### Title

Deep Reinforcement Learning for Mitigating Cyber-Physical DER Voltage Unbalance Attacks




### ISBN

978-1-6654-4197-1



### Authors

Roberts, Ciaran

Ngo, Sy-Toan

Milesi, Alexandre

et al.



### Publication Date

2021-05-28

### DOI

10.23919/acc50511.2021.9482815

Peer reviewed

# Deep Reinforcement Learning for Mitigating Cyber-Physical DER Voltage Unbalance Attacks

Ciaran Roberts†, Sy-Toan Ngo†, Alexandre Milesi†, Anna Scaglione⋆, Sean Peisert†, Daniel Arnold†
†*Lawrence Berkeley National Laboratory* {cmroberts,sytoanngo,amilesi,sppeisert,dbarnold}@lbl.gov
⋆*Arizona State University* ascaglio@asu.edu

***Abstract*— The deployment of DER with smart-inverter functionality is increasing the controllable assets on power distribution networks and, consequently, the cyber-physical attack surface. Within this work, we consider the use of reinforcement learning as an online controller that adjusts DER Volt/Var and Volt/Watt control logic to mitigate network voltage unbalance. We specifically focus on the case where a network-aware cyber-physical attack has compromised a subset of single-phase DER, causing a large voltage unbalance. We show how deep reinforcement learning successfully learns a policy minimizing the unbalance, both during normal operation and during a cyber-physical attack. In mitigating the attack, the learned stochastic policy operates alongside legacy equipment on the network, i.e. tap-changing transformers, adjusting optimally predefined DER control-logic.**

## I. Introduction

The proliferation of distributed energy resources (DER) in electrical distribution systems is causing a fundamental shift in how these networks are operated. Historically, these networks had minimal controllable devices and exhibited largely predictable time-varying demand profiles. This, however, is changing with the increase in customer-owned DER. This increase in controllable devices, coupled with a shifting resource ownership model, presents new challenges in reliably operating our electrical power networks, particularly in the context of cyber-physical security [1], [2].

Residential DER, in particular, are unique when it comes to cyber-physical security. These controllable devices are disrupting the traditional resource ownership model in that they are neither utility owned nor directly controlled. Rather, many manufacturers and/or aggregators remotely control large populations of these devices via cellular networks, customers' Wi-Fi routers, or wired internet connections [3]. This makes ensuring the integrity of commands significantly more difficult and presents a new attack vector for adversaries seeking to disrupt distribution grid operations. A single breach into the network of a single manufacturer and/or aggregator could result in a rollout of malicious DER controller settings to an entire DER fleet [3].

Within this work, we adopt a purely physics-based approach for the mitigation of cyber-physical attacks on DER. That is, the proposed approach only relies on locally sensed electrical measurements, rather than information about the communication network. We build upon our previous work [4], focused on three-phase DER, balanced networks and voltage oscillatory behavior, by considering the case where an adversary seeks to create a network voltage unbalance (VU) by exploiting standardized smart inverter functionality and the single-phase nature of residential DER. Such an attack may seek to trip VU relays and/or cause sensitive equipment to trip offline [5]. Similar to [4], we assume that the adversary has already gained access to a subset of network DER and seeks to maliciously re-configure their control logic to disrupt distribution grid operations.

VU is one of the main power quality concerns for distribution utilities, with standards and/or requirements establishing VU limits [6], [7]. Historically, the major cause of VU has been the unequal distribution of single-phase loads within a three-phase distribution network [5]. Recently, however, the addition of single-phase residential photovoltaic (PV) generation had further raised the level of concern [8]. Previous work has examined VU in low-voltage networks, primarily due to inherent unequal load distribution [9]–[11]. The authors in [9] explore dynamically switching single-phase residential customers between phases, via static transfer switches, in order to minimize VU. The authors in [10] propose a secondary control loop at the inverter level for VU compensation in islanded microgrids, while in [11] a parallel positive- and negative-sequence control loop for DER VU mitigation is presented. These proposed solutions require either extensive communication and switching capabilities [9] and/or redesigning the control loops of DER [10], [11].

This work differs in that we seek to control residential DER, subject to existing minimal control requirements codified in IEEE 1547 [12], to mitigate VU. We are particularly focused on the case where an adversary has gained control of a subset of the DER and seeks to exploit these aforementioned minimal control requirements to cause an abnormally large and disruptive VU. We consider the application of Deep Reinforcement Learning (DRL) for learning an online controller that re-dispatch the Volt/Var (VV) and Volt/Watt (VW) settings of single-phase residential PV inverters. The controller is trained offline through extensive simulation and deployed in a distributed manner, where local communication between neighbors is required to estimate VU.

Reinforcement learning (RL) has been gaining increasing attention in recent years, including in power systems, for

This research was supported in part by the Director, Cybersecurity, Energy Security, and Emergency Response, Cybersecurity for Energy Delivery Systems program, of the U.S. Department of Energy, under contract DE-AC02-05CH11231. 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.

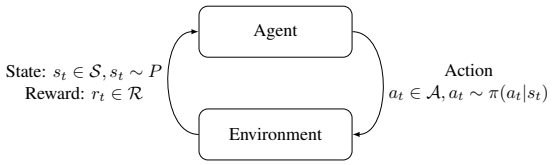

Fig. 1: Reinforcement learning loop.

determining control policies for highly complex non-linear systems. In [13], the authors explored the use of DRL for optimal energy management among grid-connected microgrids. Deep Q-Network (DQN) learning, a RL algorithm that combines Q-Learning with deep neural networks, was employed in [14] to control both generator dynamic braking and load shedding in the event of a contingency to ensure post-fault recovery. In [15], the authors use a DQN network to learn an optimal policy for capacitor bank switching. In [16], a deep deterministic policy gradient (DDPG) RL agent is used to co-ordinate DER and directly modulate active and reactive power to regulate the grid voltage during normal operations. This work further examines the use of DRL for determining optimal control policies, with particular emphasis on mitigating the effects of cyber-physical attacks.

The remainder of the paper is organized as follows. Section II gives a brief introduction to DRL and the terms we use throughout the paper. Section III gives an overview of the power system models used in the study. Finally, Section IV presents the results and Section V summarizes some of the key conclusions.

## II. Background on Reinforcement Learning

Reinforcement learning refers to an area of machine learning where the goal is to determine, through training, the optimum policy for a decision-maker (called the *agent*) that maximize a cumulative expected reward while choosing actions that modify the stochastic environment. The interaction between the agent and the environment is depicted in Fig. 1. In its basic instantiation, RL is a Markov Decision Process (MDP), whose formal definition includes the following elements:

- A state space $\mathcal{S}$;
- An action space $\mathcal{A}$, including all actions the agent can perform;
- A state transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$, indicating the probability of transitioning to state $s_{t+1}$ when an action, $a_t$, is taken at state $s_t$;
- A reward function $\mathcal{R} : \mathcal{A} \times \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$, specifying the reward, $r$, the agent receives when action $a$ is taken in state $s$, and the environment transitions to state $s'$;
- A discount factor, $\gamma \in [0, 1]$, weighting the value of future rewards in the expression of the expected discounted reward $J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \gamma^t r_t\right]$, with $T$ is the end of the optimization horizon (possibly infinite).

The goal of the agent is to learn a policy $\pi(a|s)$ (deterministic or stochastic) mapping the state, $s_t$, onto optimal actions, $a_t$, that maximize the expected cumulative environment reward over a given time horizon. The policy is derived from successive interactions with the environment, yielding rewards, $r_t \in \mathbb{R}$, dependent on the actions taken, $a_t \in \mathcal{A}$, at a given state, $s_t \in \mathcal{S}$. As the figure shows, at state $s_t$, the agent generates action $a_t$ from $\pi(a_t|s_t)$. Applying $a_t$ causes the environment to transition to state $s_{t+1}$ which subsequently generates a reward $r_t$ given to the agent. The tuple $(s_t, a_t, r_t, s_{t+1})$ serves as input to the agent which uses the information to update $\pi(a|s)$ to maximize the discounted reward $J(\pi)$.

### A. Deep Reinforcement Learning

Classical RL presents a number of problems. The first challenge is to train a policy to apply to environments with high dimensional continuous action and/or state spaces [17]. The remedy is to discretize theses action and/or state spaces first. Unfortunately, due to the *curse of dimensionality*, this results in a combinatorial explosion in complexity and unreasonable training time. Another challenge is the convergence of RL training in the presence of noisy or incomplete data.

DRL solves these issues by leveraging neural networks with multiple hidden layers that take agents observations as input and output a policy indicating what is the most advantageous action to take for the given state.

The weights of these neural networks are efficiently learned end-to-end via gradient-based optimization to find the best intermediate features and an optimal output policy. This greatly reduces the need for precise feature engineering, due to the automatic high-dimensional feature extraction of the hidden layers. In DRL, we can use a neural network to explicitly approximate an optimal policy distribution $\pi$ over possible actions. The agent then samples this distribution to determine the next action, as in Policy Gradient methods. They may also be used to approximate either a value function, $V^\pi(s)$, or an action-value function, $Q^\pi(s, a)$, from gathered data, leading to an action decision based on inferred values for all possible future states, as in DQN. The value function, $V^\pi(s)$, is the expected discounted reward when starting in state $s$ and following the policy $\pi$, whereas the action-value function $Q^\pi(s, a)$ is defined as the expected discounted reward when starting in state $s$, taking action $a$, and then following the policy $\pi$ thereafter.

### B. Policy Gradient and PPO

Policy Gradient methods employ a policy modeled by a neural network that is trained directly by gradient ascent on the expected return. The most basic method, Vanilla Policy Gradient, is simple to implement but has the drawback of having a high gradient variance. In response, actor-critic (AC) methods were proposed[18], where another, possibly shared, neural network approximates the value function.

Let $\pi_\theta(a|s)$ indicate a stochastic policy, where $\theta$ refers to the parameters of the deep neural network, and let $V_\phi^\pi(s)$ denote a value function approximation by a deep neural network with parameters $\phi$, estimating the cumulative discounted reward from the current state to the terminal state.

The gradient of $J(\theta)$ is:

$$\nabla_\theta J(\theta) = \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\nabla_\theta \log \pi_\theta(a_t|s_t) A^\pi_\phi(s_t,a_t)\right], \quad (1)$$

where $\tau$ is the trajectory generated by $\pi_\theta$, and $A^\pi_\phi(s_t,a_t) = r_t + \gamma V^\pi_\phi(s_{t+1}) - V^\pi_\phi(s_t)$ is the *advantage function* estimation, representing how much better taking action $a_t$ is, as opposed to following the policy $\pi$ when in state $s_t$. The policy and value function is then updated by gradient ascent/descent:

$$\theta_{k+1} = \theta_k + \alpha\nabla_\theta J(\theta), \quad (2)$$

$$\phi_{k+1} = \phi_k - \beta\nabla_\phi (r_t + V^\pi_\phi(s_{t+1}) - V^\pi_\phi(s_t))^2. \quad (3)$$

When the data distribution changes due to large policy updates, the training of AC methods can be unstable. The Trust Region Policy Optimization (TRPO) was introduced [19] to remedy this problem by enforcing a Kullback–Leibler divergence constraint on the size of each update. The Proximal Policy Optimization (PPO) [20] of the corresponding problem amounts to using a clipped surrogate objective, yielding similar performance:

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

$$\text{where } \rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta\,\text{old}}(a_t|s_t)} \text{ and } \hat{A}_t = A^\pi_\phi(s_t,a_t)$$

It is intuitive that the clip operator induces more gradual updates to the policy compared to the unconstrained gradient descent, since the minimum between the unclipped and the clipped objective is used, also implying that the final objective is a lower bound on the unclipped objective [20]. The $\hat{(\cdot)}$ operator over the expectation means that we compute a Monte Carlo estimate.

PPO is a state-of-the-art method that has successfully been used in video games [21] and robotics in simulation [22]. Within this work, we empirically found PPO to be the most stable for the use cases considered and therefore present its application.

Note that in many applications, including ours in this paper, the RL agents do not get to observe the state of the entire system, $s_t$, but rather a function of it, called the observation, $o_t$. Such RL settings fall in the class of Partially Observable MDPs (POMDPs). In a POMDP, the additional element in the model is:

- An observation transition function (also called perceptual distribution or emission probability) $\mathcal{V} : \mathcal{S}\times\mathcal{O} \to [0,1]$ that specifies the probability distribution of the observation $o_t$ given the state $s_t$.

Rather than the state, the policy function in this case takes as input the observation, i.e. the goal is to find the optimum probability density function $\pi(a_t|o_t)$. Also in this case, expectations are approximated via realistic Monte Carlo simulations.

## III. Methodology

Within this work, we seek to train an intelligent agent that continuously monitors grid conditions and adaptively adjusts the VV/VW control logic of distributed DER. This is depicted in Fig. 2 where the agent takes as an input an observation vector, of locally measured grid conditions, and outputs an action that re-configures the VV/VW control logic.

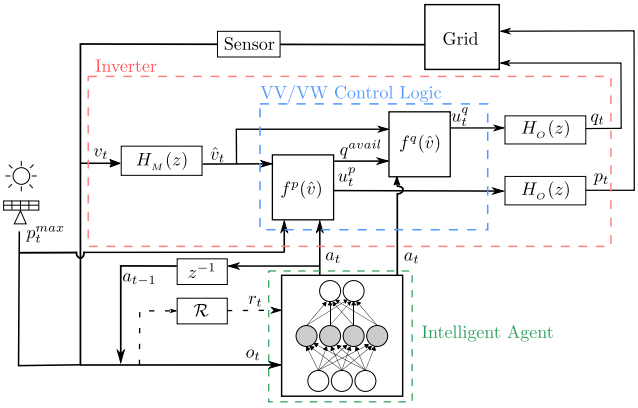


Fig. 2: DER with RL agent.

### A. Modeling DER

Recent standards, such as IEEE 1547 and California Rule 21, provide guidelines that enable DER to dynamically respond to locally measured voltage. Following these standards, smart inverters will modulate their power outputs to ensure nodal voltages are kept within acceptable limits. This modulation is dictated by VV/VW control logic, often referred to as "droop" curves, which adjust the injected active and reactive power according to piece-wise linear functions of voltage.

Figs. 3 - 4 depict the VV and VW control functions, $f^q(\hat{v})$ and $f^p(\hat{v})$ respectively, parameterized by the voltage breakpoints, $\eta_1$ - $\eta_5$. We note that different parameterizations of these functions exist (e.g. purely linear, or consisting of more piece-wise linear segments), but the functions shown in Figs. 3 - 4 reflect the dominant implementation. As shown in Fig. 2, these functions take as input a low-pass filtered measurement of the grid voltage, $\hat{v}_t$, and output an active and reactive power setpoint, $u^p_t$ and $u^q_t$ respectively. These setpoints are themselves passed through a low-pass filter to limit the ramp rate of active and/or reactive power injection into the grid.

Under VW precedence, priority is given to the VW controller to determine any required active power curtailment before determining the VARs available ($q^{\text{avail}}$). After $q^{\text{avail}}$ is fixed, $u^q_t$ is computed.

In the event of a cyber-physical attack, we assume that an adversary has the capability to re-dispatch a set of voltage breakpoints (e.g. $\eta_1$ - $\eta_5$) that parameterize the droop curves in Figs. 3 - 4 for a subset of DER on the network in order to cause VU. Within the context of this work, the remaining

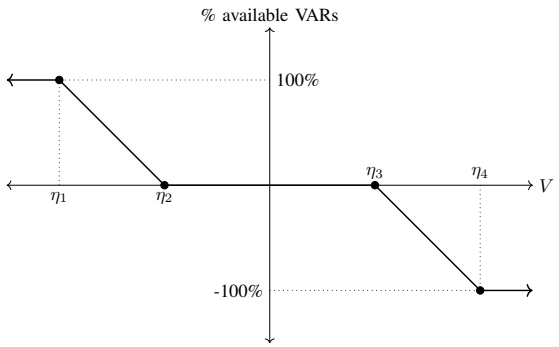


Fig. 3: Inverter Volt-VAR curve. Positive percent denotes VAR injection.

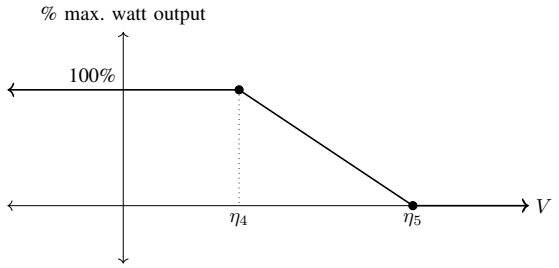


Fig. 4: Inverter Volt-Watt curve. Positive percent denotes watt injection.

set of non-compromised DER can then be updated with new parameters ($\eta_1$ - $\eta_5$) to re-shape their own local droop curves to mitigate VU. The VU at bus $i$ at time $t$ is calculated using (4), similar to the equation in [23]

$$\text{vu}_{i,t} = \frac{\max(|\bar{v}_{i,t} - \bar{v}^a_{i,t}|, |\bar{v}_{i,t} - \bar{v}^b_{i,t}|, |\bar{v}_{i,t} - \bar{v}^c_{i,t}|)}{\bar{v}_{i,t}} \quad (4)$$

where $\bar{v}_i$ denotes the mean measured voltage magnitude at bus $i$, and $\bar{v}^a_{i,t}$, $\bar{v}^b_{i,t}$, $\bar{v}^c_{i,t}$ are the measured voltage magnitudes on phase $a$, $b$, and $c$ respectively.

### *B. Training Environment*

The primary goal of the DRL controller is to mitigate VU, particularly VU caused by DER smart inverter VV/VW controllers due to maliciously chosen setpoints. Let the graph $G = (\mathcal{N}, \mathcal{L})$ represent the topology of the distribution feeder considered, where $\mathcal{N}$ is the set of nodes of the feeder and $\mathcal{L}$ is the set of lines. For simplicity of presentation, we assume the presence of a VV/VW capable smart-inverter at every node in the system, so that the total number of inverters in the system is $|\mathcal{N}|$. We suppose the set $\mathcal{N}$ is partitioned into two sets, $\mathcal{H}$ and $\mathcal{U}$, which represent the "compromised" and '"uncompromised" inverters respectively, where $\mathcal{H} \bigcup \mathcal{U} = \mathcal{N}$. Furthermore, we assume that $\mathcal{U} \neq \emptyset$, i.e. we have some controllable resources to mitigate the effects of the cyber-physical attack.

**Training**: Rather than training multiple agents simultaneously, we adopt the following heuristics to aid convergence:

1) For agent training, we define a single agent whose input observation vector is from a single node in the network at time $t$ (e.g. worst case VU or feeder head) and whose multi-head output action, $a^i_t \forall i \in \{a, b, c\}$, is a deviation/offset, $\Delta\eta$, from default VV/VW control curves that apply across single-phase inverters.
2) Once a single agent has been trained, this agent is deployed in a distributed manner and only requires sharing information among its immediate neighbors. This information is necessary for single phase inverters to estimate the voltage unbalance and determine the output action.
3) Rather than optimize over arbitrarily shaped VV/VW curves, we optimize over the deviation, i.e. $a = \Delta\eta$, from the default parameters defining the curves in Figs. 3 - 4. We assume that this default parameterization is determined by an upper-level optimization and corresponds to desirable grid operating conditions. An example of an action is shown in Fig. 5. The action range is from -0.1 pu to 0.1 pu around an inverters default VV/VW curve, with the action space being discretized into $k$ bins.
4) New parameterizations of VV/VW functions will be chosen so that measurement and power injection dynamics evolve on a faster timescale. This choice will preserve the Markov property between actions taken by the RL controller.

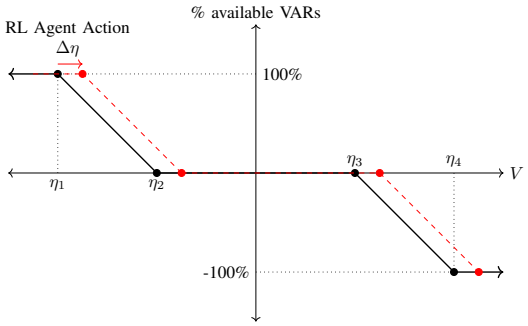


Fig. 5: Action example.

**Observation**: The observation vectors $o_{i,t}, i \in \mathcal{U}$ at each RL agent (i.e. the input to the neural network that learns the optimum policy $\pi(a|o)$), consist of:

1) $\text{vu}_t$: An estimation of the VU at time $t$
2) $q_t^{\text{avail, nom}}$: the available reactive power capacity without active power curtailment.
3) $a^{\text{a}}_{t-1}$, $a^{\text{b}}_{t-1}$, $a^{\text{c}}_{t-1}$: one-hot encoding of the previous action taken by the agent across each phase.
4) $v^{\text{a}}_t$, $v^{\text{b}}_t$, $v^{\text{c}}_t$: voltage phase measurements at time $t$

During the training of the agent, the VU estimation and voltage measurements will be from a specific node of interest, e.g. worst case VU, and $q_t^{\text{avail, nom}}$ will be the average value across all uncompromised inverters. When the agent is deployed in a distributed manner, the VU estimation and voltage measurements will be form the nearest three-phase node.

**Reward**: At a time $t$, the reward function, $r_t(a_t, o_t)$ for

our single agent in training is:

$$r_t = -\left(\sigma_u||\text{vu}_t||_\infty + \sum_{i\in\{a,b,c\}} \sigma_a \mathbf{1}_{a_t^i \neq a_{t-1}^i} + \sum_{i\in\{a,b,c\}} \sigma_0 \|a_t^i\|_2 + \frac{1}{|\mathcal{U}|}\sum_{j=1}^{|\mathcal{U}|} \sigma_p \left(1 - \frac{p_{j,t}}{p_{j,t}^{\max}}\right)^2\right). \tag{5}$$

The first term seeks to minimize the maximum VU, $||\text{vu}_t||_\infty$, over all nodes in the network; the second term penalizes configuration changes on inverters; the third component encourages the agent to use the default inverter configurations in the absence of VU and the final component penalizes any active power curtailment. We penalize the agent for deviating from its default parameterization as we assume that this default paramaeterization was determined by some higher-level optimization under normal grid conditions. Therefore, in the absence of any abnormal VU, we would like our agent to remain close to these default parameterizations. The reward in (5) penalizes the agent for the maximum VU across the entire network. For networks with known vulnerable equipment, (5) could instead penalize the VU at those specific nodes.

## IV. Results

### A. Experiment setup

We conduct experiments on an IEEE 37-bus feeder with all load buses having a peak active power generation of 100% of the nominal load with an additional 10% inverter over-sizing for reactive power headroom. The agent trains in environments of 700 one-second timesteps in OpenDSS. Load, solar generation, percentage of the compromised inverters, and the phase the voltage regulator monitors are all randomized at environment reset to create rich scenarios for the agent to train. The voltage regulator interacts with the RL agent through the system voltages. That is, should the action of the agent push the voltage outside the deadband of the regulator, the regulator will begin a countdown timer and actuate if the voltage does not re-enter its deadband. At a specific time in the simulation, the attacker controls 10% to 40% of all inverter capacity in the power grid to create a voltage unbalance.

The agent is allowed to translate the VV/VW curves (offset action) of the uncompromised single-phase inverters to minimize voltage unbalance. Within this experiment we discretized the action into $k = 21$ bins. This was chosen to balance the granularity of control with the increase in training time. The reward weights, $\boldsymbol{\sigma}$, were chosen so that the penalty for a 1% voltage unbalance was an order of magnitude more than the penalty for taking an action. Within this range, hyper-parameter tuning was done to tune $\boldsymbol{\sigma}$ to achieve what was deemed satisfactory behavior. Fig 6 shows the mean and standard deviation of the total reward over 10 runs for the experiment considered.

Fig **??** shows the baseline case in the morning caused by 30% of the connected DER becoming compromised with no further action taken to minimize the VU. At $t = 200s$, the attacker maliciously re-configures the VV/VW of the compromised inverters to create a VU within the network.

Fig. 7 shows the behavior of the trained RL agent with active control for the same case. We see that prior to the attack the agent takes an action to minimize the inherent VU in the system due to unequal load distribution. After the attack, the agent action moves the VV/VW curve of phase B to lower the voltage on phase B and reduce the VU. Due to the low voltage on Phase A, the inverter is already operating in the saturation region of its VV curve and injecting its maximum available reactive power. Therefore, the reward is maximized by the agent taking no action on this phase. The RL agents are applied in a distributed manner, i.e., each agent receives a different local observation, and, correspondingly, may take a different action. We see this manifest itself in Fig. 7 where a subset of agents on Phase B take a different action at t $\approx$ 320s due to a change in their local observation vector.

Fig 8 and 9 show the behavior of the agent around midday and in the morning respectively with 40% of the connected DER becoming compromised. The scenarios differ in the nominal available reactive power without active power curtailment. For the attack considered in Fig 8 we can see that the agent significantly reduces the VU and keeps it under 2%. In Fig. 9, however, we observe that a subset of the agents exhibit an oscillatory action. This oscillatory action behavior may be attributed to both the severity of the attack as well as the training approach adopted, where a single agent was trained to minimize the worst case VU. Although the training exhibited stable convergence, once these agents were deployed in a distributed manner the disparate observation vectors resulted in unexpected behavior. Future work will seek to increase the number of output action heads, where each output action head controls a cluster of DER within a distribution network [24]. This will result in a more network-aware optimal policy while keeping training time computationally tractable.

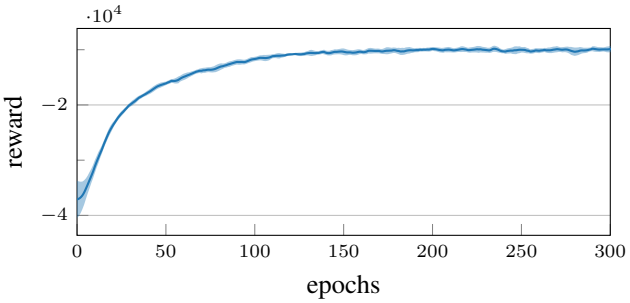


Fig. 6: Average training reward. The band represents the standard deviation over 10 runs.

## V. Conclusions

This paper has proposed a reinforcement learning approach for mitigating voltage unbalance, specifically due to maliciously re-configured smart inverter settings. We utilize DRL to learn optimal policies for online re-configuration of single-phase VV/VW functions to minimize VU. The

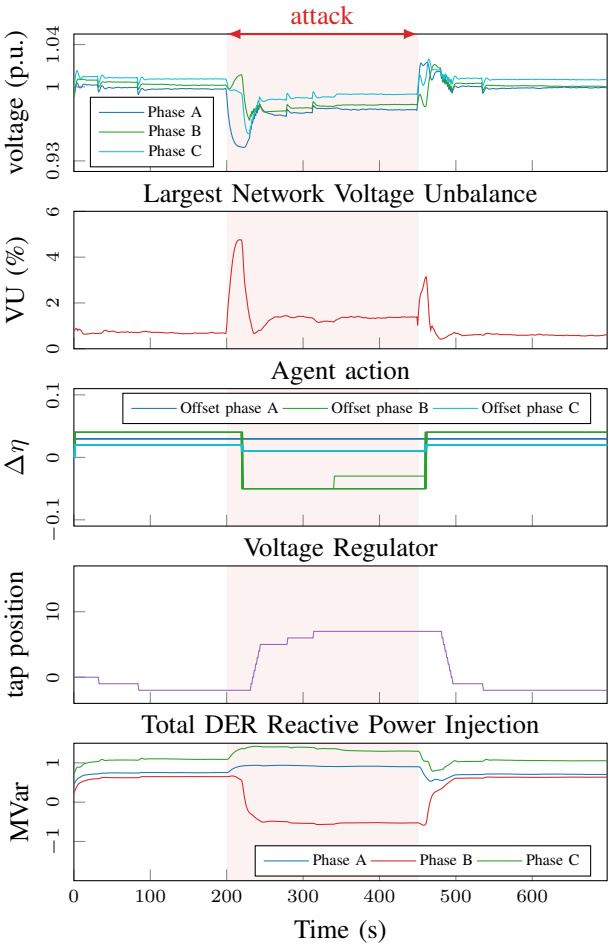


Fig. 7: 30% DER VU attack at 9 A.M

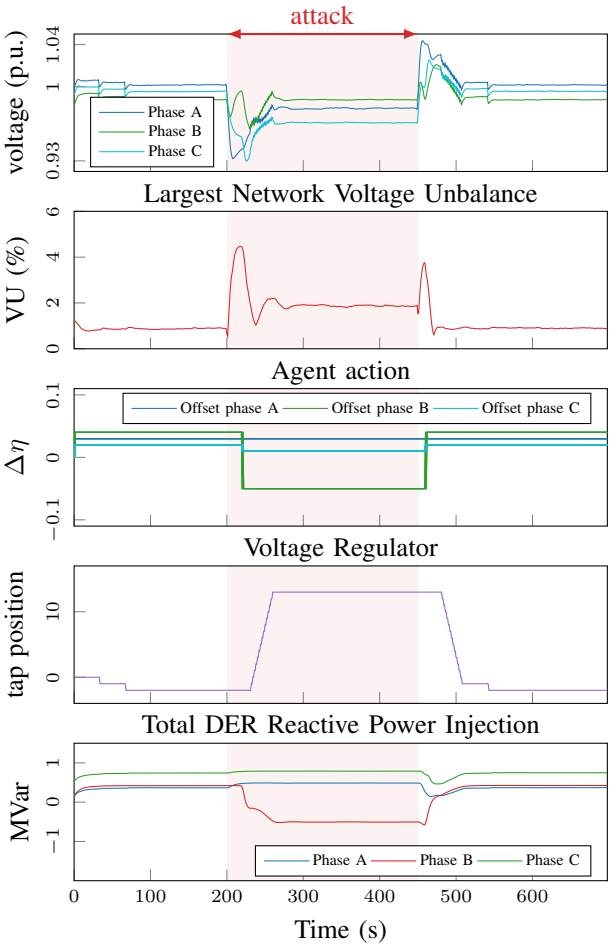


Fig. 8: 40% DER VU attack at 1 P.M

proposed approach successfully mitigated VU for the majority of cases considered. However, it was observed that at higher levels of compromised DER, a subset of agents exhibited oscillatory behavior in their action output. This was attributed to the training of a single policy that applied across all inverters. Future work will seek to cluster inverters within a network and determine a network-aware optimal policy that is dependent on the location of the DER within the feeder and understand the network sensitivity of the resultant trained agents. Additionally, we will investigate the incremental performance gain from having individual offsets for each of the VV and VW droop curves as well as continue to expand the library of attacks considered.

## References


[1] J. Qi, A. Hahn, X. Lu, J. Wang, and C.-C. Liu, "Cybersecurity for distributed energy resources and smart inverters," *IET Cyber-Physical Systems: Theory & Applications*, vol. 1, no. 1, pp. 28–39, 2016.

[2] S. Sahoo, T. Dragičević, and F. Blaabjerg, "Cyber security in control of grid-tied power electronic converters–challenges and vulnerabilities," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 2019.

[3] "Modernizing Hawai'i's Grid For Our Customers," Tech. Rep., 2017.

[4] C. Roberts, S.-T. Ngo, A. Milesi, S. Peisert, D. Arnold, S. Saha, A. Scaglione, N. Johnson, A. Kocheturov, and D. Fradkin, "Deep reinforcement learning for der cyber-attack mitigation," in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2020, pp. 1–7.

[5] A. Von Jouanne and B. Banerjee, "Assessment of voltage unbalance," *IEEE Transactions on Power Delivery*, vol. 16, no. 4, pp. 782–790, 2001.

[6] "American national standard for electric power systems and equipment—voltage ratings (60 hz)," *ANSI C84.1-2016*, 2016.

[7] "IEEE recommended practice for monitoring electric power quality," *IEEE Std 1159-2019 (Revision of IEEE Std 1159-2009)*, pp. 1–98, 2019.

[8] A. Dubey, S. Santoso, and A. Maitra, "Understanding photovoltaic hosting capacity of distribution circuits," in *2015 IEEE Power & Energy Society General Meeting*. IEEE, 2015, pp. 1–5.

[9] F. Shahnia, P. J. Wolfs, and A. Ghosh, "Voltage unbalance reduction in low voltage feeders by dynamic switching of residential customers among three phases," *IEEE Transactions on Smart Grid*, vol. 5, no. 3, pp. 1318–1327, 2014.

[10] M. Savaghebi, A. Jalilian, J. C. Vasquez, and J. M. Guerrero, "Secondary control scheme for voltage unbalance compensation in an islanded droop-controlled microgrid," *IEEE Transactions on Smart Grid*, vol. 3, no. 2, pp. 797–807, 2012.

[11] S. Acharya, M. S. El-Moursi, A. Al-Hinai, A. S. Al-Sumaiti, and H. H. Zeineldin, "A control strategy for voltage unbalance mitigation in an islanded microgrid considering demand side management capability," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 2558–2568, 2018.

[12] "IEEE standard for interconnection and interoperability of distributed energy resources with associated electric power systems interfaces," *IEEE Std 1547-2018 (Revision of IEEE Std 1547-2003)*, pp. 1–138, 2018.

[13] Y. Cheng, J. Peng, X. Gu, F. Jiang, H. Li, W. Liu, and Z. Huang,

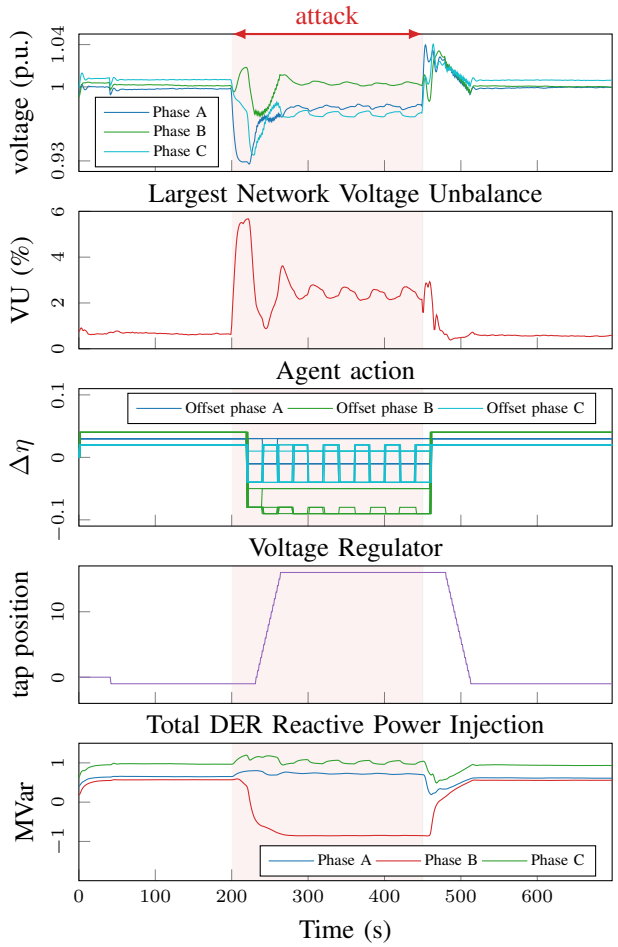


Fig. 9: 40% DER VU attack at 9 A.M

"Optimal energy management of energy internet: A distributed actor-critic reinforcement learning method," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 521–526.

[14] Q. Huang, R. Huang, W. Hao, J. Tan, R. Fan, and Z. Huang, "Adaptive power system emergency control using deep reinforcement learning," *IEEE Transactions on Smart Grid*, 2019.

[15] Q. Yang, G. Wang, A. Sadeghi, G. B. Giannakis, and J. Sun, "Two-timescale voltage control in distribution grids using deep reinforcement learning," *IEEE Transactions on Smart Grid*, 2019.

[16] C. Li, C. Jin, and R. K. Sharma, "Coordination of pv smart inverters using deep reinforcement learning for grid voltage regulation," *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pp. 1930–1937, 2019.

[17] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[18] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 1928–1937.

[19] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International Conference on Machine Learning*, 2015, pp. 1889–1897.

[20] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[21] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[22] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017.

[23] "IEEE standard test procedure for polyphase induction motors and generators," *IEEE Std 112-2017 (Revision of IEEE Std 112-2004)*, pp. 1–115, 2018.

[24] X. Cheng and J. M. Scherpen, "Clustering approach to model order reduction of power networks with distributed controllers," *Advances in Computational Mathematics*, vol. 44, no. 6, pp. 1917–1939, 2018.

## APPENDIX

| **Hyperparameter** | **Value** |
|---|---|
| $\alpha$ (learning rate) | $1 \times 10^{-4}$ |
| $\gamma$ (reward discount factor) | 0.5 |
| $\lambda$ (GAE parameter) | 0.95 |
| $\epsilon$ (PPO clip param) | 0.1 |
| batch size | 500 |
| activation function | tanh |
| network hidden layers | dense (64, 64, 32) |
| $\sigma_u$ (unbalance penalty) | 50000 |
| $\sigma_a$ (action penalty) | 50 |
| $\sigma_0$ (deviation from default parameterization) | 75 |
| $\sigma_p$ (penalty for curtailing active power) | 100 |
| action range | $-0.1$ pu to 0.1 pu |
| $k$ (discretization of action range) | 21 |

TABLE I: Hyperparameters of the network, training and reward