UNIVERSITY OF CALIFORNIA
RIVERSIDE

Data-Driven Monitoring and Control of Smart Grid

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering

by

Yuanqi Gao

September 2020

Acknowledgments

I would like to sincerely thank my Ph.D. advisor Dr. Nanpeng Yu for his guidance and patience throughout my Ph.D. studies at University of California, Riverside (UCR). During my four years of research in his lab, Dr. Yu offered me great research flexibility, and has always been supportive for my research ideas, study plans, and technical writings, for which I found extremely helpful to become competent and independent. Moreover, Dr. Yu sets a great example on the pursuit of perfection and how to communicate with other researchers. I believe I have been strongly influenced by and benefit from his skills. I would also like to thank my committee members, Dr. Yingbo Hua and Dr. Weixin Yao, for providing valuable discussions and feedback during the development of this dissertation, as well as the comments and suggestions for my oral qualification exam.

I deeply appreciate the professors with whom I have taken classes at UCR and my undergraduate alma mater Donghua University. Their knowledge, understanding of the subjects, and passion for teaching offered me a joyful trip of beautiful knowledge, and helped me build solid foundations for my later study and research. I would like to thank Dr. Zak Kassas for his guidance and supervision during the first year of my Ph.D. I sincerely thank Jun Wang for his kind support during my first year at UCR as a transfer student. I owe a lot to my friends, families, former and current labmates. During six years of my stay in the U.S., they have helped me in all aspects of life and study. due to them, I was able to broaden my knowledge, develop better skills, and maintain good mood during tough times in my Ph.D. Their support, encouragement, and help made me grow, and shaped my identity as a better person.

Materials of this dissertation have been published in the Institute of Electrical and Electronics Engineers (IEEE) Transactions on Smart Grid, IEEE Transactions on Industrial Informatics, IEEE Power and Energy Society Innovative Smart Grid Technologies Conference, IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, and China International Conference on Electrcity Distribution.

Finally, financial support of my work came from National Science Foundation, Department of Energy, California Energy Commission, UCR Dean's Distinguished Fellowship, and UCR TAship.

This dissertation is dedicated to my father Yingchun Gao (高迎春) and my mother Yujie Bian (边毓杰). I could not achieve any success whatsoever in my life without their unconditional support, belief, and love.

ABSTRACT OF THE DISSERTATION

Data-Driven Monitoring and Control of Smart Grid

by

Yuanqi Gao

Doctor of Philosophy, Graduate Program in Electrical Engineering
University of California, Riverside, September 2020
Dr. Nanpeng Yu, Chairperson

As new technologies such as renewable energy resources, distributed generation, and plug-in electric vehicles penetrating the power distribution systems, intelligent monitoring and control become increasingly important for reliable and efficient operation of smart grids, and continuous delivery of high quality electricity to the customers. To accommodate these new technologies, supporting hardware including advanced two-way communication and remotely controllable devices had a significant development over the last decade. However, how to manage, coordinate, and supervise the distribution systems remains a great challenge for the electric utilities.

To address these challenges, we developed four use cases and applications for smart grid monitoring and control from both model-based and data-driven perspectives. Namely, distribution system state estimation (DSSE), distribution system anomaly detection, distribution network reconfiguration (DNR), and Volt-VAR control (VVC). For data-driven algorithms, we derive algorithms that are sample efficient, interpretable, and theoretically justifiable.

Specifically, for distribution system monitoring, we address the low observability and numerical instability issues with the unbalanced DSSE problem with a constrained maximum likelihood (CML) estimator and a sparse subspace Gauss-Newton algorithm. The uncertainty estimate is also derived within the CML framework. To address the physical interpretability of anomaly detection algorithms, we established the connection between the linear approximation of the power flow manifold and a class of modified linear regression models. Algorithmically, we estimate the model parameters and detect anomalies using smart meter data only, without detailed network parameters or confirmed anomaly cases.

For distribution system control, we propose model-based decentralized and data-driven centralized approaches to the DNR problem. For decentralized algorithms, we improve the convergence speed of alternative direction method of multipliers (ADMM) by an approximated Newton's update. For data-driven centralized algorithms, we improve the sample efficiency of existing reinforcement learning (RL) algorithms by a heuristic data augmentation approach, and a principled framework termed batch-constrained soft policy iteration theory. Both approaches improve the existing control policy in the historical datasets, and the batch-constrained RL scales up to networks with hundreds of nodes and switching devices. Lastly, a randomized communication-efficient consensus multi-agent RL (C-MARL) based VVC framework is developed.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Over the last two decades, power distribution systems have seen a tremendous development. New technologies such as distributed energy resources (DER), two-way communication, and remotely controllable devices are being integrated into the legacy distribution system to improve the reliability, efficiency, and sustainability, in a increasingly faster pace. According to the U.S. Energy Information Administration (EIA), in 2019, renewable energy resources such as solar [1], wind, and hydropower, supplied 11.5 quadrillion Btu of energy, equaling 11.4% of total U.S. energy consumption. This is 184% growth compared to 2000 [2]. The number of electric vehicles (EVs) have been growing rapidly over the last decade, and the annual amount of domestic energy required by charging EVs is estimated to be 4.3 TWh in 2018 [3].

While DERs such as energy storage systems, demand responses resources [4, 5, 6], and distributed generation are being integrated, new issues and challenges regarding the management and operation of distribution systems emerges. Renewable energies such as wind and solar are volatile and causes intermittent power supply [7] or even system instability. Distributed generation also causes bidirectional power flow and elevated voltage at the grid edge. The large amount of EV charging demand and energy storage systems [8] raises concerns about the need to upgrade the existing distribution assets, as well as the new infrastructure footprint [9, 10].

To address these challenges, supporting hardware is needed to accommodate these changes in modern distribution system. In recent years, the number of AMI installations is growing significantly. As of 2018, 86.8 million smart meters have been installed in the U.S. [2], supporting more than half of the electricity customers ($\sim$150 million) in the country. Smart meters have the capability to establish two-way communication between the electric utilities and customers, which support smart consumption applications and distributed generation, improve reliability, and help consumers to save money by managing their electricity usage. New control devices such as remotely controllable switches, on load tap changers, and capacitor banks are gradually replacing the legacy control devices. With all these technologies, managing, coordinating, and supervising the working of distribution systems and the assets becomes extremely complex. How to achieve reliability and efficiency remains a great challenge for the electric utilities. This opens up numerous research questions and efforts.

To this end, both industry and academia have taken efforts to improve their methods for distribution system operation, targeting at the ultimate goal-the "smart grid" [11]. For instance, utility companies have been utilizing AMI data to collect outage information and restore service to customers more quickly. By leveraging remotely controllable devices, communications networks, and computational methods, many utility companies have implemented the fault location, isolation, and service restoration (FLISR) procedure to reduce the impact of outages and improve system reliability. In a 2014 study, FLISR was found to reduce the number of customers whose service was interrupted by up to 45%, and reduced the customer minutes of interruption by up to 51% for an outage event [12].

Meanwhile, a growing number of articles and conference proceedings are publishing computational methods for grid management, monitoring, and control. At present, the literature on distribution network monitoring and control can be broadly categorized into two classes: model-based and data-driven methods. Model-based methods use physical models of the distribution system to perform the task. For example, distribution system state estimation treats the nodal voltages as state variables, and constructs the measurement equation based on the distribution network topology and impedances [13]. Distribution system Volt-VAR control calculates the optimal operating schedule for voltage regulating and VAR control devices, based on the network model [14]. Model-based methods are generally theoretically rigorous and reliable. However, they can be difficult to be adopted in practice because of the complexity of the real world distribution systems and the unavailability of the detailed physical models. For example, utility companies typically do not have accurate and reliable primary and secondary feeders' topology and parameter information [15, 16]. To overcome these limitations, big data and machine learning/deep learning are two of the most promising solutions. A purely data-driven method uses only the operational data to achieve monitoring or control, without system parameters. However, this brings about its own challenges. Most data-driven methods are not physically interpretable, and lack guarantee of performance. In addition, unlike model-based algorithms, the design and test of data-driven methods are not quantitative. Accurately reproducing the results is a hard problem. As a result, it is challenging to directly adopt existing artificial intelligence method on power distribution system, which consists of a large number of critical infrastructures. As a result, maintaining a reliable operation is unlikely to be achieved by data-driven method alone.

1.2 Research Objectives and Contributions

Despite the rich theories and algorithms developed in the model-based literature, it is still difficult to adopt these techniques in the real world due to incomplete system parameters. On the other hand, data-driven methods have the potential to be implemented and tested. Nevertheless, they are generally hard to interpret and lack theoretical performance guarantee. In this dissertation work, we answer the question of how to combine the advantages of model-based and data-driven methods to develop algorithms which has the potential to be implemented in the real world, with performance guarantee, and physically interpretable. Specifically, we will develop four major use cases and applications for smart grid data-driven monitoring and control:

- first, we will derive a distribution system state estimation algorithm using the AMI data. In transmission networks, state estimation (SE) is routinely implemented and supports various key applications such as contingency analysis, preventive control, and corrective control. However, SE still does not see a similar level of adoption in distribution system, primarily due to low measurement redundancy, communication delay, unbalanced system operation, and incomplete network parameters. In addition, the existing power system state estimation algorithms are numerically unstable and has poor scalability. We propose to address three of the aforementioned challenges: low measurement redundancy, numerical instability, and scalability. Our approach is to model the physical constraints in the maximum likelihood estimation framework to improve the observability. Then we will develop numerically robust algorithm to solve the resulting constrained maximum likelihood problem.

- second, we will develop an AMI data-driven anomaly detection framework for distribution system secondary circuits. One of the most prevalent form of anomaly is electricity thefts, which causes up to 3.5% of utility companies' annual revenue in the U.S. Traditionally, such activities are detected by laborious field inspections. In the literature, numerous model-based and artificial intelligence theft detection methods have been proposed. However, model-based methods require the unavailable distribution network model information, whereas artificial intelligence methods are physically uninterpretable or require confirmed cases to train. We propose to use the distribution network secondary power flow equation to derive a smart meter power and voltage measurement physical model, then use the AMI historical data to estimate the model parameters to perform anomaly detection. The resulting framework requires minimum distribution system knowledge while being fully data-driven.

- third, we will create model-based distributed, and data-driven centralized algorithms for distribution network reconfiguration (DNR). DNR is an advanced smart grid technology, which works by changing the status of remotely controllable switching devices to optimize certain operational objectives. Currently, nearly all DNR algorithms proposed in the literature are model-based and centralized. However, these algorithms are difficult to be adopted in practice due to uncertain and incomplete network parameters, communication delays, and slow computation speed. We will address these problems separately by proposing distributed and data-driven algorithms. In the model-based distributed control setup, each remotely controllable switch can perform local computation and communicate with their neighbors. The group of agents work collaboratively to find an optimal network configuration without communicating with a central controller. In the data-driven control setup, the network parameter information is assumed to be unavailable. Therefore we propose to use model-free, off-policy reinforcement learning (RL) algorithms to learn a control policy from historical operational dataset. No detailed system model is required to train the algorithms and perform the control task. In addition, the training of the algorithms can be done off-line. Thus the computation speed can be much more faster than model-based approaches.

- fourth, we will develop data-driven distributed algorithm for distribution network Volt-VAR control (VVC). VVC determines the operation schedule of voltage regulating and VAR control devices to lower network losses, improve voltage profile, and reduce voltage violations. In the literature, model-based centralized, model-based distributed, and data-driven centralized have been studied. To further improve the results, we propose to use multi-agent reinforcement learning to derive fully data-driven and distributed algorithm. To address the exploration-exploitation tradeoff, we will develop the algorithm in the maximum entropy RL framework. Moreover, we propose to use stochastic approximation to improve the algorithm's communication efficiency.

The unique contributions of this dissertation are as follows:

- We established a constrained maximum likelihood estimation framework for unbalanced distribution system state estimation problems with low observability, with a numerically stable and sparsity-preserving subspace Gauss-Newton algorithm [13]. The uncertainty estimate was also derived for the algorithm.

- We developed the first physically-inspired data-driven anomaly detection method using customer active power and voltage magnitude measurements alone [17]. While being fully data-driven, the model is physically interpretable and does not need training samples from confirmed cases. We also proposed a novel modified linear regression model to improve the detection accuracy, and proved preliminary properties of the model when subject to normal and abnormal smart meter data.

- We proposed three algorithm frameworks for the distribution network reconfiguration (DNR): model-based alternating direction method of multipliers and approximated Newton (ADMM/Newton) [18], data-driven deep Q-learning (DQL) [19], and a novel batch reinforcement learning algorithm termed batch-constrained soft actor critic (BCSAC) [20]. The ADMM/Newton accelerated the convergence speed of the ADMM by a factor of 5 on a small test distribution feeder. In the DQL based DNR framework, we proposed a novel training data augmentation technique based on spanning forest enumeration and Gaussian process, which was shown to reduce the operational cost. Lastly, in the BCSAC framework, we proved a batch-constrained soft policy iteration theorem, and applied the state-of-the-art actor critic framework to approximate the policy iteration. The algorithm was successfully applied to distribution networks with up to over 100 buses with 3.8×10^{15} feasible configurations, using as few as 8000 training data pairs.

- We proposed a consensus multi-agent reinforcement learning framework for data-driven distributed Volt-VAR control [14]. We expressed the agent consensus constraints in a stochastic approximation format, and proved its equivalence to the original semi-infinite programming form. Based on this result, a communication efficient randomized consensus algorithm was developed. Experimental study on three test distribution networks showed that the consensus multi-agent RL algorithm matches the performance of a single agent benchmark. Also, the algorithm continue to work well under the failures of individual controllers and communication links, with minimum performance degradation.

1.3 Thesis Organization

The remainder of the dissertation is organized as follows: we give the presentations of distribution system monitoring and control in Chapter 2-Chapter 3 and Chapter 4-Chapter 7, respectively. In Chapter 2, we present the constrained maximum likelihood estimation based distribution system state estimation framework and the proposed subspace Gauss-Newton algorithm. In Chapter 3, we detail the problem formulation and the technical methods for the physically-inspired, data-driven distribution system anomaly detection. The distribution network reconfiguration technique will be presented in three chapters: Chapter 4, Chapter 5, and Chapter 6. We discuss, in turn, model-based distributed algorithm based on ADMM/Newton, data-driven deep Q-learning with experience augmentation, and the theory of batch-constrained reinforcement learning and the proposed batch-constrained soft actor critic algorithm, respectively. In Chapter 7, we develop the multi-agent consensus deep reinforcement learning based Volt-VAR control algorithm. Finally, Chapter 8 concludes this dissertation and points out future research directions.

Chapter 2

Unbalanced Distribution System State Estimation

2.1 Introduction

State estimation is one of the most important functions in modern energy management systems (EMS) of interconnected transmission networks [21]. Various key applications such as contingency analysis, preventive control, and corrective control all depend on state estimation solutions. The power system state estimation algorithm was first introduced by Schweppe [22] in 1970 and has since been implemented in almost every EMS of transmission networks around the world. However, the state estimation algorithm has not seen similar level of adoption and application in the distribution management systems (DMS).

An increasing amount of distributed energy resources (DERs) is being integrated into the electric power distribution systems. To proactively manage the large-scale and heterogeneous DERs, the distribution system operators first need a robust distribution system state estimator (DSSE). The installation of supervisory control and data acquisition (SCADA) system at the feeder level [23] and the widespread adoption of advanced metering infrastructure (AMI) have finally made the implementation of a reliable DSSE feasible.

It is not simple to extend the state estimation algorithm developed for the transmission system to the distribution system due to reasons on two levels [24]. Unlike the transmission systems, the distribution systems are typically radial and unbalanced at the network structure level. The unbalanceness is reflected in two ways. First, the electricity loads are unbalanced on three phases. Second, there is a mixture of single-, two-, and three-phase laterals in the distribution feeders. At the sensor level, the DMS typically only has access to low-frequency smart meter readings due to the bottleneck of communication systems. In addition, the level of measurement redundancy in distribution networks is much lower than that of transmission systems. Finally, the smart meter measurements are asynchronous as the built-in real-time clock of smart meters is only periodically synchronized with actual time [25]. The low level of measurement redundancy can cause the system to be unobservable. Asynchronous measurements make it difficult to interpret the state estimation results.

2.2 Prior Work

Several researchers have attempted to address the state estimation problem in the electric power distribution systems. Baran *et al.* pointed out that the voltage and electric load measurements at customer sites can be used for state estimation in the distribution systems [23]. As shown in [26], the access to accurate AMI data can improve the state estimation results in power distribution systems. In [27], the author provided a detailed discussion of three-phase distribution system modeling, measurement functions, and constraints. In terms of the DSSE algorithm design, early works in distribution system state estimation adopted the weighted least squares (WLS) method [28]. A current-based fast decoupled state estimation algorithm was developed for distribution systems [29]. A distribution system state estimation algorithm considering non-synchronized smart meter data was developed by modeling the load variations [30].

2.3 Problem Formulation

2.3.1 Distribution System Overview

Figure 2.1 provides an illustration of a typical electric power distribution system. Labels a, b, and c represent the three phases. n represents the neutral wire. L stands for a lateral and T stands for a transformer. The laterals can be single-phase (L_1 and L_2), two-phase (L_3 and L_4), or three-phase. Residential customers can be served by either a single-phase transformer (T_1, T_2) or a center-tapped transformer (T_3, T_4). Commercial customers are typically served by a three-phase transformer (T_5). There are one or multiple service transformers on each secondary feeder in the distribution network. Each service transformer serves one or multiple buildings which are equipped with smart meters. The smart meters measure the real power consumption (kWh) and voltage magnitudes of each building (for example, $\sim$240V for center-tapped transformers). The SCADA system at

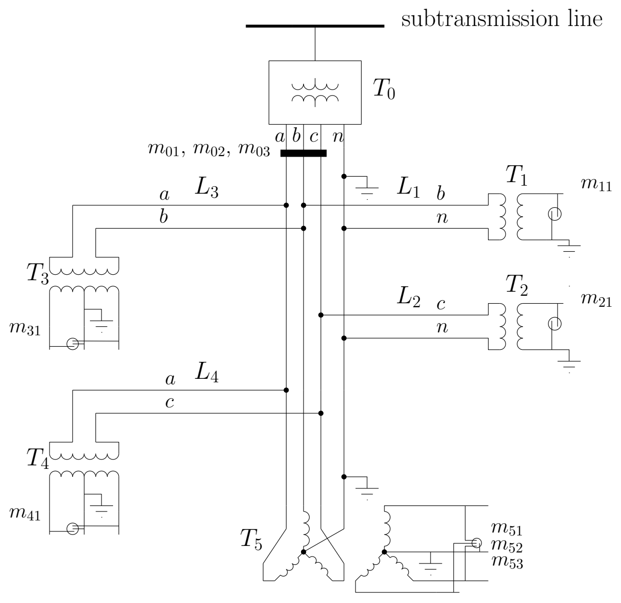


Figure 2.1: Illustration of a distribution system

the distribution feeder level also measures the phase currents, neutral current, line-to-line voltage, and complex power flow on the secondary side of the substation transformer.

2.3.2 General Formulation for State Estimation Problems

The state estimation problem aims at finding the states of the electric power distribution systems, given the network connectivity information and measurement data. The state variable vector $\boldsymbol{x}$ is typically defined as the voltage angles and magnitudes at each node and each phase.

$$\boldsymbol{x} = \left[|V_1^a|, |V_1^b|, \cdots, |V_N^c|, \theta_1^b, \cdots, \theta_N^c\right]^{\mathrm{T}}$$

where $|V_i^p|$ denotes voltage magnitude of bus i with phase p. θ_i^p stands for the voltage angle of bus i with phase p. $\theta_1^a \equiv 0$ is chosen as the reference angle.

The measurement model can be written as follows, where $\boldsymbol{h}(\cdot)$ is a system of nonlinear equations that map the state variables into the measurement space:

$$\boldsymbol{z} = \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{e} \tag{2.1}$$

It is assumed that all of the measurement noise terms $\boldsymbol{e}$ are additive and zero mean Gaussian. In addition to measurement functions associated with measurement devices, there are equality constraints introduced by network topology or circuit elements. They are denoted as:

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0} \tag{2.2}$$

The state estimation result is defined as the solution of (2.1) and (2.2) in the maximum likelihood sense. Under the Gaussianity assumption, the estimation problem is equivalent to the following nonlinear non-convex program.

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & (\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x}))^{\mathrm{T}} \mathbf{R}^{-1} (\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x})) \\ \text{s.t.} \quad & \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0} \end{aligned} \tag{2.3}$$

where $\mathbf{R}$ stands for the measurement error weighting matrix. (2.3) applies to unbalanced three-phase systems. The exact form of the measurement functions and equality constraints are presented in the next two subsections.

2.3.3 Measurement Models

The development of measurement models and equality constraints are based on three-phase power flow equations:

$$P_i^p = |V_i^p| \sum_{k=1}^{N} \sum_{m\in\{a,b,c\}} |V_k^m| (g_{ik}^{pm} \cos(\theta_{ik}^{pm}) + b_{ik}^{pm} \sin(\theta_{ik}^{pm})) \tag{2.4}$$

$$Q_i^p = |V_i^p| \sum_{k=1}^{N} \sum_{m\in\{a,b,c\}} |V_k^m| (g_{ik}^{pm} \sin(\theta_{ik}^{pm}) - b_{ik}^{pm} \cos(\theta_{ik}^{pm})) \tag{2.5}$$

where P_i^p and Q_i^p are the real and reactive net injected power at bus i with phase p. V_i^p is the voltage at bus i with phase p. $\theta_{ik}^{pm} = \theta_i^p - \theta_k^m$ is the voltage angle differences between bus i with phase p and bus k with phase m. The Y-bus matrix is given by:

$$\mathbf{Y} = [\mathbf{Y}_{ik}] \quad \mathbf{Y}_{ik} = \left[y_{ik}^{pm}\right] = \left[g_{ik}^{pm} + jb_{ik}^{pm}\right]$$

where $i, k \in$ set of buses and $p, m \in \{a, b, c\}$. Each $[\mathbf{Y}_{ik}]$ is a 3×3 block with off-diagonal elements representing mutual magnetic coupling between phases.

There are two types of measurement models in unbalanced distribution systems: the single-phase and two-phase measurements. The single-phase measurements include single-phase voltage magnitudes and single-phase electricity loads. The two-phase measurements include line-to-line voltage magnitudes and two-phase electricity loads. We assume the following measurements are available through SCADA and AMI:

- At the distribution substation, phase-to-neutral voltage magnitude, magnitude of current injection, and three-phase complex power injection measurements are available.

- At every distribution center-tapped transformer, two-phase voltage magnitude and aggregated real two-phase power injection measurements are available.

- At every single-phase or three-phase distribution transformer, phase-to-neutral voltage magnitudes and real power injection measurements are available.

Finally, we assume that the center-tapped transformers and the single-phase transformers are approximately ideal, and that the series impedance and shunt admittance of the lines under the transformers' secondaries are negligible. It follows that the three phase power at the transformer primary is (approximately) equal to the sum of customer power measurements, and that the voltage measurements for all customers under the same secondary are (approximately) equal. We are now ready to write down the measurement equations. The single-phase measurement equations (one superscript p) and two-phase measurement equations (double superscript pm) are listed below:

$$|V_i^p|_{meas} = |V_i^p| + e_{|V_i^p|} \tag{2.6}$$

$$P_{i\ meas}^p = P_i^p + e_{P_i^p} \tag{2.7}$$

$$Q_{1meas}^p = Q_1^p + e_{Q_1^p} \tag{2.8}$$

$$|I_1^p|_{meas} = \left\| \begin{bmatrix} \mathrm{Re}(I_1^p) & \mathrm{Im}(I_1^p) \end{bmatrix}^{\mathrm{T}} \right\|_2 + e_{|I_1^p|}$$

$$\mathrm{Re}(I_1^p) = \sum_{k=1}^{N} \sum_{m \in \{a,b,c\}} |V_k^m| (g_{1k}^{pm} \cos(\theta_k^m) - b_{1k}^{pm} \sin(\theta_k^m)) \tag{2.9}$$

$$\mathrm{Im}(I_1^p) = \sum_{k=1}^{N} \sum_{m \in \{a,b,c\}} |V_k^m| (g_{1k}^{pm} \sin(\theta_k^m) + b_{1k}^{pm} \cos(\theta_k^m)) \tag{2.10}$$

$$|V_i^{pm}|_{meas} = \sqrt{|V_i^p|^2 + |V_i^m|^2 - 2|V_i^p||V_i^m| \cos(\theta_i^p - \theta_i^m)} + e_{|V_i^{pm}|} \tag{2.11}$$

$$
\begin{aligned}
P_{i\ \ meas}^{pm} =& |V_i^p| \sum_{k=1}^{N} \sum_{n \in \{a,b,c\}} |V_k^n| (g_{ik}^{pn} \cos(\theta_{ik}^{pn}) + b_{ik}^{pn} \sin(\theta_{ik}^{pn})) \\
& - |V_i^m| \sum_{k=1}^{N} \sum_{n \in \{a,b,c\}} |V_k^n| (g_{ik}^{pn} \cos(\theta_{ik}^{mn}) + b_{ik}^{pn} \sin(\theta_{ik}^{mn})) + e_{P_i^{pm}} \qquad (2.12)
\end{aligned}
$$

where $|V_i^{pm}| = |V_i^p - V_i^m|$ denotes line-to-line voltage magnitudes and $P_i^{pm} = \text{Re}((V_i^p - V_i^m) I_i^{p*})$ stands for two-phase real power injections.

2.3.4 Equality Constraints

Two types of equality constraints are modeled in the DSSE problem. The first type of equality constraints are associated with buses with neither load nor generation. The net injected power for these buses must be zero:

$$P_{\text{tap},i}^{p} = 0 \qquad (2.13)$$

$$Q_{\text{tap},i}^{p} = 0 \qquad (2.14)$$

The second type of equality constraints are associated with center-tapped transformers, at which the sum of currents flowing through the two phases are zero:

$$\text{Re}(I_i^p) + \text{Re}(I_i^m) = 0 \qquad (2.15)$$

$$\text{Im}(I_i^p) + \text{Im}(I_i^m) = 0 \qquad (2.16)$$

Note that a distribution system node may have a combination of these constraints and measurements. For example, in Figure 2.2, node i has two measurement functions, two zero injection constraints, and two equality constraints associated with the center-tapped transformer. The measurement functions are real power ${P_i^{bc}}_{meas}$ and line-to-line voltage magnitude $|V_i^{bc}|_{meas}$. The zero injection constraints are $P_i^a = 0$ and $Q_i^a = 0$. The constraints associated with the center tapped transformer are $\mathrm{Re}(I_i^b) + \mathrm{Re}(I_i^c) = 0$ and $\mathrm{Im}(I_i^b) + \mathrm{Im}(I_i^c) = 0$.

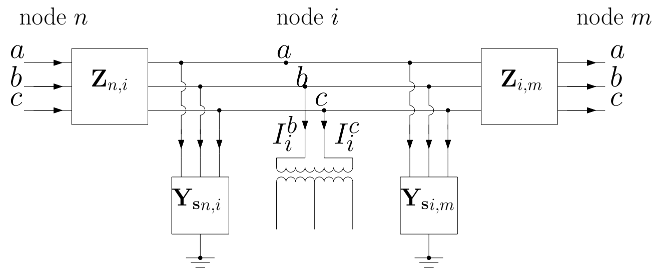


Figure 2.2: Constraints and measurements at node i

In the next section, we discuss the proposed subspace Gauss-Newton algorithm to solve the formulated problem.

2.4 The Subspace Gauss-Newton Algorithm

2.4.1 Algorithm Derivation

To solve (2.3), we invoke the Lagrange multiplier theorem [31], which states that the gradient of the Lagrangian must vanish at local minima or maxima.

$$\nabla_{\boldsymbol{x}}\mathcal{L} = -2\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\Delta\boldsymbol{z} - \mathbf{F}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{0} \tag{2.17}$$

$$\nabla_{\boldsymbol{\lambda}}\mathcal{L} = -\boldsymbol{f}(\hat{\boldsymbol{x}}) = \mathbf{0} \tag{2.18}$$

where $\mathbf{H} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}}$, $\mathbf{F} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}}$, and $\Delta \boldsymbol{z} = \boldsymbol{z} - \boldsymbol{h}(\hat{\boldsymbol{x}})$. $\hat{\boldsymbol{x}}$ denotes a local minimum or maximum. (2.17) and (2.18) are necessary but not sufficient conditions for a local optimal solution. Nevertheless, the local optimality can easily be checked by examing the positive definiteness of the Hessian matrix.

The nonlinear system of equations (2.17)-(2.18) can be solved using the Newton's method: starting from some initial guess $\boldsymbol{x}^0$ and $\boldsymbol{\lambda}^0$, then proform the iteration $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \Delta \boldsymbol{x}$, $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \Delta \boldsymbol{\lambda}$, where the increments can be found by solving the following equation.

$$\begin{bmatrix} \nabla_{\boldsymbol{x}\boldsymbol{x}}\mathcal{L} & \nabla_{\boldsymbol{x}\boldsymbol{\lambda}}\mathcal{L} \\ \nabla_{\boldsymbol{\lambda}\boldsymbol{x}}\mathcal{L} & \nabla_{\boldsymbol{\lambda}\boldsymbol{\lambda}}\mathcal{L} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{x} \\ \Delta \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \nabla_{\boldsymbol{x}}\mathcal{L} \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L} \end{bmatrix} \tag{2.19}$$

where the first and second derivatives are evaluated at the current iterate $\boldsymbol{x}^k$, $\boldsymbol{\lambda}^k$. However, computing the term $\nabla_{\boldsymbol{x}\boldsymbol{x}}\mathcal{L}$ requires the derivative of $\mathbf{H}$ and $\mathbf{F}$, which are not strateforward to write down. Following the sequential quadratic programming (SQP) approximation [32], we may approximate the increment $\Delta \boldsymbol{x}$ by the solution of the following linear constrained quadratic programming:

$$\begin{aligned} \min_{\Delta \boldsymbol{x}} \quad & (\Delta \boldsymbol{z} - \mathbf{H}\Delta \boldsymbol{x})^{\mathrm{T}} \mathbf{R}^{-1} (\Delta \boldsymbol{z} - \mathbf{H}\Delta \boldsymbol{x}) \\ \text{s.t.} \quad & \mathbf{F}\Delta \boldsymbol{x} = \Delta \boldsymbol{f} \end{aligned} \tag{2.20}$$

where $\Delta \boldsymbol{x} = \boldsymbol{x}^{k+1} - \boldsymbol{x}^k$ and $\Delta \boldsymbol{f} = -\boldsymbol{f}(\boldsymbol{x}^k)$. Jacobian matrices $\mathbf{H}$ and $\mathbf{F}$ are evaluated at current iterate $\boldsymbol{x}^k$. The necessary optimality condition of formulation (2.20) differs from (2.19) in two respects: first, the constraints are linearized first rather than appearing in the second order derivative $\nabla_{\boldsymbol{x}\boldsymbol{x}}\mathcal{L}$; second, the Hessian matrix of the objective function is approximated by:

$$\nabla_{\boldsymbol{x}\boldsymbol{x}}(\boldsymbol{z}-\boldsymbol{h}(\boldsymbol{x}))^{\mathrm{T}}\mathbf{R}^{-1}(\boldsymbol{z}-\boldsymbol{h}(\boldsymbol{x})) \approx 2\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H} \tag{2.21}$$

Approximation (2.21) is reasonable when the measurement residuals and nonlinearity of the measurement models are small.

Problem (2.20) can be solved in closed form. Also, the solution only involves calculation of first order derivatives. However, its Newton-KKT matrix is indefinite and can be ill-conditioned. One commonly used algorithm to improve DSSE numerical stability is the Hachtel's augmented matrix method [33]. But this method requires specifying a tuning parameter, which might be difficult to tune in practice. In the following, we present an efficient algorithm to find the solution to (2.20), which is numerically stable and preserves the sparsity of the problem, with no tunning parameters.

Our proposed method is to constrain the search of solution within the affine space defined by the constraint equations in (2.20). A similar approach has appeared in other textbook [34]. The affine space consists of all vectors which is a summation of a particular solution $\boldsymbol{x}^p$ and a homogeneous solutions $\boldsymbol{x}^h$:

$$\Delta\boldsymbol{x} = \boldsymbol{x}^p + \boldsymbol{x}^h \tag{2.22}$$

One possible particular solution can be $\boldsymbol{x}^p = \mathbf{F}^{\mathrm{T}}(\mathbf{F}\mathbf{F}^{\mathrm{T}})^{-1}\Delta\boldsymbol{f}$. This is also the minimum 2-norm solution to the constraint equations. The set of homogeneous solutions is the nullspace of constraints Jacobian, which can be expressed as arbitrary linear combinations of nullspace basis vectors $\mathbf{V}$:

$$\boldsymbol{x}^h = \mathbf{V}\boldsymbol{\beta} \qquad \forall \boldsymbol{\beta} \tag{2.23}$$

Multiple matrix factorization methods can find such bases. For example the singular value decomposition which gives an orthonormal set of basis vectors. However, It is preferable to have a sparse nullspace basis to improve the computation speed in later steps. In this work, we adopt the ABS sparse nullspace algorithm developed in [35]. An implementation of this algorithm is available at [36].

If we substitute $\Delta\boldsymbol{x} = \boldsymbol{x}^p + \boldsymbol{x}^h$ into the objective, then minimize it with respect to $\boldsymbol{\beta}$, we get the following necessary optimality condition:

$$\mathbf{V}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{V}\boldsymbol{\beta} = \mathbf{V}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}(\Delta\boldsymbol{z} - \mathbf{H}\boldsymbol{x}^p) \tag{2.24}$$

In DSSE problem, the matrix on the left hand side (LHS) of (2.24) is positive definite. This suggests a more numerically stable solution compared to directly solving the Newton-KKT system. We solve (2.24) by the QR factorization, which is considered as numerically stable:

$$\mathbf{R}^{-\frac{1}{2}}\mathbf{H}\mathbf{V} = \mathbf{Q}\mathbf{U} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{bmatrix} \begin{bmatrix} \mathbf{U}_1 \\ \mathbf{0} \end{bmatrix} \tag{2.25}$$

In our case study, we found that the product $\mathbf{HV}$ is still sparse in a sparsely connected distribution network. Hence, we use the Givens rotation to compute this QR factorization, which is computationally efficient for large and sparse matrix [37]. After the QR factorization, (2.24) is reduced to:

$$\mathbf{U}_1\boldsymbol{\beta} = \mathbf{Q}_1^{\mathrm{T}}\mathbf{R}^{-\frac{1}{2}}(\Delta\boldsymbol{z} - \mathbf{H}\boldsymbol{x}^p) \tag{2.26}$$

where $\mathbf{U}_1$ is an upper triangular matrix. Note that in (2.26), forming the gain matrix $\mathbf{G}' = \mathbf{V}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{V}$ was avoided. We solve for $\boldsymbol{\beta}$ in (2.26), obtain the increment $\Delta\boldsymbol{x}$ using (2.24), then the overall iteration proceeds. This completes our algorithm derivation. In practical implementation, we precondition the objective function by a number c:

$$\min\, cJ = c(\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x}))^{\mathrm{T}}\mathbf{R}^{-1}(\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x})) \tag{2.27}$$

The value c can be set as the average of the diagonal elements of $\mathbf{R}$. For example, when all elements of the $\mathbf{R}$ diagonal are the same, the objective function becomes $cJ = (\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x}))^{\mathrm{T}}(\boldsymbol{z} - \boldsymbol{h}(\boldsymbol{x}))$. This further improves the algorithm's numerical stability.

2.4.2 Uncertainty Estimation with the Subspace Gauss-Newton

The converged solution of the subspace Gauss-Newton iteration is the constrained maximum likelihood estimate (2.3). In the context of distribution system state estimation, we are interested in how uncertain this estimate is. The uncertainty assessment provides additional insights into the quality of the estimate and the operation of the grid. In this subsection, we derive an uncertainty estimate after the subspace Gauss-Newton algorithm converges.

First, we note that at the converged estimate, the equality constraints must be strictly satisfied. Therefore $\Delta\boldsymbol{x}$ must be in the nullspace of $\mathbf{F}$ in the last iterate (that is, $\Delta\boldsymbol{x} = \mathbf{V}\boldsymbol{\beta}$). Therefore the update of $\boldsymbol{\beta}$ in (2.24) becomes

$$\boldsymbol{\beta} = (\mathbf{V}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{V})^{-1}\mathbf{V}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\Delta\boldsymbol{z} \tag{2.28}$$

Thus the covariance of $\boldsymbol{\beta}$ is given by:

$$\mathrm{cov}_{\boldsymbol{\beta}} = E(\boldsymbol{\beta}\boldsymbol{\beta}^{\mathrm{T}}) = (\mathbf{V}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{V})^{-1} \tag{2.29}$$

where the linearity of expectation operator and the definition $E(\Delta\boldsymbol{z}\Delta\boldsymbol{z}^{\mathrm{T}}) = \mathbf{R}$ are used. Let the true state be $\boldsymbol{x}_{\mathrm{true}}$, and the maximum likelihood estimate be $\boldsymbol{x}^*$, the error $\tilde{\boldsymbol{x}} = \boldsymbol{x}_{\mathrm{true}} - \boldsymbol{x}^*$ can be approximated as:

$$\tilde{\boldsymbol{x}} \approx \mathbf{V}\boldsymbol{\beta}$$

Therefore the transformation of covariance we find the covariance of estimation error:

$$\begin{aligned} \mathrm{cov}_{\tilde{\boldsymbol{x}}} &\approx \mathbf{V}\mathrm{cov}_{\boldsymbol{\beta}}\mathbf{V}^{\mathrm{T}} && (2.30)\\ &= \mathbf{V}(\mathbf{V}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{V})^{-1}\mathbf{V}^{\mathrm{T}} && (2.31) \end{aligned}$$

The covariance matrix (2.31) is singular because the constraints make the estimation errors correlated.

2.5 Numerical Studies

The proposed subspace Gauss-Newton based DSSE algorithm is implemented on a modified IEEE 13-bus test feeder. Both state estimation results and numerical properties of the proposed method are presented in this section.

The IEEE 13-bus test feeder [38] is modified to include center-tapper transformers as follows.

- Node 650 is considered as the secondary side of the substation transformer. The primary side is assumed to be connected to an infinite bus. The voltage regulator is ignored.
- The circuit breaker between node 671 and 692 is removed.
- The uniformly distributed loads are removed from the feeder.
- Node 646 is assumed to be connected to a center-tapped transformer across phases b and c.

The simulations are set up as follows. First, load flow calculations are carried out using Newton-Raphson method. Noise corrupted measurements are then generated using the measurement models described in Section 2.3. Second, the measurements are fed into the DSSE algorithm. At last, the differences between the load flow results and the DSSE results are recorded. The simulations are repeated M times. The normalized root-mean-square errors of DSSE are computed.

$$|\tilde{V_i^p}| = \sqrt{\frac{\sum_{k=1}^{M}(\frac{|V_i^p|-|\hat{V_i^p}|_k}{|V_i^p|})^2}{M}} \quad \tilde{\theta_i^p} = \sqrt{\frac{\sum_{k=1}^{M}(\frac{\theta_i^p-\hat{\theta_i^p}_k}{2\pi})^2}{M}} \tag{2.32}$$

where $(|\hat{V}|\ \hat{\theta})$ denote results of state estimation. The stopping criterion for the load flow calculation and the state estimation are set to be the same as follows: $\max_i |\Delta x_i| < 0.00001$ p.u..The measurement noise covariance matrix was set to be $\mathbf{R} = \mathrm{diag}\left[\sigma^2, \sigma^2, \cdots, \sigma^2\right]$. With $\sigma = 0.01$ p.u., the results of DSSE are shown in Table 2.1. The simulation results show that under 0.01 p.u. measurement noise, most of the voltage magnitude estimation errors are less than 1%, and all of the voltage angle estimation errors are less than $1\% \cdot 2\pi$. Additional simulations are conducted to analyze the impact of measurement noise on state

Table 2.1: RMSE of $|\hat{V}|$ and $\hat{\theta}$ (per unit)

	$\lvert\tilde{V}\rvert$			$\tilde{\theta}$		
Node \ Ph	a	b	c	a	b	c
ref	0.0092	0.0099	0.0089	0.0	0.0036	0.0056
650	0.0066	0.0074	0.0080	0.0020	0.0038	0.0029
646	-	0.0091	0.0061	-	0.0045	0.0031
645	-	0.0094	0.0061	-	0.0046	0.0031
632	0.0072	0.0058	0.0062	0.0021	0.0042	0.0031
633	0.0088	0.0069	0.0073	0.0022	0.0043	0.0033
634	0.0102	0.0088	0.0084	0.0022	0.0043	0.0033
611	-	-	0.0103	-	-	0.0044
684	0.0068	-	0.0067	0.0027	-	0.0038
671	0.0082	0.0073	0.0083	0.0025	0.0043	0.0034
675	0.0090	0.0087	0.0076	0.0032	0.0049	0.0040
652	0.0091	-	-	0.0039	-	-
680	0.0082	0.0073	0.0083	0.0025	0.0043	0.0034

estimation errors. The standard deviation of the measurement noise σ is increased systematically from 0.0001% to 1.5%. Under each setting of σ, Monte Carlo simulations are conducted. The state estimation errors under each measurement noise setting are reported in Figure 2.3. In the figure, each curve represents the change of estimation error in response to measurement noise for one node and one phase. As expected, the measurement errors increase as the measurement noise level increases. In this experiment, the numerical stability and computational efficiency of the proposed algorithm is compared to that of the Hachtel's augmented matrix method. In addition, a separate tunable parameter that determines the numerical stability is no longer required. The numerical stability is evaluated

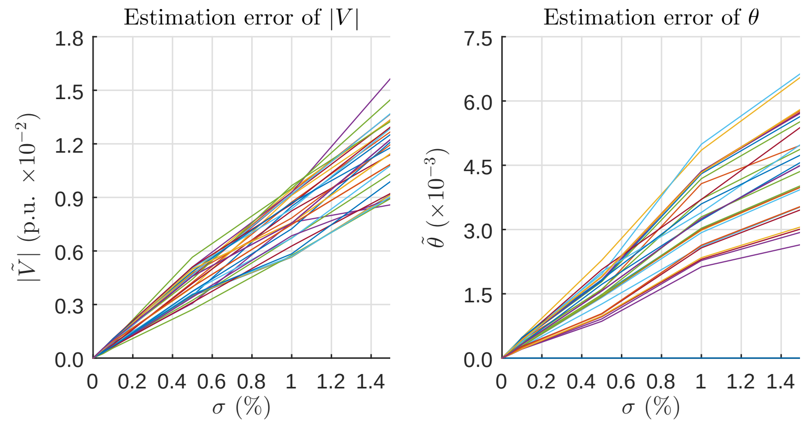


Figure 2.3: Estimation error v.s. measurement noise

by measuring the condition number of the coefficient matrix which is used in solving the linear equation (2.26). The computational efficiency of the proposed algorithm is evaluated by measuring the number of nonzero elements (**nnz**) in the coefficient matrix. To demonstrate the effect of using different sparse nullspace basis, we also provide comparisons to the nullspace obtained with singluar value decomposition (SVD). The **nnz** and condition

Table 2.2: **nnz** of coefficient matrices

	Lagrange	Hachtel's	SGN (SVD)	SGN
nnz	2081	1526	561	496

number of coefficient matrices are reported in Table 2.2 and Table 2.3. As shown in Table 2.2 and Table 2.3, the coefficient matrix in the proposed algorithm has a smaller number of

Table 2.3: Condition number of coefficient matrices ($\times 10^3$), $\sigma = 0.005$ p.u.

Method \ Iteration		$k = 0$	$k = 1$	$k = 2$	$k = 3$
Lagrange	-	2819.008	3038.544	2930.839	2805.181
Hachtel's	$\alpha = 1$	18.628	17.616	18.597	18.652
	$\alpha = 0.1$	3.500	3.398	3.135	3.113
	$\alpha = 0.05$	6.998	6.469	5.637	6.186
SGN (SVD)	-	25.835	2.553	2.833	2.912
SGN	-	0.938	0.984	1.038	0.964

nonzero elements and lower condition number. This demonstrates that the proposed algorithm is more computational efficient and numerically stable than the Hachtel's augmented matrix method.

Finally, we verify the validity of the error covariance matrix given by (2.31). The true covariance matrix, is defined in (2.33) and can be approximated by the Monte Carlo estimation (2.34):

$$\text{cov}_{\tilde{\boldsymbol{x}}} = E(\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^{\mathrm{T}}) \tag{2.33}$$

$$\approx \frac{1}{N}\sum_{i}^{N} \tilde{\boldsymbol{x}}_i\tilde{\boldsymbol{x}}_i^{\mathrm{T}} \tag{2.34}$$

To assess how well (2.31) approximate (2.33), we compare it with (2.34). For $\sigma = 0.01$ p.u. and $\sigma = 0.001$ p.u., diagonal elements of each matrix were plotted in Figure 2.4. We found that the two covariance matrices matches very well. This justifies (2.31) as an estimation error covariance in the sense that had we collect the measurement N times under the same underlying state, the diagonal terms in (2.31) is the variance of our estimated state.

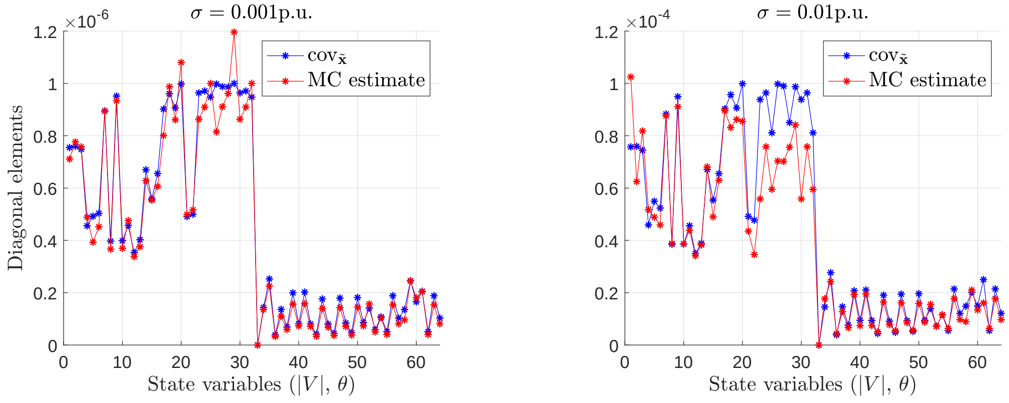


Figure 2.4: Approximation of covariances

2.6 Summary

In this chapter, a distribution system state estimation problem is formulated considering unbalanced single-phase and two-phase measurements. Constraints associated with zero injections and center-tapped transformers are incorporated into the problem formulation. An subspace Gauss-Newton based algorithm is developed to solve the DSSE problem. Also derived is an uncertainty estimate of the solution which is useful to assess the quality of the DSSE results. The proposed algorithm yields better numerical stability and computational efficiency than existing methods. The simulation results on a modified IEEE test feeder validated the accuracy, numerical stability, and computational efficiency of the proposed algorithm.

Chapter 3

Physically Inspired Data-driven Electricity Theft Detection

3.1 Introduction

Electricity theft refers to the practice of manipulating one's electricity data to reduce his or her electricity bill [39]. Electricity theft not only leads to significant revenue losses, but also creates the risk of fires and fatal electrical shocks. In the United States, utilities lose between 0.5% and 3.5% of their annual revenue to theft [40]. In some developing countries, the revenue loss from electricity theft is even larger [41] [42].

In the past, utilities have fought electricity theft by sending field operation groups to conduct physical inspections of electrical equipment based on suspicious activity reported by the public. However, the recent rapid penetration of advanced metering infrastructure makes it possible to detect electricity theft by analyzing the information gathered from smart meters. In this dissertation work, we develop a physically inspired data driven model to detect electricity theft with smart meter data. The model requires the network operator to have the customer active power consumption and voltage magnitude data, as well as the customer to transformer association map. In practice, these data are readily available to the electric utility companies, hence a speedy and widespread adoption of the proposed model is feasible.

In this chapter, we will detail the development process of the detection algorithm in three steps. First, we analyze the physical relationship between smart meter voltage readings and electricity consumptions of customers. Then we propose a modified linear regression model to capture this physical relationship on the Kron-reduced distribution secondaries. Third, we create a set of training/testing dataset pairs on a rolling window basis to perform training and predicting of the modified linear regression model. The prediction results will be compared to the actual measurements to compute an anomaly score. Finally, the anomaly scores for all customers will be ranked to indicate the level of abnormalities. We provide theoretical and empirical analysis for the proposed detection model. We prove that electricity theft on a distribution secondary will lead to negative and positive residuals from the regression for dishonest and honest customers respectively. Experimental results with real-world smart meter data confirms this finding and shows that the model is effective in identifying electricty theft cases.

The rest of this chapter is organized as follows. Section 3.2 reviews the prior work. Section 3.3 presents the modeling of distribution secondaries considering smart meter measurements. Section 3.4 details the technical methods for the proposed modified linear model and electricity theft detection process. Section 3.5 presents the experimental results. Finally, Section 3.6 provides the summary of this chapter.

3.2 Prior Work

We focus our review on the existing work which uses smart meter or other collected customer data to detect electricity theft. They can be categorized into three groups based on the type of data.

Methods in the first group assume that smart meter data is not available. Instead, they leverage ancillary information such as biannual electricity consumption and credit scores [43] [44] [45], which can be used as features in supervised machine learning. Many supervised methods have been tested in literature. Examples include support vector machines (SVM) [45], optimum-path forests [44], and artificial neural networks [46]. Supervised learning only work if verified cases of electricity theft are available. If this is not the case, then unsupervised methods, which do not use electricity theft labels, must be used. Examples include fuzzy c-means clustering [43] and optimum-path forests clustering [47].

Methods in the second group assume that granular power consumption data is available. For example, reference [48] analyzed consumption profiles through a self-organizing map (SOM). Reference [49] proposed an entropy-based method to analyze the distribution of differenced consumption data. Reference [50] used an extreme learning machine to detect anomalies in electricity usage. Reference [51] combined a decision tree and an SVM to predict smart meter abnormalities. Reference [52] used a convolutional neural network trained on such data to perform detection.

Some studies in the second group assume the existence of a “central observer” [53] [54] [55] [56]. This observer measures the aggregated consumption from a group of customers. In particular, such central observers can be placed on the distribution transformers. Meter malfunction or tampering can thus be identified using linear regression.

Methods in the third group assume that network topology and parameter information are available. Under this assumption, state estimation based approaches become feasible. Early work on this direction [57] perform distribution system state estimation based on estimated load. The non-technical losses are then detected by comparing the results with the billed consumption. The approaches proposed in [58] [59] [60] first perform three phase state estimation procedures on the network. They then analyze variances [58] [60] or apply heuristic methods [59] to locate meter defects or tampering. Recently, these time-snapshot based methods have been improved by adopting phasor measurement unit (PMU) data [61]. Another method in this group formulates anomaly detection as an optimization problem [62]. The method finds a sparse power mismatch matrix whose non-zero elements correspond to the bypassed power from dishonest customers.

Perhaps, the most directly relevant work is [63], in which the authors proposed analyzing sample covariance matrices of smart meter measurement error statistics, voltage magnitude and active power data to detect electricity theft . Compared to [63], our proposed work does not need to make any assumption about the smart meter measurement error distributions. In addition, we provide a theoretical justification based on physical network model for using real power consumption and voltage magnitude measurement to detect electricity theft.

The existing literature on data-driven electricity theft detection has three limitations. First, it is not realistic to assume that the transformer power measurements, reliable topology documentation [64, 65], and network parameter information are available to electric utilities. Network parameter information is typically known only up to the type of conductors. Good parameter estimation methods for single-phase models [66] and balanced three-phase models [67, 68] do exist. But methods for estimating unbalanced three-phase network parameters are still in their infancy. Hence all techniques in the third group are usually infeasible. Furthermore, Pole mounted distribution transformers are generally not equipped with operational monitoring devices [69] in Europe and America. Thus the "central observer" techniques in the second group are infeasible as well. Second, residential customer loads are irregular and are dependent on many external factors [70]. Analyzing such profiles alone produces very limited interpretability and justification of the results. Worse yet, they might not distinguish between electricity theft and non-malicious customer activities. Many of these methods would detect the installation of a new electric device as theft. This diminishes the usability of methods in the second group. Finally, supervised approaches in the first and second groups need theft samples. But obtaining labeled datasets in this case is usually a hard task [47]. As a result, the number of labeled (inspected) customers is very small compared to the total number of customers.

3.3 Problem Analysis

In this section, we analyze the physical relationship between smart meter voltage magnitude and real power consumption measurements. Such analysis is particularly valuable for detecting anomalies since voltage measurements are not easy to tamper and modify in a sensible way, therefore can be considered as informative to indicate abnormalities in the power measurements. In addition, this analysis reveals the way how different customers' voltage measurements can contribute to the estimate of each customer's power consumption profiles. All equations/derivations in this section are to provide justifications for the algorithms presented in Section 3.4. They are not used for actual computation.

3.3.1 Linearization of Secondary Circuit Power Flow Equations

In North America, distribution secondaries serving residential customers typically have a 120/240V three-wire two-phase configuration. The two phases have voltages with an angle difference of 180 degrees. A sample distribution secondary with n_c customers is shown in Figure 3.1.

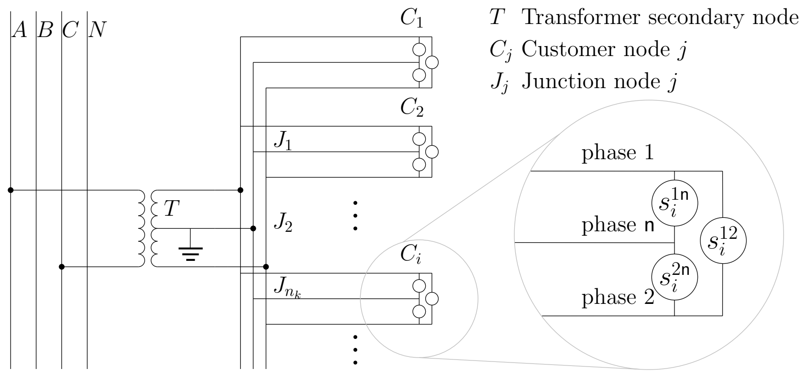


Figure 3.1: Triplex line secondary circuit

We will derive a linearized power flow model for this configuration. Following [71], the idea is to find a tangent plane to the power flow manifold centered at a suitable point, which can be the modified flat voltage solution. We construct this solution as follows.

First, assume that shunt admittances are zero. Denote the line to ground voltage phasor and current phasor at node m as $\bar{\mathbf{u}}_m$ and $\bar{\mathbf{i}}_m$, and the real and reactive power injections as $\bar{\mathbf{p}}_m$ and $\bar{\mathbf{q}}_m$. Then the modified flat voltage solution has, at each node m, the following set of values: $\bar{\mathbf{u}}_m = [1, 1 \cdot e^{-\pi j}]^\mathsf{T} = [1, -1]^\mathsf{T}$, $\bar{\mathbf{i}}_m = \mathbf{0}$, $\bar{\mathbf{p}}_m + j\bar{\mathbf{q}}_m = \mathbf{0}$.

Denote v_m^p as the deviation of the line-to-ground voltage from the flat voltage solution $\bar{v}_m^p$ at node m and phase p. Let $\mathbf{v}^1$ and $\mathbf{v}^2$ be the reindexed voltage vectors:

$$\left[\mathbf{v}^{1\mathsf{T}}, \mathbf{v}^{2\mathsf{T}}\right] = \left[v_1^1, v_2^1, ..., v_n^1, v_1^2, v_2^2, ..., v_n^2\right]$$

where n is the number of non-zero injection nodes in a secondary circuit. The voltage angles $\boldsymbol{\theta}^1$ $\boldsymbol{\theta}^2$, real power injections $\mathbf{p}^1$ $\mathbf{p}^2$, and reactive power injections $\mathbf{q}^1$ $\mathbf{q}^2$ are defined in a similar manner. The linearization around the modified flat voltage solution yields:

$$\begin{bmatrix} \mathbf{p}^1 \\ \mathbf{p}^2 \\ \mathbf{q}^1 \\ \mathbf{q}^2 \end{bmatrix} = \begin{bmatrix} \mathbf{G}^{11} & -\mathbf{G}^{12} & -\mathbf{B}^{11} & \mathbf{B}^{12} \\ -\mathbf{G}^{21} & \mathbf{G}^{22} & \mathbf{B}^{21} & -\mathbf{B}^{22} \\ -\mathbf{B}^{11} & \mathbf{B}^{12} & -\mathbf{G}^{11} & \mathbf{G}^{12} \\ \mathbf{B}^{21} & -\mathbf{B}^{22} & \mathbf{G}^{21} & -\mathbf{G}^{22} \end{bmatrix} \begin{bmatrix} \mathbf{v}^1 \\ \mathbf{v}^2 \\ \boldsymbol{\theta}^1 \\ \boldsymbol{\theta}^2 \end{bmatrix} \tag{3.1}$$

where $\mathbf{G}^{ij}$ and $\mathbf{B}^{ij}$ are the real and imaginary blocks of the reindexed nodal admittance matrix $\mathbf{Y}^\mathbf{r}$:

$$\mathbf{Y}^\mathbf{r} = \begin{bmatrix} \mathbf{G}^{11} & \mathbf{G}^{12} \\ \mathbf{G}^{21} & \mathbf{G}^{22} \end{bmatrix} + j \begin{bmatrix} \mathbf{B}^{11} & \mathbf{B}^{12} \\ \mathbf{B}^{21} & \mathbf{B}^{22} \end{bmatrix} \tag{3.2}$$

which is a permutation of the well-known bus admittance matrix $\mathbf{Y}$. Explicitly, $\mathbf{Y}^\mathbf{r}$ is obtained by taking every odd indexed row and column of $\mathbf{Y}$ and relocating them to the bottom-most and right-most positions respectively. In the following, we refer to (3.1) as $\mathbf{y}^{\mathbf{r}} = \mathbf{L}^{\mathbf{r}}\mathbf{x}^{\mathbf{r}}$. The derivation of (3.1) is provided in Appendix A.1.

In (3.1), $\mathbf{p}^1$, $\mathbf{p}^2$, $\mathbf{q}^1$ and $\mathbf{q}^2$ are single-phase net injections. But electric loads at the customers' site can be single-phase or two-phase as shown in Figure 3.1. We can get the single-phase net injections from the electric loads by using (3.3):

$$\begin{bmatrix} s_i^1 \\ s_i^2 \end{bmatrix} = \begin{bmatrix} \frac{u_i^{1\mathsf{n}}}{u_i^{1\mathsf{n}}+u_i^{2\mathsf{n}}} & 0 & 1 \\ \frac{-u_i^{2\mathsf{n}}}{u_i^{1\mathsf{n}}+u_i^{2\mathsf{n}}} & 1 & 0 \end{bmatrix} \begin{bmatrix} s_i^{12} \\ s_i^{2\mathsf{n}} \\ s_i^{1\mathsf{n}} \end{bmatrix} \tag{3.3}$$

The derivation of (3.3) is in Appendix A.2. This can be simplified near the flat voltage operating condition to:

$$\begin{bmatrix} p_i^1 \\ p_i^2 \end{bmatrix} + j \begin{bmatrix} q_i^1 \\ q_i^2 \end{bmatrix} = \begin{bmatrix} s_i^1 \\ s_i^2 \end{bmatrix} \approx \begin{bmatrix} \frac{1}{2} & 0 & 1 \\ \frac{1}{2} & 1 & 0 \end{bmatrix} \begin{bmatrix} s_i^{12} \\ s_i^{2\mathsf{n}} \\ s_i^{1\mathsf{n}} \end{bmatrix} \tag{3.4}$$

3.3.2 Conversion to Smart Meter Measurements

In practice, smart meters read line-line voltage magnitudes $|u_i^1 - u_i^2|$ and the sum of single-phase powers $p_i^1 + p_i^2$. But (3.1) relates single-phase net injections to line-ground voltages. We need to change (3.1)) such that it relates the former quantities.

First, we assume that the following approximation holds (u_i^1, u_i^2 are line-to-ground voltage phasors, θ_i^1, θ_i^2 are line-to-ground voltage angles, of node i phase 1 and 2. The number 120 and 240 are the nominal line-ground and line-line voltages). (3.5) is valid when all θ_i^1, θ_i^2 are near those of the modified flat voltage solution. Thus the measured voltage magnitude can be approximately written as $|u_i^1 - u_i^2| \approx v_i^1 + v_i^2 + 240$.

$$\begin{aligned}|u_i^1 - u_i^2| &= |(v_i^1+120)\cos(\theta_i^1) + j(v_i^1+120)\sin(\theta_i^1) - (v_i^2+120)\cos(\theta_i^2) - j(v_i^2+120)\sin(\theta_i^2)| \\ &\approx |v_i^1 + v_i^2 + 240| \qquad (3.5)\end{aligned}$$

Next, we introduce two new vectors $\mathbf{x^s}$ and $\mathbf{y^s}$:

$$\begin{aligned}\mathbf{x^{s\mathsf{T}}} &= \Big[(\mathbf{v}^1+\mathbf{v}^2)^\mathsf{T}, (\mathbf{v}^1-\mathbf{v}^2)^\mathsf{T}, \boldsymbol{\theta}^{1\mathsf{T}}, \boldsymbol{\theta}^{2\mathsf{T}}\Big] = [\mathbf{v^{s\mathsf{T}}}, \boldsymbol{\theta}^{\mathbf{s}\mathsf{T}}] \\ \mathbf{y^{s\mathsf{T}}} &= \Big[(\mathbf{p}^1+\mathbf{p}^2)^\mathsf{T}, (\mathbf{p}^1-\mathbf{p}^2)^\mathsf{T}, \mathbf{q}^{1\mathsf{T}}, \mathbf{q}^{2\mathsf{T}}\Big] = [\mathbf{p^{s\mathsf{T}}}, \mathbf{q^{s\mathsf{T}}}]\end{aligned}$$

which are related to $\mathbf{x^r}$ and $\mathbf{y^r}$ via $\mathbf{M} = \mathrm{diag}\left(\left[\begin{smallmatrix}\mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I}\end{smallmatrix}\right], \left[\begin{smallmatrix}\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\end{smallmatrix}\right]\right)$: $\mathbf{x^r} = \mathbf{M}^{-1}\mathbf{x^s}$ and $\mathbf{y^r} = \mathbf{M}^{-1}\mathbf{y^s}$. Substituting these relationships into (3.1) yields $\mathbf{y^s} = \mathbf{M}\mathbf{L^r}\mathbf{M}^{-1}\mathbf{x^s} = \mathbf{L^s}\mathbf{x^s}$, or:

$$\begin{bmatrix}\mathbf{p^s} \\ \mathbf{q^s}\end{bmatrix} = \begin{bmatrix}\mathbf{L}_{11}^{\mathbf{s}} & \mathbf{L}_{12}^{\mathbf{s}} \\ \mathbf{L}_{21}^{\mathbf{s}} & \mathbf{L}_{22}^{\mathbf{s}}\end{bmatrix}\begin{bmatrix}\mathbf{v^s} \\ \boldsymbol{\theta}^{\mathbf{s}}\end{bmatrix} \qquad (3.6)$$

Further, we remove the dependency on $\boldsymbol{\theta}^1$ and $\boldsymbol{\theta}^2$ to obtain:

$$\mathbf{p^s} = \big(\mathbf{L}_{11}^{\mathbf{s}} - \mathbf{L}_{12}^{\mathbf{s}}\mathbf{L}_{22}^{\mathbf{s}\dagger}\mathbf{L}_{21}^{\mathbf{s}}\big)\mathbf{v^s} + \mathbf{L}_{12}^{\mathbf{s}}\mathbf{L}_{22}^{\mathbf{s}\dagger}\mathbf{q^s} \qquad (3.7)$$

Where $\mathbf{L}_{22}^{\mathbf{s}\dagger}$ is the pseudoinverse of $\mathbf{L}_{22}^{\mathbf{s}}$. We prove (3.7) in Appendix A.3.

Now, if each node has a constant lagging power factor over the entire time period, we can write $\left[\begin{smallmatrix}\mathbf{q}^1 \\ \mathbf{q}^2\end{smallmatrix}\right] = \mathbf{D}\left[\begin{smallmatrix}\mathbf{p}^1 \\ \mathbf{p}^2\end{smallmatrix}\right]$ where $\mathbf{D}$ is a positive definite diagonal matrix. Then we have

$\mathbf{q}^{\mathbf{s}} = \mathbf{D}\mathbf{M}_u^{-1}\mathbf{p}^{\mathbf{s}}$ where $\mathbf{M}_u$ is the upper left block of $\mathbf{M}$. Under the constant lagging power factor assumption, we can then simplify (3.7) to

$$\mathbf{p}^{\mathbf{s}} = (\mathbf{I} - \mathbf{L}_{12}^{\mathbf{s}}\mathbf{L}_{22}^{\mathbf{s}\dagger}\mathbf{D}\mathbf{M}_u^{-1})^{-1}\big(\mathbf{L}_{11}^{\mathbf{s}} - \mathbf{L}_{12}^{\mathbf{s}}\mathbf{L}_{22}^{\mathbf{s}\dagger}\mathbf{L}_{21}^{\mathbf{s}}\big)\mathbf{v}^{\mathbf{s}} \tag{3.8}$$

We argue that $(\mathbf{I} - \mathbf{L}_{12}^{\mathbf{s}}\mathbf{L}_{22}^{\mathbf{s}\dagger}\mathbf{D}\mathbf{M}_u^{-1})$ is nonsingular in practice in Appendix A.3. The first n_c equations of (3.8) are

$$\underbrace{\mathbf{p}^1 + \mathbf{p}^2}_{-\boldsymbol{p}} = \mathbf{L}_{pv}^{(+)}\underbrace{(\mathbf{v}^1 + \mathbf{v}^2)}_{\boldsymbol{v} - \mathbf{1}\cdot 240} + \mathbf{L}_{pv}^{(-)}(\mathbf{v}^1 - \mathbf{v}^2) \tag{3.9}$$

where $\boldsymbol{p}$ and $\boldsymbol{v}$ are smart meter net power consumption and voltage measurements respectively. (3.9) shows that the real power measurements depend on the observed voltage sums and the unobserved voltage differences. This latter term is negligible. Accounting for measurement errors with a noise term $\boldsymbol{\epsilon}$, we thus have

$$\boldsymbol{p} = \mathbf{L}_{pv}(\boldsymbol{v} - \mathbf{1}\cdot 240) + \boldsymbol{\epsilon} = \mathbf{L}_{pv}\boldsymbol{v} + \boldsymbol{\epsilon} \tag{3.10}$$

where $\mathbf{L}_{pv} = -\mathbf{L}_{pv}^{(+)}$ whose nullspace contains $\mathbf{1}$.

3.3.3 A Remedy for Not Having Transformer Data

In most cases, transformers do not have smart meters installed on them. That is, if we partition (3.10) with respect to transformer node and customer nodes:

$$\begin{bmatrix} p_T \\ \boldsymbol{p}_C \end{bmatrix} = \begin{bmatrix} l_{TT} & \boldsymbol{l}_{TC} \\ \boldsymbol{l}_{CT} & \mathbf{L}_{CC} \end{bmatrix} \begin{bmatrix} v_T \\ \boldsymbol{v}_C \end{bmatrix} + \begin{bmatrix} \epsilon_T \\ \boldsymbol{\epsilon}_C \end{bmatrix} \tag{3.11}$$

Then the measurements for v_T and p_T in (3.11) are missing. We can remedy this by using conservation of energy to write $p_T \approx -\mathbf{1}^T \mathbf{p}_C$. This relationship is exact when there are no losses. We can then eliminate v_T from (3.11) and replace the remaining p_T term to obtain:

$$\boldsymbol{p}_C = -\boldsymbol{l}_{CT} l_{TT}^{-1} \mathbf{1}^{\mathsf{T}} \boldsymbol{p}_C + (\mathbf{L}_{CC} - \boldsymbol{l}_{CT} l_{TT}^{-1} \boldsymbol{l}_{TC}) \boldsymbol{v}_C + \boldsymbol{\epsilon}'_C \tag{3.12}$$

Where $\boldsymbol{\epsilon}'_C = \boldsymbol{\epsilon}_C - \boldsymbol{l}_{CT} l_{TT}^{-1} \epsilon_T$. (3.12) motivates the use of $\mathbf{1}^{\mathsf{T}} \boldsymbol{p}_C$ as a covariate in model estimation. We will use it in Section 3.4 where we develop the modified linear model.

3.4 Technical Methods

This section details each step of the proposed framework outlined in Section 3.1. To avoid tedious notation, all quantities correspond to *one* rolling window.

3.4.1 Data Preprocessing

Most real-world smart meter datasets contain missing values and outliers. The time stamps with missing values and/or power outages are discarded from the analysis.

The amount of discarded data is usually negligible. Outliers are much more frequent and can hurt our regression models [72]. We discuss the problem of outliers in voltage data in this subsection.

A properly trained model would be sensitive to voltage measurement errors because the voltages vary around flat condition by a very small amount. In this work, the time stamps where voltage error is large will be removed. The method is as follows. First, we train a regression model that is robust to outliers on the training dataset. We then apply the model to each customer $i \in \{1, 2, \cdots, n_c\}$ and search for *training* time stamps $\mathcal{T}_i^{\text{out}} = \{t_{i,1}^{\text{out}}, \cdots, t_{i,i_o}^{\text{out}}\}$ with large residuals. For notational clarity, all references to quantities involving robust regression will carry the superscript rb. Then:

$$t \in \mathcal{T}_i^{\text{out}} \quad \text{if} \quad \begin{array}{l} \left(\tilde{y}_i^{rb}(t)/\text{var}(\tilde{y}_i^{rb})\right)^2 > F_{\chi_1^2}^{-1}(0.999) \quad \text{and} \\ \\ (\boldsymbol{v}(t) - \bar{\boldsymbol{v}})^{\mathsf{T}} \boldsymbol{\Sigma}_{\boldsymbol{v}}^{-1} (\boldsymbol{v}(t) - \bar{\boldsymbol{v}}) > F_{\chi_{n_c}^2}^{-1}(0.999) \end{array} \tag{3.13}$$

where $\tilde{y}_i^{rb}(t) = y_i(t) - \hat{y}_i^{rb}(t)$ is the estimation residual for customer i at time t, $\text{var}(\tilde{y}_i^{rb}) = \sum_t (\tilde{y}_i^{rb}(t) - \bar{\tilde{y}}_{,i}^{rb})^2/(T-1)$ is its empirical variance. $\bar{\boldsymbol{v}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{v}}$ are the sample mean vector and covariance matrix of voltage measurements. $F_{\chi_1^2}^{-1}$ is the inverse of chi-square CDF with one degree of freedom.

After the sets $\{\mathcal{T}_i\}_{i=1}^{n_c}$ are found, we remove any time instances that are a member of *two or more* of these sets. That is, if $t_p \in \mathcal{T}_i^{\text{out}} \cap \mathcal{T}_j^{\text{out}}, i \neq j$, then the measurements $p_i(t_p), v_i(t_p)$ are discarded for all $i \in \{1, 2, \cdots, n_c\}$. The reasoning behind this final rule is as follows. If there is a voltage outlier, then at least two of the customers' regression residuals will be severely affected. This can be understood from (3.10).

Robust regression methods such as least median of squares (LMS) [73]; M-estimator [74]; and random sample consensus (RANSAC) [75] can be used. In this work, we use RANSAC for its simplicity and efficiency.

3.4.2 Modified Linear Model

The ideas outlined in Section 3.3 are combined to produce the following *modified linear model (MLM)*:

$$\begin{aligned} y_i(t) &= \begin{bmatrix} \mathbf{x}(t)^\mathsf{T} & \sum_{j=1}^{n_c} y_j(t) \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}_i^{\mathcal{X}} \\ \beta_i^y \end{bmatrix} + \epsilon_i'(t) \\ &= \mathcal{X}(t)^\mathsf{T}\mathcal{B}_i + \epsilon_i'(t) \end{aligned} \tag{3.14}$$

where $\mathbf{x}(t) = [v_1(t), v_2(t), \cdots, v_{n_c}(t)]^\mathsf{T}$: vector of voltage readings at time t, $y_i(t) = p_i(t)$: kWh readings of customer i at time t, and $\epsilon_i'(t)$ accounts for measurement noise and unobserved dependencies as in (3.12).

The parameter vectors $\{\mathcal{B}_i\}_{i=1}^{n_c}$ will be estimated by using ordinary least squares (OLS) on the training data ($\mathcal{X}^\mathcal{D}$, $\mathbf{y}^\mathcal{D}$) [76], which is a portion of the rolling window. This estimate is achieved by solving the normal equations:

$$\left(\mathcal{X}^{\mathcal{D}\mathsf{T}}\mathcal{X}^{\mathcal{D}}\right)[\mathcal{B}_1, \cdots, \mathcal{B}_{n_c}] = \mathcal{X}^{\mathcal{D}\mathsf{T}}[\mathbf{y}_1^\mathcal{D}, \cdots, \mathbf{y}_{n_c}^\mathcal{D}] \tag{3.15}$$

Variations of OLS such as total least squares (TLS) [77] [78] can be used instead, but these do not exhibit the properties described in the next subsection.

The fitted model is then used to predict kWh consumption values for the testing data ($\mathcal{X}$, $\mathbf{y}$) within the rolling window:

$$[\hat{\mathbf{y}}_1, \cdots, \hat{\mathbf{y}}_{n_c}] = \mathcal{X}[\mathcal{B}_1, \cdots, \mathcal{B}_{n_c}] \tag{3.16}$$

The LHS are used to calculate the residual time series $\tilde{\mathbf{y}}_i = \mathbf{y}_i - \hat{\mathbf{y}}_i$, which are used to perform electricity theft detection.

In this work, the length of each rolling window is chosen to be 67 days. The first 60 days and the last 7 days form the training and testing data, respectively. Each rolling window is 1 day ahead of the one preceding it.

3.4.3 Properties of Residuals Under Theft

The residuals of the MLM change when there is an energy thief. Denote $\mathbf{y}_i^{(e)}$ as the kWh meter data for customer i when one of the customers in the same secondary is a thief. Let the original symbol $\mathbf{y}_i$ denote the kWh meter data of the same customer had there been no theft activities. Suppose without loss of generality that customer i is the thief, then

$$\mathbf{y}_i^{(e)} = \mathbf{y}_i - \mathbf{y}_i^s; \quad \mathbf{y}_j^{(e)} = \mathbf{y}_j \quad \forall j \neq i$$

where the non-negative vector $\mathbf{y}_i^s$ denotes the difference between the imagined kWh measurement and the actual one. Let $\tilde{\mathbf{y}}_i^{(e)}$ and $\tilde{\mathbf{y}}_i$ denote the out-of-sample residual time series for the energy thief. Then the following results hold

Lemma 3.4.1

$$\tilde{\mathbf{y}}_i^{(e)} - \tilde{\mathbf{y}}_i = -\sum_{j \neq i} \beta_j^y \mathbf{y}_i^s \tag{3.17}$$

Lemma 3.4.2

$$\sum_j \tilde{\mathbf{y}}_j^{(e)} = \sum_j \tilde{\mathbf{y}}_j = \mathbf{0} \tag{3.18}$$

Lemma 3.4.3 *For any $\delta > 0$, there exists a training data window length $T > 0$ such that for each j*

$$\mathbb{P}(\beta_j^y \geq -\delta) > 1 - \delta \tag{3.19}$$

Lemma 3.4.1 and Lemma 3.4.3 combine to show that a thief's residuals will become negative once he or she begins to steal power. Lemma 3.4.2 shows that the residuals of the other customers will raise in order to balance their sum. These conclusions are useful to the design of the postprocessing method discussed in Section 3.4.4. The proofs are given in Appendix A.4.

3.4.4 Energy Theft Detection

We define an anomaly score in terms of the residuals $\tilde{\mathbf{y}}_i$ for each customer. Customers with high anomaly scores are likely to be thieves or have malfunctioning smart meters. Before defining the anomaly score, we post-process the residuals in two steps. The first step removes outliers. This step is analogous to the preprocessing stage except here, we substitute its residual value by that of the nearest future non-outlier. The second step sets all positive residuals to zero. This rule comes from experimentation and the lemmas of the previous subsection. We denote the resulting residual time series after the two steps as $\tilde{\mathbf{y}}_i'$.

Until now, we have ignored the subscript for the rolling windows. It is necessary to introduce it here. We use the symbol $f = 1, 2, \cdots$ to index the rolling windows. The anomaly score for each customer i and each rolling window f is defined as $d_i(f) = w_i(f) \left\| \tilde{\mathbf{y}}_i'(f) \right\|_2$ where $w_i(f) = \sqrt{|t^{\mathcal{D}}(f)|} / \left\| \tilde{\mathbf{y}}^{\mathcal{D},i}(f) \right\|_2$ is a weighting coefficient. Energy thefts are identified by ranking $d_i(f)$ for all i and all f. The higher $d_i(f)$ is, the higher priority of investigation customer i should have. This ranking method is simplified to ranking $\max_f d_i(f)$ for all i when theft time is unimportant.

3.5 Numerical Studies

This section evaluates the performance of the proposed method on a real dataset with synthetic electricity theft cases. In Section 3.5.1, we describe the dataset in detail. In Section 3.5.2, we test the performance of the modified linear model without energy theft. In Section 3.5.3-Section 3.5.5, we demonstrate the impact of energy theft on out-of-sample residuals and anomaly scores. In Section 3.5.6, we compare the performance of our proposed anomaly detection method with comparable methods.

3.5.1 Experimental Data Description

Real-world Smart Meter Data

The smart meter dataset comes from a 12 KV distribution feeder in Southern California Edison (SCE)'s service territory. The schematic of the testing distribution feeder is shown in Figure 3.2. Measurements were taken from August 1, 2015 to Feb 1, 2016, including the customers' hourly average voltage magnitudes and electricity consumption. A majority of the customers on the distribution feeder are residential customers. The transformer to customer association information is also provided by SCE. 190 such transformers were selected for the experimental study. This accounts for 980 residential customers.

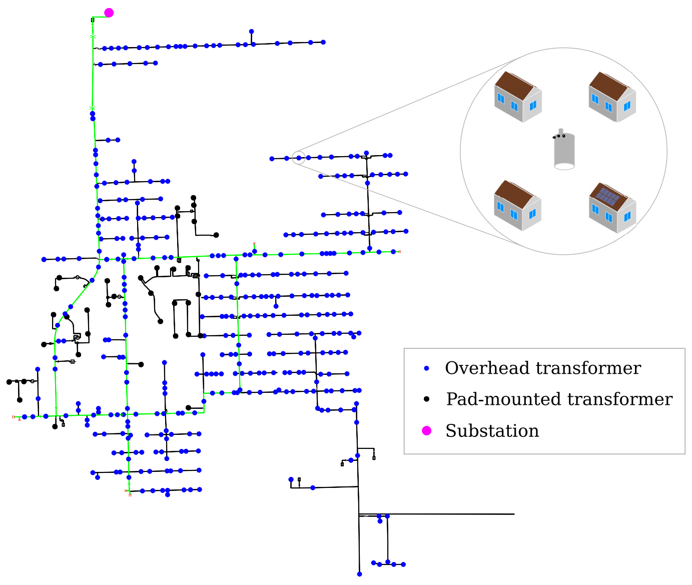


Figure 3.2: Schematic of the test distribution feeder.

Synthetic Electricity Theft Data

Similar to other literature, we synthesize electricity theft data [55] [79] [80]. The *attack invariant* principle [81] is followed during the data synthesis process: $\sum_{t\in\mathcal{T}_e} p_k^{(e)}(t) < \sum_{t\in\mathcal{T}_e} p_k(t)$ where the kth customer is stealing power during time period $\mathcal{T}_e$. $p_k(t)$ denotes the actual electric power consumed by the kth customer. $p_k^{(e)}(t)$ is the electricity consumption of the kth customer recorded by the electric utility.

The amount of electricity theft from the kth customer during hour t, $p_k^s(t)$, is defined as $p_k^s(t) = p_k(t) - p_k^{(e)}(t)$ where $0 \leqslant p_k^s(t)$. Within the attack invariant principle, four electricity theft cases are simulated.

Case 1: 100% of electricity theft for n hours: $p_k^s(t) = p_k(t)$

Case 2: A constant amount of electricity theft: $p_k^s(t) = \alpha_{\mathrm{c2}}$

Case 3: A uniformly distributed electricity theft: $p_k^s(t) \sim \mathcal{U}(0, \alpha_{\mathrm{c3}})$

Case 4: A constant percentage of electricity theft: $p_k^s(t) = \alpha_{\mathrm{c4}} p_k(t)$

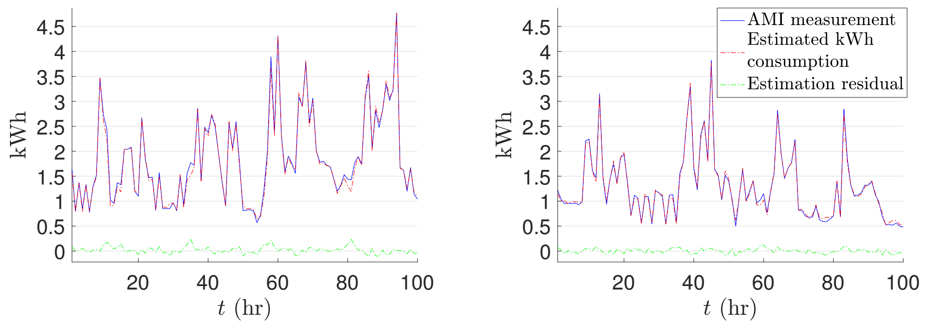
In this work, we assume the time period when electricity theft occurs is a consecutive subset of all time stamps of our dataset, that is, $\mathcal{T}_e(t_1^{(e)}, t_2^{(e)}) = \{t : t_1^{(e)} \leq t < t_2^{(e)}\}$. Data synthesis is performed within $\mathcal{T}_e(t_1^{(e)}, t_2^{(e)})$.

The synthetic electricity theft case for the kth customer is created as follows. If customer k does not have DERs, then $p_k(t) - p_k^s(t) \mapsto p_k^{(e)}(t)$ and $\max(p_k^{(e)}(t), 0) \mapsto p_k^{(e)}(t)$. The synthesized electricity consumption of a customer without DERs should be higher than zero. If customer k does have DERs, then the floor for net electricity consumption recording should be the electricity delivered back to the grid.

3.5.2 Performance of the Modified Linear Model

Consider a distribution secondary circuit consisting of 4 residential customers as highlighted in Figure 3.2. The rolling window under study is set up as follows. The training dataset $t^{\mathcal{D}}$ starts at hour 1 and ends at hour 1440 from 60 days. The testing dataset $t^{\mathcal{D}_a}$ includes 168 consecutive hours from 7 days following the training dataset.

We first show that the proposed MLM accurately estimates the electricity consumption of a given customer. This customer's true consumption, estimated consumption, and residuals are depicted in Figure 3.3. We plot this data for the first 100 hours of the in-sample and out-of-sample periods. The average electric load consumed by this customer



(a) In-sample ($t^{\mathcal{D}}$) (b) Out-of-sample ($t^{\mathcal{D}_a}$)

Figure 3.3: Electricity consumption residuals from the MLM for one customer.

is 1.6 kWh. The mean of the estimation residual is -0.01 kWh and its standard deviation is 0.1 kWh. Both of these are small compared to the customer's average load. This result shows that the MLM estimates the electricity consumption of a customer quite well.

We next apply this analysis to every customer on the feeder over 59 rolling windows. The first window is the same as above. Each other window is 24 hours ahead of the one preceding it. The box plot of the in-sample residual sample standard deviation, out-of-sample residual sample mean and standard deviation for all customers are shown in Figure 3.4. The statistics of the example customer shown in Figure 3.3 are highlighted by the yellow dashed lines. Most customers' have a small residual mean and standard deviation.

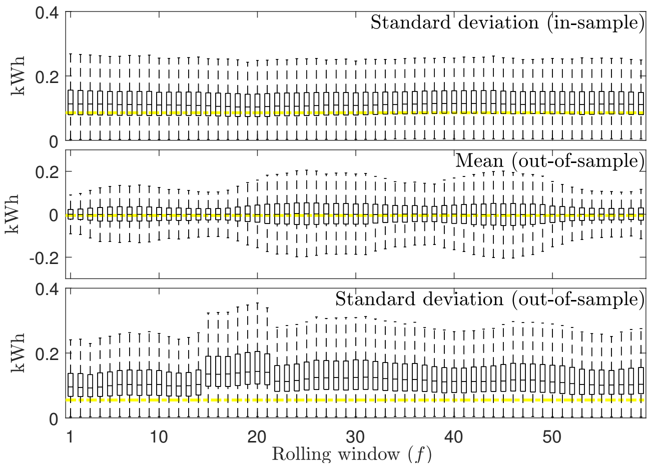


Figure 3.4: Residual statistics of all customers on the distribution feeder.

Hence the proposed MLM is accurate in estimating the consumption of most customers on the feeder. Yet, some customers do have relatively large residuals. These are likely due to errors in the customer to transformer mapping and noisy smart meter data.

Finally, we compare the performance of the MLM with three nonlinear regression models. A Feed-forward Neural Network (FNN), a Radial Basis Function Network (RBF) [82], and a Support Vector Regression (SVR) model [83]. The inputs and outputs of the nonlinear models are the same as MLM. The number of hidden units in the FNN is one plus the number of inputs, the number of neurons in the RBF is 200, and the kernel for the SVR is a degree 2 polynomial. Five equally spaced rolling windows between the 1st and 59th - shown in Figure 3.4 - were selected to perform the regression analysis. All other experimental setups are identical as in Figure 3.4. The results are reported in Table 3.1. Each cell of Table 3.1 shows the $\mu \pm 2\sigma$ of the corresponding performance measure. All values are in kWh and have been rounded to 2 decimal places. Table 3.1 shows that all regression

Table 3.1: Comparison of regression models

	std (in-sample)	mean (out-of-sample)	std (out-of-sample)
MLM	0.12 ± 0.09	0.00 ± 0.09	0.13 ± 0.10
FNN	0.11 ± 0.08	0.00 ± 0.09	0.13 ± 0.10
RBF	0.12 ± 0.07	0.00 ± 0.08	0.17 ± 0.13
SVR	0.12 ± 0.09	0.01 ± 0.08	0.13 ± 0.10

models perform similarly over a wide range of customers. But we value interpretability and simplicity over sophistication and complexity. For this reason, our experimental studies shall focus on the proposed modified linear model.

3.5.3 Properties of the Anomaly Score

This subsection studies the proposed energy theft detection scheme of section IV. The goal of the following experiments is twofold. First, it confirms that ranking the maximum anomaly score $\max_f d_i(f)$ for all i is a good way to detect energy thieves. Second, we show that $\max_f d_i(f)$ generally occurs during a rolling window with nice properties. The properties in question are cleanliness of the training dataset and theft strength of the testing dataset.

For illustrative purposes, we consider the following experiment. First, we give synthetic theft data to the customer depicted in Figure 3.3. This synthesized data follows case 3 with parameter $\alpha_{c3} = 1.8$ kWh. We then increase $|\mathcal{T}_e \cap t^{\mathcal{D}}|/|t^{\mathcal{D}}|$ and $|\mathcal{T}_e \cap t^{\mathcal{D}_a}|/|t^{\mathcal{D}_a}|$ from 0 to 1 and average the results from 10 such simulations. The resulting anomaly scores for the first window are shown in Figure 3.5. Figure 3.5 shows that the anomaly score increases with the amount of $\mathcal{T}_e$ contained in the testing dataset. However, it decreases with the amount of $\mathcal{T}_e$ contained in the training dataset. A maximum occurs when $|\mathcal{T}_e \cap t^{\mathcal{D}}|/|t^{\mathcal{D}}|$ is 0 and $|\mathcal{T}_e \cap t^{\mathcal{D}_a}|/|t^{\mathcal{D}_a}|$ is 1.

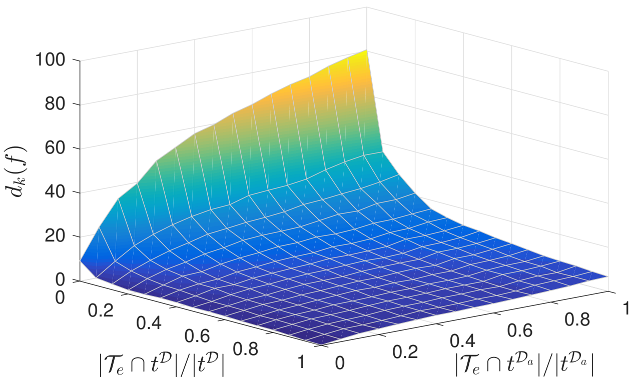


Figure 3.5: Anomaly scores for the example customer

We extend these properties of the anomaly score to all rolling windows, all synthetic theft cases, and all customers. To do this, we consider 5 different theft intervals $\mathcal{T}_e$, which begin at 20%, 30%, 40%, 50%, and 60% of the way through the dataset and end at the last sample. The 4 synthetic theft cases were considered with parameters given by $\alpha_{c2} = 1$ kWh; $\alpha_{c3} = 1.8$ kWh; $\alpha_{c4} = 0.5$. In total, we have 20 different synthesized datasets for each customer. The 59 rolling windows described in Section 3.5.2 were simulated $980 \cdot 20$ times. That is, for each rolling window, there is a simulation for each customer in each of the 20 theft modes. For each simulation, denote k as the index of that simulation's thief. We report the value and ranking percentile of $\max_f d_k(f)$ among $\max_f d_i(f)$ of all other customers i in Table 3.2. The numbers in the parenthesis are the ranking percentile of the

Table 3.2: Maximum anomaly score $\max_f d_i(f)$ and ranking percentile averaged over all customers

Case \ $\mathcal{T}_e$	20%	30%	40%	50%	60%
1	16.2 (55)	30.7 (10)	79.3 (1)	88.8 (1)	94.0 (1)
2	16.6 (53)	26.9 (16)	67.4 (1)	70.2 (1)	69.8 (1)
3	17.5 (47)	27.1 (16)	61.3 (1)	64.3 (1)	64.7 (1)
4	17.0 (50)	25.2 (19)	43.4 (3)	46.8 (2)	49.6 (2)

anomaly score in that cell.

In the first two columns of Table 3.2, the anomaly periods $\mathcal{T}_e$ intersect the training dataset. As a result, the maximum anomaly scores $\max_f d_i(f)$ are indistinguishable from the anomaly scores of non-thieves. However, these scores increase as $\mathcal{T}_e$ takes up smaller portions of the training dataset as shown in the last three columns of Table 3.2. Thus the rolling window approach is useful when there is no theft for the first part of the analysis. The exact amount of time necessary for this part of the analysis depends on the length of the training window.

The highest anomaly scores correspond to the rolling window which has a maximum amount of clean data in training set and a minimum amount of clean data in testing set. This intuition is confirmed in Table 3.3. Each cell is the difference $|t^{\mathcal{D}_a} \cap \mathcal{T}_e|/|t^{\mathcal{D}_a}| - |t^{\mathcal{D}} \cap \mathcal{T}_e| \cdot |t^{\mathcal{D}}|$ averaged over all customers.

Table 3.3: Difference $|t^{\mathcal{D}_a} \cap \mathcal{T}_e|/|t^{\mathcal{D}_a}| - |t^{\mathcal{D}} \cap \mathcal{T}_e| \cdot |t^{\mathcal{D}}|$

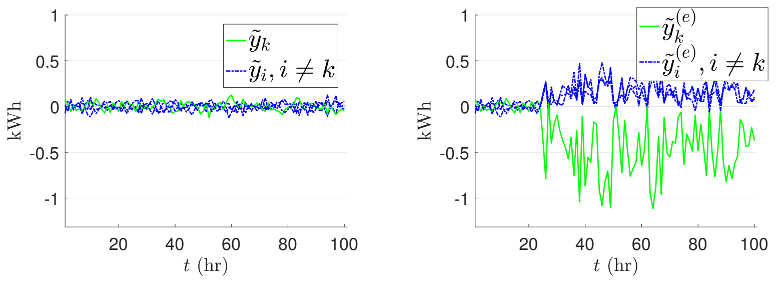
Case \ $\mathcal{T}_e$	20%	30%	40%	50%	60%
1	0.44	0.77	0.92	0.93	0.93
2	0.37	0.77	0.93	0.92	0.92
3	0.35	0.76	0.92	0.92	0.92
4	0.25	0.71	0.88	0.89	0.87

The detection abilities of these anomaly scores need a window that is both clean and strong in theft. Such a window will exist so long as the thief does not steal power throughout the entire analysis. Table 3.2 shows that this window will be recognizable because the anomaly score of the thief will increase substantially during this window. We may idealize this window as one with $|t^{\mathcal{D}} \cap \mathcal{T}_e| \cdot |t^{\mathcal{D}}| = 0$ and $|t^{\mathcal{D}_a} \cap \mathcal{T}_e| \cdot |t^{\mathcal{D}_a}| \approx 1$. Since this particular rolling window is of crucial importance, we study it in detail in Section 3.5.4 and Section 3.5.5.

3.5.4 The Impact of Energy Theft on Out-of-sample Residuals

This subsection focuses on the behavior of out-of-sample residuals when the training set is clean. It emphasizes a difference in behavior between theft and non-theft scenarios.

We first synthesize smart meter data for customer k under synthetic case 3. We assume that the electricity theft activities occur from hour $t_1^{(e)} = 25$ to hour $t_2^{(e)} = 168$ in the out-of-sample period. The amount of electricity theft is assumed to follow a uniform distribution with $p_k^s(t) \sim \mathcal{U}(0, 1.8)$ (kWh). The MLM is applied for all 4 customers in the same secondary. The out-of-sample residuals for the 4 customers are shown in Figure 3.6 (b). The figure represents the residuals of customer k by the solid green line. The other 3 customers' residuals are represented by blue dash lines. The out-of-sample residuals obtained from the original data (without electricity theft) are shown in Figure 3.6 (a) for comparison purposes. As shown in Figure 3.6 (b), customer k has negative residuals while the honest customers have positive residuals. The sum of them at any given hour is zero as stated in Lemma 3.4.2. For all customers, the regression coefficients $\sum_{\ell \neq j} \beta_\ell^y$ are positive.



(a) original data (b) synthesized data

Figure 3.6: Out-of-sample residuals.

In this case, $\sum_{\ell \neq k} \beta_\ell^y$ took on the values of 0.74, 0.77, 0.67 and 0.81 for $j = 1, 2, 3$ and 4. Hence, the residuals of the dishonest customer k will always be negative according to Lemma 3.4.1. These results show that the residual plots of all customers on the same secondary are helpful in detecting electricity theft.

3.5.5 The Impact of Energy Theft on Anomaly Scores

Next, we will show that electricity theft can be easily detected by anomaly scores in a wide variety of cases. We further show that the anomaly scores increase with the amount of stolen electricity.

We first calculate the anomaly scores for all customers on the distribution feeder under the experiment detailed in Section 3.5.4. The anomaly score of customer k and the summary statistics of all customers' anomaly scores are reported in Table 3.4. As shown in the table, the kth customer has an anomaly score of 79.4. This is the highest among all 980 customers in the distribution feeder. The second highest anomaly score of any customer is 42.2 which is much lower than that of customer k. The average and 95th percentile of all customers' anomaly scores PR(d_i,95) are 8.1 and 14.6. Both of these are much lower than that of customer k. In this case, the anomaly score is quite useful and easily detects the electricity theft activity.

Table 3.4: Anomaly Scores

d_k	Ranking	$\sum_i d_i/N$	PR(d_i,95)	$\max_{i \neq k} d_i$
79.4	1 (0.1%)	8.1	14.6	42.2

Synthetic electricity theft datasets for customer k are then created for each of the synthetic cases. The parameters used for the cases are as follows. In case 1, the theft activity starts from hour 1 in the out-of-sample period. Multiple datasets for this case are then created by increasing the theft ending hour in the out-of-sample period. We create a dataset for each ending hour from hour 2 to hour 168. In cases 2-4, the theft activity starts and ends with hours $t_1^{(e)} = 0.2|t^{\mathcal{D}_a}|$ and $t_2^{(e)} = |t^{\mathcal{D}_a}|$. Multiple datasets for this case are then created by increasing the parameters $\alpha_{\mathrm{c}2}$, $\alpha_{\mathrm{c}3}$, and $\alpha_{\mathrm{c}4}$. The parameters are varied such that the total amount of stolen electricity ranges from 1 kWh to 128 kWh. Again the electricity theft activities are assumed to occur during the out-of-sample period.

We then calculate the residual and anomaly score for each constructed dataset. The anomaly score of customer k and the summary statistics of all customers' anomaly scores are depicted in Figure 3.7. The colored solid curve in each subplot represents the anomaly scores of customer k. The numbers along the curve show the ranking percentile of customer k's anomaly score. The solid black line represents the 95th percentile of all customers' anomaly scores. The dashed black line represents the average anomaly score of all customers. Both axes are on a logarithmic scale. The figure shows that the anomaly score of customer k increases monotonically with the amount of stolen electricity. In all cases, customer k's anomaly score will surpass the 95th percentile of all customers if it steals more than 32 kWh. This averages to 0.19 kW of power. A stronger result holds for cases 1-3. In these cases, customer k's anomaly score will be the absolute largest of all customers if it steals more than 0.38 kW of power.

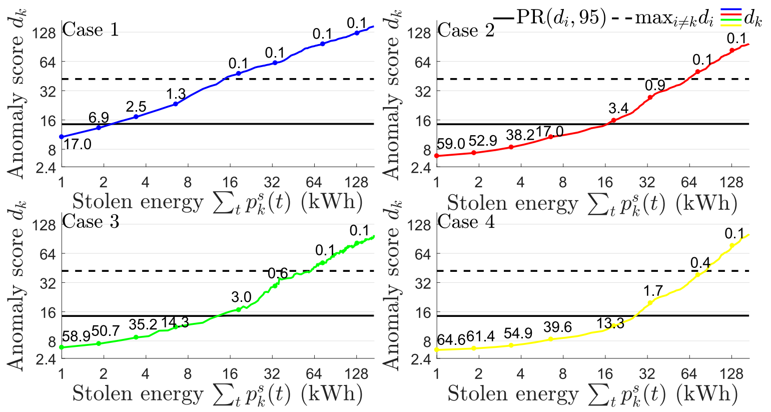


Figure 3.7: Anomaly scores versus amount of stolen electricity

To further prove the validity of the proposed framework, we extend this analysis to all customers. That is, the previous case study is repeated 980 times. Each new set of cases sets a new customer as the thief.

The anomaly scores of the customers who are stealing electricity are binned and reported in Figure 3.8. The x- and y-axes of the figure represent the amount of stolen electricity and the anomaly scores. The z-axis represents the number of customers who have an anomaly score which falls into a particular bin. The color of the each bar indicates the ranking (in percentage) of anomaly score of the customers in that bin against all honest customers. A darker color represents a higher ranking. Each row of bars add up to the total number of customers in the distribution feeder. When the amount of stolen electricity increases, the distribution of dishonest anomaly scores shifts to the right. The ranking of the anomaly scores also increases. Finally, the figure can be used to predict the probabilities of detection. For example, if a customer steals more than 0.38 kW of power, then it is has a 97 percent chance of surpassing the 95th percentile of all customers. It further has a 57 percent chance that its anomaly score will be the highest among all customers. These results show that framework is effective in catching even small amounts of theft.

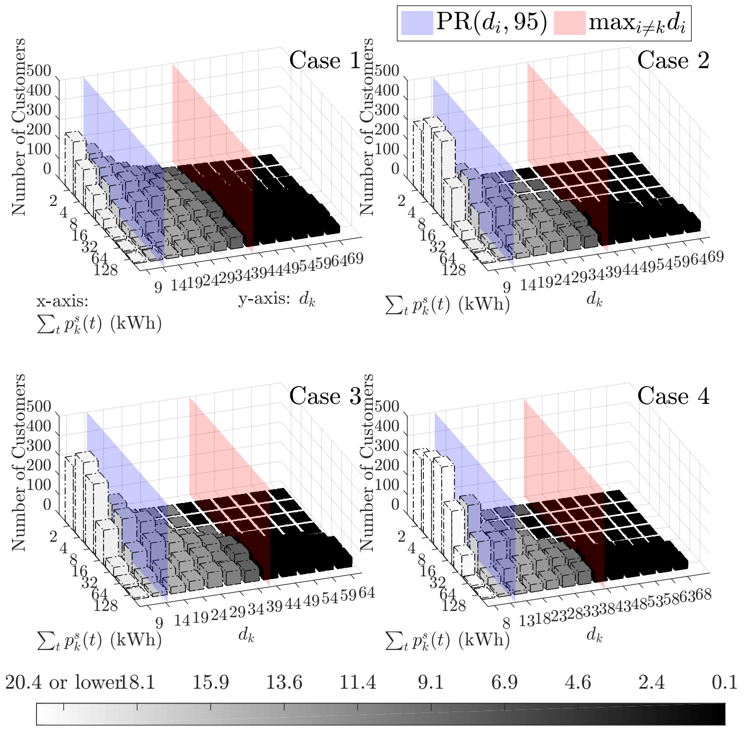


Figure 3.8: Numerical evaluation for all customers on the distribution feeder

3.5.6 Comparison with Existing Techniques

We compare the performance of the proposed anomaly detection method with the Fuzzy C-means (FCM) based method [43], the Self-Organizing Maps (SOM) based method [48], and Random Matrix Theory (RMT) based method [63]. We excluded the comparisons with state estimation based methods, supervised machine learning based methods, and the "central observer" based methods such as [56] because they represent solutions to different classes of problems. The original methods described in [43], [48], and [63] need to be modified slightly to match our experimental data and our performance measure.

For the FCM method. First, missing values were imputed by the average values of the two nearest time stamps of the same customer. Next, the time series data was dimension-reduced via the t-distributed Stochastic Neighbor Embedding (t-SNE) algorithm [84]. This was performed separately on the consumption and voltage time series data. The results were concatenated to form a feature vector. Second, the definition of the anomaly score in [43, Figure 3] is modified to be $d_k = \|\mathbf{u}_{0k} - \mathbf{u}_k\mathbf{P}^\star\|_2$ where $\mathbf{P}^\star$ is a column permutation matrix such that $\mathbf{P}^\star = \text{argmin } \|\mathbf{U}_0 - \mathbf{UP}\|$. For the SOM method. First, the kWh and voltage time series of each customer were converted to kWh and voltage daily profiles. Missing values and outage values were treated similarly to the FCM method above. The step of comparing with the contracted power demand was removed. We further defined the minimum quantization error [85] as the anomaly score. For the RMT method. First, each distribution transformer secondary is considered as a region. Next, a window of 67 days of hourly two-phase voltage magnitude and active power measurements are collected and undergo the same preprocessing procedure as described in Section 3.4.1. The active power measurement noise is assumed to be zero mean normal with a standard deviation of 0.02 times the range of active power.

We setup experiments as follows. The four different anomaly cases discussed in Section 3.5.1 will be simulated with the parameters $\mathcal{T}_e, \alpha_{c2}, \alpha_{c3}$ and α_{c4} being varied such that the total amount of bypassed electricity ranges from 2 kWh to 128 kWh. We performed the experiments using the same set of customers and training/testing dataset as depicted in Figure 3.8. The results are shown in Table 3.5. Each cell is the anomaly score ranking for anomalous customers with respect to normal customers averaged for all selection of anomalous customers and expressed in percentage. Table 3.5 shows that the proposed

Table 3.5: Performance comparison with [43], [48], and [63]

$\sum_t p_k^s(t)$ (kWh)	2	4	8	16	32	64	128
Case 1: disconnection of meters model							
FCM	49.65	50.30	48.94	47.75	42.75	38.64	34.34
SOM	47.07	44.69	42.19	38.18	30.90	21.12	16.37
RMT	49.22	46.10	40.06	30.42	20.70	15.03	10.95
MLM	**13.29**	**7.07**	**4.20**	**2.30**	**1.29**	**0.85**	**0.73**
Case 2: constant bypassing model							
FCM	50.49	49.49	49.48	48.40	45.10	40.59	36.94
SOM	49.86	49.51	48.38	44.59	34.21	22.33	18.97
RMT	51.23	51.09	50.87	50.60	48.92	41.35	26.99
MLM	**40.54**	**32.23**	**19.95**	**8.55**	**2.73**	**1.21**	**0.91**
Case 3: random uniform bypassing model							
FCM	50.11	49.51	48.93	48.35	44.96	41.02	38.52
SOM	49.83	49.43	47.93	43.38	33.46	24.55	21.07
RMT	50.95	51.17	50.74	49.73	45.04	34.88	22.79
MLM	**40.15**	**31.00**	**17.96**	**7.10**	**2.39**	**1.29**	**0.99**
Case 4: constant percentage bypassing model							
FCM	50.32	50.33	49.60	48.89	46.27	41.55	36.09
SOM	50.33	50.53	50.71	50.08	45.17	28.78	19.36
RMT	51.27	51.19	51.33	50.90	47.62	33.29	16.20
MLM	**44.30**	**38.93**	**28.65**	**15.07**	**5.15**	**1.43**	**0.83**

method beats the modified existing techniques in all cases. For all four methods, the rankings of the anomaly scores decrease in response to increasing level of anomaly. However, only the proposed method consistently ranks the anomalous customers at the top.

3.6 Summary

This chapter developed a physically inspired data-driven algorithm for electricity theft detection. The proposed algorithm leverages an approximate linear relationship between the power consumption and voltage data of customers on the same secondary. The proposed modified linear model produces accurate estimates of the electricity consumption for the majority of the customers. The modified linear model is able to detect inconsistencies among smart meter measurements of a group of customers from the same distribution secondary thereby identifying electricity thefts. An evaluation of the proposed electricity theft detection algorithm was then performed with real-world smart meter data and synthesized electricity theft cases. The evaluation results show that the proposed anomaly score developed in this chapter is effective in identifying electricity theft cases even when the amount of stolen electricity is small. The method was compared with existing unsupervised electricity theft detection techniques. The comparison results show that the proposed method is more effective in identifying the electricity thefts.

Chapter 4

Model-based Distributed Distribution Network Reconfiguration

4.1 Introduction

Distribution network reconfiguration (DNR) is an advanced smart grid technology. It works by changing the status of remotely controllable switching devices [86] to optimize certain operational objectives while satisfying operational constraints, including the voltage magnitude/line flow limit and network radiality. With the increasing penetration of remotely controllable switches and distributed generations (DGs), DNR [87] became critical in increasing the hosting capacity of distributed energy resources (DERs) [88], minimizing the curtailment of DGs [89], and reducing network line losses [90]. Meanwhile, both federal sponsored programs and market forces are facilitating the wide-spread adoption of smart grid technologies such as the advanced metering infrastructure and remote controllable switches [91]. These two technologies enabled remote data collection and actuation of loads and switches which are critical to the implementation of distribution network reconfiguration.

DNR is typically formulated as a mixed-integer programming (MIP) problem, where the integer variables represent the status of remotely controllable switches. This problem is usually solved in a centralized approach. In this dissertation work, we propose a distributed algorithm to overcome the communication bottleneck problem in the centralized approach by distributing the computation task among the network switches (agents) with only neighbor-to-neighbor communications. Our contributions are as follows. First, a novel decomposed formulation of the distribution network reconfiguration problem is developed. Second, an alternating direction method of multipliers (ADMM) release and fix algorithm is adopted to solve the problem in a distributed manner. Third, we introduce a distributed approximated Newton's method to speed up the distributed optimization algorithm.

The rest of the chapter is organized as follows. Section 4.2 reviews the existing literature on distribution network reconfiguration. Section 4.3 formulates the distribution network reconfiguration problem. Section 4.4 presents the distributed algorithm. Section 4.5 shows the simulation results. Section 4.6 provides the summary.

4.2 Prior Work

The existing literature on distribution network reconfiguration can be divided into two groups based on the solution methodology. The first group adopts heuristic methods within which there are two approaches. The first approach starts with a meshed network and then open the switch that will contribute the most to the objective function [92] [93] [94] [95]. The procedure continues until a radial network is achieved. The second approach starts from a radial network and selects a pair of closed and open switches and exchange their status [96] [97]. Selecting such a pair requires an accurate estimation of loss reduction due to the exchange.

The second group of literature formulates the distribution network reconfiguration problem as a mixed-integer program or a combinatorial optimization problem. The optimization problem is solved by either general-purpose metaheuristic algorithms or deterministic ones. Metaheuristic algorithms such as simulated annealing [98], genetic algorithm [99], and ant colony algorithm [100] have been used to solve the network reconfiguration problem. The deterministic algorithms work by linearizing or convexifying the original problem and converting it to a mixed-integer linear or convex optimization problem. Then mixed-integer linear or mix-integer convex optimization algorithms are adopted to solve the problem [87] [101] [102] [90]. The deterministic approach has a number of advantages such as the repeatability of solutions, guarantees of global optimality, and ease of implementation thanks to the optimization solvers.

Most of the existing methods followed the centralized control framework within which all network data are collected and sent back to the control center to determine the reconfiguration solution. The switch control signals are then sent from the control center via the communication network for switch actuation. Although centralized approaches have shown good numerical performance on some distribution test feeders,they usually result in high latency and communication bottleneck in the system. The distributed approaches on the other hand have great potential in reducing the communication burden, improving cybersecurity and preserving the privacy of smart meter data [103].

4.3 Problem Formulation

One of the most commonly used objectives of the network reconfiguration problem is the minimization of line losses. The constraints of the optimization problem include the operating limits such as the line flow limits, the power flow constraints, and the network radiality constraint. The power flow constraints ensure that the steady-state operating conditions are consistent with the electric loads, distributed generations and the physics of the distribution network. The network radiality constraints require that every primary feeder of the distribution network have a radial topology. The goal of network reconfiguration is to find on/off status for all switches that minimize the network losses while satisfying all operating constraints. In this work, the distribution network is assumed to be reasonably balanced so that the single-phase representation of the three-phase network is acceptable. It is also assumed that each line segment has a switch installed which can be remotely controlled for network reconfiguration.

The objective function of line loss minimization is given by (4.1) where r_{ij} is the resistance of line ij. l_{ij}^2 denotes the squared magnitude of current flowing on line ij. E is the set of all lines in the network. Note that each line ij has a reference direction $i \to j$ associated with it. Throughout this chapter we use ij to denote a line if the reference direction is needed; otherwise we will simply use ℓ in place of ij.

$$\min \sum_{ij \in E} r_{ij} l_{ij}^2 \tag{4.1}$$

Two operating limits will be considered in the problem formulation, namely the nodal voltage magnitude limit (4.2) and branch flow limit (4.3):

$$V^{2\min} \leq v_i^2 \leq V^{2\max} \quad \forall i \in N \setminus N^0 \tag{4.2}$$

$$l_{ij}^2 \leq \alpha_\ell I^{2\max} \quad \forall ij \in E \tag{4.3}$$

where v_i^2 is the squared nodal voltage magnitude of node i; $\alpha_\ell \in \{0, 1\}$ is a binary variable representing the close ($\alpha_\ell = 1$) and open ($\alpha_\ell = 0$) status of each switch; N denotes the set of all nodes in the distribution network; N^0 is the set of substation nodes (reference nodes).

The DistFlow equations [90] are adopted to capture the power flow constraints.

$$P_i = \sum_{ij \in E} p_{ij} - \sum_{ki \in E} (p_{ki} - r_{ki} l_{ki}^2) + g_i v_i^2 \qquad \forall i \in N \setminus N^0 \tag{4.4}$$

$$Q_i = \sum_{ij \in E} q_{ij} - \sum_{ki \in E} (q_{ki} - x_{ki} l_{ki}^2) + b_i v_i^2 \qquad \forall i \in N \setminus N^0 \tag{4.5}$$

$$v_j^2 = v_i^2 - 2 r_{ij} p_{ij} - 2 x_{ij} q_{ij} + (r_{ij}^2 + x_{ij}^2) l_{ij}^2 \qquad \forall ij \in E \tag{4.6}$$

$$l_{ij}^2 = \frac{p_{ij}^2 + q_{ij}^2}{v_i^2} \qquad \forall ij \in E \tag{4.7}$$

$$v_i^2 = v^{\mathrm{ref}2} \qquad \forall i \in N^0 \tag{4.8}$$

where $P_i + jQ_i$ is the complex net power injection at node i; $p_{ij} + jq_{ij}$ is the complex branch power flow of line ij; $r_{ij} + jx_{ij}$ is the impedance of line ij; $g_i + jb_i$ is the shunt admittance from bus i to ground. It has been shown [104] that for practical radial networks, the system of equations (4.4)-(4.8) has a unique solution near the flat voltage profile. Therefore, they are suffice for the reconfiguration application. Since (4.7) defines a non-convex feasible set, the relaxation is typically applied [90]:

$$l_{ij}^2 \geq \frac{p_{ij}^2 + q_{ij}^2}{v_i^2} \qquad \forall ij \in E \tag{4.9}$$

Note that (4.9) defines a quadratic cone and can be handled by many optimization solvers.

To enforce the network radiality in the reconfiguration problem, the method proposed in [87] [101] is adopted:

$$\beta_{ij} + \beta_{ji} = \alpha_\ell \quad \forall \ell \in E \tag{4.10}$$

$$\beta_{ij} = 0 \qquad \forall i \in N^0 \; j \in N(i) \tag{4.11}$$

$$\sum_{j \in N(i)} \beta_{ij} = 1 \quad \forall i \in N \setminus N^0 \tag{4.12}$$

$$\beta_{ij} \in \{0, 1\} \qquad \forall i \in N \setminus N^0 \; j \in N(i) \tag{4.13}$$

$$0 \leq \alpha_\ell \leq 1 \qquad \forall \ell \in E \tag{4.14}$$

where β_{ij}, β_{ji}, α_ℓ are variables associated with each line ij; $N(i)$ is the set of neighbor nodes of i. It has been shown [101] that (4.10)-(4.14) are sufficient for the radiality for each graph component that is connected to one of the reference nodes. Although it does not imply that the entire feeder is radial, for practical feeders each load bus absorbs certain amount of power from at least one of the substations. Therefore the feeder must be connected. Consequently (4.10)-(4.14) defines a radial topology for the entire feeder.

The DistFlow equations (4.4)-(4.8) needs to incorporate the possible change of topologies dictated by (4.10)-(4.14). In particular, if line ℓ is disconnected then the line current must be zero. This is already enforced by (4.3). Also, if line ℓ is disconnected, there won't be the end voltage relationship described by (4.6). We use the big-M method to correct this constraint:

$$v_j^2 \leq M(1-\alpha_\ell) + v_i^2 - 2r_{ij}p_{ij} - 2x_{ij}q_{ij} + (r_{ij}^2 + x_{ij}^2)l_{ij}^2 \tag{4.15}$$

$$v_j^2 \geq -M(1-\alpha_\ell) + v_i^2 - 2r_{ij}p_{ij} - 2x_{ij}q_{ij} + (r_{ij}^2 + x_{ij}^2)l_{ij}^2 \tag{4.16}$$

for all $ij \in E$. M is a number big enough to free the relationship between v_j^2 and v_i^2 when line ℓ is not connected ($\alpha_\ell = 0$).

We summarize the final optimization problem as:

$$\begin{aligned} \min \quad & \text{Network loss: } (4.1) \\ \text{s.t.} \quad & \text{Operating limits: } (4.2), (4.3) \\ & \text{Power flow: } (4.4), (4.5), (4.15), (4.16), (4.7), (4.8) \\ & \text{Network radiality: } (4.10) - (4.14) \end{aligned} \tag{4.17}$$

The decision variables are $p_{ij}, q_{ij}, l_{ij}^2, \beta_{ij}, \beta_{ji}, \alpha_\ell \; \forall ij \in E$; $v_i^2 \; \forall i \in N$. Problem (4.17) is a mixed-integer conic programming problem and can be solved by existing solvers in a centralized manner. In the next section, we decompose (4.17) into a distributed formulation, and propose a Newtom/ADMM distributed algorithm.

4.4 Distributed Solution Methodology

In this section, we propose a distributed solution to problem (4.17). First, (4.17) will be decomposed into a collection of coupled sub-problems. Second, each of the sub-problems is solved by an agent (switch) via local computation and neighbor-to-neighbor communication. In the following, we first define the agents and their communication graph, then we present the two-step distributed algorithm.

4.4.1 Definition of Agents and Communication Graph

We assume each switch has computing capability and can communicate with its neighbors. The agents are defined as the switches in the network. It is assume that each line has a switch. Hence, we do not distinguish the concept of switch, line, and agent and refer to them as agent in the rest of the chapter.

We define the neighbors $E(ij)$ of each agent ij as the agents that have a node in common with agent ij. ij itself is not in $E(ij)$. In other words, let $G = (N, E)$ be the graph representing the distribution network, then the communication graph is $G^c = (E, M)$ where if $\ell \in E$ and $m \in E$, then $\ell m \in M$ if ℓ and m are incident in N. In graph-theoretic terms, G^c is called the line graph of G.

An agent ij's set of neighbors $E(ij)$ is partitioned into four subsets based on their and agent ij's reference directions. Denote $E_i^{from}(ij)$ as the neighbors that connect to node i

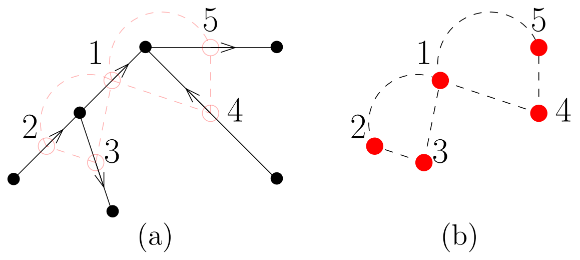


Figure 4.1: Example of agents and their communications.

with their “from” nodes being i. $E_i^{to}(ij)$, $E_j^{from}(ij)$, $E_j^{to}(ij)$ are defined in a similar manner. Figure 4.1 shows an example illustrating the concepts mentioned in this section. There are five switch agents in Figure 4.1 (a). The arrows denote the reference directions. Figure 4.1 (b) shows the communication graph of the network. According to the reference direction, $E(1) = \{2, 3, 4, 5\}$ can be partitioned into $E_i^{to}(1) = \{2\}$, $E_i^{from}(1) = \{3\}$, $E_j^{to}(1) = \{4\}$, and $E_j^{from}(1) = \{5\}$.

4.4.2 ADMM Release-and-Fix

This subsection describes the distributed solution to problem (4.17). The state vector of agent ij is defined as $\mathbf{x}_{ij} = [p_{ij}, q_{ij}, l_{ij}^2, v_i^{2(ij)}, v_j^{2(ij)}, \beta_{ij}, \beta_{ji}, \alpha_\ell]^T$.

$$v_i^{2(ij)} = v_i^{2(ik)} \quad \forall ij \in E, \forall ik \in E_i^{from}(ij) \tag{4.18}$$

$$v_i^{2(ij)} = v_i^{2(ki)} \quad \forall ij \in E, \forall ki \in E_i^{to}(ij) \tag{4.19}$$

$$v_j^{2(ij)} = v_j^{2(jk)} \quad \forall ij \in E, \forall jk \in E_j^{from}(ij) \tag{4.20}$$

$$v_j^{2(ij)} = v_j^{2(kj)} \quad \forall ij \in E, \forall kj \in E_j^{to}(ij) \tag{4.21}$$

The branch variables are assigned for each agent; the nodal variables (the voltages) have superscripts (ij) associated with them. This is because the same voltage variable v_i^2 is shared by all agents that are incident to node i and must be distinguished. As a result, agents must agree on the value of shared voltage variables, as shown in (4.18)-(4.21):

Using the definition of $\mathbf{x}_{ij}$, problem (4.17) can be written as:

$$\begin{aligned} \min_{\mathbf{x}_{ij}, ij \in E} \quad & \sum_{ij \in E} \mathbf{c}_{ij}^T \mathbf{x}_{ij} \\ \text{s.t.} \quad & \mathbf{x}_{ij} \in \mathbb{X}_{ij} \quad \forall ij \in E \\ & \sum_{j \in N(i)} \mathbf{A}_{ij} \mathbf{x}_{ij} = \mathbf{b}_i \quad \forall i \in N \end{aligned} \tag{4.22}$$

where $\mathbb{X}_{ij}$ is a local mixed-integer set whose continuous relaxation is convex; $N(i)$ is the set of neighbor nodes of i; the matrices $\mathbf{A}_{ij}$ and vectors $\mathbf{b}_i$ are identified through problem (4.17) as well as (4.18)-(4.21). We refer the first set of constraints in (4.22) as local constraints and the second the coupling constraints. To solve problem (4.22) in a distributed manner, we derive the augmented Lagrangian function by absorbing the coupling constraints into the objective function:

$$\begin{aligned} \min_{\mathbf{x}_{ij}, ij \in E} \quad & L_\rho = \sum_{ij \in E} \mathbf{c}_{ij}^T \mathbf{x}_{ij} + \sum_{i \in N} \boldsymbol{\mu}_i^T \left(\sum_{j \in N(i)} \mathbf{A}_{ij} \mathbf{x}_{ij} - \mathbf{b}_i \right) + \frac{\rho}{2} \sum_{i \in N} \left\| \sum_{j \in N(i)} \mathbf{A}_{ij} \mathbf{x}_{ij} - \mathbf{b}_i \right\|_2^2 \\ \text{s.t.} \quad & \mathbf{x}_{ij} \in \mathbb{X}_{ij} \quad \forall ij \in E \end{aligned} \tag{4.23}$$

where $\rho > 0$ is called the penalty parameter. Problem (4.23) may be solved in a distributed manner by the alternating direction method of multipliers (ADMM) [105] [106] [107].

The ADMM algorithm can be used to solve convex separable problems. However, the presence of binary variables β_{ij}, β_{ji} in (4.23) destroys the convexity. A heuristic remedy to handle binary variables was introduced in [105] and the resulting modified algorithm is called ADMM Release-and-Fix. The modified algorithm proceeds by iterating between two stages. The first stage (ADMM-Release) is identical to conventional ADMM with the exception of the presence of binary variables. The goal of ADMM-Release is to search for feasible binary solutions, which are "stable" across multiple runs of ADMM-Release. In order to encourage exploration of new binary solutions, the penalty parameter ρ will be gradually decreased after a feasible solution is found. In order to force convergence to a "stable" solution ρ will gradually increase. After a new stable binary solution is found, the second stage (ADMM-Fix) fixes the binary solution from ADMM-Release and solves the simplified optimization problem with only continuous variables. These two stages will alternate until the stopping criteria is met.

The ADMM-Fix step converges slowly. To speed up the distributed computation we propose an approximated Newton's method to replace the ADMM-Fix step.

4.4.3 Approximated Newton's Method

After a feasible binary solution is found by ADMM-Release, the ADMM-Fix problem becomes the same as solving the DistFlow equations (4.4)-(4.8) in a distributed manner with a given network configuration. We first linearize (4.7) of the DistFlow equations as:

$$2p_{ij}^{\nu}\tilde{p}_{ij} + 2q_{ij}^{\nu}\tilde{q}_{ij} - v_i^{2\nu}\tilde{l}_{ij}^2 - l_{ij}^{2\nu}\tilde{v}_i^2 = p_{ij}^{2\nu} + q_{ij}^{2\nu} - l_{ij}^{2\nu}v_i^{2\nu} \quad \forall ij \in E \tag{4.24}$$

where ν is the iteration number and variables with a tilde $\tilde{}$ denotes the increment, e.g. $\tilde{p}_{ij} = p_{ij}^{\nu} - p_{ij}$. The resulting system of linear equations, namely (4.4) (4.5) (4.6) (4.8), and (4.24)is denoted as $\mathbf{Ax} = \mathbf{b}$. Next, we propose a distributed algorithm which solve this linear system in an iterative manner.

We define a new vector $\mathbf{x}_{ij}^c = [p_{ij}, q_{ij}, l_{ij}^2, v_i^{2(ij)}, v_j^{2(ij)}]^T$ with continuous variables only. Solving $\mathbf{Ax} = \mathbf{b}$ is equivalent to solving the following unconstrained optimization problem [108]:

$$\min_{\mathbf{x}_{ij}^c, ij \in E} \quad f = \frac{1}{2} \sum_{ij \in E} \left\| \mathbf{A}_{ij}^c \mathbf{x}_{ij}^c - \mathbf{b}_{ij}^c \right\|_2^2 \tag{4.25}$$

where $\mathbf{A}_{ij}^c$ and $\mathbf{b}_{ij}^c$ are identified from $\mathbf{A}$ and $\mathbf{b}$ by rearranging equations and variables accordingly and appending (4.18)-(4.21) to enforce voltage constraints. Note that $\mathbf{A}_{ij}^c$ is a constant matrix while $\mathbf{b}_{ij}^c$ depends linearly on $\mathbf{x}_m^c$ for all neighbors m of agent ij (excluding ij itself). We would like to solve problem (4.25) using Newton's iteration where the gradient and Hessian matrix can be derived as follows:

$$\nabla_\ell f = \mathbf{A}_\ell^{cT} (\mathbf{A}_\ell^c \mathbf{x}_\ell^c - \mathbf{b}_\ell^c) \quad \forall \ell \in E \tag{4.26}$$

$$\mathbf{H}_{\ell\ell} = \frac{\partial}{\partial \mathbf{x}_\ell^c} \nabla_\ell f = \mathbf{A}_\ell^{cT} \mathbf{A}_\ell^c \quad \forall \ell \in E \tag{4.27}$$

$$\mathbf{H}_{\ell m} = \frac{\partial}{\partial \mathbf{x}_m^c} \nabla_\ell f = -\mathbf{A}_\ell^{cT} \frac{\partial}{\partial \mathbf{x}_m^c} \mathbf{b}_\ell^c \quad \forall \ell, m \in E, m \in E(\ell) \tag{4.28}$$

$$\mathbf{H}_{\ell m} = \mathbf{0} \quad \forall \ell, m \in E, m \notin E(\ell) \tag{4.29}$$

However, it is challenging to invert the Hessian matrix $\mathbf{H}$ of the objective function in a distributed manner. Therefore, an method to approximate the inverse Hessian is needed. To do so, let's define two new matrices: $\mathbf{D} = \text{diag}(\mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_{|E|})$ where $\mathbf{D}_\ell = \gamma \mathbf{H}_{\ell\ell}$;

and $\mathbf{B}$ where $\mathbf{B}_{\ell\ell} = (1-\gamma)\mathbf{H}_{\ell\ell}$ and $\mathbf{B}_{\ell m} = \mathbf{H}_{\ell m}$. With these two ancillary matrices, we can approximate $\mathbf{H}^{-1}$ as follows [108]:

$$\begin{aligned}
\mathbf{H}^{-1} &= (\mathbf{D}+\mathbf{B})^{-1} \\
&= \mathbf{D}^{-\frac{1}{2}}(\mathbf{I}+\mathbf{D}^{-\frac{1}{2}}\mathbf{B}\mathbf{D}^{-\frac{1}{2}})^{-1}\mathbf{D}^{-\frac{1}{2}} \\
&\approx \mathbf{D}^{-\frac{1}{2}}(\mathbf{I}-\mathbf{D}^{-\frac{1}{2}}\mathbf{B}\mathbf{D}^{-\frac{1}{2}})\mathbf{D}^{-\frac{1}{2}} \\
&= \mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{B}\mathbf{D}^{-1} \qquad (4.30)
\end{aligned}$$

where the third equation with the approximation sign is analogous to the first order Taylor series expansion $\frac{1}{1+x} \approx 1-x$ near $x=0$. Note than (4.30) enables the computation of $\mathbf{H}^{-1}$ and the update of local variables to be carried out (approximately) locally as follows:

$$\mathbf{x}_{ij}^{c} \leftarrow \mathbf{x}_{ij}^{c} - \mathbf{D}_{\ell}^{-1}\nabla_{\ell}f + \mathbf{D}_{\ell}^{-1}\sum_{m\in E(\ell)\cup\ell}\mathbf{B}_{\ell m}\mathbf{D}_{m}^{-1}\nabla_{m}f \qquad (4.31)$$

Since $\mathbf{B}_{\ell m} = \mathbf{0}$ if $m \neq \ell$, $m \notin E(\ell)$, the computation of each term in (4.31) requires only the information of agent ij and its neighbors. In summary, the approximated Newton's method has two levels of iterations. In the outer iteration, problem (4.25) is formed by obtaining $\mathbf{A}_{ij}^{c}$, $\mathbf{b}_{ij}^{c}$, and $\mathbf{x}_{ij}^{c}$ from previous iteration; in the inner iteration, variables are updated using (4.31).

4.5 Simulation Results

This section presents a simulation study to validate the proposed distributed algorithm for network reconfiguration. We first describe the test system and then discuss the results from both centralized and the proposed distributed algorithm. In particular, the computation speed of the ADMM algorithm and our proposed approximated Newton's method is compared.

The 16-bus distribution test feeder described in [109] is used in the simulation and is shown in Figure 4.2, where the dots represent load buses; the solid lines represent

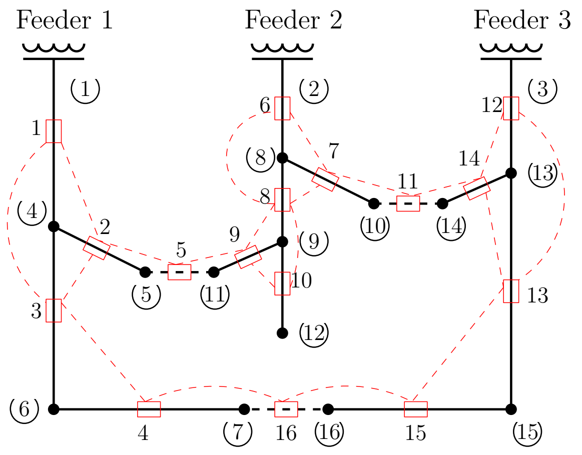


Figure 4.2: 16-bus test feeder and the agents

sectionalizing switches and dashed lines represent tie switches. Agents are represented by a red box. The edges of the communication graph are represented by red dashed lines.

Initially, all sectionalizing switches are closed and all tie switches are open. The global optimum solution found by the centralized algorithm was reported in [109]. The network reconfiguration results of the proposed distributed algorithm and the centralize one are shown in Table 4.1. It can be seen that the proposed method found the same global optimum solution as that of the centralized algorithm.

In order to evaluate the computation speed of the ADMM algorithm and our proposed approximated Newton's method, we conducted testing using 20 different radial

Table 4.1: Reconfiguration results

	Original configuration	Centralized method (MICP)	Proposed method
Opened switches	5,11,16	7,9,16	7,9,16
Power loss (kW)	511.4	466.1	466.1
Loss reduction	-	8.85%	8.85%
Voltage magnitude (p.u.)	Vmax=1.000 (Bus 1,2,3) Vmin=0.969 (Bus 12)	Vmax=1.000 (Bus 1,2,3) Vmin=0.972 (Bus 12)	Vmax=1.000 (Bus 1,2,3) Vmin=0.972 (Bus 12)

network configurations of the test system. The tunable parameters are $\rho = 1$ for ADMM and $\gamma = 1.5$ for the approximated Newton's method. To make a fair comparison, both of the algorithms terminate when the solutions reach the same level of accuracy. The computation time are reported in Table 4.2. As shown in the table, the proposed approximated Newton's method achieves roughly 5 times speed up compared to the ADMM.

Table 4.2: Computation time of ADMM and approximated Newton's method

	ADMM	Approximated Newton
Min (second)	3.87	0.64
Max (second)	18.32	7.66
Average (second)	10.41	2.05

4.6 Summary

This chapter presents a distributed algorithm to solve the distribution network reconfiguration problem. The proposed algorithm can be implemented on a group of switch agents in the distribution network, which work collaboratively via neighbor-to-neighbor communication to find the optimum network reconfiguration. The simulation results show that the distributed algorithm correctly finds the global optimum solution on a 16-bus distribution test system. In addition, the proposed approximated Newton's method dramatically improves the computation speed of the distributed algorithm.

Chapter 5

Deep Reinforcement Learning for Distribution Network Reconfiguration Part 1: Deep Q-Learning

5.1 Introduction

In the previous chapter, we established a model-based, distributed distribution network reconfiguration (DNR) framework. In the framework, a group of computational agents collectively determines the status of remotely controllable switches, resulting in an optimal *static* network configuration. That is, determining the network configuration which will stay the same for the entire study period. DNR can also be performed dynamically [88]

[19]. In dynamic DNR, the goal is to find a sequence of network configurations over time. We focus on the dynamic DNR in this and next chapter.

As discussed in the previous chapter, DNR is typically solved by model-based control approaches. However, one challenge with these physical model based approaches is that the uncertain or incomplete distribution network parameters makes practical implementations difficult. In addition, for the dynamic DNR, the problem size is typically much larger than the static ones. For the switch statuses of multiple time steps need to be identified. Furthermore, it is more difficult to handle uncertainties associated with loads and DGs in dynamic DNR problems.

To address these limitations, we use a data-driven approach to formulate the dynamic DNR as a reinforcement learning (RL) problem. In a typical RL setup, an agent tries to learn an optimal control policy by interacting with the real physical environment or a simulated one. However, it is costly and time consuming for the agent to learn an optimal network reconfiguration strategy by directly interacting with the physical distribution network. Furthermore, it is difficult to create a reliable simulated environment when the network parameters are inaccurate. Thus, it is desirable for the agent to learn from the historical network reconfiguration data collected by the electric utilities.

In this dissertation work, we develop two RL algorithms to solve the dynamic DNR problem. Both of the algorithms are data-driven and are capable of learning a control policy from a finite historical operational dataset. For the ease of presentation, we treat each algorithm in a separate chapter. This chapter discuss the first algorithm based on deep Q learning (DQL), and is used to solve small-scale network reconfiguration problem. An improved algorithm that handles much larger networks is the subject of the next chapter. To address the low sample efficiency issue with DQL, we propose to augment past grid operational experiences with synthetic ones to construct additional training data. Simulation results on a 16-bus distribution feeder reveal that the proposed deep RL is capable of finding a decent control policy without using the network parameter information. The proposed operational experience augmentation technique further improves the performance of DQL.

The rest of this chapter is organized as follows: Section 5.2 reviews prior work on dynamic distribution network reconfiguration. Section 5.3 formulates the distribution network dynamic reconfiguration problem. Section 5.4 presents the proposed reinforcement learning algorithm. Section 5.5 shows the simulation results. Section 5.6 summarizes this chapter.

5.2 Prior Work

The existing literature on dynamic DNR can be categorized into three groups: the mixed-integer programming based approaches, the heuristic or meta-heuristic algorithms, and dynamic programming methods.

The first group of literature utilizes mixed-integer programming framework to formulate the dynamic DNR as a deterministic, stochastic, or robust optimization problem. Deterministic optimization formulations do not take stochastic power injections into consideration. The optimization methods used to solve deterministic problems include mixed-integer linear programming (MILP) [110] [111], mixed-integer conic programming (MICP)

[90], mixed-integer nonlinear programming (MINLP) [112], and MIP combined with other problem size reduction heuristics [88]. Unlike deterministic optimization, stochastic and robust optimization methods fully incorporate the uncertainties of loads and DGs into the problem formulation. Robust optimization methods are developed to find the reconfiguration with optimal performance in the worst-case scenario [113] [114] [115] and simultaneously identify the critical switch [89]. Stochastic optimization methods are developed to optimize the expected control objective [116], or incorporate the uncertainties of the loads and DGs by combing MILP with unscented transforms [117].

The second group of literature uses heuristics or meta-heuristic algorithms. The minimum spanning tree [118] and the branch exchange [119] methods are used to heuristically solve the dynamic DNR problem. Meta-heuristic algorithms such as genetic algorithm [120], fuzzy adaptive inference-based particle swarm optimization (PSO) [121], and a hybrid PSO with time-partitioning [122] have been adopted to identify the optimal network configurations.

The third group of literature leverages dynamic programming (DP) methods [123] to determine the optimal sequence of network configurations. This approach first identifies the set of radial configurations and treats them as the states. It then applies the DP backward iteration [124] to determine the optimal sequence of hourly network configurations.

Most of the existing literature uses a physical model-based control approach to solve the dynamic DNR problem. However, this approach has two limitations. First, model-based algorithms may not be reliable when electric utilities do not have complete and accurate distribution network parameters. It is well known that it is difficult for electric utilities to maintain accurate primary and secondary feeders' parameters for distribution networks covering millions of nodes [17]. Second, the computation time for model-based control algorithms increases exponentially with the number of remotely controllable switches, the number of DERs, and the length of the operation horizon, which makes it difficult to apply in real-time network reconfiguration.

5.3 Problem Formulation

In this section, we present our formulation for the dynamic DNR problem as a Markov decision process (MDP) [125]. First we review the preliminaries of MDPs. Next we describe the dynamic DNR problem as an MDP. Finally we state the set up of the reinforcement learning problem for DNR.

5.3.1 Basics of Markov Decision Process

An MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, r, \gamma, T)$ consists of a state space $\mathcal{S}$, an action space $\mathcal{A}$, a state transition probability $P(s'|s,a)$ $\forall s', s \in \mathcal{S}, \forall a \in \mathcal{A}$, a reward function $r(s,a) : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, $\forall s \in \mathcal{S}$, $\forall a \in \mathcal{A}$, a discount factor $\gamma \in [0,1)$, and a time horizon T. In an MDP, an agent selects an action $A_t \in \mathcal{A}$ based on the environment's state $S_t \in \mathcal{S}$ at each discrete time step t. Then the agent receives a reward $R_{t+1} = r(S_t, A_t)$ and the environment's state transitions to S_{t+1} according to the state transition probability $P(S_{t+1}|S_t, A_t)$. The process either terminates when $t = |T|$ if T is finite or continues indefinitely if T is infinite.

The goal of the agent is to find a control policy π that maximizes the expected discounted return $J(\pi) = \mathbb{E}_{\tau \sim \pi}[G(\tau)]$, where control policy $\pi(\cdot|s)$ maps each state to an action selection probability distribution over the action space $\mathcal{A}$. τ is a trajectory or sequence of states and actions, $\{S_0, A_0, S_1, A_1, ..., S_{T-1}, A_{T-1}, S_T\}$. $G(\tau)$ is the discounted return along a trajectory. $G(\tau) = \sum_{t=0}^{T} \gamma^t R_{t+1}$.

Finally, we define two important value functions, the state-value function $v_\pi(s)$ and the action-value function $q_\pi(s,a)$ with respect to the control policy π:

$$v_\pi(s) = \mathop{\mathbb{E}}_{\tau \sim \pi} \left[\sum_{k=0}^{T} \gamma^k R_{t+k+1} | S_t = s \right] \quad \forall s \tag{5.1}$$

$$q_\pi(s,a) = \mathop{\mathbb{E}}_{\tau \sim \pi} \left[\sum_{k=0}^{T} \gamma^k R_{t+k+1} | S_t = s, A_t = a \right] \forall s, a \tag{5.2}$$

where $v_\pi(s)$ and $q_\pi(s,a)$ represent the expected discounted return starting from state s or state-action pair (s,a), and following control policy π thereafter. Next we formulate the dynamic DNR problem as an MDP.

5.3.2 Formulate Dynamic DNR as an MDP

We first introduce some notations for the dynamic DNR. Consider a distribution network with n load nodes and n^0 substations. Let v_{it}, p_{it}, and q_{it} be the nodal voltage magnitude, real and reactive power injections of node i at time t. Define vectors p_t and q_t as $p_t = [p_{1t}, p_{2t}, \cdots, p_{nt}]$ and $q_t = [q_{1t}, q_{2t}, \cdots, q_{nt}]$. Let p_t^l be the network's total real line losses at t. We denote a radial configuration of the distribution network at time t by a_t^r. That is, a_t^r represents a rooted spanning forest of the graph associated with the no-shunt distribution network [124]. Each root corresponds to a substation.

Next, we construct the dynamic DNR problem as an MDP as follows. We define the state at time t to be $S_t = [p_t, q_t, a_{t-1}^r, t]$ and the action A_t as changing the topology of the network to a^r_t. Therefore, $\mathcal{S}$ consists of the set of all injection patterns together with the set of all possible radial configurations. The latter is also equal to $\mathcal{A}$. The reward function reflects both the network loss and the switching cost and is defined as

$$r(S_t, A_t = a^r_t) = -C^l(p_t, q_t, a^r_t) - C^s(a^r_{t-1}, a^r_t) \tag{5.3}$$

where C^l is the cost associated with the network loss and C^s is the cost incurred by the change of network configuration. The detailed formulation of C^l and C^s will be shown in Section 5.5. The expected discounted return $E\left[\sum_{t=0}^{T} \gamma^t r(S_t, A_t)\right]$ with some initial configuration a^r_{-1} for the dynamic DNR problem includes both network losses and switching costs. This completes the construction of the MDP.

Note that the injection patterns p_t and q_t time series might not be strictly Markovian. Nevertheless, we shall still use this definition of S_t because the algorithms that we will be discussing are still applicable even if the Markovian property is slightly violated in practice [125].

During the distribution network reconfiguration process, we need to ensure that the nodal voltages always stay within allowable range. In the MDP framework, physical constraints are typically modeled via a constraint function $V^\pi_C(s) = E_\pi\left[\sum_{t=0}^{T} \gamma^t C_{t+1} | S_0 = s\right]$, where $C_{t+1} = c(S_t, A_t)$ is the amount of constraint violation at time t. We define $c(S_t, A_t)$ to be the sum of absolute value of voltage violations at all metered nodes:

$$c(S_t, A_t) = \sum_{i \in N^v} [\max(0, v_{it} - \bar{v}) + \max(0, \underline{v} - v_{it})] \tag{5.4}$$

where N^v is the set of all nodes that have voltage measurement devices; $\bar{v}$ and $\underline{v}$ are the upper and lower bounds for voltage. Now the dynamic DNR can be formulated as a constrained MDP problem:

$$\max_{\pi} \quad V^{\pi}(s) \quad \text{s.t. } V_C^{\pi}(s) \leq 0 \tag{5.5}$$

The Lagrangian of (5.5) is

$$V^{\pi}(s) - \lambda V_C^{\pi}(s) = E_{\pi}\left[\sum_{t=0}^{T} \gamma^t (R_{t+1} - \lambda C_{t+1})\right] \tag{5.6}$$

Although on-policy RL algorithms such as constrained policy optimization [126] have been developed to solve the constrained MDP problem, their sample efficiency is much lower than that of the off-policy algorithms. In this chapter, we approximately solve (5.5) by replacing the original reward function in the (unconstrained) MDP by an augmented reward function $r(S_t, A_t, \lambda) \doteq r(S_t, A_t) - \lambda c(S_t, A_t)$. The multiplier $\lambda \geq 0$ can be estimated if an estimation of $V_C^{\pi}(s)$ is available. Nevertheless, we shall use a fixed λ in this initial study.

5.4 Technical Methods

In this section, we review basic concepts of deep reinforcement learning and describe an algorithm to find radial configurations. Finally, we present a noval operational experience data generation algorithm.

5.4.1 Deep Q-Learning

In this subsection, an off-policy RL algorithm will be developed to solve the dynamic DNR problem. An off-policy RL algorithm is more suitable than an on-policy one for the following two reasons. First, an off-policy learner allows an agent to learn the value of optimal policy independently of the actions took by the agent. Thus, off-policy RL algorithms enable distribution operators to learn from a wealth of historical network reconfiguration operation data. Second, off-policy RL algorithms have much higher sample efficiency than that of the on-policy ones.

One of the most widely used off-policy RL algorithm for MDP problems is the Q-learning, which updates the action-value function iteratively:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \text{max}_a Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{5.7}$$

This allows the learned action-value function, Q, to converge to the optimal action-value function Q^* provided that all state-action pairs continue to be updated. Once the optimal action-value functions are learned, the optimal control policy $\pi^*(s)$, which maximizes $V^\pi(s)$, can be found by:

$$\pi^* : S_t \mapsto \text{argmax}_a\, Q^*(S_t, a) \tag{5.8}$$

However, it is infeasible to directly apply Q-learning for dynamic DNR problems. This is because even if we discretize the continuous state variables, the dimensionality of the state space still increases exponentially. To deal with high-dimensional state space and continuous state variables, we parameterize an approximate action-value function $Q(S_t, A_t; \theta^Q)$ with a neural network, where θ^Q are the parameters of the neural network.

Nonetheless, this brings its own challenges. Divergence may occur during learning [127]. One cause of divergence is the high correlations between the action values $Q(S_t, A_t)$ and the target values $R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$. To ease this, we adopt a target Q network whose parameters θ^{Q-} are only updated every C steps [128] by $\theta^{Q-} \leftarrow \theta^Q$. The θ^Q update uses the loss function $L(\theta^Q) \doteq E\left[(r + \gamma \max_{a'} Q(s', a'; \theta^{Q-}) - Q(s, a; \theta^Q))^2\right]$. To further reduce the high correlation, we adopt the replay mechanism [127]. As such, we store the past operational experiences for network reconfiguration $e_t = (S_t, A_t, R_{t+1}, S_{t+1})$ in a 'memory data set' $D_t \doteq \{e_1, ..., e_t\}$, which is sampled during learning. Each sample forms a replay in the learning process.

5.4.2 Finding Radial Configurations

When constructing the action domain, all feasible radial configurations need to be enumerated. For single substation distribution networks, this could be done by the tree enumeration algorithm [129, p.464]. For distribution networks with multiple substations, we enhance the algorithm by adding a merge and a split step. Figure 5.1 provides an example of this enhanced algorithm. First, we merge all the substation nodes (0 and 1) into a single root node X. Then we enumerate spanning trees on the resulting graph. Finally, we split the root node X by identifying the branch-node connectivity on the original graph. This algorithm guarantees that all the rooted spanning forests can be discovered. Due to the operational constraints, the agent must choose configurations that would lead to safe

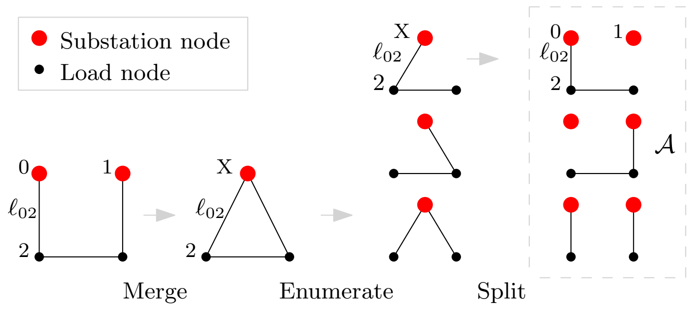


Figure 5.1: Rooted spanning forest enumeration process.

operation of the grid. As a result, many of the actions $a^r \in \mathcal{A}$ cannot be selected under certain injection pattern. Therefore, we reduce the action space $\mathcal{A}$ to include only those configurations that appeared in the historical operation data set. This allows the agent to avoid selecting unacceptable network configurations. However, this will limit the potential of discovering the optimal control policies.

5.4.3 Operational Experience Augmentation

One major drawback of the existing deep reinforcement learning algorithms is the poor sample efficiency. To improve the performance of our proposed deep Q-learning algorithm for dynamic DNR problem, we propose an innovative technique to generate reliable synthetic operational experience data from historical operational data set.

We propose a three-step algorithm to create a set of synthetic operational experiences $\tilde{D}_t \doteq \{\tilde{e}_1, ..., \tilde{e}_t\}$ where $\tilde{e}_t = (\tilde{S}_t = [\tilde{p}_t, \tilde{q}_t, \tilde{a}^r_{t-1}, t], \tilde{A}_t = \tilde{a}^r_t, \tilde{R}_{t+1} = r(\tilde{S}_t, \tilde{A}_t) + \lambda c(\tilde{S}_t, \tilde{A}_t), \tilde{S}_{t+1})$. The steps are 1) synthesizing the injection time series $\tilde{p}_t$ and $\tilde{q}_t$, 2) generating the network configuration at each time step $\tilde{a}^r_t$, and 3) estimating the corresponding reward values $r(\tilde{S}_t, \tilde{A}_t) + \lambda c(\tilde{S}_t, \tilde{A}_t)$ for the data created in steps 1 and 2. Step 1 takes the historical load time series and outputs a new one. For example, We can either directly use the historical injection data or train an load time series model using historical data [130]. In step 2, we generate a sample path $\{a_t^r\}$ from a stochastic process defined on the sample space $\mathcal{A}$. In step 3, we estimate $r(\tilde{S}_t, \tilde{A}_t)$ and $c(\tilde{S}_t, \tilde{A}_t)$ for each time step t. The algorithms for estimating the network losses and voltage magnitudes are described below.

Two sets of regression models are trained on the historical data to estimate total network loss and nodal voltage magnitudes, respectively. For both sets of regression models, the input variables are the injection patterns and the network configurations. After the training, the reward $r(\tilde{S}_t, \tilde{A}_t) + \lambda c(\tilde{S}_t, \tilde{A}_t)$ can then be calculated based on the out-of-sample prediction of the regression models applied to the synthesized data points $\tilde{S}_t, \tilde{A}_t$. It has been shown that inaccurate rewards in training data can hurt the learning process. Therefore, we must determine if the estimated rewards are reliable and discard the ones which have high uncertainty.

We choose the Gaussian process (GP) [131] as the regression model to learn both the estimated values and their uncertainties. In the GP setting, the target y and the input vector x are modeled by the relationship $y = f(x) + \epsilon$ where ϵ represents the observation noise and is typically a zero mean Gaussian $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$. f is a GP $f(x) \sim \mathcal{GP}(m(x), k(x, x'))$. If the mean function $m(x)$, the covariance function $k(x, x')$, and σ_ϵ^2 are known, then the probability distribution of any data $p(y|x)$ can be evaluated and the uncertainty is represented by the variance of $p(y|x)$. Typically, the mean and covariance functions of f are in some parametric families $m_{\theta_M}(x)$ and $k_{\theta_K}(x, x')$. For example, the constant mean function and the squared exponential covariance function are given by (5.9) In this example, $\theta_M = \{C\}$ and $\theta_K = \{A, \ell\}$. The parameters θ_M, θ_K, and σ_ϵ^2 can be estimated by marginalizing the Gaussian process $\mathcal{GP}(m_{\theta_M}, k_{\theta_K})$ onto the training data points $\mathbf{x}$. That is, $\mathbf{y} \sim \mathcal{N}(\mu_{\mathbf{x}}, \Sigma_{\mathbf{xx}} + \sigma_\epsilon^2 I)$ where $\mu_{\mathbf{x}} = m_{\theta_M}(\mathbf{x})$ and $\Sigma_{\mathbf{xx}} = k_{\theta_K}(\mathbf{x}, \mathbf{x})$. Then we can perform maximum likelihood estimation of the parameters on this marginal distribution.

$$m_{\theta_M}(x) = C \quad k_{\theta_K}(x, x') = A^2 \exp\left(-\frac{\|x - x'\|_2^2}{2\ell^2}\right) \tag{5.9}$$

Let the estimated parameters be $\hat{\theta}_M$, $\hat{\theta}_K$, and $\hat{\sigma}_\epsilon^2$. The posterior distribution of a testing instance $y^* = f(x^*) + \epsilon$ is again Gaussian, with the conditional mean and variance:

$$\hat{\mu}(y^*|x^*, \mathbf{x}, \mathbf{y}) = \hat{\mu}_{x^*} + \hat{\Sigma}_{x^*\mathbf{x}}(\hat{\Sigma}_{\mathbf{xx}} + \hat{\sigma}_\epsilon^2 I)^{-1}(\mathbf{y} - \hat{\mu}_{\mathbf{x}}) \tag{5.10}$$

$$\hat{\sigma}^2(y^*|x^*, \mathbf{x}, \mathbf{y}) = \hat{\sigma}_{x^*}^2 + \hat{\sigma}_\epsilon^2 - \hat{\Sigma}_{x^*\mathbf{x}}(\hat{\Sigma}_{\mathbf{xx}} + \hat{\sigma}_\epsilon^2 I)^{-1}\hat{\Sigma}_{\mathbf{x}x^*} \tag{5.11}$$

where $\hat{}$ means that the quantity is obtained by using the parameter estimates $\hat{\theta}_M$ and $\hat{\theta}_K$. $\hat{\sigma}^2(y^*|x^*, \mathbf{x}, \mathbf{y})$ is not quite the model uncertainty due to the lack of information about θ_M,θ_K, and σ_ϵ^2 [132]. An improved version is given by [132]:

$$\mathring{\sigma}^2(y^*|x^*, \mathbf{x}, \mathbf{y}) = \hat{\sigma}^2(y^*|x^*, \mathbf{x}, \mathbf{y}) + g^T M^{-1} g \tag{5.12}$$

where $g = \frac{\partial}{\partial \theta_M}[m_{\theta_M}(x^*) - m_{\theta_M}(\mathbf{x})^T(\hat{\Sigma}_{\mathbf{xx}} + \hat{\sigma}_\epsilon^2 I)^{-1}\hat{\Sigma}_{\mathbf{x}x^*}]$ and $M = \frac{\partial m_{\theta_M}(\mathbf{x})}{\partial \theta_M}(\hat{\Sigma}_{\mathbf{xx}} + \hat{\sigma}_\epsilon^2 I)^{-1}$ $\left[\frac{\partial m_{\theta_M}(\mathbf{x})}{\partial \theta_M}\right]^T$. Now, $\hat{\mu}(y^*|x^*, \mathbf{x}, \mathbf{y})$ and $\mathring{\sigma}^2(y^*|x^*, \mathbf{x}, \mathbf{y})$ represent the estimated target and its uncertainty. In the dynamic DNR problem, each x represents an injection pattern and a radial configuration and each y represents the corresponding network loss or a voltage magnitude. If the uncertainty of the target estimate $\mathring{\sigma}^2(y^*|x^*, \mathbf{x}, \mathbf{y})$ is larger than some threshold, then the synthetic data generated (x^*, y^*) will be discarded. In this chapter, the threshold is heuristically set to be $3 \cdot [\mathrm{std}(\mathring{\boldsymbol{\sigma}} - \mathrm{avg}(\mathring{\boldsymbol{\sigma}}))]$ where $\mathring{\boldsymbol{\sigma}}$ is the set of uncertainty estimates for all y^*.

5.5 Numerical Study

5.5.1 Experimental Data Description

The 16-bus distribution test feeder presented in [109, Example 1] is used in this study. The line impedances, remote controllable switches, and complex power base S_{base} of [109] are kept unchanged. After applying the rooted spanning forest enumeration procedure in Section 5.4.2, a total of 190 radial configurations are found. To validate our proposed RL algorithm for dynamic DNR problems, we replace the original static load data in [109] with 26 weeks of aggregated hourly real-world smart meter data of residential and commercial customers taken from a 12 KV distribution feeder.

The real-world smart meter data are reprocessed as follows. First, each nodal injection in the 16-bus feeder is set to be the aggregated consumption of a group of randomly selected customers. We assume a constant power factor for each node. Then, we scale these aggregated consumption by a common factor β (i.e., $(p_t, q_t) \mapsto (\beta p_t, \beta q_t)$ for all t) in order to create a realistic feeder loading level. β is chosen such that the resulting total line loss under βp_t and βq_t is roughly 1.5% of the total demand [133]. Next, we select 83 medium to low line loss configurations from a total of 190 feasible ones. A sample path of 26 weeks with hourly granularity is then generated from a Markov chain defined on those 83 configurations with transition probability $p_{ii} = 0.9$ and $p_{ij} = p_{ik}$. Finally, we find the power flow solutions for all hours in the 26-week period and record the total line losses as well as the voltage magnitude measurements at bus 7, 12, and 16 of the network to form the historical operational data set.

5.5.2 Setup of the Reward Function

By Section 5.3.2, the reward function is defined as the sum of negative costs of line losses (denoted as $C^l(s,a)$), switching actions (denoted as $C^s(s,a)$), and a weighted constraint violation term $\lambda c(s,a)$. C^l equals to the product of a fixed retail electricity price and the network losses. We set the retail electricity price at \$0.13/kWh. C^s equals the product a fixed cost per switching and the number of switching actions.

The fixed cost per switching is determined as follows. First, the lifetime cost of a sectionalizing switch can be calculated as the summation of the equipment cost, installation cost, and maintenance costs over its useful life [134]. The sum of equipment cost and installation cost is assumed to be \$4,700. The useful life and annual maintenance cost of a switch are set to be 15 years and \$94. Thus, the lifetime cost of a sectionalizing switch is \$6,110. If we assume that the number of operations of a switch over its lifetime is 657 [135], then the fixed cost per switching is approximately \$4.6.

The upper and lower bounds $\bar{v}, \underline{v}$ for the voltage violation term is chosen as 1.1 and 0.9 p.u., respectively. λ is chosen to be $\$0.13/kWh \times 100MVA = \$13,000$/p.u.

5.5.3 Performance of Operational Experience Augmentation

In this subsection, we validate the quality of the synthetic operational experience data generated by our proposed GP based model. In particular, the quality of estimated network losses under the augmented network configuration and injection patterns will be evaluated. Recall that we have 26 weeks of historical data set, which are divided into training data set (D_t) and testing data set. The first 25 weeks of historical data are chosen as the training data set and the data of the last week are chosen as the testing data set.

We then create a 25-week synthetic operational experience data set $\tilde{D}_t$ as follows. First, we generate a 25-week sample path of network configurations from a Markov chain defined on those configurations that appeared in the training data set with transition probability $p_{ii} = 0.8$ and $p_{ij} = p_{ik}$. We then estimate the network losses for this new sequence of configurations under the injection patterns of the first 25 weeks of historical data set. For the network loss estimation task, we compare our proposed GP model in Section 5.4.3 with the Monte Carlo (MC) dropout neural network [136], which is shown to be equivalent to a Bayesian approximation of a GP. When building the GP model, the mean function is chosen to be zero and the covariance function is chosen to be the same as in (5.9). Both the GP and the MC dropout model are trained with the first 25 weeks of historical operational data. We apply the trained model to the 1-week testing data set and the 25-week synthetic operation experience data set. Figure 5.2 shows the performance of network losses prediction for the two models under 50 samples of both the testing data set and the synthetic data set. As shown in the figure, compared to the MC dropout model, the GP model is much more accurate in predicting network losses.

Although GP model produces fairly accurate predictions, it occasionally leads to large error for some network configurations and injection patterns as shown by the orange curve in Figure 5.3. Fortunately, the uncertainty estimates of the GP model represented by

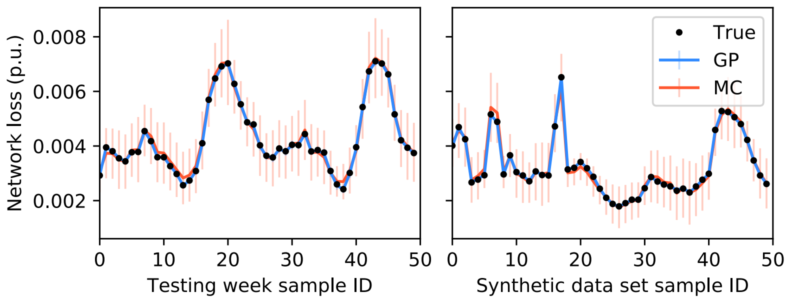


Figure 5.2: Performance of out-of-sample predictions for network losses.

the blue curve in Figure 5.3 correlates very well with the estimation error. This suggests our proposed strategy of removing the samples with large uncertainty estimates significantly improves the quality of the augmented operational experience data set.

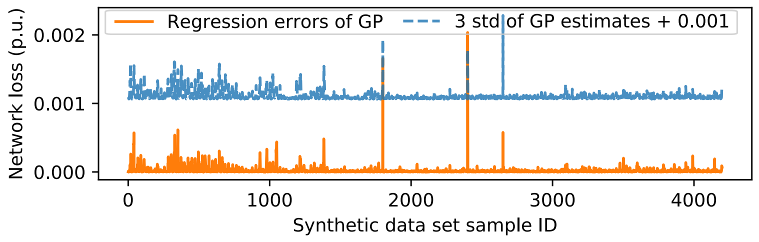


Figure 5.3: Regression errors versus uncertainty estimates of the GP model.

5.5.4 Performance of Deep Q-Learning Algorithms

In this subsection, we compare the performance of three deep Q-learning algorithms with two benchmarks. In the first benchmark algorithm, global optimal solution of the dynamic DNR problem is obtained by dynamic programming with perfect knowledge of the network parameters and future injection pattern. The second benchmark simply uses the historical network configurations generated in the data set. The first deep Q-learning algorithm is developed and trained using only historical operational data. The second deep

Q-learning algorithm is trained with both historical and synthetic operational experiences, where the network losses are estimated based on the GP model. The third deep Q-learning algorithm is trained with both historical and synthetic operational experiences, where the network losses are obtained with the power flow models assuming perfect knowledge of the network parameters.

We divide the 26-week historical data set into a 25-week training D_t data set and a 1-week testing data set. The 25-week synthetic operational experience data set $\tilde{D}_t$ is generated in the same way as in Section 5.5.3. During the training iterations, we periodically save the parameters of the trained neural network and test its performance on the testing data set. The performance of the three Q-learning algorithms and two benchmark algorithms are shown in Figure 5.4. The left subfigure shows the minimum voltage magnitude over all metered nodes and all hours. The right subfigure displays the total operational cost. For the three deep Q-learning algorithms, the average, the 10th, and the 90th percentile of the results from 10 independent runs are depicted.

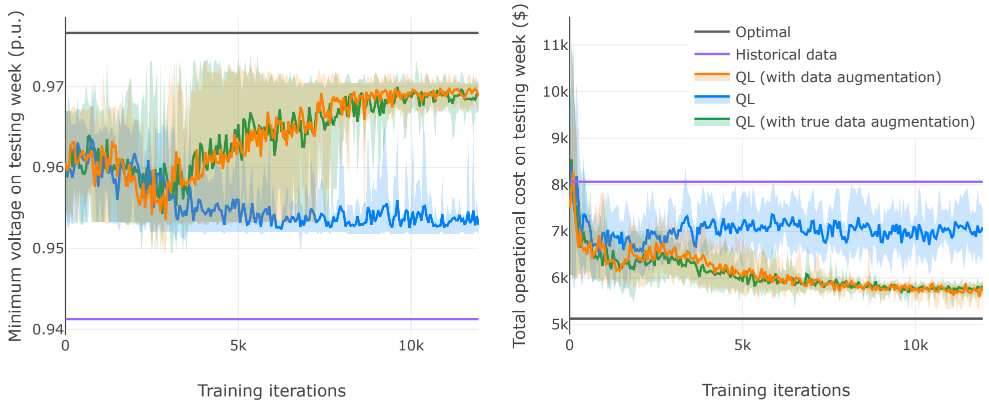


Figure 5.4: Performance of Q-learning. Hyperparameters of the neural network: 2-layer feed-forward (hidden: 600, output: 190); activation function: ReLU; optimizer: Adam; batch size: 64; discount factor γ: 0.95; update steps C: 30.

Compared to the network configurations in the testing week of the historical data, the deep Q-learning algorithm quickly learned how to reduce the operational cost in a dynamic DNR problem. When we augment the historical operational experiences with synthetic operational data, then the operational cost of the deep Q-learning algorithm further reduces and the minimum voltage magnitudes get even closer to the nominal voltage values. As the learning process proceeds, the performance of the deep Q-learning algorithms with augmented operational experiences approaches that of the global optimal solution. Note that the Q-learning agents achieved these results without knowing the actual network parameters or future power injection patterns. It can also be seen from the figure that the orange curve almost coincides with the green curve. It means that the network losses estimated by our proposed GP model are almost as good as that of the power flow solutions with perfect network parameter information.

We conclude the numerical study by showing that similar results can be obtained without extensive tuning of hyperparameters, which is crucial for practical applications. We demonstrate this by showing that the performance of the Q-learning algorithm is relatively consistent under different hyperparameter settings. The following combinations of hyperparameters are tested, batch size $B \in \{32, 64, 128, 256\}$, number of hidden layers $L \in \{1, 2\}$, number of hidden neurons $H \in \{300, 400, 500, 600\}$, and number of steps the target Q network's parameters are updated $C \in \{30, 60, 90, 120\}$. We generate a Taguchi's orthogonal array for these hyperparameter combinations and report the results in Table 5.1. Each calculated cost represents the average of 5 independent runs for Q learning with operational data augmentation. Compared to the historical operational cost and the optimal cost, the operational cost of the Q-learning algorithm under different hyperparameter settings are quite consistent.

Table 5.1: Operational Costs with Various Hyperparameters

Original cost : \$8066.7. Optimal cost: \$5128.8									
B	H	C	L	QL cost	B	H	C	L	QL cost
32	300	30	1	5752.9	128	300	90	1	5490.6
32	400	60	1	5686.2	128	400	120	1	5647.4
32	500	90	2	5608.8	128	500	30	2	5441.0
32	600	120	2	5464.3	128	600	60	2	5396.7
64	300	60	2	5523.9	256	300	120	2	5480.4
64	400	30	2	5456.8	256	400	90	2	5487.8
64	500	120	1	5579.3	256	500	60	1	5652.2
64	600	90	1	5507.9	256	600	30	1	5531.3

5.6 Summary

This chapter presents a reinforcement learning based algorithm to solve the dynamic distribution network reconfiguration problem without accurate network parameter information. The proposed framework first formulates the dynamic DNR problem as a Markov decision process, then learns the approximated optimal action-value function with a neural network. The optimal network configuration is selected to be the action that yields the highest action-value. A novel synthetic operational experience data generation technique based on the Gaussian process is developed to improve the performance of Q-learning algorithms. Simulation results show that the proposed Q-learning algorithm successfully reduces the operational cost of the network under various hyperparameter settings.

Chapter 6

Deep Reinforcement Learning for Distribution Network Reconfiguration Part 2: Batch-Constrained Soft Actor Critic

6.1 Introduction

In the previous chapter, we discussed the deep Q learning (DQL) and operational experience augmentation to solve to solve small-scale network reconfiguration problem. However, for larger scale distribution networks, the number of feasible configurations can be so large that the learning becomes almost impossible. In this chapter, we develop a novel deep RL algorithm, called batch-constrained soft actor critic (BCSAC), to improve the RL algorithm's scalability. The name of the algorithm is derived from the so-called batch RL, in which the agent is learning from a fixed experience dataset, rather than interacting with the environment. BCSAC scales to large networks with limited operational data by training a control policy in the following way: it maximizes the total discounted return while minimizing the dissimilarity between the learned control policy and the behavior policy of the operational data (the batch data). Therefore, it avoids unwanted extrapolation errors in the region of the state-action space beyond the batch data.

In this chapter, we discuss the details of the BCSAC algorithm. First, we prove the convergence of the KL-divergence regularized (batch-constrained) version of the policy iteration. This provides a theoretical justification of the correctness of the algorithm. Then, we discuss the actor-critic framework and apply it in the batch RL setup. Finally, to train the BCSAC algorithm so that it can minimize the dissimilarity between its policy and the behavior policy, we represent the behavior policy with a conditional variational autoencoder, and regularize the reward function with the Kullback–Leibler (KL) divergence between the learned control policy and the behavior policy. All components are trained using standard machine learning optimization routine. We evaluate the trained algorithm on several test distribution networks with real-world smart meter data. Numerical study results show that our proposed BCSAC algorithm is able to successfully learn a network reconfiguration strategy for very large scale distribution networks. It not only improves the behavior control policy but also outperforms state-of-the-art off-policy RL algorithms.

The rest of this chapter is organized as follows: Section 6.2 revisits the dynamic DNR problem as a Markov decision process. Section 6.3 presents the technical methods of our proposed BCSAC algorithm. Section 6.4 shows the numerical study results. Section 6.5 summarizes this chapter.

6.2 Problem Formulation

In this section, we present our formulation for the dynamic DNR problem as a Markov decision process (MDP) [125]. The preliminaries of MDP was reviewed in the previous chapter. Here we only briefly review the MDP formulation of DNR.

6.2.1 Formulate Dynamic DNR as an MDP

We consider a distribution network with n load nodes, m lines, and n_S substations. Let v_{it}, p_{it}, and q_{it} denote the voltage magnitude, real and reactive power net injections of node i at time t. p_t^l denotes the network's total real line losses. The binary variable $\alpha_{\ell t}$ represents the status of the switch ℓ. $\alpha_{\ell t} = 1$ if switch ℓ is closed at time t. We define vectors for nodal real and reactive power injections and branch status at time t as $\boldsymbol{p}_t = [p_{1t}, \cdots, p_{nt}]$, $\boldsymbol{q}_t = [q_{1t}, \cdots, q_{nt}]$, and $\boldsymbol{\alpha}_t = [\alpha_{1t}, \cdots, \alpha_{mt}]$.

Now, we formulate the dynamic DNR problem as an MDP by identifying the agent, state, action, and reward. The agent is the distribution system operator or controller. The state at time t is defined as $S_t = [\boldsymbol{p}_t, \boldsymbol{q}_t, \boldsymbol{\alpha}_t, t]$. Thus, $\mathcal{S}$ consists of the set of all power injection patterns together with the set of all radial configurations. We define the action taken at time t, A_t, as changing the topology of the network by a single pair of branch status exchange, that is, closing a switch in $\{1, \cdots, m\}$ and opening another one, such that the resulting configuration $\boldsymbol{\alpha}_{t+1}$ is still radial [96]. We deem opening and closing the same switch as staying in the same configuration, and it does not incur a switching cost. Note that starting from a given configuration $\boldsymbol{\alpha}_t$, only a subset of all switch pairs is feasible; the others will result in a loop or disconnected network. Thus, in each state s, only a subset of actions are allowed to be chosen. We defer the implementation details to Section 6.3.8.

The above formulated states and actions uniquely define a state-transition probability model $P(S_{t+1}|S_t, A_t)$ between the current state $S_t = [\boldsymbol{p}_t, \boldsymbol{q}_t, \boldsymbol{\alpha}_t, t]$ and the next state S_{t+1}. The new state's variables $[\boldsymbol{p}_{t+1}, \boldsymbol{q}_{t+1}, \boldsymbol{\alpha}_{t+1}, t+1]$ are determined by S_t and A_t as follows. The transition probability between adjacent time steps' power injections $P(\boldsymbol{p}_{t+1}, \boldsymbol{q}_{t+1}|\boldsymbol{p}_t, \boldsymbol{q}_t)$ can be described by the random process of the power injections and is not affected by the action. $\boldsymbol{\alpha}_{t+1}$ is determined by $\boldsymbol{\alpha}_t$ and the open/closing switches in A_t. The global time variable t is increased by 1.

The reward function reflects the cost associated with network line losses p_t^l, the switching cost, and operating limit constraint violation penalty and is defined as follows.

$$R_{t+1} = r(S_t, A_t) = -C^l p_t^l(\boldsymbol{p}_t, \boldsymbol{q}_t, \boldsymbol{\alpha}_{t+1}) - C^s|\boldsymbol{\alpha}_{t+1} - \boldsymbol{\alpha}_t| - \lambda c(\boldsymbol{p}_t, \boldsymbol{q}_t, \boldsymbol{\alpha}_{t+1}) \tag{6.1}$$

where C^l is the unit cost of electricity. C^s is the cost of opening or closing of a switch. The third term in (6.1) describes the voltage constraint violation penalty [19]:

$$c(\boldsymbol{p}_t, \boldsymbol{q}_t, \boldsymbol{\alpha}_{t+1}) = \sum_{i \in N^v} [\max(0, v_{it} - \bar{v}) + \max(0, \underline{v} - v_{it})] \tag{6.2}$$

where N^v is the set of all nodes that have voltage measurement devices; $\bar{v}$ and $\underline{v}$ are the upper and lower bounds for voltage; λ is the penalty factor associated with the voltage constraint violation. The value of λ can be determined based on operational considerations and empirical performance. The exact value of λ will be provided at Section 6.4.1.

Finally, we choose a discount factor γ that is less than 1 and set $T = \infty$. This completes the MDP formulation for the dynamic DNR problem. In sum, the dynamic DNR problem is a continuing task with a finite action space and a state space with continuous variables. The learning setup of this MDP is explained in the next subsection.

6.2.2 The Learning Setup

The RL task for the dynamic DNR problem is to learn a good control policy from a given set of historical operational data. The setup of learning from a given historical dataset rather than from directly interacting with the environment is known as batch reinforcement learning [137]. From now on, the term batch and historical operational data will be used interchangeably. The historical operational data should contain relevant information about the state, action, and reward of the MDP and will be explained in detail below.

We assume the historical operational data are to be derived through the following measurements collected by an electric utility. First, the nodal power injections $p_{it} + jq_{it}$ at each time step and node with non-zero injection are recorded by smart meters or other sensors. Second, the SCADA system records real and reactive power at the substations. Third, the nodal voltage magnitude data v_{it} is available from the SCADA system at a subset of nodes in the network. Finally, the switch status $\boldsymbol{\alpha}_t$ are available from the remotely controllable switches. With these measurements, we can construct the historical states, actions, and rewards of the MDP. In particular, the network loss can be estimated as the sum of all net power injections of the distribution network $p_t^l = \sum_{i=1}^{n+n_S} p_{it}$.

Two factors make it challenging to develop a batch RL algorithm to solve the MDP representing the dynamic DNR problem. First, the state space of the MDP is high-dimensional and grows exponentially with the size of the distribution network. Leveraging function approximators such as neural networks to estimate the value function or control policy associated with this high-dimensional state space is not straightforward. Second, the batch RL controller can only learn from the limited information contained in a finite amount of historical operational data.

6.3 Technical Methods

In this section, we first present the preliminaries of actor-critic algorithms and batch RL algorithms. Then we develop our proposed BCSAC algorithm. Finally, we provide the RL algorithm implementation details for the dynamic DNR problem.

6.3.1 Actor-Critic Algorithms

For dynamic DNR with large state-space and action-space, it can be difficult to apply value-based RL algorithms to approximate the action-value function. To deal with this, actor-critic algorithms have been proposed. Actor-critic algorithms uses an actor to learn a parameterized control policy that directly selects actions without consulting a value function; and a critic to estimate the policy value of the actor [125]. Since the policy is explicitly represented, actor-critic algorithms can handle RL problems with much larger action spaces, even continuous ones. To further improve the sample efficiency and robustness of the actor-critic methods, state-of-the-art maximum entropy RL algorithms such as soft actor-critic (SAC) [138] have been developed. Next, we provide a brief review of the SAC algorithm.

Soft Actor Critic

Soft actor critic [138] regularizes the reward function by the entropy of the policy: $r(s,a) + \tau H(\pi(\cdot|s))$, whose contribution to the reward is controlled by the temperature parameter τ. The entropy regularized state-value functions $v_\pi^{\mathrm{h}}(s)$ and action-value functions $q_\pi^{\mathrm{h}}(s,a)$ are shown to satisfy [139]:

$$v_\pi^{\mathrm{h}}(s) = \mathbb{E}_{a\sim\pi}\mathbb{E}_{s'\sim P}\left[r + \gamma v_\pi^{\mathrm{h}}(s')\right] + \tau H(\pi(\cdot|s)) \tag{6.3}$$

$$q_\pi^{\mathrm{h}}(s,a) = r + \gamma \mathbb{E}_{s'\sim P}\left[v_\pi^{\mathrm{h}}(s')\right] \tag{6.4}$$

$$v_\pi^{\mathrm{h}}(s) = \mathbb{E}_{a\sim\pi}[q_\pi^{\mathrm{h}}(s,a)] + \tau H(\pi(\cdot|s)) \tag{6.5}$$

To deal with large continuous domains, the value functions (critic) and the policy function (actor) shown above can be approximated by neural networks: $v_\psi(s), q_\theta(s,a), \pi_\phi(a|s)$, where ψ, θ, and ϕ are the parameters of the corresponding neural networks. The SAC algorithm works by iteratively updating the parameters of the value functions and the policy function.

$$\pi_{new}(\cdot|s) = \arg\min_\pi \ D_{\mathrm{KL}}\left(\pi(\cdot|s)||\frac{\exp(q_{\pi_{old}}^{\mathrm{h}}(s,a)/\tau)}{Z_{\pi_{old}}(s)}\right) \tag{6.6}$$

The parameters of value functions can be updated according to the gradient of the squared residual error of state value function and the soft Bellman residual of action value function. The parameters of the policy can be updated by (6.6), where $Z_{\pi_{old}}(s)$ is the partition function that normalizes the numerator to a probability distribution. $D_{\mathrm{KL}}(p||q)$ is the KL-divergence between distributions p and q.

6.3.2 Batch-Constrained Reinforcement Learning

In the batch RL setup, the agent can only learn from a finite dataset collected by some sampling procedure. For example, the historical operational dataset may be generated from a model-based controller and/or heuristic control actions selected by operators. Therefore, if we directly apply off-policy RL algorithms such as DQN or SAC in the batch RL setup, then the action-value function $q_\pi(s,a)$ of a given policy π may not be accurately evaluated. As a result, the learning agent may erroneously extrapolate $q_\pi(s,a)$ of some actions a to higher values [140]. Formally, let $q_\pi(s,a)$ denote the true action-value function of a policy π and $q_\pi^{\mathcal{D}}(s,a)$ denote the action-value function of policy π estimated using the batch data. Then the extrapolation error of a state-action pair $\epsilon_\pi(s,a)$ and the extrapolation error of policy ϵ_π can be defined as:

$$\epsilon_\pi(s,a) = q_\pi(s,a) - q_\pi^{\mathcal{D}}(s,a) \tag{6.7}$$

$$\epsilon_\pi = \textstyle\sum_s \mu_\pi(s) \sum_a \pi(a|s)|\epsilon_\pi(s,a)| \tag{6.8}$$

where μ_π is the state-visitation probability induced by π in the *original* MDP $\mathcal{M}$. It has been shown that [140], $\epsilon_\pi = 0$ if and only if the empirical transition probability of the batch data $\hat{p}(s'|s,a)$ is equal to the true $p(s'|s,a)$ for all state-action pairs (s,a) with non-zero visitation probability under policy π. In this case, $q_\pi(s,a)$ can be evaluated with no error. In other words, to accurately estimate state-value functions, the agent should try to learn control policies, which tend to visit the state-action pairs contained in the batch data. A policy that satisfies this condition is denoted as batch-constrained.

6.3.3 KL-Divergence Regularization and the Bellman Equation

To find a batch-constrained policy, we propose to regularize the reward function by the *KL-divergence* between the target policy and the behavior policy:

$$r^{\mathrm{d}}(s,a) = r(s,a) - \tau D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s)) \tag{6.9}$$

where $r(s,a)$ is the reward function of the original MDP. $\pi^b(a|s)$ is the behavior policy, which has the same conditional probability distribution of the actions given state as that of the historical data. The KL-divergence can be calculated as $D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s)) = \mathbb{E}_{a\sim\pi}\left[\log\pi(a|s) - \log\pi^b(a|s)\right]$. This term encourages the agent to learn batch-constrained policies that are similar to the policy generating the historical operational data. We can rewrite the KL-divergence as

$$D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s)) = H(\pi(\cdot|s), \pi^b(\cdot|s)) - H(\pi(\cdot|s)) \tag{6.10}$$

where $H(\pi(\cdot|s), \pi^b(\cdot|s))$ is the cross entropy of $\pi(\cdot|s)$ and $\pi^b(\cdot|s)$. $H(\pi(\cdot|s))$ is the entropy of the target policy. Therefore minimizing the KL-divergence can be thought of as maximizing the target policy's entropy coupled with minimizing the cross entropy. We denote the value functions for a given policy π with KL-divergence regularized reward function as $v_\pi^{\mathrm{d}}(s)$ and $q_\pi^{\mathrm{d}}(s,a)$. The Bellman equations under this setup are derived as:

$$v_\pi^{\mathrm{d}}(s) = \mathbb{E}_{a\sim\pi}\mathbb{E}_{s'\sim P}\left[r + \gamma v_\pi^{\mathrm{d}}(s')\right] - \tau D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s)) \tag{6.11}$$

$$q_\pi^{\mathrm{d}}(s,a) = r + \gamma\mathbb{E}_{s'\sim P}\left[v_\pi^{\mathrm{d}}(s')\right] \tag{6.12}$$

$$v_\pi^{\mathrm{d}}(s) = \mathbb{E}_{a\sim\pi}[q_\pi^{\mathrm{d}}(s,a)] - \tau D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s)) \tag{6.13}$$

Our next result shows that for a given policy π, the value function $q_\pi^{\mathrm{d}}(s,a)$ can be found by the following iterative scheme:

Lemma 6.3.1 (Batch-Constrained Soft Policy Evaluation) *Consider the operator $\mathcal{T}^\pi$ given by:*

$$\mathcal{T}^\pi q(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim P}[v(s')] \quad \forall s,a \tag{6.14}$$

$$v(s') = \mathbb{E}_{a'\sim\pi}[q(s',a')] - \tau D_{\mathrm{KL}}(\pi(\cdot|s')||\pi^b(\cdot|s')) \tag{6.15}$$

and an initial $q^0(s,a)\in\mathbb{R}, \forall(s,a)\in\mathcal{S}\times\mathcal{A}$. Assuming that $D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s))$ is bounded for all $s\in\mathcal{S}$, the sequence defined by $q^{k+1} = \mathcal{T}^\pi q^k$ will converge to the KL-divergence regularized Q function q_π^{d} as $k\to\infty$.

After $q_\pi^{\mathrm{d}}(s,a)$ is computed, we can invoke the following update rule to find an improved policy π'.

Lemma 6.3.2 (Batch-Constrained Soft Policy Improvement) *Given a policy π and* *its soft Q function q^{d}_{π}, define a new policy π' as follows:*

$$\pi'(\cdot|s) = \arg\max_{\tilde{\pi}} \ \mathbb{E}_{a\sim\tilde{\pi}}[q^{\mathrm{d}}_{\pi}(s,a)] - \tau D_{\mathrm{KL}}(\tilde{\pi}(\cdot|s)||\pi^{b}(\cdot|s))$$

for every $s \in \mathcal{S}$. Then $q^{\mathrm{d}}_{\pi'}(s,a) \geq q^{\mathrm{d}}_{\pi}(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

By Lemma 6.3.1 and Lemma 6.3.2, we can establish the following batch-constrained version of the policy iteration theorem:

Theorem 6.3.3 (Batch-Constrained Soft Policy Iteration) *Starting from any policy π and alternatively applying the batch-constrained soft policy evaluation and improvement, the sequence of policies converges to a policy π_* such that $q^{\mathrm{d}}_{\pi_*}(s,a) \geq q^{\mathrm{d}}_{\pi}(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.*

All proofs can be found in Appendix B.1. Theorem 6.3.3 establishes the theoretical foundation for finding the optimal batch-constrained soft policy. However, it cannot be directly implemented due to infinite state space and finite training data in the dynamic DNR problem. Later in this section, we will derive a practical algorithm that approximately implements the batch-constrained soft policy iteration. Before that, we first provide an overview of the proposed reinforcement learning based dynamic DNR control framework in the next subsection.

6.3.4 Overview of the Proposed Framework

This subsection provides an overview of the proposed RL based dynamic DNR control framework. Figure 6.1 shows the sub-modules of the proposed framework.

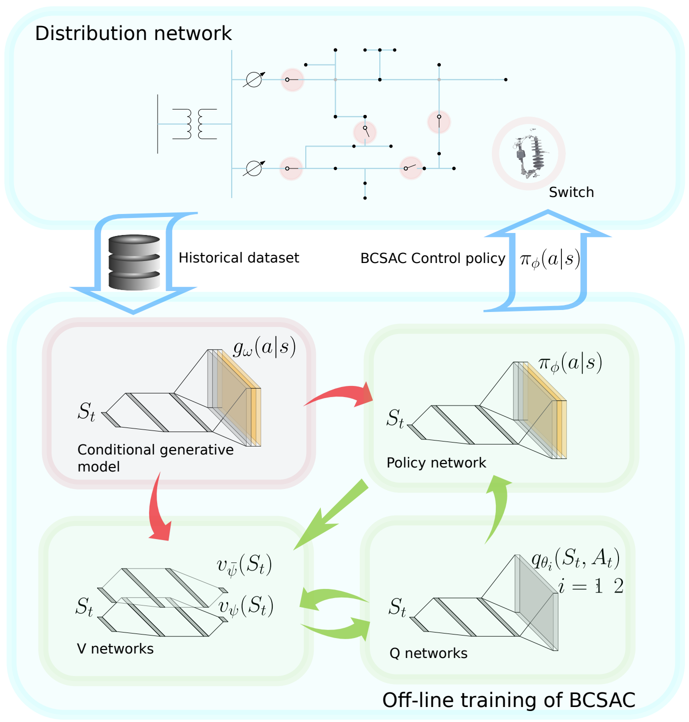


Figure 6.1: The proposed RL based dynamic DNR control framework

The electric utility first collects the historical operational dataset as described in Section 6.2.2. This dataset will then be used for off-line training of the proposed batch-constrained soft actor-critic (BCSAC) RL algorithm. The algorithm consists of a conditional generative model, represented by the red block, and three groups of neural networks represented by the three green blocks. The conditional generative model is trained independently from the other neural networks and thus marked as red. The neural networks in the three green blocks are trained simultaneously. The red and green arrows represent the dependencies among the neural network training processes. After off-line training, the "policy network" module will contain a trained neural network $\pi_\phi(a|s)$, which takes the network configuration and injection pattern as the input, and outputs a reconfiguration action. The policy network is trained to approximate the optimal batch-constrained soft policy $\pi_*(a|s)$.

The V and Q networks are trained to approximate $v^{\mathrm{d}}_{\pi_*}(s)$ and $q^{\mathrm{d}}_{\pi_*}(s,a)$, respectively. In the next subsection, we present the details of the off-line training processes.

6.3.5 Batch-Constrained Soft Actor Critic

We propose an actor-critic algorithm which approximates the policy iteration and hence, learns a batch-constrained policy from the finite historical operational dataset. The algorithm consists of a *critic*, which approximates $v^{\mathrm{d}}_{\pi_*}(s)$ and $q^{\mathrm{d}}_{\pi_*}(s,a)$, and an *actor*, which approximates $\pi_*(a|s)$.

The Critic

We parameterize $v^{\mathrm{d}}_{\pi}(s)$ and $q^{\mathrm{d}}_{\pi}(s,a)$ by neural networks and update them using the sample estimate of RHS of the (6.12)-(6.13). In addition, we adopt the target value network [127] and the clipped-double Q method [141] to stabilize the training. Specifically, we maintain four neural networks $q_{\theta_1}, q_{\theta_2}, v_{\psi}, v_{\bar{\psi}}$, and update them by:

$$\min_{\theta_i} \frac{1}{|\mathcal{B}|} \sum_{(s,a,r,s')\in\mathcal{B}} \left[q_{\theta_i}(s,a) - (r + \gamma v_{\bar{\psi}}(s'))\right]^2 \; i = 1,2 \tag{6.16}$$

$$\min_{\psi} \, (1/|\mathcal{B}|) \sum_{(s,a,r,s')\in\mathcal{B}} (v_{\psi}(s) - v^{\text{target}}(s))^2 \tag{6.17}$$

$$v^{\text{target}}(s) = \min_{i=1,2} q_{\theta_i}(s,\hat{a}) - \tau\log(\pi_{\phi}(\hat{a}|s)) + \tau\log(\pi^b(\hat{a}|s)) \tag{6.18}$$

$$\bar{\psi} \leftarrow \rho\bar{\psi} + (1-\rho)\psi \tag{6.19}$$

where $\mathcal{B}$ is a mini-batch sampled from the historical data $\mathcal{D} = \{(s,a,r,s')\}$. ρ is an exponential smoothing parameter. $\hat{a}$ is a sampled action from the policy network $\pi_{\phi}(\cdot|s)$. The input of all the neural networks is the state s. For the value networks v_ψ and $v_{\bar{\psi}}$, the output is a single number indicating the state value. The output of the Q networks q_{θ_1} and q_{θ_2} is a vector including the action-values. All networks are standard feedforward neural networks with a number of hidden layers. For the dynamic DNR problem, the detailed architecture design of the Q and V networks are described in Section 6.3.8.

When performing the minimization (6.16) to train the Q networks q_{θ_1} and q_{θ_2}, the parameter vector $\bar{\psi}$ is held fixed. Similarly, when performing the minimization (6.17), all the parameters appearing in $v^{\text{target}}(s)$ are fixed and only ψ is to be optimized. The training data of these networks are obtained from the historical operational dataset and converted into the state s, action a, reward r, and next state s' format. The process was described in Section 6.2.1 and Section 6.2.2. In addition, sampled actions $\hat{a}$ from the current policy are also used for the training. But these samples $\hat{a}$ do not need to be the same as the actions in the historical dataset.

Next, we discuss the design of the actor and the derivation of the policy gradient.

The Actor

We approximate the policy function (the actor) by a neural network parameterized by ϕ. Ideally, the parameters should be updated using gradient ascent $\phi \leftarrow \phi + \eta \nabla v^{\mathrm{d}}_{\pi_\phi}(s)$, where the $\nabla v^{\mathrm{d}}_{\pi_\phi}(s)$ is given by

$$\begin{aligned}\nabla v^{\mathrm{d}}_{\pi_\phi}(s) &= \nabla[\mathbb{E}_{a\sim\pi_\phi}[q^{\mathrm{d}}_\pi(s,a)] - \tau D_{\mathrm{KL}}(\pi_\phi(\cdot|s)||\pi^b(\cdot|s))] \\ &= \nabla\mathbb{E}_{a\sim\pi_\phi}[q^{\mathrm{d}}_\pi(s,a) - \tau(\log\pi_\phi(a|s) - \log\pi^b(a|s))] \qquad (6.20)\end{aligned}$$

However, this policy gradient requires computing the derivative of $q^{\mathrm{d}}_{\pi}(s,a)$. It is shown that this derivative will result in an *on-policy* policy gradient [125], which cannot be estimated from a given historical dataset.

Fortunately, as shown by Lemma 6.3.2, the gradient of $q^{\mathrm{d}}_{\pi}(s,a)$ can be omitted. This is because the objective function of Lemma 6.3.2 treats $q^{\mathrm{d}}_{\pi}(s,a)$ as a constant. In other words, updating the actor without the gradient information of $q^{\mathrm{d}}_{\pi}(s,a)$ still approximates the monotonic policy improvement. With this theoretical guarantee, we can derive an *off-policy* policy gradient, which can be estimated from the historical dataset. The derivation is done in three steps. In the first step, we omit the gradient of $q^{\mathrm{d}}_{\pi}(s,a)$. This is justified by Lemma 6.3.2. In the second step, we change the order of the gradient operator and the expectation operator and define a new term $f_\phi(s,a) = q^{\mathrm{d}}_{\pi}(s,a) - \tau(\log \pi_\phi(a|s) - \log \pi^b(a|s))$ to simplify the notation.

$$\begin{aligned}
&\nabla \mathbb{E}_{a\sim\pi_\phi}[f_\phi(s,a)] \\
=&\sum_a f_\phi(s,a)\nabla\pi_\phi(a|s) - \tau \underbrace{\sum_a \pi_\phi(a|s)\nabla\log\pi_\phi(a|s)}_{=0} \\
=&\sum_a \pi_\phi(a|s) f_\phi(s,a)\nabla\log\pi_\phi(a|s) \\
=&\mathbb{E}_{a\sim\pi_\phi} f_\phi(s,a)\nabla\log\pi_\phi(a|s) \qquad (6.21)
\end{aligned}$$

where we have used the identity $\sum_a \pi_\phi(a|s)\nabla\log\pi_\phi(a|s) = \sum_a \nabla\pi_\phi(a|s) = \nabla\sum_a \pi_\phi(a|s) = \nabla 1 = 0$. In the third step, we replace the expectation in (6.21) by its one-sample estimate. The final form of the approximate policy gradient is given by (6.22) where $\hat{a}$ is sampled from $\pi_\phi(\cdot|s)$. This completes the derivation of the actor network update process.

The structure of the policy network is as follows. The input of π_ϕ is the state s, and the output is a conditional probability distribution of actions given the state. For the dynamic DNR problem, the detailed architecture for the policy network is described in Section 6.3.8. Note that in order to evaluate (6.18) and (6.22), we need to approximate the behavior policy $\pi^b(\hat{a}|s)$ by a parametric function $g_\omega(\hat{a}|s)$. This will be discussed in the next subsection.

$$\hat{\nabla} v^{\mathrm{d}}_{\pi_\phi}(s) = \nabla \log \pi_\phi(\hat{a}|s)[q_{\theta_1}(s, \hat{a}) - \tau(\log \pi_\phi(\hat{a}|s) - \log \pi^b(\hat{a}|s))] \tag{6.22}$$

6.3.6 Representing the Batch Distribution as a Parametric Model

Given the difficulty of estimating the behavior policy $\pi^b(a|s)$ with high-dimensional state and action space, we propose using the conditional variational autoencoder (CVAE) [142] as the parametric generative model for $g_\omega(a|s)$, where ω is the model parameter. Using non-parametric models for this learning task can be very difficult because they suffer from the curse of dimensionality. Furthermore, non-parametric models have difficulty handling mixed discrete and continuous variables. On the other hand, CVAE model is well suited for our application due to three reasons. First, CVAE model is very scalable and can approximate high-dimensional distributions. Second, CVAE model can easily handle mixed discrete and continuous state-action space. Third, as will be shown in Section 6.4.3, CVAE model has good empirical performance for our application.

CVAE consists of an encoder $c_{\omega'}(z|s, a)$, which maps a given state-action pair to a latent representation z, and a decoder $g_\omega(a|s, z)$, which produces the probability of taking an action a given z and s. CVAE maximizes the following objective function to obtain the parameters for the encoder, ω', and the decoder ω:

$$\mathbb{E}_{z\sim d_{\omega'}}[\log g_{\omega}(a|s,z)] - D_{\mathrm{KL}}(c_{\omega'}(z|s,a)||p(z)) \tag{6.23}$$

where $p(z)$ is the latent variable distribution and is chosen as a Gaussian $\mathcal{N}(0, I)$. To train CVAE, we sample mini-batches of state-action pairs (s, a) from the historical data $\mathcal{D}$ and perform stochastic gradient ascent for the sample objective function. The trained decoder $g_{\omega}(a|s,z)$ is used to represent the behavior policy of the historical operational data.

6.3.7 Summary of BCSAC Algorithm

Our proposed batch-constrained soft actor-critic (BCSAC) algorithm is summarized in Algorithm 1. The algorithm takes as inputs the operational historical dataset $\mathcal{D}$ (the batch) as well as the trained CVAE model g_{ω}. Before the training starts, the policy and value networks are initialized using general-purpose deep neural network initialization algorithms (we choose the Xavier initialization in this work). In each iteration, the algorithm first samples a mini-batch of experiences from the batch, and then samples actions from the current policy. Afterwards, the algorithm conducts policy evaluation by training the V and Q-networks using (6.17) and (6.16), respectively. At the end of each iteration, the policy improvement step is taken by training the policy network, which updates the parameters ϕ using the gradient shown in (6.22). Note that since the historical dataset $\mathcal{D}$ does not change during the training process, the trained CVAE model g_{ω} does not need to be updated.

The proposed algorithm differs from existing actor-critic frameworks (e.g. [138]) in three ways. First, the framework is developed from a novel batch-constrained soft policy iteration theory presented in Theorem 6.3.3. Second, we utilize finite action space policy gradient in (6.22) to update the actor network, instead of the reparameterization trick [138]. Third, a pre-trained conditional generative model is incorporated for the training of the batch-constrained RL algorithm.

For the dynamic DNR problem, the historical data $\mathcal{D}$ consists of the nodal power injections, substation SCADA power measurements, nodal voltage magnitudes, and the status of remotely controllable switches. These data have been converted into the state, action, reward, next state tuple (s, a, r, s') prior to the training. The detailed procedure was described in Section 6.2.1-Section 6.2.2. To apply the BCSAC algorithm to the dynamic DNR problem, we design unique neural network architectures and the representation of distribution network topology in these networks. This is the subject of the next subsection.

Algorithm 1 BCSAC with Finite Action Space

```
Input: Batch $\mathcal{D}$, conditional generative model $g_\omega \approx \pi^b$
1: Initialize $\phi, \theta_1, \theta_2, \psi, \bar{\psi}$
2: for $i = 1, \cdots,$ do
3:     Sample mini-batch $\mathcal{B} = \{(s, a, r, s')\}$ from $\mathcal{D}$
4:     Sample actions from the current policy: $\hat{a} \sim \pi_\phi(\cdot|s)$
5:     Train Q networks $\theta_1, \theta_2$ by (6.16)
6:     Train V network $\psi$ by (6.17)
7:     Update V target network $\bar{\psi}$ by (6.19)
8:     Train policy network $\phi$ by $\phi \leftarrow \phi + \eta \hat{\nabla} v^{\mathrm{d}}_{\pi_\phi}(s)$
       where $\hat{\nabla} v^{\mathrm{d}}_{\pi_\phi}(s)$ is given by (6.22)
```

6.3.8 Algorithm Implementation

This subsection provides the technical details of implementing BCSAC algorithm for the dynamic DNR problem. The neural network architecture design and representation of distribution network topology are covered.

• Representation of distribution network configuration as an input to neural networks: we use a binary vector of on/off status of each line segment to encode the distribution network configurations. Since the configuration at each time step must be radial, the next feasible state configurations $\boldsymbol{\alpha}_{t+1}$ starting from an existing configuration $\boldsymbol{\alpha}_t$ are discovered as follows. First, we identify all closeable switches in $\boldsymbol{\alpha}_t$. Closing any one of these closeable switches i creates exactly one fundamental cycle. Each line segment j in this fundamental cycle can be opened. We store all such switchable pairs (i, j) at time t in a binary 2-D array M^t. $M^t_{ij} = 1$ if (i, j) is a valid switching pair, and is 0 otherwise.

• Policy network π_ϕ structure: The output of the policy network is a 2-D array $\pi_{ij}(S_t)$ and is the probability distribution of switching pairs of branches (i, j), that is, $\pi_{ij}(S_t) \geq 0$ and $\sum_{i=1}^{m} \sum_{j=1}^{m} \pi_{ij}(S_t) = 1$. $\pi_{ij}(S_t)$ must be zero if $M^t_{ij} = 0$. To enforce this, we use a masked softmax layer as the output of the policy network:

$$\pi_{ij}(S_t) = \frac{e^{h_{ij}(S_t)} \cdot M^t_{ij}}{\sum_{kl} e^{h_{kl}(S_t)} \cdot M^t_{kl}} \tag{6.24}$$

where $h_{ij}(S_t)$ are the outputs of the previous layer. M^t_{ij} is the binary mask. The same masked softmax layer is used as the output layer of the parametric generative model g_ω.

• Q-network q_θ structure: The input to the Q-network is the state encoding, along with the one-hot encoding of the closeable switches. The number of outputs of the Q-network equals the number of switches of the distribution network, which correspond to openable switches.

• V-network v_ψ structure: The value network $v_\psi(S_t)$ is a standard multilayer perceptron. The input of the V-network is the state encoding and the output is the value of that state.

6.4 Numerical Studies

To verify the performance of our proposed BCSAC algorithm on dynamic DNR problems, we conduct comprehensive numerical studies on four distribution networks. We start by presenting the experimental data and the algorithm setup in Section 6.4.1-Section 6.4.2. The optimality, scalability, and computation efficiency of the proposed algorithm and benchmark algorithms are shown in Section 6.4.3-Section 6.4.6.

6.4.1 Experimental Data Setup

Distribution Networks

The 16-bus [109], 33-bus [97], 70-bus [143], and 119-bus [144] distribution networks are chosen for the numerical study. The schematic diagram of the 119-bus distribution network is shown in Figure 6.2. For notational convenience, we have modified the bus numbering described in [144]. It is assumed that each line segment has a remotely controllable switch. The total number of feasible configurations is used as a measure of complexity of the learning task and is shown in Table 6.1. The number of feasible configurations are calculated by matrix-tree theorem [19]. Note that the number of feasible configurations increases exponentially with the number of remotely controllable switches. In Table 6.1, the Solar bus column shows the buses with solar generation. For all test cases, the retail electricity price C^l is set as 0.13 \$/kWh. The maximum and minimum nodal voltages are set as $\bar{v} = 1.1$ and $\underline{v} = 0.9$, and the voltage violation penalty is set as $\lambda = C^l$. An alternative modeling approach is to use hard constraints to limit the variations of voltage. For example, constrained policy optimization [126] and constrained soft actor-critic [145] can be implemented to eliminate the need to specify λ. However, these methods are either on-policy or require implementing several additional neural networks. As such, we propose selecting λ based on operational considerations and empirical performance. The switching cost C^s that appeared in (6.1) is also given in Table 6.1.

Table 6.1: Test Distribution Networks

Case	S_{base} (MVA)	Solar buses	C^s (\$)	# configuration
16-bus	100	{11}	4.0	190
33-bus	175	{4,6,12}	0.5	50,751
70-bus	500	{8,10,26,28,50,52}	2.0	22,621,020,015
119-bus	500	{33,45,46,55,80,86,101}	0.8	3,853,525,605,824,176

Nodal Power Data

The time series of load data are taken from the Irish Commission for Energy Regulation Smart Metering Project [146]. The dataset contains one and a half years (76 weeks) of smart meter kWh measurements from approximately 1,000 customers. For each of the test distribution networks, we aggregate the power consumption from 30 (15 for the 70-bus and 119-bus network) customers as the nodal real power injections. We assume a constant power factor of 0.98 lagging. For each of the test networks, the solar generation data are obtained from southern California sites [147]. All nodal power injections are scaled by a common factor β (i.e., $(\boldsymbol{p}_t, \boldsymbol{q}_t) \mapsto (\beta\boldsymbol{p}_t, \beta\boldsymbol{q}_t)$ for all t) to create a realistic network

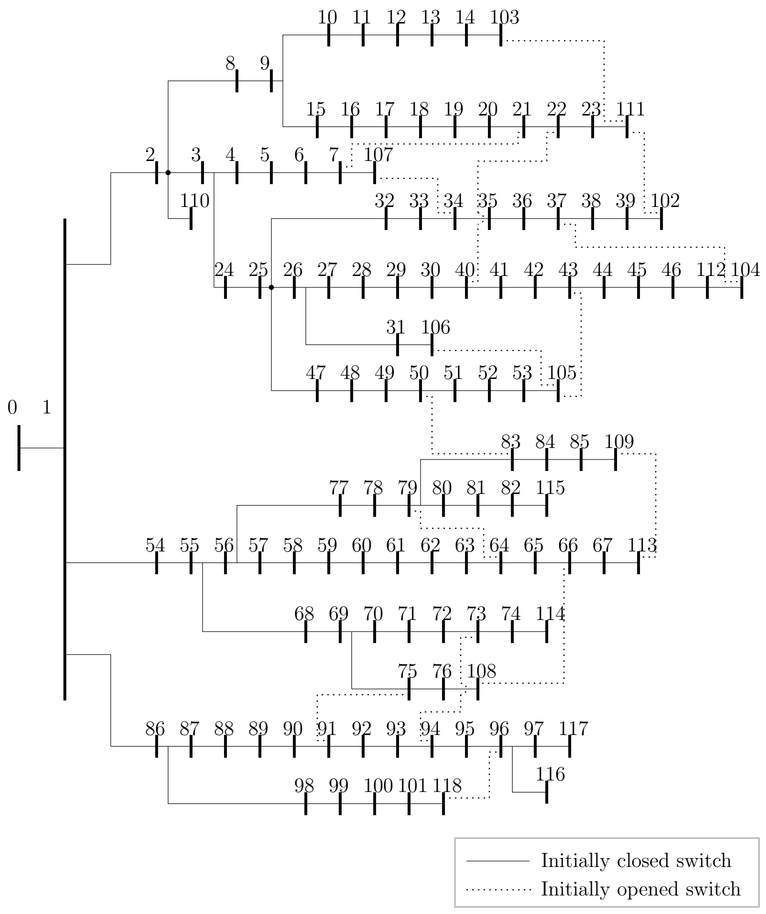


Figure 6.2: The 119-bus test feeder and its initial configuration

loading level. β is chosen such that the resulting average total line losses are roughly 1.5% of the total demand [19]. For all case studies, the first 52 weeks of data are used for training and data of the following week are used for testing.

Network Configuration Data

The last piece of information in the historical operational data is the network configuration data. Unfortunately, we are unable to obtain real world switch configuration data. Thus, the historical configuration data is created by the simulation. In practice, the algorithm will be trained on real world data rather than simulated ones. Therefore, no network parameter information is needed. We create different sets of historical configuration data as follows. At each time step t, the configuration $\boldsymbol{\alpha}_t$ can be changed to $\boldsymbol{\alpha}_{t+1}$ by a single pair of branch-exchange in one of the three scenarios:

s.1 The network is reconfigured by the one-step model-based reconfiguration algorithm assuming inaccurate knowledge of network parameters. To simulate a model-based controller with inaccurate information, we synthesize a different set of line parameters, which deviate from their true values by 10%. We used the mixed-integer conic programming (MICP) formulation in [90] with a time horizon of 1 hour and the number of switching actions of 2 per time step.

s.2 The network configuration is kept the same.

s.3 The configuration is randomly changed to another constraint-satisfying topology.

Scenario 1 represents the active distribution grid reconfiguration performed by a model-based controller with inaccurate information. Scenario 2 corresponds to passive grid management or periods with SCADA system failure, where the network configuration stays the same. Scenario 3 represents periods with isolating faults, when network reconfiguration must be performed to restore power. To create a synthetic network reconfiguration sequence, at time t, we choose a scenario to obtain the new network configuration based on the probability assigned to each scenario. We denote probabilities for the three scenarios as P_{mod}, P_{fix}, and P_{rnd}. By varying these three probabilities, we obtain historical dataset for network configurations with different characteristics. In particular, $P_{mod} = 1$ corresponds to the case where a model-based controller with inaccurate network parameter information is always used to reconfigure the distribution network. The initial configurations $\boldsymbol{\alpha}_0$ of all datasets are the all-tie-switch-open configuration.

6.4.2 Algorithm Setup

The setup of the proposed BCSAC algorithm and two benchmark RL algorithms are summarized in this subsection. The hyperparameters of the BCSAC algorithm and the benchmark DQN and SAC algorithms are provided in Table 6.2. The hyperparameters of the three RL algorithms are tuned individually to reach their best performance. The last row of Table 6.2 shows the parameters shared by all algorithms. Note that we scale the reward (in per unit) to match the weights of neural networks. If not specified otherwise, these parameters will be used for all the numerical studies. Four parameters in the curly brackets are for the three distribution networks, from left to right, 16, 33, 70, and 119-bus, respectively. We also compare the performance of the proposed BCSAC algorithm with that of the historical operational strategy, which is a mix of the model-based, passive, and random control scenarios.

6.4.3 Approximating Behavior Policy by CVAE

This subsection provides the experimental justification of using the CVAE model $g_\omega(a|s)$ to approximate the behavior policy $\pi^b(a|s)$. We first present the performance of CVAE on one of the synthetic datasets. The sample synthetic dataset is obtained with $[P_{mod}, P_{fix}, P_{rnd}] = [0.1, 0.72, 0.18]$ for the 16-bus feeder. We train the CVAE model to approximate the behavior policy. Figure 6.3 shows the ground-truth $\pi^b(a|s = S_{18})$ and

Table 6.2: Hyperparameters of RL Algorithms

DQN	learning rate	$\{10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}\}$
	number of hidden units	$200, 200, 250, 250\}$
	copy steps	$\{30, 30, 30, 30\}$
	minibatch size	$\{32, 64, 64, 64\}$
SAC	τ	$\{0.002, 0.001, 0.0005, 0.0005\}$
	learning rate	$\{5 \cdot 10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}\}$
	number of hidden units	$100, 200, 200, 250\}$
	ρ	$\{0.99, 0.99, 0.99, 0.99\}$
	minibatch size	$\{32, 64, 64, 64\}$
BCSAC	τ	$\{0.1, 10, 25, 50\}$
	learning rate	$\{10^{-4}, 10^{-4}, 5 \cdot 10^{-5}, 5 \cdot 10^{-5}\}$
	number of hidden units	$\{100, 100, 200, 250\}$
	ρ	$\{0.995, 0.995, 0.995, 0.995\}$
	minibatch size	$\{32, 32, 64, 64\}$
CVAE	learning rate	10^{-4}
	number of hidden units	1400
	latent space dimension	$\{20, 40, 60, 70\}$
shared	discount factor	0.95
	number of hidden layers	2
	hidden unit nonlinearity	ReLU
	optimizer	Adam
	reward scale	500

the CVAE approximation $g_\omega(a|s = S_{18})$ for the 18-th time step of the dataset. In Figure 6.3, the (i, j)-th cell of each of the table shows the discrete probability of closing switch i and opening switch j. The cell with the highest probability corresponds to fixing the configuration (Scenario s.2); the cell with the second largest probability corresponds to the branch-exchange obtained from MICP (Scenario s.1); the other cells correspond to randomly changing reconfiguration (Scenario s.3). Cells correspond to infeasible opening/closing pairs (result in non-radial configuration) are left as white.

Figure 6.3 shows that, the CVAE model approximates the behavior policy for all reconfiguration actions a at state S_{18} with high accuracies, *even if the training dataset only*

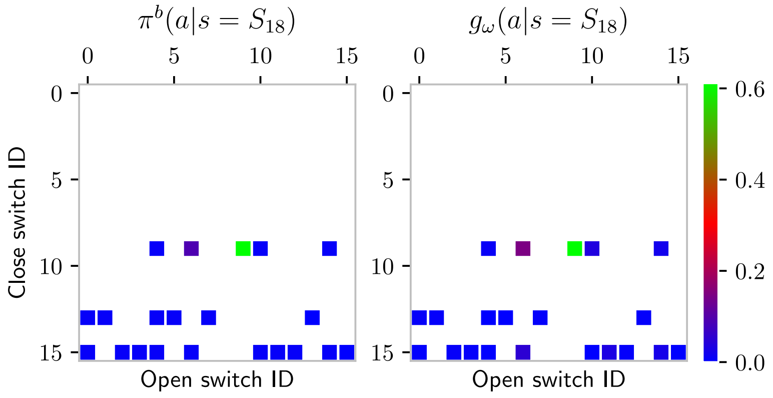


Figure 6.3: A sample result of CVAE on the 16-bus test feeder (TV-distance = 0.19)

contains one reconfiguration action at this state. In other words, the trained CVAE model can generalize the training dataset to unseen state-action pairs.

We use the total variation (TV) distance between $\pi^b(a|s)$ and $g_\omega(a|s)$ to measure their dissimilarity:

$$||\pi^b(\cdot|s) - g_\omega(\cdot|s)||_{\text{TV}} = \frac{1}{2} \sum_{a \in \mathcal{A}(s)} |\pi^b(a|s) - g_\omega(a|s)|$$

In Table 6.3, we report the average TV distance across all states contained in $\mathcal{D}$. Table 6.3 shows that the CVAE model generalizes very well across all states and different data distributions. This makes the CVAE model well suited for training the BCSAC algorithm. This is because during the training process, different actions $\hat{a}$ might be sampled from the policy network (Algorithm 1, Line 4). The CVAE model always yields a good approximation for $\pi^b(\hat{a}|s)$.

Table 6.3: Average TV-Distance Between $\pi^b(a|s)$ and $g_\omega(a|s)$

P_{mod}	0.1	0.3	0.5	0.7	0.9	1.0
16-bus	0.13	0.15	0.13	0.11	0.06	0.03
33-bus	0.24	0.24	0.19	0.15	0.12	0.08
70-bus	0.28	0.26	0.21	0.16	0.08	0.07
119-bus	0.31	0.24	0.17	0.10	0.04	0.01

6.4.4 Optimality and Scalability

We first present the performance of various algorithms on one of the synthetic datasets. More comprehensive evaluations will be provided shortly. The sample synthetic network configuration dataset is obtained with $[P_{mod}, P_{fix}, P_{rnd}] = [0.5, 0.4, 0.1]$ for the 16-bus feeder. During the training process of the RL algorithms, we periodically record the weights of the value and policy neural networks, which are used to evaluate the algorithm performance on the testing week. Experiments with five random historical dataset and neural network initialization and training are conducted. Figure 6.4 shows the cumulative operational cost plus voltage violation penalty over the testing week.

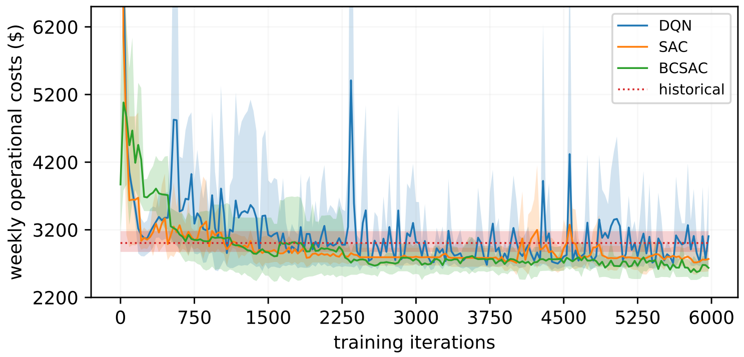


Figure 6.4: 16-bus test feeder

As shown in Figure 6.4, by adopting the proposed BCSAC algorithm, the RL agent is capable of finding a control policy, which yields a lower weekly operational cost than state-of-the-art RL algorithms (DQN and SAC) and historical operational strategy. It should be noted that overfitting could occur in batch RL. This is because the agent is learning from a fixed dataset rather than interacting with the environment. Nevertheless, by using a small-sized neural network and stop training early, we found both of the benchmark and the proposed BCSAC algorithm have little or no overfitting problem as shown in Figure 6.4.

The selection of temperature parameter τ is very important to the BCSAC algorithm. Next, we provide a sensitivity analysis of τ. Consider the same experiment as in Figure 6.4, but with varying τ parameters. The median weekly operational costs of the BCSAC algorithm over 5 independent runs for 5 different temperature parameters, are shown in Figure 6.5.

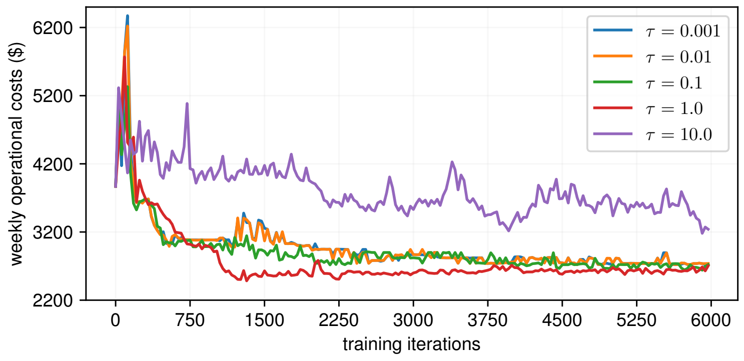


Figure 6.5: Sensitivity of BCSAC to the temperature parameter τ on the 16-bus test feeder

As shown in Figure 6.5, the performance of the proposed algorithm does depend on the temperature parameter when τ is beyond a certain range. The performances of the algorithm are nearly identical when τ varies from 0.001 to 0.1. This suggests that the proposed algorithm is fairly robust with respect to the temperature parameter τ. However, a very large τ does degrade the algorithm performance. This is because the policy is not learned based on the reward but mostly from the behavior policy. In practice, the value of τ should be chosen such that the numerical range of $r(s, a)$ and that of the $|\mathcal{A}(s)|/\tau$ are roughly within the same order of magnitude, where $|\mathcal{A}(s)|$ denotes the size of the action space. Both $|\mathcal{A}(s)|$ and $r(s, a)$ can be calculated based on the historical dataset.

Next, we conduct numerical studies on six other historical operational datasets, which are generated by varying P_{mod} from 0.1 to 1.0 while fixing the ratio of P_{fix} to P_{rnd} at 4. Since our proposed and benchmark RL algorithms are agnostic to the data generation process, the same set of hyperparameters must be used for all test networks and datasets. The testing results on four distribution networks after 6,000 training steps are given in Table 6.4-Table 6.7. As shown in the tables, the proposed BCSAC algorithm outmatches state-of-the-art RL algorithms (DQN and SAC) for most of the experiments in terms of weekly operational costs. It consistently outperforms the RL benchmarks for large test feeders such as the 33-, 70-, and 119-bus feeders. For most of the historical datasets, the BCSAC algorithm improves the behavior policy that generates the dataset.

Table 6.4: Weekly operational costs for 16-bus feeder ($)

P_{mod}	0.1	0.3	0.5	0.7	0.9	1.0
DQN	3268.8	3036.8	3105.4	4448.3	2900.8	3604.8
SAC	2832.6	2715.7	2768.3	3276.8	2595.7	3613.5
BCSAC	2792.7	2642.0	2720.6	2558.4	2466.5	2451.7
Historical	4527.0	3875.2	3004.8	2685.7	2510.4	2384.7

The scalability of our proposed BCSAC algorithm is demonstrated by its performance shown in Table 6.6-Table 6.7 on the 70- and 119-bus distribution network, which has

Table 6.5: Weekly operational costs for 33-bus feeder ($)

P_{mod}	0.1	0.3	0.5	0.7	0.9	1.0
DQN	3613.0	7286.3	3074.7	3502.2	3211.1	6185.6
SAC	4976.0	2796.3	2195.4	2250.1	3658.7	7359.2
BCSAC	2388.6	1921.3	1732.3	1732.3	1716.7	1690.1
Historical	3534.6	2580.0	1961.3	1757.8	1776.4	1686.4

Table 6.6: Weekly operational costs for 70-bus feeder ($)

P_{mod}	0.1	0.3	0.5	0.7	0.9	1.0
DQN	-	6839.6	-	-	-	-
SAC	-	6801.2	3930.0	5522.8	-	4034.6
BCSAC	4143.0	3535.2	3622.1	3331.3	3449.5	3369.3
Historical	6262.0	4643.6	4437.7	3507.2	3453.6	3334.4

more than 3.8 quadrillion feasible configurations. Learning a control strategy with limited historical operational data is extremely difficult on these test cases. This is because, the historical operational data only contain an extremely small subset of all feasible state-action pairs. The DQN and SAC algorithm even fail to learn a dynamic DNR strategy for the highly resistive 70-bus feeder [143]. On the other hand, by learning a batch-constrained policy, our proposed BCSAC algorithm not only outperforms DQN and SAC, but also outmatches the existing behavior control policy, which is a mixed model-based, passive, and random control strategy.

6.4.5 Behavior of BCSAC in Response to Unforeseen States

In this subsection, we test how the trained BCSAC agent would respond to an unforeseen/extreme scenario during the testing time. These scenarios are likely to happen in actual grid operation. For example, power injection patterns might change abruptly due to extreme weather condition or special event. the RL control policy might be overridden by a human operator to perform higher priority tasks such as fault isolation, resulting in an

Table 6.7: Weekly operational costs for 119-bus feeder ($)

P_{mod}	0.1	0.3	0.5	0.7	0.9	1.0
DQN	4952.4	3432.3	7098.4	2881.4	3728.1	5336.5
SAC	3930.0	3102.2	2715.5	2388.0	2743.9	4448.6
BCSAC	3673.1	2432.3	2071.6	2102.9	2164.4	2046.2
Historical	4758.6	2811.8	2323.2	2112.7	2127.0	2046.2

"unfamiliar" network configuration to the RL agent. In any case, the RL agent is expected to perform safely and effectively.

Consider the 16-bus feeder and the same training dataset $[P_{mod}, P_{fix}, P_{rnd}] = [0.7, 0.24, 0.06]$. We train the BCSAC algorithm by the same procedure as described in Section 6.4.4. This trained BCSAC agent is then evaluated on two experiments.

• In the first experiment, we intentionally change the network configuration to some new configurations $\tilde{\boldsymbol{\alpha}}_t$ for each hour t of the testing week. Each of these configurations does not appear in the historical dataset.

• In the second experiment, we change the power injections to some extreme patterns $[\tilde{\boldsymbol{p}}_t, \tilde{\boldsymbol{q}}_t]$ for each hour of the testing week. These patterns are obtained by disconnecting the solar generation and connecting large amount of loads at various buses. The resulting injection patterns deviate significantly from the training dataset's average $[\bar{\boldsymbol{p}}, \bar{\boldsymbol{q}}]$. That is, $||[\tilde{\boldsymbol{p}}_t, \tilde{\boldsymbol{q}}_t] - [\bar{\boldsymbol{p}}, \bar{\boldsymbol{q}}]||_2$ is 1.3 times greater than the largest deviation within the training dataset for a typical t.

The dynamic DNR results for the first and second experiments are shown in upper and lower half of Figure 6.6, respectively.

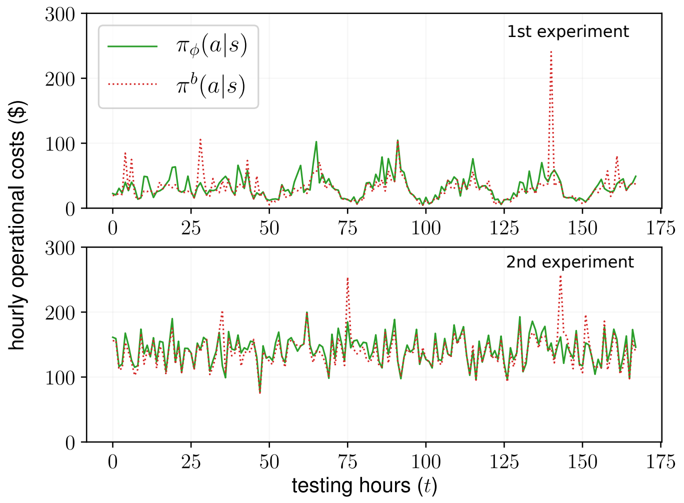


Figure 6.6: BCSAC agent's response to unforeseen states

The green curve represents results from the BCSAC agent; the dotted red curve is the behavior policy defined by $[P_{mod}, P_{fix}, P_{rnd}] = [0.7, 0.24, 0.06]$. Figure 6.6 shows that even if the network configuration is new or the power injection pattern is unfamiliar, the trained BCSAC agent yields lower or nearly the same operational cost as that of the behavior policy. These two experiments show that the BCSAC algorithm is robust against unfamiliar or extreme scenarios.

6.4.6 Computation Speed

This subsection demonstrates the superior computation speed of RL-based control over the model-based control methods. We adopt the MPC-based dynamic DNR algorithm [90] as the model-based benchmark. The MICP is implemented in MATLAB with YALMIP optimization modeling toolbox [148] and MOSEK 9.1 optimization solver. The reinforcement learning algorithms are implemented in Python with TensorFlow 1.14 deep learning framework. They are executed on a desktop with a 4-core Intel i5 3.3GHz CPU and an

Nvidia GeForce GTX 1060 GPU. The training and testing time of all methods are provided in Table 6.8.

Table 6.8: Total Computation Time of Testing Week

		16-bus	33-bus	70-bus	119-bus
Training (seconds)	DQN	15.3	22.4	46.2	71.5
	SAC	84.2	107.4	204.3	410.9
	BCSAC	86.3	112.7	312.4	907.4
	CVAE	222.0	664.5	2040.0	6487.1
Testing (seconds)	DQN	0.2	0.4	0.9	2.1
	SAC	0.2	0.4	1.0	2.2
	BCSAC	0.2	0.4	0.9	2.2
	MICP MPC				
	$H = 1$	63.1	241.6	533.9	1341.4
	$H = 2$	143.6	2439.8	8397.2	–
	$H = 5$	876.3	–	–	–

The training of RL-based algorithms can be done in an off-line manner. Therefore, it is more meaningful to compare the testing time of RL-based and model-based control algorithms. As shown in Table 6.8, the computation time of RL-based algorithms are at least two orders of magnitudes shorter than the model-based control algorithms. The advantage of the RL-based algorithms becomes more pronounced when the size of the distribution network increases. With an optimization horizon H of 5 hours, the optimization solver of the model-based controller fails to converge within one hour.

6.5 Summary

This chapter presents a batch-constrained reinforcement learning algorithm to solve the dynamic distribution network reconfiguration problem. Although state-of-the-art off-policy reinforcement learning algorithms have shown great promise as controllers for power distribution systems, they can have lackluster performance when the training dataset is uncorrelated to the true distribution under the current policy or when the state and action domains are extremely large. To learn an effective control policy for dynamic distribution network reconfiguration problems from a limited historical operational dataset, we develop a batch-constrained soft actor-critic (BCSAC) algorithm, which is trained to minimize both the system operational cost and the discrepancy between the policy under evaluation and the historical operational strategy.

Comprehensive test results on four distribution networks show that the proposed BCSAC algorithm not only outperforms state-of-the-art off-policy RL algorithms but also outmatches or achieves similar level of performance as that of the behavior control policy without any information about the network parameters. The proposed algorithm is also very scalable and has much lower computation time than model-based controllers.

Chapter 7

Multi-Agent Reinforcement Learning for Volt-VAR Control

7.1 Introduction

Volt-VAR control (VVC) determines the operation schedule of voltage regulating and VAR control devices to lower network losses, improve voltage profile, and reduce voltage violations [149]. Traditional VVC adjust the tap positions of the on-load tap changers (OLTC) based on a line drop compensator (LDC), which models the voltage drop of the distribution line from the voltage regulator to the load center. However, the rapid growth of distributed energy resources makes it increasingly difficult to manage the voltage profile on active distribution networks.

In this chapter, we propose a consensus multi-agent RL (C-MARL) algorithm for VVC in power distribution systems, which does not rely on accurate network model and handles state space with higher dimensionality. The proposed framework consists of a group of networked agents managing different VVC devices. Each agent learns two parametric models to approximate the global state value function and the local policy, respectively. These models are trained to maximize the agents' own expected cumulative local rewards, while minimizing the dissimilarity between their neighbors' and their own value functions in a communication-efficient manner. The performance of C-MARL is evaluated on three IEEE test feeders. The experimental results show that our proposed C-MARL algorithm is capable of learning a distributed Volt-VAR control policy that matches the performance of the single-agent RL benchmark. The proposed algorithm is resilient against the failure of individual agents and communications links. Furthermore, the proposed algorithm is much more communication-efficient than the ADMM-based consensus scheme.

The remainder of the chapter is organized as follows: Section 7.2 reviews the existing literature on VVC. Section 7.3 presents the VVC problem formulation. Section 7.4 provides the technical methods. Section 7.5 discusses the setup and results of experimental studies. Section 7.6 provides the summary.

7.2 Prior Work

To address the challenge of distribution system voltage control, a number of physical model-based and data-driven control methodologies have been proposed. The existing literature on VVC problem can be categorized into four groups according to the model assumption and the communication scheme: 1) model-based centralized, 2) model-based distributed, 3) data-driven centralized, and 4) data-driven distributed methods.

Model-based centralized methods assume that all distribution network measurements are collected by a central controller, which also has perfect knowledge of the distribution network parameters. The technical methods to solve the VVC problem include deterministic optimization, robust optimization, and meta-heuristic methods. The deterministic methods include dynamic programming [150], mixed-integer linear programming (MILP) [151], mixed-integer quadratically constrained programming (MIQCP) [149], and bi-level mixed-integer programming [152]. To account for the uncertainties in loads/DGs, robust VVC algorithms [153] [154] [155] have been developed. Meta-heuristic algorithms such as genetic algorithm [156] and particle swarm optimization [157] have been adopted.

To reduce the communication burden and enhance algorithms' resiliency against the failure of the centralized controller, model-based distributed algorithms for VVC have been studied. These methods include simulated annealing [158], distributed decision making [159], and alternating direction method of multipliers (ADMM) considering the continuous relaxation of the discrete variables [160].

Model-based approaches assume complete and accurate physical network model, which are difficult to maintain for regional electric utilities. To overcome this problem, data-driven methods are deployed to determine control actions based on the operational data. A number of data-driven centralized methods have been proposed. In [161], a k-nearest neighbor (kNN) regression model is used to estimate power loss and voltage change in response to the status change of VVC devices. Then, a heuristic approach is taken to determine the appropriate device status. In [162], a support vector regression (SVR) model is trained to approximate the power flow equation. The trained model is then embedded in a model predictive control (MPC) framework to obtain a one-day horizon VVC solution. Reinforcement learning (RL) and deep RL algorithms have also been developed for VVC. A batch RL algorithm that augments the historical dataset and trains a linear approximated action value function is proposed in [163]. The VVC problem is modeled as a constrained Markov decision process (CMDP) [145]. A safe off-policy RL algorithm is developed to avoid voltage violation while minimizing network losses and wear and tear of equipment.

Data-driven centralized methods are particularly advantageous when the distribution network model is unavailable. However, if the central controller fails, then the entire VVC system breaks down. Thus, extending data-driven centralized methods to enable decentralized communication and control will significantly improve the resiliency of the algorithm against individual controller or communication link failure.

Very few data-driven decentralized VVC algorithms have been developed. Reference [164] proposes a multi-agent tabular Q-learning algorithm, in which the agents discover the global reward through a diffusion consensus protocol. Then the local Q values are updated by the standard Q-learning update. Reference [165] developed a multi-agent deep Q-network (DQN) algorithm, which decouples the global action space into individual device's control space. However, the existing methods are either incapable of handling large state space or do not enable coordination between the individual agents.

7.3 Problem Formulation

In this section, we formulate the Volt-VAR control problem as a networked multi-agent Markov decision process (MAMDP). We first introduce the concept of networked

MAMDP and the learning objective, then we discuss the problem formulation of the Volt-VAR control within the networked MAMDP framework.

7.3.1 Basics of MAMDP

A networked MAMDP [166] is a tuple $\mathcal{M} = (\mathcal{S}, \{\mathcal{A}^i\}_{i=1}^K, P, \{r^i\}_{i=1}^K, \mathcal{G}, \gamma)$ which consists of a global state space $\mathcal{S}$, K local action spaces $\mathcal{A}^i$, a global state transition probability $P(s'|s, a^1, a^2, \cdots, a^K)\ \forall s, s' \in \mathcal{S}, \forall a^i \in \mathcal{A}^i$, K local reward functions $r^i(s, a^1, a^2, \cdots, a^K) : \mathcal{S} \times \mathcal{A}^1 \times \mathcal{A}^2 \times \cdots \times \mathcal{A}^K \mapsto \mathbb{R}$, a communication network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and a discount factor γ. In a networked MAMDP, a set of K learning agents select their local actions $A_t^i \in \mathcal{A}^i$ based on the current state $S_t \in \mathcal{S}$ at each discrete time step t. Then each of the agents receives a numerical reward $R_{t+1}^i = r^i(S_t, A_t^1, A_t^2, \cdots, A_t^K)$ and the environment's global state transitions to S_{t+1} based on the state transition probability $P(S_{t+1}|S_t, A_t^1, A_t^2, \cdots, A_t^K)$. Also at time t, each agent i can communicate and share its local information with its neighbors defined in the communication graph $\mathcal{G}$. In this work, we assume $\mathcal{G}$ is connected. The neighbors of agent i are denoted as $\mathcal{V}^i$. For notational simplicity, we denote the joint action and action space as $A_t = [A_t^1, A_t^2, \cdots, A_t^K]$ and $\mathcal{A} = \prod_{i=1}^K \mathcal{A}^i$, respectively. We also denote $R_{t+1} = r(S_t, A_t) = \frac{1}{K}\sum_{i=1}^K R_{t+1}^i = \frac{1}{K}\sum_{i=1}^K r^i(S_t, A_t)$ as the global averaged reward.

The goal of the networked agents is to find each agent's local control policy $\pi^i(a^i|s)$, such that the joint policy $\pi(a^1, a^2, \cdots, a^K|s)$ of all agents maximizes the expected discounted averaged return $J(\pi) = \mathbb{E}[G(\tau)]$, where τ is a trajectory of global states and global actions $S_0, A_0, S_1, A_1, \cdots$, and G is the function that maps a trajectory to the discounted averaged return $G(\tau) = \sum_{t=0}^T \gamma^t \frac{1}{K}\sum_{i=1}^K R_{t+1}^i$. The local policy $\pi^i(a^i|s)$ represents a conditional probability distribution of local actions given the global state s. We assume the global policy is factored as $\pi(a^1, a^2, \cdots, a^K|s) = \prod_{i=1}^{K} \pi^i(a^i|s)$. Two important functions for the multi-agent RL are the global state value function $v_\pi(s)$ and the global action value function $q_\pi(s, a)$ with respect to a given joint policy π. They are defined formally as:

$$v_\pi(s) = \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\sum_{k=0}^{T} \gamma^k R_{t+k+1}|S_t = s\right] \tag{7.1}$$

$$q_\pi(s, a) = \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\sum_{k=0}^{T} \gamma^k R_{t+k+1}|S_t = s, A_t = a\right] \tag{7.2}$$

$v_\pi(s)$ and $q_\pi(s, a)$ capture the expected return committing to a given policy for the starting state s and action a. The optimal policy is thus the one that maximizes $v_\pi(s)$ for all s (or maximizes $q_\pi(s, a)$ for all s, a).

In the next subsection, the distributed VVC problem will be formulated as a networked MAMDP.

7.3.2 Formulate VVC as an MAMDP

In this subsection, we first provide a brief introduction of the proposed multi-agent RL (MARL) VVC framework. Then we present the problem formulation.

We consider a radial distribution network whose node set is denoted as $\mathcal{N}$. The substation is numbered as 0 and all other nodes are numbered as $1, \cdots, n$. The nodal voltage magnitude, real and reactive power at time t of node $i \in \mathcal{N}$ is denoted as V_t^i, p_t^i, and q_t^i, respectively. Vectors $\mathbf{p}_t = [p_t^1, p_t^2, \cdots, p_t^n]$ and $\mathbf{q}_t = [q_t^1, q_t^2, \cdots, q_t^n]$ group all nodal real and reactive power injections except for the substation node.

In this work, three types of VVC devices are considered. Namely, the voltage regulators, on-load tap changers, and the capacitor banks.

• A voltage regulator is placed at the substation (reference node). Thus, the reference voltage of the network at time step t can take on several discrete values $V_t^0 = 1\text{p.u.} + x_t^{\text{reg}} \cdot M^{\text{reg}}$ according to the tap position x_t^{reg} and the fixed step size M^{reg}. The numerical values will be provided in Section 7.5.

• A capacitor bank's reactive power output $q_t^{i,\text{cap}}$ is determined by its on/off status and nodal voltage as: $q_t^{i,\text{cap}} = x_t^{\text{cap}} \cdot M^{\text{cap}} \cdot (V^i)^2$. $x_t^{\text{cap}} \in \{0, 1\}$ denotes the on/off status. M^{cap} denotes the rated reactive power of the capacitor.

• An on-load tap changer (OLTC) is modeled as an ideal transformer with a variable turns ratio. When an OLTC is present on a branch (i, j), its branch power flow is described by (7.3) in the DistFlow equation:

$$(V_t^j)^2/a_t^2 = (V_t^i)^2 - 2r^{ij}p_t^{ij} - 2x^{ij}q_t^{ij} + [(r^{ij})^2 + (x^{ij})^2]l_t^{ij} \tag{7.3}$$

where p_t^{ij} and q_t^{ij} are the branch power flow and l_t^{ij} is the square of branch current. The turns ratio at time t is given by $a_t = 1 + x_t^{\text{tsf}} \cdot M^{\text{tsf}}$, where x_t^{tsf} is the tap position and M^{tsf} denotes the step size.

An overview of the proposed MARL based VVC framework is shown in Figure 7.1. Each agent is associated with one VVC device and determines its own control actions. Therefore, for the substation voltage regulator agent 1, the local action is the discrete tap number $A_t^1 = x_{t+1}^{\text{reg}}$. The subscript $t + 1$ in x_{t+1}^{reg} designates that it is the new tap position after action A_t^1 is taken. The local action spaces for capacitor agents and OLTC agents are defined in a similar manner. The design of the local reward functions should satisfy two

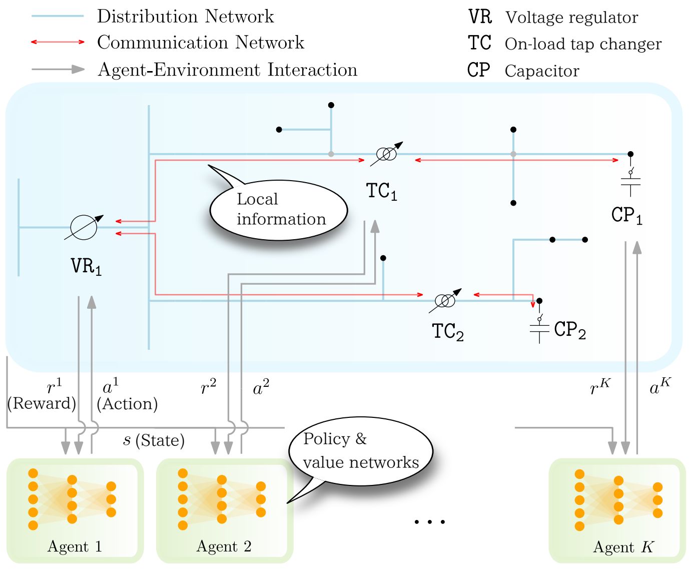


Figure 7.1: The proposed MARL based VVC framework

requirements: 1) the averaged rewards $\frac{1}{K}\sum_{i=1}^{K} r^i(s,a)$ should reflect the networked agents' VVC objective and 2) each local reward must be calculated based on the local metering data received by the corresponding agent. Thus, we define the local reward function as

$$R^i_{t+1} = r^i(S_t, A_t) = -C^l \sum_{\ell \in \mathcal{L}_i} p_t^{l,\ell} - C^s |x^i_t - x^i_{t+1}| - \bar{\lambda} C^i_{t+1} \tag{7.4}$$

where $p_t^{l,\ell}$ is the real power loss on branch ℓ after the joint action A_t is taken; $\mathcal{L}_i$ is the set of branches metered by agent i; C^l and C^s are the costs associated with power loss and devices' switching actions, respectively. x^i_t is the generic term of the discrete control. For example, $x^i_t = x^{\text{reg}}_t$ if agent i is associated with a voltage regulator. The term C^i_{t+1} describes the voltage constraint violation and $\bar{\lambda}$ is the associated penalty factor. The constraint violation is given by (7.5) where $\mathbb{I}$ is the indicator function; $\mathcal{N}_i$ is the set of nodes metered by agent i; V^k_t represents the voltage magnitude followed by the joint action A_t. The exact formulation of the sets $\mathcal{L}_i$ and $\mathcal{N}_i$, as well as various parameters C^l, C^s, and λ will be described for each test feeder in Section 7.5. The global state at time t is defined as $S_t = [\mathbf{p}_t, \mathbf{q}_t, A_{t-1}, t]$.

$$C_{t+1}^i = c^i(S_t, A_t) = \sum_{k \in \mathcal{N}_i} [\mathbb{I}(V_t^k > \bar{V}) + \mathbb{I}(V_t^k < \underline{V})] \tag{7.5}$$

That is, the global state contains the network power injections, the existing VVC devices' status at the previous time step, and a discrete time step t. Finally, we choose a global discount factor γ that is less than one. This completes the formulation of the distributed VVC problem as an MAMDP.

With this MAMDP formulation, we can interpret the value functions (7.1) and (7.2) in terms of VVC as follows: At each time step t, the networked agent's goal is to minimize the long term discounted operational cost and the constraint violation. This long term objective does not easily break into a set of unrelated single time step objectives, because the cost of device switching links the goal of adjacent time steps.

In the next section, we present the technical details of the multi-agent RL algorithm.

7.4 Technical Methods

In this section, we present the proposed consensus multi-agent deep RL-based VVC algorithm. We derive the proposed algorithm in three stages. First, we review the preliminary of centralized off-policy maximum entropy RL framework. Then we reformulate this framework into a distributed multi-agent framework. Finally, we present the proposed communication-efficient C-MARL algorithm to solve the VVC problem.

7.4.1 Off-policy Maximum Entropy RL

In maximum entropy RL, the policy maximizes both the return and the entropy of the policy [138]. In the context of data-driven VVC, the policy entropy maximization is introduced for two reasons. First, without an accurate physical model, all data-driven methods must involve some sort of exploration [161]. That means, it must try different control actions before becoming informed about which of them is the best. To this end, maximum entropy RL provides an efficient and principled way for balancing the exploration and exploitation [125]. Second, an off-policy algorithm can be derived within the maximum entropy RL framework. Off-policy RL algorithms are capable of learning from past experiences so that it can be trained using a much smaller amount of samples collected from the distribution grid. Next, we provide a mathematical characterization of the optimal policy in maximum entropy RL. This is critical to the development of off-policy RL algorithms.

The maximum entropy RL regularizes the reward function by the entropy of the policy $r(s,a)+\alpha H(\pi(\cdot|s))$. α is a temperature parameter that determines the contribution of the entropy to the reward. The state value function in this case is defined as:

$$\mathsf{v}_\pi(s) = \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\textstyle\sum_{k=0}^{T}\gamma^k(R_{t+k+1}+\alpha H(\pi(\cdot|S_{t+k})))|S_t=s\right] \tag{7.6}$$

where $\mathsf{v}_\pi(s)$ denotes the entropy regularized value function. It follows from (7.6) that a policy, which maximizes $\mathsf{v}_\pi(s)$ is maximizing the combined return and policy entropy. The latter maintains a certain level of stochasticity of the policy. Thus we can balance the exploration and exploitation by following the current policy π throughout the learning process.

The Bellman equation for $\mathsf{v}_\pi(s)$ is derived as (7.7) The definition of entropy is used to derive the second equality. The optimal entropy-regularized state value function is defined as $\mathsf{v}_*(s) = \max_\pi \mathsf{v}_\pi(s)$. Similarly, the optimal entropy-regularized policy π^* is defined as the one whose entropy-regularized value function is $\mathsf{v}_*(s)$.

$$\begin{aligned} \mathsf{v}_\pi(s) &= \mathbb{E}_{a\sim\pi}\mathbb{E}_{s'\sim P}\left[r + \gamma \mathsf{v}_\pi(s')\right] + \alpha H(\pi(\cdot|s)) \\ &= \mathbb{E}_{a\sim\pi}\left[r + \gamma \mathbb{E}_{s'\sim P}\left[\mathsf{v}_\pi(s')\right] - \alpha \log \pi(a|s)\right] \end{aligned} \tag{7.7}$$

We term the two-tuple (v_*, π^*) an optimality pair, which is shown to be the solution to the off-policy consistency equation [139, Corollary 21]:

$$\mathsf{v}(s) = r(s,a) + \gamma \mathbb{E}_{s'\sim P}\left[\mathsf{v}(s')\right] - \alpha \log \pi(a|s) \quad \forall s, a \tag{7.8}$$

7.8 characterizes the optimal policy and motivates our subsequent algorithm developments. However, it is very challenging to solve 7.8 directly due to high-dimensional and continuous state space. In addition, 7.8 is stated in a centralized format. In the next subsection, we derive a distributed and off-policy algorithm to approximate the solution to 7.8 in a sample-efficient manner. To make the learning tractable, we restrict the class of functions we consider for the value function and the policy function.

7.4.2 Distributed Optimization

In this subsection, we transform the problem of finding optimal local policy and state value function as a distributed consensus optimization problem of the following form:

$$\min_{\boldsymbol{w}} \quad \textstyle\sum_{i=1}^{K} J^i(w^i) \quad \text{s.t. } w^1 = w^2 = \cdots = w^K \tag{7.9}$$

where J^i are the local objective functions. (7.9) appears ubiquitously in distributed adaptive learning [167], distributed algorithms for linear algebraic systems [108], and distributed parameter estimation [168].

We first approximate the solution of optimal policy and value function with the following stochastic nonlinear program, which is commonly done in deep RL literature [127]:

$$\min_{\mathsf{v},\pi} \underset{s,a\sim\mathcal{D}}{\mathbb{E}} \left(\mathsf{v}(s) - \{r + \gamma\mathbb{E}_{s'\sim P}\left[\mathsf{v}(s')\right] - \alpha\log\pi(a|s)\}\right)^2 \tag{7.10}$$

$\mathcal{D}$ is the data distribution, which will be approximated by an experience replay buffer [127]. Next, we define w^i in (7.10) as each agent's local copy of the global value and policy functions $\mathsf{v}^i(s)$ and $\pi^i(a|s)$. At optimality, these local functions need to reach consensus. Thus $\mathsf{v}^i = \mathsf{v}^j$ and $\pi^i = \pi^j, \forall i, j$ constitute the constraints in (7.9). Although the joint policy $\pi^i(a|s)$ is maintained by all agents, only the i-th coordinate $\pi^i(a^i|s)$ (the i-th local action) is actuated by agent i. Note that v^i and π^i are infinite dimensional.

Now to decompose the objective function in (7.10), For notational simplicity, we first declare two sets of functions for later derivations. Let $f(s,a) = \mathsf{v}(s) - \gamma\mathbb{E}_{s'\sim P}\left[\mathsf{v}(s')\right] + \alpha\log\pi(a|s)$ and $f^i(s,a) = \mathsf{v}^i(s) - \gamma\mathbb{E}_{s'\sim P}\left[\mathsf{v}^i(s')\right] + \alpha\log\pi^i(a|s)$. $\zeta(s,a) = [\mathsf{v}(s), \pi(a|s)]^T$

and $\zeta^i(s,a) = [\mathsf{v}^i(s), \pi^i(a|s)]^T$. Then, we can rewrite the minimization problem (7.10) as follows:

$$\begin{aligned}
&\underset{\zeta}{\operatorname{argmin}} \quad \mathop{\mathbb{E}}_{s,a,r\sim\mathcal{D}} \big(f(s,a) - r\big)^2 && (7.11)\\
=&\underset{\zeta}{\operatorname{argmin}} \quad \mathop{\mathbb{E}}_{s,a,r\sim\mathcal{D}} f(s,a)^2 - 2rf(s,a) + r^2 && (7.12)\\
=&\underset{\boldsymbol{\zeta}\in\Omega}{\operatorname{argmin}} \quad \mathop{\mathbb{E}}_{s,a,r^i\sim\mathcal{D}} \frac{1}{K}\sum_{i=1}^{K} f^i(s,a)^2 - \\
&\qquad\qquad \frac{1}{K}\sum_{i=1}^{K} 2r^i f^i(s,a) + \frac{1}{K}\sum_{i=1}^{K} (r^i)^2 && (7.13)\\
=&\underset{\boldsymbol{\zeta}\in\Omega}{\operatorname{argmin}} \quad \mathop{\mathbb{E}}_{s,a,r^i\sim\mathcal{D}} \frac{1}{K}\sum_{i=1}^{K} \big(f^i(s,a) - r^i\big)^2 && (7.14)
\end{aligned}$$

where $\boldsymbol{\zeta} = [(\zeta^1)^T, (\zeta^2)^T, \cdots, (\zeta^K)^T]^T$. Ω is the set containing all $\boldsymbol{\zeta}$ such that $\zeta^1 = \zeta^2 = \cdots = \zeta^K$. Using the degree matrix D and the adjacency matrix A of $\mathcal{G}$, the constraints can be rewritten as $(D_{ii} \otimes I_2)\zeta^i = (A_i \otimes I_2)\boldsymbol{\zeta}$, $\forall i$. I_2 is the identity matrix of size 2, $\otimes$ designates Kronecker product, and A_i denotes the ith row of A. We will use the notations $\bar{D} = D \otimes I_2$ and $\bar{A} = A \otimes I_2$, with $\bar{D}_{ii}$ and $\bar{A}_i$ being understood as the ii-th block and i-th block row, respectively. (7.14) decomposes the global learning objective and is compatible with the MAMDP model. Specifically, each agent i receives the local reward r^i and takes local actions a^i. The consensus is achieved through neighbor-to-neighbor communication.

To derive a tractable learning algorithm for optimization problem (7.14), we parameterize v^i and π^i as function approximators such as deep neural networks (NN): $\mathsf{v}_{\psi_i} \approx \mathsf{v}^i$ and $\pi_{\phi_i} \approx \pi^i$ with the parameters of deep NNs denoted by $\varphi_i = [\psi_i, \phi_i]$. Additionally, we use a separate target network $\mathsf{v}_{\bar{\psi}_i}$ for evaluating $\mathsf{v}^i(s')$ [138]. We denote the $\varphi_i, \bar{\psi}_i$-

parameterization of f^i and ζ^i as $\bar{f}_{\varphi_i}$ and ζ_{φ_i}. Therefore, (7.14) can be rewritten as (7.15), where $\boldsymbol{\zeta_\varphi} = [\zeta_{\varphi_1}^T, \zeta_{\varphi_2}^T, \cdots, \zeta_{\varphi_K}^T]^T$. Note that, (7.15) has a finite number of decision variables and infinitely many constraints.

$$\begin{aligned} \min_{\varphi_i} \quad & \mathop{\mathbb{E}}_{s,a,r^i,s' \sim \mathcal{D}} \frac{1}{K} \sum_{i=1}^{K} \left(\bar{f}_{\varphi_i}(s,a,s') - r^i \right)^2 \\ \text{s.t.} \quad & \bar{D}_{ii} \zeta_{\varphi_i}(s,a) = \bar{A}_i \boldsymbol{\zeta_\varphi}(s,a) \quad \forall i \in \mathcal{V}, s, a \end{aligned} \tag{7.15}$$

This infinite constraint set can be reformulated as a finite one, $\varphi_i = \varphi_j$, $\forall i, j$. Several methodologies such as ADMM [169] and diffusion adaptation strategies [167] can be used to solve (7.15) with the finite constraint set. However, it is extremely costly to communicate the full set of parameters of deep neural networks. To address this problem, in the next subsection, we leave the constraints as they are and derive a stochastic approximation type algorithm to solve (7.15), which significantly improves the communication efficiency.

7.4.3 Communication-Efficient Multi-Agent Policy Consensus

The goal of this subsection is to approximate the solution to (7.15) by randomization. That is, we randomly enforce a subset of all constraints in each iteration. First, we adopt the model for semi-infinite programming in [170] to convert the infinite constraint set into a finite one. Specifically, we approximate the constraints represented in (7.15) as the following stochastic programming representation:

$$\int_{s,a \in \mathcal{S} \times \mathcal{A}} \mathbf{h}\big(\bar{D}_{ii} \zeta_{\varphi_i}(s,a) - \bar{A}_i \boldsymbol{\zeta_\varphi}(s,a)\big) d\mu_{\mathcal{S}\mathcal{A}} = \mathbf{0} \tag{7.16}$$

for all $i \in \mathcal{V}$. $\mathbf{h}(x, y) = [h(x), h(y)]$ stacks two penalty functions h, which satisfies $h(0) = 0$ and $h(x) > 0, \forall x \neq 0$. $\mu_{\mathcal{SA}}$ is a probability measure defined on the global state-action space $\mathcal{S} \times \mathcal{A}$. Assuming the continuity of the function ζ_{φ_i} and h, as well as full support assumption of μ_{SA}, we can establish the following propositions:

Proposition 7.4.1 *Let d be a metric on $\mathcal{S} \times \mathcal{A}$. Assume ζ_{φ_i} is continuous with respect to d for every φ_i and h is continuous with respect to the Euclidean metric. Also assume $\mu_{\mathcal{SA}}(X) > 0$ for every non-empty open subset $X \subseteq \mathcal{S} \times \mathcal{A}$. Then the constraint set in* (7.15) *is equivalent to* (7.16).

Proposition 7.4.2 *With the same assumptions in Proposition 7.4.1 except for the continuity assumption about ζ_{φ_i}. Then every feasible point of* (7.16) *satisfies most of the constraints in* (7.15)*, except for a subset of measure zero.*

Proposition 7.4.1 and Proposition 7.4.2 are theoretically reassuring. In practice, $\mu_{\mathcal{SA}}$ will be approximated by $\mathcal{D}$, which is the data distribution in (7.15). The proofs of the propositions can be found in Appendix C.1

Consider the quadratic penalty for non-consensus $h(x) = \frac{1}{2}x^2$. Under this approximated stochastic programming representation, the Lagrangian of (7.15) is given by:

$$\mathcal{L}(\boldsymbol{\varphi}, \boldsymbol{\lambda}) = \frac{1}{K}\sum_{i=1}^{K} \mathcal{L}^i(\boldsymbol{\varphi}, \lambda_i) \tag{7.17}$$

$$\begin{aligned}\mathcal{L}^i(\boldsymbol{\varphi}, \lambda_i) = \mathop{\mathbb{E}}_{s,a,r^i,s' \sim \mathcal{D}} \Big(& \big(\bar{f}_{\varphi_i}(s, a, s') - r^i\big)^2 \\ & + \frac{\lambda_i}{2} ||\bar{D}_{ii}\zeta_{\varphi_i}(s, a) - \bar{A}_i \boldsymbol{\zeta}_{\boldsymbol{\varphi}}(s, a)||^2 \Big)\end{aligned} \tag{7.18}$$

The primal variables $\boldsymbol{\varphi}$ and the multipliers $\boldsymbol{\lambda}$ can be solved by the primal-dual method. However, we found that using a fixed $\boldsymbol{\lambda}$ parameter achieves good empirical performance. The detailed value for the multipliers will be provided in Section 7.5. (7.18) has a tractable sample gradient and can be readily tackled by established deep learning routines such as stochastic gradient descent (SGD). Specifically, each agent performs the minimization of sample-estimated $\mathcal{L}^i(\boldsymbol{\varphi}, \lambda_i)$. In addition, similar to the adapt-then-combine (ATC) algorithm [171], we first perform the minimization of the first term in (7.18), then use the immediately updated weights to evaluate and minimize the second term:

$$\tilde{\varphi}_i^{\nu} = \varphi_i^{\nu} - \eta \nabla_{\varphi_i} \big(\bar{f}_{\varphi_i^{\nu}}(s, a, s') - r^i \big)^2 \tag{7.19}$$

$$\varphi_i^{\nu+1} = \tilde{\varphi}_i^{\nu} - \eta \frac{\lambda_i}{2} \nabla_{\varphi_i} || \bar{D}_{ii} \zeta_{\tilde{\varphi}_i^{\nu}}(s, a) - \bar{A}_i \boldsymbol{\zeta}_{\boldsymbol{\varphi}^{\nu}}(s, a) ||^2 \tag{7.20}$$

$$\bar{\psi}_i^{\nu+1} = \rho \bar{\psi}_i^{\nu} + (1 - \rho) \psi_i^{\nu+1} \tag{7.21}$$

where ν is the iteration count and ρ is an exponential smoothing parameter. s, a, r^i, s' are sampled data from the experience replay $\mathcal{D}$, which is assumed to be initialized by the historical data. When conducting the update in (7.20), a communication of each agent i with its neighbors is established. The information being transmitted includes s, a and $\zeta_{\varphi_j}(s, a)$. The full algorithm is summarized in Algorithm 2. The proposed C-MARL algorithm proceeds as follows: First all agents initialize their deep NN parameters. Then the agents communicate and update their local policy and value functions according to the scheme described in (7.19)-(7.21). We let each agent communicate and update C times (on average) between adjacent control actuation steps t and $t+1$. At time t, all agents take their control actions

Algorithm 2 C-MARL for VVC

```
Input: Historical dataset $\mathcal{D}$, update frequency $C$, communication graph $\mathcal{G}$
1: for $i = 1, \cdots, K$ do
2:     Initialize $\varphi_i^0 = [\psi_i^0, \phi_i^0], \bar{\psi}_i^0$
3: for $\nu = 0, \cdots,$ do
4:     Sample $i$ from $[1, 2, \cdots, K]$ uniformly
5:     Sample mini-batch $\mathcal{B} = \{(s, a, r^i, s')\}$ from $\mathcal{D}$
6:     Update $\varphi_i^\nu$ by (7.19)
7:     Collects $\zeta_{\varphi_j}(s, a)$ from $i$'s neighbors.
8:     Update $\varphi_i^\nu$ by (7.20)
9:     Update $\bar{\psi}_i^\nu$ by (7.21)
10:    if $\mathrm{mod}(\nu, K \cdot C) = 0$ then
11:        for $i = 1, \cdots, K$ do
12:            Take control actions $A_t^i \sim \pi_{\phi_i^{\nu+1}}(\cdot|S_t)$
13:            $\mathcal{D} = \mathcal{D} \cup \{(S_t, A_t^i, R_{t+1}^i, S_{t+1})\}$
```

(tap positions of the voltage regulating devices), and store the transition information into the experience replay buffer $\mathcal{D}$.

7.4.4 Algorithm Implementation

This subsection provides additional implementation details for the proposed C-MARL VVC algorithm. We will discuss the NN architecture design and variable encoding in these NNs.

• $\mathsf{v}_{\psi_i}(s)$ and $\mathsf{v}_{\bar{\psi}_i}(s)$ networks: the value networks are standard multilayer perceptrons whose inputs are the global state s and the outputs are the value of that state.

• $\pi_{\phi_i}(a|s)$ networks: we adopt the device-decoupled network structure [145], which divides the outputs of the policy network into K groups. The output neurons in each group corresponds to the local action space $|\mathcal{A}^i|$ for each device. In addition, we adopt the ordinal encoding layer [172] for each group to represent the order information of the devices' tap positions. The hidden layers are shared by all groups.

• Encoding the global time step t: in this study, we only encode the hour-of-week part of the global time step t, which ranges from 0 to 167. t is encoded in two coordinates $[\cos(2\pi t/168), \sin(2\pi t/168)]$ to reflect its periodic nature.

7.5 Numerical Study

The numerical studies of the proposed C-MARL algorithm are conducted on three test feeders. The experimental setup for the three test feeders are provided in Section 7.5.1. The sample efficiency, communication efficiency, and resiliency of the proposed algorithm are validated in Section 7.5.3-Section 7.5.4.

7.5.1 Numerical Setup

Distribution Networks and Nodal Power Data

The IEEE 4-bus, 34-bus, and 123-bus distribution test feeders [38] are used in the numerical studies. The VVC devices are setup on these test feeders as follows: For all test feeders, a voltage regulator ($\mathtt{VR}_1$) is located at the substation node and controls the reference voltage. Voltage regulators have 21 tap positions with step size $M^{\text{reg}} = 0.005$, which evenly divides the turns ratios between 0.95 and 1.05. We assume the same tap position configuration for the OLTCs. For the 4-bus feeder, an OLTC is placed between node 2 and 3 ($\mathtt{TC}_1$) and a capacitor with rating $M^{\text{cap}} = 200$ kVar is placed at node 4 ($\mathtt{CP}_1$). For the 34-bus test feeder, two OLTCs are placed between node 814 and 850 ($\mathtt{TC}_1$), and node 852 and 832 ($\mathtt{TC}_2$). Two capacitors are placed at node 844 ($\mathtt{CP}_1$: 100 kVar) and node 847 ($\mathtt{CP}_2$: 150 kVar). For the 123-bus, three OLTCs are placed between node 10 and 15

($\mathtt{TC}_1$), node 67 and 160 ($\mathtt{TC}_2$), and node 25 and 26 ($\mathtt{TC}_3$). Four capacitors are placed at node 83 ($\mathtt{CP}_1$: 200 kVar), node 88 ($\mathtt{CP}_2$: 50 kVar), node 90 ($\mathtt{CP}_3$: 50 kVar), and node 92 ($\mathtt{CP}_4$: 50 kVar). The initial turns ratios of voltage regulators and OLTCs are 1. Initially, the capacitors are switched off.

The time series of hourly load data are obtained from the London smart meter dataset [173]. The dataset contains one year of half-hourly smart meter kWh measurements from approximately 5,000 customers. The measurements are aggregated and scaled to match the test feeders' loading level. The final load data have the same spatial load distribution and power factors as that of the IEEE standard test cases.

Local Reward Setups and Communication Networks

The parameters that appear in the local reward (7.4)) and local operation constraint violation (7.5) are as follows: For all test cases, the cost of electricity, cost per switching action, and the constraint violation penalty are set as $C^l = \$0.04/\text{kWh}$, $C^s = \$0.1$, and $\bar{\lambda} = 2C^l$, respectively. The voltage bounds are $\bar{V} = 1.05$ and $\underline{V} = 0.95$ p.u. The capacitors meter the voltage at its own node and the line real power loss within one-degree neighbors. The voltage regulators meter the voltage at the first downstream node from the substation. The OLTCs meter the voltage at its first downstream node and the power loss on its branch. A fixed (time-invariant) communication graph is assumed for each of the test feeders. The neighbor-to-neighbor relationships are summarized in Table 7.1. Each line represents a bi-directional communication link.

Table 7.1: Communication Networks

4-bus	34-bus	123-bus
$\begin{tikzcd} \mathtt{VR}_1 \arrow[r, dash] & \mathtt{TC}_1 \arrow[r, dash] & \mathtt{CP}_1 \end{tikzcd}$	$\begin{tikzcd} \mathtt{VR}_1 \arrow[r, dash] \arrow[d, dash] & \mathtt{TC}_1 \arrow[r, dash] \arrow[d, dash] & \mathtt{TC}_2 \\ \mathtt{CP}_1 \arrow[r, dash] & \mathtt{CP}_2 & \end{tikzcd}$	$\begin{tikzcd} \mathtt{VR}_1 \arrow[r, dash] \arrow[d, dash] & \mathtt{TC}_1 \arrow[r, dash] \arrow[dd, dash] \arrow[dr, dash] & \mathtt{TC}_2 \\ \mathtt{CP}_1 \arrow[rr, dash] \arrow[dr, dash] & & \mathtt{TC}_3 \arrow[d, dash] \\ \mathtt{CP}_2 \arrow[r, dash] & \mathtt{CP}_3 & \mathtt{CP}_4 \end{tikzcd}$

7.5.2 Algorithm Setup

In the numerical studies, we compare the performance of our proposed algorithm with two benchmarks: the single-agent SAC [138] and the multi-agent off-policy RL using the linearized ADMM consensus strategy [174, Algorithm 1]. The single-agent SAC serves as a stability baseline and the ADMM is used for comparison purpose.

• For the single-agent SAC, the reward is defined as the average of the local rewards. The agent's action is defined as the union of the local actions.

• For the linearized ADMM, the optimization variables for each agent are the deep NN parameters φ_i. We maintain a separate deep NN $\zeta_{\underline{\varphi}_i}$, whose structure is the same as ζ_{φ_i} and the parameters $\underline{\varphi}_i$ are the local dual variables. The same target network construct for evaluating $\mathsf{v}(s')$ is adopted.

The hyperparameters of the algorithms are provided in Table 7.2. The hyperparameters of the algorithms are tuned individually to reach their best performance. The last row of Table 7.2 shows the parameters shared by all algorithms. If not specified otherwise, these parameters will be used for all the numerical studies. Three parameters in the curly brackets are for the three distribution networks, from left to right, 4-bus, 34-bus, and 123-bus, respectively.

Table 7.2: Hyperparameters of Benchmark and Proposed Algorithms

SAC	temperature parameter α	$\{0.5, 0.2, 0.1\}$
	learning rate	0.001
	number of hidden units	$\{64, 80, 128\}$
	smoothing parameter ρ	0.99
	minibatch size	16
ADMM	temperature parameter α	$\{0.5, 0.2, 0.1\}$
	c in [174]	1
	ρ in [174]	500
	number of hidden units	$\{32, 64, 64\}$
	smoothing parameter ρ	0.99
	minibatch size	16
C-MARL	temperature parameter α	$\{0.5, 0.2, 0.1\}$
	learning rate	0.001
	number of hidden units	$\{32, 64, 128\}$
	smoothing parameter ρ	0.99
	minibatch size	16
shared	discount factor	0.95
	update frequency C	1
	consensus parameter λ_i	1
	number of hidden layers	2
	hidden unit nonlinearity	tanh
	optimizer	Adam
	reward scale	5

7.5.3 Stability, Sample Efficiency, and Communication Efficiency

In this subsection, we report the stability, sample efficiency, and communication efficiency of the proposed and benchmark VVC algorithms. The average of the hourly rewards in (7.4) and the average of the constraint violations in (7.5) versus the number of training samples and the number of transmitted data points are shown in Figure 7.2-Figure 7.4. The horizontal axis beneath the plots shows the number of training samples of the form $(S_t, A_t, R_{t+1}, S_{t+1})$. For the proposed C-MARL algorithm, the data being transmitted include the global time steps $\{t\}$ and the corresponding values $\mathsf{v}_{\psi_i}(\{S_t\}), \pi_{\phi_i}(\{A_t\}|\{S_t\})$ of the mini-batch; for the ADMM consensus strategy, the data being transmitted are the neural network weights $\varphi_i = [\psi_i, \phi_i]$. For all figures, the solid curve represents the median of five independent runs; the shaded areas are the upper and lower error bounds.

As shown in Figure 7.2-Figure 7.4, all three algorithms' performances stabilize after a certain amount of training samples are collected and used for training. Our proposed algorithm achieves a similar level of performance as the single-agent benchmark in all test cases in terms of hourly reward and constraint violation. This demonstrates the effectiveness of the proposed randomized consensus protocol. The proposed algorithm yields significant improvement on communication efficiency compared with the ADMM consensus protocol. In addition, the communication cost of our proposed algorithm stays constant across the test feeders. This is because only the sample data are transmitted. The communication burden in the ADMM consensus strategy grows quickly with the size of the physical network and the number of agents.

7.5.4 Resiliency Against Agent and Communication Link Failure

One key advantage of distributed algorithms over the centralized ones is that when an individual agent or communication link fails, the rest of the system can continue to function. In this subsection, a few experiments are carried out to evaluate the proposed algorithm's resiliency against failures of individual components. Two types of component failures are considered:

E.1 An agent experiences an internal error so that the computation and control cannot be properly executed. However, it is still able to communicate with its neighbors. In

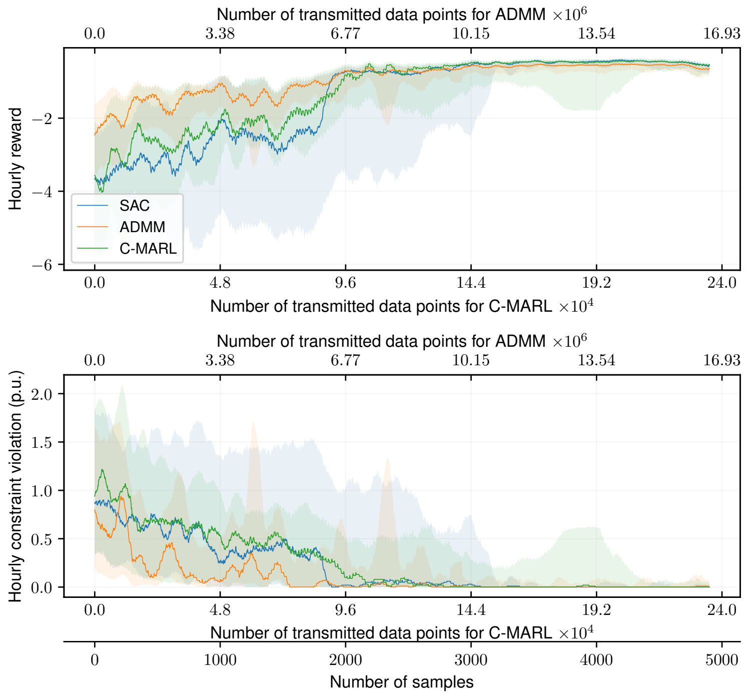


Figure 7.2: Hourly reward and voltage violation of 4-bus feeder

this scenario, the agent freezes the tap position of its device and stops training, while other agents continue their controls and training.

E.2 A communication link is temporarily down. If the overall communication graph is still connected, then the agents function normally except for the altered communication graph connectivity. If the overall communication graph is disconnected, then the distribution network becomes partially observable. In this case, the agents will create a replacement state $\hat{S}_t = [\hat{\boldsymbol{p}}_t, \hat{\boldsymbol{q}}_t, \hat{A}_{t-1}, t]$ and take action based on $\hat{S}_t$. The nodal power $\hat{\boldsymbol{p}}_t, \hat{\boldsymbol{q}}_t$ are obtained from the historical average; the joint actions $\hat{A}_{t-1}$ are sampled from the agent's own policy network. Please note that each agent maintains a local copy of the joint policy network. The agent's experience involving replacement states won't be stored in the replay memory $\mathcal{D}$.

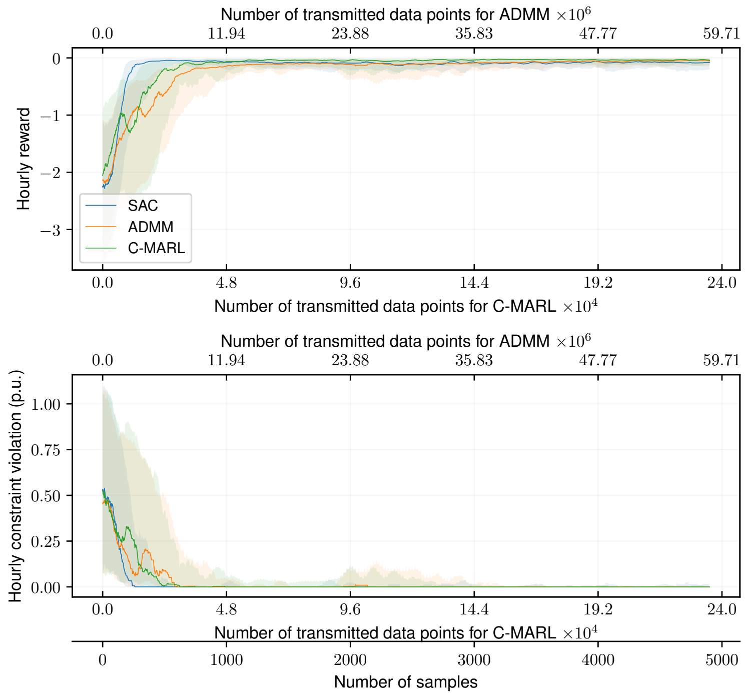


Figure 7.3: Hourly reward and voltage violation of 34-bus feeder

We create two sets of experiments for the test feeders to demonstrate the consequences of the two types of failures E.1 and E.2. For all experiments, the occurrences of component failures are assumed to follow a Poisson process with rate $\lambda = \frac{1}{168}$. That is, the inter-event times are independent exponential random variables with scale parameter $\beta = \frac{1}{\lambda} = 168$ (hr). The duration of each failure is assumed to follow the geometric distribution with success probability 0.2. For the first experiment, all agents are assumed to have an equal chance of failure. The communication link failures in the second experiment are treated similarly.

Simulation results for the two experiments are shown in Figure 7.5. Each experiment occupies one column. The blue curves represent the failure scenarios. The orange curves represent the corresponding counterfactual experiment, which has the identical simu-

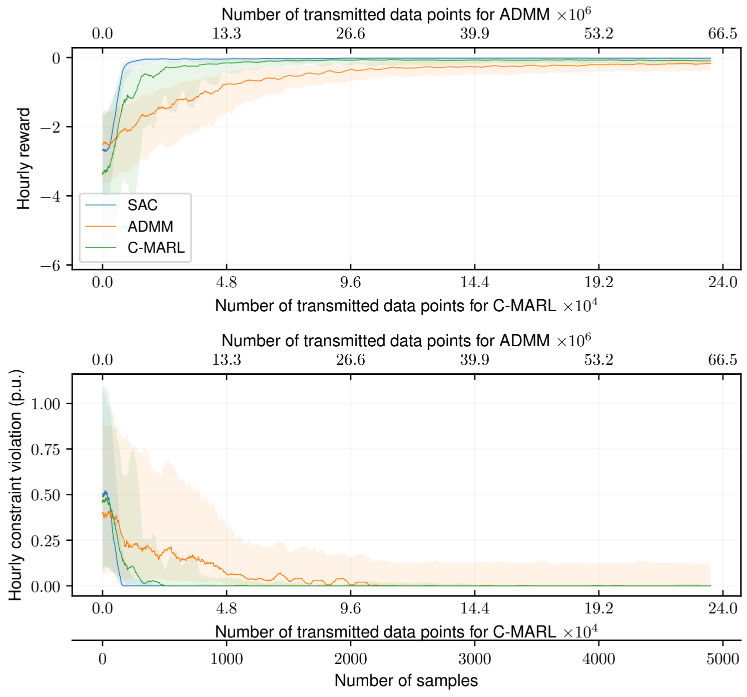


Figure 7.4: Hourly reward and voltage violation of 123-bus feeder

lation setup but without agent or communication failure. Figure 7.5 shows that the proposed algorithm is resilient facing random agent or communication link failures. The algorithm performance degradation is negligible if the time to clear component failure is not too long. The impact of agent or communication failure on long-term algorithm performance is much smaller than that of short-term performance.

7.6 Summary

This chapter proposes a multi-agent reinforcement learning algorithm to solve the Volt-VAR control problem in power distribution systems. We extend the centralized off-policy maximum entropy RL framework to a networked multi-agent MDP model. A

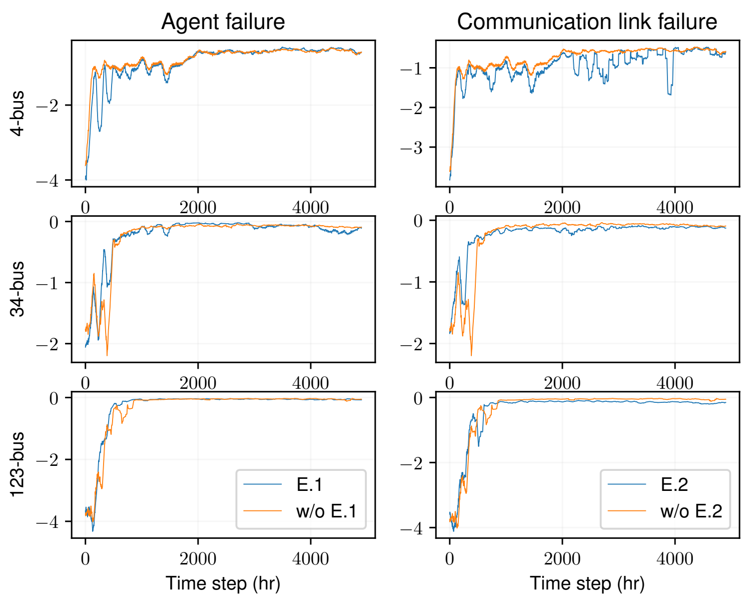


Figure 7.5: Hourly reward under agent/communication link failure

randomization-based consensus algorithm is developed to solve the networked multi-agent MDP. Our proposed algorithm is decentralized and fully data-driven, which enables the control of voltage regulating devices without a central controller or knowledge of the distribution network topology and parameter information. Numerical study results of a comprehensive set of IEEE test feeders show that the proposed algorithm achieves a similar level of performance as the centralized RL benchmark. Our proposed algorithm is much more communication efficient than existing consensus strategy such as ADMM. Moreover, our proposed algorithm is resilient against communication link and agent failure as demonstrated by the simulation results.

Chapter 8

Conclusions

8.1 Summary of Thesis

In this dissertation, four use cases and applications of smart grid monitoring and control are developed by combining the merits of model-based and data-driven methods. First, the unbalanced distribution system state estimation problem with low measurement redundancy is addressed. Second, a physically-inspired data-driven method for distribution system anomaly detection is developed in an unsupervised manner. Third, model-based distributed and data-driven centralized reinforcement learning algorithms are developed to solve the network reconfiguration problem. Fourth, multi-agent reinforcement learning based Volt-VAR control algorithm is developed. The achievements of this dissertation work are summarized as follows:

- In Chapter 2, we showed how constrained maximum likelihood can restore the observability for low measurement redundant systems state estimation problem. We also introduced a novel sparse subspace Gauss-Newton algorithm to significantly improve the numerical robustness of the state estimation algorithm. We also provided the method to compute the uncertainty estimate in the subspace Gauss-Newton algorithm.

- In Chapter 3, we established a link between the linear power flow model and a modified linear regression model to propose a physically interpretable data-driven anomaly detection framework. Additional insights for the relationship between smart meter voltage and real power consumption measurements are obtained by analyzing the two-phase distribution secondary circuits. Also, the behavior of the modified linear regression model under normal and abnormal data are mathematically characterized.

- In Chapter 4, we adopted the ADMM-Release-and-Fix algorithm to solve the distributed mixed-integer network reconfiguration problem. We innovated the ADMM-Fix step by introducing an approximated Newton's iteration, which significantly improved the convergence speed.

- In Chapter 5-Chapter 6, we developed a comprehensive framework for historical data-driven network reconfiguration. The proposed methods learn from historical dataset only, without creating a simulation program or interacting with the real-world system. To improve the sample efficiency, a heuristic data-augmentation technique and a rigorous batch-constrained soft policy iteration theory were developed. Both algorithms can learn a improved control policy superior to the historical one.

- In Chapter 7, we described a novel consensus multi-agent reinforcement learning algorithm for distribution system Volt-VAR control. We utilized the maximum entropy reinforcement learning framework to balance the exploration-exploitation tradeoff, and to decompose the centralized learning problem into a decentralized format. To improve the communication efficiency, we proposed a stochastic approximation (randomization) based scheme to reduce the number of communicated data points. We proved the equivalence of the stochastic approximation representation to the original semi-infinite programming representation.

8.2 Future Research Directions

There are a number of additional research questions that need to be addressed. In the future, we plan to explore the following topics:

- The state estimation algorithm in Chapter 2 is developed for smart meter and SCADA data only. However, AMI data typically experience communication bottleneck and measures low granularity data (e.g., 15-minute or hourly). Therefore the algorithm works for a relatively slow time scale and cannot achieve real time monitoring. To improve the state estimation as a situation awareness tool, we plan to combine the AMI measurements with more granular and near real time SCADA data and micro-PMU data in distribution systems. As a result, we need to address the problem of coordinating data with heterogeneous sampling rate, as well as the observability issue with the fast sampling devices.
- The modified linear model developed for theft detection in Chapter 3 requires certain training data to fit the parameters. However, as the actual anomaly data points are

unknown, the model parameter estimation may not be accurate. In the future, we will develop mixture regression models to avoid a separate dataset for parameter fitting. Instead, by using mixture models, normal and abnormal data can be distinguished and clustered into individual components. Hence, detecting anomaly can be done without assuming a subset of "clean" data is available.

- The historical data-driven deep Q learning and batch-constrained soft actor critic algorithms developed in Chapter 5-Chapter 6 learn the control policy from the operational data passively. An alternative approach to learn a control policy is to combine the off-policy model-free reinforcement learning with model-based planning. We will consider the problem of learning a simulated model of the environment using real world data and planning a policy and a value function through the simulated model. Another research direction is to characterize the relationship between the historical data distribution and the learning agent performance. This is important for both real world application and theoretical understanding of the sample efficiency issue in deep reinforcement learning.

- The consensus multi-agent reinforcement learning algorithm developed in Chapter 7 is found to work well empirically. However, it is still not backed by sufficient theory. In the future, we will develop more rigorous arguments for the consensus multi-agent reinforcement learning algorithm considering the following problem features: consensus multi-agent learning, parameterizations of value function and policy function, stochastic approximation, and bootstrap target neural networks.

Bibliography

[1] Wenyu Wang, Nanpeng Yu, and Raymond Johnson. A model for commercial adoption of photovoltaic systems in california. *Journal of Renewable and Sustainable Energy*, 9(2):025904, 2017.

[2] U.S. Energy Information Administration (EIA), 2020. https://www.eia.gov/. Accessed: July 12, 2020.

[3] Grid Integration Tech Team and Integrated Systems Analysis Tech Team. Summary report on EVs at scale and the U.S. electric power system, 2019.

[4] Nanpeng Yu, Tianshu Wei, and Qi Zhu. From passive demand response to proactive demand participation. In *2015 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1300–1306. IEEE, 2015.

[5] Xiaoyang Zhou, Nanpeng Yu, Weixin Yao, and Raymond Johnson. Forecast load impact from demand response resources. In *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2016.

[6] Tianshu Wei, Qi Zhu, and Nanpeng Yu. Proactive demand participation of smart buildings in smart grid. *IEEE Transactions on Computers*, 65(5):1392–1406, 2015.

[7] Nanpeng Yu, Hongyan Sheng, and Raymond Johnson. Economic valuation of wind curtailment rights. In *2013 IEEE Power & Energy Society General Meeting*, pages 1–5. IEEE, 2013.

[8] Zhenhai Zhang, Jie Shi, Yuanqi Gao, and Nanpeng Yu. Degradation-aware valuation and sizing of behind-the-meter battery energy storage systems for commercial customers. In *2019 IEEE PES GTD Grand International Conference and Exposition Asia (GTD Asia)*, pages 895–900. IEEE, 2019.

[9] Jie Shi, Yuanqi Gao, and Nanpeng Yu. Routing electric vehicle fleet for ride-sharing. In *2018 2nd IEEE Conference on Energy Internet and Energy System Integration (EI2)*, pages 1–6. IEEE, 2018.

[10] Jie Shi, Yuanqi Gao, Wei Wang, Nanpeng Yu, and Petros A Ioannou. Operating electric vehicle fleet for ride-hailing services with reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[11] Wikipedia. “Smart grid”, 2020. Last modified: June 20, 2020. https://en.wikipedia.org/wiki/Smart_grid.

[12] U.S. Department of Energy Office of Electricity Delivery and Energy Reliability. Fault location, isolation, and service restoration technologies reduce outage impact and duration, 2014.

[13] Y. Gao and N. Yu. State estimation for unbalanced electric power distribution systems using ami data. In *2017 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5, 2017.

[14] Yuanqi Gao, Wei Wang, and Nanpeng Yu. Consensus multi-agent reinforcement learning for Volt-VAR control in power distribution networks. *arXiv preprint arXiv:2007.02991*, 2020.

[15] Wenyu Wang, Nanpeng Yu, Brandon Foggo, Joshua Davis, and Juan Li. Phase identification in electric power distribution systems by clustering of smart meter data. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 259–265. IEEE, 2016.

[16] Brandon Foggo and Nanpeng Yu. A comprehensive evaluation of supervised machine learning for the phase identification problem. *World Acad. Sci. Eng. Technol. Int. J. Comput. Syst. Eng*, 12(6), 2018.

[17] Y. Gao, B. Foggo, and N. Yu. A physically inspired data-driven model for electricity theft detection with smart meter data. *IEEE Transactions on Industrial Informatics*, 15(9):5076–5088, Sep. 2019.

[18] Y. Gao, P. Wang, and N. Yu. A distributed algorithm for distribution network reconfiguration. In *2018 China International Conference on Electricity Distribution (CICED)*, pages 1730–1734, 2018.

[19] Y. Gao, J. Shi, W. Wang, and N. Yu. Dynamic distribution network reconfiguration using reinforcement learning. In *IEEE SmartGridComm’19*, Beijing, P.R. China, October 2019.

[20] Y. Gao, W. Wang, J. Shi, and N. Yu. Batch-constrained reinforcement learning for dynamic distribution network reconfiguration. *IEEE Transactions on Smart Grid*, pages 1–1, 2020.

[21] A. Abur and A.G. Expósito. *Power System State Estimation: Theory and Implementation*. Power Engineering (Willis). CRC Press, 2004.

[22] F. C. Schweppe and J. Wildes. Power system static-state estimation, part I: Exact model. *IEEE Transactions on Power Apparatus and Systems*, PAS-89(1):120–125, Jan. 1970.

[23] M. Baran and T. E. McDermott. Distribution system state estimation using AMI data. In *Power Systems Conference and Exposition, 2009. PSCE ’09. IEEE/PES*, pages 1–3, Mar. 2009.

[24] B. Hayes and M. Prodanovic. State estimation techniques for electric power distribution systems. In *Modelling Symposium (EMS), 2014 European*, pages 303–308, Oct. 2014.

[25] K. S. K. Weranga, S. Kumarawadu, and D. P. Chandima. *Smart Metering Design and Applications*. Springer Briefs in Applied Science and Technology. Springer, 2014.

[26] Z. Jia, J. Chen, and Y. Liao. State estimation in distribution system considering effects of AMI data. In *Southeastcon, 2013 Proceedings of IEEE*, pages 1–6, Apr. 2013.

[27] A. Majumdar and B. C. Pal. A three-phase state estimation in unbalanced distribution networks with switch modelling. In *2016 IEEE First International Conference on Control, Measurement and Instrumentation (CMI)*, pages 474–478, Jan. 2016.

[28] C. N. Lu, J. H. Teng, and W. H. E. Liu. Distribution system state estimation. *IEEE Transactions on Power Systems*, 10(1):229–240, Feb. 1995.

[29] W.-M. Lin and J.-H. Teng. State estimation for distribution systems with zero-injection constraints. *IEEE Transactions on Power Systems*, 11(1):518–524, Feb. 1996.

[30] A. Alimardani, F. Therrien, D. Atanackovic, J. Jatskevich, and E. Vaahedi. Distribution system state estimation based on nonsynchronized smart meters. *IEEE Transactions on Smart Grid*, 6(6):2919–2928, Nov. 2015.

[31] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.

[32] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.

[33] A. Gjelsvik, S. Aam, and L. Holten. Hachtel's augmented matrix method - a rapid method improving numerical stability in power system static state estimation. *IEEE Power Engineering Review*, PER-5(11):22–23, Nov. 1985.

[34] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Series in Automatic Computation. Prentice-Hall, Englewood Cliffs, NJ 07632, USA, 1974.

[35] M. Khorramizadeh and N. Mahdavi-Amiri. An efficient algorithm for sparse null space basis problem using ABS methods. *Numerical Algorithms*, 62(3):469–485, 2013.

[36] Martin Holters. Null space for sparse matrix. https://www.mathworks.com/matlabcentral/fileexchange/42922-null-space-for-sparse-matrix.

[37] Alan George and Michael T. Heath. Solution of sparse linear least squares problems using Givens rotations. *Linear Algebra and its Applications*, 34:69 – 83, 1980.

[38] W. H. Kersting. Radial distribution test feeders. In *Power Engineering Society Winter Meeting, 2001. IEEE*, volume 2, pages 908–912, 2001.

[39] Stephen McLaughlin, Dmitry Podkuiko, and Patrick McDaniel. *Energy Theft in the Advanced Metering Infrastructure*, pages 176–187. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[40] Thomas B Smith. Electricity theft: a comparative analysis. *Energy Policy*, 32(18):2067–2076, 2004.

[41] Soma Shekara Sreenadh Reddy Depuru, Lingfeng Wang, and Vijay Devabhaktuni. Support vector machine based data classification for detection of electricity theft. In *Power Systems Conference and Exposition (PSCE), 2011 IEEE/PES*, pages 1–8. IEEE, 2011.

[42] Tanveer Ahmad, Huanxin Chen, Jiangyu Wang, and Yabin Guo. Review of various modeling techniques for the detection of electricity theft in smart grid environment. *Renewable and Sustainable Energy Reviews*, 82:2916 – 2933, 2018.

[43] E. W. S. Angelos, O. R. Saavedra, O. A. C. Cortés, and A. N. de Souza. Detection and identification of abnormalities in customer consumptions in power distribution systems. *IEEE Transactions on Power Delivery*, 26(4):2436–2442, Oct 2011.

[44] C. C. O. Ramos, A. N. de Sousa, J. P. Papa, and A. X. Falcao. A new approach for nontechnical losses detection based on optimum-path forest. *IEEE Transactions on Power Systems*, 26(1):181–189, Feb 2011.

[45] J. Nagi, K. S. Yap, S. K. Tiong, S. K. Ahmed, and M. Mohamad. Nontechnical loss detection for metered customers in power utility using support vector machines. *IEEE Transactions on Power Delivery*, 25(2):1162–1171, April 2010.

[46] Breno Costa, Bruno L. A Alberto, Andre M. Portela, Maduro W, and Esdras O.Eler. Fraud detection in electric power distribution networks using an ann-based knowledge-discovery process. 4:17–23, 11 2013.

[47] Leandro Aparecido Passos Júnior, Caio César Oba Ramos, Douglas Rodrigues, Danillo Roberto Pereira, André Nunes de Souza, Kelton Augusto Pontara da Costa, and João Paulo Papa. Unsupervised non-technical losses identification through optimum-path forest. *Electric Power Systems Research*, 140:413 – 423, 2016.

[48] J. E. Cabral, J. O. P. Pinto, and A. M. A. C. Pinto. Fraud detection system for high and low voltage electricity consumers based on data mining. In *2009 IEEE Power Energy Society General Meeting*, pages 1–5, July 2009.

[49] S. K. Singh, R. Bose, and A. Joshi. Entropy-based electricity theft detection in ami network. *IET Cyber-Physical Systems: Theory Applications*, 3(2):99–105, 2018.

[50] A. H. Nizar, Z. Y. Dong, and Y. Wang. Power utility nontechnical loss analysis with extreme learning machine method. *IEEE Transactions on Power Systems*, 23(3):946–955, Aug 2008.

[51] A. Jindal, A. Dua, K. Kaur, M. Singh, N. Kumar, and S. Mishra. Decision tree and svm-based data analytics for theft detection in smart grid. *IEEE Transactions on Industrial Informatics*, 12(3):1005–1016, June 2016.

[52] Z. Zheng, Y. Yang, X. Niu, H. N. Dai, and Y. Zhou. Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids. *IEEE Transactions on Industrial Informatics*, 14(4):1606–1615, April 2018.

[53] C. J. Bandim, J. E. R. Alves, A. V. Pinto, F. C. Souza, M. R. B. Loureiro, C. A. Magalhaes, and F. Galvez-Durand. Identification of energy theft and tampered meters using a central observer meter: a mathematical approach. In *2003 IEEE PES Transmission and Distribution Conference and Exposition (IEEE Cat. No.03CH37495)*, volume 1, pages 163–168 Vol.1, Sept 2003.

[54] Sook-Chin Yip, KokSheik Wong, Wooi-Ping Hew, Ming-Tao Gan, Raphael C.-W. Phan, and Su-Wei Tan. Detection of energy theft and defective smart meters in smart grids using linear regression. *International Journal of Electrical Power & Energy Systems*, 91:230 – 240, 2017.

[55] P. Jokar, N. Arianpoo, and V. C. M. Leung. Electricity theft detection in AMI using customers consumption patterns. *IEEE Transactions on Smart Grid*, 7(1):216–226, Jan 2016.

[56] M. Tariq and H. V. Poor. Electricity theft detection and localization in grid-tied microgrids. *IEEE Transactions on Smart Grid*, 9(3):1920–1929, May 2018.

[57] R. V. Cruz, C. V. Quintero, and F. Perez. Detecting non-technical losses in radial distribution system transformation point through the real time state estimation method. In *2006 IEEE/PES Transmission Distribution Conference and Exposition: Latin America*, pages 1–5, Aug 2006.

[58] S. C. Huang, Y. L. Lo, and C. N. Lu. Non-technical loss detection using state estimation and analysis of variance. *IEEE Transactions on Power Systems*, 28(3):2959–2966, Aug 2013.

[59] W. Luan, G. Wang, Y. Yu, J. Lin, W. Zhang, and Q. Liu. Energy theft detection via integrated distribution state estimation based on ami and scada measurements. In *2015 5th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, pages 751–756, Nov 2015.

[60] S. Salinas, C. Luo, W. Liao, and P. Li. State estimation for energy theft detection in microgrids. In *9th International Conference on Communications and Networking in China*, pages 96–101, Aug 2014.

[61] Côme Carquex and Catherine Rosenberg. Multi-timescale electricity theft detection and localization in distribution systems based on state estimation and pmu measurements. In *Proceedings of the Ninth International Conference on Future Energy Systems*, e-Energy '18, pages 282–290, New York, NY, USA, 2018. ACM.

[62] D. Drzajic. *Energy Theft Detection using Compressive Sensing Methods.* Semester thesis, ETH Zürich, 2015.

[63] F. Xiao and Q. Ai. Electricity theft detection in smart grid using random matrix theory. *IET Generation, Transmission Distribution*, 12(2):371–378, 2018.

[64] Wenyu Wang and Nanpeng Yu. Maximum marginal likelihood estimation of phase connections in power distribution systems. *IEEE Transactions on Power Systems*, 2020.

[65] Brandon Foggo and Nanpeng Yu. Improving supervised phase identification through the theory of information losses. *IEEE Transactions on Smart Grid*, 11(3):2337–2346, 2019.

[66] Sejun Park, Deepjyoti Deka, and Michael Chertkov. Exact topology and parameter estimation in distribution grids with minimal observability. *CoRR*, abs/1710.10727, 2017.

[67] B. Das. Estimation of parameters of a three-phase distribution feeder. *IEEE Transactions on Power Delivery*, 26(4):2267–2276, Oct 2011.

[68] Wenyu Wang and Nanpeng Yu. Parameter estimation in three-phase power distribution networks using smart meter data. In *16th International Conference on Probabilistic Methods Applied to Power Systems*, pages 1–6, 2020.

[69] Zdenek Zumr. Last Mile Asset Monitoring: Low Cost Rapid Deployment Asset Monitoring. Master's thesis, Portland State University, 2014.

[70] A. Albert and R. Rajagopal. Smart meter driven segmentation: What your consumption says about you. *IEEE Transactions on Power Systems*, 28(4):4019–4030, Nov 2013.

[71] Saverio Bolognani and Florian Dörfler. Fast power system analysis via implicit linearization of the power flow manifold. In *Communication, Control, and Computing (Allerton), 2015 53rd Annual Allerton Conference on*, pages 402–409. IEEE, 2015.

[72] Peter J Rousseeuw and Annick M Leroy. *Robust regression and outlier detection*, volume 589. John wiley & sons, 2005.

[73] Peter J Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79(388):871–880, 1984.

[74] J. Fox and S. Weisberg. *An R Companion to Applied Regression.* SAGE Publications, 2011.

[75] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[76] C.R. Rao, A. Fieger, C. Heumann, H. Toutenburg, T. Nittner, and S. Scheid. *Linear Models: Least Squares and Alternatives.* Springer Series in Statistics. Springer New York, 2006.

[77] Ivan Markovsky and Sabine Van Huffel. Overview of total least-squares methods. *Signal Processing*, 87(10):2283 – 2302, 2007. Special Section: Total Least Squares and Errors-in-Variables Modeling.

[78] S. Rhode, K. Usevich, I. Markovsky, and F. Gauterin. A recursive restricted total least-squares algorithm. *IEEE Transactions on Signal Processing*, 62(21):5652–5662, Nov 2014.

[79] S. McLaughlin, B. Holbert, A. Fawaz, R. Berthier, and S. Zonouz. A multi-sensor energy theft detection framework for advanced metering infrastructures. *IEEE Journal on Selected Areas in Communications*, 31(7):1319–1330, July 2013.

[80] M. Zanetti, E. Jamhour, M. Pellenz, M. Penna, V. Zambenedetti, and I. Chueiri. A tunable fraud detection system for advanced metering infrastructure using short-lived patterns. *IEEE Transactions on Smart Grid*, PP(99):1–1, 2017.

[81] Daisuke Mashima and Alvaro A Cárdenas. Evaluating electricity theft detectors in smart grid networks. In *International Workshop on Recent Advances in Intrusion Detection*, pages 210–229. Springer, 2012.

[82] Christopher M Bishop. *Neural networks for pattern recognition.* Oxford university press, 1995.

[83] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[84] Wenyu Wang, Nanpeng Yu, and Zhouyu Lu. Advanced metering infrastructure data driven phase identification in smart grid. *GREEN 2017 Forward*, page 22, 2017.

[85] Jing Tian, Michael H Azarian, and Michael Pecht. Anomaly detection using self-organizing maps-based k-nearest neighbor algorithm. In *Proceedings of the European Conference of the Prognostics and Health Management Society.* Citeseer, 2014.

[86] F. Capitanescu, I. Bilibin, and E. R. Ramos. A comprehensive centralized approach for voltage constraints management in active distribution grid. *IEEE Transactions on Power Systems*, 29(2):933–942, Mar. 2014.

[87] R. A. Jabr, R. Singh, and B. C. Pal. Minimum loss network reconfiguration using mixed-integer convex programming. *IEEE Transactions on Power Systems*, 27(2):1106–1115, May 2012.

[88] F. Capitanescu, L. F. Ochoa, H. Margossian, and N. D. Hatziargyriou. Assessing the potential of network reconfiguration to improve distributed generation hosting capacity in active distribution systems. *IEEE Transactions on Power Systems*, 30(1):346–356, Jan. 2015.

[89] S. Lei, Y. Hou, F. Qiu, and J. Yan. Identification of critical switches for integrating renewable distributed generation by dynamic network reconfiguration. *IEEE Transactions on Sustainable Energy*, 9(1):420–432, Jan. 2018.

[90] M. R. Dorostkar-Ghamsari, M. Fotuhi-Firuzabad, M. Lehtonen, and A. Safdarian. Value of distribution network reconfiguration in presence of renewable energy resources. *IEEE Transactions on Power Systems*, 31(3):1879–1888, May 2016.

[91] Nanpeng Yu, Sunil Shah, Raymond Johnson, Robert Sherick, Mingguo Hong, and Kenneth Loparo. Big data analytics in power distribution systems. In *Innovative Smart Grid Technologies Conference (ISGT), 2015 IEEE Power & Energy Society*, pages 1–5. IEEE, 2015.

[92] D. Shirmohammadi and H. W. Hong. Reconfiguration of electric distribution networks for resistive line losses reduction. *IEEE Transactions on Power Delivery*, 4(2):1492–1498, Apr. 1989.

[93] S. K. Goswami and S. K. Basu. A new algorithm for the reconfiguration of distribution feeders for loss minimization. *IEEE Transactions on Power Delivery*, 7(3):1484–1491, July 1992.

[94] F. V. Gomes, S. Carneiro, J. L. R. Pereira, M. P. Vinagre, P. A. N. Garcia, E. J. Oliveira, and L. R. Araujo. A new distribution system reconfiguration approach using optimal power flow technique and sensitivity analysis for loss reduction. In *IEEE Power Engineering Society General Meeting, 2005*, pages 897–901 Vol. 1, 2005.

[95] H. P. Schmidt, N. Ida, N. Kagan, and J. C. Guaraldo. Fast reconfiguration of distribution systems considering loss minimization. *IEEE Transactions on Power Systems*, 20(3):1311–1319, Aug. 2005.

[96] Seyhan Civanlar, JJ Grainger, Ho Yin, and SSH Lee. Distribution feeder reconfiguration for loss reduction. *IEEE Transactions on Power Delivery*, 3(3):1217–1223, Jul. 1988.

[97] M. E. Baran and F. F. Wu. Network reconfiguration in distribution systems for loss reduction and load balancing. *IEEE Transactions on Power Delivery*, 4(2):1401–1407, Apr. 1989.

[98] H. . Chiang and R. Jean-Jumeau. Optimal network reconfigurations in distribution systems. i. a new formulation and a solution methodology. *IEEE Transactions on Power Delivery*, 5(4):1902–1909, Oct. 1990.

[99] E. M. Carreno, R. Romero, and A. Padilha-Feltrin. An efficient codification to solve distribution network reconfiguration for loss reduction problem. *IEEE Transactions on Power Systems*, 23(4):1542–1551, 2008.

[100] Ching-Tzong Su, Chung-Fu Chang, and Ji-Pyng Chiou. Distribution network reconfiguration for loss reduction by ant colony search algorithm. *Electric Power Systems Research*, 75(2-3):190–199, 2005.

[101] J. A. Taylor and F. S. Hover. Convex models of distribution system reconfiguration. *IEEE Transactions on Power Systems*, 27(3):1407–1413, Aug. 2012.

[102] H. Ahmadi and J. R. Martí. Distribution system optimization based on a linear power-flow formulation. *IEEE Transactions on Power Delivery*, 30(1):25–33, Feb. 2015.

[103] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei. A survey of distributed optimization and control algorithms for electric power systems. *IEEE Transactions on Smart Grid*, 8(6):2941–2962, Nov. 2017.

[104] H. . Chiang and M. E. Baran. On the existence and uniqueness of load flow solution for radial distribution power networks. *IEEE Transactions on Circuits and Systems*, 37(3):410–416, Mar. 1990.

[105] Mohammadjavad Feizollahi. *Large-scale unit commitment: Decentralized mixed integer programming approaches.* PhD thesis, Georgia Institute of Technology, 2015.

[106] Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers.* Now Publishers Inc, 2011.

[107] Dimitri P. Bertsekas. *Convex optimization algorithms.* Athena Scientific, 2015.

[108] P. Wang, Y. Gao, N. Yu, W. Ren, J. Lian, and D. Wu. Communication-efficient distributed solutions to a system of linear equations with laplacian sparse structure. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 3367–3372, 2018.

[109] C. T. Su and C. S. Lee. Network reconfiguration of distribution systems using improved mixed-integer hybrid differential evolution. *IEEE Transactions on Power Delivery*, 18(3):1022–1027, Jul. 2003.

[110] Branimir Novoselnik and Mato Baotić. Dynamic reconfiguration of electrical power distribution systems with distributed generation and storage. *IFAC-PapersOnLine*, 48(23):136–141, 2015.

[111] H. M. A. Ahmed and M. M. A. Salama. Energy management of AC–DC hybrid distribution systems considering network reconfiguration. *IEEE Transactions on Power Systems*, 34(6):4583–4594, Nov. 2019.

[112] E. Kianmehr, S. Nikkhah, V. Vahidinasab, D. Giaouris, and P. C. Taylor. A resilience-based architecture for joint distributed energy resources allocation and hourly network reconfiguration. *IEEE Transactions on Industrial Informatics*, 15(10):5444–5455, Oct. 2019.

[113] C. Lee, C. Liu, S. Mehrotra, and Z. Bie. Robust distribution network reconfiguration. *IEEE Transactions on Smart Grid*, 6(2):836–842, Mar. 2015.

[114] H. Haghighat and B. Zeng. Distribution system reconfiguration under uncertain load and renewable generation. *IEEE Transactions on Power Systems*, 31(4):2666–2675, Jul. 2016.

[115] A. Akrami, M. Doostizadeh, and F. Aminifar. Optimal reconfiguration of distribution network using μPMU measurements: A data-driven stochastic robust optimization. *IEEE Transactions on Smart Grid*, 11(1):420–428, Jan. 2020.

[116] F. V. Dantas, D. Z. Fitiwi, S. F. Santos, and J. P. S. Catalão. Dynamic reconfiguration of distribution network systems: A key flexibility option for RES integration. In *EEEIC/ICPS Europe*, pages 1–6, Jun. 2017.

[117] A. Kavousi-Fard, A. Zare, and A. Khodaei. Effective dynamic scheduling of reconfigurable microgrids. *IEEE Transactions on Power Systems*, 33(5):5519–5530, Sep. 2018.

[118] Mustafa Mosbah, Salem Arif, Ridha Djamel Mohammedi, and Abdelhafid Hellal. Optimum dynamic distribution network reconfiguration using minimum spanning tree algorithm. In *2017 5th International Conference on Electrical Engineering-Boumerdes (ICEE-B)*, pages 1–6, 2017.

[119] D. P. Bernardon, A. P. C. Mello, L. L. Pfitscher, L. N. Canha, A. R. Abaide, and A. A. B. Ferreira. Real-time reconfiguration of distribution network with distributed generation. *Electric Power Systems Research*, 107:59–67, 2014.

[120] Z. Liu, Y. Liu, G. Qu, X. Wang, and X. Wang. Intra-day dynamic network reconfiguration based on probability analysis considering the deployment of remote control switches. *IEEE Access*, 7:145272–145281, 2019.

[121] S. Chen, W. Hu, and Z. Chen. Comprehensive cost minimization in distribution networks using segmented-time feeder reconfiguration and reactive power control of distributed generators. *IEEE Transactions on Power Systems*, 2016.

[122] Y. Fu and H. Chiang. Toward optimal multiperiod network reconfiguration for increasing the hosting capacity of distribution networks. *IEEE Transactions on Power Delivery*, 33(5):2294–2304, Oct. 2018.

[123] Jianzhong Wu and Yixin Yu. Global algorithm to time-varying reconfiguration for operation cost minimization. *Proceedings of the CSEE*, 2003. (in Chinese).

[124] Eugene A Feinberg, Jiaqiao Hu, and Kan Huang. A rolling horizon approach to distribution feeder reconfiguration with switching costs. In *IEEE SmartGridComm'11*, Brussels, Belgium, Oct. 2011.

[125] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[126] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 22–31. JMLR. org, 2017.

[127] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[128] Arun Nair, Praveen Srinivasan, Sam Blackwell, Cagdas Alcicek, Rory Fearon, Alessandro De Maria, Vedavyas Panneershelvam, Mustafa Suleyman, Charles Beattie, Stig Petersen, Shane Legg, Volodymyr Mnih, Koray Kavukcuoglu, and David Silver. Massively parallel methods for deep reinforcement learning. In *Intl. Conf. on Machine Learning*, Lille, France, 2015.

[129] Donald E Knuth. *The art of computer programming, volume 4A: combinatorial algorithms, part 1.* Pearson Education India, 2011.

[130] James W Taylor. Short-term electricity demand forecasting using double seasonal exponential smoothing. *Journal of the Operational Research Society*, 54(8):799–805, 2003.

[131] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[132] Johan Wågberg, Dave Zachariah, Thomas B. Schön, and Petre Stoica. Prediction performance after learning in gaussian process regression. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, FL, USA, 4 2017.

[133] Michael D Simms. The National Commission for Energy State Regulation of Ukraine Energy, The Public Utilities Commission of Ohio-Distribution System Losses, 2013. Duke Energy®.

[134] A. Abiri-Jahromi, M. Fotuhi-Firuzabad, M. Parvania, and M. Mosleh. Optimized sectionalizing switch placement strategy in distribution systems. *IEEE Transactions on Power Delivery*, 27(1):362–370, Jan 2012.

[135] The Reliability and Power Quality Performance of Overhead Lines – with reference to the Electrical Properties of Wood and Covered Conductors. https://espace.library.uq.edu.au/view/UQ:9818/saha-mat-dist200.pdf. Accessed: 2018-09-10.

[136] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning (ICML)*, 2016.

[137] Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

[138] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[139] Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2775–2785, 2017.

[140] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 2052–2062. PMLR, 09–15 Jun. 2019.

[141] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

[142] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pages 3483–3491, 2015.

[143] D. Das. A fuzzy multiobjective approach for network reconfiguration of distribution systems. *IEEE Transactions on Power Delivery*, 21(1):202–209, Jan. 2006.

[144] Dong Zhang, Zhengcai Fu, and Liuchun Zhang. An improved TS algorithm for loss-minimum reconfiguration in large-scale distribution systems. *Electric Power Systems Research*, 77(5-6):685–694, 2007.

[145] W. Wang, N. Yu, Y. Gao, and J. Shi. Safe off-policy deep reinforcement learning algorithm for volt-var control in power distribution systems. *IEEE Transactions on Smart Grid*, 11(4):3008–3018, July 2020.

[146] Commission for Energy Regulation (CER). CER smart metering project - electricity customer behaviour trial, 2009-2010 [dataset]. 1st edition. Irish social science data archive. SN: 0012-00. www.ucd.ie/issda/CER-electricity.

[147] Bri Mathias Hodge. Solar power data for integration studies. https://www.nrel.gov/grid/solar-power-data.html. Accessed: July 11, 2019.

[148] J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[149] H. Ahmadi, J. R. Martí, and H. W. Dommel. A framework for Volt-VAR optimization in distribution systems. *IEEE Transactions on Smart Grid*, 6(3):1473–1483, May 2015.

[150] Ruey-Hsun Liang and Chen-Kuo Cheng. Dispatch of main transformer ULTC and capacitors in a distribution system. *IEEE Transactions on Power Delivery*, 16(4):625–630, Oct. 2001.

[151] Alberto Borghetti. Using mixed integer programming for the Volt/VAR optimization in distribution feeders. *Electric Power Systems Research*, 98:39–50, 2013.

[152] Rahul Ranjan Jha, Anamika Dubey, Chen-Ching Liu, and Kevin P Schneider. Bi-level Volt-VAR optimization to coordinate smart inverters with voltage control devices. *IEEE Transactions on Power Systems*, 34(3):1801–1813, May 2019.

[153] Novalio Daratha, Biswarup Das, and Jaydev Sharma. Robust voltage regulation in unbalanced radial distribution system under uncertainty of distributed generation and loads. *International Journal of Electrical Power & Energy Systems*, 73:516–527, 2015.

[154] Weiye Zheng, Wenchuan Wu, Boming Zhang, and Yongjie Wang. Robust reactive power optimization and voltage control method for active distribution networks via dual time-scale coordination. *ArXiv*, abs/1608.04879, 2017.

[155] F. U. Nazir, B. C. Pal, and R. A. Jabr. A two-stage chance constrained Volt/VAR control scheme for active distribution networks with nodal power uncertainties. *IEEE Transactions on Power Systems*, 34(1):314–325, Jan. 2019.

[156] T. Senjyu, Y. Miyazato, A. Yona, N. Urasaki, and T. Funabashi. Optimal distribution voltage control and coordination with distributed generation. *IEEE Transactions on Power Delivery*, 23(2):1236–1242, Apr. 2008.

[157] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi. A particle swarm optimization for reactive power and voltage control considering voltage security assessment. *IEEE Transactions on Power Systems*, 15(4):1232–1239, Nov. 2000.

[158] Alfredo Vaccaro and Ahmed F Zobaa. Voltage regulation in active networks by distributed and cooperative meta-heuristic optimizers. *Electric power systems research*, 99:9–17, 2013.

[159] H. E. Z. Farag and E. F. El-Saadany. A novel cooperative protocol for distributed voltage control in active distribution systems. *IEEE Transactions on Power Systems*, 28(2):1645–1656, May 2013.

[160] B. A. Robbins, H. Zhu, and A. D. Domínguez-García. Optimal tap setting of voltage regulation transformers in unbalanced distribution systems. *IEEE Transactions on Power Systems*, 31(1):256–267, Jan. 2016.

[161] P. Bagheri and W. Xu. Model-free Volt-VAR control based on measurement data analytics. *IEEE Transactions on Power Systems*, 34(2):1471–1482, Mar. 2019.

[162] E. Pourjafari and M. Reformat. A support vector regression based model predictive control for Volt-VAR optimization of distribution systems. *IEEE Access*, 7:93352–93363, 2019.

[163] H. Xu, A. Dominguez-Garcia, and P. W. Sauer. Optimal tap setting of voltage regulation transformers using batch reinforcement learning. *IEEE Transactions on Power Systems*, pages 1–1, May 2019.

[164] Y. Xu, W. Zhang, W. Liu, and F. Ferrese. Multiagent-based reinforcement learning for optimal reactive power dispatch. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1742–1751, Nov. 2012.

[165] Ying Zhang, Xinan Wang, Jianhui Wang, and Yingchen Zhang. Deep reinforcement learning based Volt-VAR optimization in smart distribution systems. *arXiv preprint arXiv:2003.03681*, 2020.

[166] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 5872–5881. PMLR, 10–15 July 2018.

[167] J. Chen and A. H. Sayed. Diffusion adaptation strategies for distributed optimization and learning over networks. *IEEE Transactions on Signal Processing*, 60(8):4289–4305, Aug. 2012.

[168] L. Xie, D. Choi, S. Kar, and H. V. Poor. Fully distributed state estimation for wide-area monitoring systems. *IEEE Transactions on Smart Grid*, 3(3):1154–1169, Sept. 2012.

[169] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin. On the linear convergence of the ADMM in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, Apr. 2014.

[170] Vladislav B Tadić, Sean P Meyn, and Roberto Tempo. Randomized algorithms for semi-infinite programming problems. In *Probabilistic and Randomized Methods for Design under Uncertainty*, pages 243–261. Springer, 2006.

[171] Ali H Sayed. Adaptation, learning, and optimization over networks. *Foundations and Trends in Machine Learning*, 7:311–801, 2014.

[172] Yunhao Tang and Shipra Agrawal. Discretizing continuous action space for on-policy optimization. *arXiv preprint arXiv:1901.10500*, 2019.

[173] UK Power Networks. Smart meter energy consumption data in London households. https://data.london.gov.uk/dataset/smartmeter-energy-use-data-in-london-households.

[174] Q. Ling, W. Shi, G. Wu, and A. Ribeiro. DLM: decentralized linearized alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 63(15):4051–4064, Aug. 2015.

[175] AH Nizar, ZY Dong, M Jalaluddin, and MJ Raffles. Load profiling method in detecting non-technical loss activities in a power utility, 2006.

[176] Deepjyoti Deka, Scott Backhaus, and Michael Chertkov. Structure learning and statistical estimation in distribution networks-part I. *arXiv preprint arXiv:1501.04131*, 2015.

[177] F. Dorfler and F. Bullo. Kron reduction of graphs with applications to electrical networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(1):150–163, Jan 2013.

[178] A. J. Berrisford. A tale of two transformers: An algorithm for estimating distribution secondary electric parameters using smart meter data. In *2013 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–6, May 2013.

[179] G. Casella and R.L. Berger. *Statistical Inference.* Duxbury advanced series. Brooks/Cole Publishing Company, 1990.

[180] Donald E. Knuth. *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[181] Anirban DasGupta. *Probability for Statistics and Machine Learning: Fundamentals and Advanced Topics.* Springer Publishing Company, Incorporated, 1st edition, 2011.

[182] Isabelle Albert and Jean-Baptiste Denis. Dirichlet and multinomial distributions: properties and uses in jags. *Analysis*, 31:1141–1155, 2011.

[183] Z. Ma, J. H. Xue, A. Leijon, Z. H. Tan, Z. Yang, and J. Guo. Decorrelation of neutral vector variables: Theory and applications. *IEEE Transactions on Neural Networks and Learning Systems*, PP(99):1–15, 2017.

[184] L. Devroye. *Non-Uniform Random Variate Generation.* SpringerLink : Bücher. Springer New York, 2013.

[185] William H Kersting. *Distribution system modeling and analysis, Third edition.* CRC press, 2012.

[186] J.D. Gibbons and S. Chakraborti. *Nonparametric Statistical Inference, Fourth Edition: Revised and Expanded.* Taylor & Francis, 2014.

[187] Deepthi Cheboli. *Anomaly detection of time series.* PhD thesis, University of Minnesota, 2010.

[188] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[189] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques.* Elsevier, 2011.

[190] Ali Ghodsi. Dimensionality reduction a short tutorial. *Department of Statistics and Actuarial Science, Univ. of Waterloo, Ontario, Canada*, 37:38, 2006.

[191] T.F. Cox and M.A.A. Cox. *Multidimensional Scaling, Second Edition.* Chapman & Hall/CRC Monographs on Statistics & Applied Probability. CRC Press, 2000.

[192] Understanding support vector machine regression. https://www.mathworks.com/help/stats/understanding-support-vector-machine-regression.html. Accessed: 2017-09-28.

[193] S. Bolognani and S. Zampieri. A distributed control strategy for reactive power compensation in smart microgrids. *IEEE Transactions on Automatic Control*, 58(11):2818–2833, Nov 2013.

[194] A.R. Bergen and V. Vittal. *Power Systems Analysis.* Prentice Hall, 2000.

[195] M. Bockarjova and G. Andersson. Transmission line conductor temperature impact on state estimation accuracy. In *2007 IEEE Lausanne Power Tech*, pages 701–706, July 2007.

[196] Jerzy K Baksalary and Oskar Maria Baksalary. Particular formulae for the moore–penrose inverse of a columnwise partitioned matrix. *Linear algebra and its applications*, 421(1):16–23, 2007.

[197] Steve Pischke. Lecture notes on measurement error. *Lecture Notes on Measurement Error*, 2007.

[198] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Appendix A

Appendices for Chapter 3

A.1 Linearization of Distribution Secondary Power Flow Equations

We wish to approximate the nonlinear power flow equation as a linear one:

$$\mathcal{F}(\mathbf{v}, \boldsymbol{\theta}, \mathbf{p}, \mathbf{q}) = \mathbf{0} \quad \rightarrow \quad \mathbf{F}_{\bar{\mathbb{X}}}[\mathbf{v}^{\mathsf{T}}, \boldsymbol{\theta}^{\mathsf{T}}, \mathbf{p}^{\mathsf{T}}, \mathbf{q}^{\mathsf{T}}]^{\mathsf{T}} = \mathbf{0}$$

where $\mathbf{v} = [\mathbf{v}^{1\mathsf{T}}, \mathbf{v}^{2\mathsf{T}}]^{\mathsf{T}}$ (same token for $\boldsymbol{\theta}, \mathbf{p}, \mathbf{q}$); $\mathbf{F}_{\bar{\mathbb{X}}}$ is the Jacobian matrix of $\mathcal{F}$ evaluated at some operating point $\bar{\mathbb{X}} = \begin{bmatrix} \bar{\mathbf{v}} & \bar{\boldsymbol{\theta}} & \bar{\mathbf{p}} & \bar{\mathbf{q}} \end{bmatrix}^T$. This point must itself be a solution to the power flow equation $\mathcal{F}(\bar{\mathbb{X}}) = \mathbf{0}$. When $\bar{\mathbb{X}}$ is fixed, this Jacobean is given by [71]

$$\mathbf{F}_{\bar{\mathbb{X}}} = \begin{bmatrix} \left(\langle \mathrm{diag}(\mathbf{Y}\mathbf{u})^* \rangle + \langle \mathrm{diag}(\mathbf{u}) \rangle \, \mathbf{N}_{2n} \, \langle \mathbf{Y} \rangle\right) R(\mathbf{u}) & -\mathbf{I} \end{bmatrix} \tag{A.1}$$

where $\mathbf{Y}$ is the bus admittance matrix, $\mathbf{u}$ is the vector of complex bus voltages, $\mathbf{N}_{2n} = \begin{bmatrix} \mathbf{I}_{2n} & \mathbf{0} \\ \mathbf{0} & -\mathbf{I}_{2n} \end{bmatrix}$, and

$$R(\mathbf{u}) = \begin{bmatrix} \text{diag}(\cos(\boldsymbol{\theta})) & -\text{diag}(\mathbf{v}\sin(\boldsymbol{\theta})) \\ \text{diag}(\sin(\boldsymbol{\theta})) & \text{diag}(\mathbf{v}\cos(\boldsymbol{\theta})) \end{bmatrix}$$

$$\langle \mathbf{A} \rangle = \begin{bmatrix} Re\{\mathbf{A}\} & -Im\{\mathbf{A}\} \\ Im\{\mathbf{A}\} & Re\{\mathbf{A}\} \end{bmatrix}$$

Recall that our modified flat voltage solution is given by $\bar{\mathbf{u}} = [\mathbf{1}_n, -\mathbf{1}_n]^\mathsf{T}$, $\bar{\mathbf{p}} + j\bar{\mathbf{q}} = \mathbf{0}$. Assuming that no shunt resistances are present, this is a solution to the power flow manifold with zero branch currents. Thus $\text{diag}(\mathbf{Y}\mathbf{u}) = \mathbf{0}$. Furthermore, $\text{diag}(\mathbf{u}) = \begin{bmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & -\mathbf{I}_n \end{bmatrix} \triangleq \mathbf{N}_n$, so $\langle \text{diag}(\mathbf{u}) \rangle = R(\mathbf{u}) = \begin{bmatrix} \mathbf{N}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_n \end{bmatrix}$. Thus the left hand block matrix of (A.1) reduces to

$$\begin{aligned} &\begin{bmatrix} \mathbf{N}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_n \end{bmatrix} \begin{bmatrix} \mathbf{G}^\mathbf{r} & -\mathbf{B}^\mathbf{r} \\ -\mathbf{B}^\mathbf{r} & -\mathbf{G}^\mathbf{r} \end{bmatrix} \begin{bmatrix} \mathbf{N}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{N}_n \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{N}_n\mathbf{G}^\mathbf{r}\mathbf{N}_n & -\mathbf{N}_n\mathbf{B}^\mathbf{r}\mathbf{N}_n \\ -\mathbf{N}_n\mathbf{B}^\mathbf{r}\mathbf{N}_n & -\mathbf{N}_n\mathbf{G}^\mathbf{r}\mathbf{N}_n \end{bmatrix} \end{aligned}$$

where $\mathbf{G^r}$ and $\mathbf{B^r}$ are the real an imaginary components of $\mathbf{Y^r}$. In this final expression, each product $\mathbf{N}_n\mathbf{A}\mathbf{N}_n$ negates the off diagonal blocks of $\mathbf{A}$, yielding the desired linearization

$$\begin{bmatrix}\mathbf{p}^1\\ \mathbf{p}^2\\ \mathbf{q}^1\\ \mathbf{q}^2\end{bmatrix} = \begin{bmatrix}\mathbf{G}^{11} & -\mathbf{G}^{12} & -\mathbf{B}^{11} & \mathbf{B}^{12}\\ -\mathbf{G}^{21} & \mathbf{G}^{22} & \mathbf{B}^{21} & -\mathbf{B}^{22}\\ -\mathbf{B}^{11} & \mathbf{B}^{12} & -\mathbf{G}^{11} & \mathbf{G}^{12}\\ \mathbf{B}^{21} & -\mathbf{B}^{22} & \mathbf{G}^{21} & -\mathbf{G}^{22}\end{bmatrix}\begin{bmatrix}\mathbf{v}^1\\ \mathbf{v}^2\\ \boldsymbol{\theta}^1\\ \boldsymbol{\theta}^2\end{bmatrix} \tag{A.2}$$

A.2 Conversion from Loads to Net Injections

(3.3) is derived as the follows. First define the reference direction of voltages and currents as shown in Figure A.1. u variables refer to voltages, i variables refer to currents, and s variables refer to VA power consumptions. We then have:

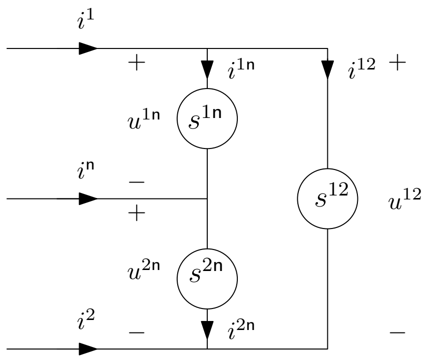


Figure A.1: A triplex line load circuit with reference directions

$$\begin{aligned} s^1 &= u^{1n} i^{1*} = u^{1n}(i^{1n*} + i^{12*}) = s^{1n} + u^{1n}\frac{s^{12}}{u^{1n}+u^{2n}} \\ s^2 &= u^{2n} i^{2*} = u^{2n}(-i^{2n*} - i^{12*}) = s^{2n} - u^{2n}\frac{s^{12}}{u^{1n}+u^{2n}} \end{aligned} \tag{A.3}$$

A.3 The Elimination of Dependencies on Voltage Angles and Reactive Powers

We first show that the pseudoinverse of $\mathbf{L}^{\mathbf{s}}_{22}$ can eliminates the voltage angle dependence in our model. We first rearrange the model equations to

$$\mathbf{p}^{\mathbf{s}} = \mathbf{L}^{\mathbf{s}}_{11}\mathbf{v}^{\mathbf{s}} + \mathbf{L}^{\mathbf{s}}_{12}\boldsymbol{\theta}^{\mathbf{s}} \tag{A.4}$$

$$\mathbf{L}^{\mathbf{s}}_{22}\boldsymbol{\theta}^{\mathbf{s}} = \mathbf{q}^{\mathbf{s}} - \mathbf{L}^{\mathbf{s}}_{21}\mathbf{v}^{\mathbf{s}} \tag{A.5}$$

Since (A.5) is enforced by our model, its right hand side is in $Range(\mathbf{L}^{\mathbf{s}}_{22})$. Let $\boldsymbol{\theta}^{*}$ denote its least norm solution. Then any other solution can be written as $\boldsymbol{\theta}^{s} = \boldsymbol{\theta}^{*} + \boldsymbol{\theta}_n$ where $\boldsymbol{\theta}_n$ is in $Null(\mathbf{L}^{\mathbf{s}}_{22})$.

Now,

$$\mathbf{L}^{\mathbf{s}}_{22} = \begin{bmatrix} -\mathbf{G}^{11} & \mathbf{G}^{12} \\ \mathbf{G}^{21} & -\mathbf{G}^{22} \end{bmatrix} \tag{A.6}$$

and each of the blocks $\mathbf{G}^{ij}$ is a Laplacian matrix having nullspace $\mathbf{1}$. In practice the mutual conductances are much smaller than the self conductances [185]. Hence the entries in $\mathbf{G}^{12}$ and $\mathbf{G}^{21}$ are much smaller than that in $\mathbf{G}^{11}$ and $\mathbf{G}^{22}$. Then the overall matrix $\mathbf{L}^{\mathbf{s}}_{22}$ has nullspace

$$Null(\mathbf{L}^{\mathbf{s}}_{22}) = Span\left(\begin{bmatrix} \mathbf{1}_n \\ \mathbf{0}_n \end{bmatrix}, \begin{bmatrix} \mathbf{0}_n \\ \mathbf{1}_n \end{bmatrix}\right)$$

$$\mathbf{L}^{\mathbf{s}}_{12} = \begin{bmatrix} -\mathbf{B}^{11} + \mathbf{B}^{21} & \mathbf{B}^{12} - \mathbf{B}^{22} \\ -\mathbf{B}^{11} - \mathbf{B}^{21} & \mathbf{B}^{12} + \mathbf{B}^{22} \end{bmatrix} \tag{A.7}$$

On the other hand, in (A.7), each block $\mathbf{B}^{ij}$ is again a Laplacian matrix. Thus the nullspace of this matrix contains the above spanning vectors, so $Null(\mathbf{L}^{\mathbf{s}}_{22}) \subseteq Null(\mathbf{L}^{\mathbf{s}}_{12})$. It follows that $\mathbf{L}^{\mathbf{s}}_{12}\boldsymbol{\theta}^{\mathbf{s}} = \mathbf{L}^{\mathbf{s}}_{12}\left(\boldsymbol{\theta}^{*} + \boldsymbol{\theta_n}\right) = \mathbf{L}^{\mathbf{s}}_{12}\boldsymbol{\theta}^{*}$. We can then write the above system as

$$\mathbf{p}^{\mathbf{s}} = \mathbf{L}^{\mathbf{s}}_{11}\mathbf{v}^{\mathbf{s}} + \mathbf{L}^{\mathbf{s}}_{12}\boldsymbol{\theta}^{*} \tag{A.8}$$

$$\mathbf{L}^{\mathbf{s}}_{22}\boldsymbol{\theta}^{*} = \mathbf{q}^{\mathbf{s}} - \mathbf{L}^{\mathbf{s}}_{21}\mathbf{v}^{\mathbf{s}} \tag{A.9}$$

where $\boldsymbol{\theta}^{*} = \mathbf{L}^{\mathbf{s}\dagger}_{22}\left(\mathbf{q}^{\mathbf{s}} - \mathbf{L}^{\mathbf{s}}_{21}\mathbf{v}^{\mathbf{s}}\right)$ because it is the least norm solution. Substituting this into (A.8) yields the desired result

$$\mathbf{p}^{\mathbf{s}} = \left(\mathbf{L}^{\mathbf{s}}_{11} - \mathbf{L}^{\mathbf{s}}_{12}\mathbf{L}^{\mathbf{s}\dagger}_{\mathbf{22}}\mathbf{L}^{\mathbf{s}}_{21}\right)\mathbf{v}^{\mathbf{s}} + \mathbf{L}^{\mathbf{s}}_{12}\mathbf{L}^{\mathbf{s}\dagger}_{\mathbf{22}}\mathbf{q}^{\mathbf{s}} \tag{A.10}$$

We conclude this appendix by showing that $(\mathbf{I} - \mathbf{L}^{\mathbf{s}}_{12}\mathbf{L}^{\mathbf{s}\dagger}_{22}\mathbf{D}\mathbf{M}_u^{-1})$ is nonsingular. We do this by showing that 1 is *not* an eigenvalue of $\mathbf{L}^{\mathbf{s}}_{12}\mathbf{L}^{\mathbf{s}\dagger}_{22}\mathbf{D}\mathbf{M}_u^{-1}$.

First, it is easy to show that $\mathbf{L}^{\mathbf{r}}_{22} = \mathbf{L}^{\mathbf{s}}_{22}$, and $\mathbf{L}^{\mathbf{s}}_{12} = \mathbf{M}_u\mathbf{L}^{\mathbf{r}}_{12}$, so

$$\mathbf{L}^{\mathbf{s}}_{12}\mathbf{L}^{\mathbf{s}\dagger}_{22}\mathbf{D}\mathbf{M}_u^{-1} = \mathbf{M}_u\mathbf{L}^{\mathbf{r}}_{12}\mathbf{L}^{\mathbf{r}\dagger}_{22}\mathbf{D}\mathbf{M}_u^{-1} \tag{A.11}$$

Thus, if 1 is an eigenvalue of $\mathbf{L}^{\mathbf{s}}_{12}\mathbf{L}^{\mathbf{s}\dagger}_{22}\mathbf{D}\mathbf{M}_u^{-1}$, then 1 is an eigenvalue of $\mathbf{L}^{\mathbf{r}}_{12}\mathbf{L}^{\mathbf{r}\dagger}_{22}\mathbf{D}$. Then there exists a vector $\mathbf{p}$ such that $\mathbf{p} = \mathbf{L^r_{12}}\mathbf{L^{r\dagger}_{22}}\mathbf{D}\mathbf{p}$. Then a vector $\boldsymbol{\theta_x} \in Range(\mathbf{L^{r\dagger}_{22}})$ exists:

$$\mathbf{L^r_{22}}\boldsymbol{\theta_x} = \mathbf{QDp} \tag{A.12}$$

$$\mathbf{L^r_{12}}\boldsymbol{\theta_x} = \mathbf{p} \tag{A.13}$$

where $\mathbf{Q}$ is the orthogonal projector onto the range of $\mathbf{L^r_{22}}$. Then, since $\boldsymbol{\theta_x} \perp Null(\mathbf{L^r_{22}})$ we have

$$\left(\mathbf{L^r_{22}} - (\mathbf{I} - \mathbf{N})\mathbf{D}\mathbf{L^r_{12}}\right)\boldsymbol{\theta_x} = 0 \tag{A.14}$$

$$\mathbf{N}\boldsymbol{\theta_x} = \mathbf{0} \tag{A.15}$$

Where $\mathbf{N}$ is the orthogonal projector onto the nullspace of $\mathbf{L^r_{22}}$ and is given by

$$\mathbf{N} = \frac{1}{n}\begin{bmatrix} \mathbf{1}\mathbf{1}^\mathsf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{1}\mathbf{1}^\mathsf{T} \end{bmatrix} \tag{A.16}$$

Thus, for a solution to exist, the following augmented matrix cannot have full column rank ($= 2n_c$):

$$\begin{bmatrix} \mathbf{L^r_{22}} - (\mathbf{I} - \mathbf{N})\mathbf{D}\mathbf{L^r_{12}} \\ \mathbf{N} \end{bmatrix} \tag{A.17}$$

But clearly $(\mathbf{I} - \mathbf{N})\mathbf{D}\mathbf{L^r_{12}}$ has the same range and nullspace as $\mathbf{L^r_{12}}$. The sum of the first n_c rows is therefore zero. The same holds for the last n_c rows. Thus we can perform row operations to show that this has the same rank as the matrix with the n_c^{th} and $2n_c^{th}$ rows removed. The upper matrix also has the property that the sum of the first n_c *columns* is zero and the sum of the last n_c *columns* is zero. Thus column operations show that our matrix has the same rank as

$$\begin{bmatrix} [\mathbf{L}^{\mathbf{r}}_{22} - (\mathbf{I}-\mathbf{N})\mathbf{D}\mathbf{L}^{\mathbf{r}}_{12}]_{red} & & \mathbf{0} & \mathbf{0} \\ \mathbf{1}\mathbf{1}^{\mathsf{T}} & 0 & 1 & 0 \\ 0 & \mathbf{1}\mathbf{1}^{\mathsf{T}} & 0 & 1 \end{bmatrix} \tag{A.18}$$

where the *red* subscript indicates that the n_c^{th} row, n_c^{th} column, $2n_c^{th}$ row, and $2n_c^{th}$ column have been removed.

Now, the lower right hand block of this matrix indicates two pivots, so a solution can only exist if $[\mathbf{L}^{\mathbf{r}}_{22} - (\mathbf{I}-\mathbf{N})\mathbf{D}\mathbf{L}^{\mathbf{r}}_{12}]_{red}$ does not have full rank ($= 2(n_c - 1)$). Since this matrix only removes rows and columns from its constituents, we can write it as $[\mathbf{L}^{\mathbf{r}}_{22}]_{red} - [(\mathbf{I}-\mathbf{N})\mathbf{D}\mathbf{L}^{\mathbf{r}}_{12}]_{red}$ where the constituent matrices now have full rank.

This matrix subtraction is unlikely to have less than full rank for two reasons. First, it is an extremely precise requirement on the relationship between the network parameters and the power factors. It is precise in the sense that the set of all invertible matrices sum to a singular matrix has Lebesgue measure zero. Second, the matrix $[\mathbf{L}^{\mathbf{r}}_{22}]_{red}$ contains conductance values and the matrix $[(\mathbf{I}-\mathbf{N})\mathbf{D}\mathbf{L}^{\mathbf{r}}_{12}]_{red}$ contains transformed susceptance values. Since susceptance values are typically much larger than conductance values, it follows that the rows of the matrix subtraction will be primarily dominated by $[(\mathbf{I}-\mathbf{N})\mathbf{D}\mathbf{L}^{\mathbf{r}}_{12}]_{red}$ which has full rank. Thus a real network is unlikely to have this difference be singular or even close to singular. Therefore in realworld cases, $(\mathbf{I} - \mathbf{L}^{\mathbf{s}}_{12}\mathbf{L}^{\mathbf{s}\dagger}_{22}\mathbf{D}\mathbf{M}_u^{-1})$ will be invertible.

A.4 Proof of Lemma 3.4.1, Lemma 3.4.2, and Lemma 3.4.3

Proof of Lemma 3.4.1. Suppose without loss of generality that customer i is the electricity thief. Suppose that our training window lasts T time instances. Then at any time t:

$$\begin{aligned}
&\sum_j \left(\tilde{y}(t)^e_j - \tilde{y}(t)_j\right) \\
&= \sum_j \Big((y(t)^e_j - y(t)_j) - (\mathcal{X}(t)^e - \mathcal{X}(t))\beta^y_j\Big) \\
&= (y(t)^e_i - y(t)_i) - \sum_j \beta^y_j \sum_k (y(t)^e_k - y(t)_k) \\
&= (y(t)^e_i - y(t)_i)(1 - \sum_j \beta^y_j)
\end{aligned} \tag{A.19}$$

because $(y(t)^e_j - y(t)_j)$ is nonzero at index i only and $\mathcal{X}^e$ and $\mathcal{X}$ differ only in their last component.

Now, due to the use of ordinary least squares, β_j is the pseudoinverse of the matrix $\begin{bmatrix}\mathcal{X}_v & \mathbf{y}_\Sigma\end{bmatrix}$ applied to $\mathbf{y}^{\mathcal{D}}_j$. Here, $\mathcal{X}_v$ is a T by n_c+1 matrix of in sample voltage measurements, $\mathbf{y}_\Sigma$ is a T dimensional vector of in sample power sums, and $\mathbf{y}^{\mathcal{D}}_j$ is a T dimensional vector of in sample power measurements. We can write the pseudoinverse in block form [196] to obtain

$$\begin{bmatrix} \boldsymbol{\beta}^{\mathcal{X}}_j \\ \beta^y_j \end{bmatrix} = \begin{bmatrix} (\mathcal{X}_v^\mathsf{T} Q_y \mathcal{X}_v)^{-1}\mathcal{X}_v^\mathsf{T} Q_y \\ (\mathbf{y}_\Sigma^\mathsf{T} Q_\mathcal{X} \mathbf{y}_\Sigma)^{-1}\mathbf{y}_\Sigma^\mathsf{T} Q_\mathcal{X} \end{bmatrix} \mathbf{y}^{\mathcal{D}}_j \tag{A.20}$$

where Q_y and $Q_{\mathcal{X}}$ are the residual projection matrices

$$Q_y = \mathbf{I} - \mathbf{y}_\Sigma(\mathbf{y}_\Sigma^T\mathbf{y}_\Sigma)^{-1}\mathbf{y}_\Sigma^T$$

$$Q_{\mathcal{X}} = \mathbf{I} - \mathcal{X}_v(\mathcal{X}_v^T\mathcal{X}_v)^{-1}\mathcal{X}_v^T$$

Now, $\sum_j \mathbf{y}_j^{\mathcal{D}} = \mathbf{y}_\Sigma$, so summing (A.20) over j yields

$$\sum_j \begin{bmatrix} \boldsymbol{\beta}_j^{\mathcal{X}} \\ \beta_j^y \end{bmatrix} = \begin{bmatrix} (\mathcal{X}_v^{\mathsf{T}} Q_y \mathcal{X}_v)^{-1}\mathcal{X}_v^{\mathsf{T}} Q_y \mathbf{y}_\Sigma \\ (\mathbf{y}_\Sigma^{\mathsf{T}} Q_{\mathcal{X}} \mathbf{y}_\Sigma)^{-1}\mathbf{y}_\Sigma^{\mathsf{T}} Q_{\mathcal{X}} \mathbf{y}_\Sigma \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \tag{A.21}$$

where the $\mathbf{0}$ comes from the residual of the projection of $\mathbf{y}_\Sigma$ onto itself. (A.19) and (A.21) show that $\sum_j \left(\tilde{y}(t)_j^e - \tilde{y}(t)_j\right) = 0$. Vectorizing over time yields the left hand equality of the lemma. Finally, since

$$\hat{\mathbf{y}}_j^{\mathcal{D}} = \begin{bmatrix} \mathcal{X}_v & \mathbf{y}_\Sigma \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}_j^{\mathcal{X}} \\ \beta_j^y \end{bmatrix} \tag{A.22}$$

we have

$$\sum_j \hat{\mathbf{y}}_j^{\mathcal{D}} = \mathcal{X}_v \sum_j \boldsymbol{\beta}_j^{\mathcal{X}} + \mathbf{y}_\Sigma \sum_j \beta_j^y = \mathbf{y}_\Sigma = \sum_j \mathbf{y}_j^{\mathcal{D}} \tag{A.23}$$

Subtracting the leftmost term from the rightmost term yields the right hand equality of the lemma for in sample data. A similar argument shows that the above equation also holds for out of sample data. ■

Proof of Lemma 3.4.2. Repeating the derivation of (A.19), but omitting index

i from the sum yields

$$\sum_{j \neq i} \left(\tilde{y}(t)_j^e - \tilde{y}(t)_j \right) = (y(t)_i^e - y(t)_i)(1 - \beta_i^y) \tag{A.24}$$

The remaining terms present in the right hand side of (A.19) but absent in (A.24) are $-(y(t)_i^e - y(t)_i) \sum_{j \neq i} \beta_j^y$. Therefore it must be the case that $\tilde{y}(t)_i^e - \tilde{y}(t)_i = -(y(t)_i^e - y(t)_i) \sum_{j \neq i} \beta_j^y$. Vectorizing over time yields the result

$$\tilde{\mathbf{y}}_i^{(e)} - \tilde{\mathbf{y}}_i = -\sum_{j \neq i} \beta_j \mathbf{y}_i^s \tag{A.25}$$

■

Proof of Lemma 3.4.3. Consider the true (unestimated) model

$$p_j(t) = \mathbf{x}(t)^{\mathsf{T}} \mathbf{r}_j + c_j y_\Sigma(t) \tag{A.26}$$

Consider further the hypothetical scenario where *only* the transformer voltage deviates from its flat value. In this scenario, we have for customer j

$$p_j(t) = c_j y_\Sigma(t) \tag{A.27}$$

so $c_j = p_j(t) / \sum_{i=1} p_i(t)$ is the portion of the total power injection contributed by customer j in this scenario. But if all voltages are flat except the transformer voltage, then the power injections must all have the same sign, so $c_j \geq 0$.

Now β_j^y is an estimator of c_j. We will repeat its equation here:

$$\beta_j^y = (\mathbf{y}_\Sigma^\mathsf{T} Q_\mathcal{X} \mathbf{y}_\Sigma)^{-1} \mathbf{y}_\Sigma^\mathsf{T} Q_\mathcal{X} \mathbf{y}_j^\mathcal{D} \tag{A.28}$$

This estimator is *biased*. This is because the term $\mathbf{y}_\Sigma$ is confounded by the sum of all noise terms for each individual dependent variable. Thus this estimator suffers from the *Classical Errors in Variable Problem* [197]. But since this lemma only relies on the sign of β_j^y, this does not pose much of a problem. We still have

$$\text{plim } \beta_j^y = \lambda c_j,\ 0 < \lambda < 1 \tag{A.29}$$

Then there are two cases. If $c_j > 0$, then $\mathbb{P}(\beta_j^y < 0) \leq \mathbb{P}(|\beta_j^y - \lambda c_j| \geq \lambda c_j) \to 0$ as the training window length goes to infinity. Thus for any $\delta > 0$, there exists a window length T_1^j such that $\mathbb{P}(\beta_j^y < 0) < \frac{\delta}{2}$. If, however, $c_j = 0$, then there exists a window length T_2^j such that $\mathbb{P}(|\beta_j^y| \geq \delta) < \frac{\delta}{2}$. Let $T^j = \max(T_1^j, T_2^j)$. Then for window length T^j, $\mathbb{P}(\beta_j < -\delta) \leq 2(\frac{\delta}{2}) = \delta$. Letting $T = \max_j\{T^j\}$ completes the proof. ■

Appendix B

Appendices for Chapter 6

B.1 Proof of Lemma 6.3.1, Lemma 6.3.2, and Theorem 6.3.3

Proof of Lemma 6.3.1. Defining the augmented reward $r^\pi(s,a) = r(s,a) - \tau\mathbb{E}_{s'\sim p}D_{\mathrm{KL}}(\pi(\cdot|s')||\pi^b(\cdot|s'))$, the operator $\mathcal{T}^\pi$ can be expressed as:

$$\mathcal{T}^\pi q(s,a) = r^\pi(s,a) + \gamma\mathbb{E}_{s'\sim P}\mathbb{E}_{a'\sim\pi}[q(s',a')]$$

Or in vector notation $\mathcal{T}^\pi q = r^\pi + \gamma P^\pi q$, where the entry of the vector $P^\pi q$ is given by

$$(P^\pi q)(s,a) = \int_{\mathcal{S}\times\mathcal{A}} q(s',a')dP^\pi(s',a'|s,a)$$

$P^\pi(s', a'|s, a) = p(s'|s, a)\pi(a'|s')$. As $D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s))$ is assumed to be bounded for all s, $r^\pi(s, a)$ is bounded for all s, a. Therefore, for any $q(s, a) \in \mathbb{R}, q'(s, a) \in \mathbb{R}, \forall (s, a) \in \mathcal{S} \times \mathcal{A}$:

$$||\mathcal{T}^\pi q - \mathcal{T}^\pi q'||_\infty = \gamma ||P^\pi(q - q')||_\infty \leq \gamma ||q - q'||_\infty$$

The inequality is due to $||P^\pi||_\infty = 1$. Therefore, for any $\gamma < 1$, $\mathcal{T}^\pi$ is a contraction mapping with respect to the supremum norm. By the Banach fixed point theorem, the operator $\mathcal{T}^\pi$ has a unique fixed point q^{d}_π and the sequence defined by $q^{k+1} = \mathcal{T}^\pi q^k$ converges to this fixed point as $k \to \infty$. ■

Proof of Lemma 6.3.2. Since $\pi'(\cdot|s)$ is a maximizer of the objective function $J_\pi(\tilde{\pi}(\cdot|s)) = \mathbb{E}_{a\sim\tilde{\pi}}[q^{\mathrm{d}}_\pi(s, a)] - \tau D_{\mathrm{KL}}(\tilde{\pi}(\cdot|s)||\pi^b(\cdot|s))$, therefore we have $J_\pi(\pi'(\cdot|s)) \geq J_\pi(\pi(\cdot|s))$. Thus

$$\begin{aligned} &\mathbb{E}_{a\sim\pi'}[q^{\mathrm{d}}_\pi(s,a)] - \tau D_{\mathrm{KL}}(\pi'(\cdot|s)||\pi^b(\cdot|s)) \geq \\ &\qquad \mathbb{E}_{a\sim\pi}[q^{\mathrm{d}}_\pi(s, a)] - \tau D_{\mathrm{KL}}(\pi(\cdot|s)||\pi^b(\cdot|s)) \triangleq v^{\mathrm{d}}_\pi(s) \end{aligned} \tag{B.1}$$

Let (B.1) holds for every $s \in \mathcal{S}$, we can obtain the following chain of inequalities by repeatedly invoking (B.1) and (6.12):

$$\begin{aligned}
q^{\mathrm{d}}_{\pi}(s,a) &= r + \gamma \mathbb{E}_{s'\sim P}[v^{\mathrm{d}}_{\pi}(s')] \\
&\leq r + \gamma \mathbb{E}_{s'\sim P}[\mathbb{E}_{a'\sim \pi'}[q^{\mathrm{d}}_{\pi}(s',a')] \\
&\qquad\qquad - \tau D_{\mathrm{KL}}(\pi'(\cdot|s')||\pi^{b}(\cdot|s'))] \\
&= r + \gamma \mathbb{E}_{s'\sim P}[\mathbb{E}_{a'\sim \pi'}[r + \gamma \mathbb{E}_{s''\sim P}[v^{\mathrm{d}}_{\pi}(s'')]] \\
&\qquad\qquad - \tau D_{\mathrm{KL}}(\pi'(\cdot|s')||\pi^{b}(\cdot|s'))] \\
&\leq \ldots \\
&\leq q^{\mathrm{d}}_{\pi'}(s,a) \qquad \text{(B.2)}
\end{aligned}$$

Continuously expanding the terms, we obtain $q^{\mathrm{d}}_{\pi'}(s,a)$ on the right hand side by its definition. ■

Proof of Theorem 6.3.3. The sequence $q^{\mathrm{d}}_{\pi^k}(s,a), k = 1, 2, \ldots$ generated by repeated applications of the policy evaluation and improvement is non-decreasing and is bounded above. Thus convergence follows from the monotone convergence principle. Denote $q^{\mathrm{d}}_{\pi^\infty}(s,a)$ as the converged value function and π^∞ the associated policy. We need to show that π^∞ is indeed optimal. Since at convergence, the policy is no longer changing. Therefore $\pi^\infty(\cdot|s)$ is a maximizer of the objective function $J_{\pi^\infty}(\tilde{\pi}(\cdot|s)) = \mathbb{E}_{a\sim\tilde{\pi}}[q^{\mathrm{d}}_{\pi^\infty}(s,a)] - \tau D_{\mathrm{KL}}(\tilde{\pi}(\cdot|s)||\pi^b(\cdot|s))$. In other words, $J_{\pi^\infty}(\pi(\cdot|s)) \leq J_{\pi^\infty}(\pi^\infty(\cdot|s))$ for any policy π. By the same token as the proof of Lemma 2, this means that $q^{\mathrm{d}}_{\pi}(s,a) \leq q^{\mathrm{d}}_{\pi^\infty}(s,a)$. Since π is an arbitrary policy, π^∞ is indeed the optimal policy. ■

Appendix C

Appendices for Chapter 7

C.1 Proof of Proposition 7.4.1 and Proposition 7.4.2

Lemma C.1.1 *Let (X, d_X) and (Y, d_Y) be two metric spaces and (Y, Σ, μ) be a measure space. Further, let Σ be generated by the open sets in (Y, d_Y); and μ has full support in the sense that $\mu(S) > 0$ for all non-empty open sets S in Σ. Let $f : X \times Y \mapsto \mathbb{R}^k, k \geq 1$ be a non-negative function that is continuous for every x. Then the two sets C and D are equal:*

$$C = \{x | f(x, y) = 0, \forall y \in Y\}$$
$$D = \{x | \int f(x, y) d\mu(y) = 0\}$$

Equalities and inequalities are understood to be element-wise.

Proof. It is clear that $C \subseteq D$ since the condition in C implies that in D. To demonstrate that they are equal, let a point $x \notin C$, therefore $f_i(x, y) = c > 0$ for some y and some coordinate i of f. Then by the continuity of $f_i(x, \cdot)$, for every $\epsilon > 0$, there exists

$\delta > 0$, such that the condition $d(f_i(x, y'), f_i(x, y)) < \epsilon$ is satisfied for every $y' \in B^\delta(y) \triangleq \{y' | d_Y(y', y) < \delta\}$. Pick a small enough $\epsilon < c$, then we have

$$\begin{aligned}\int f_i(x, y')d\mu(y') &\geq \int_{B^\delta(y)} f_i(x, y')d\mu(y') \\ &\geq (c - \epsilon)\mu(B^\delta(y)) \\ &> 0\end{aligned}$$

The last inequality is due to the full support assumption of μ. This shows that $x \notin D$. Therefore $C = D$. ■

Proof of Proposition 7.4.1. We identify the space of all neural network weights $W \subseteq \mathbb{R}^N$ with Euclidean metric as the X space in Lemma C.1.1. We can identify the state-action space $\mathcal{S} \times \mathcal{A}$ as the Y space by defining a metric d on $\mathcal{S} \times \mathcal{A}$. The measure $\mu_{\mathcal{SA}}$ on the state-action space has the full support property stated in Lemma C.1.1. We identify the function f as $\mathbf{h}(\bar{D}_{ii}\zeta_{\varphi_i}(s, a) - \bar{A}_i\boldsymbol{\zeta}_{\boldsymbol{\varphi}}(s, a))$ in the statement of Proposition 7.4.1. We further identify that the set C and D in Lemma C.1.1 correspond to the constraint set in (7.15) and the set expressed by (7.16). By Lemma C.1.1, these two sets are equal. ■

Lemma C.1.2 *Consider the situation in Lemma C.1.1 except that f may not be continuous, and that μ may not have full support. Then the following two sets are equal:*

$$\begin{aligned}C &= \{x | \mu(\{y | f_i(x, y) > 0 \textit{ for some } i\}) = 0\} \\ D &= \{x | \int f(x, y)d\mu(y) = 0\}\end{aligned}$$

Proof. Let $x \in C$, then $\int f(x,y)d\mu(y) = 0$ since the set $f(x,y) > 0$ has measure zero. Thus $x \in D$. Hence $C \subseteq D$. On the other hand, let $x \in D$, define two sets $N_1(x) = \{y | f_i(x,y) > 0 \text{ for some } i\}$ and $N_2(x) = \{y | f(x,y) = 0\}$. Thus

$$\begin{aligned}\int f(x,y)d\mu(y) = 0 & \\ &= \int_{N_1(x)} f(x,y)d\mu(y) + \int_{N_2(x)} f(x,y)d\mu(y) \\ &= \int_{N_1(x)} f(x,y)d\mu(y)\end{aligned}$$

Thus $\mu(N_1(x)) = 0$ and therefore $x \in C$. Hence $D \subseteq C$. Combining the two directions shows that $C = D$. Therefore, every point x in D satisfies $f(x,y) = 0$ for most of y except for some y with measure zero. ■

Proof of Proposition 7.4.2. Ditto as proof of Proposition 7.4.1. Since the measure $\mu_{\mathcal{SA}}$ can be arbitrary, it can be selected as full support which covers almost all s, a pairs. ■