of those measurements:

$$s := (s^P_1, s^V_1, \ldots, s^P_n, s^V_n)$$

where $s^P_i \leftarrow \frac{P^c_i}{0.9S^i} - 1$

$s^V_i \leftarrow \frac{1 - V_i}{0.05}$

That is, during maximum solar production, the first term in the local state is zero, and at nominal voltage, the second term is zero. Thus, ideal scenarios correspond to zero state, and critical scenarios correspond to magnitudes of order one. Such scaling helps initialize and train the agent’s policy.

2) Action space: $A$ is the set of possible scaled increments in real and reactive power setpoints, bounded by $\pm 1$:

$$A = \{(a^P_1, a^Q_1, \ldots, a^P_n, a^Q_n) \in [-1, 1]^2n\}$$

where $\Delta P^C_i \leftarrow a^P_i \cdot \Delta P^C_{max}$

$\Delta Q^C_i \leftarrow a^Q_i \cdot \Delta Q^C_{max}$

Constants $\Delta P^C_{max}$ and $\Delta Q^C_{max}$ explicitly limit the size of actual (as opposed to scaled) increments $(\Delta P^C, \Delta Q^C)$, since elements in $A$ are bounded by $\pm 1$.

3) Transition Model: In our context, we assume that next states are obtained by interaction either with a real-world distribution grid or with a simulator, such as OpenDSS [23]. In either case next states are determined directly by states and actions based on the definitions of $S$ and $A$, with one caveat: load $(P^l, Q^l)$ needs to be known. This is addressed in the following subsection (IV-C). Action is mapped to state using OpenDSS as follows:

$$V(t + 1) = \text{OpenDSS}(P^C(t), Q^C(t))$$

4) Reward function: System-wide reward at every time step is obtained as follows:

$$r_t = \frac{1}{|C|} \sum_{i \in C} R_{V_i}(t) + \mu_i R_{P_i}(t)$$

where $R_{V_i}(t)$ and $R_{P_i}(t)$ are defined in Eq. (38-39).

This concludes the definition of tuple $(S, A, P, R)$. The state and action spaces have been defined in such a way that each element ranges from $-1$ to $1$, with an exception where the voltage-related state my exceed $\pm 1$ if the p.u. voltage exceeds $1 \pm 0.05$ under abnormal conditions. This is a suitable choice for training an RL policy as it allows for initialization and adjustment of policy parameters $\theta_{\pi}$ in a standardized way. It allows us to take advantage of existing state of the art algorithms which recommend that state and action spaces be a box inside $\pm 1$ along all dimensions, as we have here.

C. RL agent nested in integral controller

Based on the definition of action space, the RL agent seeks to learn the magnitude and direction in which to incrementally change the setpoints, for every starting state. This raises the question: what information does the agent need to guide this action? The state defined in Eq. (9) has the following advantage: If both the voltage term and the power term are zero (i.e. maximum power drawn and nominal voltage), then the scenario is ideal and no extra injection is needed. If the load changes in the system, though, a simple amendment to the RL controller is needed:

$$\Delta P^C \leftarrow \Delta P^C_{max} \cdot a^P + \Delta P^l$$

$$\Delta Q^C \leftarrow \Delta Q^C_{max} \cdot a^Q + \Delta Q^l$$

(13)
where \( a_P \) and \( a_Q \) (both in \([-1, +1]\)) are determined by the RL agent’s zero-centered policy \( \pi \), and \((\Delta P^i, \Delta Q^i)\) is the observed change in load at the controllable buses.

The strategy adopted in Eq. (13) is termed an integral controller since setpoint \((P^*, Q^*)\) behaves as a discrete-time integrator of changes in operating condition. Moreover, this controller tracks the state to zero in steady state, within resource limits, since all terms in Eq. (13) go to zero if \( s = 0 \). Under scarcity of resources, one or more of the state terms in Eq. (9) will be non-zero, which calls for a balance between maximum power point tracking and voltage regulation.

Note that state-tracking incremental setpoint changes are bounded by \( \Delta P_{\text{max}} \) and \( \Delta Q_{\text{max}} \) to limit fluctuations. These values are chosen heuristically as 0.09\( S_i \) and 0.25\( S_i \) respectively, since those are one tenth of the maximum possible jumps in setpoints \( P^* \) and \( Q^* \).

\[ \Delta \theta_i \]

V. CONTROL POLICY ARCHITECTURE AND OPTIMIZATION

In this section, we present the design and architecture of policy \( \pi \) and review Proximal Policy Optimization (PPO) [20], an RL actor-critic approach which we use in this paper to handle continuous action spaces. Fig. 3 illustrates the MDP framework introduced in the previous section, where \( \Delta \theta_i \) symbolizes changes in policy parameters, guided by the critic.

\[ \theta \leftarrow \pi_\theta(s) \]

A. Policy Architectures

PPO assumes that there is a single agent which fully observes \( S \) and can pick any point in \( A \) as action. Thus, policy \( \pi \) is designed as a neural network with \( 2n \) inputs and \( 2n \) outputs. This raises two questions; concerning our power systems problem: 1) how does the training process scale with \( n \), and 2) is it still possible to perform decentralized control considering the fact that \( \pi \) maps states of all buses to actions at all buses?

We consider two modes of control, centralized and decentralized. In either mode, \( s_i \) and \( a_i \) refer local state and action (at bus \( i \)) respectively. As defined in Eq. (9), 10, each of \( s_i \) and \( a_i \) contains two terms per bus, relating to real power and voltage measurements for \( s_i \), and to changes in real and reactive power setpoints for \( a_i \).

In both cases, baseline value \( V(s) \) (see Section II-B) estimates the expected value of system-wide reward-to-go, or return \( G \) (see Section II-A) having started at some state \( s \). It is used only during training, not during execution of the policy, and is labeled as critic in Fig. 3. It is worth noting here that in the decentralized control setting, agents are not designed to compete for local reward maximization, rather they are trained to maximize global (system-wide) reward. In that sense, the voltage regulation problem is not a Markov game.

1) Centralized control: In this mode, we assume the existence of a communication infrastructure that can receive local measurements from every bus in \( C \), and transmit commands back to the photovoltaic inverters, to change real and reactive power setpoints. Those values are determined using a fully connected neural network that maps \( S \) to \( A \). This network is parametrized by a group of weights and biases, denoted collectively as \( \theta \). The PPO algorithm introduced in the previous section optimizes over this \( \theta \), in search of optimal policy \( \pi_\theta \), where

\[ a \leftarrow \pi_\theta(s) \]

2) Decentralized control: Based on experience in the domain of power systems, we know it’s possible for voltage to be regulated by DER’s locally, albeit sub-optimally, for example using droop control. We propose a neural network architecture for \( \pi \) that connects input to output only at the same bus, rendering it equivalent to a decentralized controller, to compete with conventional methods. That is, there are \( n \) neural networks in parallel, each with just 2 inputs and 2 outputs. Each of those smaller networks is parametrized by a group of weights and biases, denoted collectively as \( \theta_i \), and notation \( \pi_\theta \) is shortened to \( \pi_i \). This time, the PPO algorithm optimizes over \( \theta_1, \ldots, \theta_n \), in search of optimal policies \( \pi_1, \ldots, \pi_n \), where

\[ a_i \leftarrow \pi_i(s_i) \quad \forall i \in C \]

Note that the only difference between this case and the centralized case (optimizing over \( \theta \)), is that here we enforce the strict rule that all neural network weights connecting states at bus \( i \) to actions at bus \( j \) are fixed at zero iff \( i \neq j \). That is, the optimizer (e.g. Adam optimizer in PyTorch) is told to ignore those weights (initialized and left at zero). One can also replace the condition \( i \neq j \) with \((i, j) \notin \mathcal{L}\), if the desired setup involves neighboring buses communicating with one another.

We perform orthogonal initialization on all neural network weights and assign very small initial values to those in the last layer to prevent instability in the feedback controller.

3) Comparison: Based on numerical simulations, as shown in Fig. 5, we have found that the decentralized agent is more sample efficient and trains with less fluctuations and variance in episodic rewards over the learning process. On the other hand, the centralized agent takes a bit less computation time (about 20% less) per iteration, yet more iterations to converge.

VI. NUMERICAL SIMULATION

In this section, we apply the proposed policy architecture and use PPO to solve the MDP. Numerical simulations are conducted on a 240-node distribution grid (see Fig. 4) using...
OpenDSS to solve unbalanced power flow. All parameters associated with this network are obtained from real line parameters and real load data, based on an anonymous distribution grid in Midwest U.S. [24]. Experiment details (e.g. software and hardware details) are found in the Appendix A.

A. Simulation Setup

The RL agent interacts with the distribution grid every 10 ms (the time step), and each episode contains 100 time steps, for a total of one second per episode. $\mu$ is set to 0.1 to favor voltage regulation over solar production maximization. We use the distribution grid shown in Fig. 4 where $N = 240$, and we select $n = 16$ and $n = 194$ for the case studies that follow. 194 is the number of controllable nodes provided originally with the OpenDSS model of this grid. For each of these 194 nodes, we have 1 year of real historical load data ($P^l$, $Q^l$), which we take advantage of to generate random samples for our simulation at the start of every episode.

Since each episode is 1 second, it is fair to assume that fluctuations in $p^{mv}$ and $(P^l, Q^l)$ are negligible within one episode. For this reason, during the training/learning process, a reset command is issued at the beginning of each episode to randomly generate and fix $p^{mv}$ and $(P^l, Q^l)$ for the remainder of the episode. Nonetheless, upon execution, we allow for variations in those quantities within an episode. Note that the agent experiences different system operating conditions every episode during the training process.

Since every episode is 1 second, it is fair to assume that fluctuations in $p^{mv}$ and $(P^l, Q^l)$ are negligible within one episode. For this reason, during the training/learning process, a reset command is issued at the beginning of each episode to randomly generate and fix $p^{mv}$ and $(P^l, Q^l)$ for the remainder of the episode. Nonetheless, upon execution, we allow for variations in those quantities within an episode. Note that the agent experiences different system operating conditions every episode during the training process.

Fig. 4: Real distribution grid located in the Midwest U.S. [24], with 240 nodes excluding the substation node.

B. Case Study on smaller (16-bus) subsystem

In this case study, we compare the use of a centralized policy to that of a decentralized policy, presented in Section V-A.

Since $n = 16$, neural networks of both centralized and decentralized policies have 32 inputs and 32 outputs. The standard choice of 2 hidden layers with 64 neurons in each layer is made for the centralized policy, with $\tanh(\cdot)$ activation functions, whereas the decentralized policy splits into 16 sub-policies each with 2 inputs, 2 outputs and two hidden layers each with 4 neurons. This gives both the centralized and decentralized policies a ‘height’ of 64 neurons in the hidden layer ($4 \times 16 = 64$), but a total of 8352 parameters to tune for the former and 672 for the latter. In fact, in the decentralized case, we assign 16 different Adam optimizers, one to tune each sub-policy, so it’s not so much 672 parameters to optimize in each PPO iteration, rather 42 per optimizer, compared to 8352 per (single) optimizer in the centralized setting.

The training curves for each is shown in Fig. 5 where each ‘PPO iteration’ on the x-axis refers to 2048 steps of interacting with the environment (or 20s, considering 10ms time step). It is evident that the centralized agent does not out-perform the decentralized agent, and is clearly less interpretable and requires a wide communication infrastructure to implement in practice. Note: in both centralized and decentralized cases, value function $V$ is centralized (fully connected neural network). That is, the RL agent is centralized during training (computer simulation), but decentralized during execution (real-world).

In classic RL benchmarks, a threshold is chosen to determine when the learning problem is solved. In our context, the threshold is 0 as shown in Fig 5 and justified as follows. We know that $R_V \leq 0$ and $R_P \geq 0$, from Eq. (3g,3f). Both terms have been designed in such a way that the magnitudes of the rewards are of order 1 or less during normal operating conditions. Moreover, say the user desires to keep voltages within $1 \pm \delta$. It is then a fact that $(R_V + \mu R_P) \geq 0$ at every bus where voltage is within the desired region. It logically follows that if the inequality does not hold, then the voltage at the bus is certainly outside the desired region. We extend this to $n$ buses: if the total system reward is negative, we know that not all voltages are inside $1 \pm \delta$. This necessary condition on voltage serves as a useful tool for monitoring progress in the reinforcement learning process, as shown in Fig. 5. Note: in that figure, the term ‘voltage safety’ simply refers to voltages being inside $1 \pm \delta$. The decentralized agent permanently crosses this threshold in 2 iterations, while the centralized takes 4 to do so.

By these results, we claim that one can obtain results for a decentralized agent that are similar to, or even better than, those for a centralized agent, simply by manipulating the
always the best strategy to absorb all the solar power available.

distribution grid based on real parameters show that it is not production maximization. Numerical simulations on a 240-node
tance between voltage deviation minimization and solar pro-
Markov Decision Process with rewards parametrized to bal-
penetration. The voltage regulation problem is posed as a
power support for distribution grids with deep photovoltaic
ade control strategy with joint provision of real and reactive
power systems perspective. We ask the following question:
what is the impact of joint real and reactive power control (as opposed to just the latter) on system-wide voltage profile in
the midst of deep photovoltaic penetration?
Consider \( n = 194 \) buses, and the same grid as before, with controllable real and reactive power inverter setpoints. As
shown in Fig. 6 when maximum real power is drawn from the solar panels, leaving less reactive power support, deep
photovoltaic penetration causes over-voltage. Terms Proportional Reactive and Integral Reactive refer to policies where
maximum power is injected and whatever remains within inverter limits is used for reactive power compensation to
regulate voltage. With joint real and reactive power, the RL agent manages to keep voltage within user-defined safety
region \((1 \pm \delta)\). Surprisingly, a small reduction in real power injection was needed to achieve this effect. Fig. 7 shows the
steady state distribution of real power consumption per bus, as a ratio to maximum possible injection \((p^{env})\). It is worth noting
how well the voltage was improved system-wide, even though most solar panels produced near maximum output (note the
0.85 on the y-axis of both figures 6 and 7).

C. Case Study on larger (194-bus) subsystem
In the previous subsection, we compared centralized and
decentralized policy architectures. In this subsection, we dig
deeper to examine our proposed framework from a purely
power systems perspective. We ask the following question:
what is the impact of joint real and reactive power control (as opposed to just the latter) on system-wide voltage profile in
the midst of deep photovoltaic penetration?

This paper also proposes and verifies a fully decentralized (communication-free) approach for this type of control, which
can be implemented on existing physical infrastructure, helping
alleviate problems related to communication failure or
cyber attacks. In future work, competition between agents is
considered, whereby the inverter at each bus seeks to maximize
local, not system-wide, rewards.

Appendix A
Experiment Details
Simulations are conducted in Python 3.7, interfacing with
OpenDSS \cite{opendss} and using PyTorch \cite{pytorch} to model, build and train actor and critic neural networks. Machine: Lenovo, 64-
bit Windows 10, Intel®Core™ i7-6700HQ CPU @ 2.60Ghz, 16.0 GB RAM.

Fig. 6: Comparison between control policies under deep photovoltaic penetration. Each marker represents a steady-state
value of \( P^*/p^{env} \) vs. \( V \) at one bus.

VII. Concluding Remarks
This paper introduces a reinforcement learning-based volt-
age control strategy with joint provision of real and reactive
power support for distribution grids with deep photovoltaic
penetration. The voltage regulation problem is posed as a
Markov Decision Process with rewards parametrized to bal-
cence between voltage deviation minimization and solar pro-
duction maximization. Numerical simulations on a 240-node
distribution grid based on real parameters show that it is not
always the best strategy to absorb all the solar power available.

Fig. 7: Histogram of \( P^*/p^{env} \) under RL policy, using results of Fig. 6. Most buses inject near maximum real power.

REFERENCES
\cite{bib:1} M. Farivar, R. Neal, C. Clarke, and S. Low, “Optimal inverter VAR control in distribution systems with high PV penetration,” in 2012 IEEE Power and Energy Society General Meeting.
\cite{bib:2} J. Seuss, M. J. Reno, R. J. Broderick, and S. Grijalva, “Improving distribution network pv hosting capacity via smart inverter reactive power support,” in 2015 IEEE Power Energy Society General Meeting, 2015, pp. 1–5.
\cite{bib:3} V. Kekatos, G. Wang, A. J. Conejo, and G. B. Giannakis, “Stochastic reactive power management in microgrids with renewables,” IEEE Transactions on Power Systems, vol. 30, no. 6, pp. 3386–3395, 2015.
\cite{bib:4} “IEEE standard for interconnection and interoperability of distributed energy resources with associated electric power systems interfaces,” IEEE Std 1547-2018.
\cite{bib:5} X. Su, M. A. S. Masoum, and P. J. Wolfs, “Optimal pv inverter reactive power control and real power curtailment to improve performance of unbalanced four-wire lv distribution networks,” IEEE Transactions on Sustainable Energy, vol. 5, no. 3, pp. 967–977, 2014.
\cite{bib:6} B. Zhang, A. Y. Lam, A. D. Dominguez-Garcia, and D. Tse, “An optimal and distributed method for voltage regulation in power distribution systems,” IEEE Transactions on Power Systems, vol. 30, no. 4, pp. 1714–1726, Jul. 2015.
\cite{bib:7} H. Zhu and H. J. Liu, “Fast local voltage control under limited reactive power: Optimality and stability analysis,” IEEE Transactions on Power Systems, vol. 31, no. 5, pp. 3794–3803, Sep. 2016.
\cite{bib:8} W. Lin, R. Thomas, and E. Bitar, “Real-time voltage regulation in distribution systems via decentralized pv inverter control,” in Proceedings of the 51st Hawaii International Conference on System Sciences. Hawaii International Conference on System Sciences, 2018.
\cite{bib:9} G. Qu and N. Li, “Optimal distributed feedback voltage control under limited reactive power,” arXiv preprint arXiv:1810.11121v1, 2018.
