Optimal Coordination of Distributed Energy Resources in Smart Grids Enabled by Distributed Optimization and Transactive Energy

by

Rabab Haider

B.Sc. University of Toronto (2018)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2020

Signature redacted

Author .....
Department of Mechanical Engineering
January 15, 2020

Signature redacted

Certified by .....
Anuradha M. Annaswamy
Senior Research Scientist, Department of Mechanical Engineering
Signature redacted Thesis Supervisor

Accepted by .....
Nicolas Hadjiconstantinou
Professor of Mechanical Engineering
Chairman, Department Committee on Graduate Theses

See Appendix C - MATLAB Code
p.213 - 301

Optimal Coordination of Distributed Energy Resources in Smart Grids Enabled by Distributed Optimization and Transactive Energy

by

Rabab Haider

Submitted to the Department of Mechanical Engineering
on January 15, 2020, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

Modern active distribution grids are characterized by the increasing penetration of distributed energy resources (DERs). The proper coordination and scheduling of a large numbers of these DERs can only be achieved at the nexus of new technological approaches and policies, primarily distributed computation and transactive energy. Transactive energy is a control mechanism which uses economic incentives, such as time-of-use or real-time electricity prices, to influence the behaviour of independent agents (i.e. DERs in the grid) as needed by the grid operator. This thesis tackles the problem of DER coordination by considering the role of distributed optimization algorithms in solving the optimal power flow (OPF) problem, when a large number of small scale DERs are present. The OPF problem minimizes costs to operate the grid, while subject to network constraints. The distributed implementation makes large-scale problems computationally tractable, while also maintaining privacy of local information. First, we utilize a new convex formulation of the power grid based on current injection (CI) and McCormick Envelopes, to model distribution grids of meshed topology and unbalanced structure. The OPF problem is then solved for such grid structures, using the distributed proximal atomic coordination (PAC) algorithm, which has several advantages over other distributed algorithms. These advantages include reduced network communication requirements, reduced local computational effort, and improved privacy. The DER coordination problem is then extended to consider storage devices. This requires a multi-period OPF formulation, which is also solved using the PAC algorithm. Results from active configurations of the IEEE 123-bus network corroborate the need for a multi-period optimization approach under high penetration of renewable resources. Finally, we propose a retail market mechanism which can be viewed as a transactive energy scheme within the distribution grid. Through the retail market, DERs are optimally scheduled in an energy market, and leveraged in alert system cases in an ancillary services market. The transactions of the energy market are carried out at each primacy feeder through bilateral agreements

between the Distribution System Operator (DSO) and agents representing DERs at feeder buses, leveraging the PAC algorithm. These interactions determine the optimal real-time resource schedules and distributed Locational Marginal Price. The PAC algorithm and proposed market are extensively validated on a real distribution grid in Tokyo, a balanced IEEE 123-bus distribution grid, and a three feeder model using the IEEE 13-bus. The energy market is shown to result in an overall increase in revenue for the DSO.

Thesis Supervisor: Anuradha M. Annaswamy
Title: Senior Research Scientist, Department of Mechanical Engineering

Acknowledgments

To all those countless people, experiences, and mishaps which have made it possible for me to arrive at the steps of MIT.

To my advisor for her words of wisdom, and reminders that research is the process of uncovering truths.

To my friends for their well-timed moments of levity.

To my family, and especially my parents, who have guided me and supported me through all endeavours. Your boundless love and infinite encouragement will always get me through.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The modern distribution grid is characterized by the high penetration of Distributed Energy Resources (DERs), including distributed generation (DG) units, flexible consumption units (demand response, DR), and storage. The large majority of these DERs are small-scale resources which are located behind-the-meter, meaning they are not visible to utilities or control authorities. They are also largely owned by different third-party agents. These characteristics - the distributed nature, small scale, and third-party ownership - make the efficient integration of these resources into the grid a challenging and open research problem.

The penetration of small-scale DERs in the distribution grid is expected to continue growing. Figure 1-1 shows the breakdown of these behind-the-meter resources in 2017, which contributed to 46.4GW of impact on the US power peak. The expected growth is shown in Fig. 1-2, reaching upwards of 104.2GW by 2023. This amounts to a substantial percentage of peak summer load of around 15%[1]. The growth of these resources is led by distributed solar (primarily rooftop solar photovoltaic, PV), increased capacity in DR from smart home appliances, and increasing penetration of electric vehicles (EV) and home charging infrastructures (distributed storage) [1]. If efficiently utilized, these DERs are a valuable source of flexibility to grid operators. Namely, DGs can be used to reduce costly power imported from the main grid; storage can be used for storing excess renewable generation or for energy arbitrage; and

[1]In 2016, the US had 769 GW summer peak load [1].

DR provides necessary load-side flexibility under the new paradigm of uncontrollable generation [2].

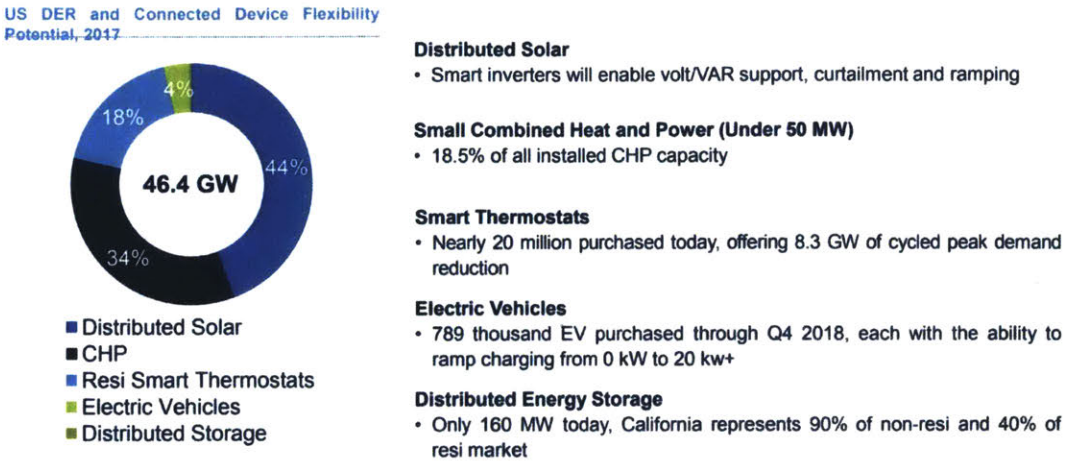


Figure 1-1: Breakdown of behind-the-meter DER potential in the US, for 2017, as measured by GTM Research and Department of Energy [1].

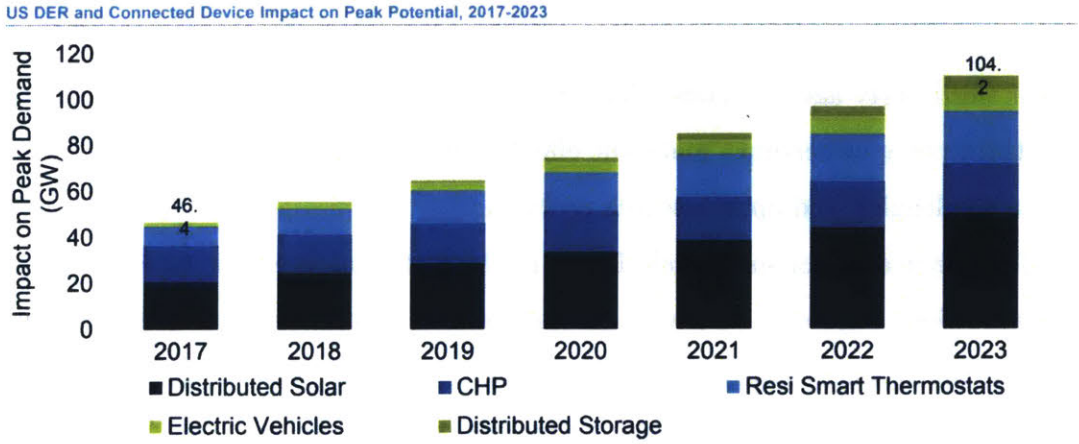


Figure 1-2: Forecasted growth of behind-the-meter DER impact on peak potential, showing growth from 46.4 GW in 2017 to 104.2 GW in 2023, leading to large penetration of these flexible resources, as measured by GTM Research and Department of Energy [1].

Efficiently leveraging this flexibility, however, is challenging. These DERs are

[2]The changing energy generation mix includes many renewable resources, which cannot be controlled the way traditional resources are. We cannot control when the sun shines or wind blows. As such, generation can no longer be assumed to follow uncontrollable demand. To address this, we must have some controllable components in demand - this is where demand-side management steps in. More on this later.

typically located behind-the-meter, and as such are not visible to or accessible by control authorities. Most small-scale renewable resources are solar PV: these are highly variable resources whose output are entirely dependent on weather and cloud cover. Accurate long-term solar forecasting is an open area of research - even 24hr advance forecasting has high error margins. Another challenge comes in forecasting demand, where net metering and EV charging can change consumer behaviour patterns. Historical data which is currently (and successfully) used by grid operators to forecast demand may no longer be accurate in predicting future demand patterns. The private ownership of DERs also introduces concerns for data privacy and user autonomy. Many distribution grid DERs such as small-scale storage and DGs are owned by small businesses in the distribution grid, which may prefer to retain operational control over the resources. These resource owners should then have the option to behave on an opt-in basis, rather than being required to cede control to and share private information with a central authority [3].

The optimization problem (optimal power flow, OPF) which is needed to efficiently schedule such a large number of resources (i.e. when and how each DER is being utilized), which are all owned by different independent agents, is a computationally challenging task. In today's electricity system, a similar OPF problem is solved by transmission grid operators and in the wholesale electricity market (WEM). The transmission level OPF is concerned primarily with large-scale transmission level resources, and determines their dispatch and scheduling decisions, along with the electricity prices at every node in the transmission network (termed the locational marginal prices, LMPs). In general, the OPF problem determines the desired set point of resources constrained by power flow physics, grid limits, and meeting electricity demand, all subject to minimizing the cost of operating the grid. These OPF formulations involve a large number of variables to optimize over, complex feasibility regions, and may have complicated objective functions. The computational chal-

[3]Note that many smart home appliances can be remotely controlled, such as smart thermostats or refrigeration units. These resources are often aggregated within a residential building or unit. Access is given to a central authority (like a utility), which can then adjust set points in the building based on power generation and demand across the network. This is another form of demand response, but is not the key focus of this thesis.

lenges are further increased when considering integer variables representing on-off unit commitment decisions.

Compared to the transmission grid, however, the OPF formulation for the distribution grid is more complex. An active distribution grid with high DER penetration will have many more resources, decision variables, and little to no large controllable resources. In addition, the physical structure of the distribution grid varies greatly from the transmission system: distribution grids can have both radial and meshed topologies and are highly unbalanced electrical systems, while the transmission grid is a balanced 3-phase system. These topological and structural differences render power flow models such as the popular Branch Flow (BF) model [5–7] unsuitable for OPF formulations for the distribution grid [8]; as such, new power flow models are required.

Moreover, the increased penetration of renewable generation (both at the transmission level and locally within distribution grids) motivates the need for demand-side management (such as DR) and storage. Storage can include large grid-scale batteries which are typically owned and/or operated by utilities, small-scale batteries on-site of businesses or residential complexes, and electric vehicles. Regardless of size, all storage devices introduce new complexity to the OPF problem: there are constraints coupled in time from the storage state of charge. Discharging a storage unit in one period leaves less charged capacity for a future period. This requires the need for multi-period optimization, to account for the impact current actions have on future capabilities, and increases the size of the already large-scale optimization problem.

Finally, the key to efficient resource scheduling in distribution grids is to utilize their flexibility, while also maintaining privacy and autonomy of DERs. This autonomy, or indirect control of resources, is often termed "economic" or "transactive" control. The concept of transactive energy uses a service-based value of power generation or consumption to influence desirable behaviours from autonomous agents (i.e. DERs) at fast timescales [9, 10]. Price signals are used as the control signal. Agents are compensated for the services they provide to the grid based on appropriately designed market derivates. This "appropriate" design takes the form of a retail market

structure which behaves much like the WEM, but within the distribution grid. The challenge here is to solve OPF problems in the distribution grid for resource scheduling and determining distribution-level LMPs (d-LMPs) through coordinated markets in a computationally tractable way.

The large-scale nature of the OPF problem applied to the distribution grid would render centralized approaches [4] to perform poorly. To make these large-scale optimization problems tractable, various distributed optimization approaches have been developed, including many that employ distributed alternating direction method of multipliers (ADMM) [5, 6, 11–19] and proximal atomic coordination (PAC) [20, 21]. These distributed solvers solve the original centralized optimization formulation through three key ideas. First, they properly allocate the computational burden of optimizing the problem's variables amongst different processing units. This can be different processing units housed at the same facility, or units in different geographic locations which are also owned by different parties; we refer to these parties are "agents". Second, they exploit parallel or sequential computation to speed up iteration convergence. Third, they ensure that all agents agree on coupling variables so that the distributed solution matches the original centralized solution, and a common objective [5] is reached. To achieve agreement, agents must (at a minimum) communicate with its' peers/neighbours. Ideally, a distributed algorithm should minimize the data shared and communication requirements between agents. Since distributed methods rely primarily on local computations at each node and peer-to-peer (P2P) exchange of information, they are computationally tractable [6]. Moreover, if well designed, distributed algorithms can be resilient to communication link and single-point failures. Finally, the most attractive distributed algorithms will preserve the private information of agents, while still realizing global objectives and meeting global constraints. Indeed, distributed methods that are able to accomplish the above goals are highly useful tools for solving optimization problems in various engineering and data science

[4] which are the best understood and most widely implemented class of optimization solvers

[5] This is the objective function of the central optimization problem.

[6] Especially compared to central methods which require all agents to send data to a central entity, which then solves a huge optimization problem.

problems. Further discussion of distributed optimization, and in particular the PAC algorithm, is provided in Chapter 3.

In the context of distribution grids, the centralized optimization problem can be distributed across the DERs, which are assumed to be independent agents with computational capabilities [7]. Private information for DERs can include the objective functions (operating costs), operating constraints, and price of electricity (either to be paid to or by the DER), and we desire for these to remain private. The PAC algorithm has been applied to the OPF problem for radial balanced networks in [20], to the OPF problem for meshed unbalanced networks in [8] (contributions by this author, further discussed in Chapter 3), to the multi-period OPF problem (further discussed in Chapter 4), and for coordinating a retail market in [22] (by this author, further discussed in Chapter 5). A discussion on privacy of the PAC algorithm in the context of electricity markets is also presented in [22] and Chapter 5.

1.1 Summary of Contributions

The high penetration of DERs in distribution grids provides operational challenges when considering the optimal coordination of these resources. The three pillars of this problem are: the high dimensionality of the OPF problem; the coupling constraints introduced by storage devices; and the coordination of devices from independent owners. To address these challenges, we need three corresponding tools: a distributed computation approach; multi-period optimization; and a financial structure viewed as a transactive energy scheme. These are the main contributions of this thesis. The three components are each described below.

Optimal Power Flow for Meshed and Unbalanced Grids: Solutions with Distributed Optimization. Distributed computation is applied to a new convex relaxation of the OPF problem, based on current injection (CI) and McCormick Envelopes (MCE), which can model meshed and unbalanced distribution grids. An

[7] This is an assumption made for simplicity. It is not a requirement from the design of the algorithm, nor the specific problem of distribution grid optimization.

eigenvalue analysis of the CI model and a proposed reformulation is carried out to show how modelling decisions influence the applicability of distributed computation, with a focus on convergence parameters of the PAC algorithm. The computational performance of the PAC when using the CI model is compared to that of the Branch Flow (BF) model, which is a popular OPF formulation, but is applicable only to radial balanced grids.

A Story of Storage: Multi-period Distributed Optimization for Distribution Grids. The CI formulation of the OPF problem and the PAC algorithm are extended to include decision variables with state retention, which require multi-period optimization. Distributed computation is once again a tool used to tackle the large scale problem of resource scheduling when there are many independent participating agents within the distribution grid. This multi-period extension enables storage to be considered within the dispatch problem.

Using Transactive Energy: Proposal for a DSO-Centric Retail Electricity Market for Smart Distribution Grids. A novel retail market mechanism for distribution grids under high penetration of DERs is proposed, where decisions are carried out by a Distribution System Operator (DSO). The DSO oversees two markets: (1) an energy market for DER scheduling and real-time market settlements (d-LMP) through bilateral contracts; and (2) an ancillary market which oversees transactions for alert conditions between primary feeders in the distribution network. The DSO acts like a proactive utility, in that it accepts the LMP as traditionally determined by the WEM, but optimally uses DERs within the distribution network, and only requests service from the WEM for net loads beyond its DER capabilities. The proposed market structure is numerically verified, showing potential savings for the DSO under this new market mechanism.

The rest of the thesis is organized as follows. Chapter 2 introduces necessary background information that is required for sufficient understanding of the contributions of this thesis, including an introduction to power physics, the changing distribution grid, and optimization theory. Chapter 3 introduces the CI model and the PAC algorithm, applies the distributed computation to the OPF problem, and makes computational

comparisons between the CI model and BF model. Chapter 4 discusses the extension of the PAC algorithm to multi-period optimization, introduces models for DERs, and presents simulation results for an active configuration of the IEEE 123-bus network. Chapter 5 provides an overview of the US power system, with a focus on electricity markets, then introduces the proposed DSO-centric retail market with validation metrics and simulation results on three different test networks. Finally, conclusions and future research directions are provided.

Chapter 2

Background

This chapter provides a brief introduction to several topics necessary to understand the material in the thesis. Key nomenclature, notation, and terminology used throughout the thesis will be introduced. An overview of electric power systems, the changing distribution grid, and optimization theory will be provided.

2.1 Introduction to the Electric Power System

The electric power system is a large, highly complex dynamic system operating over varying spatial and temporal scales. Power systems describes the interconnected subnetworks of generation, transmission, and distribution. Generation has typically been comprised of larger, centralized units such as coal and gas-fired power plants, nuclear plants, and hydroelectric plants. Once power has been generated [1], it is carried across the transmission system, which consists of high voltage AC lines. Through a series of primary and secondary substations, the electrical power is stepped down to lower voltages in the distribution network, which then supply end loads. A more complete discussion of the transmission and distribution networks will follow in a subsequent subsection.

[1] We engineers use this term loosely, as energy cannot be in fact created nor destroyed. Rather, more appropriate terminology would be "energy has been converted from chemical or thermal energy to electrical energy".

2.1.1 Fundamentals of Electric Power

The basic power physics variables are current, I, and voltage, V. Loosely speaking, these variables measure the flow of charge, and the difference in electrical potential energy, respectively. In an ideal world where all elements are lossless, the voltage drop across a section of wire would be zero. However, when considering realistic systems, and frankly interesting problems, losses in the form of voltage drop or changes across electrical elements becomes important. The three fundamental components are:

- Resistance, R, measured in Ohms [Ω]. Voltage drop over an element or segment of wire, as described by Ohm's Law: $V(t) = RI(t)$
- Capacitance, C, measured in Farads [F]. Energy stored in an electric field (between two parallel plates): $I(t) = C\frac{\mathrm{d}}{\mathrm{d}t}V(t)$
- Inductance, L, measured in Henrys [H]. Change of voltage w.r.t. magnetic field strength: $V(t) = L\frac{\mathrm{d}}{\mathrm{d}t}I(t)$

By way of the Fourier Transform and using complex numbers, a simpler representation of these elements can be obtained, where $\omega = 2\pi f$, and f is the signal frequency, and j is the imaginary number.

$$
\begin{aligned}
R &\longrightarrow R \\
L &\longrightarrow j\omega L \\
C &\longrightarrow \frac{1}{j\omega C} = -\frac{j}{\omega C}
\end{aligned}
$$

Every electrical system can be described using these three elements. More precisely, we define the reactance, impedance, and admittance, to fully describe an electrical component, along with the resistance.

- Reactance, X, measured in Ohms [Ω]. $X = \omega L - \frac{1}{\omega C}$
- impedance, Z, measured in Ohms [Ω]. $Z = R + jX$
- Admittance, Y, measured in Siemens [S]. $Y = 1/Z$

Thus the electrical relationship can be simplified to

$$V(t) = ZI(t) \tag{2.1}$$

$$I(t) = YI(t) \tag{2.2}$$

By introducing reactance into our model, both voltage and current also take on complex values. When dealing with complex numbers, there are three equivalent notations: the rectangular, polar, and exponential forms. Polar form is often called *phasor notation* when studying electrical systems. The exponential form is derived from Euler's Identity. For an arbitrary complex number $X = a + jb$

- Rectangular Form: $X = a + jb$
- Polar Form: $X = A\angle\theta$, where $A = \sqrt{a^2 + b^2}$ and $\theta = \tan^{-1}(y/x)$
- Exponential Form: $X = A(\cos\theta + j\sin\theta)$

These representations can be used interchangeably, although the ease with which the complex variables can then be manipulated depends on the application. While they are all equivalent, the polar and exponential forms provide some key insights. For a purely resistive network ($X = 0$), angles will be zero; for a purely reactive network ($R = 0$), angles will be 90°. Thus for a network with both resistive and reactive components, the angle will depend on the ratio of these elements. Further, the angle of the current, ϕ, and the voltage, ψ, are related to one another by the ratio of the resistance and reactance in the network. We will revisit this after discussing electrical power and AC circuits.

With knowledge of current and voltages in our network, one can calculate the power drop across elements. When considering a circuit with reactive components, the resulting power is complex (known as apparent power, S), with real and reactive

parts, denoted by P and Q respectively:

$$S = V(t)I(t)^* \tag{2.3}$$

$$S = |I(t)|^2 Z = \frac{|V(t)|^2}{Z^*} \tag{2.4}$$

$$S = P + jQ \tag{2.5}$$

where $|\bullet|^*$ indicates the complex conjugate. The real and reactive powers can also be expressed as functions of the complex power and the power angle, as below:

$$P = |S| \cos\theta \tag{2.6}$$

$$Q = |S| \sin\theta \tag{2.7}$$

The concept of real and reactive power extends beyond being a mathematical construct. In the most fundamental sense, real power P is the power used to do actual work (dissipated by the load). Reactive power Q represents energy exchange between reactive loads and parasitic elements (such as inductors and capacitors), and does not contribute to work done. This "sloshing" back and forth of power increases the overall current flowing in a circuit without contributing to any work, resulting in increased thermal losses due to power line resistance. The power dissipated over an element (or a segment of wire) is calculated as

$$P_w = I^2 R \tag{2.8}$$

In order to minimize losses in a network, lower current is desirable. Thus, high voltage lines are used, as halving the current (equivalently doubling the voltage) results in a quarter of the losses due to the lines resistance. However, since loads are typically 120/240 V, transformers are needed to step up voltages for transmission, and step down voltages for distribution. A basic transformer consists of an ferromagnetic core which can sustain a magnetic flux, and two sets of coils wrapped around either side - a primary coil with N_{pri} loops and a secondary coil with N_{sec} loops. Transformers

make use of Ampere's Law, where a time-varying current induces a time-varying magnetic field, and Faraday's Law of Induction, where a time-varying magnetic field in turn induces an electromotive force (EMF) which drives a time-varying current. The primary and secondary voltages are related to the turns of coil; for an ideal (lossless) transformer the ratio is

$$\frac{V_{\text{pri}}}{V_{\text{sec}}} = \frac{N_{\text{pri}}}{N_{\text{sec}}} \tag{2.9}$$

Since transformers rely on time-varying currents to manipulate voltages, the electric power system is built on alternating current (AC), rather than direct current (DC) which is not time-varying. AC current typically follows a sinusoidal waveform, though some specific applications may require a square or triangular waveform. Since power is a product of the voltage and current, both of which are time-varying signals, the power transfer is dependent on the phase angle between the two signals. Referring back to the discussion on phasors, the difference in angle, $\theta = \psi - \phi$, describes the quality of power transfer. This is denoted as the power factor, pf:

$$\text{pf} = \cos\theta = \frac{P}{|S|} \tag{2.10}$$

$$\text{equivalently: } \text{pf} = \cos\arctan\frac{Q}{P} \tag{2.11}$$

$$\text{and: } Q = P\tan\arccos\text{pf} \tag{2.12}$$

A high power factor (close to 1) is preferred. Comparing two loads which draw the same current at the same voltage, but have different power factors, the load with the lower pf will have less useful power transferred - namely, $P_{\text{lowpf}} < P_{\text{highpf}}$. This means there are more resistive losses due to the higher current, for less work being done. Power factor correction is frequently employed using power electronics to increase the power factor of a load, and thereby improve the efficiency of the distribution system. Note a negative power factor implies the device is generating power - and similarly, a power factor of -1 indicates better quality of generation.

Within a given circuit, the currents and voltages can be determined based on the

generation and load at the different nodes. The two fundamental laws, Kirchhoff's Current and Voltage Laws (KCL and KVL respectively) relate conservation of charge and conservation of energy respectively, to the electric power quantities. Namely, KCL dictates that the sum of currents into a node equals the sum of currents out of a node ($\sum I_{in} = \sum I_{out}$), while KVL dictates that the algebraic sum of voltage drops around a loop equals zero ($\sum V = 0$). These two laws, along with the definitions of power and losses over elements, fully describe the power flow throughout a circuit or electric grid.

2.1.2 Electric Power Grid

The traditional electric grid is composed of three main sections: generation, transmission, and distribution, see Fig. 2-1. Generation has typically comprised of centralized large-scale powerplants, including coal or gas fired plants, nuclear, and/or hydro, located far from loads. The transmission grid consists of the necessary transformers, substations, and high-voltage power lines which transport the electricity from generation centers to load centers. Finally, the transmission grid is connected to the distribution network by step-down transformers located at substations. The distribution grid distributes the power to various loads, including industrial facilities, and residential units through distribution feeders. While large-scale renewable resources can be connected directly to the transmission system, such as wind farms or solar arrays, many smaller-scale renewables are connected to the distribution system, or are behind-the-meter (such as rooftop solar). With this, we focus on the distribution grid. However, to appreciate some of the more nuanced details, some background on transmission line modeling and the differences between transmission and distribution systems is warranted.

Three Phase Systems

From the discussion on AC power, higher efficiency in a network, when considering resistive losses, can be gained by using high voltage lines. However, this does not

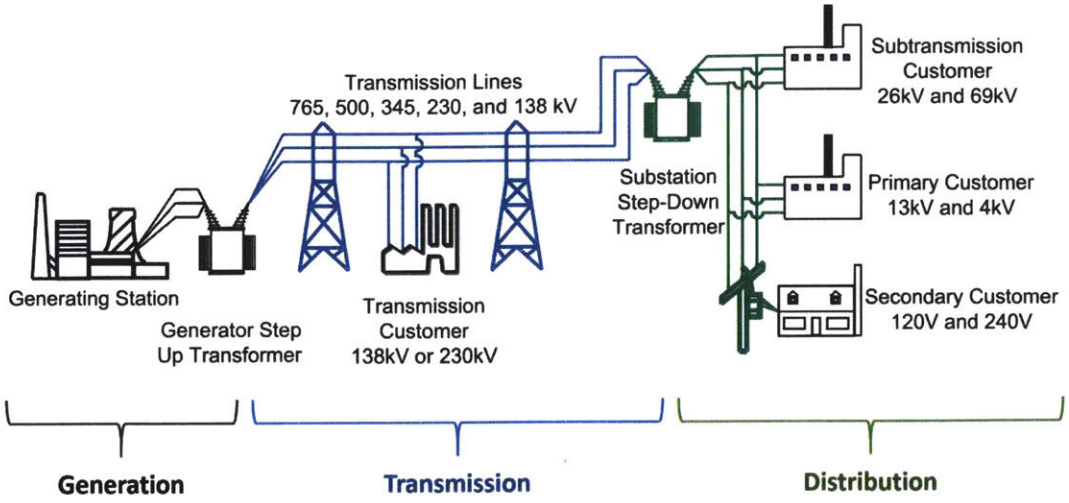


Figure 2-1: Traditional Electric Power Grid, showing generation, transmission, and distribution systems.

motivate why three-phase power is used within the power grid. To understand this, we consider the cases of single phase and polyphase (specifically two and three phase) power. We first recall that AC voltages and currents are of sinusoidal nature:

$$v(t) = V\cos(\omega t) \tag{2.13}$$

$$i(t) = I\cos(\omega t + \phi) \tag{2.14}$$

$$p(t) = v(t)i(t) = \frac{VI}{2}\cos(\phi)\left[1 + \cos(2\omega t)\right] + \frac{VI}{2}\sin(\phi)\sin(2\omega t) \tag{2.15}$$

- **Single Phase:** Consider an AC voltage source supplying power to a load with a single wire. From basic KCL, the current must have a return path to the source. Since the source is AC, the power delivered to the load is also time-varying. This, single-phase systems have non-zero return current and deliver non-constant power.

- **Two Phases:** Consider an AC voltage source supplying loads with two phases. We have two cases:

  - "Split Phase": The two phases are 180° apart. This lends itself to zero return current ($I_a + I_b = 0$). However, the total instantaneous power

delivered to the load is non-constant.

$$i_a(t) = I\cos(\omega t + \phi) \tag{2.16}$$

$$i_b(t) = I\cos(\omega t + \phi + \pi) \tag{2.17}$$

- "Two Phase": The two phases are 90° apart. This lends itself to constant instantaneous power delivered. However, there is non-zero return current ($I_a + I_b \neq 0$).

$$i_a(t) = I\cos(\omega t + \phi) \tag{2.18}$$

$$i_b(t) = I\cos(\omega t + \phi + \pi/2) \tag{2.19}$$

- **Three Phases:** Consider an AC voltage source supplying loads with three phases, where the phases are 120° apart. Similar to the split phase case, there is zero return current, eliminating the need for an extra return cable. Further, the instantaneous power delivered is constant.

$$i_a(t) = I\cos(\omega t + \phi) \tag{2.20}$$

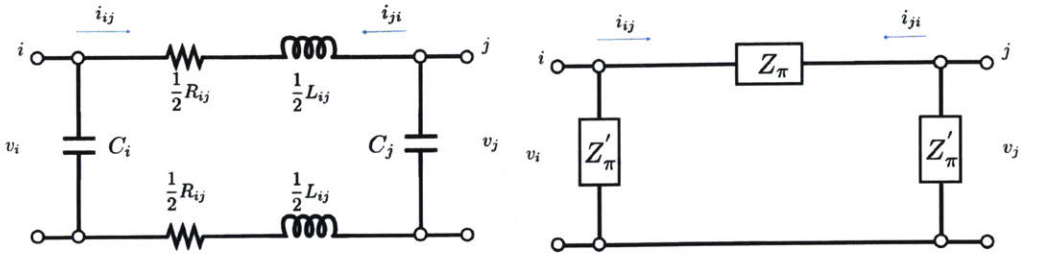
$$i_b(t) = I\cos(\omega t + \phi + 2\pi/3) \tag{2.21}$$

$$i_c(t) = I\cos(\omega t + \phi + 4\pi/3) \tag{2.22}$$

For these reasons (constant power delivery and zero return current), the power system uses three phase power.

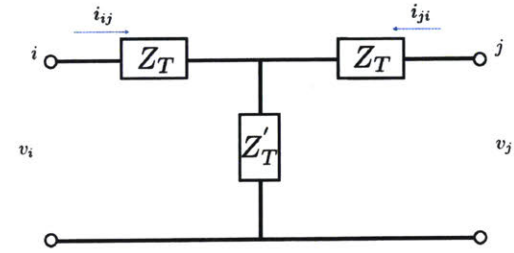
Modeling of Electrical Lines

A fundamental part of modeling the power system is to model the electrical lines. Consider a single power line, carrying a single phase of current. There will be electrical resistances and inductances along the length of the line, due to the conductor's electrical and electromagnetic properties. Additionally, the electromagnetic interactions with the physical ground (i.e. Earth's surface) can be modeled as a capacitor,

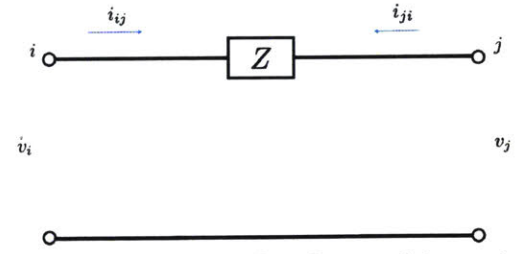


(a) Lumped parameter model of transmission line

(b) Approximation: Pi model of transmission line



(c) Approximation: T model of transmission line



(d) Approximation: Simplest model, nominal model for short lines

Figure 2-2: Approximate models of transmission line

as the charge imbalance between the wire and ground can be roughly imagined as two capacitor plates with air as the dielectric medium. Although these impedances occur over the entire power line, they can be approximated with lumped parameters, which describe the entire line with a single resistor, inductor, and capacitor between nodes **i** and **j**, as shown in Fig. 2-2a. For analysis, often approximations of this model are used to simplify the mathematics, lending to transmission line models in Fig. 2-2b and Fig. 2-2c.

The simplest case as shown in Fig. 2-2d is often used to model short lines, which considers the self-impedance ($Z = R + jX$) which is the wire's own resistance, R, and inductance, $X = \omega L$. However, in three phase power systems, the three conducting wires are in close proximity to one another, lending itself to electromagnetic interference. This is captured as line-to-line impedances, as shown in Fig. 2-3. For the three phase line, phases a, b and c, the matrix form of Ohm's law ($V_{abc} = ZI_{abc}$) can be

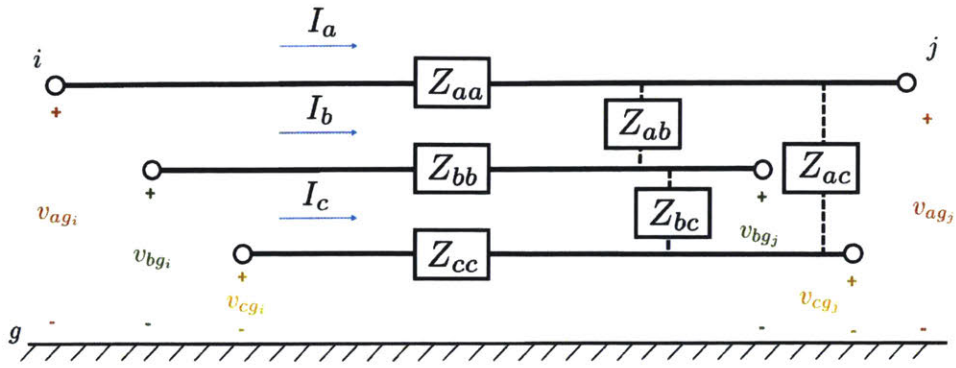


Figure 2-3: Complete model of three phase line segment

written as

$$\begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} = \begin{bmatrix} Z_{aa} & Z_{ab} & Z_{ac} \\ Z_{ab} & Z_{bb} & Z_{bc} \\ Z_{ac} & Z_{bc} & Z_{cc} \end{bmatrix} \begin{bmatrix} I_a \\ I_b \\ I_c \end{bmatrix} \tag{2.23}$$

where Z_{xx} are the self impedance and Z_{xy} are the mutual impedances between phases x and y. Note that for a single or two phase line, the missing line(s) will have zero entries in the impedance matrix.

Balanced and Unbalanced Systems

With three phase lines and the corresponding line models introduced, the concept of balanced and unbalanced systems can be discussed. We begin by introducing some basic concepts, then discussing their implications on modeling transmission and distribution grids.

Definitions In a balanced system, each phases' line will have identical electrical properties. Specifically, between node **i** and **j**, the impedance of each line is the same. In this case, when a balanced voltage is applied to each phase (same magnitude, just shifted by 120°) and the loads are the same, the current drawn will also be balanced (same magnitude, each shifted by 120°). In an unbalanced system, the electrical properties of the lines will be different. In this system, when a balanced voltage is applied and the loads are the same, the current magnitudes along each phase will be different,

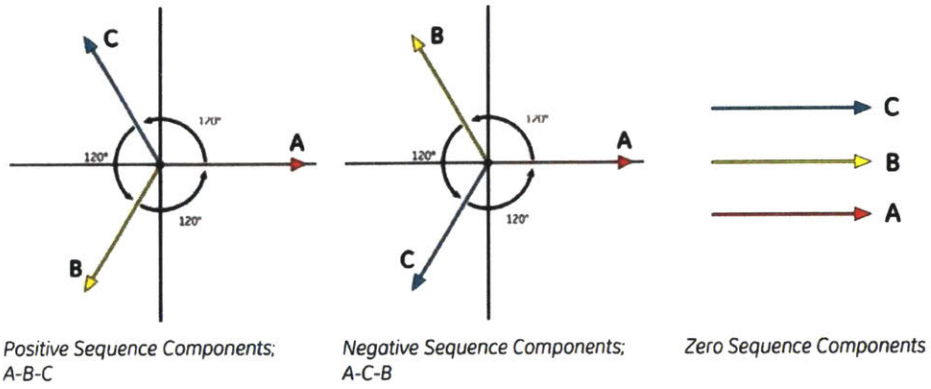


Figure 2-4: Graphical description of sequence domain components

and the phase shift between the voltage and currents of each phase ($\theta = \psi - \phi$) will also be different.

Sequence Domain Until now, the polyphase AC voltages and currents have been denoted by their phases, which for three phase systems are each 120° (equivalently $2\pi/3$) apart. This is known as the phase domain. An equivalent representation is in the sequence domain, using symmetrical components. The sequence domain consists of three sets of independent components: positive, negative, and zero sequences. The positive sequence is when phasors are 120° apart with a clockwise rotation. The negative sequence is when phasors are 120° apart with a counterclockwise rotation. The zero sequence is composed of phasors which are equal in magnitude and phase (colinear phasors). The sequence domain is shown in Fig. 2-4. The sequence domain will be denoted with subscript $[0+-]$, denoting the zero, positive, and negative sequences respectively.

The sequence domain can be used to determine if a voltage or current is balanced or unbalanced. In a balanced system, the sequence domain representation will have only a non-zero positive sequence component. An unbalanced system will also have a negative sequence and/or zero sequence component. The decomposition is shown in Fig. 2-5. Note in the balanced network, the three voltages are simply colinear vectors, while the negative and zero sequence vectors are zero. In the unbalanced case, there are non-zero negative and zero sequence components. The 'a' multiplicative factor in the diagram is a rotational operator, and will be discussed in the next section which

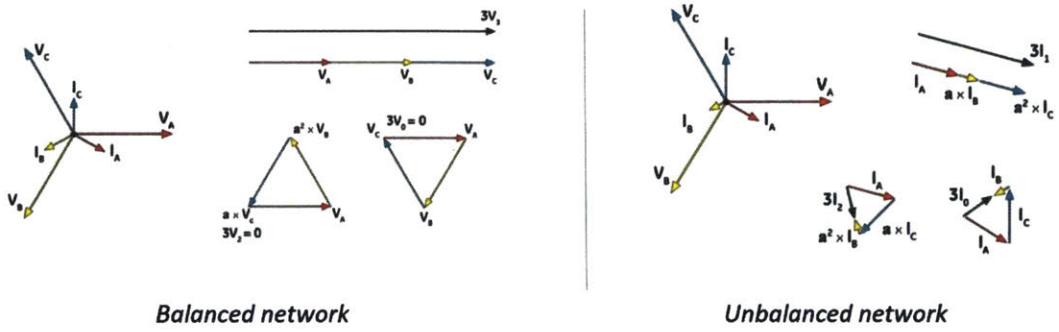


Figure 2-5: Examples showing decomposition of three-phase balanced (left) and unbalanced (right) systems, using sequence components. Here, the positive, negative, and zero sequence components are labeled as $1, 2,$ and 0 respectively.

introduces the Fortescue Transformation.

Fortescue Transformation According to Fortescue, who developed the methodology of working with the sequence domain in 1913, every three-phase system can be cast into sequence domain. The conversion from phase domain ($[Z_{abc}]$) to sequence domain ($[Z_{0+-}]$) is done through the Fortescue Transformation T, and is applicable to both balanced and unbalanced networks.

$$T = \begin{bmatrix} 1 & 1 & 1 \\ 1 & a^2 & a \\ 1 & a & a^2 \end{bmatrix} \text{ and } T^{-1} = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 1 & a & a^2 \\ 1 & a^2 & a \end{bmatrix} \tag{2.24}$$

Using Symmetrical Components The sequence domain is used when studying electrical faults. The calculated sequence components are used as data inputs to protective relays, which act to minimize the impact of the outage, which could entail islanding part of the grid. The sequence domain can also be used to understand the degree of imbalance in a network - if the negative and zero sequence components are large, the network has a high degree of imbalance. Symmetrical components can also make analysis of balanced networks much simpler. Balanced networks are created by transposing the cables, which is when the position of the cables relative to each other are swapped in intervals, as in Fig. 2-6. By twisting the cables in this way,

the mutual impedances between the phases are equal, $Z_{ab} = Z_{bc} = Z_{ac}$. The self and mutual impedances for a transposed cable can be easily calculated, and the balanced impedance is as below:

$$Z_s = \frac{1}{3}(Z_{aa} + Z_{bb} + Z_{cc}) \tag{2.25}$$

$$Z_m = \frac{1}{3}(Z_{ab} + Z_{bc} + Z_{ac}) \tag{2.26}$$

$$[Z_{\text{balanced}}] = \begin{bmatrix} Z_s & Z_m & Z_m \\ Z_m & Z_s & Z_m \\ Z_m & Z_m & Z_s \end{bmatrix} \tag{2.27}$$

By applying the Fortescue Transformation to the three-phase impedance matrix, the phase domain is obtained, as below

$$[Z_{0+-}] = T^{-1}[Z_{abc}]T = \begin{bmatrix} Z_{00} & Z_{0+} & Z_{0-} \\ Z_{+0} & Z_{++} & Z_{+-} \\ Z_{-0} & Z_{-+} & Z_{--} \end{bmatrix} \tag{2.28}$$

For transposed networks, the off-diagonals which describe the mutual coupling are zero. The balanced impedance matrix in sequence domain is then

$$[Z_{[0+1]}] = \begin{bmatrix} Z_{00} & 0 & 0 \\ 0 & Z_{++} & 0 \\ 0 & 0 & Z_{--} \end{bmatrix} \tag{2.29}$$

where:

$$Z_{00} = Z_s + 2Z_m \tag{2.30}$$

$$Z_{++} = Z_{--} = Z_s + Z_m \tag{2.31}$$

Thus in the sequence domain, the three components are decoupled and can be represented with separate line diagrams. This decoupling makes the analysis much simpler, as each circuit can be analyzed separately, and the final phase domain result deter-

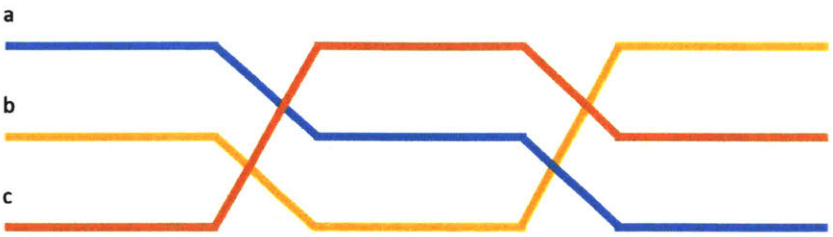


Figure 2-6: Three-phase transposed cables

mined with the inverse Fortescue Transformation.

Transmission Grids

The transmission grid isn't the primary focus of this work, but should be briefly discussed to highlight the challenges in modeling the distribution grid. Since the primary purpose of the transmission system is to transport electricity over large distances, the minimization of losses is a key concern. As such, the cables are designed to have very low impedances resulting in low R/X ratios. The cables are transposed to minimize mutual impedance losses. These design features result in networks with little or no unbalances, and small difference in voltage angles between nodes.

Distribution Grid

The distribution grid handles power delivery to customers, including industrial facilities, large buildings, and small-scale residential units. The distribution grid is broken down into the primary lines and the secondary lines. The high voltage transmission line systems arrive at a substation, which steps down the voltage for the primary feeders of the distribution grid. A single substation is typically serviced by at least two transmission lines, and can feed anywhere between three to 30 primary distribution feeders. The primary feeders are typically still three-phase lines, and deliver power to large industrial loads, large office buildings, and can have DERs such as large PV arrays, wind turbines, and utility-scale storage units connected. In addition to these loads, a primary feeder will also connect to distribution transformers which service the secondary feeders. There can be over 100 secondary feeders for each primary

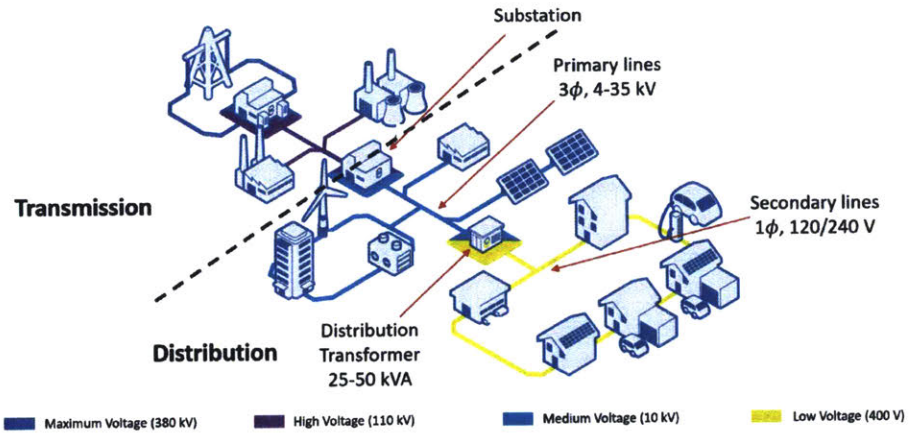


Figure 2-7: Graphic of the interface between the transmission network and distribution network, with details pertaining to the primary and secondary feeders in the distribution network

feeder. These secondary feeders are typically one- or two-phase lines, and consist of very short lines at 120/240 V. The secondary feeders service neighbourhoods and residential units. The DERs in the secondary feeder are typically rooftop solar PV and electric vehicles (EVs). Figure 2-7 shows this topology, and the connection between the transmission and distribution grids.

The primary feeders are classified as medium voltage lines (between 1-35 kV), and are often organized in a mesh topology. The redundancy in meshed networks allows for increased reliability, as alternate paths for electricity flow are available in the event of an outage. Critical loads, such as hospitals, police stations, and fire stations, are typically connected to the primary feeder with backup substations, in case the primary substation is damaged or inoperable. This, along with the meshed topology, ensures critical loads are always serviced. Meshed networks are more expensive to build, however, so most low voltage grids (secondary distribution lines) are radial topologies. This means a single point of failure can take out the downstream branch in the grid, and requires on-site repairs rather than rerouting through electronics or switches. In North America, most distribution grids are radial topology, with the exception of high-density urban areas, which enjoy meshed grids. Many European electricity infrastructure enjoys meshed, looped, or tied-ring network structures, which are more connective graph structures than radial networks. Meshed grids, however, are more

desirable due to increased resiliency and enabled bidirectional power transfer between many nodes.

Since the distribution grid consists of lower voltage and shorter length lines, the electrical impedances are typically higher than that of transmission lines - thus the lines have a higher R/X ratio. Further, the distribution grid lines are not transposed, and secondary feeders have many one- and two-phase lines, which connect residential loads. These all contribute to a highly unbalanced distribution grid. Reactive power support becomes more important in these grids, and are typically implemented as power electronics and capacitor banks. These devices help minimize reactive power in the grid, and improve the efficiency of power transfer. The unbalanced distribution grid cannot be approximated as a balanced system, meaning the coupled three-phase equations must be solved together.

From the above description, lets highlight the key points which motivate the work presented in this thesis. The distribution grid:

1. has both radial and meshed topologies.

2. is highly unbalanced, due to the impedances of the lines, and the presence of single- and two-phase lines, and corresponding loads.

3. has a large number of nodes - think up to 30 primary feeders per substation, 100 secondary feeders per primary feeder, and many 100's of nodes connected to the secondary line. Now imagine that each node has a device - the coordination of all these devices is a large problem which quickly gets out of hand.

2.2 State of the Power System

The section above introduced the fundamentals of electric power and the structure of the power system, which have been present since the conception of a connected power transmission and delivery system. Rather, the new challenges come from the increased penetration of distributed energy resources, such as renewables and batteries.

2.2.1 Challenges with DERs

Within the transmission system, large-scale renewables such as wind and solar farms have been integrated. While these resources are intermittent and uncontrollable, they are large systems which can be operated directly by the load serving entity (LSE). DERs in the distribution grid however, are small resources typically located behind the meter where the LSEs and grid operators have little to no visibility. The penetration of these resources is increasing, including increased rooftop solar PV, storage and batteries (which also include electric vehicles), and onsite nonrenewable generators (such as controllable diesel generators). With more DERs in the distribution grid, which has 100's of nodes on each secondary line, the number of devices which must be controlled is very large. Finally, some residential units or small businesses may have privacy concerns, meaning the LSE cannot (or perhaps should not) have direct access to all information and control over the DERs. Thus the problems of visibility, scale, and privacy are unique to DERs in the distribution grid. This, along with the intermittency and uncontrollability of renewable resources, and the fundamental requirement that supply meets demand in the power grid creates a rich and complex resource coordination problem.

2.2.2 Paradigm Shift: Centralized to Decentralized

Based on the challenges discussed above, much of the leading research and development in the power grid is moving from a centralized perspective to decentralized. Thus far, the centralized perspective has been hugely successful - supply was fully controllable with large scale powerplants and generators producing power in few point locations. The grid operated by requiring that supply follow demand, and customers paid flat rates for their electricity usage. However, with the increased penetration of DERs, this paradigm no longer holds. First, renewables result in variable and uncontrollable supply. Thus enters the concept of demand-side management and demand response, where both supply and demand are variable. This includes both direct control over a load or using transactive energy schemes (such as time-of-use electric-

ity rates or real-time pricing) to influence consumer behaviour. This adds another dimension to the overall problem – where the cost of compensating a customer for changing their consumption as needed by the grid – must now be accounted for in the optimization scheduling problem. With both distributed generation and demand response spread within the distribution grid, resources are no longer large agglomerates at point locations. These distributed resources can all provide important functions to the grid operator; however, the grid operator cannot have direct control or full knowledge of each of these resources as the coordination problem quickly gets out of hand. Rather, these resources must be "smart". When equipped with computational capabilities, these resources can schedule themselves, leveraging local information as well as some shared information from neighbours and/or other nodes in the network. Thus, the physical system and the operations both shift towards decentralized models, or further, to distributed models.

As defined in [23] – an excellent survey paper on distributed optimization and control algorithms applied to electric power systems – the following terminology will be used here:

1. **Centralized:** Each agent communicates with a centralized controller that performs computations and sends new commands;

2. **Hierarchical:** Algorithms where computations are done by agents that communicate with other agents at a higher level in a hierarchical structure, eventually leading to a centralized controller;

3. **Distributed:** Algorithms where each agent communicates with its neighbours, but there is not a centralized controller;

4. **Decentralized:** Purely local algorithms, i.e. no communication between agents.

2.3 Optimization Theory

We now shift the discussion to the mathematical formulation of the resource coordination problem. The concept of optimization is quite simple: find the minimum of a function within a certain area. Consider a 2-dimensional plane. We want to find the minimum value of some curve within, say, a known box. The curve represents a cost function. The box is represented by a set of equations. Extending this visual in n-dimensions, the 'box' is now an n-dimensional surface with 'flat' surfaces; this is called a polytope. The set of equations describing the boundaries are the constraints. The function to be minimized is the objective. From these, an optimization problem is formed, where the objective function is minimized subject to a set of constraints.

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.32}$$

$$\text{s.t.} \quad g(x) = 0 \tag{2.33}$$

$$h(x) \geq 0 \tag{2.34}$$

2.3.1 Duality

In the idea of duality, the constraints in the optimization problem are penalized by way of an indicator function of sorts, to render an unconstrained optimization problem. Each constraint $ax = b$ is penalized as $\lambda(ax - b)$, where λ is the Lagrange multiplier corresponding to the constraint. A simple indicator function is a linear penalty of scalar λ is used, whereby $\lambda \to \infty$ when the constraint is not satisfied. Then for an objective function of $f(x)$ which we are trying to minimize, the Lagrangian is the following: $\mathcal{L}(x, \lambda) = f(x) + \lambda(ax - b)$. Then the set of points which satisfy the feasibility requirements can be found as $J(x) = \max_\lambda \mathcal{L}(x, \lambda)$. However, what we really wanted was to minimize the objective function (and also thereby the Lagrangian), so the optimal solution would be for the following problem: $\min_x \max_\lambda \mathcal{L}(x, \lambda)$. However, the min-max problem is challenging to solve, but can be converted into a max-min problem by simply reversing the order of operations. Then, the resulting maximization problem of $\max_\lambda \min_x \mathcal{L}(x, \lambda) = \max_\lambda g(\lambda)$, where $g(\lambda) = \min_x \mathcal{L}(x, \lambda)$ is the

dual problem. The Lagrange multipliers λ are also called the dual variables, and are determined from the dual problem, as well as the optimality conditions discussed in the next section. The dual problem gives the tightest upper bound for a minimization problem (equivalently tightest lower bound for a maximization problem). For the solution p^* and d^* as the optima for the primal and dual problems respectively, the optimality duality gap is defined as $p^* - d^*$.

2.3.2 Optimality Conditions

Returning to the 2-dimensional case, consider a general curve which has some minimum and maximum points. From basic calculus principles, we know the slope of the line describes the rate of change of the function, described by the first derivative of the function, $\frac{dy}{dx}$. We know the derivative is zero when at an extreme point - this includes both minima, maxima, and saddle points. To differentiate between these points, we consider the curvature of the line, described by the second derivate of the function, $\frac{d^2y}{dx^2}$. The simplest case is the saddle point, where the curvature of the line does not change, thus $\frac{d^2y}{dx^2} = 0$. For a minimum point, the curvature is positive, $\frac{d^2y}{dx^2} > 0$, and for a maximum point, the curvature is negative, $\frac{d^2y}{dx^2} < 0$. Extending to n-dimensions, the first and second derivates are the gradient and hessian of the function, respectively, and are defined as below:

Gradient

$$f(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \dots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Hessian

$$f(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$$

$$\nabla^2 f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Further, for a general curve, we also consider local and global extreme points. A local minimum, x^*, is defined as the minimum of the function within a ball of some finite radius r: $f(x^*) \leq f(x) \quad \forall x \in B_{r>0}(x^*)$. A global minimum extends this definition to a ball of infinite radius, whereby $f(x^*) <= f(x) \quad \forall x \in \mathbb{R}^n$. We

will return to the notions of global and local optimality in the section on Convex Optimization.

For an unconstrained problem, the minimum is described as:

$$\underline{\text{Necessary Condition}}: \nabla f(x) = 0, \qquad \nabla^2 f(x) \succeq 0$$

$$\underline{\text{Sufficient Condition}}: \nabla f(x) = 0, \qquad \nabla^2 f(x) \succ 0$$

For a constrained problem as in (2.32), the minimum points obtained by the relations above may no long be valid within the feasible polytope. As such, new relations are needed to determine optimality of a point. These are known as the Karush-Kuhn-Tucker (KKT) optimality conditions, and are shown below in (2.35). Here we take a geometric approach to reaching the KKT conditions, and introducing the Lagrangian function (which is defined in (2.35a)). In order to do so, we need some additional terminology and concepts around level sets, and make use of the gradient vector.

$$L(x, \mu, \lambda) = f(x) - \sum_{i=1}^{n_g} \mu_i g_i(x) - \sum_{i=1}^{n_h} \lambda_i h_i(x) \tag{2.35a}$$

$$\nabla_x L(x, \mu, \lambda) = 0 \tag{2.35b}$$

$$\nabla_{\mu_i} L(x, \mu) = 0 \longrightarrow -g_i(x) = 0, \quad i = 1, ..., n_g \tag{2.35c}$$

$$h_i(x) \geq 0, \quad i = 1, ..., n_h \tag{2.35d}$$

$$\lambda_i \geq 0 \tag{2.35e}$$

$$\lambda_i h_i(x) = 0, \quad i = 1, ..., n_h \tag{2.35f}$$

Level Sets

A level set, or contour of a function, is a set describing all points of function $f(x)$ where the function takes on a constant value c. Similarly a sub-level and supper-level sets describe the inequality conditions, where the function takes on values less than

or greater than the constant c:

$$\text{Level set} : \{x \in \mathbb{R}^n : f(x) = c\} \tag{2.36}$$

$$\text{Sub-level set} : \{x \in \mathbb{R}^n : f(x) \leq c\} \tag{2.37}$$

$$\text{Supper-level set} : \{x \in \mathbb{R}^n : f(x) \geq c\} \tag{2.38}$$

The gradient vector for any point on the level surface is perpendicular to the surface. This falls simply from multivariable calculus, by looking at the dot product of the gradient vector evaluated at a point on the surface, and a curve on the level surface. Further, the gradient vector points in the direction of steepest ascent; thus starting at point P, the direction of $\nabla f(P)$ indicates which direction to follow to increase $f(x)$.

Consider two functions, $f(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$ and $g(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$, with level sets $\{x \in \mathbb{R}^n : f(x) = c_1\}$ and $\{x \in \mathbb{R}^n : g(x) = c_2\}$. Consider the point where these two level sets intersect, in particular, where they are tangent to each other, at point x^*. At this point, the gradient vectors for the two functions (which are tangent to the level sets) are perpendicular to each other. Thus, they can be related as $\nabla f(x^*) = \mu \nabla g(x^*)$, where μ is a nonzero constant.

Optimality: Equality Constraints

Consider the optimization problem with equality constraints:

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.39}$$

$$\text{s.t.} \quad g(x) = 0 \tag{2.40}$$

The equality constraints represent level sets. Points where these level sets are tangent to the level sets of the function $f(x)$ represent the minimum and maximum points. For any other point where the constraint level set intersects with the level sets of $f(x)$, the negative gradient of $f(x)$, $-\nabla f(x)$ points in the direction of steepest descent, which when followed, will eventually lead to another intersection point with the constraint

level sets. Thus, the tangency points indicate the maximum and minimum points. From this, the *necessary* conditions for optimality can be written out, as in (2.41). Note that these are no sufficient conditions, as these will give both minimum and maximum points of the function.

$$L(x, \mu) = f(x) - \mu g(x) \tag{2.41a}$$

$$\nabla_x L(x, \mu) = 0 \longrightarrow \nabla f(x) = \mu \nabla g(x) \tag{2.41b}$$

$$\nabla_\mu L(x, \mu) = 0 \longrightarrow -g(x) = 0 \tag{2.41c}$$

Optimality: Inequality Constraints

Consider the optimization problem with inequality constraints:

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.42}$$

$$\text{s.t.} \quad h(x) \geq 0 \tag{2.43}$$

The equality constraints represent supper-level sets. In this problem, the unconstrained optimal point can exist outside (or on the boundary) of the the feasible region, or lies within the feasible region. These two cases are similar to the equality constrained problem and the unconstrained problem respectively. Considering Case A, where the unconstrained optimal point lies outside of the feasible set, the extreme point must lie on the boundary of the supper-level set. Thus, the extreme point occurs at the tangent point between the two level sets, $\{x \in \mathbb{R}^n : f(x) = c_1\}$ and $\{x \in \mathbb{R}^n : h(x) = c_2, c_2 \geq 0\}$. More specifically, $\nabla f(x)$ points in the same direction as $\nabla g(x)$ as the steepest ascent directions will be the same - the unconstrained minima lies outside of the feasible region, so the function is increasing in the same space where function $g(x)$ is increasing. In this case, the optimality conditions are the same as with the equality constraints, with an additional constraint requiring $\lambda \geq 0$, to describe that both gradients point in the same direction. Thus for Case A, the

necessary conditions for optimality are described by (2.44).

$$L(x, \lambda) = f(x) - \lambda h(x) \tag{2.44a}$$

$$\nabla_x L(x, \lambda) = 0 \longrightarrow \nabla f(x) = \lambda \nabla h(x) \tag{2.44b}$$

$$\nabla_\lambda L(x, \lambda) = 0 \longrightarrow h(x) = 0 \tag{2.44c}$$

$$\lambda \geq 0 \tag{2.44d}$$

Considering Case B, the unconstrained minimum lies within the feasible region - the unconstrained minimum is the same as the constrained minimum. We can then treat the problem as an unconstrained optimization problem, where we simply require the first order condition that the gradient is zero. Recalling the level sets of the functions, we then have no requirement on the two gradients being parallel. Thus for Case B, the necessary conditions for optimality are described by (2.45).

$$L(x, \lambda) = f(x) - \lambda h(x) \tag{2.45a}$$

$$\nabla_x L(x, \lambda) = 0 \longrightarrow \nabla f(x) = \lambda \nabla h(x) = 0 \tag{2.45b}$$

$$\nabla_\lambda L(x, \lambda) > 0 \longrightarrow h(x) > 0 \tag{2.45c}$$

$$\lambda = 0 \tag{2.45d}$$

Then combining the two cases, the necessary conditions for optimality in an inequality constrained problem are described by (2.46).

$$L(x, \lambda) = f(x) - \lambda h(x) \tag{2.46a}$$

$$\nabla_x L(x, \lambda) = 0 \longrightarrow \nabla f(x) = \lambda \nabla h(x) \tag{2.46b}$$

$$\nabla_\lambda L(x, \lambda) \geq 0 \longrightarrow h(x) \geq 0 \tag{2.46c}$$

$$\lambda \geq 0 \tag{2.46d}$$

$$\lambda h(x) = 0 \tag{2.46e}$$

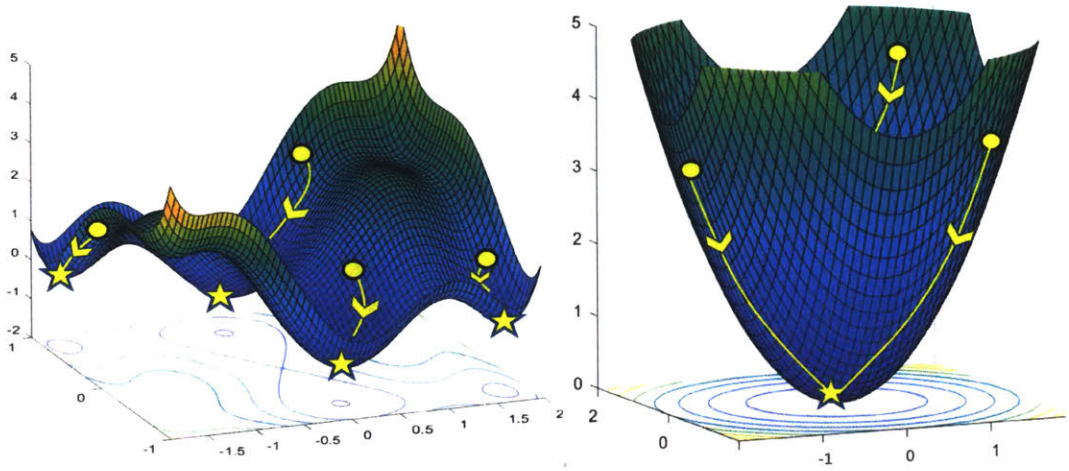


Figure 2-8: Image of nonconvex and convex surfaces, left and right respectively. Local/global minima are indicated with yellow stars, and the initial started point for the algorithm is indicated with yellow dots.

2.3.3 Convex Optimization

Convex optimization is a subset of nonlinear optimization, where the objective function is a convex function and the feasible set is a convex set. In convex optimization, there is a unique minimum, meaning there is only one minimum point which is both local and global. This is unlike most nonlinear problems, where the function has many local minima, and the point reached through and optimization algorithm is sensitive to the initial condition. This is shown in Fig. 2-8, where the function on the right has a global minimum which is guaranteed to be reached regardless of the starting point.

Convex Sets and Functions

A convex set is a set in which the line connecting any two given points within the set, is entirely within the set. Consider a convex set $\mathcal{C}$, where $\forall x_1, x_2 \in \mathcal{C}$, the linear combination of these two points also lies within the set: $\theta x_1 + (1-\theta)x_2 \in \mathcal{C}, \theta \in [0,1]$, as shown in Fig. 2-9.

A convex function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex if the domain of f is a convex set and the line connecting two points on the function lies above the function: $f(\theta x_1 + (1-\theta)x_2) \leq \theta f(x_1) + (1-\theta) f(x_2), \forall x_1, x_2 \in \mathbf{dom} f, \theta \in [0,1]$. Note that the sum of two convex

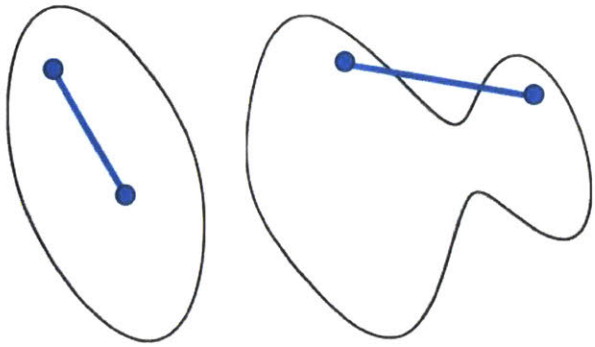


Figure 2-9: Image of convex (left) and nonconvex (right) sets.

functions is also convex, and the negative of a convex function is a concave function.

Common convex functions include:

1. Affine: $a^T x + b$ for $a, b \in \mathbb{R}^n$
2. Powers: x^α for $\alpha \geq 1$ or $\alpha \leq 0$ on $\mathbb{R}_{++}$
3. Exponentials: e^{ax} for $a \in \mathbb{R}$
4. Absolute value powers: $|x|^p$ for $p \geq 1$
5. Logarithms of form $x \log x$ on $\mathbb{R}_{++}$
6. Norms: $||x||_p$ for $p \geq 1$

Chapter 3

Optimal Power Flow for Meshed and Unbalanced Grids: Solutions with Distributed Optimization

The optimal power flow problem looks to find the operating levels for energy resources so that supply meets demand, while minimizing some cost. Typically this problem is solved by a central agent, such as a grid operator, which looks to minimize operating costs of electric power plants and transmission costs, while ensuring that supply equals demand. Note however, that the objective function need not be operating costs - the objective could be to maintain voltages around desired setpoints (Volt-VAr control), to minimize thermal losses in the network, or to minimize deviation from a predetermined schedule.

The full AC formulation of the OPF problem is very challenging to solve. The power flow equations are nonlinear and nonconvex functions which model the flow of electricity throughout the network. Further, security constrained unit commitment (SCUC) problems also use integer variables to help decide which generators will participate in electricity production over a period of time. This introduces additional integer constraints, resulting in a mixed-integer nonlinear program (MINLP) which are even more challenging to solve. To further complicate matters, real operation requires the OPF problem to be solved for the full distribution system every 5 minutes,

to ensure demand is met. Because of these challenges, many different approaches to solving OPF is taken. Many approaches look to linearize the power flow equations by making small angle assumptions - this is referred to as DC OPF. However, this assumption breaks down in the distribution grid where voltage angles are not close to one another. Other approaches use heuristics to solve the OPF problem, and others use assumptions which may result in results which are not physically implementable.

This problem is exacerbated in the distribution grid where the physics of the lines prevent the DC approximations from being made. Further, there is an increasing penetration of DERs including solar PV, storage devices such as batteries, and demand response, which can all be utilized to provide services to the grid; however, the challenge of solving the OPF simply grows as the number of participating devices grows. A more detailed model is required to appropriately utilize these resources, but makes the problem computationally intractable, especially for solving every 5 minutes.

This suggests the need for distributed computation, where multiple computing agents can be utilized to solve the OPF problem. This is discussed briefly in the next section. We also need accurate models of the power physics of the distribution grid, which retain the physical constraints of the system while enabling better computation and guaranteed global optimality. A convex formulation which is well conditioned is desired. These are the focus of this chapter.

3.1 Notation for Power System Model

In this section we introduce the common formulations for power flow in AC grids. We begin with the nonlinear branch flow (BF) model, then discuss the proposed current injection (CI) model. We consider distribution grids with radial or meshed topologies that can be represented as a directed graph $\mathbf{\Gamma}_{\mathrm{D}} = \langle \mathcal{B}, \mathcal{T}_{\mathrm{D}} \rangle$, where $\mathcal{B}$ represents the graph vertices and $\mathcal{T}_{\mathrm{D}}$ the graph directed edges denoted as $(\mathbf{i}, \mathbf{j})$ where $\mathbf{i} < \mathbf{j}$. For a node $\mathbf{j}$, the set of parent node(s) are $\mathbf{i} \in \{\mathbf{i_j}\} \forall (\mathbf{i}, \mathbf{j}) \in \mathcal{T}_{\mathrm{D}}$, and the set of child nodes are $\mathbf{k} \in \{k_j\} \forall (\mathbf{j}, \mathbf{k}) \in \mathcal{T}_{\mathrm{D}}$.

Consider Fig. 3-1 which depicts a simple 4-bus radial network. We will model both

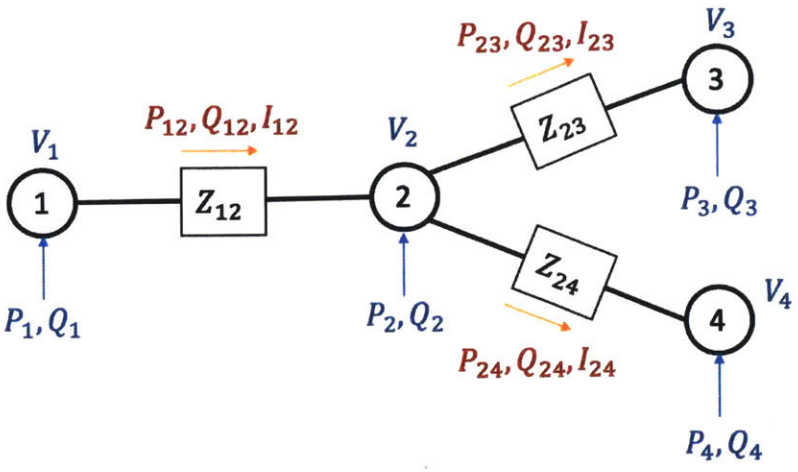


Figure 3-1: Example network

nodal and line power flow quantities, which are shown in blue and red respectively. The nodal quantities are the real and reactive power injection (P_j, Q_j) and the voltage at the node (V_j), where we denote $v_j = |V_j|^2$. The line quantities for a line $(\mathbf{j}, \mathbf{k}) \in \mathcal{T}_D$ are the power flows through the line (P_{jk}, Q_{jk}), and the current through the line (I_{jk}), where we denote $l_{jk} = |I_{jk}|^2$. These quantities will be used in the models as we move forward. Any additional terms will be defined as they arise.

3.2 Branch Flow Model

The branch flow model applies to primarily radial balanced networks, using branch variables derived from Kirchhoff's laws to describe the power flow relationships. The model leverages two relaxations to arrive at a convex formulation which can be readily solved by commercial optimization solvers [5, 6]. The angle relaxation applies to Ohm's law $V_j - V_i = Z_{ij} I_{ij}$, and assumes small angles and balanced flows to yield (3.1). In this step we also change our variable space to use v_j, l_{ij} (the squared magnitude of the voltage and current) instead of V_j, I_{ij}. The second relaxation is to convexify (3.4) in the form of a second-order cone (SOC) constraint, which yields (3.5). Then (3.1)-(3.3) and (3.5) define the convex power flow relationships known as the BF model,

which can now be used within an OPF problem.

$$v_j - v_i = (R_{ij}^2 + X_{ij}^2)l_{ij} - 2(R_{ij}P_{ij} + X_{ij}Q_{ij}) \tag{3.1}$$

$$P_{ij} = R_{ij}l_{ij} - P_j + \sum_{k\in\{k_j\}} P_{jk} \tag{3.2}$$

$$Q_{ij} = X_{ij}l_{ij} - Q_j + \sum_{k\in\{k_j\}} Q_{jk} \tag{3.3}$$

$$P_{ij}^2 + Q_{ij}^2 = v_i l_{ij} \tag{3.4}$$

$$P_{ij}^2 + Q_{ij}^2 \leq v_i l_{ij} \tag{3.5}$$

3.2.1 Limitations of Branch Flow model

The main limitations of the branch flow model are the requirements on the topology and structure of the grid. First, the grid must have a radial topology. There are proposed convexifications for meshed networks under the BF model, but these complicate the formulation and increase the size of the problem, by adding additional variables. Second, the grid must be balanced. There are proposed techniques to use BF for unbalanced networks, but they apply only to networks with limited degree of unbalances. In [24] and [25] the authors model an unbalanced distribution network and modify the BF model by assuming that the reciprocal unbalance of the phases are small. This implies that the relative difference between the angles of each phase are ignored, which is not a valid assumption in most distribution grids where the system is highly unbalanced, and many single- and two-phase lines and/or loads exist.

Further, the BF model restricts the total amount of power injections to ensure an exact convex relaxation. In other words, there is a limit on how active a network can be, and the total DG and storage penetration. Note however, that this is a mild condition for most networks, and can be determined a priori, with knowledge of the network and operating limits. This is condition C1 and Theorem 4 in [7]. The condition C1 depends only on the line parameters (R_{ij}, X_{ij}), the upper power injection bounds $(\overline{P}_j, \overline{Q}_j)$, and the lower voltage bound $(\underline{v}_{ij})$. The physical interpretation of this constraint is to limit the reverse power flows in the network: if power loss across a

line is reduced, upstream reverse power flows will increase. The power loss on a line is reduced when the current is reduced, to the extreme where power flow reverses on the line. Note however that from case studies in [7], C1 is shown to be a mild condition, even for networks with high penetration of DGs. Further details and conditions under which C1 is met can be found in [7].

3.3 Current Injection Model

To overcome the limitations of the BF model for distribution grid, a new convex model is proposed. The Current Injection (CI) model which leverages McCormick envelopes (MCE) [26] as the convex relaxation for the bilinear form of the power terms - recall power is a bilinear function of the current and voltages.

Consider the example network shown in Fig. 3-2, consisting of N nodes. We consider a segment of nodes $\mathbf{i}, \mathbf{j}$ and $\mathbf{k}$, which is radial and single-phase for simplicity [1]. By applying Kirchhoff's Current Law (KCL) and Ohm's Law at node $\mathbf{j}$, we obtain (3.6) and (3.7)-(3.8) respectively.

$$I_{\mathbf{j}} + I_{\mathbf{ij}} - I_{\mathbf{jk}} = 0 \tag{3.6}$$

$$I_{\mathbf{ij}} = \frac{V_{\mathbf{i}} - V_{\mathbf{j}}}{Z_{\mathbf{ij}}} = Y_{\mathbf{ij}}(V_{\mathbf{i}} - V_{\mathbf{j}}) \tag{3.7}$$

$$I_{\mathbf{jk}} = \frac{V_{\mathbf{j}} - V_{\mathbf{k}}}{Z_{\mathbf{jk}}} = Y_{\mathbf{jk}}(V_{\mathbf{j}} - V_{\mathbf{k}}) \tag{3.8}$$

where $Z_{\mathbf{ij}} = R_{\mathbf{ij}} + jX_{\mathbf{ij}}$ is the impendance and $Y_{\mathbf{ij}} = \frac{1}{Z_{\mathbf{ij}}}$ is the admittance between nodes $\mathbf{i}$ and $\mathbf{j}$ [2]. Note that all variables here can be complex numbers (namely $X \neq 0$).

[1] Note that the CI model does not require radial and single-phase balanced networks. Rather, the benefit of the CI model is its applicability to networks of any topology and to unbalanced multi-phase networks as found in distribution grids.

[2] The imaginary number 'j' is not to be confused with the subscript $\mathbf{j}$ indicating the node. Note also that the convention for using 'j' to denote the imaginary part of the complex number as done in electrical engineering is used here, as opposed to the mathematical convention of using 'i'.

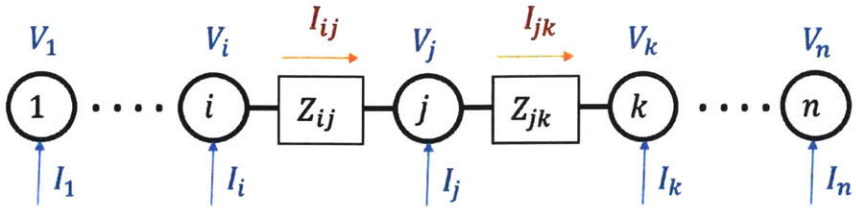


Figure 3-2: Example network

The nodal current injections and voltages can then be related, as follows:

$$I_{\mathbf{j}} = I_{\mathbf{jk}} - I_{\mathbf{ij}} \tag{3.9}$$

$$= Y_{\mathbf{jk}}(V_{\mathbf{j}} - V_{\mathbf{k}}) - Y_{\mathbf{ij}}(V_{\mathbf{i}} - V_{\mathbf{j}}) \tag{3.10}$$

$$= V_{\mathbf{j}}(Y_{\mathbf{jk}} + Y_{\mathbf{ij}}) - V_{\mathbf{i}}Y_{\mathbf{ij}} - V_{\mathbf{k}}Y_{\mathbf{jk}} \tag{3.11}$$

This equation can be rewritten in matrix form as:

$$I_{\mathbf{j}} = \begin{bmatrix} -Y_{\mathbf{ij}} & Y_{\mathbf{ij}} + Y_{\mathbf{jk}} & -Y_{\mathbf{jk}} \end{bmatrix} \begin{bmatrix} V_{\mathbf{i}} \\ V_{\mathbf{j}} \\ V_{\mathbf{k}} \end{bmatrix} \tag{3.12}$$

Similar equations can be written for every node in the network, with the nodal current injections and voltages being related by the network admittance bus matrix, $\mathbf{Y}$, which is constructed as:

$$Y_{\mathbf{ij}} = \begin{cases} \sum_{\mathbf{k}=1,..,N} Y_{\mathbf{ik}} & \text{if } \mathbf{i} = \mathbf{j} \\ -Y_{\mathbf{ij}} & \text{if } \mathbf{i} \neq \mathbf{j} \end{cases} \tag{3.13}$$

where $Y_{\mathbf{ik}} \neq 0$ only where a line exists between nodes $\mathbf{i}$ and $\mathbf{k}$. Then, the relationship is $I = \mathbf{Y}V$. This is the typical model used in power flow analysis.

We can also calculate the power injection at the nodes as below, noting that the complex power (apparent power) is composed of the real and reactive power

components:

$$S_j = V_j I_j^* = (V_j^r I_j^r + V_j^i I_j^i) + j(-V_j^r I_j^i + V_j^i I_j^r) \tag{3.14}$$

$$P_j = (V_j^r I_j^r + V_j^i I_j^i) \tag{3.15}$$

$$Q_j = (-V_j^r I_j^i + V_j^i I_j^r) \tag{3.16}$$

where the superscripts r and i denote the real and imaginary components, respectively; the rectangular form here makes it very easy to manipulate the terms.

We can easily make the extension to three-phase systems. The fundamental equations stay the same, but now the nodal and line variables can be extended to 3-element vectors, and the admittances Y_{ij} now become matrices $\mathbf{Y_{ij}} \in \mathbb{R}^{3\times3}$. The diagonal elements of $\mathbf{Y_{ij}}$ represent the resistance and inductance of the line (as before), while the off-diagonal terms represent the coupling between lines resulting from the electromagnetic interactions between the lines which are in close proximity to one another. Further, the admittance line matrices are symmetric, so knowing the upper triangular (or equivalently lower triangular) components fully characterize the matrix. The network admittance bus matrix is then $\mathbf{Y} \in \mathbb{R}^{3N\times3N}$ where N is the number of nodes in the network.

Moving forward, we generally assume that all the vectors are column vectors composed of the real and reactive components of all phases. When a specific line does not exist (for example a 2-phase line) the variable will simply be 0.

Statement of the CI Model

Using the above, we can formulate the OPF problem as in (3.17)-(3.26).

$$
\begin{aligned}
\min_{\mathbf{y}} \quad & f(\mathbf{y}) \\
\text{s.t.} \quad & I = \mathbf{Y}V && && (3.17)\\
& I_{\mathbf{j}} = I^r_{\mathbf{j}} + jI^i_{\mathbf{j}} && \forall \mathbf{j} \in [N] && (3.18)\\
& V_{\mathbf{j}} = V^r_{\mathbf{j}} + jV^i_{\mathbf{j}} && \forall \mathbf{j} \in [N] && (3.19)\\
& P_{\mathbf{j}} = V^r_{\mathbf{j}} I^r_j + V^i_{\mathbf{j}} I^i_{\mathbf{j}} && \forall \mathbf{j} \in [N] && (3.20)\\
& Q_{\mathbf{j}} = -V^r_{\mathbf{j}} I^i_j + V^i_{\mathbf{j}} I^r_{\mathbf{j}} && \forall \mathbf{j} \in [N] && (3.21)\\
& \underline{V^r_{\mathbf{j}}} \le V^r_{\mathbf{j}} \le \overline{V^r_{\mathbf{j}}} && \forall \mathbf{j} \in [N] && (3.22)\\
& \underline{V^i_{\mathbf{j}}} \le V^i_{\mathbf{j}} \le \overline{V^i_{\mathbf{j}}} && \forall \mathbf{j} \in [N], \mathbf{j} \neq \mathbf{j}^{\#} && (3.23)\\
& \underline{P_{\mathbf{j}}} \le P_{\mathbf{j}} \le \overline{P_{\mathbf{j}}} && \forall \mathbf{j} \in [N] && (3.24)\\
& \underline{Q_{\mathbf{j}}} \le Q_{\mathbf{j}} \le \overline{Q_{\mathbf{j}}} && \forall \mathbf{j} \in [N] && (3.25)\\
& V^r_{\mathbf{j}^{\#}} = 1, \quad V^i_{\mathbf{j}^{\#}} = 0 && && (3.26)
\end{aligned}
$$

where $I_{\mathbf{j}} = I^r_{\mathbf{j}} + jI^i_{\mathbf{j}}$, $V_{\mathbf{j}} = V^r_{\mathbf{j}} + jV^i_{\mathbf{j}}$, and I, V, P, Q are column vectors of all the nodal currents, voltages, real power, and reactive power injections respectively. Specifically, (3.17)-(3.21) describe the power physics, and (3.22)-(3.25) describe the limits on nodal voltages, and real and reactive power injections. Finally, (3.26) describes the slack bus, denoted as node $\mathbf{j}^{\#}$, which is the point of common coupling (PCC) of the distribution grid to the transmission grid (i.e. the substation). Note that a load will have $\overline{P_{\mathbf{j}}} = \overline{Q_{\mathbf{j}}} = 0$, while a generator will have $\underline{P_{\mathbf{j}}} = 0$, as the convention for power/current injection being positive is used.

Relaxed CI Model

Note that until now, there have been no approximations or assumptions made. We've been using the definitions and power relationships to describe the power flow in the network. While the OPF problem in (3.17)-(3.26) fully describes the network, the

constraints in (3.20)-(3.21) render the problem nonconvex. This is challenging from an optimization point of view, there is no guarantee for global optimality. Different approaches can be taken to address this, including linearized approaches, approximations, and convex relaxations. Here we use the latter. The bilinear terms in the constraints can be relaxed using McCormick Envelopes (MCE) [27], which provides a lower bound on the original nonconvex problem. Tighter MCE bounds will produce tighter lower bounds (thus closest to optimal), and will also decrease computation time as the solution space is restricted. For a general bilinear form $w = xy$ defined over a set $\boldsymbol{S} \subset \mathbb{R}^3 = \{x, y : x \in [\underline{x}, \overline{x}], y \in [\underline{y}, \overline{y}]\}$, the MCE convex relaxation is as follows:

$$w \geq \underline{x}y + x\underline{y} - \underline{x}\underline{y} \tag{3.27}$$

$$w \geq \overline{x}y + x\overline{y} - \overline{x}\overline{y} \tag{3.28}$$

$$w \leq \underline{x}y + x\overline{y} - \underline{x}\overline{y} \tag{3.29}$$

$$w \leq \overline{x}y + x\underline{y} - \overline{x}\underline{y} \tag{3.30}$$

The (3.20)-(3.21) consist of four bilinear terms, requiring the introduction of four new variables per node per phase, a, b, c, d which will then replace each term. This relaxation also requires the introduction of current bounds. The calculation of the tightest current bound is outside the scope of this thesis, but a possible technique to recover current limits is presented in the following section.

The relaxed CI model is then formulated as a relaxed OPF as in the problem

formed with constraints (3.31)-(3.36).

$$\begin{aligned}
\min_{\mathbf{y}} \quad & f(\mathbf{y}) \\
\text{s.t.} \quad & \text{(3.17), (3.22)-(3.26)} && && (3.31)\\
& P_{\mathbf{j}} = a_{\mathbf{j}} + b_{\mathbf{j}} && \forall \mathbf{j} \in [N] && (3.32)\\
& Q_{\mathbf{j}} = -c_{\mathbf{j}} + d_{\mathbf{j}} && \forall \mathbf{j} \in [N] && (3.33)\\
& \underline{I^r_{\mathbf{j}}} \leq I^r_{\mathbf{j}} \leq \overline{I^r_{\mathbf{j}}} && \forall \mathbf{j} \in [N] && (3.34)\\
& \underline{I^i_{\mathbf{j}}} \leq I^i_{\mathbf{j}} \leq \overline{I^i_{\mathbf{j}}} && \forall \mathbf{j} \in [N] && (3.35)\\
& \text{MCE bounds for } a_{\mathbf{j}}, b_{\mathbf{j}}, c_{\mathbf{j}}, d_{\mathbf{j}} \text{ of form (3.27)-(3.30)} && \forall \mathbf{j} \in [N] && (3.36)
\end{aligned}$$

3.3.1 Current Limits

In order to apply the MCE convexification to the bilinear terms of the power flow model, we require bounds on the current injection at the nodes. A simple set of nonlinear optimization problems can be setup to determine upper and lower bounds for the real and imaginary components of the current injection. For brevity, the optimization problem is formulated for $\overline{I^r_{\mathbf{j}}}$, for a single phase. The same problem must be solved for $\underline{I^r_{\mathbf{j}}}, \overline{I^i_{\mathbf{j}}}, \underline{I^r_{\mathbf{j}}}$, for each phase.

$$\begin{aligned}
\max_{I^r_{\mathbf{j}}} \quad & I^r_{\mathbf{j}} \\
\text{s.t.} \quad & P_{\mathbf{j}} = V^r_{\mathbf{j}} I^r_j + V^i_{\mathbf{j}} I^i_{\mathbf{j}} && (3.37)\\
& Q_{\mathbf{j}} = -V^r_{\mathbf{j}} I^i_j + V^i_{\mathbf{j}} I^r_{\mathbf{j}} && (3.38)\\
& \underline{V^r_{\mathbf{j}}} \leq V^r_{\mathbf{j}} \leq \overline{V^r_{\mathbf{j}}} && (3.39)\\
& \underline{V^i_{\mathbf{j}}} \leq V^i_{\mathbf{j}} \leq \overline{V^i_{\mathbf{j}}} && (3.40)\\
& \underline{P_{\mathbf{j}}} \leq P_{\mathbf{j}} \leq \overline{P_{\mathbf{j}}} && (3.41)\\
& \underline{Q_{\mathbf{j}}} \leq Q_{\mathbf{j}} \leq \overline{Q_{\mathbf{j}}} && (3.42)
\end{aligned}$$

For a general 3-phase network with N nodes, this small nonlinear problem must

be solved $3 \times N \times 4$ times, for the 4 current bounds per node per phase. While each nonlinear problem is very small and can be solved quite efficiently with a solver such as Ipopt [28], the processing for a small network of 100 nodes required 1200 problems to be solved. This is a large computational burden, and can be time-consuming, especially in real-time applications where power injection forecasts are regularly updated. Further analysis of these problems and development of heuristic methods to improve current bounds are outside the scope of this thesis. More details can be found in [29].

3.4 Distributed Optimization

As discussed in Chapter 2, the resource coordination and scheduling problem for the distribution grid under high DER penetration is a challenging problem. We can however, leverage distributed computational devices, which may be located at every DER location or shared amongst some smaller DERs. In order to do so, a distributed algorithm is needed to ensure global optimality is reached while using only local information and limited information exchange between computational agents in the network.

This section provides a brief introduction to the distributed optimization algorithm used in this thesis. It is based on proximal atomic coordination (PAC). For more detailed information of the construction of the algorithm and proof of convergence, please refer to [20, 21].

Distributed algorithms take a global (or centralized) problem as in (3.43)-(3.45), and employ different techniques to break this into smaller problems. In the global problem, $\mathbf{y}$ denotes the global decision vector, $\mathbf{y}_j$ denotes the components of the decision vector owned by agent j, and matrices $\mathbf{G}$ and $\mathbf{H}$ correspond to the global equality and inequality constraint matrices respectively. Smaller optimization problems are typically easier to solve than those with more variables and constraints, especially

when dealing with nonlinear problems and/or integer constraints.

$$\min_{\mathbf{y}} \quad \sum_{j} f(\mathbf{y}_j) \tag{3.43}$$

$$\text{s.t.} \quad \mathbf{G}\mathbf{y} = 0 \tag{3.44}$$

$$\mathbf{H}\mathbf{y} \geq 0 \tag{3.45}$$

Often, these smaller problems can be solved concurrently, thus reducing computational time. When designed well, distributed algorithms can reduce communication requirements as all agents are not required to communicate with a single central entity. They can also be robust to various modes of failure as all computations are not dependent on a single point. Finally, if designed well, they can also be privacy preserving; agents do not have to disclose their cost functions, operating constraints, or sensitivity to setpoints with other agents. These motivate the use of distributed algorithms for a wide range of applications, especially for machine learning (fast computation), Internet of Things (IoT) applications (multiple coordinated agents), and market applications (privacy is fundamental).

3.4.1 Atomic Decomposition

Some distributed algorithms require the global problem to be fully decomposable, in that the objective function and constraints of each agent depend only on those variables owned by the agent. This can limit the applicability of the algorithm on different problems, or limits the number of agents which can be utilized. In comparison, the PAC algorithm does not have this requirement. Instead, coupling between agents is handled by agent j creating an estimate of the variables owned by agent i, which agent j also depends on; these estimates are termed as "copies". With the introduction of these new variables, we also required new constraints to ensure that the copies converge to the actual variable value. The inter-agent dependencies are depicted in Fig. 3-3. The matrix graph on the left shows each agent numbered 1 thru 4, with the coloured blocks indicating a dependence on that variable; specifically, the

black blocks indicate owned variables, while the grey blocks indicated inter-agent dependencies. We can then distribute the problem into "atoms", noting that PAC does not require a one-to-one mapping of agents to atoms. Here, we distribute amongst three atoms labelled A, B, and C, with colour coordination showing ownership of variables in blue, orange, and purple respectively. Finally, the cross-colours indicate a variable copy, which is required because of the inter-atomic dependency.

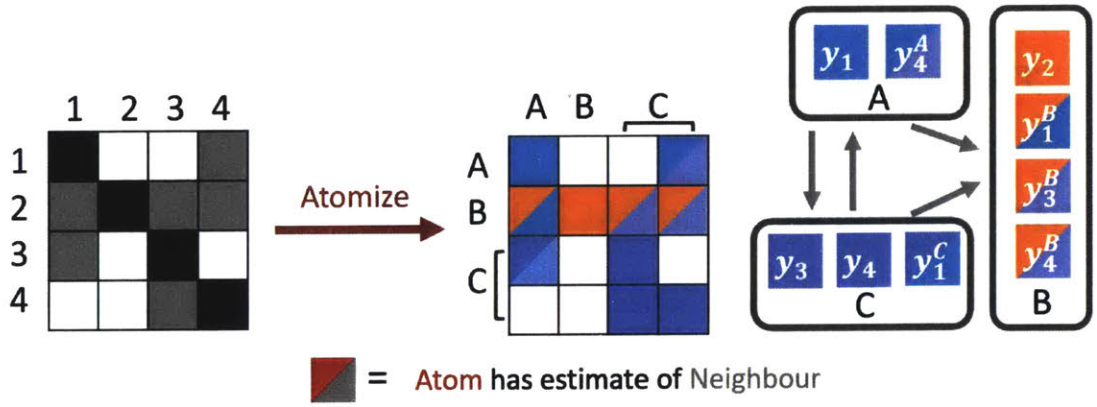


Figure 3-3: Pictorial description of the atomization, and the use of variable copies in PAC. The diagram also stresses that a one-to-one mapping of agents to atoms is not required.

The complete atomic optimization problem can be written as in (3.46)-(3.49). In this formulation, $\mathbf{a}_j$ denotes the atomic decision vector for agent j, and is composed of both the variables owned by atom j and those copied to satisfy the dependencies on other atoms. The vector $\mathbf{a}$ denotes the vertical concatenation of all $\mathbf{a_j}$ vectors. Matrices $\tilde{\mathbf{G}}_\mathbf{j}$ and $\mathbf{H_j}$ correspond to the submatrices of $\mathbf{G}$ and $\mathbf{H}$ which correspond to the rows and columns pertinent to atom j. The global distributed equality matrix $\tilde{\mathbf{G}}$ is a block diagonal matrix composed of the individual $\tilde{\mathbf{G}}_\mathbf{j}$ matrices. Finally, the constraint in (3.49) describes the new coordination constraints added to ensure for a variable x owned by atom i and copied by atom j, that $x_i^j - x_i = 0$ where the subscript denotes the owner of the variable, and the superscript (where present) denotes the atom who is copying the variable. Matrix $\mathbf{B}$ is the complete coordination matrix, which takes on the form of an adjacency matrix, while the notation $\mathbf{B_{j,-}}$ denotes the

rows belonging to the coupling variables for atom j.

$$\min_{\mathbf{a}} \quad \sum_j f_j(\mathbf{a}_j) \tag{3.46}$$

$$\text{s.t.} \quad \tilde{\mathbf{G}}_\mathbf{j}\mathbf{a}_\mathbf{j} = 0 \quad \forall j \tag{3.47}$$

$$\mathbf{H}_\mathbf{j}\mathbf{a}_\mathbf{j} \geq 0 \quad \forall j \tag{3.48}$$

$$\mathbf{B}_{\mathbf{j},-}\mathbf{a}_\mathbf{j} \geq 0 \quad \forall j \tag{3.49}$$

The complete decomposition profile can be found in [20, 21].

3.4.2 Statement of the PAC Algorithm

The Proximal Atomic Coordination (PAC) algorithm is a variant of the prox-linear algorithm, and adopts ideas for distributed optimization from distributed Alternating Method of Multipliers (dADMM), namely the addition of the coordination problem. PAC enjoys linear convergence, with a rate of $o(1/\tau)$ [20, 21]. See Appendix A.1 for a summary of the design of the algorithm. Detailed analysis is in [20, 21].

The algorithm is stated as below:

$$\mathbf{a}_\mathbf{j}[\tau+1] = \mathbf{argmin}_{\mathbf{a}_\mathbf{j}} \left\{ \tilde{\mathcal{L}}_{\rho,\gamma}\left(\mathbf{a}_\mathbf{j}, \tilde{\boldsymbol{\mu}}_\mathbf{j}[\tau], \tilde{\boldsymbol{\nu}}[\tau]; \mathbf{a}_\mathbf{j}[\tau]\right) \right\} \tag{3.50}$$

$$\boldsymbol{\mu}_\mathbf{j}[\tau+1] = \boldsymbol{\mu}_\mathbf{j}[\tau] + \rho\gamma_\mathbf{j}\tilde{\mathbf{G}}_\mathbf{j}\mathbf{a}_\mathbf{j}[\tau+1] \tag{3.51}$$

$$\tilde{\boldsymbol{\mu}}_\mathbf{j}[\tau+1] = \boldsymbol{\mu}_\mathbf{j}[\tau+1] + \rho\hat{\gamma}_\mathbf{j}\tilde{\mathbf{G}}_\mathbf{j}\mathbf{a}_\mathbf{j}[\tau+1] \tag{3.52}$$

$$\text{Communicate } \{\mathbf{a}_\mathbf{j}\} \text{ to neighbours} \tag{3.53}$$

$$\boldsymbol{\nu}_\mathbf{j}[\tau+1] = \boldsymbol{\nu}_\mathbf{j}[\tau] + \rho\gamma_\mathbf{j}\mathbf{B}_{\mathbf{j},-}\mathbf{a}[\tau+1] \tag{3.54}$$

$$\tilde{\boldsymbol{\nu}}_\mathbf{j}[\tau+1] = \boldsymbol{\nu}_\mathbf{j}[\tau+1] + \rho\hat{\gamma}_\mathbf{j}\mathbf{B}_{\mathbf{j},-}\mathbf{a}[\tau+1]. \tag{3.55}$$

$$\text{Communicate } \{\tilde{\boldsymbol{\nu}}_\mathbf{j}\} \text{ to neighbours} \tag{3.56}$$

In the above, $\rho > 0$ is the step-size and $\gamma_\mathbf{j}, \hat{\gamma}_\mathbf{j} > 0$ are two over-relaxation terms with $\gamma_\mathbf{j} > \hat{\gamma}_\mathbf{j}$. As shown in [20, 21] and summarized in Appendix A.2, the primal variables $\mathbf{a}$, and dual variables $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ converge to the optimal solution $\mathbf{a}^*$, $\boldsymbol{\mu}^*$, and $\boldsymbol{\nu}^*$. A

key feature of the PAC algorithm is its privacy preserving structure; specifically, the atomic objective functions, constraints, and dual variables (both $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$, which act as shadow prices describing the atom's sensitivity to the feasibility and coordination constraints) are all kept private. This is discussed in more detail in [20], and the interpretation of the dual variables in the context of a market is discussed in this thesis in Chapter 5, as well as [22].

The convergence results of the distributed PAC algorithm can be measured using the following two metrics, as evaluated at each iteration τ:

- Distance to Feasibility: $\left\|\tilde{\mathbf{G}}\mathbf{a}[\tau]\right\|_2$. This metric describes the feasibility gap of the solution (whether power flow equations are satisfied).

- Distance to Consistency: $\|\mathbf{B}\mathbf{a}[\tau]\|_2$. This metric describes how far the variable copies are from their true values. This can also be thought of as distance to coordination. Recall the variable copies were introduced to distribute the problem.

3.4.3 Distributed Optimization with CI Model

With the CI model introduced in the previous section, we can now move on to discuss the application of the PAC algorithm to solve the CI formulation of the OPF. We begin first by assuming the objective function is convex. This is, in general, a very reasonable assumption as costs typically follow a squared law (and so are convex). Note that we do not require the objective function to be separable amongst the nodes; this is because the decomposition profile for PAC details how to address the coupling variables between atoms, and atoms are already coupled within the power flow constraints. The centralized problem as introduced in (3.31)-(3.36) has variables as follows:

$$\mathbf{y} = \begin{bmatrix} I^r & I^i & V^r & V^i & P & Q & a & b & c & d \end{bmatrix} \tag{3.57}$$

where each term in the vector $\mathbf{y}$ is of the form:

$$I^r = [I^r_{\mathbf{j}}] \qquad \forall \mathbf{j} \in [N] \tag{3.58}$$

$$I^r_{\mathbf{j}} = \begin{bmatrix} I^r_{\mathbf{j},a} & I^r_{\mathbf{j},b} & I^r_{\mathbf{j},c} \end{bmatrix} \tag{3.59}$$

where the superscripts r and i denote real and imaginary components respectively, the subscript j denotes the node, and the subscripts a, b, c denote the phases. The same follows for all other terms in y. The atomic vector, $\mathbf{a}_j$ then takes on the form of (3.60), where the last term are the coupling variables denoted by the $\hat{}$ symbol.

$$\mathbf{a}_j = \begin{bmatrix} I^r_{\mathbf{j}} & I^i_{\mathbf{j}} & V^r_{\mathbf{j}} & V^i_{\mathbf{j}} & P_{\mathbf{j}} & Q_{\mathbf{j}} & a_{\mathbf{j}} & b_{\mathbf{j}} & c_{\mathbf{j}} & d_{\mathbf{j}} & \{\hat{V}^r_{\mathbf{k}}, \hat{V}^i_{\mathbf{k}}\}_{\forall \mathbf{k}, \{(\mathbf{j},\mathbf{k}),(\mathbf{k},\mathbf{j})\} \in \mathcal{T}_{\mathrm{D}}} \end{bmatrix} \tag{3.60}$$

The OPF problem can be solved for voltage regulation, whereby the objective is to regulate the real and imaginary components of the voltages about some desired reference value. The objective function can be written as follows:

$$f(\mathbf{y}) = \sum_{\mathbf{j} \in [N]} \sum_{p \in a,b,c} [(V^r_{\mathbf{j},p} - \hat{V}^r_{\mathbf{j}})^2 + (V^i_{\mathbf{j},p} - \hat{V}^i_{\mathbf{j}})^2] \tag{3.61}$$

where $\hat{V}^r_{\mathbf{j}}$ and $\hat{V}^i_{\mathbf{j}}$ are the desired setpoints. In the rest of this section, $\hat{V}^r_{\mathbf{j}}$ and $\hat{V}^i_{\mathbf{j}}$ are the same for all nodes in the network; we note that node $j^{\#}$ which is the point of common coupling (PCC) to the transmission grid is treated as a slack node, with $V^r_{\mathbf{j}^{\#}} = 1$ and $V^i_{\mathbf{j}^{\#}} = 0$, $\forall p \in a, b, c$. This 'slack bus' is the boundary condition which allows us to uniquely determine power flows in the network for given nodal injections.

Simulations & Computational Setup

The OPF problem is run with different power flow models in the day-ahead and real-time energy markets, for both scheduling and real-time operations. As such, the computational setup must be efficient enough to run in real-time. Figure 3-4 describes the workflow as implemented in this work. The General Model is the PAC decomposition of a specific network. This can be run when setting up the operations

to make use of the CI model and PAC algorithm, and only needs to be re-run if the network topology or participating DER agents are changed.

The next set of steps is the integration of Scenario Specific Data, which must be re-run for every run of the PAC algorithm. This means these steps must be run once for every day-ahead OPF problem, and run every 5 minutes [3] for a real-time OPF solution. Before these steps are run, the forecast for demand and generation must be updated. For real-time operation, the forecasts for period $\tau+1$ can be developed while the PAC algorithm is solving the OPF for period τ. Once forecasts are generated, the Scenario Specific Data can be integrated; thus, if the forecasts are ready before the PAC solution for period τ is completed, the problem can be setup for period $\tau + 1$, giving more time for the PAC algorithm to converge for period $\tau + 1$. Note that the function setup only needs to be rerun if any changed variable bounds or item costs are present in the objective function.

The final step before running the PAC algorithm is the PAC setup, which includes determining the optimal PAC step sizes (ρ and γ_j), and the initial conditions for the algorithm. Note that for most networks and scenarios, the PAC parameters are the same; thus in real implementation, this step can be run with the General Model and values saved. Finally, the initial conditions can be either computed to an estimated optimal point, fixed at some values relative to the upper and lower bounds of the variables, or for real-time operations, set to be the optimal solution from the previous period.

Finally, the PAC algorithm must be run in parallel over multiple cores or workstations to mimic the distributed nature of the algorithm. Note that the implementation for this thesis is in MATLAB, where only limited functionalities are present for parallelizing code, as MATLAB readily makes use of all cores in a machine. An implementation in Python, as a comparison, would enable separate computational agents to be formed, which could each use data pipes to selectively share information. For the scope of this thesis, a `parfor` loop is used to parallelize the primal update across

[3]The 5 minute market clearing for the real-time market is for the New England ISO (ISO-NE). This can vary from 15 minutes to 5 minutes, depending on the market structure within a given ISO or RTO.

the computational agents. Once the primal update - which consists of solving the atomic OPF for the k^{th} iteration - is complete, the dual variables are updated in a centralized manner. Note that this is not a limitation of the PAC algorithm, but rather a design choice to simplify the implementation.

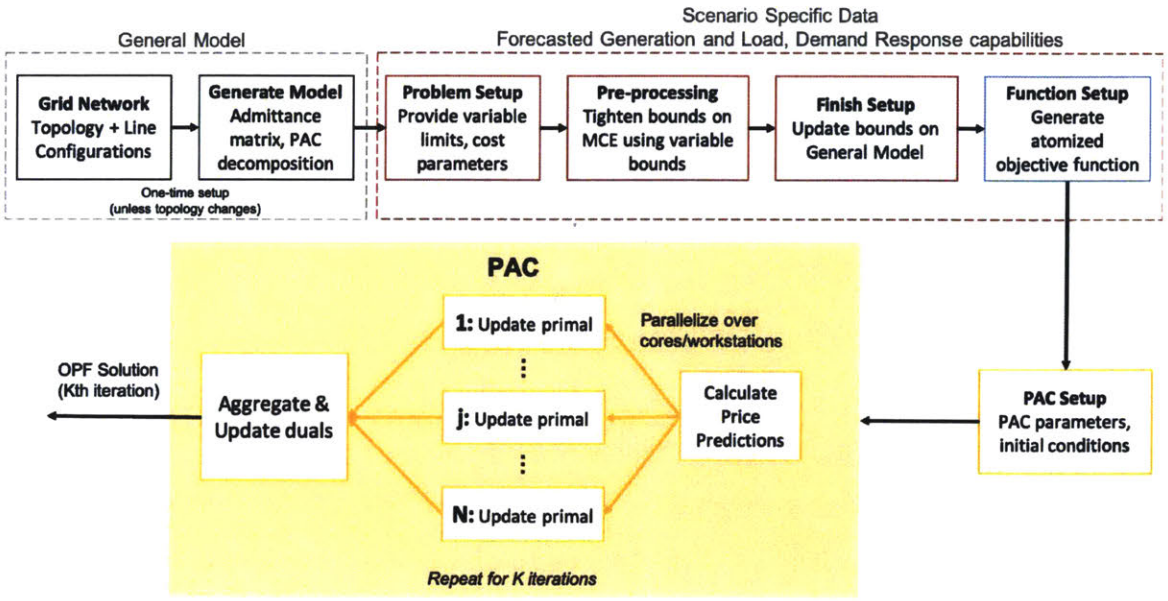


Figure 3-4: Flowchart of the computational process. The PAC algorithm decomposition is fixed for a network with a given topology. The scenario or time-period specific updates must be run each time the PAC algorithm is run.

For simulations in Chapters 3-4, the following computational tools are used:

- All simulations are carried out in MATLAB [30].
- General Model: the global optimization problem with the CI model is setup using the YALMIP package, which takes a set of objective functions and constraints, and maps them into a format acceptable by a third-party solver [31].
- Pre-processing: the current bounds are obtained by solving the set of nonlinear optimization problems using Ipopt, a nonlinear solver, within the OPTI package.
- PAC Setup: each atomic optimization problem is setup using the YALMIP package, which formats the objective function and constraints into a format acceptable by the Gurobi commercial optimization solver [32]. For each iteration

of the PAC algorithm, the augmented Lagrangian is updated with the new predictions of the dual variables.

- PAC Primal Update: the primal problem is solved directly in Gurobi.

Results from CI Model

The proposed CI model is solved using the PAC decomposition profile and algorithm. The simulation results for the IEEE 13 node network, which has been modified to include DERs (see Appendix B.2 for data), is shown in Fig. 3-5. The convergence metrics of Distance to Feasibility and Consistency are plotted.

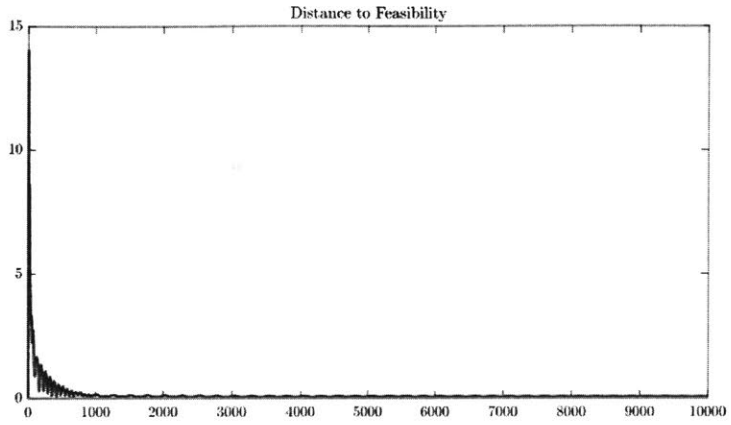
From the convergence trajectory, we see that the algorithm takes an unreasonably long time to approach the optimal solution. In particular, the dual problem of consistency is very slow. The step sizes (ρ, γ) are very small, as shown in Table 3.1. These can be attributed to the ill-conditioned system matrix $\tilde{\mathbf{G}}^T\tilde{\mathbf{G}} + \mathbf{B}^T\mathbf{B}$, which has a very high condition number and $\lambda_{\max}$.

$\alpha = 1$	$\gamma = 3.4701e-4$	Condition number: $2.505e7$
$L = 12.1696$	$\rho = 0.2867$	$\lambda_{\max} = 3.5070e+4$

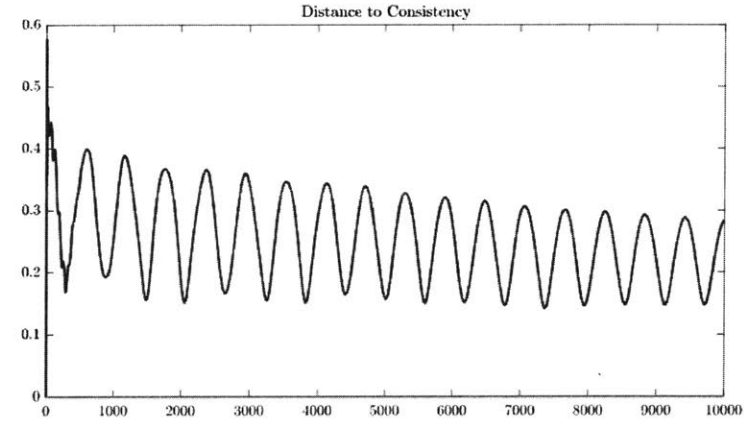
Table 3.1: Parameters for PAC algorithm for the Active IEEE 13 node network, using the CI model.

3.5 Reformulation of CI Model

The results from the previous section show unacceptable performance of the PAC algorithm on the proposed CI model due to the ill-conditioned system matrix. There are various approaches to dealing with ill-conditioned systems, however, the source should first be investigated. In this section, the CI model is reformulated to result in better system conditioning and performance for the PAC algorithm. Then, a comparison of the two formulations is presented, where the eigenvalues of the system matrix are analyzed.



(a) Feasibility Metric



(b) Consistency Metric

Figure 3-5: Convergence results for the IEEE 13 node network, modified to have DERs.

3.5.1 Source of Ill-Conditioning and Reformulation

In this problem formulation, the modelling choices are the source of ill-conditioning, particularly in the $I = \mathbf{Y}V$ constraint. The other constraints are simply definitions with unity coefficients; further all the variables are bounded on the order of 1, because we are dealing with per unit analysis of the electrical quantities. However, the admittance per length are all on the order of 0.1, leading to larger coefficients once the line length and per unit conversion are accounted for.

The first possibility is to reformulate the CI model [4] to use the line impedances,

[4]Hereby called CI-Y to indicate the CI model formulated using admittances, or CI-Z to indicate

rather than the admittances. Lets return to (3.6)-(3.8). Here we have KCL giving the relationship between nodal injections and line flows, and Ohm's law relating the nodal voltages to line flows. In our case, however, we want to stay in the realm of impedances and not move towards the cleaner representation we get from admittances. We can now define the graph incidence matrix, A. In this case, we define the outgoing edge as positive and incoming edge as negative. Then, if we assemble the diagonal matrix of line impedances, $\mathbf{Z}$, and denote the vector of line currents as I_{flow} and the vector of voltages as V, we can rewrite Ohm's law and KCL as:

$$\mathbf{A}V = \mathbf{Z}I_{\text{flow}} \tag{3.62}$$

$$I = \mathbf{A}^T I_{\text{flow}} \tag{3.63}$$

where the column vector I is the nodal current injections.

Then under this new model, we can rewrite the CI model using impedances (CI-Z) as follows:

the CI model formulated using impedances.

$$\min_{\mathbf{y}} \quad f(\mathbf{y})$$

$$\text{s.t.} \quad I = \mathbf{A}^T I_{\text{flow}} \tag{3.64}$$

$$\mathbf{A}V = \mathbf{Z}I_{\text{flow}} \tag{3.65}$$

$$I_{\mathbf{j}} = I_{\mathbf{j}}^r + jI_{\mathbf{j}}^i \qquad \forall \mathbf{j} \in [N] \tag{3.66}$$

$$V_{\mathbf{j}} = V_{\mathbf{j}}^r + jV_{\mathbf{j}}^i \qquad \forall \mathbf{j} \in [N] \tag{3.67}$$

$$P_{\mathbf{j}} = a_{\mathbf{j}} + b_{\mathbf{j}} \qquad \forall \mathbf{j} \in [N] \tag{3.68}$$

$$Q_{\mathbf{j}} = -c_{\mathbf{j}} + d_{\mathbf{j}} \qquad \forall j \in [N] \tag{3.69}$$

$$\underline{I_{\mathbf{j}}^r} \leq I_{\mathbf{j}}^r \leq \overline{I_{\mathbf{j}}^r} \qquad \forall \mathbf{j} \in [N] \tag{3.70}$$

$$\underline{I_{\mathbf{j}}^i} \leq I_{\mathbf{j}}^i \leq \overline{I_{\mathbf{j}}^i} \qquad \forall \mathbf{j} \in [N] \tag{3.71}$$

$$\underline{V_{\mathbf{j}}^r} \leq V_{\mathbf{j}}^r \leq \overline{V_{\mathbf{j}}^r} \qquad \forall \mathbf{j} \in [N], \mathbf{j} \neq \mathbf{j}^{\#} \tag{3.72}$$

$$\underline{V_{\mathbf{j}}^i} \leq V_{\mathbf{j}}^i \leq \overline{V_{\mathbf{j}}^i} \qquad \forall \mathbf{j} \in [N], \mathbf{j} \neq \mathbf{j}^{\#} \tag{3.73}$$

$$\underline{P_{\mathbf{j}}} \leq P_{\mathbf{j}} \leq \overline{P_{\mathbf{j}}} \qquad \forall \mathbf{j} \in [N] \tag{3.74}$$

$$\underline{Q_{\mathbf{j}}} \leq Q_{\mathbf{j}} \leq \overline{Q_{\mathbf{j}}} \qquad \forall \mathbf{j} \in [N] \tag{3.75}$$

$$V_{\mathbf{j}^{\#}}^r = 1, \quad V_{\mathbf{j}^{\#}}^i = 0 \tag{3.76}$$

$$\text{MCE bounds for } a_{\mathbf{j}}, b_{\mathbf{j}}, c_{\mathbf{j}}, d_{\mathbf{j}} \text{ of form (3.27)-(3.30)} \qquad \forall \mathbf{j} \in [N] \tag{3.77}$$

The centralized problem of (4.26)-(3.77) has variables as follows:

$$\mathbf{y} = \begin{bmatrix} I^r & I^i & V^r & V^i & P & Q & a & b & c & d & I_{flow}^r & I_{flow}^i \end{bmatrix} \tag{3.78}$$

where each term takes on the form of (3.58)-(3.59). The atomic vector, $\mathbf{a}_j$ then takes on the form of (4.48), where the convention that an atom owns its downstream current flows is used [5]. Once again the last two terms gives the coupling variables denoted

[5]This convention is chosen as most nodes will have more downstream nodes than upstream nodes. Thus, by choosing that an atom owns downstream current flows, we attempt to minimize the number of variables which must be copied to satisfy constraints.

by the $\hat{}$ symbol.

$$\mathbf{a}_j = [I^r_{\mathbf{j}} \quad I^i_{\mathbf{j}} \quad V^r_{\mathbf{j}} \quad V^i_{\mathbf{j}} \quad P_{\mathbf{j}} \quad Q_{\mathbf{j}} \quad a_{\mathbf{j}} \quad b_{\mathbf{j}} \quad c_{\mathbf{j}} \quad d_{\mathbf{j}} \quad \{I^r_{flow,\mathbf{jk}}, I^i_{flow,\mathbf{jk}}\}_{\forall \mathbf{k},\{(\mathbf{j},\mathbf{k})\}\in\mathcal{T}_\mathbf{D}} \quad \{\hat{V}^r_{\mathbf{k}}, \hat{V}^i_{\mathbf{k}}\}_{\forall \mathbf{k},\{(\mathbf{j},\mathbf{k})\}\in\mathcal{T}_\mathbf{D}} \quad \{\hat{I}^r_{flow,\mathbf{ij}}, \hat{I}^i_{flow,\mathbf{ij}}\}_{\forall \mathbf{i},\{(\mathbf{i},\mathbf{j})\}\in\mathcal{T}_\mathbf{D}}] \quad (3.79)$$

We note that the CI-Z formulation has increased the number of variables, the number of constraints, and the number of coupling variables (copies of the parent flows are required). Despite the larger central and atomic optimization problems however, this formulation significantly improves the performance of the PAC algorithm.

From the simulation results on the IEEE 13 node network (see Appendix B.2 for data) with the CI-Z model, the performance improvement is readily seen as in Fig. 3-6. The PAC algorithm now gets close to the optimal solution within 2000 iterations, and converges within 3000 iterations, with just this simple modeling change. The evolution of the objective cost is also shown in Fig. 3-6c for this model. In particular, the matrix $\tilde{\mathbf{G}}^T\tilde{\mathbf{G}} + \mathbf{B}^T\mathbf{B}$ now has much lower condition number, allowing for larger step sizes, with the relevant parameters shown in Table 3.2.

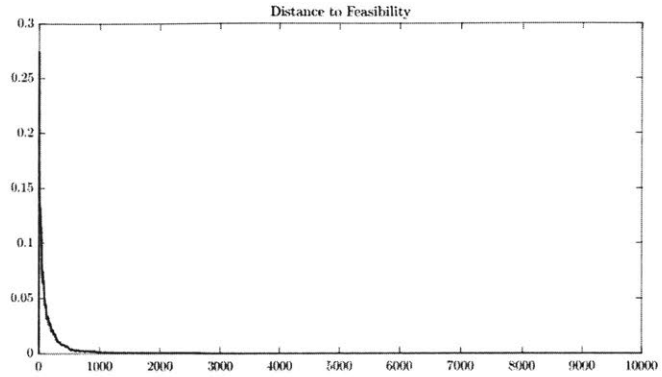
$\alpha = 1$	$\gamma = 1.931$	Condition number: 120
$L = 12.1696$	$\rho = 0.2873$	$\lambda_{\max} = 6.2758$

Table 3.2: Parameters for PAC algorithm for the Active IEEE 13 node network, with the modified CI model.

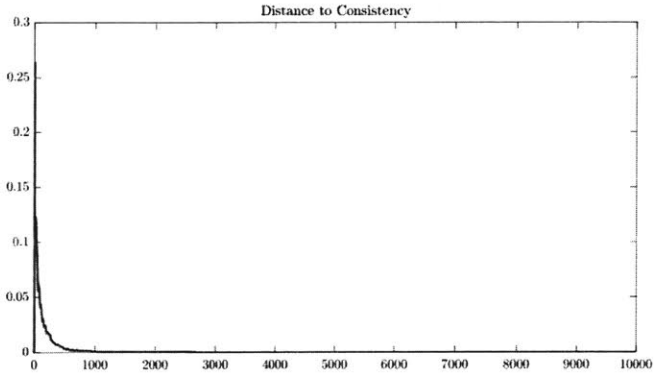
With this new formulation, larger networks can be simulated,, such as the IEEE 123 node network, as presented in Appendix B.3. The algorithm approaches optimal within 20000 iterations, as shown in Fig. 3-7.

3.5.2 Analysis of System Condition Number

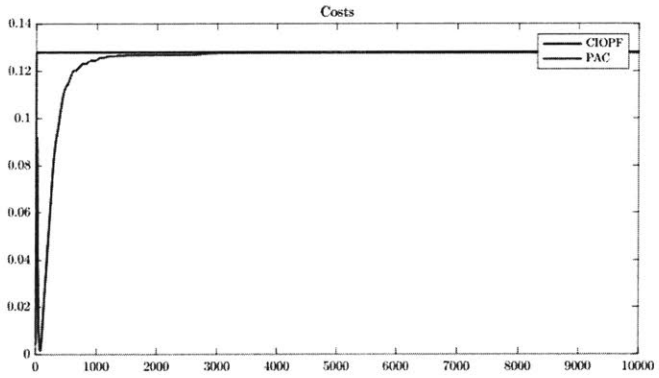
The performance of the PAC algorithm for the two models (CI-Y and CI-Z) can be further analyzed by looking at the matrices $\tilde{\mathbf{G}}_\mathbf{j}$ and $\mathbf{B}$, which correspond to the atomized equality constraint matrix and the coordination matrix respectively. In particular, from [20, 21] we note the selection of the convergence rate parameters



(a) Feasibility Metric



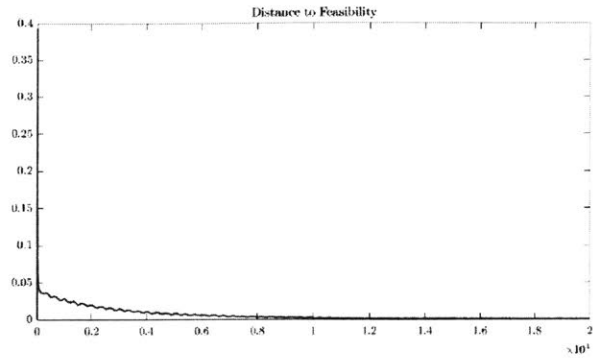
(b) Consistency Metric



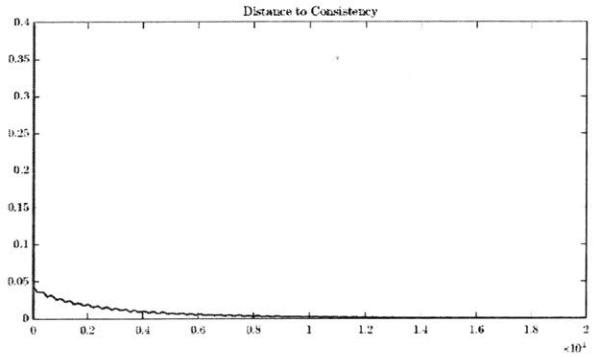
(c) Objective cost

Figure 3-6: Convergence results for the IEEE 13 node network, modified to have DERs with the modified CI model.

depends on these matrices; these are the step-size ρ which is common to all agents, and the over-relaxation terms γ_j and $\hat{\gamma}_j[\tau]$, the latter of which is time-varying. For simplicity, here we consider the case where γ_j is the same for all agents, and $\gamma_j =$



(a) Feasibility Metric



(b) Consistency Metric

Figure 3-7: Convergence results for the IEEE 123 node network with the modified CI model.

$\hat{\gamma}_{\mathbf{j}}, \forall \mathbf{j}, \forall \tau$.

We begin by defining the following as from [20, 21]:

$$\tilde{\mathbf{G}} = \text{diag}(\tilde{\mathbf{G}}_{\mathbf{j}})_{\forall j} \tag{3.80}$$

$$\mathbf{V} = \tilde{\mathbf{G}}^T \tilde{\mathbf{G}} + \mathbf{B}^T \mathbf{B} \tag{3.81}$$

$$\rho^* = \frac{1}{\sqrt{\gamma^* \lambda_{\max}(\mathbf{V})}} \tag{3.82}$$

$$\gamma^* = \frac{2\alpha L}{2\lambda_{\max}(\mathbf{V}) + \hat{\lambda}_{\min}(\mathbf{V})} \tag{3.83}$$

where $\tilde{\mathbf{G}}$ is a block diagonal matrix composed of the atomic equality constraints; $\lambda_{\max}(\mathbf{V})$ is the largest eigenvalue of $\mathbf{V}$; and $\hat{\lambda}_{\min}(\mathbf{V})$ is the smallest non-zero eigenvalue of $\mathbf{V}$. We also assume that each $f_{\mathbf{j}}$ is differentiable, α-strongly convex and L-strongly

smooth. Then ρ^* and γ^* are the convergence parameters which maximize the primal convergence rate. For the CI-Y model, the convergence is very slow because ρ^* and γ^* are very small, as governed by the very large $\lambda_{\max}(\mathbf{V})$. Thus, analysis of the eigenvalues of matrix $\mathbf{V}$ is necessary; we will bound the value of the eigenvalues and use the results to show why the CI-Z model outperforms the CI-Y model when solved with PAC.

We begin our analysis by noting that the matrix $\mathbf{V}$ is a positive semi-definite matrix (PSD), as it is the sum of two PSD matrices. From *Courant-Fischer min-max Theorem*, we have

$$\lambda_{\max}(\mathbf{V}) \leq \lambda_{\max}(\tilde{\mathbf{G}}^T\tilde{\mathbf{G}}) + \lambda_{\max}(\mathbf{B}^T\mathbf{B}) \tag{3.84}$$

which allows us to analyze each component separately.

3.5.3 Eigenvalue analysis of Coordination Constraints

Let's first consider the eigenvalues of $\mathbf{B}^T\mathbf{B}$, which is the Laplacian of the coordination matrix. Namely, the coordination matrix can be thought of as an adjacency matrix of a directed graph, where the nodes in the graph are the variables a_1 (a variable owned by agent 1), and $\hat{a}_1$ (a variable copy of a_1 owned by another agent). Then the Laplacian ($\mathbf{B}^T\mathbf{B}$) will consist of the elements -1 on the off-diagonals, and the degree of each self-owned variable on the diagonals (namely the number of times a variable has been copied). We can apply *Gershgorin's Circle Theorem*, which states that every eigenvalue of a square matrix ($\in \mathbb{R}^{n\times n}$) lies within a Gershgorin disc, which is a circle centered at $b_i i$ with radius R_i, where:

$$R_i = \sum_{j=1, j\neq i}^{n} |b_{ij}|$$

b_{ii} is the element on the diagonal

Then:

$$|\lambda_i - b_{ii}| \leq R_i \tag{3.85}$$

For the Laplacian, the element on the diagonal is the degree of the i^{th} variable, denoted as dg_i. We also know that $R_i = \text{dg}_i$ for all rows, as the sum of the rows of the Laplacian are all zero. Then we can simplify the inequality as follows:

$$|\lambda_i - b_{ii}| \leq R_i \tag{3.86}$$

$$|\lambda_i - \text{dg}_i| \leq \text{dg}_i \tag{3.87}$$

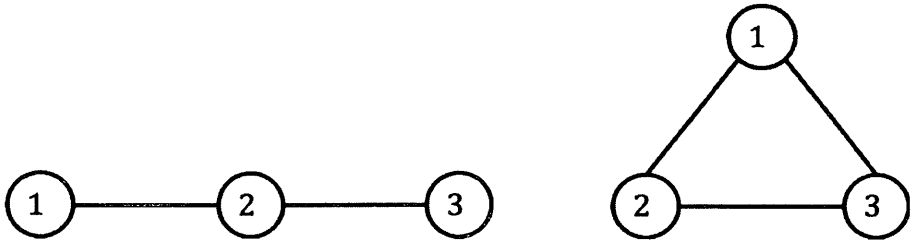
$$\rightarrow \lambda_i \leq 2\text{dg}_i \tag{3.88}$$

$$\rightarrow \lambda_i \geq 0 \tag{3.89}$$

where the condition that $\lambda_i \geq 0$ is known as the Laplacian is a PSD matrix. Then the maximum eigenvalue of $\mathbf{B}^T\mathbf{B}$ is bounded by the maximum degree:

$$\lambda_{\max}(\mathbf{B}^T\mathbf{B}) \leq 2\max(\text{dg}_i) \tag{3.90}$$

Since the decomposition profile is determined by the model chosen and our rules on which agents retain copies of which coupling variables, we can further analyze the CI-Y and CI-Z models. Throughout our analysis, we will consider the two following simple cases: a 3-node radial network, and a 3-node meshed network as in Fig. 3-8



(a) Radial network (b) Mesh network

Figure 3-8: Simple topologies for discussing the system condition number

Eigenvalues of Coordination: CI-Y Model

As seen in (3.60), the agents must retain copies of the voltages of all neighbouring agents. Then for the radial network in Fig. 3-8a, agent 1 will require copies of voltages from agent 2; agent 2 requires copies of voltages from agent $1, 3$; agent 3 requires copies of voltages from agent 2. We see that only the voltages from agent 2 are copied twice, while the others are only copied once. Thus, $\max(\mathrm{dg}_i) = 2$ for this example. For the meshed network in Fig. 3-8b, all the voltages are copied by two other agents, thus $\max(\mathrm{dg}_\mathbf{i}) = 2$ as well. In the CIY model, the maximum number of variable copies is equal to the maximum degree of the nodes in the network (i.e. the number of neighbours):

$$\lambda_{\max}(\mathbf{B}^T\mathbf{B}) \leq 2 \max_{\mathbf{j}} (\deg(\mathbf{j})) \tag{3.91}$$

In the two example networks:

$$\text{Radial: } \lambda_{\max}(\mathbf{B}^T\mathbf{B}) \leq 4 \tag{3.92}$$

$$\text{Mesh: } \lambda_{\max}(\mathbf{B}^T\mathbf{B}) \leq 4 \tag{3.93}$$

Eigenvalues of Coordination: CI-Z Model

As seen in (4.48), the agents must retain copies of the voltages of downstream nodes ($\hat{V}_\mathbf{k}$) and copies of the flows from upstream nodes ($\hat{I}_\mathbf{ij}$). Note here, that a node $\mathbf{j}$ has downstream nodes $\mathbf{k}$ where the numbers $\mathbf{j} < \mathbf{k}$ (child nodes), and upstream nodes $\mathbf{i}$ where the numbers $\mathbf{i} < \mathbf{j}$ (parent nodes). Then, the voltages of a node are copied by its parents, and the outward flows of a node are copied by its children. However, we note that only voltages can be copied multiple times, as lines flows have unique {parent,child} pairs. Thus, only the number of parents is relevant.

In a radial network, as in Fig. 3-8a, nodes can have only one parent. Then, we

can uniquely determine dg_i for all radial networks as:

$$\max(dg_i) = 1 \tag{3.94}$$

$$\lambda_{max}(\mathbf{B}^T\mathbf{B}) \leq 2 \tag{3.95}$$

In the meshed case, this becomes more complicated as each node can have multiple parents (consider node 3) or multiple children (consider node 1). Then we can define the maximum number of copies as:

$$\max(dg_i) = \max_j(\deg^{in}(\mathbf{j})) \tag{3.96}$$

where $\deg^{in}(\mathbf{j})$ denotes the in-degree of the node $\mathbf{j}$.

In the two example networks:

$$\text{Radial: } \lambda_{max}(\mathbf{B}^T\mathbf{B}) \leq 2 \tag{3.97}$$

$$\text{Mesh: } \lambda_{max}(\mathbf{B}^T\mathbf{B}) \leq 4 \tag{3.98}$$

where we note that all radial networks will be bounded by 2.

Eigenvalues of Coordination: Comparison of CI-Y and CI-Z Models

From the analysis above, we can make the following conclusions:

- For a radial network, the CI-Z model always yields smaller λ_{max} than the CI-Y model (other than the trivial case of two connected nodes).
- For a meshed network, the CI-Z model will yield the same or smaller λ_{max}, as $\max_j(\deg^{in}(\mathbf{j})) \leq \max_j(\deg(\mathbf{j}))$. This depends on the specific meshed topology.

It's interesting to note that although the CI-Z model requires more coupling variables, the impact on the maximum eigenvalues is the opposite - this is because of the particular decomposition chosen. We also note that the difference between CI-Y and CI-Z here is quite small - there isn't an order of magnitude difference which could explain

the huge performance difference. Thus, the eigenvalues of $\tilde{\mathbf{G}}^T\tilde{\mathbf{G}}$ must dominate, and will explain the difference between the two models. This will be discussed next.

3.5.4 Eigenvalue analysis of Equality Constraints

We can now consider the eigenvalues of $\tilde{\mathbf{G}}^T\tilde{\mathbf{G}}$. We begin by noting that $\tilde{\mathbf{G}}$ is a block diagonal matrix, and so the following holds:

$$\lambda(\tilde{\mathbf{G}}^T\tilde{\mathbf{G}}) = \{\lambda(\tilde{\mathbf{G}}_\mathbf{j}^T\tilde{\mathbf{G}}_\mathbf{j})\}_{\forall j} \tag{3.99}$$

$$\lambda_{\max}(\tilde{\mathbf{G}}^T\tilde{\mathbf{G}}) = \max_{\mathbf{j}}(\lambda_{\max}(\tilde{\mathbf{G}}_\mathbf{j}^T\tilde{\mathbf{G}}_\mathbf{j})) \tag{3.100}$$

We now have a simpler problem, where we just need to characterize the eigenvalues of the atomic equality constraint matrices. We can also show that:

$$\lambda_{\max}(\tilde{\mathbf{G}}_\mathbf{j}^T\tilde{\mathbf{G}}_\mathbf{j}) = \|\tilde{\mathbf{G}}_\mathbf{j}^T\tilde{\mathbf{G}}_\mathbf{j}\|_2 \leq \|\tilde{\mathbf{G}}_\mathbf{j}\|_2^2 \leq \|\tilde{\mathbf{G}}_\mathbf{j}\|_F^2 \tag{3.101}$$

where $\|\cdot\|_2$ denotes the usual Euclidean norm and $\|\cdot\|_F$ denotes the Frobenius norm. We also note the following for a matrix $A \in \mathbb{R}^{m\times n}$:

$$\|A\|_F = \sqrt{\mathrm{Tr}(A^\dagger A)} = \sqrt{\sum_{\mathbf{i}=1}^{m}\sum_{\mathbf{j}=1}^{n}|a_{\mathbf{ij}}|^2} \tag{3.102}$$

where $A^\dagger$ is the Hermitian transpose (conjugate transpose). We note that our matrix $\tilde{\mathbf{G}}_\mathbf{j}^T\tilde{\mathbf{G}}_\mathbf{j}$ is a real matrix, so this is just a transpose. Since the equality constraints (feasibility) are fully defined for our two models, we can further analyze the CI-Y and CI-Z models as we did for the coordination constraints.

Eigenvalues of Feasibility: CI-Y Model

We begin by recalling the optimization problem formulated by (3.31)-(3.36). The constraints from the CI-Y model can be easily distributed, and the $\tilde{\mathbf{G}}_\mathbf{j}$ matrix is

defined as:

$$\tilde{\mathbf{G}}_{\mathbf{j}} = \begin{bmatrix} I^r_{\mathbf{j}} = g_r(V^r_{\mathbf{j}}, V^i_{\mathbf{j}}, \tilde{V}^r_{\mathbf{k}}, \tilde{V}^i_{\mathbf{k}}))_{\forall \mathbf{k}, \{(\mathbf{j},\mathbf{k}),(\mathbf{k},\mathbf{j})\} \in \mathcal{T}_{\mathrm{D}}} \\ I^i_{\mathbf{j}} = g_i(V^r_{\mathbf{j}}, V^i_{\mathbf{j}}, \tilde{V}^r_{\mathbf{k}}, \tilde{V}^i_{\mathbf{k}}))_{\forall \mathbf{k}, \{(\mathbf{j},\mathbf{k}),(\mathbf{k},\mathbf{j})\} \in \mathcal{T}_{\mathrm{D}}} \\ P = a + b \\ Q = c + d \end{bmatrix} \tag{3.103}$$

where the same constraints are present for all nodes. Let's begin by considering a single-phase network, and determine the functions g_r and g_i to characterize the first two constraints.

$$I_{\mathbf{j}} = \sum_{\mathbf{l}}(-Y_{\mathbf{l}}V_{\mathbf{l}}) + (\sum_{\mathbf{l}} Y_{\mathbf{l}})V_{\mathbf{j}} \tag{3.104}$$

$$V_{\mathbf{j}} = V^r_{\mathbf{j}} + jV^i_{\mathbf{j}} \tag{3.105}$$

$$Y_{\mathbf{ij}} = Y_{\mathbf{l}} = y^r_{\mathbf{l}} + jy^i_{\mathbf{l}} \tag{3.106}$$

$$I^r_{\mathbf{j}} = -\sum_{\mathbf{l}}(y^r_{\mathbf{l}}V^r_{\mathbf{l}} - y^i_{\mathbf{l}}V^i_{\mathbf{l}}) + (\sum_{\mathbf{l}} y^r_{\mathbf{l}})V^r_{\mathbf{j}} - (\sum_{\mathbf{l}} y^i_{\mathbf{l}})V^i_{\mathbf{j}} \tag{3.107}$$

$$I^i_{\mathbf{j}} = -\sum_{\mathbf{l}}(y^r_{\mathbf{l}}V^i_{\mathbf{l}} + y^i_{\mathbf{l}}V^r_{\mathbf{l}}) + (\sum_{\mathbf{l}} y^r_{\mathbf{l}})V^i_{\mathbf{j}} + (\sum_{\mathbf{l}} y^i_{\mathbf{l}})V^r_{\mathbf{j}} \tag{3.108}$$

where $\mathbf{l} = \{(\mathbf{i},\mathbf{j}),(\mathbf{j},\mathbf{k})\} \in \mathcal{T}_{\mathrm{D}}$ is the set of all connected nodes. Then from (3.107)-(3.108), we can fully characterize the matrix based on the admittances.

We can now evaluate the Frobenius norm, $\|\tilde{\mathbf{G}}_{\mathbf{j}}\|^2_F$ as is the sum of the following terms:

- $P - a - b = 0 \rightarrow (1 + 1 + 1) = 3$
- $Q + c - d = 0 \rightarrow (1 + 1 + 1) = 3$
- $I^r_{\mathbf{j}} - g_r(\cdot) = 0 \rightarrow (1 + \sum_{\mathbf{l}}[(y^r_{\mathbf{l}})^2 + (y^i_{\mathbf{l}})^2] + [\sum_{\mathbf{l}} y^r_{\mathbf{l}}]^2 + [\sum_{\mathbf{l}} y^i_{\mathbf{l}}]^2)$
- $I^i_{\mathbf{j}} - g_i(\cdot) = 0 \rightarrow (1 + \sum_{\mathbf{l}}[(y^r_{\mathbf{l}})^2 + (y^i_{\mathbf{l}})^2] + [\sum_{\mathbf{l}} y^r_{\mathbf{l}}]^2 + [\sum_{\mathbf{l}} y^i_{\mathbf{l}}]^2)$

A discussion on this result will be left after the CI-Z model is discussed.

Eigenvalues of Feasibility: CI-Z Model

We begin by recalling the optimization problem formulated by (4.26)-(3.77). These constraints from the CI-Z model can be easily distributed, and the $\tilde{\mathbf{G}}_\mathbf{j}$ matrix can be easily defined as:

$$\tilde{\mathbf{G}}_\mathbf{j} = \begin{bmatrix} I^r_\mathbf{j} = \sum_{\mathbf{k},\{(\mathbf{j},\mathbf{k}),(\mathbf{k},\mathbf{j})\}\in\mathcal{T}_\mathrm{D}} I^r_{\mathbf{j},\mathbf{k}} \\ I^i_\mathbf{j} = \sum_{\mathbf{k},\{(\mathbf{j},\mathbf{k}),(\mathbf{k},\mathbf{j})\}\in\mathcal{T}_\mathrm{D}} I^i_{\mathbf{j},\mathbf{k}} \\ P = a + b \\ Q = c + d \\ V^r_\mathbf{j} - \tilde{V}^r_\mathbf{k} = g_r(I^r_{\mathbf{j},\mathbf{k}}, I^i_{\mathbf{j},\mathbf{k}})_{\forall \mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} \\ V^i_\mathbf{j} - \tilde{V}^i_\mathbf{k} = g_r(I^r_{\mathbf{j},\mathbf{k}}, I^i_{\mathbf{j},\mathbf{k}})_{\forall \mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} \end{bmatrix} \tag{3.109}$$

where the first 4 constraints are present for all nodes, and the last 2 are present only for nodes with children. Then, much like in the CI-Y model, we begin by considering a single-phase network to determine the functions g_r and g_i to characterize the last two constraints.

$$\sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} (V_\mathbf{j} - \tilde{V}_\mathbf{k}) = \sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} Z_{\mathbf{jk}} I_{\mathbf{jk}} \tag{3.110}$$

$$Z_{\mathbf{jk}} = Z^r_{\mathbf{jk}} + jZ^i_{\mathbf{jk}} \tag{3.111}$$

$$\sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} (V^r_\mathbf{j} - \tilde{V}_\mathbf{k}^{\,r}) = \sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} (Z^r_{\mathbf{jk}} I^r_{\mathbf{jk}} - Z^i_{\mathbf{jk}} I^i_{\mathbf{jk}}) \tag{3.112}$$

$$\sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} (V^i_\mathbf{j} - \tilde{V}_\mathbf{k}^{\,i}) = \sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}} (Z^r_{\mathbf{jk}} I^i_{\mathbf{jk}} - Z^i_{\mathbf{jk}} I^r_{\mathbf{jk}}) \tag{3.113}$$

Then from (3.112)-(3.113), we can fully characterize the matrix based on the impedances.

We can now evaluate the Frobenius norm, $\|\tilde{\mathbf{G}}_\mathbf{j}\|^2_F$ as is the sum of the following terms, which are shown one constraint at a time.

- $I^r_\mathbf{j} - \sum_\mathbf{l} I^r_\mathbf{l} = 0 \rightarrow (1 + \deg(j))$
- $I^i_\mathbf{j} - \sum_\mathbf{l} I^i_\mathbf{l} = 0 \rightarrow (1 + \deg(\mathbf{j}))$
- $P - a - b = 0 \rightarrow (1 + 1 + 1) = 3$

- $Q + c - d = 0 \rightarrow (1 + 1 + 1) = 3$
- $\sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}}(V_\mathbf{j}^r - \tilde{V_\mathbf{k}}^r) - g_r(\cdot) = 0 \rightarrow ((1{+}1){\times}\deg^{\mathrm{out}}(\mathbf{j})) + \sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}}\left[(Z_{\mathbf{jk}}^r)^2 + (Z_{\mathbf{jk}}^i)^2\right]$
- $\sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}}(V_\mathbf{j}^i - \tilde{V_\mathbf{k}}^i) - g_i(\cdot) = 0 \rightarrow ((1{+}1){\times}\deg^{\mathrm{out}}(\mathbf{j})) + \sum_{\mathbf{k},(\mathbf{j},\mathbf{k})\in\mathcal{T}_\mathrm{D}}\left[(Z_{\mathbf{jk}}^r)^2 + (Z_{\mathbf{jk}}^i)^2\right]$

where $\mathbf{l} = \{(\mathbf{i},\mathbf{j}),(\mathbf{j},\mathbf{k})\} \in \mathcal{T}_\mathrm{D}$ is the set of all connected nodes. The last two constraints can be easily combined for a 3-phase network as follows:

$$2 \times \left[2\deg^{\mathrm{out}}(\mathbf{j}) + \mathrm{sum}(\mathrm{vecnorm}(\mathrm{real}(\mathrm{Z}))^2) + \mathrm{sum}(\mathrm{vecnorm}(\mathrm{imag}(\mathrm{Z}))^2)\right] \qquad (3.114)$$

where vecnorm() is the MATLAB command for the vector norm, and $\deg^{\mathrm{out}}(j)$ denotes the out-degree of the node j

Eigenvalues of Feasibility: Comparison of CI-Y and CI-Z Models

In comparing the Frobenius norms of the CI-Y and CI-Z models, the key differences are the two components depending on admittances for the CI-Y model, and the two components depending on impedances in the CI-Z model. Note that the CI-Z model also has two components which depend on the degree of the agents, but we will show that these are minor factors. We make the following observations:

- The CI-Z model depends only on children voltages, and the impedances of downstream lines, while the CI-Y model depends on all connections and the admittances of all lines.
- The per unit impedances and admittances are comparable, with the admittances typically being smaller. Recall that distribution lines are more lossy than transmission lines (have higher resistances), and so the corresponding per unit impedance is higher and admittance is lower.
- Distribution lines are typically quite short, so the line length measured in miles is typically < 1. Since $Y = 1/Z$, the line admittances become larger when divided by the line length. Equivalently, line impedances become smaller when

multiplied by the line length. For example, the IEEE 123 node network has a maximum line length of 0.1894mi, as in Appendix B.3.

- All values are converted to per unit analysis. For the networks we are considering, SBase = 5e3kW, VBase = 4160V, ZBase = 3.4611Ω, YBase = 0.2889S. Recall that we divide by the base value to get to the per unit analysis. This operation results in even smaller impedances and larger admittances.

For a simple 3-node radial network as presented in Appendix B.1, we have the following results for the feasibility eigenvalues for node $j = 1$:

	$\lambda_{\max}(\tilde{\mathbf{G}}_\mathbf{j}^T \tilde{\mathbf{G}}_\mathbf{j})$	$\|\tilde{\mathbf{G}}_\mathbf{j}\|_F^2$
CI-Y Model	2.4015e4	8.2584e4
CI-Z Model	3	42.0045

Table 3.3: Comparison of feasibility eigenvalues for CI-Y and CI-Z models

From these observations we can also make the following comments:

- The CI-Y model may work better for a transmission line where the lines are longer. However, transmission lines have lower impedances, which would further increase the admittance value. Here we see a tradeoff in performance depending on the topology and line parameters.

- Different base values can be chosen to scale the admittances to smaller values; however, this would require lower SBase and VBase values, resulting in higher values for the inequality constraints.

3.6 Comparison with Branch Flow

As highlighted previously in this chapter, the CI model overcomes the limitations of the BF model in that it is readily applicable to unbalanced and meshed networks. However, the application here is using distributed computation, which warrants a comparison between CI and BF when solved by the PAC algorithm.

For simplicity, we consider balanced radial networks under both the CI and BF models. In order to make a fair comparison between the two models, the following changes to the OPF problem are made:

- The IEEE 13-bus network is modified to be a 3-phase balanced network. The impedance matrices are modified under the assumption that the lines are all 3-phase transposed, as per (2.29) and (2.30). The configurations are converted to symmetric matrices, with the positive-sequence element being used to model the cable (recall that the positive- and negative-sequences are identical). See Appendix B.2.3.

- The loads are modified so every node has 3-phase balanced loads. Single-phase spot loads were assumed to be 3-phase. All other loads are averaged over the phases, to give balanced load on all three phases. See Appendix B.2.3.

- The BF model is extended to be 3-phase by replicating the single-line model and solving all three independent and identical problems together. This is done to better align with the 3-phase CI formulation, which utilizes the 3-phase impedance matrix. Further discussion on implementation is provided after presentation of simulation results.

- The objective function is changed from the example of Volt-VAr control to a social welfare function. This has two components: electrical line losses, and the cost of purchasing electricity from the grid. Under the BF model, the convexification of (3.4) to yield a second-order cone (SOC) constraint is exact when branch currents, $l_{ij} = |I_{ij}|^2$, are minimized. Minimizing the line losses thus in turn minimizes l_{ij}, while retaining some physical and operational interpretation. Note that the minimization of branch currents is not required in the CI model. Further, the implementation of Volt-VAr control as done in (3.61) is challenging with the BF model, as we do not have access to voltage or current angles; rather, once the convex BF OPF model is solved, the angles can be recovered (under certain conditions) as discussed in detail in [5,6]. However, this ex-post

angle recovery does not enable us to employ such a simple Volt-VAr control equation. To retain a simple formulation, we instead consider an economic objective function, where the LMP at the feeder, $\lambda^{\mathrm{P}}_{\mathbf{j}\#}$ and $\lambda^{\mathrm{Q}}_{\mathbf{j}\#}$ for real and reactive power are considered [6]. The two components of the social welfare function are weighted by ξ to consider the trade-off between economic considerations and energy efficiency.

The social welfare function is as below:

$$f^{\text{G-Cost}}(\mathbf{y}) = \lambda^{\mathrm{P}}_{\mathbf{j}\#} P_{\mathbf{j}\#} + \lambda^{\mathrm{Q}}_{\mathbf{j}\#} Q_{\mathbf{j}\#} \tag{3.115}$$

$$f^{\text{Loss}}_{\mathbf{i},\mathbf{j}}(\mathbf{y}) = R_{\mathbf{i},\mathbf{j}} |I_{\mathbf{ij}}|^2, \ \ \forall (\mathbf{i},\mathbf{j}) \in \mathcal{T} \tag{3.116}$$

$$f^{\text{welfare}}(\mathbf{y}) = f^{\text{G-Cost}}(\mathbf{y}) + \xi \Big[\sum_{(\mathbf{i},\mathbf{j}) \in \mathcal{T}} f^{\text{Loss}}_{\mathbf{i},\mathbf{j}}(\mathbf{y}) \Big] \tag{3.117}$$

For the simulations, we take the following values: $\xi = 1, \lambda^{\mathrm{P}}_{\mathbf{j}\#} = \$30/\text{MWh}, \lambda^{\mathrm{Q}}_{\mathbf{j}\#} = \$10/\text{MWh}$. Network data and configurations can be found in Appendix B.2.3. The simulation is run to make sure both models converge, with feasibility and consistency errors on the order of $1e^{-3}$. The PAC variables are initialized to the solution of the atomized problem solved using a centralized program. For the CI model, we can easily implement the social welfare function by recalling:

$$|I_{flow,\mathbf{ij}}|^2 = I_{flow,\mathbf{ij}} I^*_{flow,\mathbf{ij}} \tag{3.118}$$

$$= (I^r_{flow,\mathbf{ij}}{}^2 + I^i_{flow,\mathbf{ij}}{}^2) \tag{3.119}$$

where the symbol * denotes the complex conjugate. To determine the optimal PAC convergence parameters, we take the objective function to be α-strongly convex [7] and

[6] Note that in electricity markets today, reactive power is not priced. We however, set a small price on reactive power to make the problem a little more interesting.

[7] The convexity measure is bounded as $0 \leq \alpha \leq 1$, where larger α values correspond to more strongly convex functions.

L-Lipschitz continuous [8]. We set $\alpha = 0.5$ [9]. We estimate the best Lipschitz constant by taking the maximum value of $\frac{df^{\text{welfare}}(\mathbf{y})}{d\mathbf{y}}$ over the feasibility region described by the inequality constraints (i.e. bounds on $\mathbf{y}$, not considering the equality constraints). Note that the optimal step sizes are a strong function of both the system condition number (which depends on the model, as discussed earlier in this chapter) and the Lipschitz parameter. For the CI model, the Lipschitz parameter is calculated as:

$$L = 3\Big[\lambda^{\mathrm{P}}_{\mathbf{j}\#} + \lambda^{\mathrm{Q}}_{\mathbf{j}\#}\Big] + \xi\Big[\sum_{(\mathbf{i},\mathbf{j})\in\mathcal{T}} R_{\mathbf{i},\mathbf{j}}(2\overline{I^r_{flow,\mathbf{ij}}} + 2\overline{I^i_{flow,\mathbf{ij}}})\Big] \tag{3.120}$$

where the factor of 3 in the first term comes from the 3-phase model of the network. As a very conservative bound, we take $\overline{I^r_{flow,\mathbf{ij}}} = \overline{I^i_{flow,\mathbf{ij}}} = 5\text{pu}$. This results in $L = 610$ for the CI model, and $\gamma = 48.5$ and $\rho = 0.0574$. The same step-sizes and parameters are used for the BF model as well in the following simulation results.

3.6.1 Qualitative Discussion on Simulation Results

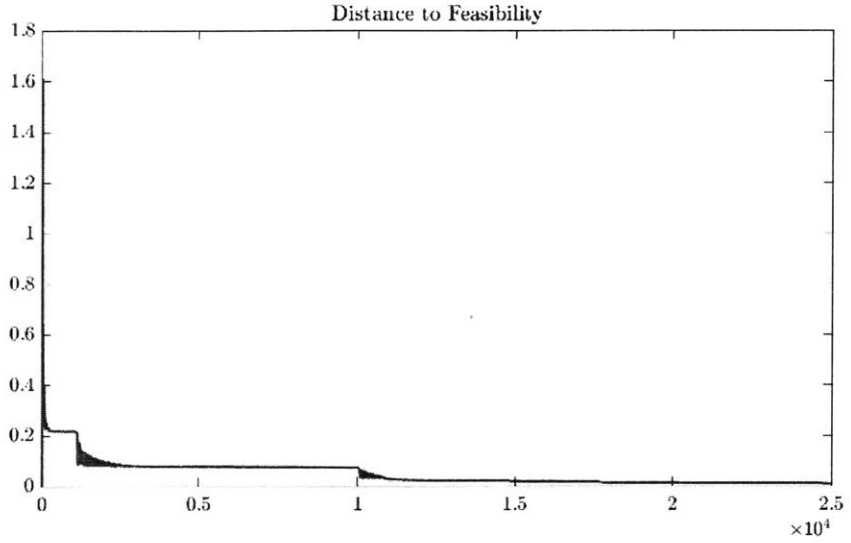
In what follows, we present results from the PAC algorithm applied to both the CI and BF models, and make both qualitative and quantitative analysis. The convergence results for the CI and BF models are shown in Figures 3-9 and 3-10 respectively, for the feasibility and consistency metrics.

From a first glance, the BF model looks to converge much faster and with smoother profile. The BF model approaches converges within 5000 iterations, while the CI model has comparable approach in a over $10,000$ iterations. This shows the BF model converging in about half of the number of iterations.

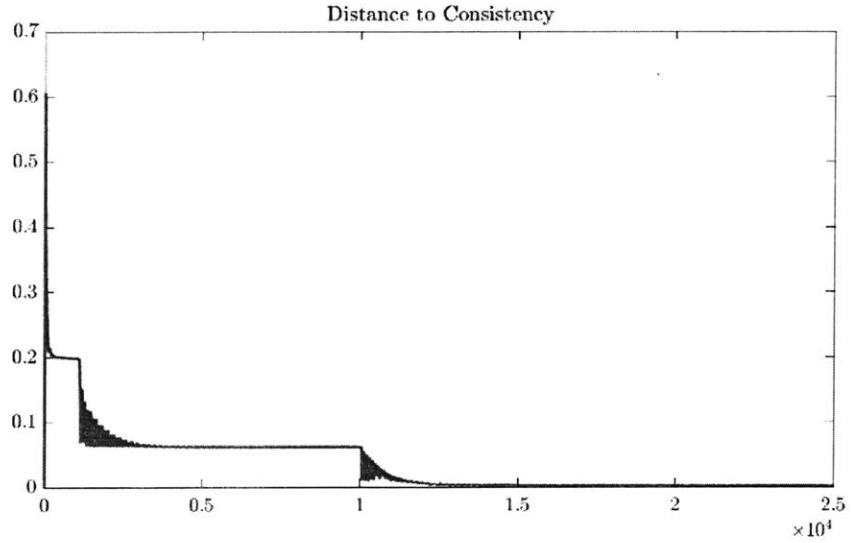
A close up of the first 100 iterations is shown in Fig. 3-11. Here we see that the BF model deviates further from feasibility and consistency as shown by the peak

[8] Lipschitz continuity describes the continuity of a function by bounding the slope of the function, or in other words, how quickly a function can change value. The Lipschitz parameter is any such that $|f(\mathbf{y_2} - f(\mathbf{y_1}))| \leq L|y_2 - y_1|$. The smallest constant is called the best Lipschitz constant. For a differentiable function, Lipschitz continuity means the function has a bounded derivative.

[9] The objective function is a simple convex function. As such, one would think $\alpha = 1$ is a suitable parameter. However, here we set it to a lower value to ensure stability and convergence in the algorithm. Further simulation results for $\alpha = 1$ are also presented after the discussion between CI and BF models.

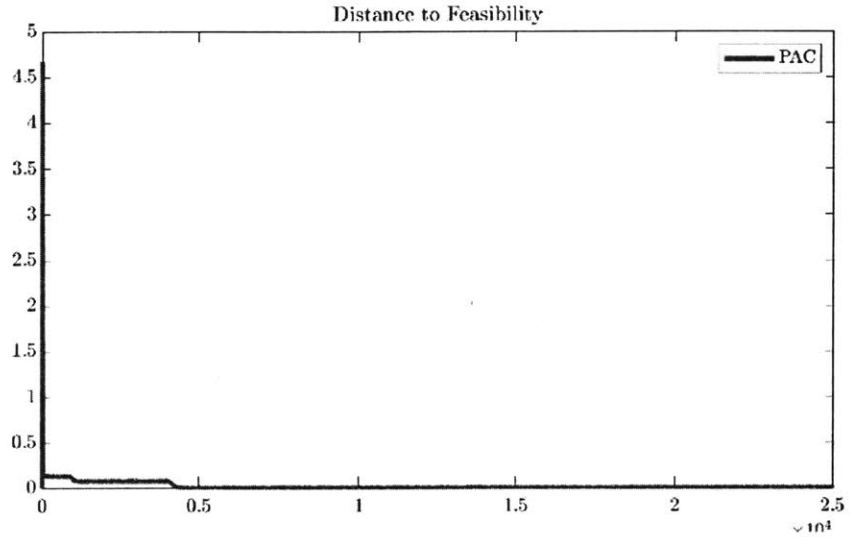


(a) Feasibility Metric for CI Model.

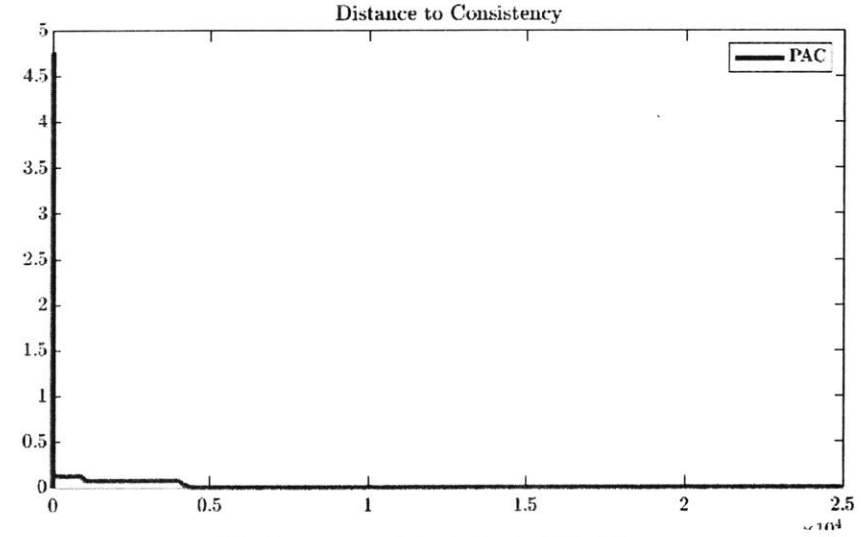


(b) Consistency Metric for CI Model.

Figure 3-9: Convergence results for the IEEE 13-bus network, under balanced conditions, for the CI model.

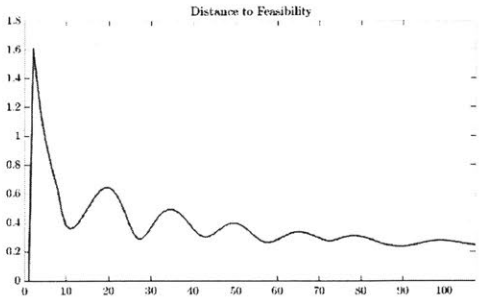


(a) Feasibility Metric for BF Model.

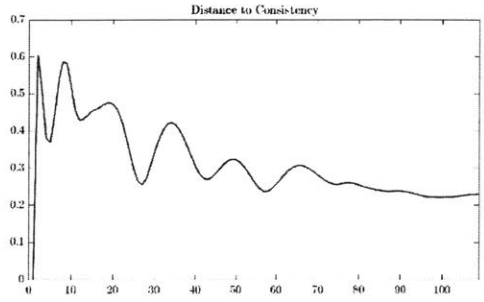


(b) Consistency Metric for BF Model.

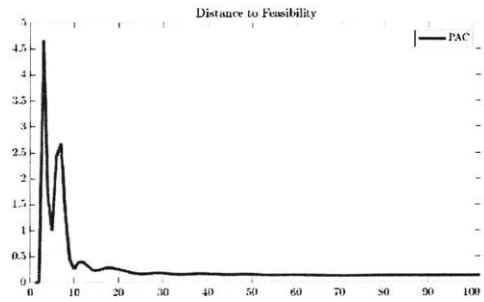
Figure 3-10: Convergence results for the IEEE 13-bus network, under balanced conditions, for the BF model.



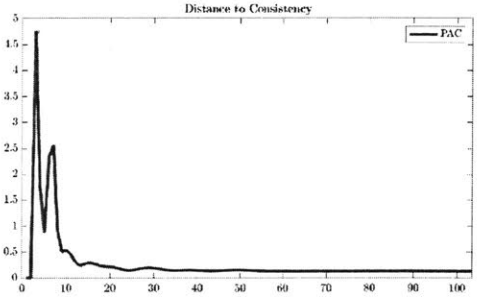
(a) Feasibility Metric for CI Model.



(b) Consistency Metric for CI Model.



(c) Feasibility Metric for BF Model.



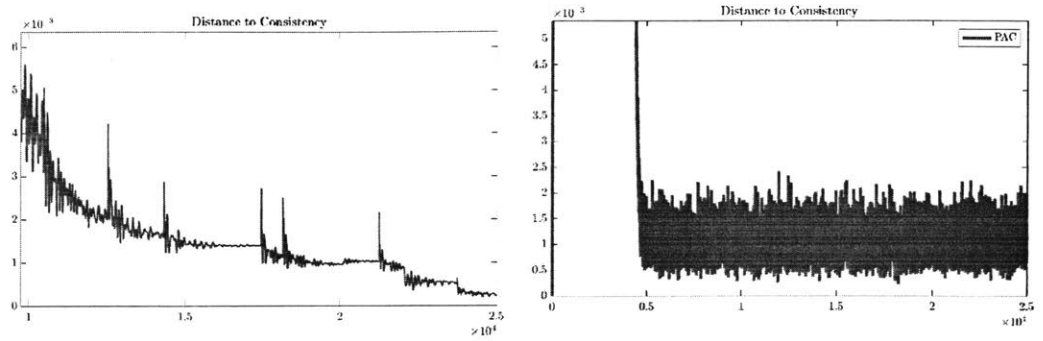
(d) Consistency Metric for BF Model.

Figure 3-11: Convergence results for the IEEE 13-bus network, zoomed in on the first 100 iterations to show the behaviour of the algorithm per each model.

values of roughly 4.8 for both metrics, in comparison to the peak of 1.6 and 0.6 for feasibility and consistency under the CI model. After the peak, both models show sinusoidal behaviour, with the CI model exhibiting fewer jumps. Although the BF model deviates further from the optimal solution, the two models reach comparable levels around the 100 iteration mark, with BF potentially achieving lower error here.

Another set of close up graphs are shown for the CI and BF models in Fig. 3-12, this time looking at the algorithm performance as it approaches convergence. here we see that the BF model does indeed approach convergence [10] before the CI model. However, this close up also gives us more information. Under the BF model, the PAC algorithm is unable to improve the solution below errors of $1e^{-3}$; under the CI model,

[10]Here we are loosely using the term convergence, as when errors are on the order of $1e^{-3}$. Typically convergence would be when errors are on the order of $1e^{-6}$, as the PAC algorithm does indeed converge to the exact global optimum. This is compared to other distributed algorithms such as dADMM which approaches the optimum, but does not reach it.



(a) Consistency Metric for CI Model.

(b) Consistency Metric for BF Model.

Figure 3-12: Convergence results for the IEEE 13-bus network, zoomed in on the first 100 iterations to show the behaviour of the algorithm per each model.

the algorithm continues to improve the solution with errors approaching $1e^{-4}$. Thus, the CI model has more numerical precision when solved with the PAC. When far from the optimal, the PAC exhibits sinusoidal behaviour as expected from the primal-dual dynamics for both models, but with higher amplitude oscillations in the CI model. When approaching the optimal solution, the algorithm is numerically limited with the BF model. Recall that when solving OPF, per unit analysis is commonly used to scale variables down for numerical stability. This means there is potential for some variables to take on very small values; this is potentially problematic for the BF model under PAC.

3.6.2 Quantitative Discussion on Simulation Results

The CI and BF models are compared under the distributed PAC implementation. As such, the focus is on computational time and iterations, rather than on the exact power flow solution. Quantitative metrics for comparison are outlined below.

1. **Primal Atomic Update:** The primal atomic optimization problem for the CI model is a linear program (LP), while for the BF model is a quadratically constrained program (QCP). Note that if the objective function is a quadratic cost function, the CI model is a quadratic program (QP) and the BF model is a quadratically constrained quadratic program (QCQP). The linear constraints

of the CI model should result in lower computational time per atomic primal update, as QPs are much easier to solve than QCQPs. However, the computational time also depends on how tight the MCE bounds are for the convex relaxation. The CI and BF models can be compared by looking at the maximum and average time taken per primal atomic update.

- Average time per primal atomic iteration, averaged across atoms: $\tilde{t}_{\text{atomic}} = \frac{1}{N}\sum_{\mathbf{j}}\{\frac{1}{K}\sum_k t_{\mathbf{j}}^k\}$, for iterations $k = 1, ..., K$ and nodes $\mathbf{j} \in N$, measured in milliseconds.
- Maximum time per primal atomic iteration: $\bar{t}_{\text{atomic}} = \max_{\mathbf{j}} \max_k t_{\mathbf{j}}^k$, for iterations $k = 1, ..., K$ and nodes $\mathbf{j} \in N$, measured in seconds.
- Average standard deviation of time per primal atomic update: $\tilde{\sigma}_{\text{atomic}} = \frac{1}{N}\sum_{\mathbf{j}}\{\sigma_{\text{atomic},\mathbf{j}}\}$ where $\sigma_{\text{atomic},\mathbf{j}}$ is the standard deviation of primal update time for atom $\mathbf{j}$ across all K iterations, measured in milliseconds.

2. **Number of iterations:** The number of iterations, K, required by the PAC algorithm depends on many different factors: the initial conditions, the choice of step-sizes ρ and γ, and the problem being solved [11]. Here the CI and BF models will be compared by fixing both the initial conditions and step-sizes to be the same and comparing number of iterations (K_{eq}), and by fixing only the initial conditions and allowing step-size to be determined optimally as per [20, 21] (K_{diff}).

These metrics are tabulated in Table 3.4 for the simulation results above, using the setup as described above with CI and BF models having the same initial conditions and convergence parameters (as calculated for the CI model), for all results except K_{diff}. For calculating K_{diff}, we estimate the Lipschitz parameter for the BF model objective function and re-calculate the convergence parameters. These values are all tabulated in Table 3.5, including the eigenvalues which influence the condition number.

[11] Note here that the problem consists of both the choice of model for the physical problem, as well as the objective function. Of course, fixing the objective function to be the same between the CI and BF models is necessary for comparison.

The key insights here are the time for primal update. We had expected the CI model to have shorter primal update times as it is a QP compared to the QCQP of the BF model. However, the two are comparable (both solved directly in Gurobi, a commercial optimization solver), with the CI model actually taking slightly longer. This may be attributed to the larger variable space and the coupling between the real and imaginary components of current and voltage. The CI model also takes more iterations than the BF model, and so overall would take more time to converge. However, the major difference here is that the CI model solves the power flow or OPF problem in full, including the voltage and current angles. In comparison, the BF model does not solve for the voltage angles; rather a post-processing step (more accurately, an angle recovery algorithm) is required to recover the full OPF solution. A more fair comparison here would be to setup the angle recovery algorithm and compare the time for the CI-OPF and BF-OPF with angle recovery. Further, more networks should be tested, including meshed networks as they can be convexified for the BF model. A better comparison would set up these models and then compare the performance of the models and the PAC algorithm.

From the data in Table 3.5 we see that the condition number for the CI-Z model is better than the BF model as well, however, we've estimated a higher L parameter for the CI model. This comes down to two main items:

- The value of ξ heavily influences the L parameter (along with the actual optimal objective cost). The per unit LMP for electricity (calculated as λ/SBase, where SBase is the base value for apparent power) is very large. In these simulations $\lambda^{P}_{j\#} = 150\$/\text{pu}, \lambda^{Q}_{j\#} = 50\$/\text{pu}$, which for a 3-phase system results in $L = 600$. The line resistances on the other hand are very small (order or 0.01pu), meaning the influence of the line losses is minimal to the optimization problem. For a higher weighting, the objective cost and L parameter would both be higher.

- The CI model has a higher L value because of the difference in model variables. The CI model models the real and reactive components of branch current, while the BF model considers only the squared magnitude of branch current. As such,

when taking the derivative of the objective function, the upper bounds on real and reactive components of branch power are needed for the CI model; the BF model does not require any information on the branch currents. Tighter bounds on the branch current estimates would give a better bound on L for the CI model, and could potentially result in improved performance.

Network	Metric	CI-Z Model	BF Model
IEEE 13 Node	$\bar{t}_{\text{atomic}}$, ms	3.0	2.2
	$\bar{t}_{\text{atomic}}$, s	0.1618	0.0177
	$\tilde{\sigma}_{\text{atomic}}$, ms	0.8	0.3
	K_{eq}	13000	5400
	K_{diff}		6300

Table 3.4: Comparison of the CI and BF model when using the distributed PAC algorithm.

Parameter	CI-Z Model	BF Model
L	610	600
γ	48.5	42.4
ρ	0.0574	0.0578
λ_{min}	0.0527	0.0544
λ_{max}	6.2439	7.0486

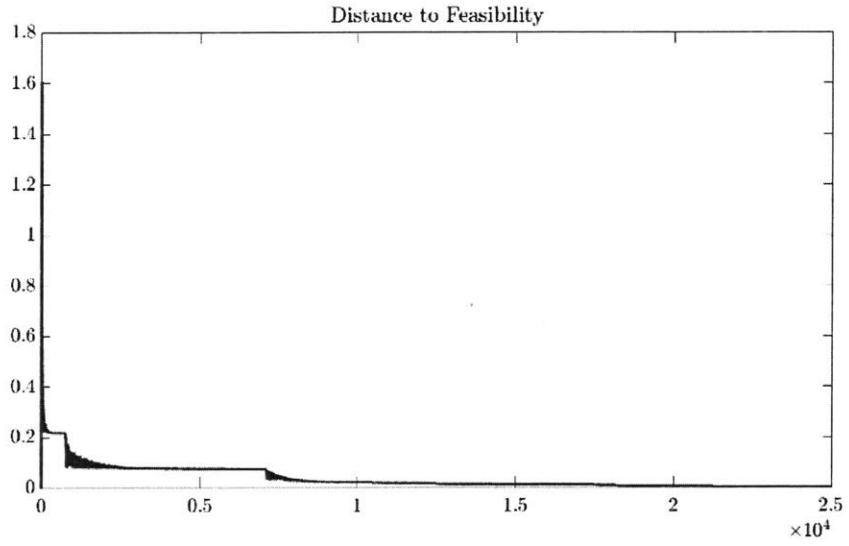
Table 3.5: Comparison of the CI and BF model when using the distributed PAC algorithm.

3.6.3 Additional Simulation Result

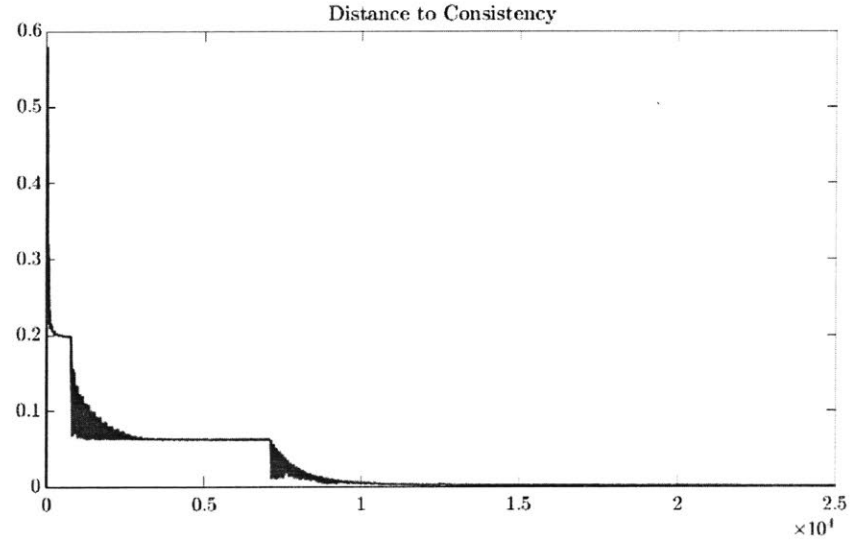
From the discussions above, we can consider different simulations setups. We simulate only the CI model and test the performance when $\alpha = 1$ is used (as the objective function is indeed convex) in Fig. 3-13 and $\xi = 10$ to weight the energy efficiency higher in Fig. 3-14. Here we see that having the higher α parameter does speed up convergence, and we get closer to optimal in under 10000 iterations, as compared to the case of $\alpha = 0.5$. The results for the more weighted objective function also shows better convergence. This may be attributed to the L value being higher ($L = 696$, and thus larger step-sizes of $\gamma = 55.47$ and $\rho = 0.0537$). This could also be as

a result of more variables influencing the optimality: the branch currents are more heavily weighted and so this may limit the solution space. Further, the branch current variables can be bound better: the OPF solution has $\overline{I^r_{flow,\mathbf{ij}}} = 0.2886\text{pu}, \overline{I^i_{flow,\mathbf{ij}}} = 1.2047e-4\text{pu}$. In the current problem setup, the branch current variables do not have pre-set limits in the formulation of the optimization problem [12], but using estimates from historical data can help to bound these variables, perhaps leading to smaller feasibility regions (and thus, faster convergence).

[12]However, note that this does not mean they are unbounded variables. Rather, the limits on power injections at nodes and the power flow equations themselves limit branch power flow, and thereby current flow. More specifically, the limits on branch current flow come primarily from line thermal and ampacity limits, which are a result of cable specifications, rather than power physics.

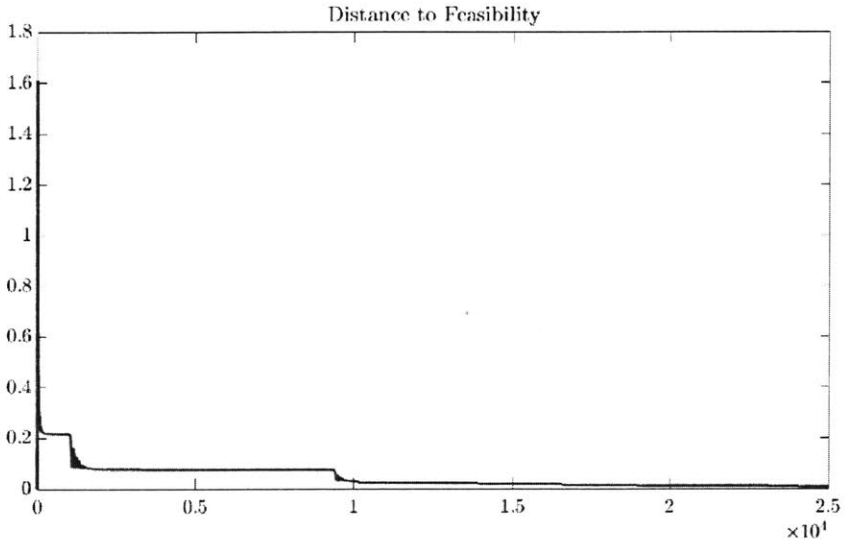


(a) Feasibility Metric for CI Model.

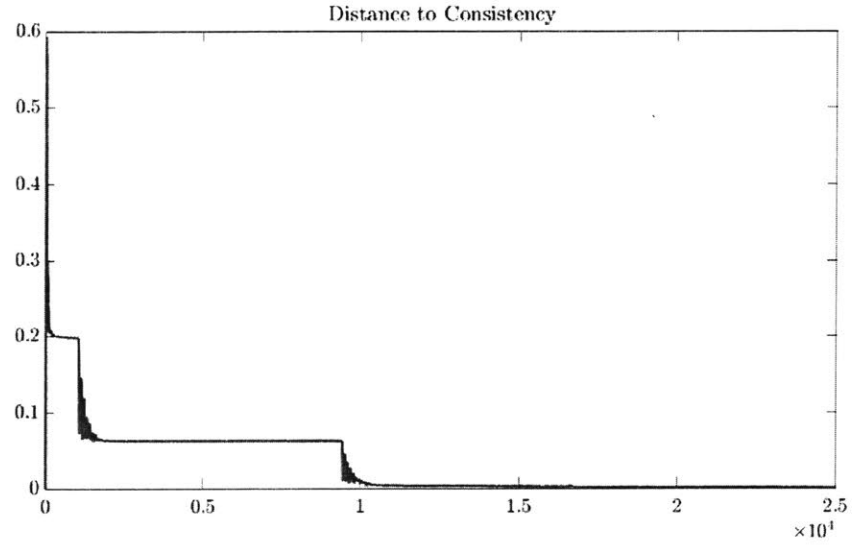


(b) Consistency Metric for CI Model.

Figure 3-13: Convergence results for the IEEE 13-bus network, under balanced conditions, for the CI model, with $\alpha = 1$, which influences the convergence parameters.



(a) Feasibility Metric.



(b) Consistency Metric.

Figure 3-14: Convergence results for the IEEE 13-bus network, under balanced conditions, for the CI model, with higher weight for the electrical losses term, $\xi = 10$.

Chapter 4

A Story of Storage: Multi-period Distributed Optimization for Distribution Grids

This chapter discusses the extended application of the PAC algorithm to address multi-period optimization problems. In previous works employing the PAC algorithm [20–22], all decision variables were assumed to have no state retention, lending the problem to single period optimization. However, in the case of distribution grids with a high penetration of DERs, and in particular storage devices, there is temporal coupling between constraints. Using a storage unit during one hour lowers the state of charge, and thus future available capacity. The extension for multi-period optimization was evaluated in the context of DER scheduling and dispatch, primarily for day-ahead decisions.

4.1 Multi-period Optimization

Consider the following multi-period optimization problem (4.1), where $\mathbf{y} \in \mathbb{R}^{(Y \times T) \times 1}$ are the Y optimization variables for each of the T periods, subject to M optimization

constraints written in standard form.

$$\min_{\mathbf{y}\in\mathbb{R}^{Y\times T}} \left\{ f(\mathbf{y}) \triangleq \sum_{\mathbf{k}\in[\mathcal{B}]} \mathbf{f_k}(\mathbf{y}) \right\} \quad \text{subj. to: } \mathbf{G}\mathbf{y} = \mathbf{0}_{\mathrm{M}} \tag{4.1}$$

For a multi-period optimization problem over T periods, the decision vector $\mathbf{y}$ consists of repeated units describing the system at each time instance of interest:

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_{y=1} \\ \mathbf{y}_{y=2} \\ \vdots \\ \mathbf{y}_{y=Y} \end{bmatrix} \quad \text{where } \mathbf{y}_y = \begin{bmatrix} y_{y,t=1} \\ y_{y,t=2} \\ \vdots \\ y_{y,t=T} \end{bmatrix} \tag{4.2}$$

Likewise, the constraints $\mathbf{G}\mathbf{y} = \mathbf{0}_{\mathrm{M}}$ can include a number of constraints linking decision variables temporally $\mathbf{G_t} \subseteq \mathbf{G}$; these are described as $\mathbf{G_t}\mathbf{y} = \mathbf{0}_{|\mathbf{G_t}|}$, where $|\cdot|$ denotes the dimension of the vector.

Following the decomposition profile for the PAC algorithm, introduced in Section 3.4, the global multi-period optimization problem can be decomposed into smaller subproblems. Note that each of the constraints in $\mathbf{G_t}$ can relate multiple variables across time steps, regardless of which atoms the variables belong to, as the decomposition profile makes use of variable copies to fully distribute the problem. Likewise, the objective function $f_{S_\mathbf{j}}(\mathbf{y})$ can depend on variables across time steps, from any atom. This is because each atom $\mathbf{j}$ retains a "copy" of the dependent variables from corresponding atom $\mathbf{i}$ for all required time periods, $\forall t \in T$.

The atomized equivalent of (4.1) can then be obtained, where $\mathbf{B}$ represents the adjacency matrix of the directed graph of the atoms (as introduced in Section 3.4):

$$\begin{aligned} &\min_{\mathbf{a}\in\mathbb{R}^{|\mathbf{a}|}} \sum_{\mathbf{j}\in\mathcal{B}} f_\mathbf{j}(\mathbf{a_j}) \\ &\text{subj. to:} \\ &\tilde{\mathbf{G}}_\mathbf{j}\mathbf{a_j} = \mathbf{0}, \quad \mathbf{j}\in\mathcal{B} \\ &\mathbf{B}_{\mathbf{j},-}\mathbf{a} = \mathbf{0}, \quad \mathbf{j}\in\mathcal{B} \end{aligned} \tag{4.3}$$

where $\mathbf{B_{j,-}}$ represents the rows of $\mathbf{B}$ that correspond to the coupling variables along $(\mathbf{i},\mathbf{j}) \in \mathcal{T}_\mathrm{D}$. The equation $\tilde{\mathbf{G}}_\mathbf{j}\mathbf{a_j} = \mathbf{0}$ corresponds to the atomized equality constraints from the global problem (4.1). The atomized decision vectors $\mathbf{a_j} \in \mathcal{R}^{(|T_\mathrm{j} \times T|)\times 1}$ can be expanded to show temporal groups, like the decision vector of the $\mathbf{y}$ (4.2):

$$\mathbf{a_j} = \begin{bmatrix} \mathbf{a_{j}}_{a=1} \\ \mathbf{a_{j}}_{a=2} \\ \vdots \\ \mathbf{a_{j}}_{a=|T_\mathrm{j}|} \end{bmatrix} \quad \text{where } \mathbf{a_{j}}_{a} = \begin{bmatrix} a_{j\,a,t=1} \\ a_{j\,a,t=2} \\ \vdots \\ a_{j\,a,t=T} \end{bmatrix} \tag{4.4}$$

For a single period optimization problem, we clearly lay out the copied and owned variables. Note that these follow for the multi-period formulation as well, with copies of a variable made for every time period.

$$\mathbf{a_j} = \begin{bmatrix} a_{j_1} \\ a_{j_2} \\ \vdots \\ a_{j_{|L_j|}} \\ \hat{a}_{i_1} \\ \vdots \\ \hat{a}_{i_{|O_j|}} \end{bmatrix} \begin{matrix} \left.\vphantom{\begin{matrix} a \\ a \\ \vdots \\ a \end{matrix}}\right\} \text{Owned variables} \\ \left.\vphantom{\begin{matrix} a \\ \vdots \\ a \end{matrix}}\right\} \text{Copied variables} \end{matrix} \tag{4.5}$$

4.2 Application to Distribution Grid

The multi-period extension of the PAC algorithm can now be applied to the problem of day-ahead scheduling of resources within the distribution grid under high penetration of DERs. By using the distributed PAC algorithm, the dispatch variables resulting in lowest cost operation can be determined. Specifically, these variables include power injection $(P_\mathbf{j}, Q_\mathbf{j})$ [1] and storage control $(P^{\mathrm{sc}}_\mathbf{j}, P^{\mathrm{sd}}_\mathbf{j})$ at each bus, for all time steps $t \in [1, ...T]$.

The OPF formulation using the CI model introduced in Chapter 3 is used, as

[1] Recall that for the CI model, we define power injection as positive P and Q, and loads as negative P and Q.

the distribution grids are highly unbalanced and can contain meshed topologies. The model is extended to account for additional time-coupling constraints, and objective function for the unit commitment problem. Specifically, the storage capacity and ramp rates, and generator ramp constraints introduce states that persist across time periods. Thus, including these constraints requires the solution of multi-period OPF (MOPF), also called dynamic OPF, with the constraints coupling variables across different time periods [33–35].

The formulation for modeling the DER devices is discussed here, with a focus on the multi-period constraints in the last two subsections. For notational simplicity, we discuss the formulation for a single-phase network, thus removing the subscripts pertaining to the phase; extension to multi-phase network follows accordingly from the model in Chapter 2.

4.2.1 Demand Response

Loads are modelled as negative power injections, as:

$$\underline{P_{\mathbf{j}}} \leq P_{\mathbf{j}} \leq 0 \tag{4.6}$$

Note that the above formulation implies demand response capability at node $\mathbf{j}$, to 100% of base load, $\underline{P_{\mathbf{j}}}$. For a fixed load (no demand response capabilities), the power injection is fixed as per (4.7). For a load with demand response capabilities of ζ percent, the power injection is variable as per the bounds in (4.8).

$$\underline{P_{\mathbf{j}}} \leq P_{\mathbf{j}} \leq \underline{P_{\mathbf{j}}} \tag{4.7}$$

$$\underline{P_{\mathbf{j}}} \leq P_{\mathbf{j}} \leq (1-\zeta)\underline{P_{\mathbf{j}}} \tag{4.8}$$

$$\underline{Q_{\mathbf{j}}} \leq Q_{\mathbf{j}} \leq \underline{Q_{\mathbf{j}}} \tag{4.9}$$

In this model, we consider demand response only for real power consumption, thus reactive power is fixed as per (4.9). We note, however, that with increase in power electronically controlled loads, and motor-based loads (i.e. induction motors), the

demand response is also enable on reactive power. To include this, the reactive power bounds will follow a similar form to (4.8), with additional constraints likely required to model the relationship between P_{j} and Q_{j}.

4.2.2 Distributed Generators

Generators are modelled as power injections with both real and reactive power capabilities. Generators can be categorized as controllable and uncontrollable resources. Controllable generators include traditional diesel generators and motors, where the power output can be entirely controlled. The only limiting factors are availability of fuel (which we can assume is not an issue), and the ramping up/down capabilities (which will be discussed later). Uncontrollable generators are renewable generators, where the power output is entirely dependent on the availability of solar or wind resources. Note that renewable energy can be curtailed in most cases - production beyond what is needed can be stored or curtailed (i.e. thrown away as waste), thus the lower bound of generation is assumed to be flexible. Distributed generation can be modelled as in (4.13) for controllable loads, and (4.11) for uncontrollable (renewable) resources, where ζ is the amount of allowable curtailment. The value of ζ is determined largely by policies such as renewable energy standards or requirements: a grid operator may be required to utilize all renewable energy generation and so $\zeta = 0$, while another may have more flexibility in resource use so $0 \leq \zeta \leq 1$.

$$\underline{P_{\mathrm{j}}} \leq P_{\mathrm{j},t} \leq \overline{P_{\mathrm{j}}} \tag{4.10}$$

$$(1-\zeta)\overline{P_{\mathrm{j}}, t} \leq P_{\mathrm{j},t} \leq \overline{P_{\mathrm{j},t}} \tag{4.11}$$

$$\underline{P_{\mathrm{j}}} \geq 0 \tag{4.12}$$

The values of $\overline{P_{\mathrm{j},t}}$ and $\underline{P_{\mathrm{j},t}}$ for controllable generators are determined by the technical specifications of the generator. The value of $\overline{P_{\mathrm{j},t}}$ for renewable resources is determined by weather forecasts and historical data which estimate the generation capabilities

of the resource. These are assumed to be inputs to the OPF problem, and thus this thesis does not focus on the forecasting of renewable energy generation. The same can be said for the reactive power from the generators. Here for simplicity we assume that only non-renewable generators can provide reactive power, and that this reactive power can also be negative as it can be power electronically controlled. For renewable resources, we assume $Q_{\mathrm{j}} = 0$.

$$\underline{Q_{\mathrm{j}}} \leq Q_{\mathrm{j},t} \leq \overline{Q_{\mathrm{j}}} \tag{4.13}$$
$$\overline{Q_{\mathrm{j}}} \geq 0$$

4.2.3 Generator Capability Curves

Another important constraint for generators (both controllable and uncontrollable) is the capability curve, which dictates the relationship between P_{j} and Q_{j}. The capability curve details the safe and desirable regions of operation, which can include (but not limited to) maximum current, maximum power, and power factor relationships, as shown in Fig. 4-1. To keep a simple formulation, here we only consider the power factor for the capability curve. For renewable resources, $\mathrm{pf} = 1$ already as we assume $Q_{\mathrm{j}} = 0$. However, for nonrenewable generators, the power factor should be limited to ensure efficient operation. Recall that power factor is $\mathrm{pf} = \cos(\arctan(\frac{Q_{\mathrm{j}}}{P_{\mathrm{j}}}))$. Then for a range of power factors (giving positive and negative Q_{j}, the capability curve can be modelled as:

$$P_{\mathrm{j}} \tan(\arccos(\mathrm{pf}_{\mathrm{neg}})) \leq Q_{\mathrm{j}} \leq P_{\mathrm{j}} \tan(\arccos(\mathrm{pf}_{\mathrm{pos}})) \tag{4.14}$$

where $\mathrm{pf}_{\mathrm{neg}}$ is the negative power factor giving $Q_{\mathrm{j}} \leq 0$, and $\mathrm{pf}_{\mathrm{pos}}$ is the positive power factor giving $Q_{\mathrm{j}} \geq 0$.

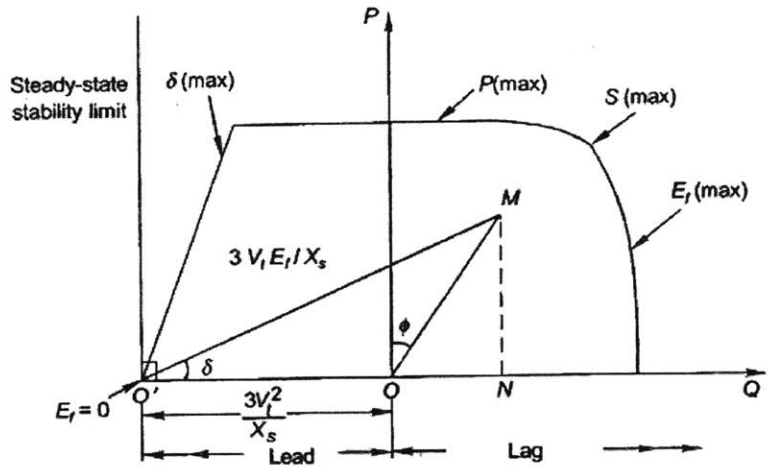


Figure 4-1: An example generator capability curve for a synchronous generator. Here additional limitations for currents of field and excitor windings are also shown, giving the curve a more complex shape. For the simple power factor bounds we are considering, the curve will take on a smooth circular shape for each quadrant of operation. Recall that $P_{\mathbf{j}} \geq 0$ for all generators, so we only have 2 quadrants of operation.

4.2.4 Storage System Modeling

In order to extend the proposed CI model to include storage resources, we introduce additional variables, $P^{\text{sd}}_{\mathbf{j},t}$ and $P^{\text{sc}}_{\mathbf{j},t}$, which are the discharge and charge power for storage unit at node $\mathbf{j}$ at time t respectively. These are related to the net power injection at node $\mathbf{j}$ as:

$$P_{\mathbf{j}} = \frac{1}{\eta^{\text{D}}_{\mathbf{j}}} P^{\text{sd}}_{\mathbf{j},t} - \eta^{\text{C}}_{\mathbf{j}} P^{\text{sc}}_{\mathbf{j},t} \tag{4.15}$$

For notational simplicity we assume that all storage units operate at unity power factor, so $Q_{\mathbf{j},t} = 0 \forall t$ [2]. The limitations on charge and discharge power can then be specified as:

$$0 \leq P^{\text{sd}}_{\mathbf{j},t} \leq \overline{P^{\text{sd}}_{\mathbf{j}}} \tag{4.16}$$

$$0 \leq P^{\text{sc}}_{\mathbf{j},t} \leq \overline{P^{\text{sc}}_{\mathbf{j}}} \tag{4.17}$$

[2]Many storage devices can actually be used to provide reactive power support as power electronics can be used very easily to adjust power factor. In this work, we simplify the case and use storage to either store excess renewable generation, energy arbitrage, or as reserve.

Now we are ready to add the capacity constraints on the storage units. These are integral constraints, and require knowledge of the system's current state of charge (SOC) $b_{\mathrm{j},t}$, as described by the previous SOC $b_{\mathrm{j},t-1}$ minus losses due to self-discharge (written as $\eta_{\mathrm{j}}^{\mathrm{self}}$), and the net charging during the period t, while also accounting for charging and discharging efficiency ($\eta_{\mathrm{j}}^{\mathrm{C}}$ and $\eta_{\mathrm{j}}^{\mathrm{D}}$ respectively). This temporal coupling is described as:

$$b_{\mathrm{j},t} = (1 - \eta_{\mathrm{j}}^{\mathrm{self}}) b_{\mathrm{j},t-1} + \eta_{\mathrm{j}}^{\mathrm{C}} P_{\mathrm{j},t}^{\mathrm{sc}} - \frac{1}{\eta_{\mathrm{j}}^{\mathrm{D}}} P_{\mathrm{j},t}^{\mathrm{sd}} \tag{4.18}$$

Note the SOC of a storage device at time period t is a dependent property, given knowledge of the initial SOC $b_{\mathrm{j},0}$ and the actuation over each prior period $t = [1, ..., t-1]$. Thus $b_{\mathrm{j},t}$ need not be a decision variable in our optimization problem; rather, it can be calculated based on the charge and discharge over each period, $P_{\mathrm{j},t}^{\mathrm{sc}}$ and $P_{\mathrm{j},t}^{\mathrm{sd}}$ respectively. The capacity constraints for the storage device is therefore

$$\underline{b_{\mathrm{j}}} \leq b_{\mathrm{j},t} \leq \overline{b_{\mathrm{j}}}$$

where

$$b_{\mathrm{j},t} = \underbrace{(1 - \eta_{\mathrm{j}}^{\mathrm{self}} \Delta t)^t b_{\mathrm{j},0}}_{\substack{\text{Self discharge} \\ \text{of initial SOC}}} + \underbrace{\sum_{\tau}^{t-1} ((1 - \eta_{\mathrm{j}}^{\mathrm{self}} \Delta t)^{t-\tau} [\eta_{\mathrm{j}}^{\mathrm{C}} P_{\mathrm{j},t}^{\mathrm{sc}} - \frac{1}{\eta_{\mathrm{j}}^{\mathrm{D}}} P_{\mathrm{j},t}^{\mathrm{sd}}])}_{\text{Self discharge of net charge over each period}} + \underbrace{\eta_{\mathrm{j}}^{\mathrm{C}} P_{\mathrm{j},t}^{\mathrm{sc}} - \frac{1}{\eta_{\mathrm{j}}^{\mathrm{D}}} P_{\mathrm{j},t}^{\mathrm{sd}}}_{\substack{\text{Net charge over} \\ \text{current period}}}$$

where the last two terms can be combined, and the upper bound on the summand becoming t. Note that in most battery systems, a minimum non-zero SOC is desirable at all times to keep the battery in good health - this is referred to as maximum depth of discharge (maxDOD); thus $\underline{b_{\mathrm{j}}} = (1 - \text{maxDOD}) \overline{b_{\mathrm{j}}}$. Typically, this is dependent on the electrochemistry of the battery. This may be on the order of $30 - 50\%$ (so maximum depth of discharge is $50 - 70\%$) to avoid damage to the storage system from excessive discharge [36], but can also be as low as 10% for vanadium flow chemistry. The lower limit can also be a technical requirement of the grid, as a contingency to ensure there

is power available in case of loss of generator [3]. Additionally, an ending capacity requirement is typically employed in the day-ahead unit commitment problem [33], in order to recharge the battery for the next day of operation. This adds the following constraint:

$$b_{\mathbf{j},0} = b_{\mathbf{j},T}$$

Note that the SOC constraint is for the entire device, not per phase as the other power constraints have been. The SOC constraint can be extended for a multi-phase system by summing $P^{\text{sc}}_{\mathbf{j},t}$ and $P^{\text{sd}}_{\mathbf{j},t}$ over the phases at the node $\mathbf{j}$, and aggregating it to one value per timestep t.

4.2.5 Ramp Constraints

Another set of necessary temporal constraint are ramp constraints on generators and storage devices[4]. These are especially important for generators such as diesel generators which have non-negligible startup and shutdown times. Note that renewable resources typically lack significant ramp constraints, further lending themselves to rapid variability. As such, the ramp rates on renewables are not modelled.

These constraints require knowledge of the power output of the previous time period, and only couple pairs of consecutive times. With ramp limits typically given in kW/h. For period t, the ramp constraint is modeled as (4.19) for both generators and storage units.

$$\underline{r_{\mathbf{j}}}\Delta t \leq P_{\mathbf{j},t} - P_{\mathbf{j},t-1} \leq \overline{r_{\mathbf{j}}}\Delta t \tag{4.19}$$

where $\underline{r_{\mathbf{j}}}$ and $\overline{r_{\mathbf{j}}}$ represent the decrease and increase ramp rate limits for the device at bus $\mathbf{j}$.

[3]This is a form of an ancillary service. Energy reserves are important for the reliability of the grid. This will be further discussed in Chapter 5, in the context of market derivatives for services.

[4]Ramp constraints on electrical storage systems are typically only relevant for sub-second timescales, as they lack the limiting inertia that systems like flywheels posses.

4.2.6 Defining Cost Function

The day-ahead unit commitment problem aims to minimize the total costs incurred by the system. This typically includes the cost to generate power, and power losses across the distribution grid lines. The cost function, $f^{\text{MOPF}}(\mathbf{y})$, is:

$$f_{\mathbf{j}}^{\text{Import}}(\mathbf{y}) = \lambda_{\mathbf{j}}^{\text{P}} P_{\mathbf{j}} + \lambda_{\mathbf{j}}^{\text{Q}} Q_{\mathbf{j}} \qquad \text{if } \mathbf{j} \in \mathcal{F} \tag{4.20}$$

$$f_{\mathbf{j}}^{\text{Local-Gen}}(\mathbf{y}) = a_{\mathbf{j}}^{\text{P}} P_{\mathbf{j}}^2 + b_{\mathbf{j}}^{\text{P}} P_{\mathbf{j}} + c_{\mathbf{j}}^{\text{P}} \qquad \text{if } \mathbf{j} \in \mathcal{B} \tag{4.21}$$

$$f_{\mathbf{j}}^{\text{Storage}}(\mathbf{y}) = \gamma_{\mathbf{j}}^{\text{S}} |P_{\mathbf{j}}| \qquad \text{if } \mathbf{j} \in \mathcal{B} \tag{4.22}$$

$$f^{\text{OPF}}(\mathbf{y}_t) = \sum_{\mathbf{j}} \left[f_{\mathbf{j}}^{\text{Import}}(\mathbf{y}_t) + f_{\mathbf{j}}^{\text{Local-Gen}}(\mathbf{y}_t) + f_{\mathbf{j}}^{\text{Storage}}(\mathbf{y}_t) \right] \qquad \forall t \in T \tag{4.23}$$

$$f^{\text{MOPF}}(\mathbf{y}) = \sum_{t} f^{\text{OPF}}(\mathbf{y}_t) \tag{4.24}$$

where

- $\lambda_{\mathbf{j}^{\#}}^{\text{P}}$ and $\lambda_{\mathbf{j}^{\#}}^{\text{Q}}$ are the locational marginal price (LMP) of electricity, for each feeder $\mathbf{j}^{\#}$, $\mathbf{j} \in \mathcal{F}$, as measured in \$/pu;

- $a_{\mathbf{j}}^{\text{P}}, b_{\mathbf{j}}^{\text{P}}, c_{\mathbf{j}}^{\text{P}}$ are quadratic cost terms for the marginal cost of locally generating power, as measured in \$/MW. These costs typically include fuel costs and any marginal Operation and Maintenance (O&M) for the non-renewable resources. Note that renewable distributed generators have no marginal cost of operation;

- $\gamma_{\mathbf{j}}^{\text{S}}$ is the marginal cost of using storage, as measured in \$/MWh. This cost is based on the battery wear cost: $\text{c}_{\text{bw}} = \frac{\text{c}_{\text{rep}}}{\text{Q}_{\text{life}} \eta_{\text{rt}}} = \frac{\text{replacement cost [\$]}}{\text{lifetime throughput [MWh]} \times \text{roundtrip efficiency}}$, where $\text{Q}_{\text{life}} = \text{No. of Cycles} \times \overline{b_{\mathbf{j}}} \times \text{DOD} \times 2\frac{\text{trips}}{\text{cycle}}$.

4.2.7 Electricity LMP Data

The price of electricity for a region (i.e. consider a single distribution grid) is determined by the wholesale electricity market (WEM) and is termed the locational marginal price (LMP). The WEM solves an optimization problem to determine the price of electricity, which is the price at which load serving entities (LSE) such as

utilities purchase electricity and then sell to its customers at a retail price. The LMP varies between different regions, reflecting operating characteristics and major constraints on the transmission system, as well as losses resulting from the physical transmission grid [5].

The LMP for the New England area are published online on the Independent System Operator (ISO-NE) website [37], and is used in this thesis to determine $\lambda^{P}_{j\#}$; specifically the 5-minute real-time LMP data for May 14 2019 is queried. The 5-minute data is averaged over 12 periods to give hourly LMP data. See Appendix B.5 for the hourly LMP data. The "LMP" for reactive power, $\lambda^{Q}_{j\#}$ is not a concept currently used in electricity markets; however, to show the potential of scheduling real and reactive power with enhanced control in smart grids using DERs, we estimate $\lambda^{Q}_{j\#} = 0.1\lambda^{P}_{j\#}$. The LMP data is provided in \$/MWh, and is converted to per unit analysis (\$/pu) as:

$$\lambda^{P}_{j\#} = \lambda^{P}_{j\#_{\$/MWh}} \frac{SBase}{1e3} \tag{4.25}$$

where SBase is the base value used for apparent power, to convert to per unit analysis.

Note that ISO-NE also uploads hourly day-ahead LMP prices, preliminary real-time 5-min and hourly LMPs, and the actual real-time 5-min and hourly LMPs. Thus, different simulations can be carried out to better understand the pricing data required for best scheduling of DERs. This is outside the scope of this thesis, as the focus is not on information availability for decision making, but is an interesting point worth mentioning.

4.2.8 Electricity Demand Profiles

The IEEE test feeders all contain single period load data. In order to consider more interesting and realistic problems with these test topologies, the provided load data is treated as a base value, upon which a demand profile is used to generate multi-

[5]The concept of the LMP and the structure of the US electricity market is discussed in more detail in Chapter 5. This includes defining what the ISO is as a market and regulatory entity.

period load forecasts. A normalized base demand profile is obtained from the ISO-NE reports on total recorded electricity demand for each five-minute interval [38], for real power. As with the LMP data, the report is queried for May 14, 2019. The demand data is averaged over 12 periods to give hourly demand data; see Appendix B.6 for data.

This base demand profile is used to generate demand profiles for all N nodes in the network. This is done to introduce some complexity into the model; rather than assuming that all nodes follow the same demand profile, we perturb the base profile and obtain more variability, as would be expected in a real load forecast. Let $\alpha(t)$ refer to the base profile obtained directly from the ISO-NE report data. Then the nodal real power demand profile for node $\mathbf{j}$, $\alpha_{P,\mathbf{j}}(t)$, is obtained by introducing a factor of $\delta_{P,\mathbf{j}}$. This factor $\delta_{P,\mathbf{j}}$ is a vector of normally distributed random variables, such that $\delta_{P,\mathbf{j}} \sim \mathcal{N}(0, 0.1)$ is of dimension T, where T is the number of time periods to be simulated. Then $\alpha_{P,\mathbf{j}}(t) = \alpha(t)(\mathbf{1} + \delta_{P,\mathbf{j}})$, resulting in every element of $\alpha(t)$ being perturbed. This new profile is then smoothed using Gaussian filters to produce a realistic demand profile. An example set of profiles is shown in Fig. 4-2, for a network with 5 nodes, over a period of 24 hours.

It is assumed that reactive power consumption is related to real power consumption (i.e. they are not independent values). To introduce additional complexity, normalized demand profiles for reactive power are also generated, using the nodal demand profiles. A perturbation vector composed of normally distributed random variables, $\delta_{Q,\mathbf{j}} \sim \mathcal{N}(0, 0.01)$, is used, of dimension T, where T is the number of time periods to be simulated. The resulting reactive power profile is $\alpha_{Q,\mathbf{j}}(t) = \alpha_{P,\mathbf{j}}(t)(\mathbf{1} + \delta_{Q,\mathbf{j}})$.

4.2.9 Renewable Generation Forecasts

Renewable generators modelled in this work include distributed residential and commercial PV, and wind farms. The NREL's System Advisory Model (SAM) tool was utilized to obtain hourly generation forecasts for three renewable generator classes, using the "Photovoltaics (detailed)" class. Wind generation data is also obtain from

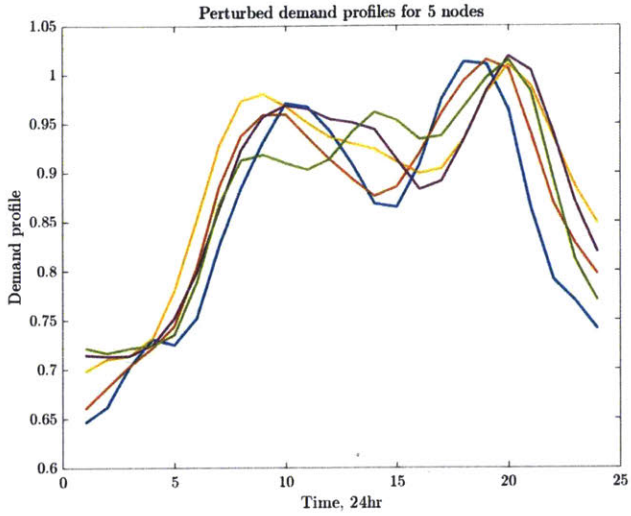


Figure 4-2: Example demand profiles for a network with 5 nodes.

SAM, for a 600kW onshore wind farm. Generation profiles are queried for May 14 [6]. See Appendix B.7 for the hourly generation profiles.

- *Small scale distributed residential PV:* This model is valid for up to 10kW residential PV resources. Resource data for Phoenix, AZ [7], using the SunPower SPR-X21-335 module, and SMA America (SB3800TL-US-22, 240V) inverter. The DC to AC ratio is set to the default of 1.2. The desired array size is set to 4kWdc, giving a total AC capacity of 3.850kWac, and uses 1 inverter. All other parameters are left unchanged.

- *Large scale distributed residential PV:* This model is valid for resources larger than 10kW residential PV. Resource data for Phoenix, AZ, using the SunPower SPR-X21-335 module, and SMA America (SB3800TL-US-22, 240V) inverter. The DC to AC ratio is set to the default of 1.2. The desired array size is set to 12kWdc, giving a total AC capacity of 11.550kWac, and uses 3 inverters. All other parameters are left unchanged.

- *Distributed commercial PV:* Resource data for Phoenix, AZ, using the Sun-

[6]Note that SAM generation data is hourly data for the entire year. Thus, seasonal variations can be readily captured using these profiles. The date of May 14 is chosen simply to align with the ISO-NE LMP and demand data being utilized.

[7]The generation data is only available for a few states, predominantly California.

Power SPR-E19-310-COM module, and SMA America (STP 60-US-10, 400V) inverter. The DC to AC ratio is set to the default of 1.2. The desired array size is set to 220kWdc, giving a total AC capacity of 179.580kWac. All other parameters are left unchanged.

Financial information for the resources is also included in the renewable generator model. The capital cost and installation costs for the solar resources are obtain from the default financial data in SAM: capital expenses (CAPEX) include the module costs, inverter costs, and balance of system equipment; the installation expenses include the installation labour, installer margin and overhead, and additional indirect costs which consist of environmental studies and permitting. These values are given in $. The annual operating and maintenance costs for solar obtained from the Annual Technology Baseline report by NREL [39]. This is typically $25/kW/yr but can be upwards of $40/kW/yr; a conservative value of $30/kW/yr is used for all solar resources.

Financial information for the wind resource is also obtained from the NREL report. We consider data for onshore wind for Techno-Resource Group (TRG)6 which has average wind speed of 6.9m/s [8]. The financial data lumps capital costs and installation costs for an average total of $1640/kW [39]. The annual operating and maintenance costs are averaged to be $51/kW [39].

The NREL report also provides average capacity factors of 0.2 for solar and 0.4 for onshore wind [39].

[8]The average desired wind speed for wind resources in New England is around 6.5m/s.

4.2.10 Statement of MOPF Problem

The centralized problem is formally stated as:

$$\min_{\mathbf{y}} \quad f^{\text{MOPF}}(\mathbf{y})$$

$$\text{s.t.} \quad I = \mathbf{A}^T I_{\text{flow}} \tag{4.26}$$

$$\mathbf{A}V = \mathbf{Z}I_{\text{flow}} \tag{4.27}$$

$$I_{\mathbf{j}} = I_{\mathbf{j}}^r + jI_{\mathbf{j}}^i \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.28}$$

$$V_{\mathbf{j}} = V_{\mathbf{j}}^r + jV_{\mathbf{j}}^i \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.29}$$

$$P_{\mathbf{j}} = a_{\mathbf{j}} + b_{\mathbf{j}} \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.30}$$

$$Q_{\mathbf{j}} = -c_{\mathbf{j}} + d_{\mathbf{j}} \qquad \forall j \in \mathcal{B} \tag{4.31}$$

$$\underline{I_{\mathbf{j}}^r} \leq I_{\mathbf{j}}^r \leq \overline{I_{\mathbf{j}}^r} \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.32}$$

$$\underline{I_{\mathbf{j}}^i} \leq I_{\mathbf{j}}^i \leq \overline{I_{\mathbf{j}}^i} \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.33}$$

$$\underline{V_{\mathbf{j}}^r} \leq V_{\mathbf{j}}^r \leq \overline{V_{\mathbf{j}}^r} \qquad \forall \mathbf{j} \in \mathcal{B}, \mathbf{j} \neq \mathbf{j}^{\#} \tag{4.34}$$

$$\underline{V_{\mathbf{j}}^i} \leq V_{\mathbf{j}}^i \leq \overline{V_{\mathbf{j}}^i} \qquad \forall \mathbf{j} \in \mathcal{B}, \mathbf{j} \neq \mathbf{j}^{\#} \tag{4.35}$$

$$\underline{P_{\mathbf{j}}} \leq P_{\mathbf{j}} \leq \overline{P_{\mathbf{j}}} \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.36}$$

$$\underline{Q_{\mathbf{j}}} \leq Q_{\mathbf{j}} \leq \overline{Q_{\mathbf{j}}} \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.37}$$

$$V_{\mathbf{j}^{\#}}^r = 1, \quad V_{\mathbf{j}^{\#}}^i = 0 \tag{4.38}$$

$$\text{MCE bounds of form (3.27)-(3.30)} \qquad \forall \mathbf{j} \in \mathcal{B} \tag{4.39}$$

$$P_{\mathbf{j}} \tan(\cos^{-1}(\text{pf}_{\text{neg}})) \leq Q_{\mathbf{j}} \qquad \forall \mathbf{j} \in \mathcal{B}_G \tag{4.40}$$

$$Q_{\mathbf{j}} \leq P_{\mathbf{j}} \tan(\cos^{-1}(\text{pf}_{\text{pos}})) \qquad \forall \mathbf{j} \in \mathcal{B}_G \tag{4.41}$$

$$\underline{r_{\mathbf{j}}^{\text{P}}} \Delta t \leq P_{\mathbf{j},t} - P_{\mathbf{j},t-1} \leq \overline{r_{\mathbf{j}}^{\text{P}}} \Delta t \qquad \forall \mathbf{j} \in \{\mathcal{B}_S \cup \mathcal{B}_G\}, \forall t \in [T] \tag{4.42}$$

$$P_{\mathbf{j},t} = \frac{1}{\eta_{\mathbf{j}}^{\text{D}}} P_{\mathbf{j},t}^{\text{sd}} - \eta_{\mathbf{j}}^{\text{C}} P_{\mathbf{j},t}^{\text{sc}} \qquad \forall \mathbf{j} \in \mathcal{B}_S, \forall t \in [T] \tag{4.43}$$

$$0 \leq P_{\mathbf{j},t}^{\text{sd}} \leq \overline{P_{\mathbf{j}}^{\text{sd}}} \qquad \forall \mathbf{j} \in \mathcal{B}_S, \forall t \in [T] \tag{4.44}$$

$$0 \leq P_{\mathbf{j},t}^{\text{sc}} \leq \overline{P_{\mathbf{j}}^{\text{sc}}} \qquad \forall \mathbf{j} \in \mathcal{B}_S, \forall t \in [T] \tag{4.45}$$

$$\underline{b_{\mathbf{j}}} \leq b_{\mathbf{j},t} \leq \overline{b_{\mathbf{j}}} \qquad \forall \mathbf{j} \in \mathcal{B}_S, \forall t \in [T] \tag{4.46}$$

$$b_{\mathbf{j},0} = b_{\mathbf{j},T} \qquad \forall \mathbf{j} \in \mathcal{B}_S, \forall t \in [T] \tag{4.47}$$

where:

- $\mathcal{S}$ refers to the set of substations, $\mathcal{F}_s$ to the set of feeders per substation s, and $\mathcal{B}$ to the set of buses connected to (and including) feeder l at bus $\mathbf{j}^{\#}$,
- $\mathcal{T} \subseteq \mathcal{B} \times \mathcal{B}$ corresponds to the grid transmission lines,
- The notation $(\underline{\cdot})$ and $(\overline{\cdot})$ represent the lower and upper limits on local bus variables;
- T represents the total number of time periods over which MOPF is being solved, with Δt representing the length of each time period, in hours. Thus, $T = \frac{\text{Hours to Schedule}}{\Delta t}$ and $t = [1, ..., T]$,
- The decision variables at time t are captured by the global decision vector $\mathbf{y}_t$ which captures the variables of each bus $\mathbf{j}$ at time t:

$$\mathbf{y}_t = \left[\mathbf{I}_t^r \;\; \mathbf{I}_t^i \;\; \mathbf{V}_t^r \;\; \mathbf{V}_t^i \;\; \mathbf{P}_t \;\; \mathbf{Q}_t \;\; \mathbf{a}_t \;\; \mathbf{b}_t \;\; \mathbf{c}_t \;\; \mathbf{d}_t \;\; \mathbf{I}_{\mathbf{flow}_t}^r \;\; \mathbf{I}_{\mathbf{flow}_t}^i \;\; \mathbf{P}_t^{\text{sd}} \;\; \mathbf{P}_t^{\text{sc}}\right]$$

  The decision variable over all time steps is denoted by vector $\mathbf{y} = [\mathbf{y}_{1,\forall T}; \mathbf{y}_{2,\forall T}; ...; \mathbf{y}_{Y,\forall T}]$. The bold characters here denote vectors.

- $b_{\mathbf{j},0}$ is the initial capacity or energy level of the storage device at time $t = 0$, at the start of the simulation

4.2.11 Decomposition for Distributed Computation

The OPF problem for the distribution grid with high penetration of DERs has high dimensionality, $\dim(\mathbf{y}) = (10N+2M)T*[\Phi]$, where M, N represent the number of grid transmission lines and buses respectively, and $[\Phi]$ denotes the number of maximum number of phases in the network [9]. As previously discussed, it may not be tractable for central solvers to reach optimal solution within a reasonable time. The use of

[9]Typically a distribution grid will have lines which are three-phase, and peripheral lines which service single- or two-phase loads. For simplicity of the calculation, here we assume that all lines have the same number of phases, namely $[\Phi] = 3$.

distributed solvers can address this. In particular, we will make use of the PAC algorithm again, as introduced in Chapter 3.

For the multi-period problem, the atomic vector, $\mathbf{a}_j$ then takes on the form of (4.48), where once again the last term are the coupling variables denoted by the $\hat{}$ symbol. Note that we have omitted the subscript for time, t, on all variables inside the vector $a_{\mathbf{j},t}$ for notational simplicity.

$$\begin{aligned}\mathbf{a_j} = [I^r_{\mathbf{j}} \quad I^i_{\mathbf{j}} \quad V^r_{\mathbf{j}} \quad V^i_{\mathbf{j}} \quad P_{\mathbf{j}} \quad Q_{\mathbf{j}} \quad a_{\mathbf{j}} \quad b_{\mathbf{j}} \quad c_{\mathbf{j}} \quad d_{\mathbf{j}} \quad \{I^r_{flow,\mathbf{jk}}, I^i_{flow,\mathbf{jk}}\}_{\forall \mathbf{k},\{(\mathbf{j},\mathbf{k})\}\in\mathcal{T}_\mathrm{D}} \\ P^{\mathrm{sd}}_{\mathbf{j}} \quad P^{\mathrm{sc}}_{\mathbf{j}} \quad \{\hat{V}^r_{\mathbf{k}}, \hat{V}^i_{\mathbf{k}}\}_{\forall \mathbf{k},\{(\mathbf{j},\mathbf{k})\}\in\mathcal{T}_\mathrm{D}} \quad \{\hat{I}^r_{flow,\mathbf{ij}}, \hat{I}^i_{flow,\mathbf{ij}}\}_{\forall \mathbf{i},\{(\mathbf{i},\mathbf{j})\}\in\mathcal{T}_\mathrm{D}}] \quad (4.48)\end{aligned}$$

where every vector element of $\mathbf{a_j}$ is composed of the variable over all time steps, as shown for a particular variable a:

$$a = [a_{t=1} \quad a_{t=2} \quad \cdots \quad a_{t=T}] \quad (4.49)$$

With the decomposed problem, we can now employ the PAC algorithm to resource scheduling problems for distribution grids with DERs.

4.3 Case Study Results

To evaluate the performance of the multi-period optimization, we consider distribution grids with varying penetration of DERs, solved for different time periods. We consider 3 main case studies:

- **Case 1:** We consider the IEEE 13-bus network with fixed DER penetration (demand response, renewable generation, and storage), and solve the MOPF problem for different time periods: 2 hours, 5 hours, 8 hours, and finally 24 hours. We use these results to motivate the need for multi-period optimization.
- **Case 2:** We consider the IEEE 13-bus network with varying penetration of DERs, and solve the MOPF problem for 24 hours. Namely, we consider networks with (a) only demand response; (b) (d) demand response and storage; (c)

demand response and renewables; and (d) demand response, renewables, and storage. These simulation results help us understand the importance of storage in active distribution grids.

- **Case 3:** We consider the IEEE 123-bus network with fixed penetration of renewable generation, and solve the OPF problem over 24 hours. This test case shows the scalability of the model, and the ability to consider more realistic networks which are unbalanced and have flexible resources.

4.3.1 Case 1: Motivating the need for Multi-period

Here we consider the IEEE 13-bus network with small renewable energy penetration, small demand response capabilities at 3 of the 5 loads, and 1MW of installed storage capacity. The MOPF problem is solved for 2, 5, 8, and 24hours, and the results are shown in Figs. 4-3, 4-4, 4-5, and 4-6 respectively. By comparing these results we see that the storage unit is discharged more rapidly in the beginning of the day when shorter time windows are used. While both the 8hr and 24hr simulations show battery discharging in hours 1, 4, and 7, the 24hr simulation looks ahead to future periods when electricity is more expensive (i.e. the evening, after 6pm) and charges in additional periods, such as in hour 3. Further the 24hr period simulation is the only one in which renewable generation is visible to the grid operator; the shorter windows don't show any generation as solar doesn't become available until 8am. From these results we see that longer windows are needed for efficient resource scheduling; else a battery may be depleted too early on in the day, or it may not be charged when electricity prices are low.

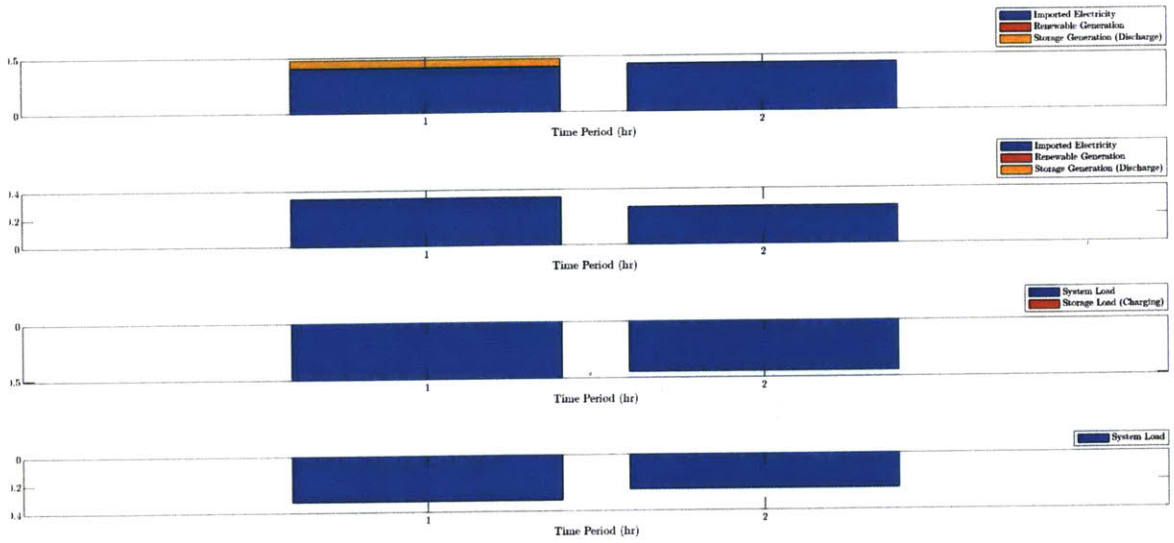


Figure 4-3: Simulation results for 13-bus network over 2hrs.

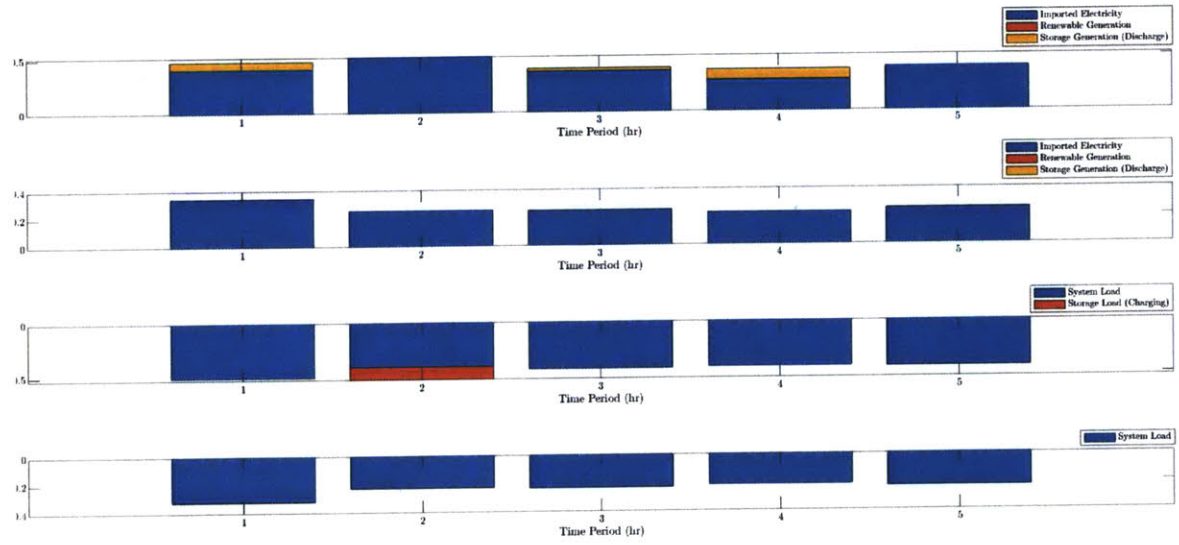


Figure 4-4: Simulation results for 13-bus network over 5hrs.

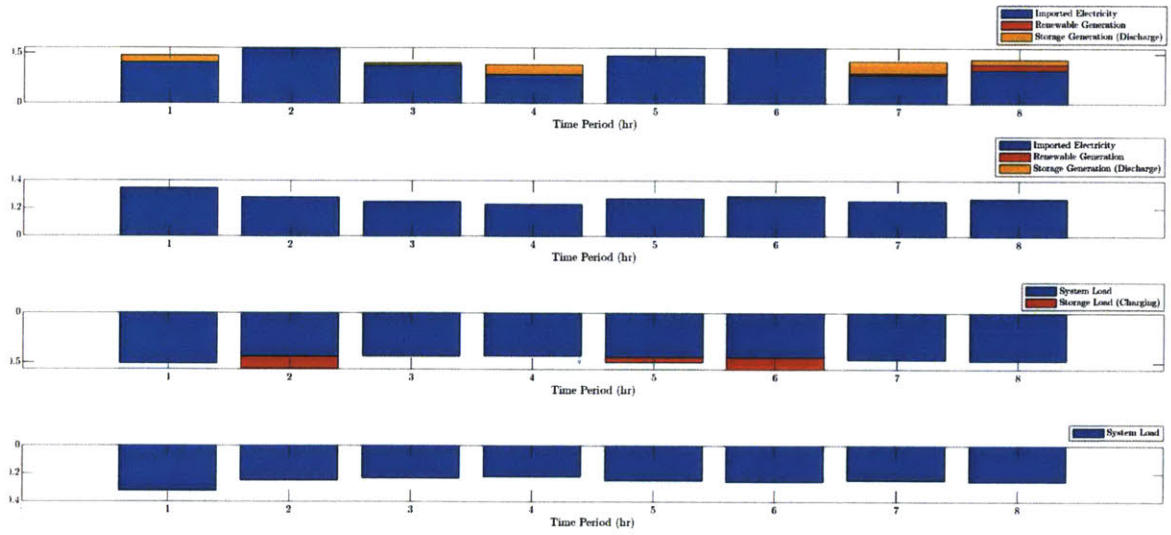


Figure 4-5: Simulation results for 13-bus network over 8hrs.

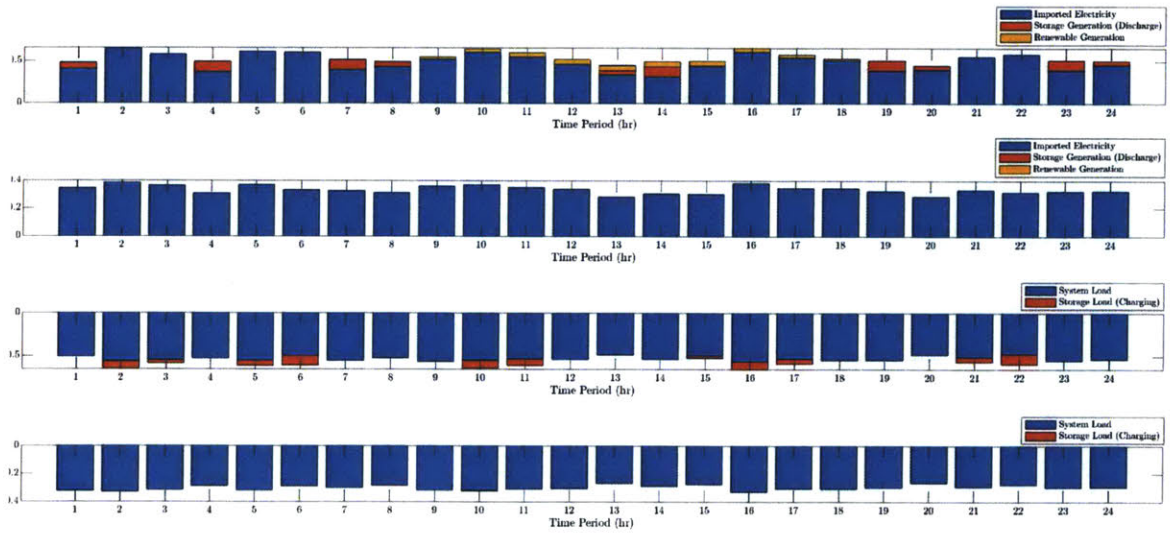


Figure 4-6: Simulation results for 13-bus network over 24hrs.

4.3.2 Case 2: Understanding the need for Storage

Here we consider 4 different combinations of DERs present in the IEEE 13-bus distribution grid, and also vary the amount of storage and/or renewable generation. Data for the simulations is provided in Appendix B.2 for the network default configuration.

Case 2.a: DR

In this test case, the only DER present is demand response, which is available at 5 of the 7 loads. The amount of available DR varies from 10%-50% of the total load at the node. See Appendix B.2.1 for full data. The simulation results in Fig. 4-7 over the 24hr horizon show roughly 10%-15% curtailment of load (through DR) throughout the network, for every hour. This is quite a substantial amount of curtailment, especially given that it persists throughout the day. A more reasonable DR strategy would curtail load only for a portion of the day, or would shift load from one hour of the day to another.

Case 2.b: DR and Storage

In this test case, both demand response and storage are present in the network. Demand response is available at 3 of 5 loads in the network, and a battery of 1MW capacity with initial charge of 150kW is present. See Appendix B.2.1 for full data. The simulation results in Fig. 4-8 show steady use of DR capabilities throughout the day - much like Case 2.a - and storage being continuously cycled. The battery is charged during periods of lower cost, and is discharged when prices are higher, namely during the morning hours of 7-9am and the evening after 6pm. In reality, the storage device may be operated with more limited cycling - rapidly switching between charging and discharging mode can wear the battery down. This suggests that the battery wear cost is not appropriately selected; it likely does not account for installation costs, and externalities such as environmental assessments and permitting fees.

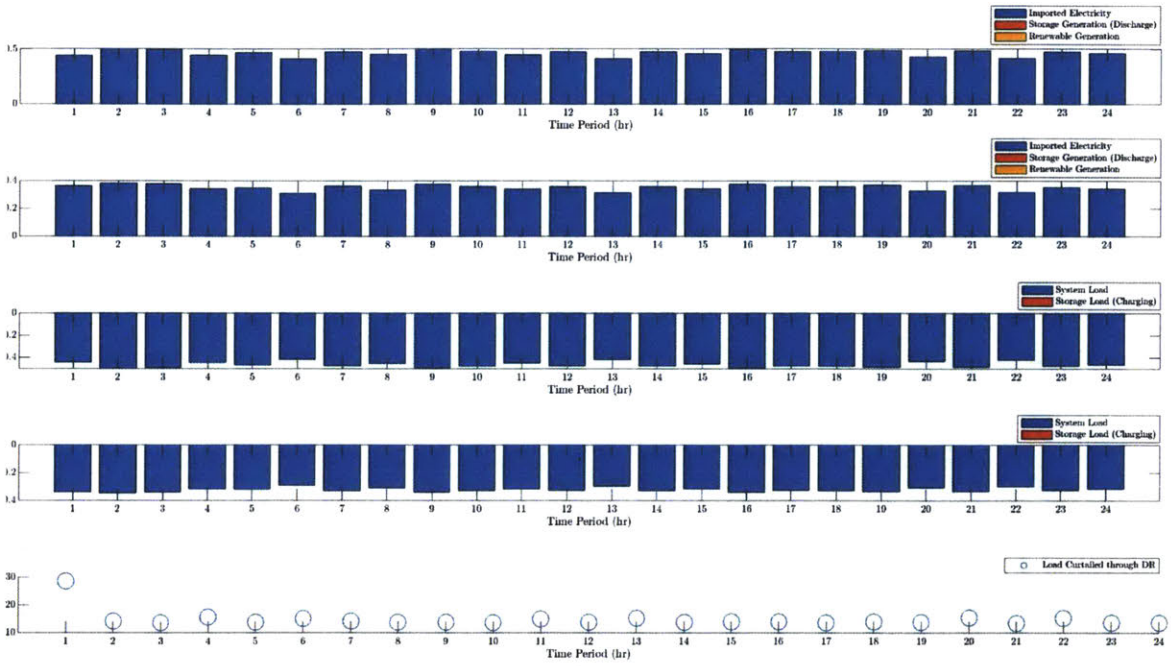


Figure 4-7: Simulation results over 24hr period for the 13-bus network with high DR capabilities.

Case 2.c: DR and Renewables

In this test case, we consider the combination of DR and renewable generation. Residential and commercial solar PV are sprinkled throughout the grid, and DR is present at 3 of the 5 loads. This configuration has lower DR capabilities than 2.a and 2.b. Two test cases are run for the 2.c grid: a lower PV penetration totalling 135kW, and a higher PV penetration totalling 265kW. See Appendix B.2.1 for full data.

The results for the two test cases are very similar, and are shown in Figs. 4-9 and 4-10 respectively. We see that renewable generation is used to reduce the quantity of imported electricity during periods of solar generation (namely 8am to 6pm). Meanwhile, DR is used throughout the day, varying slightly in total demand curtailment, but averaging around 7% of total load. One may expect lower DR utilization during periods of solar generation, but the amount of DR remains relatively constant. Again, this is not a realistic DR policy: curtailing load over every hour of

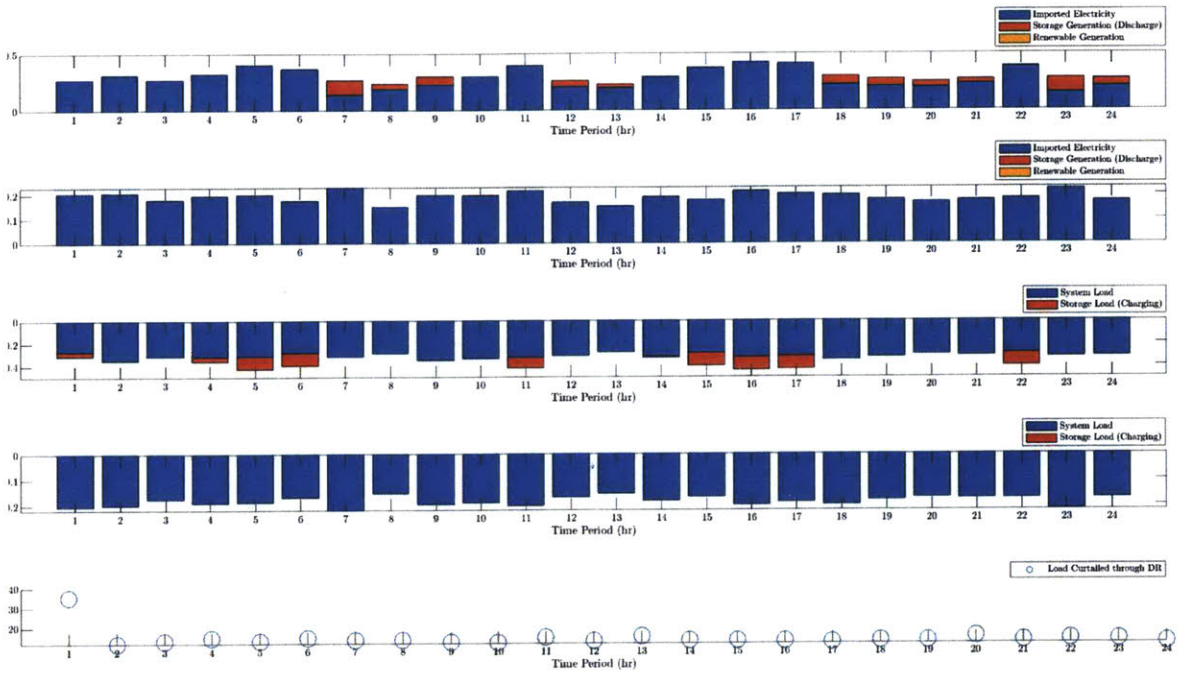


Figure 4-8: Simulation results over 24hr period for the 13-bus network with high DR capabilities and 1MW installed storage capacity.

the day is not feasible. A more realistic policy could include having a limit on the amount of curtailable load per node participating in DR, either over a 24hr period or over a week. This would ensure that grid operator only utilizes the DR resources when it most needed. Further, DR is currently modelled as a fixed percentage applied to load over every hour. This is not a realistic model. Rather, the percentage of allowable curtailment should vary throughout the day, and some base amount of electricity is likely always to be consumed by loads. Better development of DR policies is outside the scope of this thesis, but these simulation results show great potential for aiding in these decisions.

Case 2.d: DR, Renewables, and Storage

In this test case, we consider the presence of all resources (DR, renewables, and storage), with varying installed storage capacity. The network has DR present at

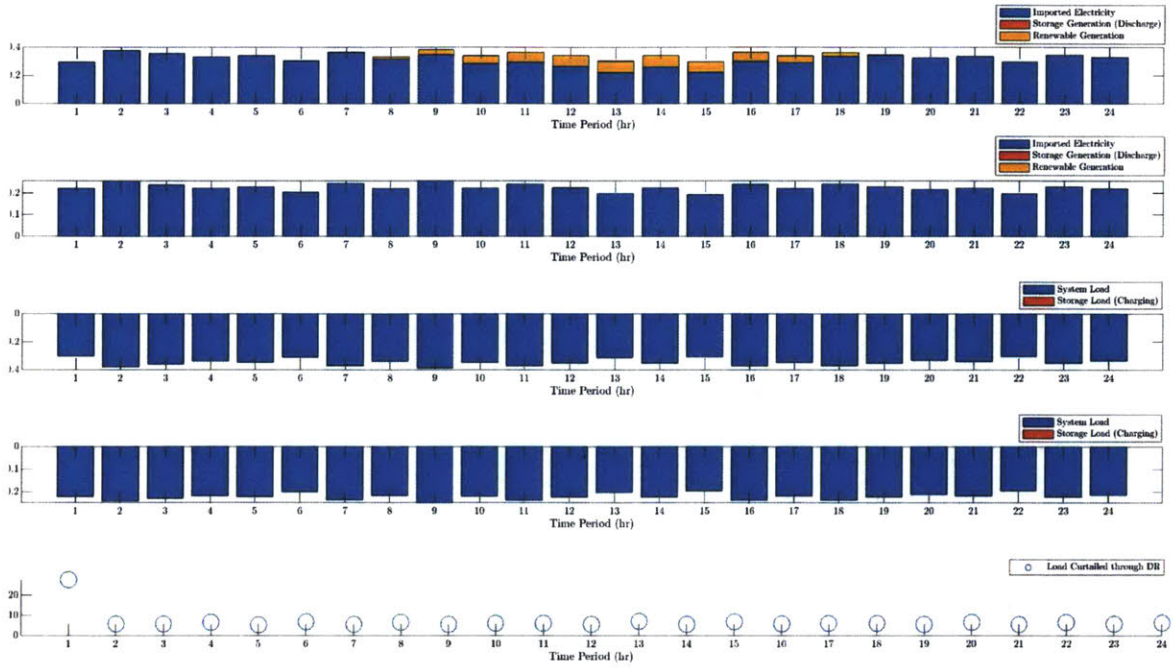


Figure 4-9: Simulation results over 24hr period for the 13-bus network with DR capabilities and lower PV penetration (135kW).

3 of the 5 loads, 135kW of solar PV, and varying battery capacities of: 1MW with 500kW initial charge, 0.5MW with 100kW initial charge, and 2MW with 300kW initial charge. Note that in the last two test cases, the day begins with the battery near or at the minimum charge it must maintain through operation (due to minimum SOC requirements). See Appendix B.2.1 for full data.

The results for these configurations are shown in Fig. 4-11, 4-12, and 4-13 respectively. Similar to other test cases, we see that DR resources are used steadily throughout the day, while solar PV generation is used once it comes available from 8am to 6pm. The use of storage is the most interesting between these test cases: the hourly LMP data is the same between all 3, while the specific demand values vary based on the generated demand profiles.

In all three cases, the battery is discharged during the night hours (6pm and onwards), even when there is limited capacity available (as with the 0.5MW case).

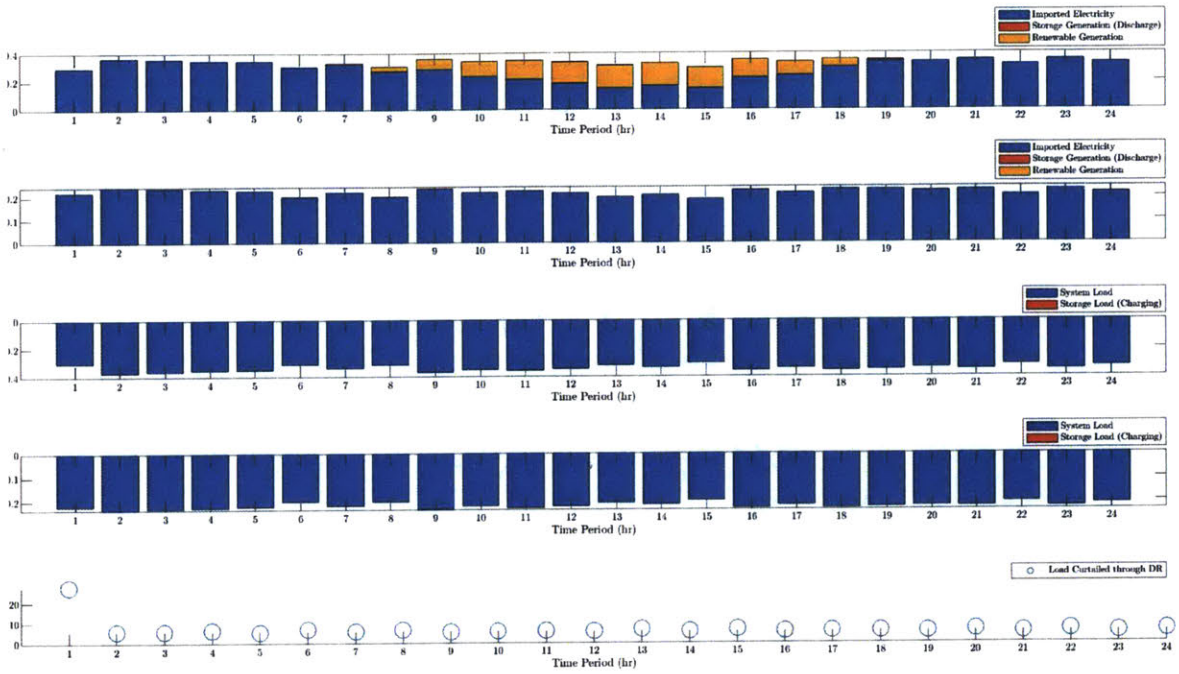


Figure 4-10: Simulation results over 24hr period for the 13-bus network with DR capabilities and higher PV penetration (265kW).

These are the hours when the LMP is the highest. Interestingly, when storage capacity is limited as with the 0.5MW capacity or is large as with the 2MW installed capacity, there is less cycling between charge and discharge, as compared to the case with 1MW installed capacity. When limited, the battery is used only for the highest prices during the day, and charged at the lowest prices, both in intervals of around 2hrs. When very large, the battery can be charged when electricity is cheaper and the stored power can be discharged over more periods, both in intervals of around 3hrs. For the middle case of 1MW installed capacity, the battery is discharged over more periods than for the 0.5MW case, but requires more charging than the 2MW case in order to meet network loads. These simple results show the importance of system planning, which must include battery placement and sizing for networks with DERs and varying demand profiles. Else, the resource will be strained to meet requirements, or may be oversized and too expensive.

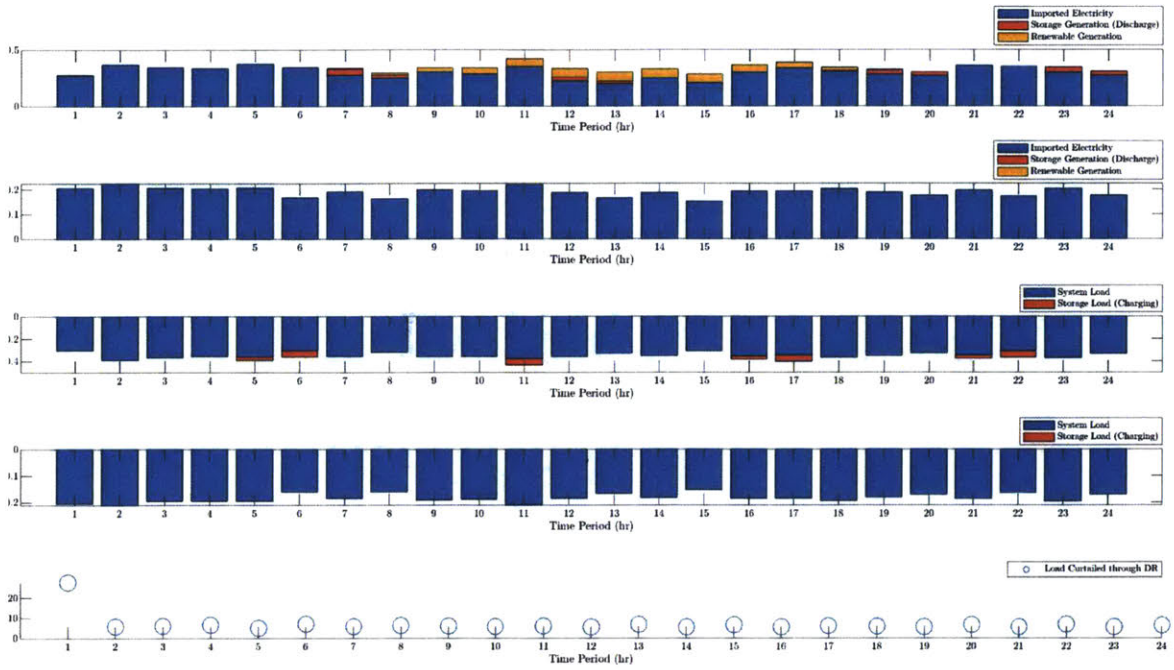


Figure 4-11: Simulation results over 24hr period for the 13-bus network with DR capabilities, PV penetration of 135kW, and 0.5MW storage capacity.

4.3.3 Case 3: Scalability of the Model

Here we consider the IEEE 123-bus network which has been modified to include high penetration of distributed renewable generation, in the form of residential solar PV (installed capacity of 179kW). Full network data is available in Appendix B.3 for the default network configuration. Here a very simple problem of fixed loads (no DR) and distributed renewable resources is considered. Note that this is still an optimization problem as the renewable generation can be curtailed. The results are shown in Fig. 4-14, where once again, renewable generation is utilized during the mid-day when generation is available.

For the network, we have $N = 119, M = 118, T = 24$ and 3-phase network, leading to $(10 \times N + 2 \times M) \times T \times 3 = 102672$ variables which need to be solved for. This is quite a sizeable problem, for what is still a small model network used as an academic example. A real distribution grid will have many more nodes, and as such, be a much

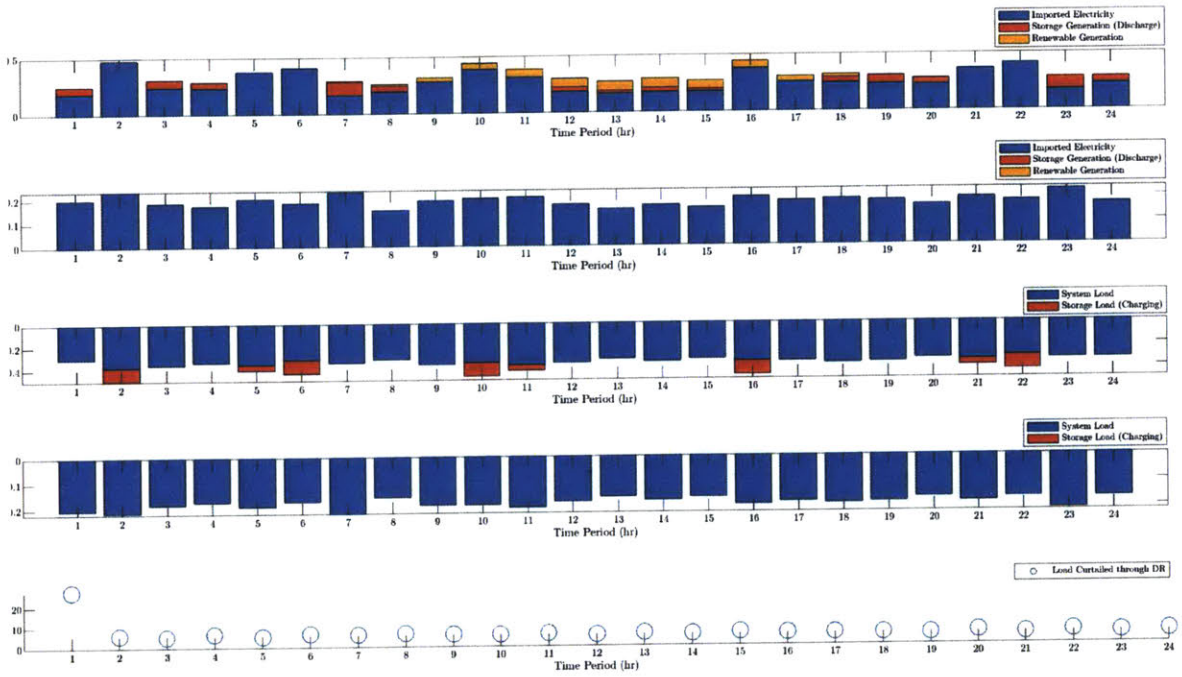


Figure 4-12: Simulation results over 24hr period for the 13-bus network with DR capabilities, PV penetration of 135kW, and 1MW storage capacity.

larger problem.

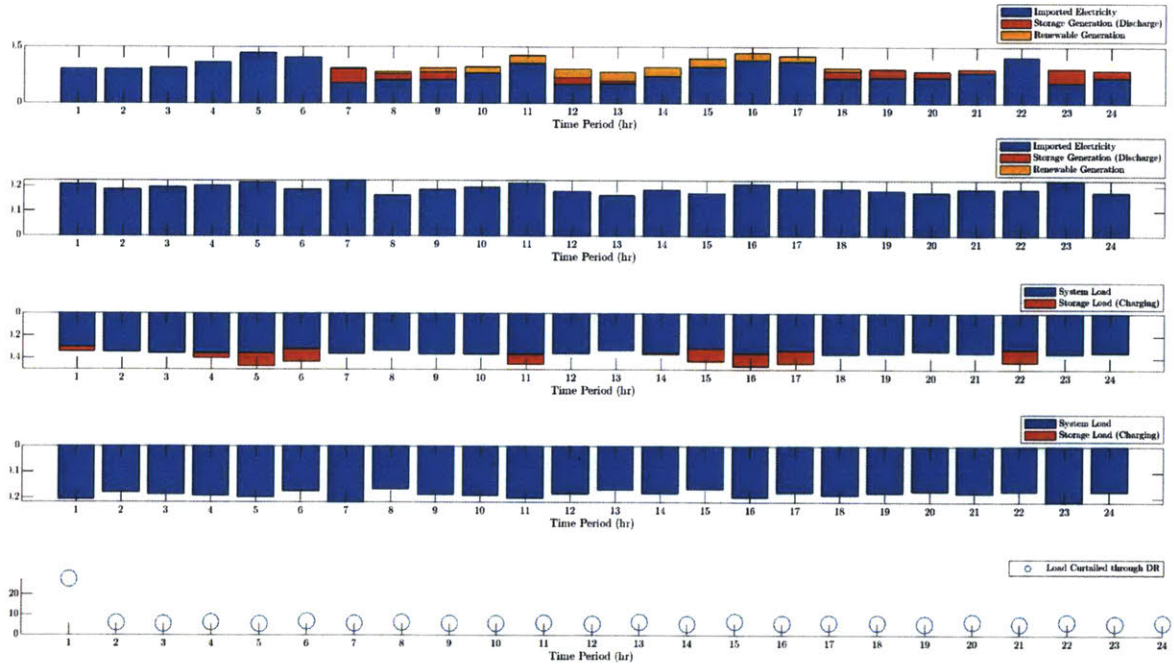


Figure 4-13: Simulation results over 24hr period for the 13-bus network with DR capabilities, PV penetration of 135kW, and 2MW storage capacity.

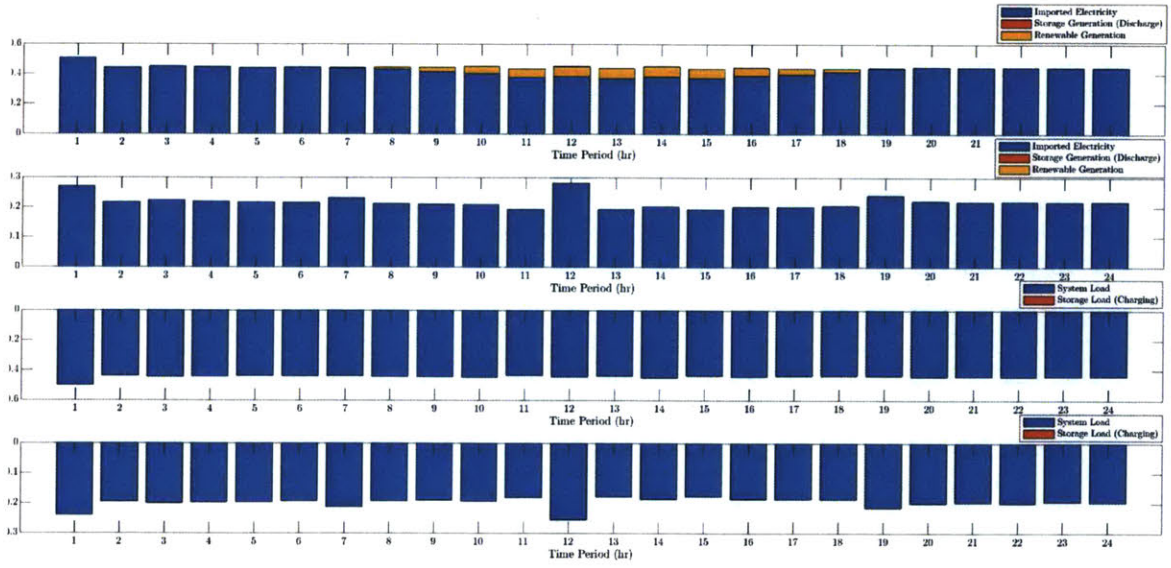


Figure 4-14: Simulation results over 24hr period for the 123-bus network with installed capacity of 179kW of renewable generation.

Chapter 5

Using Transactive Energy: Proposal for a DSO-Centric Retail Electricity Market for Smart Distribution Grids

In this chapter of the thesis, we shift away from day-ahead resource scheduling (i.e. hourly basis, or day-ahead schedules) and navigate towards real-time decision making and electricity pricing. The PAC algorithm introduced in Chapter 3 can once again be utilized to solve the OPF problem; this time however, the results of interest are both primal and dual variables. The primal variables once again describe the optimal scheduling of generating units and loads. The dual variables now give us the retail price for electricity - namely, the shadow prices tell us how much a customer should pay to consume electricity, and how much a DG should be paid for generating electricity.

We begin by providing background information on the structure of the US electric power market [1], then proceed to describe the proposed retail market and present results validating the design.

[1] While the proposed retail market mechanism can be used in any electric system, the focus within this thesis is on the US electric power market.

5.1 Introduction to US Electric Power System

The US electricity grid is composed of three main interconnections, which are connected to each other via high-voltage DC power transmission lines (DC tie lines) or variable-frequency transformers (VFTs). These tie lines and VFTs isolate the interconnections and allow them to operate largely independently, but still allow for some electricity transfer as needed for emergency balancing. The three interconnects consist of the Eastern Interconnect, the Western Interconnect, and the Electric Reliability Council of Texas (ERCOT). Both the Eastern and Western Interconnections of the US are linked to the Canadian power grid.

The US electricity grid is overseen by North American Electric Reliability Corporation (NERC) and Federal Energy Regulatory Commission (FERC), which are in charge of the creation and enforcement of reliability standards, operational requirements, and policies. NERC is a not-for-profit international regulatory authority covering the US, Canada, and the northern portion of Mexico. NERC is the electric reliability organization (ERO) for North America, and oversees the reliability and security of the grid. NERC is overseen by FERC in the US, and governmental authorities in Canada. [40] FERC is an independent regulatory agency which oversees interstate transmission of electricity, natural gas, and oil. Under the Energy Policy Act of 2005, FERC's responsibilities include regulating the transmission and wholesale electricity markets, and overseeing the reliability of high voltage interstate transmission system [2]. [41]

Within the interconnects, there are three entities which oversee the operation of the grid and ensure reliable and economic grid operation:

- Independent System Operator (ISO): Coordinates, controls, and monitors the grid. ISOs typically operate within a single US state or multiple states, but over smaller geographical areas as RTOs. The ISOs were formed upon recom-

[2]Note that here only two of the main responsibilities of FERC are considered, as we are concerned only with the electric grid operation and markets. FERC has additional responsibilities including overseeing mergers and acquisitions (M&A) of electricity companies and various other tasks pertaining to natural gas, oil pipelines, hydroelectric projects, and environmental considerations of electricity production and transportation tasks.

mendation by FERC, and act as an independent Federally regulated entity.

- Regional Transmission Organization (RTO): Is a transmission system operator (TSO) with similar functionality as the ISOs, but typically over larger geographical areas. The RTOs were voluntarily created upon the suggestion of FERC in 1999, and act as independent governing bodies.

- Balancing Authority: Controls transmission and generation within its region to ensure supply meets demand at all times, throughout the network. Balancing is primarily done by turning on/off generators, and/or importing/exporting electricity from neighbouring balancing authorities. Within an interconnect there are multiple balancing authorities. All RTOs in the US are also balancing authorities, and are subject to the reliability standards issued by NERC and approved by FERC. [3] [42]

The roles of ISOs and RTOs are similar, as they both operate regional electricity grids, oversee the wholesale and retail electricity markets to ensure open access for supply, and provide reliability planning. However, RTOs have greater responsibility for the transmission network. Within North America, there are currently seven RTOs: California ISO (CAISO), ISO-New England (ISO-NE), Electric Reliability Council of Texas (ERCOT), Midcontinent ISO (MISO), New York ISO (NYISO), PJM Interconnection (PJM), and Southwest Power Pool (SPP).

5.2 US Electricity Markets

The US electricity markets have two components: wholesale and retail. Wholesale electricity market (WEM) encompass electricity sale and exchange between electric utilities and electricity traders. The retail market involves the sale of electricity to end-use consumers, which includes large-scale facilities, small-scale businesses and buildings, and residential units. A utility (or more generally known as a Load Serving

[3]Note that the Electric Reliability Council of Texas (ERCOT) is a unique entity in that it is a balancing authority, interconnect, and RTO at the same time. Also note that ERCOT is not subject to FERC jurisdiction.

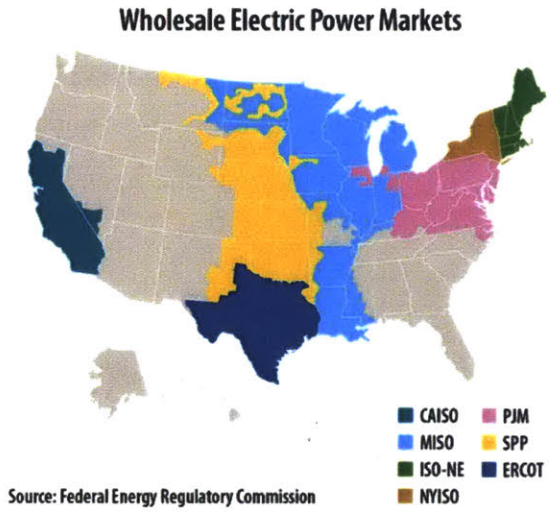


Figure 5-1: Graphic showing the wholesale electric power markets in the US. The grey regions are under traditionally regulated wholesale markets. The coloured regions are the ISOs/RTOs where deregulated competitive markets exist. [2]

Entity (LSE)) will purchase electricity from the wholesale market (at wholesale price) from producers and/or traders, and resell the electricity to end-use consumers.

Within the US, both traditionally regulated electricity markets and competitive markets exist, as shown in Fig. 5-1. In the traditionally regulated markets, utilities are vertically integrated, and are responsible for (and own) the generation, transmission, and distribution systems to serve consumers. In competitive markets, which are overseen by the ISOs (and RTOs), independent power producers and non-utility generators trade power, which is purchased by LSEs who sell power to consumers.

Similarly, within the retail market space, both traditionally regulated and competitive markets exist, as shown in Fig. 5-2. In traditionally regulated retail markets, consumers cannot choose who to purchase power from or where their electricity is generated. In competitive retail markets, consumers are given the choice between competitive retail suppliers; these can be directly from power producers or an LSE. Note that even in competitive retail markets, some municipally-owned utilities may not offer customer retail choice; different states and jurisdictions structure the market in different ways. [2]

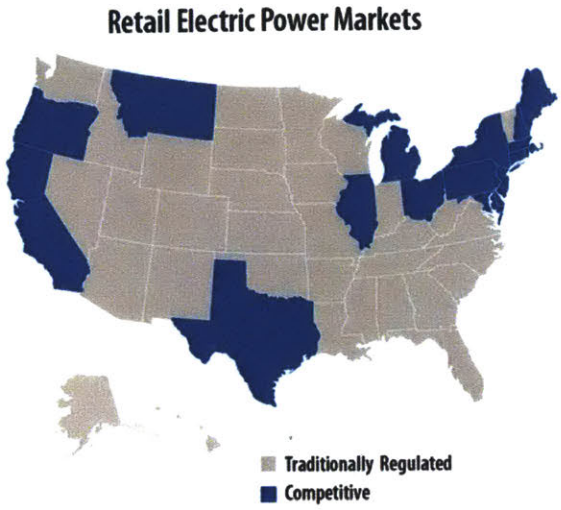


Figure 5-2: Graphic showing the retail electric power markets in the US. The grey regions are under traditionally regulated wholesale markets. The coloured regions are states where deregulated competitive markets exist. [2]

5.2.1 Market Structure and Renewable Energy

The structure of the wholesale and retail markets impacts the ability for renewable generation to become part of the electricity mix. In traditionally regulated markets, the utilities must invest into renewable energy or "green pricing" products which may include renewable energy certificates (RECs). In these markets, renewable electricity is typically more costly, and barriers to entry are higher for renewable generation as decisions are made top-down. Within deregulated and competitive markets (for both wholesale and retail), customers have electricity choice and so have more power over how much renewable energy or green pricing products are available; a higher demand for these products will encourage LSEs to purchase electricity from green resources. Finally, the market structure also influences whether customers can take part in power purchase agreements (PPAs) with renewable resources [4]. For any PPA,

[4]Most renewable/green energy projects in the US are supported on the basis of PPA. The PPA is a contract between the electricity generator - in this case a renewable generator such as a wind or solar farm - and an electricity buyer. They are typically long-term contracts which outline the sale and price of electricity, and schedule for delivery. Financial counter parties can also take part in PPA agreements as they can absorb the higher risk of engaging in a PPA with a renewable project, which have uncontrollable and intermittent generation. PPAs are overseen by FERC.

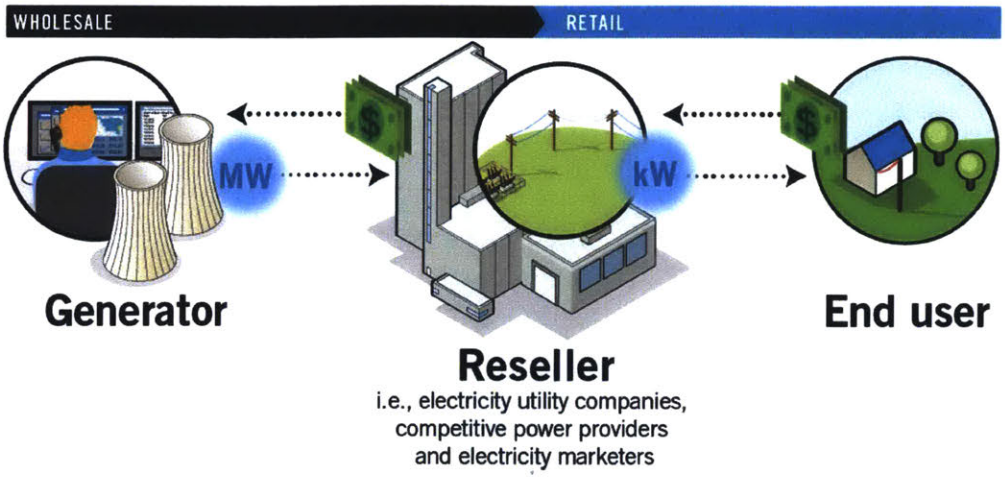


Figure 5-3: Infographic of the wholesale and retail markets from PJM, under a competitive market framework [3]. Other ISOs/RTOs have the same general structure.

the project must be within a competitive wholesale market. For physical (direct) PPAs, the consumer must also be within a competitive retail market and the project must be interconnected with the consumer's ISO. For financial PPAs, the consumer can be located anywhere in the US. [2]

5.2.2 Wholesale Electricity Markets

Within this thesis, we consider the more interesting problem of integrating DERs within the distribution grid, many of which are behind-the-meter, through market mechanisms. To do so, we consider regions with competitive wholesale electricity markets, where more complex market structures exist. In this section, we introduce the different electricity markets which work together to create the WEM. Within competitive wholesale electricity markets there exists energy markets, capacity markets, and ancillary services markets. These will be briefly described below.

Energy Market

The energy market is the largest component of the WEM, and deals with real-time and day-to-day energy balancing. This market ensures supply meets demand, by balancing the needs of buyers, sellers, and other market participants. The energy market is

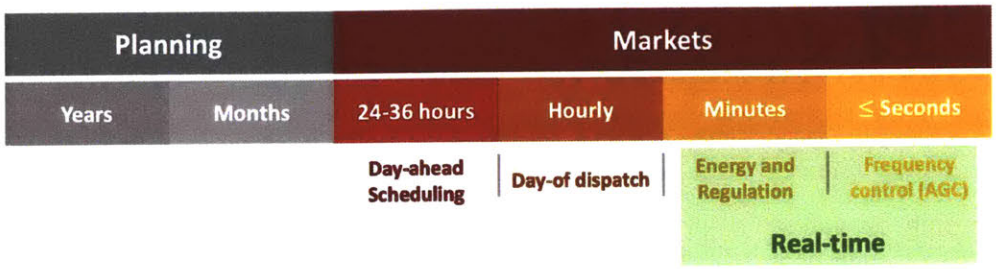


Figure 5-4: Typical timeline for planning and operations within the US electricity system. Markets include day-ahead and real-time segments.

composed of the day-ahead and real-time energy markets. A general timeline for grid planning and market operations is presented in Fig. 5-4, noting that the focus of this thesis is on the markets side of the figure. Note that in some WEMs, the price of electricity can be predetermined through bilateral contracts, which are between an electricity buyer and a seller, determining the price of electricity, and the quantities to be purchased (or sold), rather than through organized wholesale markets [3].

Day-Ahead Market The day-ahead energy market is a forward market: the prices for electricity are set on an hourly basis (typically) for the next day of operation. Typically the day-ahead market closes 24-36 hours in advance of the operating day. The hourly prices are determined through an optimization problem, which seeks to minimize costs incurred, using the offers from generators, and bids from power consumers [5]. Generally, the lowest-priced generators are "cleared" first (commitment to purchase power from them over the next day is made), save for any resources which may have prior purchase agreements. Generator offers are cleared from the lowest- to highest-priced offers until the demand bid is met, along with some extra reserves. The prices and schedule in the day-ahead market are binding, and allow buyers and sellers to hedge against price volatility in the real-time market. However, deviations from the schedule will be cleared in the real-time market. [3, 43]

[5]Note that this includes both LSEs and market-related financial transactions.

Real-Time Market The real-time energy market is a spot market: the prices and quantity of electricity is determined for immediate procurement and delivery to balance differences in day-ahead commitments and actual real-time production or demand. The real-time market can clear anywhere between every 5-15 minutes, depending on the ISO under which it is coordinated [6]. Any deviations from the schedule determined in the day-ahead market is cleared in the real-time market. The real-time market establishes a locational marginal price (LMP) that is paid or charged to participants in the day-ahead market for demand or generation which deviates from day-ahead commitments [43]. Note that the real-time market can have significant volatility, but the average price of electricity remains fairly stable.

Locational Marginal Pricing This is an integral component of the energy market and allows wholesale prices to reflect the locational differences within the power grid. More specifically, the different prices reflect the operating characteristics and major constraints on the transmission system, as well as losses resulting from the physical transmission grid. In a lossless network, the LMPs at all points would be the same. Further, LMPs provide information on congestion in the network - areas of congestion will have higher LMP prices. Further, the price of generation across an ISO's jurisdiction will also vary; thus, the LMP must reflect these regional differences, as all loads cannot be served by the lowest price generator(s). LMPs give more granular information on the entire grid, and are important as they account for more than just the price of generating the electricity.

Capacity Market

The capacity market is a smaller component of the WEM and is concerned with near long-term energy needs. Through the capacity market, the ISOs must ensure there will be enough power supply in the future, to meet the forecasted average and peak demand. Capacity markets are forward markets where contracts are determined through Forward Capacity Auctions (FCA) which are held annually, three years before

[6] ISONE and PJM clear every 5 minutes [3,43]. CAISO operates real-time markets which dispatch power every 5 and 15 minutes [44]

the operating period (for both ISONE and PJM [3, 43]). In the auction, different suppliers compete to obtain commitments to supply a given capacity of energy, in exchange for market-priced capacity payments. These resources commit to deliver electricity (or limit demand for demand-response resources) when needed, especially in the case of emergencies, for the year in which they are committed. The structure of the capacity market supports the development of new resources to meet future electricity needs, and help retain existing resources. Resources which clear in the capacity market receive a stable revenue stream, even those which only serve peak demand but don't run all year.

Ancillary Services Market

The ancillary services market work in tandem with the energy market to ensure that supply meets demand at all times. This balancing is done through the regulation market and reserves market, which make up the ancillary services. Other ancillary services include voltage regulation and black-start, but there are no clear market derivatives for these functions. In some competitive WEMs, the ancillary services may be procured on a cost-based method, rather than through a structured market. However, having ancillary services markets ensure that resources are priced adequately under shortage conditions, and provides structured market derivates for operating practices related to reliability [45].

Regulation Market The regulation market operates on the seconds timescale and corrects short-term, unforeseen fluctuations in supply/demand and focuses on the reliability of the system. The regulation market includes Automatic Generation Control (AGC), which adjusts generation and consumption of participating resources every 2-6 seconds [7], in response to automated signals. This may also be called the regulation reserve [45].

[7] ISONE requires AGC every 4 seconds [43]

Reserve Market The reserve market operates on the minutes timescale and meets demand in the event of unexpected loss of generation or transmission resource. It is composed of spinning and supplemental reserves, which both use resources which have committed to reserving a specific percentage of their generating capacity to meet emergency needs. Spinning reserves are a contingency element, which consist of resources already synchronized with the grid to reach targeted output within 10 minutes. The supplemental reserves consist of resources which may or may not be synchronized with the grid, but can be ramped up to supply the required energy within 10 minutes. The reserve market is necessary for emergency situations and ensures the reliability of the grid.

5.2.3 Participation of DERs in Electricity Markets

The current practice in the US for the integration of DGs is for them to participate in the WEM [46]. While this participation largely pertains to DGs, certain amount of participation also occurs from DR units and storage devices. FERC order 841 takes steps towards standardizing the participation of storage resources in wholesale energy, ancillary services, and capacity markets, by treating storage as a generator [47]. Despite this, there is no clear treatment of value stack benefit for storage, or aggregated distribution connected storage.

Further there is limited ability for small-scale DERs and behind-the-meter resources to participate, especially in grids where Advanced Metering Infrastructure (AMI) is not available. While some DER participation is allowed in the WEM, this is primarily for large-scale resources or through aggregators. However, there remain constraints on minimum resource size, capacity requirements, and/or ramping rates which still limit DER participation. Different ISOs/RTOs have made their own decisions on how to treat different DER classes. For example, CAISO allows demand response to participate and submit bids, with aggregation of a minimum of 100 kW, as long as these resources do not inject energy into the grid. In comparison, most other system operators require aggregations to be at least 1 MW. These differences prohibit the easy integration of small-scale DERs into the electricity system.

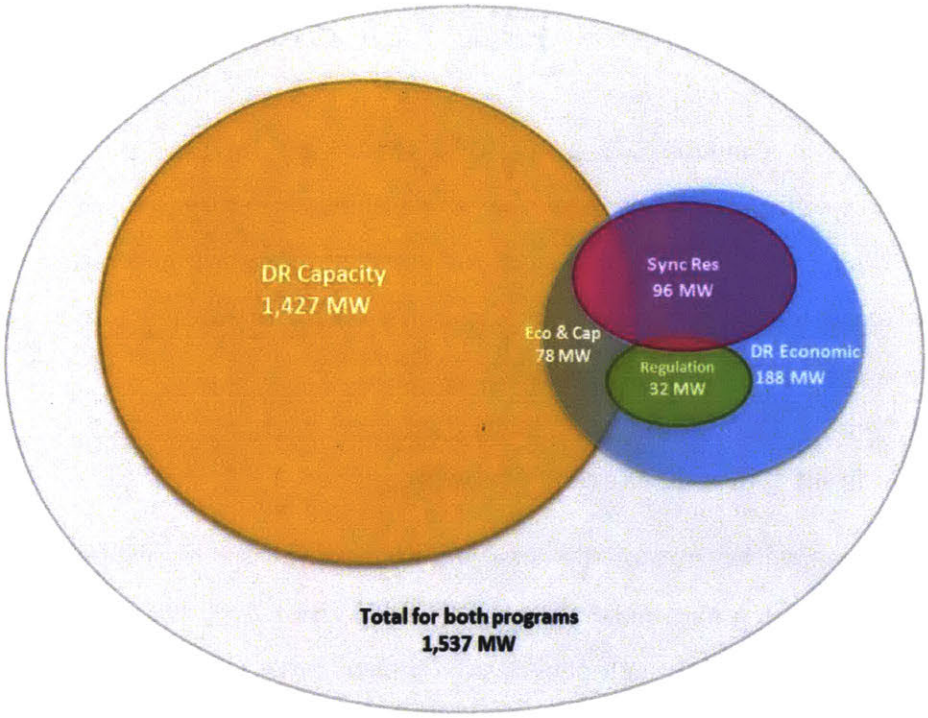
Demand response participation in the WEM is also documented by PJM, as shown in Fig. 5-5. From Fig. 5-5a we see that DER units which participate as DR primarily take part in the capacity market. The report goes on to detail that most DERs participating as economic DR are certified to provide ancillary services [4]. This looks very promising for DERs in the PJM region. However, we note that from Fig. 5-5b, all of the behind-the-meter participating units are generators which use carbon-based fuels; there is no participation of solar PV or storage in these programs, despite their penetration in the grid.

Limitations of Current Market Structure

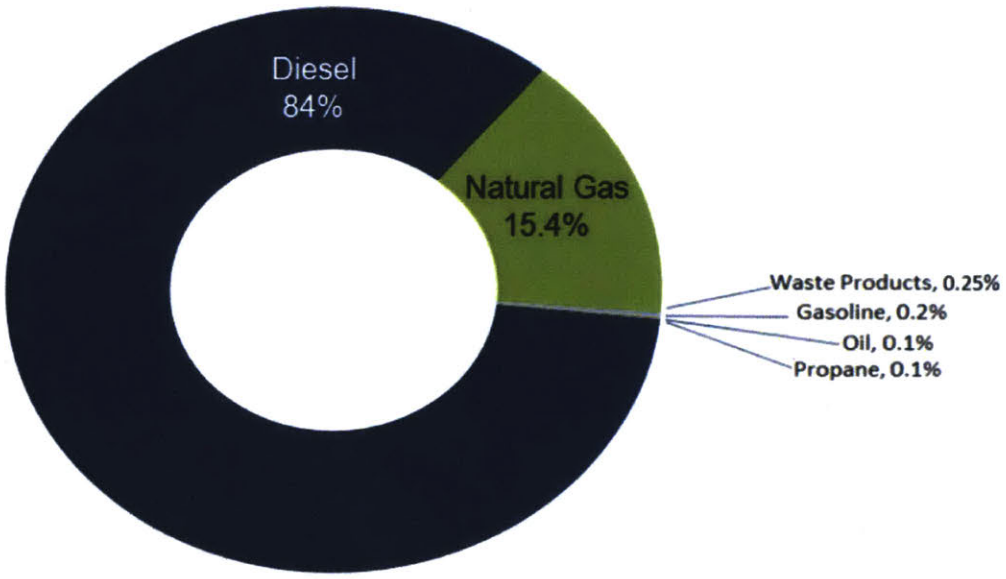
One of the central features of the emerging Smart Grid is a highly transformed distribution grid with a high penetration of DERs. Within New England, the expected growth of solar PV shows high penetration of small-scale rooftop solar resources, see Fig. 5-6. Following this trend, the current electricity market structure - namely only having a competitive WEM - may not be sufficient for efficiently integrating these resources and realizing efficient and reliable power delivery. Thus these highly flexible, low-cost, and distributed resources will be under-utilized if the current market structure persists.

The WEM largely takes on a centralized architecture, with the ISO/RTO overseeing the market. Multiple participants take part, but the overall optimization problem which considers the resource allocation based on the bids made by generators and consumers, is solved in a centralized manner. As discussed in Chapter 2, the increased penetration of DERs is pushing the structure of the distribution grid towards a decentralized - perhaps distributed - form. Thus a centralized approach which considers all resources seems to be inefficient, and such large-scale optimization problems can quickly become intractable. Thus, we may want to move towards decentralized or distributed architectures for the market as well as the computations associated with decision making.

As briefly introduced in the discussion of DER participation in the WEM, we note that minimum resource size requirements force DERs to be aggregated and



(a) Diagram of the MW capabilities of DERs registered in DR programs, showing the primary participation of DR-DERs in the capacity market.



(b) Fuel types for behind the meter DR generation units which participate in the capacity market as DR units for DY 2018/2019.

Figure 5-5: Data from PJM for the DY 2018/2019, for DERs participating as DR [4].

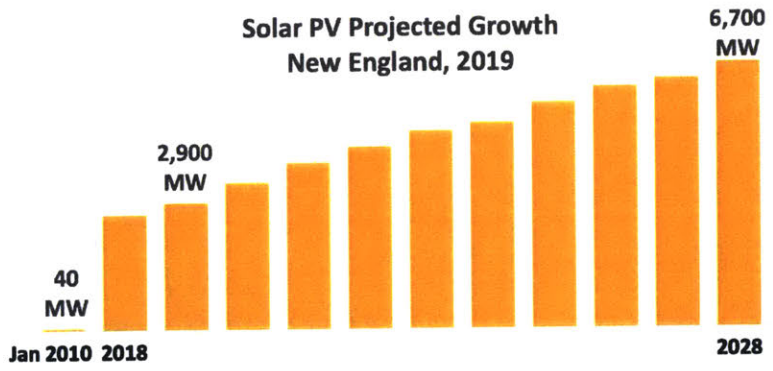


Figure 5-6: Plot showing the expected growth of solar PV in New England, as of 2019. This is primarily rooftop and small-scale solar resources.

bid directly into the WEM. This can be limiting for many DERs, and also further exacerbates the immense computational requirement of coordinating a large number of resources. We reiterate that the transmission system consists of far fewer resources than a smart grid with high penetration of DERs may have.

The lack of a real-time retail market also means the volatility of renewables (or similar resources) is not reflected in hourly energy prices. A plot of renewable energy production in a sample day for the New England system is shown in Fig. 5-7. Over the course of the day, the output from wind and solar varies greatly. Further, the demand for electricity, which is not shown on this plot, will also vary. Using a fixed-rate pricing structure means the flexibility of DERs and the variability in their availability will not be adequately accounted for. Figure 5-8 shows a sample set of price-demand curves and highlights the inefficiencies associated for fixed retail prices, where both supply and demand will vary continuously throughout the day. During times of high demand, resources should be paid more for meeting the requirements (also for peak demand hours), while during times of low demand, the resources should be paid less. This picture becomes more complex when considering the flexibility of DR units, which won't be adequately compensated for the services they can provide under a fixed pricing structure. When demand is high and generation is low, DR units are more valuable to the system operator, as they can be leveraged to help balance the system, and so should be paid accordingly.

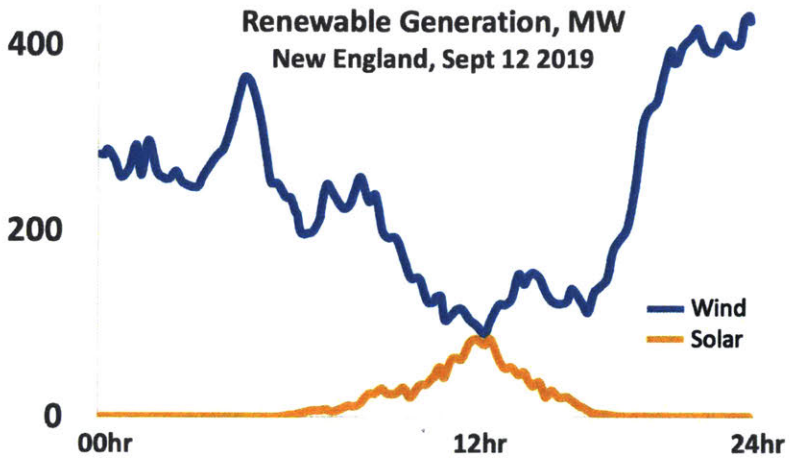


Figure 5-7: Plot showing the variability in renewable energy production over a day in New England.

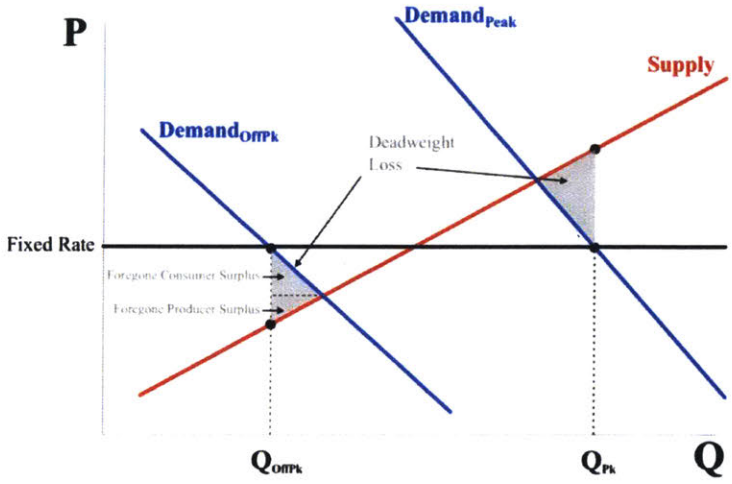


Figure 5-8: A set of price-demand curves showing the inefficiencies of using a fixed retail price. Further, fixed prices can't leverage the flexibility of DERs.

Finally, these DERs can be owned and operated by different stakeholders. The efficient and reliable operation of the distribution grid thus requires an overall market structure that allows the procurement and integration of power generation from these DERs, while maintaining data privacy. A retail market that oversees the participation of DERs in the distribution grid and implements a suitable mechanism for their scheduling and compensation is highly necessary. Figure 5-9 shows a schematic of the desired retail market structure and interactions between different stakeholders,

including the WEM and the DERs.

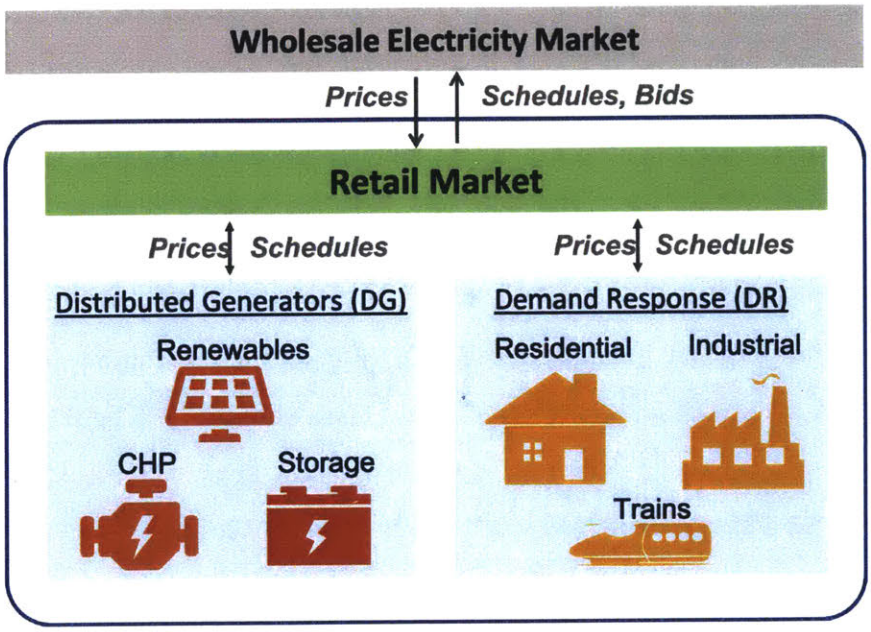


Figure 5-9: A market structure in the form of a retail market within the distribution grid is necessary to best utilize the DERs within the network.

The need for such a market has gained considerable traction, with various approaches being taken. The concept of transactive energy is one such promising design, whereby a service-based value of power is used to influence desirable behaviours from various autonomous agents, which include prosumers and DERs at the gridâĂŹs edge, at fast timescales [9, 10]. In this way, transactive energy is providing a link between the physical power flow in the grid, and market derivatives and regulatory constructs. In this framework, an independent agent does not have to cede control of its DERs to a centralized authority; rather it reacts to price signals to provide services to the grid, for which it will be compensated.

As the number of DERs and prosumers increases, more structure is warranted to coordinate and carry out the underlying transactive energy schemes. New operational entities such as a Distributed System Operator (DSO) and a retail market coordinated by DSOs may be needed. The DSO can take on various roles, including maintaining system reliability, facilitating transactions between agents and aggregators, and acting in the capacity of a utility whereby the DSO procures energy and coordinates a retail

market [48]. The latter is the focus of this thesis, in which a structured set of market interactions are proposed for the distribution grid, each one of which may be viewed as a transactive energy scheme. A discussion of approaches taken in literature and the design of the proposed retail market are presented in the next two sections.

5.3 Literature Survey

The coordination of power delivery as well as the overall market design as the number of DERs begins to increase are areas of active research [49–75]. The literature herein can be broadly grouped into centralized [52–58], decentralized [59–64], and distributed approaches [65–75], with this paper being a member of the third category. Reference [51] is a survey paper that gives an excellent overview of different market approaches to integrating DERs, particularly focusing on prosumers and flexible loads.

References [52–58] consist of centralized market perspectives. A hierarchical market structure is proposed in [52] and [53] to interlace the operation of the WEM with the retail market, where a Distributed Network Operator (DNO) is responsible for operating the market and load aggregators for smooth interactions with the WEM. In [54] the optimal dispatch of DR units in a distribution grid occurs through collective participation in the WEM, while in [55], competitive participation for the provision of ancillary services is used. The authors in [56] propose a centralized market where the DSO schedules the day-ahead transactions for DERs in the distribution grid. Bids are made by prosumers, generators, and microgrids, and both the DSO market and WEM are simulated iteratively until both wholesale price and distribution-level Locational Marginal Prices (d-LMP) are determined. The DSO is assumed to maximize their payoff. In this design, DC power flow is used, which can yield large errors for the highly unbalanced distribution grids. Meanwhile, [57] and [58] use an iterative procedure to solve the Optimal Power Flow (OPF) problem based on a Three-Phase Current Injection Method (TCIM), and determine d-LMPs per phase per bus through difference of power constraints. Due to the centralized perspective proposed in these papers, agents are required to disclose operating constraints and utility functions to

a central entity. In doing so, the agents lose data privacy; additionally, in market designs where the DSO is maximizing its own profit, this acts against the best interest of the market players. In addition, such a calculation of d-LMPs for a distribution grid is computationally significantly harder than the typically smaller sized transmission grid, and the central optimization problem can quickly become intractable with large number of DERs participating. Finally, these papers focus largely on d-LMP calculation, rather than proposing holistic market interactions.

An attractive alternative to centralized perspectives is a decentralized one. Herein, each agent is assumed to use local information to make decisions, but is otherwise unaware of the presence of others and do not therefore communicate with them. Such decentralized EMS are proposed in [59] and [60], to coordinate the operation of DERs and networked microgrids in the distribution grid. Decentralized market designs have also been proposed in [61–64]. The authors of [61] introduce a decentralized market with bilateral trading rules which result in fair allocation of the line losses. However, the relation between trading rules and optimal dispatch of DERs are not addressed. More recent works are those in [62–64], each of which consist of different approaches. The market in [62] is a peer-to-peer (P2P) structure in which different agents bid in the marketplace to determine day-ahead energy schedules. The market in [63] uses blockchain to facilitate energy transactions, while [64] makes use of game-theoretic approaches to determine real-time d-LMPs. All of these proposed markets have either limited interactions or do not accommodate interactions with the WEM. The main disadvantage of decentralized solutions simply stems from the fact that each agent's decision is local and therefore predicated on the assumption that coupling with others is weak. This leads to limited applicability without intervention from a centralized decision maker and therefore cannot be scaled up, especially as the number of DERs increases.

This brings us to the third category, of distributed solutions, where the agents communicate with peers/neighbours to achieve a common objective, thus potentially leading to an optimal global objective being achieved. Since distributed methods rely primarily on local computations at each node and P2P exchange of information,

they are computationally tractable. Moreover, if well designed, they can be resilient to communication link and single-point failures, and can preserve the private information of the DERs, while still realizing network-level objectives. Papers [65–69] correspond to this category. The work in [65] introduces an algorithm with a parallel architecture that can concurrently yield price discovery along with optimal scheduling of resources in both transmission and distribution grids. Although the problem formulation is very detailed, the proposed algorithm is quite complex and thus practical implementation is challenging. In [66], a day-ahead energy market is run by a DNO, using the predictor corrector proximal multiplier method. The DNO maximizes its profits while determining a retail price for the DERs. In the proposed structure, all agents share the same retail price, leading to inefficient pricing; since agents are providing different grid services, they should be compensated based on the need for their specific flexibility during the scheduled period. Many papers such as [67–69] use consensus based approaches. In [67], a consensus-based algorithm is used to discover electricity prices, when neglecting network constraints and congestion limits. In [68], a cooperative energy management algorithm is designed, whereby local prices and generator set points are determined; however, generators must share marginal costs with neighbours in order to determine the global price of energy, reducing privacy in the network. Further, the interpretation of this global price is unclear. Finally, consensus+innovations approaches are used in [69] to address state estimation, economic dispatch, and OPF problems. To utilize the consensus+innovations approach for economic dispatch, the marginal costs at each bus are required to be the same at optimality. This assumption is relaxed for the OPF problem, however only for congested networks. Further, agents are still required to share their marginal costs with neighbours. A discussion on the overall structure of the market and interactions with the WEM are also lacking in these works. For the most part, consensus based approaches do not preserve privacy of constraints and/or prices, reducing applicability for market applications, where even marginal prices should be kept private. It should be noted that distributed approaches have also been employed for general EMS applications (see for example [70–75]). By in large, EMS systems focus on the operational

aspects of the grid, and deal with system monitoring and control, generation and dispatch set points, and energy scheduling. These functions are not concerned with market derivatives for services provided to the grid by DERs. As our focus is on a market design with DERs, a discussion of these papers is beyond the scope and is therefore omitted.

In contrast to previous works, the focus here is to develop a comprehensive retail market with appropriate pricing for DERs in a distribution grid. This is discussed in the next section.

5.4 Proposed Market Structure

The proposed retail market consists of a Distribution System Operator (DSO) which is designated to carry out transactions in a distribution grid through two markets: (1) an energy market for DER scheduling and real-time market settlements (d-LMP) through bilateral contracts utilizing the distributed PAC optimization algorithm; and (2) in an ancillary services market which oversees transactions for alert conditions between primary feeders in the distribution network utilizing a simple optimization based algorithm. The DSO acts like a proactive utility, in that it accepts the LMP as traditionally determined by the WEM, but optimally uses DERs within the distribution network, and only requests services from the WEM for net loads beyond its DER capabilities. In both markets, DERs are compensated based on the need for their specific service.

A schematic of both the physical layer and the market layer of the proposed retail market is shown in Fig. 5-10. All data communication coming from the WEM is shown with purple arrows, while data communication moving upstream through the DSO is shown with orange arrows.

The DSO is composed of DSO Representatives (DSO_r) and DSO Workers (DSO_w), located at each primary feeder and substation, respectively. The energy market transactions occur at each primary feeder, between the DSO_r and agents representing DERs at each bus of that feeder. The ancillary market transactions occur between primary

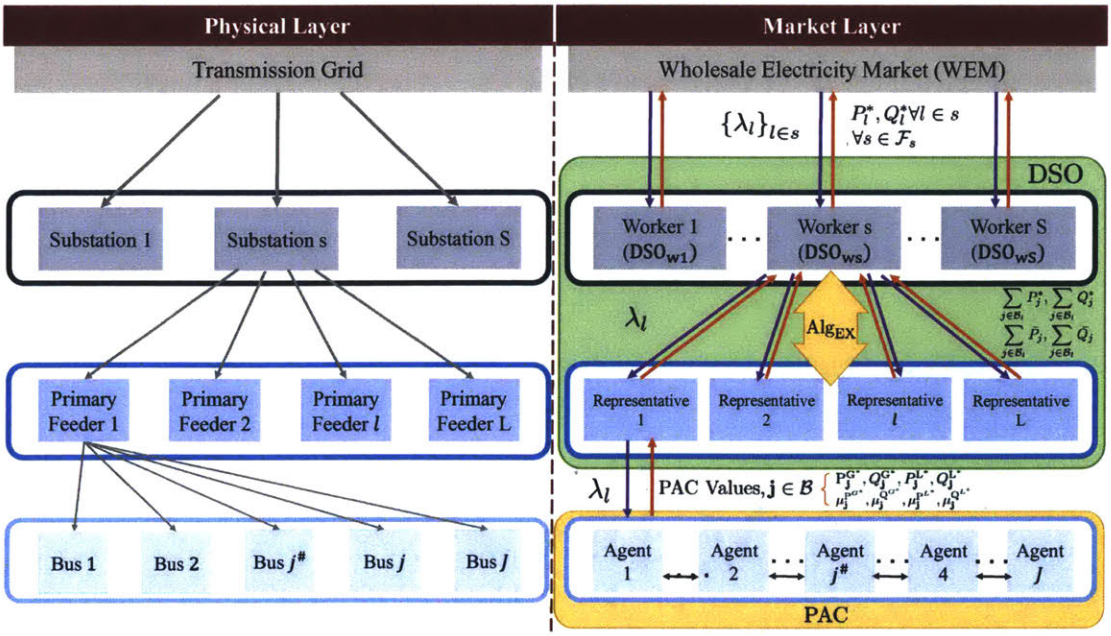


Figure 5-10: Proposed retail market structure

feeders and substations. The DERs are assumed to live at the primary feeder level; any DERs and uncontrollable loads at the secondary feeder level and below are represented through aggregators and do not directly participate in this retail market. Thus, each bus **j** in the physical layer is an independent agent in the market layer, acting to minimize its expenses while subjected to network constraints [8]. Each agent is also an atom **j** in the PAC algorithm, which specifies how these agents interact with one another [9]. Through the PAC algorithm, agents securely share information so as to converge to the optimal schedules and retail prices (d-LMPs) for each DER. These converged values are the basis of a bilateral agreement between the DSO_r and each agent. The DSO_r is responsible for making payments to the DERs, and charging all agents for their consumption (net their generation).

The DSO_r will then aggregate all data from its agents, and send the aggregated load and generation information to its respective DSO_w. Prior to the next WEM

[8]Note that this is not a limitation of the PAC algorithm or the market design. Any DERs in the primary feeder can also be aggregated to form a larger agent, but for simplicity, a one-to-one mapping is used here.

[9]Again, the one-to-one mapping is done for simplicity and proof-of-concept. In reality, a group of agents can be aggregated to form a single atom, but the system model would then move away from the a purely physical model to a more abstracted network model.

clearing, and under normal grid operation, the $\mathrm{DSO_w}$ will convey the net load (or generation) to the WEM. The assumption is that the WEM will accept to serve this net load (equivalently, accept to receive this net generation). The $\mathrm{DSO_w}$ is also responsible for making payments to (or accepting payments from) the WEM for power imported from (or exported to) the transmission grid.

The ancillary services market deals with alert system conditions. Each agent must notify its status of operation to the $\mathrm{DSO_r}$, which aggregates this data and flags any risks of service disruption. In turn, the $\mathrm{DSO_r}$ must notify its $\mathrm{DSO_w}$, which monitors the status of all its feeders. Then, the $\mathrm{DSO_w}$ will carry out an "emergency procedure", denoted as $\mathrm{Alg_{EX}}$, to dispatch ancillary units consisting of distributed generators and fast-acting demand response loads, within its feeders. For example, suppose feeder 1 receives an alert status from one of its agents who will no longer be able to generate the amount of power it had committed. Then, through $\mathrm{Alg_{EX}}$, the $\mathrm{DSO_w}$ will find, among the L connected feeders (in $\mathcal{F}_s$), the lowest cost generators to make up the shortfall, or loads to reduce consumption and redirect power to feeder 1.

The details pertaining to both of these market structures will be discussed in the next subsections.

5.4.1 Energy Market

The energy market employs the PAC algorithm to determine the optimal scheduling of $P_{\mathbf{j}}^{\mathrm{G}^*}$,$Q_{\mathbf{j}}^{\mathrm{G}^*}$ of the generating units, $P_{\mathbf{j}}^{\mathrm{L}^*}$,$Q_{\mathbf{j}}^{\mathrm{L}^*}$ of the loads, and their d-LMPs $\mu_{\mathbf{j}}^{\mathrm{P}^*}$, $\mu_{\mathbf{j}}^{\mathrm{Q}^*}$, at node $\mathbf{j}$, for all $\mathbf{j}$ in the distribution grid. For this application, we consider the BF

model of the power grid, and can state the central OPF problem as follows:

$$\min_{\mathbf{y}\in\mathbb{R}^{|\mathbf{y}|}} \; f^{\text{OPF}}(\mathbf{y})$$

subject to:

$$\begin{aligned}
&\underline{P^{\text{G}}_{\mathbf{j}}} \le P^{\text{G}}_{\mathbf{j}} \le \overline{P^{\text{G}}_{\mathbf{j}}}, && \forall \mathbf{j} \in \mathcal{B} \\
&\underline{P^{\text{L}}_{\mathbf{j}}} \le P^{\text{L}}_{\mathbf{j}} \le \overline{P^{\text{L}}_{\mathbf{j}}}, && \forall \mathbf{j} \in \mathcal{B} \\
&\underline{Q^{\text{G}}_{\mathbf{j}}} \le Q^{\text{G}}_{\mathbf{j}} \le \overline{Q^{\text{G}}_{\mathbf{j}}}, && \forall \mathbf{j} \in \mathcal{B} \\
&\underline{Q^{\text{L}}_{\mathbf{j}}} \le Q^{\text{L}}_{\mathbf{j}} \le \overline{Q^{\text{L}}_{\mathbf{j}}}, && \forall \mathbf{j} \in \mathcal{B} \\
&\underline{v_{\mathbf{j}}} \le v_{\mathbf{j}} \le \overline{v_{\mathbf{j}}}, && \forall \mathbf{j} \in \mathcal{B} \\
&P^2_{\mathbf{i},\mathbf{j}} + Q^2_{\mathbf{i},\mathbf{j}} \le \overline{S}^2_{\mathbf{i},\mathbf{j}}, && \forall (\mathbf{i},\mathbf{j}) \in \mathcal{T} \\
&P^2_{\mathbf{i},\mathbf{j}} + Q^2_{\mathbf{i},\mathbf{j}} \le v_{\mathbf{i}} l_{\mathbf{i},\mathbf{j}}, && \forall (\mathbf{i},\mathbf{j}) \in \mathcal{T} \\
&0 \le l_{\mathbf{i},\mathbf{j}} \le \overline{S}^2_{\mathbf{i},\mathbf{j}} / \overline{v}_{\mathbf{i}}, && \forall (\mathbf{i},\mathbf{j}) \in \mathcal{T} \\
&v_{\mathbf{j}} = v_{\mathbf{i}} - 2\left(R_{\mathbf{i},\mathbf{j}} P_{\mathbf{i},\mathbf{j}} + X_{\mathbf{i},\mathbf{j}} Q_{\mathbf{i},\mathbf{j}}\right) + \left(R^2_{\mathbf{i},\mathbf{j}} + X^2_{\mathbf{i},\mathbf{j}}\right) l_{\mathbf{i},\mathbf{j}}, && \forall (\mathbf{i},\mathbf{j}) \in \mathcal{T} \\
&P^{\text{G}}_{\mathbf{j}} - P^{\text{L}}_{\mathbf{j}} = -P_{\mathbf{i},\mathbf{j}} + R_{\mathbf{i},\mathbf{j}} l_{\mathbf{i},\mathbf{j}} + \textstyle\sum_{\mathbf{k}:(\mathbf{j},\mathbf{k})\in\mathcal{T}} P_{\mathbf{j},\mathbf{k}}, && \forall (\mathbf{i},\mathbf{j}) \in \mathcal{T} \\
&Q^{\text{G}}_{\mathbf{j}} - Q^{\text{L}}_{\mathbf{j}} = -Q_{\mathbf{i},\mathbf{j}} + X_{\mathbf{i},\mathbf{j}} l_{\mathbf{i},\mathbf{j}} + \textstyle\sum_{\mathbf{k}:(\mathbf{j},\mathbf{k})\in\mathcal{T}} Q_{\mathbf{j},\mathbf{k}}, && \forall (\mathbf{i},\mathbf{j}) \in \mathcal{T} \\
&P^{\text{G}}_{\mathbf{i}} - P^{\text{L}}_{\mathbf{i}} = \textstyle\sum_{\mathbf{k}:(\mathbf{i},\mathbf{k})\in\mathcal{T}} P_{\mathbf{i},\mathbf{k}}, && \forall \mathbf{i} \in \mathcal{F} \\
&Q^{\text{G}}_{\mathbf{i}} - Q^{\text{L}}_{\mathbf{i}} = \textstyle\sum_{\mathbf{k}:(\mathbf{i},\mathbf{k})\in\mathcal{T}} Q_{\mathbf{i},\mathbf{k}}, && \forall \mathbf{i} \in \mathcal{F}
\end{aligned} \tag{5.1}$$

where the cost function is defined as:

$$f^{\text{G-Cost}}_{\mathbf{j}}(\mathbf{y}) = \begin{cases} \beta^{\text{PG}}_{\mathbf{j}} (P^{\text{G}}_{\mathbf{j}})^2 + \beta^{\text{QG}}_{\mathbf{j}} (Q^{\text{G}}_{\mathbf{j}})^2, & \text{if } \mathbf{j} \in \tilde{\mathcal{B}} \triangleq \mathcal{B} \setminus \mathcal{F} \\ \lambda^{\text{P}}_{\mathbf{j}} P^{\text{G}}_{\mathbf{j}} + \lambda^{\text{Q}}_{\mathbf{j}} Q^{\text{G}}_{\mathbf{j}}, & \text{if } \mathbf{j} \in \mathcal{F} \end{cases} \tag{5.2}$$

$$f^{\text{L-Util}}_{\mathbf{j}}(\mathbf{y}) = \begin{cases} -\beta^{\text{PL}}_{\mathbf{j}} (P^{\text{L}}_{\mathbf{j}} - P^{\text{L0}}_{\mathbf{j}})^2 - \beta^{\text{QL}}_{\mathbf{j}} (Q_{\mathbf{j}} - Q^{\text{L0}}_{\mathbf{j}})^2, & \text{if } \mathbf{j} \in \tilde{\mathcal{B}} \\ 0, & \text{if } \mathbf{j} \in \mathcal{F} \end{cases} \tag{5.3}$$

$$f^{\text{Loss}}_{\mathbf{i},\mathbf{j}}(\mathbf{y}) = R_{\mathbf{i},\mathbf{j}} l_{\mathbf{i},\mathbf{j}}, \;\; \text{if } (\mathbf{i},\mathbf{j}) \in \mathcal{T} \tag{5.4}$$

$$f^{\text{OPF}}(\mathbf{y}) = \sum_{\mathbf{j}\in\mathcal{B}} \Big[f^{\text{G-Cost}}_{\mathbf{j}}(\mathbf{y}) - f^{\text{L-Util}}_{\mathbf{j}}(\mathbf{y}) \Big] + \xi \Big[\sum_{(\mathbf{i},\mathbf{j})\in\mathcal{T}} f^{\text{Loss}}_{\mathbf{i},\mathbf{j}}(\mathbf{y}) \Big]. \tag{5.5}$$

Here, $f^{\text{G-Cost}}_{\mathbf{j}}$ corresponds to the generation cost function of distributed generators, $f^{\text{L-Util}}_{\mathbf{j}}$ to the disutility function of the flexible consumers capturing their dissatisfac-

tion for curtailing their consumption, $f_{\mathbf{i},\mathbf{j}}^{\text{Loss}}$ to the power losses cost function where losses are due to the line resistances, and f^{OPF} to the aggregate cost function of the OPF problem.

Since the BF model is being used, only distribution grids with radial topology and balanced networks are considered. These can be represented as a directed graph $\mathbf{\Gamma}_{\text{D}} = \langle \mathcal{B}, \mathcal{T}_{\text{D}} \rangle$, where $\mathcal{B}$ represents the graph vertices and $\mathcal{T}_{\text{D}} \subseteq \mathcal{T}$ the graph directed edges. The PAC decomposition for this OPF problem using the BF model depends on the type of bus (i.e. feeder, end node). In particular, if $(\mathbf{i},\mathbf{j}), \{(\mathbf{j},\mathbf{h_j})\}_{\mathbf{j}\in[n]} \in \mathcal{T}_{\text{D}}$, the vector $\mathbf{y_j}$ would be:

$$\mathbf{y_j} = \left[P_{\mathbf{j}}^{\text{G}}; P_{\mathbf{j}}^{\text{L}}; Q_{\mathbf{j}}^{\text{G}}; Q_{\mathbf{j}}^{\text{L}}; v_{\mathbf{j}}; v_{\mathbf{i}}; P_{\mathbf{i},\mathbf{j}}; Q_{\mathbf{i},\mathbf{j}}; l_{\mathbf{i},\mathbf{j}}; \left\{P_{\mathbf{j};\mathbf{h_j}}; Q_{\mathbf{j},\mathbf{h_j}}\right\}_{\mathbf{j}\in[n]}\right]$$

If only $(\mathbf{i},\mathbf{j}) \in \mathcal{T}_{\text{D}}$ (end node) by :

$$\mathbf{y_j} = \left[P_{\mathbf{j}}^{\text{G}}; P_{\mathbf{j}}^{\text{L}}; Q_{\mathbf{j}}^{\text{G}}; Q_{\mathbf{j}}^{\text{L}}; v_{\mathbf{j}}; v_{\mathbf{i}}; P_{\mathbf{i},\mathbf{j}}; Q_{\mathbf{i},\mathbf{j}}; l_{\mathbf{i},\mathbf{j}}\right]$$

If only $\{(\mathbf{j},\mathbf{h_j})\}_{\mathbf{j}\in[n]} \in \mathcal{T}_{\text{D}}$ (feeder node) by :

$$\mathbf{y_j} = \left[P_{\mathbf{j}}^{\text{G}}; P_{\mathbf{j}}^{\text{L}}; Q_{\mathbf{j}}^{\text{G}}; Q_{\mathbf{j}}^{\text{L}}; v_{\mathbf{j}}; \left\{P_{\mathbf{j};\mathbf{h_j}}; Q_{\mathbf{j},\mathbf{h_j}}\right\}_{\mathbf{j}\in[n]}\right]$$

where $\mathbf{h_j}$ denotes the set of downstream buses connected to bus $\mathbf{j}$.

Then as per the PAC decomposition presented in Chapter 3, the dependent variables between atoms must be accounted for, in the form of variable copies, which are the estimate an atom $\mathbf{j}$ makes of a dependency on atom $\mathbf{i}$. In the BF model, the dependencies are the line flow variables and the nodal voltages; namely, atom $\mathbf{j}$ makes a copy of the downstream power flows $\{P_{\mathbf{j},\mathbf{h}}; Q_{\mathbf{j},\mathbf{h}}\}$, and the nodal voltage of the parent $v_{\mathbf{i}}$. Thus the global problem which can be compactly written in matrix form for the equality and inequality constraints as in (3.43)-(3.45), can be decomposed to the following atomized OPF problem, where the last three constraints are the coordination constraints resulting from the atomization of the dependencies.

$$\min_{\mathbf{a}} \sum_{\mathbf{j}\in\mathcal{B}} f_{\mathbf{j}}\left(\mathbf{a_j}\right)$$

subject to:

$$\begin{aligned}
&\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a_j} = \mathbf{0}, && \mathbf{j} \in \mathcal{B} \\
&\mathbf{H_j}\left(\mathbf{a_j}\right) \leq \mathbf{0}, && \mathbf{j} \in \mathcal{B} \\
&\tilde{v}_{\mathbf{i}}^{[\mathbf{j}]} - v_{\mathbf{i}} = 0, && \forall \mathbf{i} \in \mathcal{B} \\
&\tilde{P}_{\mathbf{j,h}}^{[\mathbf{j}]} - P_{\mathbf{j,h}} = \mathbf{0}, && \forall (\mathbf{j},\mathbf{h}) \in \mathcal{T} \\
&\tilde{Q}_{\mathbf{j,h}}^{[\mathbf{j}]} - Q_{\mathbf{j,h}} = 0, && \forall (\mathbf{j},\mathbf{h}) \in \mathcal{T}
\end{aligned} \tag{5.6}$$

Returning to the structure of the energy market, we denote ΔT and $\Delta\tau$ to be the market clearing periods of the WEM and the retail market, respectively. For ease of exposition, we assume that $\Delta\tau \ll \Delta T$, so the DSO market clears multiple times within a single WEM clearance period. However, in general, these two clearing periods can be chosen independently.

We make the following additional assumption: Between any two market clearings T_0 and $T_0 + \Delta T$ of the WEM, the LMP of all feeder buses $\lambda_{\mathbf{l}}^{\mathrm{P}}$ is fixed, for all feeders in a substation. This is denoted compactly as "$\lambda_{\mathbf{l}}^{\mathrm{P}}$ is fixed, $\forall \mathbf{l} \in \mathcal{F}_s, \forall \mathbf{s} \in \mathcal{S}$". Thus, $\lambda_{\mathbf{l}}^{\mathrm{P}}$ is a boundary condition $\forall \mathbf{l} \in \mathcal{F}_s, \forall \mathbf{s} \in \mathcal{S}$ on the scheduling problem, for the N DSO clearings over the time period $[T_0, T_0 + \Delta T]$.

Note that in PAC, the feeder node $j^{\#}$ is treated as a generator of infinite capacity ($\overline{P_{j^{\#}}^{\mathrm{G}}} \gg 0$, where $\overline{P_{j^{\#}}^{\mathrm{G}}}$ corresponds to the upper limit of power generation) – thus, $P_{j^{\#}}^{\mathrm{G}^*}$ represents power imported from the main grid. Likewise, $P_{j^{\#}}^{\mathrm{L}^*}$ represents power exported to the main grid. Depending on whether the power is imported or exported, either $P_{j^{\#}}^{L^*} = 0$ or $P_{j^{\#}}^{G^*} = 0$, respectively. Then for time period $[T_0, T_0 + \Delta T]$ the average load (generation) of the distribution grid which will be imported from (exported to) the main grid is calculated as $P_{\mathbf{l},T_0}^{*} = \frac{1}{N}\sum_{n=1}^{N} P_{j^{\#},n}^{\mathrm{G}^*} - P_{j^{\#},n}^{\mathrm{L}^*}$, $\forall \mathbf{l} \in \mathcal{F}_s, \forall \mathbf{s} \in \mathcal{S}$.

With the above assumptions in place, the operation of the energy market can now be discussed. The proposed market determines the generation, consumption, and prices at each $j \in \mathcal{B}$ in the distribution grid, using the PAC algorithm, over every

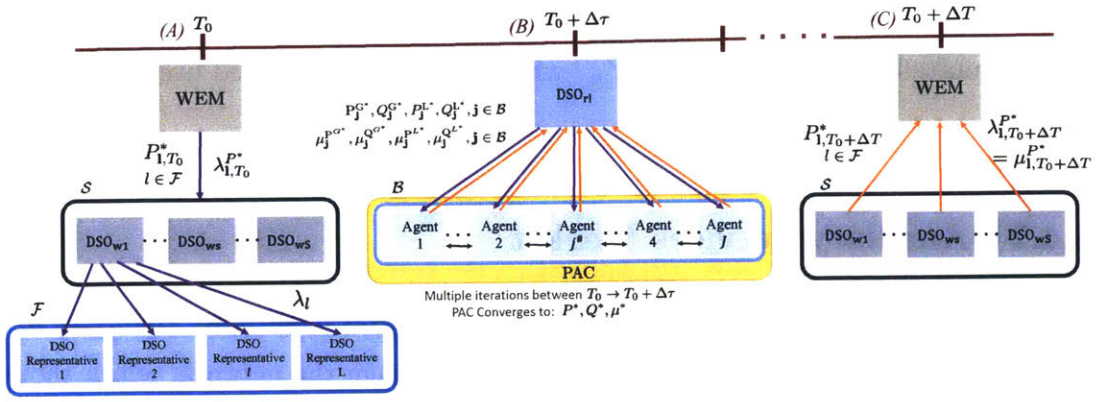


Figure 5-11: Timeline of energy market layer interactions

DSO market clearing period $\Delta\tau$. Note that the PAC algorithm is run independently for each feeder, for a single time period.

Fig. 5-11 shows the interactions between the WEM, DSO, and agents for the proposed energy market over one clearing of the WEM. The energy retail market will consist of three interactions denoted as (A), (B), and (C). The first interaction (A) occurs between the WEM and DSO_ws, when the WEM clears: at time T_0 the WEM passes to each DSO_w the forecasted load/generation over the period, and the LMPs for each of its DSO_rs, $\mathbf{l} \in \mathcal{F}_s$, $\forall \mathbf{s} \in \mathcal{S}$. This information is passed down to each DSO_r, which shares the LMP with its agents. The second set of interactions (B) occur between the DSO_r and the agents every $\Delta\tau$, prior to the next WEM clearing. For each interaction (B), the agents will coordinate amongst each other, through the PAC algorithm, and converge in some K iterations. In doing so, the converged values of the real and reactive power generation and load for each bus, $P^{G^*}_{\mathbf{j}}, Q^{G^*}_{\mathbf{j}}, P^{L^*}_{\mathbf{j}}, Q^{L^*}_{\mathbf{j}}$, in the distribution grid will be determined, along with their respective d-LMPs, $\mu^*_{\mathbf{j}}$. The agents will enter into bilateral contracts with the DSO_r with these scheduled values. The d-LMPs are valid for the period $\Delta\tau$. This interaction repeats until the next WEM clearing. Prior to the next clearing of the WEM, the DSO_r will send to the DSO_w the load to be imported from (exported to) the main grid, $P^*_{\mathbf{l},T_0}$, which has been aggregated over the N DSO clearings. The last interaction (C) from Figure 5-11, is between the WEM and the DSO_ws, and occurs at the next clearing of the WEM,

$T_0 + \Delta T$. Each DSO_w will send to the WEM per each feeder, the load to be imported from (exported to) the main grid, P^*_{l,T_0}, which has been aggregated over the N DSO clearings.

5.4.2 Ancillary Services Market

The sections above detail the market under normal grid operation. However, with increased DER penetration comes more flexibility within the network, which should be leveraged to address alert conditions at lower costs. Under the proposed market structure, the ancillary services market is employed when an alert status from a DSO_r is received. The ancillary market addresses the shortfall in generation or load reduction. The alert code, code_a, details whether a generator or a demand response resource is no longer capable of generating or reducing demand as committed in the market period, respectively. Once received, the DSO_w runs the Alg_EX procedure, whereby the optimization problem in (5.7) is solved to meet the shortfall in feeder x, ΔP_x, while minimizing the cost of deploying ancillary resources. In doing so, the DSO_w leverages the distributed network structure of the connected feeders.

$$
\begin{aligned}
&\min \ \textstyle\sum_{\text{l}\in\mathcal{F}_s} \left(\mu_\text{l}^{\text{P,G}'} P_\text{l}^{\text{G}'} + \mu_\text{l}^{\text{P,L}'} P_\text{l}^{\text{L}'}\right) \\
&\text{subject to:} \\
&\textstyle\sum_{\text{l}\in\mathcal{F}_s} \left(P_\text{l}^{\text{G}'} + P_\text{l}^{\text{L}'}\right) = \min\left(\Delta P_x, P_x^{\text{G}^*}\right) \\
&P_\text{l}^{\text{G}'} \leq \overline{P_\text{l}^{\text{G}'}} && \forall \text{l} \in \mathcal{F}_s \\
&P_\text{l}^{\text{L}'} \leq \overline{P_\text{l}^{\text{L}'}} && \forall \text{l} \in \mathcal{F}_s \\
&P_\text{l}^{\text{G}'}, P_\text{l}^{\text{L}'} \geq 0 && \forall \text{l} \in \mathcal{F}_s
\end{aligned}
\tag{5.7}
$$

where $P_\text{l}^{\text{G}'}$ is the additional generation required from feeder l, and $P_\text{l}^{\text{L}'}$ is the additional load reduction required from feeder l in order to meet the shortfall in feeder x.

In order to run Alg_EX, the DSO_w requires the following data from each of its DSO_rs, regarding their respective distribution networks:

- Maximum generating capacity: $\overline{P_\text{l}^{\text{G}}} = \sum_{\text{j}\in\mathcal{B}_\text{l}\setminus j^\#} \overline{P_\text{j}^{\text{G}}}$;

- Minimum load requirement: $\underline{P_{\mathrm{l}}^{\mathrm{L}}} = \sum_{\mathrm{j}\in\mathcal{B}_{\mathrm{l}}} \underline{P_{\mathrm{j}}^{\mathrm{L}}}$;
- Optimal generation set point, as determined by PAC and committed in the energy market: $P_{\mathrm{l}}^{\mathrm{G}^*} = \sum_{\mathrm{j}\in\mathcal{B}_{\mathrm{l}}\setminus j^{\#}} P_{\mathrm{j}}^{\mathrm{G}^*}$;
- Optimal load set point, as determined by PAC and committed in the energy market: $P_{\mathrm{l}}^{\mathrm{L}^*} = \sum_{\mathrm{j}\in\mathcal{B}_{\mathrm{l}}} P_{\mathrm{j}}^{\mathrm{L}^*}$;
- Weighted average cost of local distributed generation: $\mu_{\mathrm{l}}^{\mathrm{P,G}'} = \frac{\sum_{\mathrm{j}\in\mathcal{B}_{\mathrm{l}}\setminus j^{\#}} \mu_{\mathrm{j}}^{\mathrm{P}^*} \overline{P_{\mathrm{j}}^{\mathrm{G}}}}{\sum_{\mathrm{j}\in\mathcal{B}_{\mathrm{l}}\setminus j^{\#}} \overline{P_{\mathrm{j}}^{\mathrm{G}}}}$;
- Weighted average cost of using demand response: $\mu_{\mathrm{l}}^{\mathrm{P,L}'} = \frac{\sum_{\mathrm{j}\in\mathcal{B}_{\mathrm{l}}\setminus j^{\#}} \mu_{\mathrm{j}}^{\mathrm{P}^*} \left(\overline{P_{\mathrm{j}}^{\mathrm{L}}} - \underline{P_{\mathrm{j}}^{\mathrm{L}}}\right)}{\sum_{\mathrm{j}\in\mathcal{B}_{\mathrm{l}}\setminus j^{\#}} \left(\overline{P_{\mathrm{j}}^{\mathrm{L}}} - \underline{P_{\mathrm{j}}^{\mathrm{L}}}\right)}$;
- Alert information: $[\mathrm{code_a}\ \ \Delta P_x]$, where ΔP_x is the change in power quantity - either a decrease in generation capabilities, or a decrease in demand response capabilities (i.e. increase in $\underline{P_{\mathrm{j}}^{\mathrm{L}}}$ for some j in feeder x). We note $\Delta P_x > 0$.

The $\mathrm{Alg_{EX}}$ procedure is described next for generation shortfall, and a similar procedure is carried out for load reduction. For ease of exposition, suppose $\mathrm{DSO_r}$ 1 sends an alert code of 0 with a shortfall quantity of ΔP_{l}. The new maximum generation capacity for the feeder is then $\overline{P_{\mathrm{l},\Delta}^{\mathrm{G}}} = \overline{P_{\mathrm{l}}^{\mathrm{G}}} - \Delta P_{\mathrm{l}}$. Under this shortfall, the committed generation must also be adjusted, $P_{\mathrm{l},\Delta}^{\mathrm{G}^*} = P_{\mathrm{l}}^{\mathrm{G}^*} - \Delta P_{\mathrm{l}}$. The DR capabilities remain the same. Finally, the new uncommitted generation and DR limits for the feeder can be calculated as $\overline{P_{\mathrm{l}}^{\mathrm{G}'}} = \overline{P_{\mathrm{l},\Delta}^{\mathrm{G}}} - P_{\mathrm{l},\Delta}^{\mathrm{G}^*}$ and $\overline{P_{\mathrm{l}}^{\mathrm{L}'}} = P_{\mathrm{l}}^{\mathrm{L}^*} - \underline{P_{\mathrm{l}}^{\mathrm{L}}}$ respectively.

5.5 Validation of Proposed Market

The proposed market design must be evaluated. First, the PAC algorithm must be evaluated for privacy considerations. In a market system where individual agents (DER owners) may have sensitive information they do not wish to disclose, privacy is a key requirement within any computational framework. Second, the computational performance of the distributed PAC algorithm as applied to the OPF problem for retail market must be evaluated. The algorithm must converge within the $\delta\tau$ time, while accounting for computational time and potential latencies in the communication

network. Third, the economic performance of the market must be evaluated. At a minimum, we desire an increased DSO revenue which reflects better utilization of resources within the distribution grid. With better resource utilization, we also expect lower electricity prices for customers [10]. This can manifest in two main ways: the retail prices from the proposed market mechanism can be lower than fixed utility retail prices (i.e. $\mu_{\mathbf{j}}^{\mathrm{P}^*} < \lambda_{\mathrm{retail}}^{\mathrm{P}}$), or higher profit for the DSO will result in lower energy costs for all consumers in the market over time. These three areas of validation will be further explored.

5.5.1 Validation Metric 1: Privacy of the PAC Algorithm

The PAC algorithm is privacy preserving, thus satisfying the first requirement for use in a retail market. The cost functions of the DERs $f_j(\mathbf{a}_j)$, the retail prices $\mu_{\mathbf{j}}$, and the coordination costs $\nu_{\mathbf{j}}$, all remain private to each atom $\mathbf{j}$. The only outside variables needed at each $\mathbf{j}$ are $\mathbf{a}$ of the neighbours, and $\tilde{\nu}_{\mathbf{j}}$. Comparing PAC briefly to another popular distributed optimization algorithm, dADMM (distributed Alternating Direction Method of Multipliers), we see that PAC is a more private algorithm. Namely, both the cost functions and $\mu_{\mathbf{j}}$ remain private to each atom for both the PAC and dADMM algorithms. However, unlike dADMM, the dual variable $\nu_{\mathbf{j}}$ remains private to atom $\mathbf{j}$ in PAC.

The dual variable $\nu_{\mathbf{j}}$ can be thought of as a coordination price. This indicates how far an atom $\mathbf{j}$ must deviate from its atomic optimal point (which it would operate at when making decisions as an independent agent) in order to achieve coordination by taking into account network constraints. This latter case, when an agent accounts for network constraints, is required for efficient and lowest-cost resource use, and is enabled through the privacy-preserving communication requirements of the PAC algorithm. If $\nu_{\mathbf{j}}$ is high, then atom $\mathbf{j}$ is further from its atomic optimal point âĂŞ thus, $\nu_{\mathbf{j}}$ describes the tension between atomic and network optimality.

[10] We expect lower electricity prices *on average*. Note that in a real-time market, prices can fluctuate and can peak at high values during periods of lower generation and high consumption. However, better resource utilization can help reduce these peak values while also reducing average electricity prices.

The importance of privacy of ν_j can be explained thus: with knowledge of the ν_j values, a rogue agent can carry out a cyberattack, deliberately manipulating the variables that are communicated between atoms to be maximally different from one another. This action has two implications: First, it will take longer to converge. Second, since the values are maximally different, each atom **j** will be forced further away from its atomic optimal point, incurring "losses" as it tries to coordinate with its neighbors to achieve optimality. The losses can be thought of as an opportunity cost: to meet network constraints, atom **j** has to sacrifice revenue it could have received had it been operating closer to atomic optimality. We note that there may be more targeted ways to use the **j** values, such as by agents to coerce neighbors into certain behaviours. Detailed investigation into this, however, is beyond the scope of this thesis. Thus, by keeping ν_j private to each atom, the PAC algorithm has added privacy characteristics and may exhibit more resiliency to cyberattacks than dADMM (and other distributed algorithms or architectures). Thus, the PAC algorithm protects the interests of each agent, while ensuring no agent has enough information (of d-LMPs or coordination prices) to exercise market power, or to destabilize the market and prevent it from reaching equilibrium.

5.5.2 Validation Metric 2: Computational Performance of the PAC Algorithm

The PAC algorithm was then tested to ensure satisfactory performance for use in a retail market. In particular, we require the PAC algorithm to converge within the retail market clearing period [11] which we propose to be shorter than (or at most equivalent to) the WEM clearing period.

Initial numerical simulations were conducted on a 2.6GHz CPU Intel Core i7-5600U PC using MATLAB and CVX. To improve performance, the algorithm was then implemented using MATLAB, with optimization problems being setup using the YALMIP interface, and solved directly with Gurobi Optimizer (similar to the setup in

[11] which can be anywhere between 1-15 minutes depending on the ISO/RTO requirements

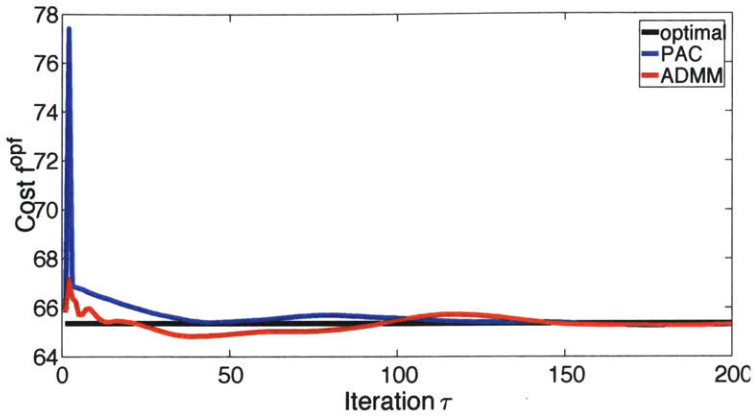


Figure 5-12: Cost evolution with PAC and dADMM

Chapter 3. These simulations were conducted on a 3.00GHz CPU Intel Xeon E5-1660 v3, 8 core workstation, with 128GB RAM. The analysis below was conducted for the JST-CREST 126 Distribution Feeder Model (see Appendix B.4 for complete data).

Preliminary Analysis

The parameters ρ, γ_j, and $\hat{\gamma}_j$ of the PAC algorithm were tuned to guarantee convergence of the PAC algorithm (for more details see [20, 21]) and the vector $\mathbf{a}$ was initiated at the optimal solution of the atomized central OPF problem, at $i = 19$h. The PAC algorithm was compared with the popular dADMM [76,77] algorithm. Both algorithms were run for 200 iterations, and results are shown in Fig. 5-12, where the black graph shows the optimal cost obtained using a centralized optimization solver. Both algorithms converge to a near-optimal value in roughly 150 iterations, with the PAC algorithm converging slightly faster. From the initial implementation using MATLAB/CVX, the total time required to complete 200 iterations with PAC is 130s (0.65s/iteration/atom) and with dADMM is 203s (1.01s/iteration/atom). Hence, when the initial conditions are initiated close to the optimal solution, the proposed PAC algorithm can converge in less than 5 minutes which is the standard time between two WEM clearings[12].

[12]for the ISO-NE WEM market

Implementation Considerations

The improved implementation considered the fact that the grid topology can be assumed to be fixed; thus, the constraint matrices, $\tilde{\mathbf{G}}_{\mathbf{j}}$ and $\mathbf{A}_{\mathbf{j},-}$ in the PAC algorithm are constant. Only the cost function (augmented Lagrangian) must be updated for each PAC iteration, and the inequality constraints (limits on power generation and load) updated for each time period.

Similar to in Chapter 3, the computational setup used to run the market simulations is provided in Figure 5-13. We note the main difference is that the PAC decomposition does not use the Yalmip package to setup the centralized model, and the Benchmarking is an optional step which compares the PAC algorithm to the dADMM algorithm (while plotting the centralized solution). Finally, there is no pre-processing step to recover current bounds, as the power flow model used for the markets work is the BF model. We note again, that the PAC algorithm is implemented in MATLAB, where a `parfor` loop is used to parallelize the primal update across the computational agents. Once the primal update - which consists of solving the atomic OPF for the k^{th} iteration - is complete, the dual variables are updated in a centralized manner. Note that this is not a limitation of the PAC algorithm, but rather a design choice to simplify the implementation.

For worst-case computational time, the vector $\mathbf{a}$ was initiated at 'zero' - all node variables are initiated at their lower bounds ($\underline{P^{\mathrm{G}}_{\mathbf{j}}}, \underline{P^{\mathrm{L}}_{\mathbf{j}}}, \underline{Q^{\mathrm{G}}_{\mathbf{j}}}, \underline{Q^{\mathrm{L}}_{\mathbf{j}}}, \underline{V_{\mathbf{j}}}$); all line variables are initiated at 0. To guarantee convergence, we run PAC for 4500 iterations. Tables 5.1 and 5.2 provide a summary of estimated computational times and latencies associated with PAC that would correspond to a parallel implementation. Latency is defined as the time between data generation and when it is acted upon by the next agent [78]. We also differentiate between latency and computational delays, which are times associated with application/script execution [78–80]. References [78–80] were used to obtain reasonable approximations for latencies and computational delays.

These tables show that the retail market completes execution in 8.8 minutes. We expect this time to be reduced by an order of magnitude through better initialization

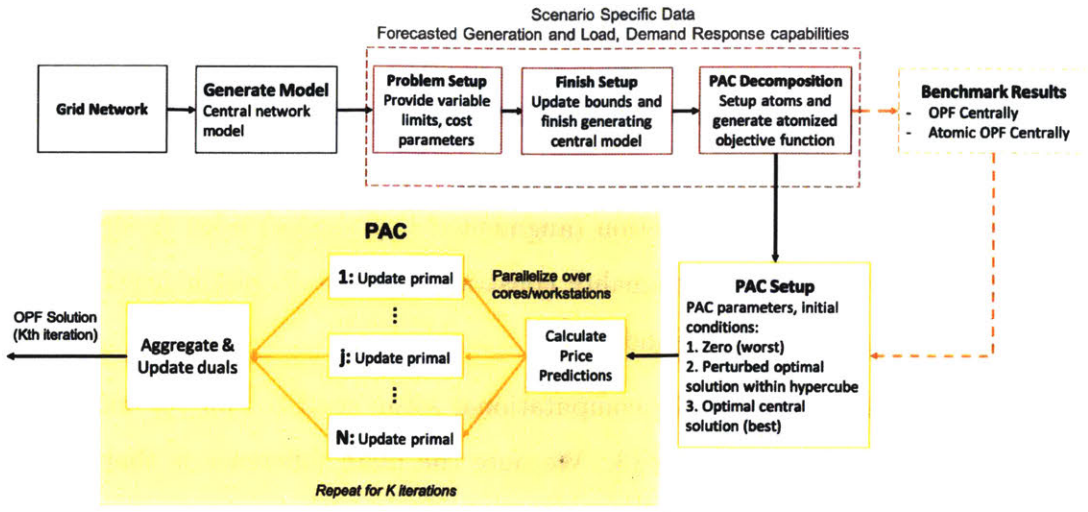


Figure 5-13: Flowchart of the computational process. The PAC algorithm decomposition is fixed for a network with a given topology. The scenario or time-period specific updates must be run each time the PAC algorithm is run.

(not zero). For example, a reduction of iterations from 4500 to 450 will reduce execution time from 8.8 to 1.4 minutes. Execution time can be further reduced as latency requirements reduce, to say 0.76 minutes, for 10ms latency compared to 100ms, at 450 iterations.

In order to implement the distributed energy markets, distributed computational units and low latency communication infrastructure within the distribution network is required. Each agent must have a computational node, representing a single or a collection of distribution transformers and/or DERs. The units represented by a single agent will have limited data privacy, and will all have the same d-LMP. Thus, it is in the best interest of each unit to set up its own computing infrastructure. Alternatively, a set of agents can share the same communication infrastructure, under a common security protocol which prevents agents from accessing others' information. Finally, a low latency communication network will be required, to allow the PAC algorithm to rapidly make successive bids, and quickly reach a market equilibrium. By reaching equilibrium faster, the energy market can have a shorter clearing time, allowing different ISOs and RTOs to determine the clearing time which best suits their needs.

SINGLE COMPUTATIONS	Market Agent(s)	Time
Data Processing: aggregating agent data	DSO_r	5
Data Processing: check for 'alert' system status	DSO_w	5
Algorithm Execution: run Alg_{EX}	DSO_w	10
LATENCY REQUIREMENTS	Market Agent(s)	Time
Data communication: at PAC convergence	Agents $\rightarrow$ DSO_r	1
Data communication: before next market clearing	$DSO_r \rightarrow DSO_w$	15

Table 5.1: Computational Delays and Latency Requirements. All times measured in seconds, (s).

PAC COMPUTATION PER ITER	Market Agent(s)	Time	No. Iter	Total Time
Algorithm Execution	Atom	0.0092	4500	41.4473
PAC information exchange (latency)	Atom $j \leftrightarrow$ Atom i	0.1	4499	449.9
Total PAC Computational Time				491.35

Table 5.2: PAC Computational Time. All times measured in seconds, (s).

5.5.3 Validation Metric 3: DSO revenue

The proposed market operation can be validated to increase DSO revenue, as per the following profit structure:

$$\mathcal{P}_{\text{DSO}} = \mathcal{R}^{\text{L}}_{\text{DSO}} - \mathcal{C}^{\text{flex}}_{\text{DSO}} - \mathcal{C}^{\text{DG}}_{\text{DSO}} - \mathcal{C}^{\text{WEM}}_{\text{DSO}} \tag{5.8}$$

with the different components that contribute to the net revenue defined as follows:

- Revenue earned from all loads: $\mathcal{R}^{\text{L}}_{\text{DSO}} = \sum_{j \in \mathcal{B}} \left(\mu^{\text{P}^*}_{\text{j}} P^{\text{L}^*}_{\text{j}} + \mu^{\text{Q}^*}_{\text{j}} Q^{\text{L}^*}_{\text{j}} \right)$
- Payments made to flexible loads (DR units): $\mathcal{C}^{\text{flex}}_{\text{DSO}} = \sum_{j \in \mathcal{B}} \left(\mu^{\text{P}^*}_{\text{j}} \left(P^{\text{L0}}_{\text{j}} - P^{\text{L}^*}_{\text{j}} \right) + \mu^{\text{Q}^*}_{\text{j}} \left(Q^{\text{L0}}_{\text{j}} - Q^{\text{L}^*}_{\text{j}} \right) \right)$
- Payments made to DGs: $\mathcal{C}^{\text{DG}}_{\text{DSO}} = \sum_{j \in \mathcal{B} \setminus j^{\#}} \left(\mu^{\text{P}^*}_{\text{j}} P^{\text{G}^*}_{\text{j}} + \mu^{\text{Q}^*}_{\text{j}} Q^{\text{G}^*}_{\text{j}} \right)$
- Payment made to WEM, for purchasing power: $\mathcal{C}^{\text{WEM}}_{\text{DSO}} = \lambda^{\text{P}}_{\text{l}} P^{\text{G}^*}_{\text{l}} + \lambda^{\text{Q}}_{\text{l}} Q^{\text{G}^*}_{\text{l}}$

To evaluate this net revenue, a benchmark system is needed. Consider a market which consists of a utility that purchases power from the WEM and sells to customers at a fixed retail price, $\lambda^{\text{P}}_{\text{retail}}$. In this system, DERs are assumed to not be compensated for their services, as they are small-scale behind-the-meter resources and not visible

to the utility. Thus, the DSO's profit *without* the retail market is:

$$\mathcal{P}_{\text{no-DSO}} = \mathcal{R}^{\text{L}}_{\text{no-DSO}} - \mathcal{C}^{\text{WEM}}_{\text{no-DSO}} \tag{5.9}$$

where the first term denotes the revenue earned from all loads, given by

$$\mathcal{R}^{\text{L}}_{\text{no-DSO}} = \lambda^{\text{P}}_{\text{retail}} \sum_{j \in \mathcal{B}} P^{\text{L}^*}_{\text{j}} + \lambda^{\text{Q}}_{\text{retail}} \sum_{j \in \mathcal{B}} Q^{\text{L}^*}_{\text{j}} \tag{5.10}$$

and the payment to the WEM is

$$\mathcal{C}^{\text{WEM}}_{\text{no-DSO}} = \lambda^{\text{P}}_{\text{l}} P^{\text{G}^*}_{\text{l}} + \lambda^{\text{Q}}_{\text{l}} Q^{\text{G}^*}_{\text{l}}. \tag{5.11}$$

From (5.8)-(5.9), it follows that the additional revenue earned by the DSO when using the retail market mechanism is given by:

$$\mathcal{P}_{\text{DSO-increase}} = \mathcal{P}_{\text{DSO}} - \mathcal{P}_{\text{no-DSO}} \tag{5.12}$$

Similarly, we can also quantify the savings for an inflexible consumer, j as:

$$\mathcal{P}_{\text{consumer-saving}} = (\lambda^{\text{P}}_{\text{retail}} - \mu^{\text{P}^*}_{\text{j}}) P^{\text{L}^*}_{\text{j}} \tag{5.13}$$

Note that flexible consumers will receive additional compensation in the form of revenue for their services.

The proposed retail market clears every $\Delta\tau$ seconds (or minutes) [13]. A 24-hour period can be grouped into 24 hour-long intervals $T_i = [i, i+1]$, where i corresponds to the ith hour, $i = 0, \ldots, 23$, over this 24 hour period. Suppose that over this T_i, there are N clearings of the retail market, which occur at $t = i + n\Delta\tau$, $n = 1, \ldots, N$, and M clearings of the WEM, which occur at $t = i + m\Delta T$, $m = 1, \ldots, M$. Since $\Delta\tau \ll \Delta T$, it follows that $n > m$ [14]. Denoting $n' = i + n\Delta\tau$ and $m' = i + m\Delta T$, we can then

[13] We propose the retail market clears more frequently than the real-time WEM to leverage the flexibility from the DERs. In particular, renewable generation is more volatile than traditional generation, so a retail market which clears more frequently would be able to react to these fluctuations with prices which better reflect the renewables.

[14] Again, assuming the retail market clears more frequently than the WEM.

represent the profit calculated above as a function of time:

$$\mathcal{P}_{\text{DSO-increase}}(i) = \sum_{n=1}^{N} \Big(\mathcal{P}_{\text{DSO}}(n', m') - \mathcal{P}_{\text{no-DSO}}(n', m') \Big) \tag{5.14}$$

where the n' is carried through to the μ_{j}, P_{j}, and Q_{j}, and the m' is carried through to the λ_{l}, λ_{retail}, and P_{l}.

Case Studies

To validate the market design (both DSO revenue and functionality of the ancillary services market), consider three networks. They will be introduced briefly here, and more detailed information provided in Appendices B.4, B.3.2 and B.2.3.

JST-CREST 126 The original network model, presented in detail in [81], represents a distribution grid in Komae city in Tokyo, Japan with an area of 5 km^2. Here we employ a compressed version of that network, as shown in Appendix B.4. Our model contains $|\mathcal{B}|$ =84 nodes and 6 lines all connected to the feeder node in a star topology. Details of base values, parameters, and information on electrical lines, loads, and generators used in the simulation can be found in Appendix B.4.

The time-dependent ratio $\alpha(i)$ which governs the demand in the network, varies according to the FY2017 average consumption profile in the Tokyo area [82]. The 24-hour profile of $\alpha(\tau)$ is shown in Fig. 5-14 (top). In addition, $v_1 = 1.023$ pu (with nominal value 6750V), and all loads are taken as unity power factor loads. We assume that the reactive power of flexible consumers can be adjusted through proper control of the air-conditioners. Further, we assume the P-LMP, λ_1^{P}, is either fixed at 10 or is time-varying and varies according to the yearly average profile of the FY2017 system price in Japan Electric Power eXchange (JEPX) [83] as shown in Fig. 5-15 (top) The retail price for electricity is fixed at 26 JY/kWh, as per average retail electricity prices in Japan [84, 85].

IEEE 123 The IEEE 123 bus network was modified to be a balanced 3-phase distribution network as further detailed in Appendix B.3.2. Further details on lines, loads, and generators are also provided in Appendix B.3.2.

The time-dependent ratio $\alpha(i)$ is deduced from the ISO-NE report of total recorded electricity demand for each five-minute interval of May 14, 2019 [38], as shown in Fig. 5-14 (middle). Loads are not assumed to be unity power factor. Finally, we assume the P-LMP, $\lambda_{\mathbf{l}}^{\mathrm{P}}$, is either fixed at \$30/MWh or is time-varying and varies according to the ISO-NE report of the final approved LMPs for each five-minute interval for May 14, 2019 [37], as shown in Fig. 5-15 (middle).

IEEE 13 To test the alert case, a multi-feeder network was developed, with $|\mathcal{F}_s| = 3$ and $|\mathcal{B}_{\mathbf{l}}| = 13 \forall \mathbf{l} \in \mathcal{F}_s$. The IEEE 13-bus network [86] was modified to be balanced 3-phase (see Appendix B.2.3). Variations of the network were developed by adding generators of varying capacity at different locations (see Appendix B.2.3), and perturbing the base loads as $\overline{P_{\mathbf{j}}^{\mathrm{L}}} \triangleq \overline{P_{\mathbf{j}}^{\mathrm{L}}}(1 + \delta_{\mathbf{j}})$, where $\delta_{\mathbf{j}} \sim \mathcal{N}(0, 0.15) \forall \mathbf{j} \in \mathcal{B}_{\mathbf{l}}, \forall \mathbf{l} \in \mathcal{F}_s$.

The time-dependent ratio *per feeder*, $\alpha_{\mathbf{l}}(i)$, is obtained by perturbing the ISO-NE demand report [38] from May 14, 2019 as $\alpha_{\mathbf{l}}(i) = \alpha(i)\delta_{\mathbf{l}}$, where $\delta_{\mathbf{l}} \sim \mathcal{N}(0, 0.075)$ and smoothing the resulting profile. The three demand curves are shown in Fig. 5-14 (bottom). The feeder P-LMPs, $\lambda_{\mathbf{l}}^{\mathrm{P}}$, are either fixed at \$30/MWh or are time-varying and varies according to the ISO-NE five-minute LMPs for May 14, 2019 [37], taken from different location IDs for each feeder, $\mathbf{l} \in \mathcal{F}_s$ as shown in Fig. 5-15 (bottom).

The results are benchmarked using the following scenarios: (1) When all the loads in the distribution grid are served only by the main grid, and (2) when the retail market dispatches DERs through the PAC algorithm with the LMP being either (a)fixed or (b)time varying (see Fig. 5-15).

Results from JST-CREST 126

The results from PAC across the JST-CREST distribution grid (see Appendix B.4 for data) are shown in Fig. 5-16. Figure 5-16a shows the voltages across lines A and F are higher than across other lines in the network. This is reasonable as the generation

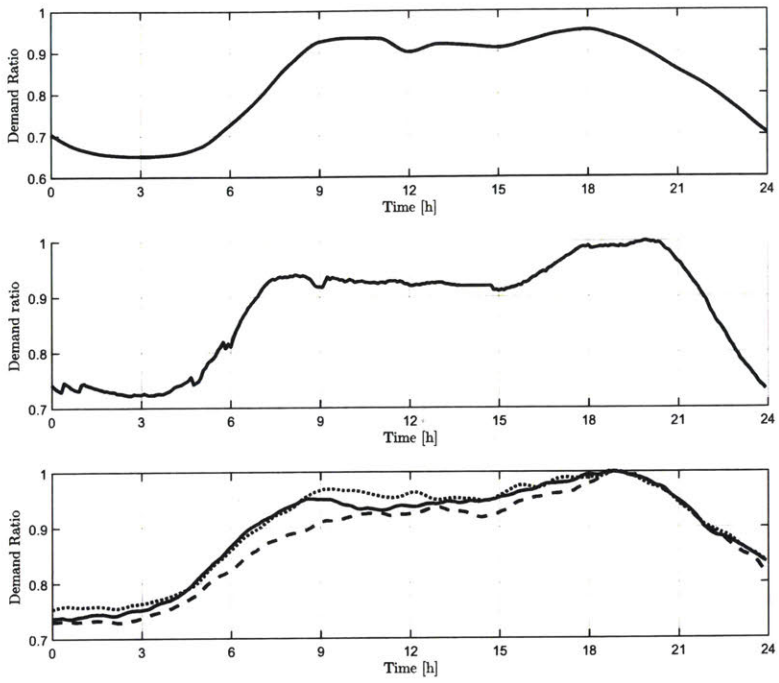


Figure 5-14: The profiles of the time-dependent demand ratio $\alpha(i), i = 0, \ldots, 24$, for Networks JST-CREST 126 (top), IEEE-123 (middle), and IEEE-13 (bottom).

from local generators exceeds the local demand of the loads. Similarly, the voltages across line C are suppressed as the total load connected to line C is higher than the local generated power at node 53. Figure 5-16b shows that the Lagrange multipliers corresponding to the voltage equality constraint, $\mu_{\mathbf{j}}^{V}$, tend to be lower for nodes with higher voltages and vice versa. The local active generation and consumption of the distribution grid can be seen in Fig. 5-16e and Fig. 5-16f, and the reactive power generation and demand response throughout the network can be seen in Fig. 5-16g and Fig. 5-16h.

To evaluate the impact of the retail market, we compare Scenario (1) and (2), running the PAC algorithm using both LMP profiles shown in Fig. 5-15. Figure 5-17a shows that generation and electricity imports are lower for Scenario (2), using the market mechanism. Specifically, the DR units are sharing some of the load. This is also illustrated in Fig. 5-17b which shows that DR units accommodate about 5 to

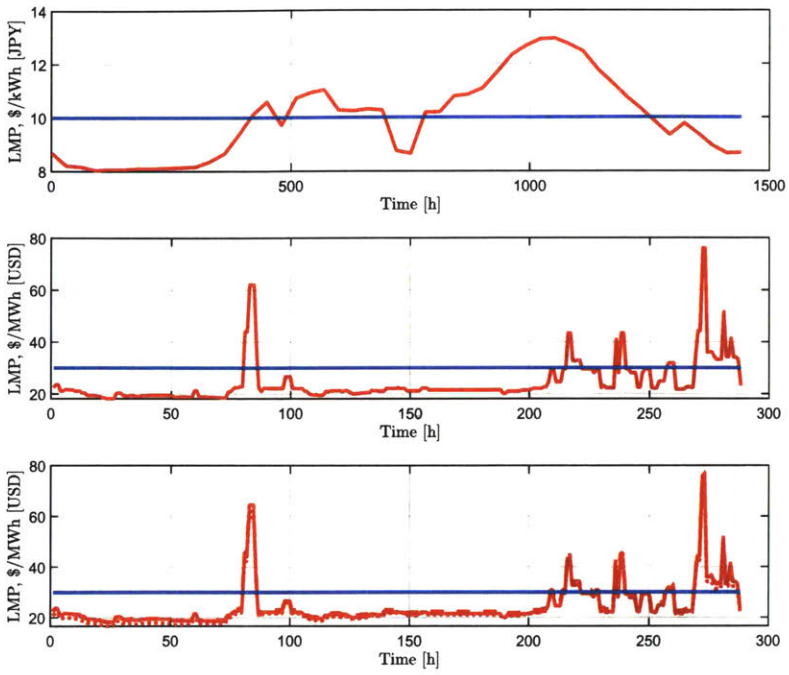
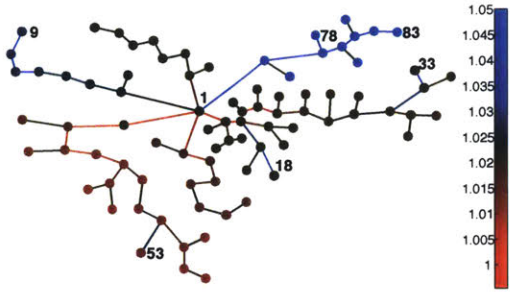


Figure 5-15: Profiles of P-LMP λ_1^{P} where red denotes the yearly average profile and blue a constant LMP, for Networks JST-CREST 126 (top), IEEE-123 (middle), and modified IEEE-13 (bottom).

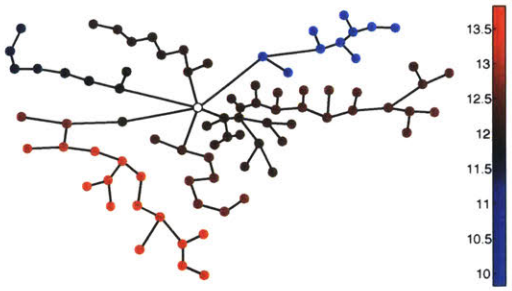
6% of the total load.

The most important results from the proposed market mechanism are the compensation of DER units, and the increase in revenue for the DSO. The DSO's hourly cost for compensating flexible loads and generating units is shown in Fig. 5-17c and 5-17d. Note that Scenario (1) assumes no scheduling or compensation of DERs in the network. Figure 5-17d shows a lower total cost for electricity generation when using the proposed market, as the PAC algorithm optimally schedules the DERs to minimize network costs. Specifically, DERs are used more when electricity prices are higher - the DR units peak usage is around hours 15-18, when the time-varying LMP is the highest.

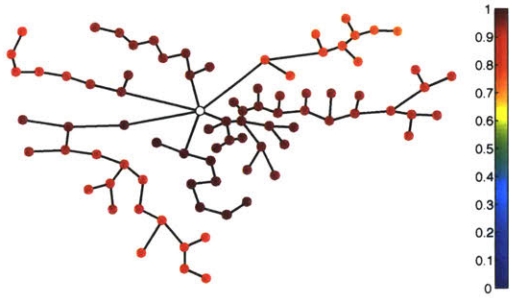
The hourly increase in revenue obtained from implementing the proposed retail market is shown in Fig. 5-18, using the DSO revenue metric introduced in (5.14). The revenue earned by the DSO without the retail market is driven by the difference in



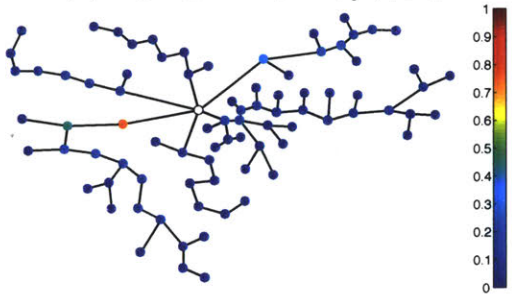
(a) Bus voltage v_j and power flow.



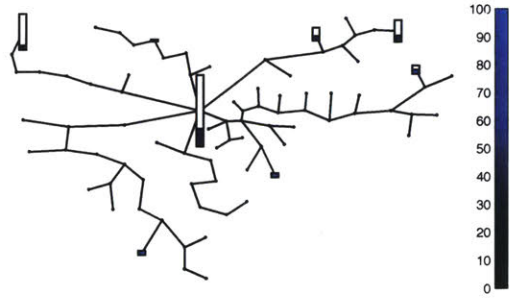
(b) Lagrange multiplier μ_j^V [1/pu]



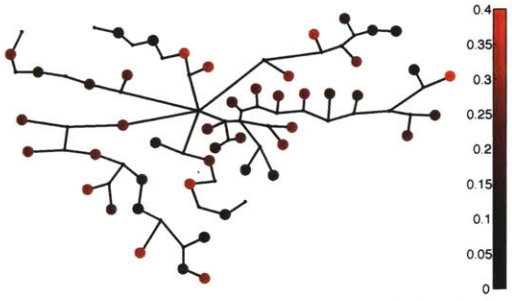
(c) Lagrange multiplier (P-LMP) μ_j^P [1000JPY/pu]



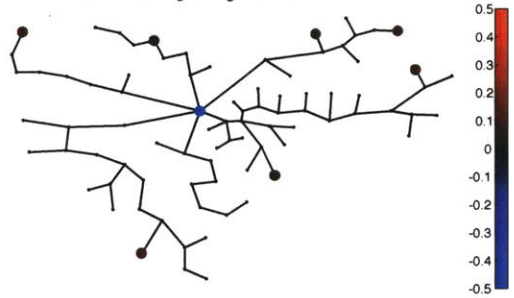
(d) Lagrange multiplier (Q-LMP) μ_j^Q [1/pu]



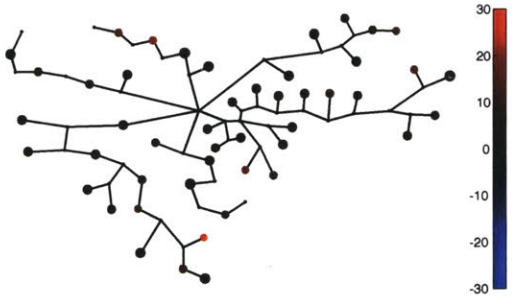
(e) Ratio of active power generation to maximum capacity $P_j^G/\overline{P_j^G}$ [%]



(f) Active power consumption P_j^L [pu]

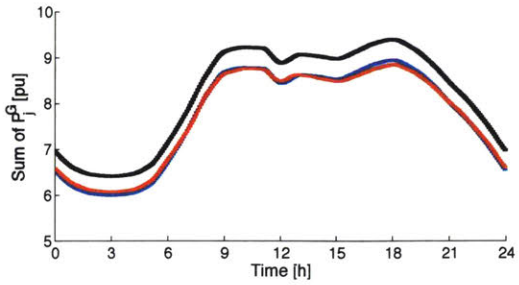


(g) Reactive power generation Q_j^G [pu]

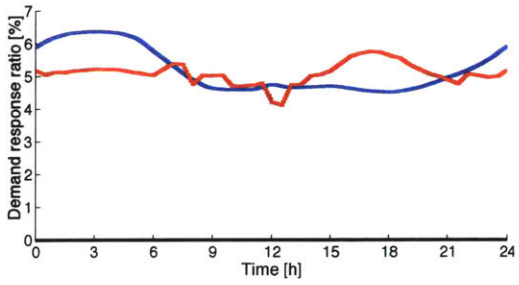


(h) Ratio of demand response to baseline active power consumption $-(P_j^L - P_j^{L0}(19))/P_j^{L0}(19)$ [%]

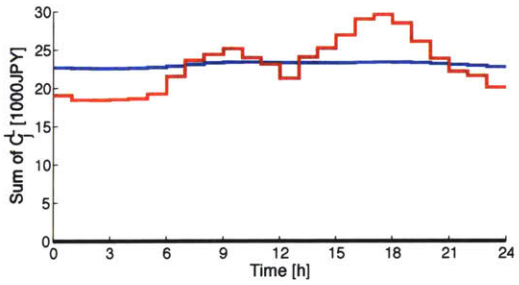
Figure 5-16: Terminal configurations with the PAC algorithm and time-variant P-LMP λ_1^P.



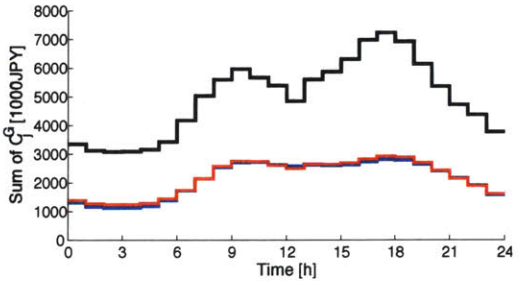
(a) Active power generations ($\sum_j P_j^G$).



(b) Demand response ratio ($\sum_j -(P_j^L - P_j^{L0}(\tau))/\sum_j P_j^{L0}(\tau)$).



(c) Economic compensation of flexible loads. ($\sum_j C_j^L$).



(d) Economic compensation of generators. ($\sum_j C_j^G$).

Figure 5-17: The black graphs correspond to the scenario where all the loads of the distribution grid are served by the main grid, the blue graph to the scenario with the fixed P-LMP and the red graph to the scenario with the time-varying P-LMP λ_1^P.

wholesale and retail price; thus, when the wholesale price of electricity is low, such as during the hours of 1-5, the revenue is high. Compared to the retail cost (26 JY/kWh), the retail market d-LMPs, μ_j^{P*}, are much lower, thus the increase in revenue is small during hours 1-5. However, as LMPs rise and DERs are efficiently scheduled, the DSO generates additional revenue despite the added costs from compensating the DERs for their services. At the end of the day when the LMP is lower but network demand is high, the retail market continues to make additional revenue. This corroborates the fact that the proposed local retail market can result in lower overall generation cost for serving a given load by efficiently dispatching DG and DR units through the distributed PAC algorithm.

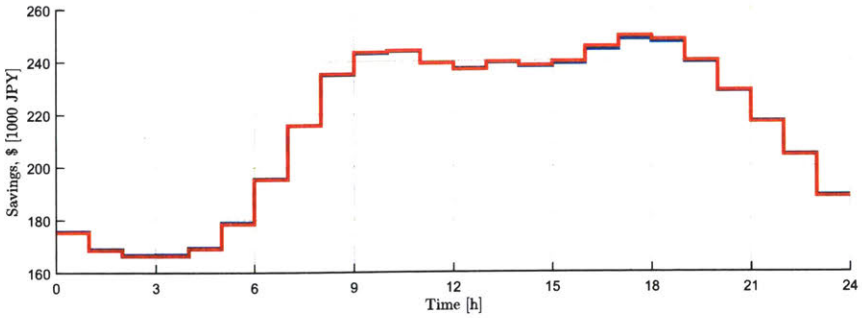


Figure 5-18: The projected savings from the proposed retail market over a 24-hour period

Results from IEEE 123

The proposed market mechanism was also evaluated on the IEEE 123 network (see Appendix B.3.2 for data). As in the JST-CREST 126 network, the retail market results in an increase in revenue for the DSO, as seen in Fig. 5-19. The increase in revenue is similar to Fig. 5-18, as the demand profiles are very similar. The increase in revenue is lowest during the first 5 hours, where the LMP and demand are both low, and increases as demand increases. The usage of DGs also follows the demand curve, while the demand response ratio is relatively constant over the 24 hours at 6.5%. In this way, PAC makes use of the DGs during periods of higher electricity prices and demand, to lower the network cost of generation. Another key observation is that during the high demand hours of 17-21 where the electricity prices are more volatile, the DSO revenue profile is still smooth, from efficient DER management.

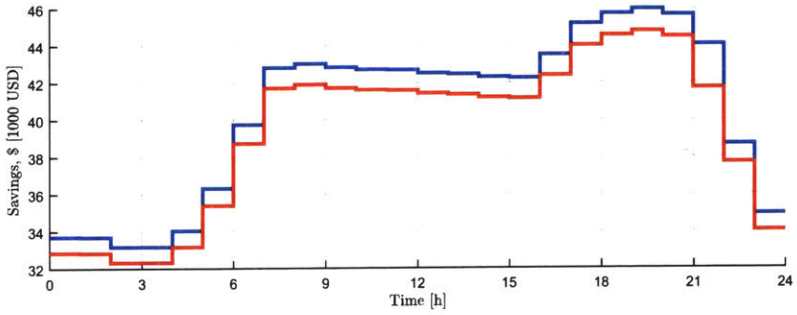


Figure 5-19: The projected savings from the proposed retail market over a 24-hour period

Results from IEEE 13

The proposed ancillary market was evaluated on the 3 feeder modified IEEE 13 network (see Appendix B.2.3 for data). The PAC algorithm was used to calculate the optimal DR schedule, based on the demand and LMP profiles in Figs. 5-14 and 5-15. A contingency was then initiated, whereby feeder 3 loses all generating capabilities ($\Delta P_3 = 2.25$ MW at $T = 230$), some of which was committed. Upon receiving the alert from $\text{DSO}_\text{r} = 3$,the DSO_w carries out Alg_EX (5.7). The weighted average cost calculated by each DSO_r is shown in Table 5.3. The resulting additional generation or load reduction needed is shown in Table 5.4. In this test case, feeder 2 has the lowest cost and picks up the slack from feeder 3. If the generating capacity of feeder 2 was all committed, then the DR units in feeder 2 would be deployed, followed by DR units in feeder 3, and so on. The additional cost to the network, based on the weighted average costs is 0.2813 units.

Item	Feeder 1	Feeder 2	Feeder 3
$\mu_\text{l}^{\text{P,G}'}$	0.8811	0.8027	0.7818
$\mu_\text{l}^{\text{P,L}'}$	0.8612	0.8194	0.8361

Table 5.3: Weighted average cost of using DGs or DRs for each feeder

Item	Feeder 1	Feeder 2	Feeder 3
$P_\text{l}^{\text{G}'}$	0	0.3505	0
$P_\text{l}^{\text{L}'}$	0	0	0

Table 5.4: New dispatch settings from Alg_EX

Chapter 6

Conclusions

In this section we summarize the contributions of the thesis and key results. We then discuss future research direction that can improve upon or extend the work presented here.

6.1 Contributions

In this thesis, we have addressed the challenges of optimally coordinating a large number of distributed energy resources (DERs) within the distribution grid. We consider the curse of dimensionality for large-scale optimization problems, the modelling and scheduling of storage devices, and the required incentives to coordinate the independently owned resources. We focus on utilizing distributed computation to solve the optimal power flow (OPF) problem, as applied to the distribution grid.

First, we apply distributed computation (the proximal atomic coordination algorithm, PAC) to a new convex relaxation of the OPF problem, which is able to model meshed and unbalanced grids. This is done by modeling current injections (CI) and using McCormick Envelopes (MCE).

Second, we extend the CI formulation to account for storage devices, by considering multi-period optimization. We run various simulations to show the necessity for multi-period optimization, which has a forward-looking nature that enables efficient scheduling of resources. We also consider the role of storage sizing through these

simulations.

Third, we propose a retail market mechanism for distribution grids, which provides the financial structure necessary to optimally coordinate independently owned resources. We take a transactive energy approach, and propose an energy market which utilizes the PAC algorithm to recover real-time scheduling and market dispatch through bilateral contracts. We propose a Distribution System Operator (DSO) which oversees the market, and interacts with the WEM. The proposed market structure is numerically verified, showing potential savings for the DSO under this new market mechanism.

6.2 Future Research Directions

There are many future avenues for research and development. Among these are future work for the CI model, PAC algorithm, and proposed retail market. Some of these are briefly discussed below.

6.2.1 Current Injection Model

The CI model has been shown to be effective in solving load flow and optimization problems for meshed and unbalanced grids. A more detailed comparison with BF is still desired: (1) comparing the performance of CI with BF+angle recovery algorithms; (2) comparing the performance of CI on meshed networks, using convexification techniques for the BF model; and (3) if possible, trying to compare the two models for unbalanced networks.

The techniques used to obtain the current bounds must also be investigated. The performance of the CI model is intimately tied with the tightness of the MCE relaxation. Using heuristics, detailed network data, or insights from operations may help tighten these bounds.

6.2.2 Development & Analysis of Distributed Optimization

While the PAC algorithm shows great potential, there remains the issue that many iterations are needed for the algorithm to converge. In addition, every iteration depends on communication with neighbours leading to more risk of communication failures such as packet drops or cyber-events. The performance of the PAC algorithm must be re-evaluated, by conducting a thorough literature review and identifying performance enhancement features. Time-varying learning rates (for γ) should also be investigated, to improve performance of the algorithm as it approaches convergence. Another aspect may be to develop heuristics to obtain the best convergence parameters (γ, ρ).

The distributed architecture of PAC suggests an ability to self-recover from cyber-events, such as cyber-attacks or communication network failures. The robustness of the PAC algorithm under these events must be investigated, by using multiple processing units and secure communication protocols. Islanding events within distribution grids can then be examined: the hypothesis is that the PAC algorithm should be able to achieve global solutions for each islanded section of the grid. What this looks like from a power systems and stability point of view is also an interesting question.

6.2.3 Extension of Retail Market

The current retail market design discusses the interactions of agents within the distribution grid, the DSO, and the WEM. The market structure must be extended to consider the interactions with the Transmission System Operator (TSO). There are various models for TSO-DSO interactions and responsibilities of each party as outlined by their coordination schemes. Most notably, [87] compares five different schemes between the DSO and the TSO to appropriately operate and compensate distribution-level assets. More consideration is needed to understand how the proposed retail market structure and the DSO entity can work with the TSO to bring efficiency to the overall grid.

Assumptions about the exchange and ownership of information made in the pro-

posed design depend heavily on the current paradigm; however, when many distribution grids begin coordinating their own retail markets and having a DSO, there may be changes to the overall structure. More analysis is needed here. For example, the proposed market structure and interactions (A)-(C) outlined in Chapter 5 have assumed that the WEM specifies the LMP λ_{j}^{P} at the initiation of the market interactions, but that the DSO is free to determine the net generation and load seen by the WEM. In practice, the DSO may be obligated to purchase power quantities $P_{\mathrm{l},T_0}^{\mathrm{G}^*}, Q_{\mathrm{l},T_0}^{\mathrm{G}^*}$ (assuming the DSO is importing power) over the period $[T_0, T_0 + \Delta T]$. In such an event, the proposed PAC can still be used, but with an additional boundary condition: between any two WEM market clearings T_0 and $T_0 + \Delta T$, the net power from the main grid, P_{l,T_0}^{*} is constant, $\forall \mathrm{l} \in \mathcal{F}_s, \forall \mathrm{s} \in \mathcal{S}$.

6.2.4 Evaluation of Retail Market Design

When evaluating an energy market, there are additional considerations which must be made. The market design must not allow agents to gain market power and influence over prices and resource allocation, so to benefit themselves unfairly. The retail prices determined by the market mechanism must also be evaluated to limit price volatility, as volatile markets can be hard to predict, and result in high costs for market participants. Historically, electricity prices have been tied with oil and gas prices as these resources were the primary fuel for electricity generation. However, as electricity production shifts to include more renewable resources, the ability for policymakers and market analysts to project and estimate electricity prices to help develop policies, rules, and make predictions on the future of the system, becomes harder.

The market design must be equitable in allocating costs to consumers. In addition to paying incremental cost of service, customers also pay a portion of the residual cost. The method by which this residual cost is allocated to different consumers largely depends on the equity of the electricity tariff design, as opposed to just the economic efficiency - of which there may be multiple solutions. One of the key issues with high DER penetration in the distribution grid is that DER adoption is primarily by high

income customers, resulting in a shift of residual cost from high income to low income customers [88]. Some reports analyzing real-time electricity rates have also concluded that these rate designs may be inequitable to low income customers [89].

The applicability of the proposed market design to different electric power systems across the US and over the world can be investigated. Different electricity prices, generation mixes, weather conditions, and of course, policy frameworks and existing market structures, will all influence the success (or failure) of such a market.

Appendix A

Supplementary Results

A.1 Design of the PAC Algorithm

For complete information about the PAC algorithm, see [20, 21]. Here a summary of the algorithm design is provided for completeness, as taken from the original works on PAC.

We start by considering the constrained optimization problem:

$$\begin{aligned} &\min_{\mathbf{a}\in\mathbb{R}^{|\mathbf{a}|}} \sum_{\mathbf{j}\in|\mathcal{B}|} \tilde{f}_{\mathbf{j}}\left(\mathbf{a}_{\mathbf{j}}\right) \\ &\text{subj. to:} \\ &\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}_{\mathbf{j}} = \mathbf{0}, \quad \mathbf{j}\in\mathcal{B} \\ &\mathbf{A}_{\mathbf{j},-}\mathbf{a} = \mathbf{0}, \quad \mathbf{j}\in\mathcal{B} \end{aligned} \tag{A.1}$$

We can then form the Lagrangian of (A.1) as given by:

$$\begin{aligned} \mathcal{L}\left(\mathbf{a},\boldsymbol{\mu},\boldsymbol{\nu}\right) &= \sum_{\mathbf{j}\in\mathcal{B}} \left[\hat{F}_{\mathbf{j}}\left(\mathbf{a}_{\mathbf{j}}\right) + \boldsymbol{\mu}^T\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}_{\mathbf{j}} + \boldsymbol{\nu}^T\mathbf{A}_{-,\mathbf{j}}\mathbf{a}_{\mathbf{j}}\right] \\ &\triangleq \sum_{\mathbf{j}\in\mathcal{B}} \mathcal{L}_{\mathbf{j}}\left(\mathbf{a}_{\mathbf{j}},\boldsymbol{\mu}_{\mathbf{j}},\boldsymbol{\nu}\right). \end{aligned}$$

We note that the two variables that are of most importance are $\mathbf{a}$, the decision variable, which corresponds to the optimal power generation and consumption of

active and reactive power, and μ, the price associated with the decision variable. By introducing augmentation and proximal regularization terms to $\mathcal{L}(\mathbf{a}, \boldsymbol{\mu}, \boldsymbol{\nu})$ we obtain the augmented Lagrangian:

$$\begin{aligned}\tilde{\mathcal{L}}_{\rho,\gamma}(\mathbf{a}, \boldsymbol{\mu}, \boldsymbol{\nu}; \mathbf{a}') = &\sum_{\mathbf{j}\in\mathcal{B}} \left[\mathcal{L}_{\mathbf{j}}(\mathbf{a}_{\mathbf{j}}, \boldsymbol{\mu}_{\mathbf{j}}, \boldsymbol{\nu}) + \frac{\rho\gamma_{\mathbf{j}}}{2} \left\|\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}_{\mathbf{j}}\right\|_2^2\right] \\ &+ \sum_{\mathbf{j}\in\mathcal{B}} \left[\frac{\rho\gamma_{\mathbf{j}}}{2} \|\mathbf{A}_{\mathbf{j},-}\mathbf{a}\|_2^2 + \frac{1}{2\rho} \left\|\mathbf{a}_{\mathbf{j}} - \mathbf{a}'_{\mathbf{j}}\right\|_2^2\right] \end{aligned} \tag{A.2}$$

Given that, $\tilde{\mathcal{L}}_{\mathbf{j}}(\mathbf{a}_{\mathbf{j}}, \boldsymbol{\mu}_{\mathbf{j}}, \boldsymbol{\nu}; \mathbf{a}'_{\mathbf{j}}) = \mathcal{L}_{\mathbf{j}}(\mathbf{a}_{\mathbf{j}}, \boldsymbol{\mu}_{\mathbf{j}}, \boldsymbol{\nu}) + \frac{1}{2\rho} \left\|\mathbf{a}_{\mathbf{j}} - \mathbf{a}'_{\mathbf{j}}\right\|_2^2$ we can express it more compactly as:

$$\begin{aligned}\tilde{\mathcal{L}}_{\rho,\gamma}(\mathbf{a}, \boldsymbol{\mu}, \boldsymbol{\nu}; \mathbf{a}') \triangleq &\sum_{\mathbf{j}\in\mathcal{B}} \left[\tilde{\mathcal{L}}_{\mathbf{j}}(\mathbf{a}_{\mathbf{j}}, \boldsymbol{\mu}_{\mathbf{j}}, \boldsymbol{\nu}; \mathbf{a}'_{\mathbf{j}})\right] \\ &+ \sum_{\mathbf{j}\in\mathcal{B}} \left[\frac{\rho\gamma_{\mathbf{j}}}{2} \left\|\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}_{\mathbf{j}}\right\|_2^2 + \frac{\rho\gamma_{\mathbf{j}}}{2} \|\mathbf{A}_{\mathbf{j},-}\mathbf{a}\|_2^2\right] \end{aligned} \tag{A.3}$$

where $\rho > 0$ is the *step-size* and $\gamma > 0$ is the over-relaxation term. The augmentation terms can be linearized around a prior primal iteration $(\mathbf{a}'_{\mathbf{j}}$ and $\mathbf{a}')$ as:

$$\frac{\rho\gamma_{\mathbf{j}}}{2} \left\|\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}_{\mathbf{j}}\right\|_2^2 \approx \frac{\rho\gamma_{\mathbf{j}}}{2} \left\|\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}'_{\mathbf{j}}\right\|_2^2 + \rho\gamma_{\mathbf{j}} \left(\mathbf{a}_{\mathbf{j}} - \mathbf{a}'_{\mathbf{j}}\right)^T \tilde{\mathbf{G}}_{\mathbf{j}}^T \tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}'_{\mathbf{j}} \tag{A.4}$$

$$\frac{\rho\gamma_{\mathbf{j}}}{2} \|\mathbf{A}_{\mathbf{j},-}\mathbf{a}\|_2^2 \approx \frac{\rho\gamma_{\mathbf{j}}}{2} \|\mathbf{A}_{\mathbf{j},-}\mathbf{a}'\|_2^2 + \rho\gamma_{\mathbf{j}} \left(\mathbf{a}_{\mathbf{j}} - \mathbf{a}'_{\mathbf{j}}\right)^T \mathbf{A}_{\mathbf{j},-}^T \mathbf{A}_{\mathbf{j},-}\mathbf{a}'_{\mathbf{j}} \tag{A.5}$$

Application of (A.4) and (A.5) to (A.3) yields:

$$\begin{aligned}\tilde{\mathcal{L}}_{\rho,\gamma}(\mathbf{a}, \boldsymbol{\mu}, \boldsymbol{\nu}; \mathbf{a}') = &\sum_{\mathbf{j}\in\mathcal{B}} \left[\tilde{\mathcal{L}}_{\mathbf{j}}(\mathbf{a}_{\mathbf{j}}, \boldsymbol{\mu}_{\mathbf{j}}, \boldsymbol{\nu}; \mathbf{a}'_{\mathbf{j}}) + \left\langle\rho\gamma_{\mathbf{j}}\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}'_{\mathbf{j}}, \tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}_{\mathbf{j}}\right\rangle\right] \\ &+ \sum_{\mathbf{j}\in\mathcal{B}} \left[\left\langle\rho\gamma_{\mathbf{j}}\mathbf{A}_{\mathbf{j},-}\mathbf{a}'_{\mathbf{j}}, \mathbf{A}_{\mathbf{j},-}\mathbf{a}_{\mathbf{j}}\right\rangle\right]. \end{aligned} \tag{A.6}$$

To maintain complete privacy of the algorithm, we use two atomic-specific over-relaxation terms: $\gamma_{\mathbf{j}}, \hat{\gamma}_{\mathbf{j}} > 0$ where $\gamma_{\mathbf{j}} > \hat{\gamma}_{\mathbf{j}}$. Thus, we can express the new 'predicted'

dual updates in (A.6) as:

$$\tilde{\boldsymbol{\mu}}_{\mathbf{j}} = \boldsymbol{\mu}_{\mathbf{j}} + \rho\hat{\gamma}_{\mathbf{j}}\tilde{\mathbf{G}}_{\mathbf{j}}\mathbf{a}'_{\mathbf{j}}, \quad \forall \mathbf{j} \in \mathcal{B}$$
$$\tilde{\boldsymbol{\nu}}_{\mathbf{j}} = \boldsymbol{\nu}_{\mathbf{j}} + \rho\hat{\gamma}_{\mathbf{j}}\mathbf{A}_{\mathbf{j},-}\mathbf{a}'_{\mathbf{j}}, \quad \forall \mathbf{j} \in \mathcal{B}$$

Finally, we can write (A.6) as:

$$\tilde{\mathcal{L}}_{\rho,\gamma}\left(\mathbf{a}, \boldsymbol{\mu}, \boldsymbol{\nu}; \mathbf{a}'\right) = \sum_{\mathbf{j}\in\mathcal{B}} \tilde{\mathcal{L}}_{\mathbf{j}}\left(\mathbf{a}_{\mathbf{j}}, \tilde{\boldsymbol{\mu}}_{\mathbf{j}}, \tilde{\boldsymbol{\nu}}; \mathbf{a}'_{\mathbf{j}}\right). \tag{A.7}$$

A.2 Convergence of the PAC Algorithm

For complete convergence proofs and results, see [20, 21]. Key results are provided here to support the use of the PAC algorithm.

Before stating the main result regarding the convergence of the PAC algorithm, it is convenient to define the following auxiliary matrices:

$$\begin{aligned}
\tilde{\mathbf{G}} &= \mathbf{diag}\left\{\tilde{\mathbf{G}}_{\mathbf{1}}, \ldots, \tilde{\mathbf{G}}_{\mathbf{K}}\right\} \\
\mathbf{\Gamma}^{\tilde{\mathbf{G}}} &= \mathbf{diag}\left\{\gamma_{\mathbf{1}}\mathbf{I}_{\mathbf{N}_{\mathbf{1}}^{\mathbf{C}}}, \ldots, \gamma_{|\mathcal{B}|}\mathbf{I}_{\mathbf{N}_{|\mathcal{B}|}^{\mathbf{C}}}\right\} \\
\mathbf{\Gamma}^{\mathbf{A}} &= \mathbf{diag}\left\{\gamma_{\mathbf{1}}\mathbf{I}_{\mathbf{N}_{\mathbf{1}}^{\mathbf{O}}} \ldots, \gamma_{|\mathcal{B}|}\mathbf{I}_{\mathbf{N}_{|\mathcal{B}|}^{\mathbf{O}}}\right\} \\
\mathbf{V_1} &= \tilde{\mathbf{G}}^T\tilde{\mathbf{G}} + \mathbf{A}^T\mathbf{A} \\
\tilde{\mathbf{V}}_{\mathbf{1}}(\mathbf{\Gamma}) &= \tilde{\mathbf{G}}^T\mathbf{\Gamma}^{\tilde{\mathbf{G}}}\tilde{\mathbf{G}} + \mathbf{A}^T\mathbf{\Gamma}^{\mathbf{A}}\mathbf{A}
\end{aligned}$$

Asymptotic convergence of the distributed PAC algorithm given by (3.50)-(3.55) is established through the following theorem. [20, 21] Consider the ROPF formulation of (5.1) and its atomized variant (A.1).

Let $\hat{F}_{\mathbf{j}}$ be a closed, convex and proper (CCP) function for all $\mathbf{j} \in \mathcal{B}$ and $\mathbf{a}[\tau], \boldsymbol{\mu}[\tau], \tilde{\boldsymbol{\mu}}[\tau], \boldsymbol{\nu}[\tau]$, and $\tilde{\boldsymbol{\nu}}[\tau]$ be the primal and dual variable trajectories of (3.50)-(3.55) under zero initial conditions. Further, let $\mathbf{a}^*$ be the optimal primal solution to (A.1) and $\rho, \gamma_{\mathbf{j}}, \hat{\gamma}_{\mathbf{j}}$ the PAC parameters.

We define $\lambda_{\mathbf{max}} \triangleq \max_{\mathbf{j}\in\mathcal{B}}\{\lambda_{\mathbf{max}}(\mathbf{V_1})\}$, $\lambda_{\mathbf{min}} \triangleq \min_{\mathbf{j}\in\mathcal{B}}\{\lambda_{\mathbf{min}}(\mathbf{V_1})\}$, $\gamma_{\mathbf{max}} \triangleq \max_{\mathbf{j}\in\mathcal{B}}\{\gamma_{\mathbf{j}}\}$, and $\gamma_{\mathbf{min}} \triangleq \in_{\mathbf{j}\in\mathcal{B}}\{\gamma_{\mathbf{j}}\}$.

Then, we have that:

a) If:

$$1 > \rho^2\gamma_{\mathbf{max}}\lambda_{\mathbf{max}}(\mathbf{V_1}) \tag{A.8}$$

then $\mathbf{lim}_{\tau\to\infty}\{\mathbf{a}[\tau]\} \to \mathbf{a}^*$ with rate $o\left(\frac{1}{\tau}\right)$

b) If:

$$1 \geq \rho^2\gamma_{\mathbf{max}}\lambda_{\mathbf{max}}(\mathbf{V_1}) \tag{A.9}$$

then for the ergodic averaged trajectory $\bar{\mathbf{a}}[\tau] \triangleq \frac{1}{\tau}\sum_{s=0}^{\tau}\mathbf{a}[s]$, we have:

$$\left|\hat{F}(\bar{\mathbf{a}}[\tau]) - \hat{F}(\mathbf{a}^*)\right| \leq \left(\frac{1}{\tau}\right)\xi_{\mathbf{E1}}^{-1}(\rho, \gamma_{\mathbf{max}}) \tag{A.10}$$

$$\left\|\mathbf{R}^{\mathbf{\Gamma}}\bar{\mathbf{a}}[\tau]\right\|_2^2 \leq \left(\frac{1}{\tau}\right)\xi_{\mathbf{E2}}^{-1}(\rho, \gamma_{\mathbf{max}}) \tag{A.11}$$

where:

$$\mathbf{R}^{\mathbf{\Gamma}}\mathbf{R}^{\mathbf{\Gamma}} \triangleq \tilde{\mathbf{V}}_{\mathbf{1}}(\mathbf{\Gamma})^{\,1} \tag{A.12}$$

and:

$$\xi_{\mathbf{E1}}(\rho, \gamma_{\mathbf{max}}) = \frac{\rho}{2}\left[\|\mathbf{a}^*\|_2^2 + \frac{4\Phi^2}{\gamma_{\mathbf{min}}\hat{\lambda}_{\mathbf{min}}(\mathbf{V_1})}\right]^{-1} \tag{A.13}$$

$$\xi_{\mathbf{E2}}(\rho, \gamma_{\mathbf{max}}) = \frac{\rho^2}{2}\left[\|\mathbf{a}^*\|_2^2 + 4\rho^2 + \frac{4\Phi^2}{\gamma_{\mathbf{min}}\hat{\lambda}_{\mathbf{min}}(\mathbf{V_1})}\right]^{-1} \tag{A.14}$$

where $\Phi = \mathbf{sup}_{\mathbf{z}\in\partial\hat{F}(\mathbf{a}^*)}\{\|\mathbf{z}\|_2\}$.

Intuitively, Theorem A.2 guarantees that the nodal generation and consumption profiles obtained upon convergence of the PAC algorithm coincide with the optimal solution of the centralized OPF problem.

[1]The *root* $\mathbf{R}$ exists since $\mathbf{V_1}$ is square p.s.d. matrix. Specifically, if $\mathbf{V_1} = \mathbf{U\Sigma U}^T$ is a suitable eigenvalue decomposition, then $\mathbf{R} = \left[\mathbf{U\Sigma}^{\frac{1}{2}}\mathbf{U}^T\right]$

Appendix B

Case Study Data

This section includes the simulation data so results can be reproduced.

B.1 Network 3 Node

Parent Node	Child Node	Line length (mi)	Impedance Configuration Code
1	2	0.0568	1
2	3	0.0568	1

Table B.1: Network Line data

Impedances for each configuration are labelled as Z_x for configuration x.

$$Z_1 = \begin{bmatrix} 0.3465 + 1.0179j & 0.1560 + 0.5017j & 0.1580 + 0.4236j \\ & 0.3375 + 1.0478j & 0.1535 + 0.3849j \\ & & 0.3414 + 1.0348j \end{bmatrix}$$

$$Z_2 = \begin{bmatrix} 0.7526 + 1.1814j & 0.1580 + 0.4236j & 0.1560 + 0.5017j \\ & 0.7475 + 1.1983j & 0.1535 + 0.3849j \\ & & 0.7436 + 1.2112j \end{bmatrix}$$

$\underline{V} = 0.95; \quad \overline{V} = 1.05$

$\underline{\delta} = -0.01; \quad \overline{\delta} = 0.01$

B.1.1 Load and DER Configuration 1

Node	P, phase A	P, phase B	P, phase C	Q, phase A	Q, phase B	Q, phase C
2	200	300	500	100	150	300
3	200	500	100	80	350	20

Table B.2: Network nodal load data, in kW and kVAr

B.2 Network IEEE 13 Node

B.2.1 Default Configuration

This configuration of the IEEE 13-bus network is used in Chapter 3 for preliminary results of OPF solved in distributed way using the CI model.

Parent Node	Child Node	Line length (ft)	Impedance Configuration Code
1	2	2000	1
2	3	500	3
2	5	500	2
5	6	300	1
3	4	300	3
2	7	2000	1
7	9	300	4
9	10	300	5
9	11	800	7
7	8	500	6
7	12	1000	1

Table B.3: Network Line data

Impedances for each configuration are labelled as Z_x for configuration x.

$$Z_1 = \begin{bmatrix} 0.3465 + 1.0179j & 0.1560 + 0.5017j & 0.1580 + 0.4236j \\ & 0.3375 + 1.0478j & 0.1535 + 0.3849j \\ & & 0.3414 + 1.0348j \end{bmatrix}$$

$$Z_2 = \begin{bmatrix} 0.7526 + 1.1814j & 0.1580 + 0.4236j & 0.1560 + 0.5017j \\ & 0.7475 + 1.1983j & 0.1535 + 0.3849j \\ & & 0.7436 + 1.2112j \end{bmatrix}$$

$$Z_3 = \begin{bmatrix} 0 & 0 & 0 \\ & 1.3294 + 1.3471j & 0.2066 + 0.4591j \\ & & 1.3238 + 1.3569j \end{bmatrix}$$

$$Z_4 = \begin{bmatrix} 1.3238 + 1.3569j & 0 & 0.2066 + 0.4591j \\ & 0 & 0 \\ & & 1.3294 + 1.3471j \end{bmatrix}$$

$$Z_5 = \begin{bmatrix} 0 & 0 & 0 \\ & 0 & 0 \\ & & 1.3292 + 1.3475j \end{bmatrix}$$

$$Z_6 = \begin{bmatrix} 0.7982 + 0.4463j & 0.3192 + 0.0328j & 0.2849 - 0.0143j \\ & 0.7891 + 0.4041j & 0.3192 + 0.0328j \\ & & 0.7982 + 0.4463j \end{bmatrix}$$

$$Z_7 = \begin{bmatrix} 1.3425 + 0.5124j & 0 & 0 \\ & 0 & 0 \\ & & 0 \end{bmatrix}$$

$\underline{V} = 0.9; \quad \overline{V} = 1.1$

$\underline{\delta} = -0.08; \quad \overline{\delta} = 0.08$

Load and DER Configuration 1

This is the passive configuration.

Node	P, phase A	P, phase B	P, phase C	Q, phase A	Q, phase B	Q, phase C
3	0	170	0	0	125	0
4	0	230	0	0	132	0
6	160	120	120	110	90	90
7	385	385	385	220	220	220
8	485	68	290	190	60	212
10	0	0	170	0	0	80
11	128	0	0	86	0	0

Table B.4: Network nodal load data, in kW and kVAr

Load and DER Configuration 2

This is the active configuration created for Chapter 3. Load data is unchanged from configuration 1.

Distributed generation, real power						
Node	$\overline{P}$, Phase A	$\overline{P}$, Phase B	$\overline{P}$, Phase C	$\underline{P}$, Phase A	$\underline{P}$, Phase B	$\underline{P}$, Phase C
9	100	100	100	0	0	0
Distributed generation, reactive power						
Node	$\overline{Q}$, Phase A	$\overline{Q}$, Phase B	$\overline{Q}$, Phase C	$\underline{Q}$, Phase A	$\underline{Q}$, Phase B	$\underline{Q}$, Phase C
8	200	200	200	-200	-200	-200
9	50	50	50	-50	-50	-50
10	0	0	100	0	0	-100
Demand Response						
Node	% enabled					
3	20					
6	10					
11	30					

Table B.5: Distributed energy resources (DERs) for active configuration. Generation capabilities in kW and kVA.

Load and DER Configuration 3

This is the active configuration used for simulations in Chapter 4, where the penetration of renewable resources is varied. Note that the loads at node 3 and 8 have been removed, and renewable generation has been introduced. Two different test cases are presented for this configuration: a lower penetration of renewables (test case 1), and a higher penetration of renewables (test case 2). The type of resource is indicated, where 'PV' is photovoltaic, 'Comm' is commercial, 'Res S' is small residential, and 'Res L' is large residential. Refer to Appendix B.7 for information and generation profiles of each resource type. All power and capacity data in kW and kWh.

Distributed generation, real power						
Node	$\overline{P}$, Phase A	$\overline{P}$, Phase B	$\overline{P}$, Phase C	$\underline{P}$, Phase A	$\underline{P}$, Phase B	$\underline{P}$, Phase C
9	150	150	150	0	0	0
Distributed generation, reactive power						
Node	$\overline{Q}$, Phase A	$\overline{Q}$, Phase B	$\overline{Q}$, Phase C	$\underline{Q}$, Phase A	$\underline{Q}$, Phase B	$\underline{Q}$, Phase C
9	75	75	75	-75	-75	-75
Renewable Generation						
Node	Capacity	Phases	Type	Test Case		
2	130	a-b-c	PV, Comm	{2}		
3	20	a-b-c	PV, Res L	{1,2}		
5	100	a-b-c	PV, Comm	{1,2}		
8	15	a-b-c	PV, Res L	{1,2}		
Demand Response						
Node	% enabled					
4	20					
6	10					
11	30					

Table B.6: Distributed energy resources (DERs) for active configuration. Generation capabilities in kW and kVA.

Load and DER Configuration 4

This is the active configuration used for simulations in Chapter 4, where the total installed storage capacity is varied. Note this uses the data from Configuration 3 (above), for test case 1. Three different test cases are presented, over which the capacity of storage present at node 12 is varied. The type of renewable energy resource

is indicated, where 'PV' is photovoltaic, 'Comm' is commercial, 'Res S' is small residential, and 'Res L' is large residential. Refer to Appendix B.7 for information and generation profiles of each resource type. All power and capacity data in kW and kWh.

Distributed generation, real power						
Node	$\overline{P}$, Phase A	$\overline{P}$, Phase B	$\overline{P}$, Phase C	$\underline{P}$, Phase A	$\underline{P}$, Phase B	$\underline{P}$, Phase C
9	150	150	150	0	0	0
Distributed generation, reactive power						
Node	$\overline{Q}$, Phase A	$\overline{Q}$, Phase B	$\overline{Q}$, Phase C	$\underline{Q}$, Phase A	$\underline{Q}$, Phase B	$\underline{Q}$, Phase C
9	75	75	75	-75	-75	-75
Renewable Generation						
Node	Capacity	Phases	Type			
3	20	a-b-c	PV, Res L			
5	100	a-b-c	PV, Comm			
8	15	a-b-c	PV, Res L			
Storage						
Node	Capacity	Power	Ramp	η^{C}/η^{D}	η^{self}	Initial SOC
12	1000	700	600	1/1	0	500
	500	300	300	0.95/0.95	0.021	100
	2000	700	600	0.95/0.95	0.021	300
Demand Response						
Node	% enabled					
4	20					
6	10					
11	30					

Table B.7: Distributed energy resources (DERs) for active configuration. Generation capabilities in kW and kVA, and capacities in kWh.

Load and DER Configuration 5

This is the active configuration used for simulations in Chapter 4, where only DR and storage are added to the grid. The loads are the same as the passive configuration. All power and capacity data in kW and kWh.

Distributed generation, real power						
Node	$\overline{P}$, Phase A	$\overline{P}$, Phase B	$\overline{P}$, Phase C	$\underline{P}$, Phase A	$\underline{P}$, Phase B	$\underline{P}$, Phase C
9	150	150	150	0	0	0
Distributed generation, reactive power						
Node	$\overline{Q}$, Phase A	$\overline{Q}$, Phase B	$\overline{Q}$, Phase C	$\underline{Q}$, Phase A	$\underline{Q}$, Phase B	$\underline{Q}$, Phase C
9	75	75	75	-75	-75	-75
Storage						
Node	Capacity	Power	Ramp	η^{C}/η^{D}	η^{self}	Initial SOC
12	1000	700	600	0.95/0.95	0.021	150
Demand Response						
Node	% enabled					
3	40					
4	20					
6	50					
8	10					
11	30					

Table B.8: Distributed energy resources (DERs) for active configuration. Generation capabilities in kW and kVA, and capacities in kWh.

Load and DER Configuration 6

This is the active configuration used for simulations in Chapter 4, where only DR is added to the grid. The loads are the same as the passive configuration. All power and capacity data in kW and kWh.

Distributed generation, real power						
Node	$\overline{P}$, Phase A	$\overline{P}$, Phase B	$\overline{P}$, Phase C	$\underline{P}$, Phase A	$\underline{P}$, Phase B	$\underline{P}$, Phase C
9	150	150	150	0	0	0
Distributed generation, reactive power						
Node	$\overline{Q}$, Phase A	$\overline{Q}$, Phase B	$\overline{Q}$, Phase C	$\underline{Q}$, Phase A	$\underline{Q}$, Phase B	$\underline{Q}$, Phase C
9	75	75	75	-75	-75	-75
Demand Response						
Node	% enabled					
3	40					
4	20					
6	50					
8	10					
11	30					

Table B.9: Distributed energy resources (DERs) for active configuration. Generation capabilities in kW and kVA, and capacities in kWh.

B.2.2 Balanced Configuration

The balanced configuration of the 13-bus network is used to compare current injection and branch flow models. All cables were assumed to be 3-phase transposed, with configurations 1 thru 7 converted to symmetric matrices, and using the positive sequence component. The impedance configurations are provided below, and are labelled as Z_x for configuration x.

$$Z_1 = \begin{bmatrix} 0.1860 + 0.5968i & 0 & 0 \\ & 0.1860 + 0.5968i & 0 \\ & & 0.1860 + 0.5968i \end{bmatrix}$$

$$Z_2 = \begin{bmatrix} 0.5921 + 0.7602i & 0 & 0 \\ & 0.5921 + 0.7602i & 0 \\ & & 0.5921 + 0.7602i \end{bmatrix}$$

$$Z_3 = \begin{bmatrix} 0.8155 + 0.7483i & 0 & 0 \\ & 0.8155 + 0.7483i & 0 \\ & & 0.8155 + 0.7483i \end{bmatrix}$$

$$Z_4 = \begin{bmatrix} 0.8155 + 0.7483i & 0 & 0 \\ & 0.8155 + 0.7483i & 0 \\ & & 0.8155 + 0.7483i \end{bmatrix}$$

$$Z_5 = \begin{bmatrix} 0.4431 + 0.4492i & 0 & 0 \\ & 0.4431 + 0.4492i & 0 \\ & & 0.4431 + 0.4492i \end{bmatrix}$$

$$Z_6 = \begin{bmatrix} 0.4874 + 0.4151i & 0 & 0 \\ & 0.4874 + 0.4151i & 0 \\ & & 0.4874 + 0.4151i \end{bmatrix}$$

$$Z_7 = \begin{bmatrix} 0.4475 + 0.1708i & 0 & 0 \\ & 0.4475 + 0.1708i & 0 \\ & & 0.4475 + 0.1708i \end{bmatrix}$$

$\underline{V} = 0.9; \quad \overline{V} = 1.1$

Load and DER Configuration 1

This is the passive configuration. The IEEE 13 bus network was modified to be a balanced 3-phase distribution network as below:

- switches were assumed in their normal positions
- single-phase spot loads were assumed to be 3-phase

Node	P	Q
3	170	125
4	230	132
6	133	97
7	385	220
8	281	154
10	170	80
11	128	86

Table B.10: Network nodal load data, in kW and kVAr

B.2.3 Configuration for Ancillary Market Simulations

The IEEE 13 bus network was modified to be 3-phase balanced as below:

- switches were assumed in their normal positions
- single-phase spot loads were assumed to be 3-phase
- cables were assumed to be 3-phase transposed, with configurations 1 thru 7 converted to symmetric matrices

- shunt capacitors were modeled as reactive power generators, and were assumed to be 3-phase

The configuration of nodes and lines varies slightly, as all nodes from the original network are retained and a new line impedance configuration is created to connect to switches and transformers. The data is provided below.

Parent Node	Child Node	Line length (ft)	Impedance Configuration Code
1	2	2000	1
2	3	500	2
2	5	500	3
5	6	300	3
3	4	1	8
2	7	2000	1
7	8	1	8
7	10	300	4
7	13	1000	1
8	9	500	6
10	11	800	7
10	12	300	5

Table B.11: Network Line data

The impedances are the same as in Appendix , with the additional configuration being a matrix of zeros.

The loads are the same as in Appendix B.2.3, except the labelling of the nodes are different. As such, the data is provided again:

Node	P	Q
4	133	97
5	170	125
6	230	132
7	385	220
8	170	151
9	281	154
11	128	86
12	170	80

Table B.12: Network nodal load data, in kW and kVAr

The three feeder variations were developed by adding generators of varying ca-

pacity at different locations as in Table B.13.

Table B.13: Generation node data for IEEE 13

Feeder 1	$j \in \mathcal{B}$	1	4	10	13
	$\overline{P_j^G}$ [MW]	40	0.8	0.5	0.3
Feeder 2	$j \in \mathcal{B}$	1	5	8	12
	$\overline{P_j^G}$ [MW]	40	1	1.5	1.5
Feeder 3	$j \in \mathcal{B}$	1	10	8	
	$\overline{P_j^G}$ [MW]	40	2	0.25	

B.3 Network IEEE 123 Node

B.3.1 Default Configuration

This configuration of the IEEE 123-bus network is used in Chapter 3 for preliminary results of OPF solved in distributed way using the CI model, and Chapter4 for multi-period optimization. Due to the large data size, the information on electrical line/loads and generators can be found in IEEE documentation [86]. Key data is provided below.

Some buses (switches) were removed, and some were relabelled. This data is summarized here:

Original Label	New Label
149	1
152	116
250	117
300	118
450	119

Table B.14: Network nodal load data, in kW and kVAr

Buses 120, 135, 160, and 197 were removed (switches) and the new line configurations are:

Load and DER Configuration 1

This is the passive configuration. See IEEE documentation [86] for loads.

Load and DER Configuration 2

This is the first active configuration created for Chapter 3. Load data is unchanged from configuration 1. Total PV penetration is 179kW.

Distributed generation, reactive power						
Node	$\overline{Q}$, Phase A	$\overline{Q}$, Phase B	$\overline{Q}$, Phase C	$\underline{Q}$, Phase A	$\underline{Q}$, Phase B	$\underline{Q}$, Phase C
82	200	200	200	-200	-200	-200
90	50	50	50	-50	-50	-50
92	50	50	50	-50	-50	-50
94	50	50	50	-50	-50	-50
Renewable Generation						
Node	Capacity	Phases	Type			
4	9	c	PV, Res S			
9	3	a-b-c	PV, Res S			
14	12	a-b-c	PV, Res L			
15	6	a	PV, Res S			
16	5	c	PV, Res S			
19	7	a-b-c	PV, Res S			
22	10	a-b-c	PV, Res S			
24	3	a-b-c	PV, Res S			
26	7	a-b-c	PV, Res S			
28	4	a-c	PV, Res S			
37	3	a-b-c	PV, Res S			
41	5	c	PV, Res S			
45	8	a-b-c	PV, Res S			
55	12	a-b-c	PV, Res L			
58	3	a-b-c	PV, Res S			
62	6	a-b-c	PV, Res S			
73	9	a-b-c	PV, Res S			
79	6	a-b-c	PV, Res S			
98	10	a-b-c	PV, Res S			
102	10	a-b-c	PV, Res S			
106	14	a-b-c	PV, Res L			
109	4	a-b-c	PV, Res S			
111	2	a	PV, Res S			
116	2	a-b-c	PV, Res S			
117	9	a-b-c	PV, Res S			
118	2	a-b-c	PV, Res S			
119	8	a-b-c	PV, Res S			

Table B.15: Distributed energy resources (DERs) for active configuration. Generation capabilities in kW and kVA.

B.3.2 Configuration for Energy Market Simulations

The IEEE 123 bus network was modified to be 3-phase balanced as below:

- switches were assumed in their normal positions
- single-phase spot loads were assumed to be 3-phase
- cables were assumed to be 3-phase transposed, with configurations 1 thru 12 converted to symmetric matrices
- shunt capacitors were modeled as reactive power generators, and were assumed to be 3-phase

Electrical line/loads and generators data is in IEEE documentation [86]. Base values and parameters are in Table B.17. Generators are in Table B.16.

Table B.16: Generation node data for IEEE 123

$j \in \mathcal{B}$	1	25	40	67	81	94
$\overline{P_j^G}$ [MW]	20	0.5	0.7	0.8	1.2	0.75

Table B.17: Base values and simulation parameters for IEEE 123

Parameter	IEEE 123
$\lvert\mathcal{B}\rvert$	123
$\lvert\mathcal{T}\rvert$	122
f_{base}, Hz	60
v_{base}, kV	4.16
S_{base}, kVA	10^3
$\underline{v_j}$, pu	0.8
$\overline{v_j}$, pu	1.2
$\overline{S^2_{i,j}}$, pu	10
$\overline{Q^G_j}, -\underline{Q^G_j}$	See [86]
$\underline{P^G_j}$	0
$\overline{P^L_j}$	0
$\overline{\overline{Q^L_j}}$	See [86]
$\underline{Q^L_j}$	0
β^{PL}_j	$\in [400, 800]$
β^{PG}_j	$\in [4, 8]$
β^{QG}_j	$\in [4, 8]$
ξ	100
λ^Q_1	1

B.4 Network JST-CREST 126 Node

The original network model is presented in detail in [81]. Here we employ a compressed version of that network, shown in Fig. B-1. Our model contains $|\mathcal{B}|$ =84 nodes and 6 lines all connected to the feeder node in a star topology.

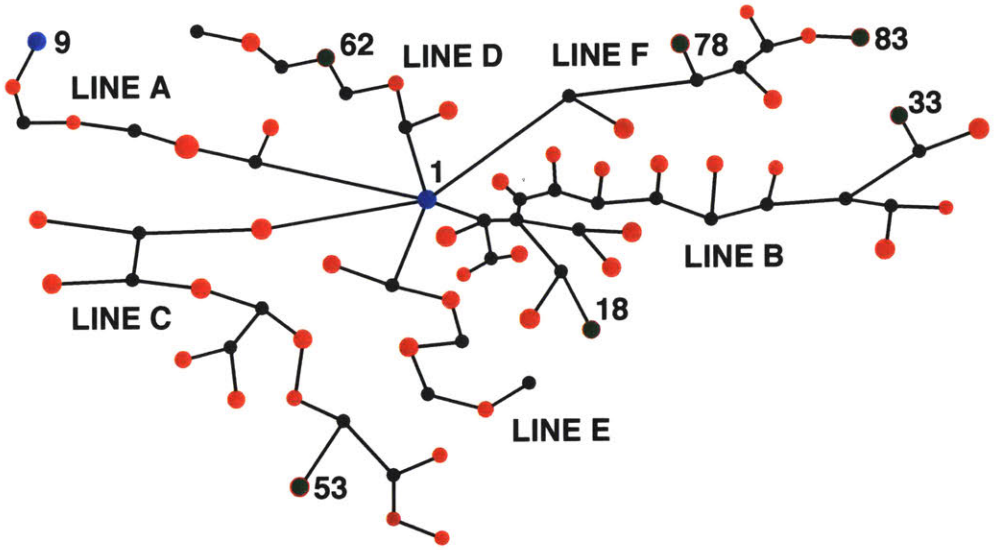


Figure B-1: An illustration of the distribution network model. The filled circles show the nodes $\mathcal{B}$. The red, blue and green filled circles indicate loads, local generators and both, respectively.

Details of base values and parameters are provided in Table B.18, with base values determined as typical to power systems in Japan [90]. Information on electrical lines/loads and generators can be found in Table B.19 and Table B.20, respectively. The parameters $\overline{P_{\mathrm{j}}^{\mathrm{L}}}$ are obtained randomly, to satisfy the total upper capacity values in Table B.19.

Table B.18: Base values and simulation parameters for J-CREST 126

Parameter	JST-CREST 126
$\lvert\mathcal{B}\rvert$	84
$\lvert\mathcal{T}\rvert$	83
f_{base}, Hz	50
v_{base}, kV	6.6
S_{base}, kVA	10^3
$\underline{v_{\mathbf{j}}}$, pu	0.95
$\overline{v_{\mathbf{j}}}$, pu	1.05
$\overline{S^2_{\mathbf{i},\mathbf{j}}}$, pu	2
$\overline{Q^{\text{G}}_{\mathbf{j}}}, -\underline{Q^{\text{G}}_{\mathbf{j}}}$	$0.15\overline{P^{\text{G}}_{\mathbf{j}}}$
$P^{\text{G}}_{\mathbf{j}}$	0
$\overline{P^{\text{L}}_{\mathbf{j}}}$	0
$\overline{Q^{\text{L}}_{\mathbf{j}}}$	0
$Q^{\text{L}}_{\mathbf{j}}$	0
$\beta^{\text{PL}}_{\mathbf{j}}$	$\in [400, 800]$
$\beta^{\text{PG}}_{\mathbf{j}}$	$\in [4, 8]$
$\beta^{\text{QG}}_{\mathbf{j}}$	$\in [4, 8]$
ξ	100
$\lambda^{\text{Q}}_{\mathbf{1}}$	1

Table B.19: Line and load node data for J-CREST 126

LINE	A	B	C	D	E	F
Nodes in $\mathcal{B}$	8	31	18	8	8	10
Branches in $\mathcal{T}$	1	14	6	1	1	4
Total # of loads $\mathbf{j}$	4	17	12	4	4	6
Sum of $\overline{P^{\text{L}}_{\mathbf{j}}}$ [MW]	0.84	3.56	2.51	0.84	0.84	1.26

Table B.20: Generation node data for JST-CREST 126

$j \in \mathcal{B}$	1	9	18	33	53	62	78	83
$\overline{P^{\text{G}}_{\mathbf{j}}}$ [MW]	10	5	0.6	1.2	0.6	0.3	1.8	3
$\overline{P^{\text{L}}_{\mathbf{j}}}$ [MW]	0	0	0.14	0.08	0.31	0.08	0.32	0.07

B.5 Electricity LMP Data

The locational marginal price (LMP) for the New England are is published online by the Independent System Operator (ISO-NE) [37]. The 5-minute real-time LMP data for May 14 2019 is queried and averaged over 12 periods to give hourly LMP data. This is the $\lambda^{P}_{j\#}$ in \$/kWh, and is provided below. Note that we take $\lambda^{Q}_{j\#} = 0.1\lambda^{P}_{j\#}$.

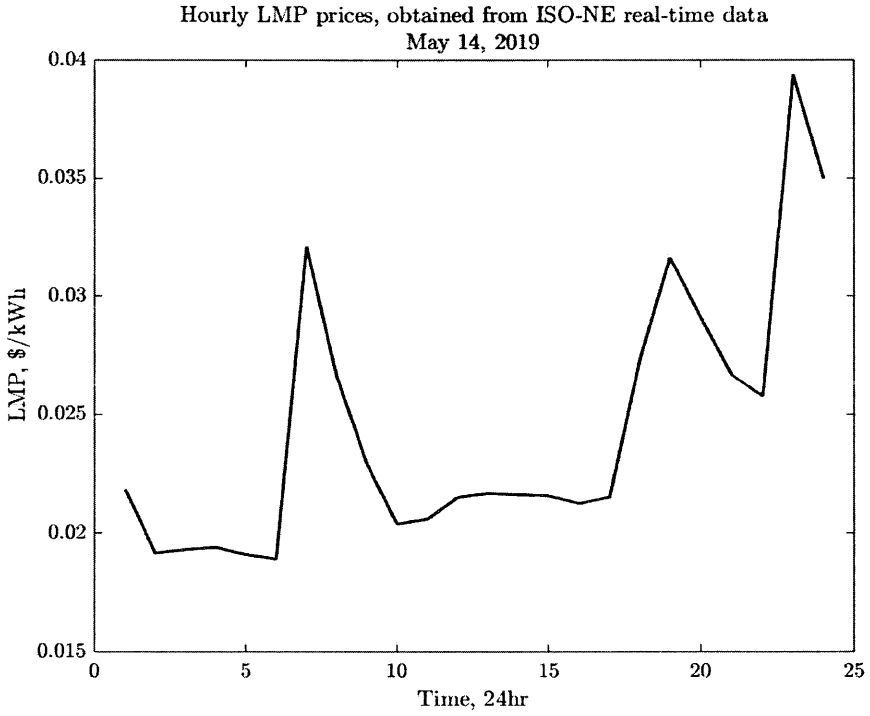


Figure B-2: Demand profile taken from the system-wide demand for ISO-NE.

Time (hr)	LMP ($/kWh)
00	0.0218
01	0.0192
02	0.0193
03	0.0194
04	0.0191
05	0.0189
06	0.0321
07	0.0266
08	0.0229
09	0.0204
10	0.0206
11	0.0215
12	0.0217
13	0.0216
14	0.0216
15	0.0212
16	0.0215
17	0.0273
18	0.0316
19	0.0291
20	0.0266
21	0.0258
22	0.0394
23	0.0350

Table B.21: Hourly electricity prices from the real-time WEM for ISO-NE.

B.6 Electricity Demand Data

The electricity demand profile is obtained from the ISO-NE reports on total recorded electricity demand for each five-minute interval [38], for real power. The report is queried for May 14, 2019, and normalized. The 5-minute data is then averaged over 12 periods to give hourly demand data.

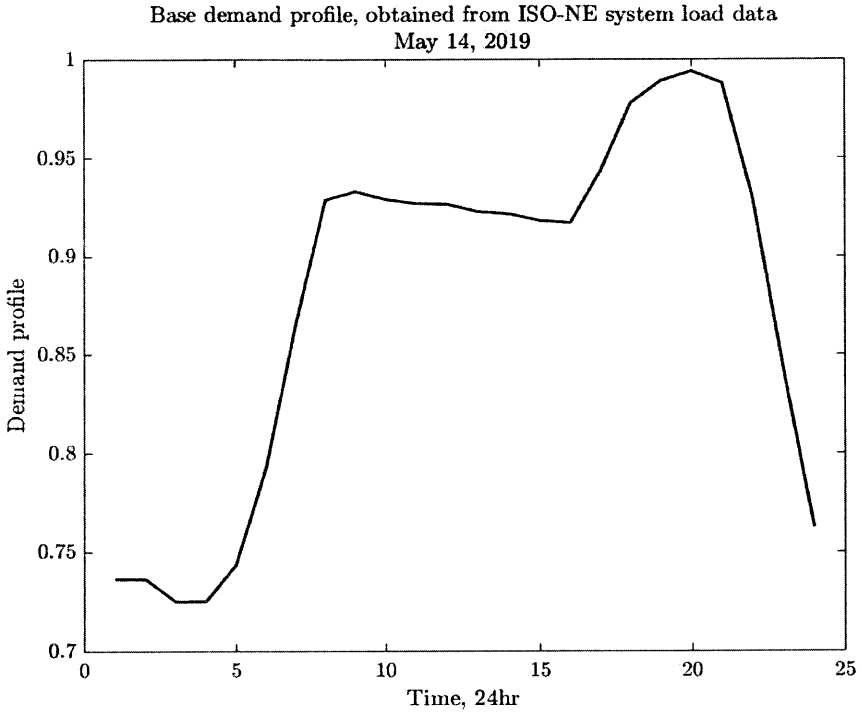


Figure B-3: Demand profile taken from the system-wide demand for ISO-NE.

Time (hr)	Demand coefficient
00	0.7366
01	0.7364
02	0.7251
03	0.7252
04	0.7437
05	0.7922
06	0.8646
07	0.9288
08	0.9330
09	0.9290
10	0.9268
11	0.9264
12	0.9227
13	0.9215
14	0.9181
15	0.9170
16	0.9434
17	0.9778
18	0.9892
19	0.9940
20	0.9880
21	0.9293
22	0.8423
23	0.7625

Table B.22: Hourly demand coefficient taken from the system-wide demand for ISO-NE.

B.7 Renewable Generation Forecasts

B.7.1 Small scale distributed residential PV

Generation profile for a 3.85kW distributed residential solar PV resource (small resource), obtained from NREL's System Advisory Model (SAM) tool, queried for May 14. This model is used for residential resources up to 10kW size. Resource data is for Phoenix, AZ, using the SunPower SPR-X21-335 module, and SMA America (SB3800TL-US-22, 240V) inverter. The DC to AC ratio is set to the default of 1.2. The desired array size is set to 4kWdc, giving a total AC capacity of 3.850kWac, and uses 1 inverter. All other parameters are left unchanged in the SAM setup.

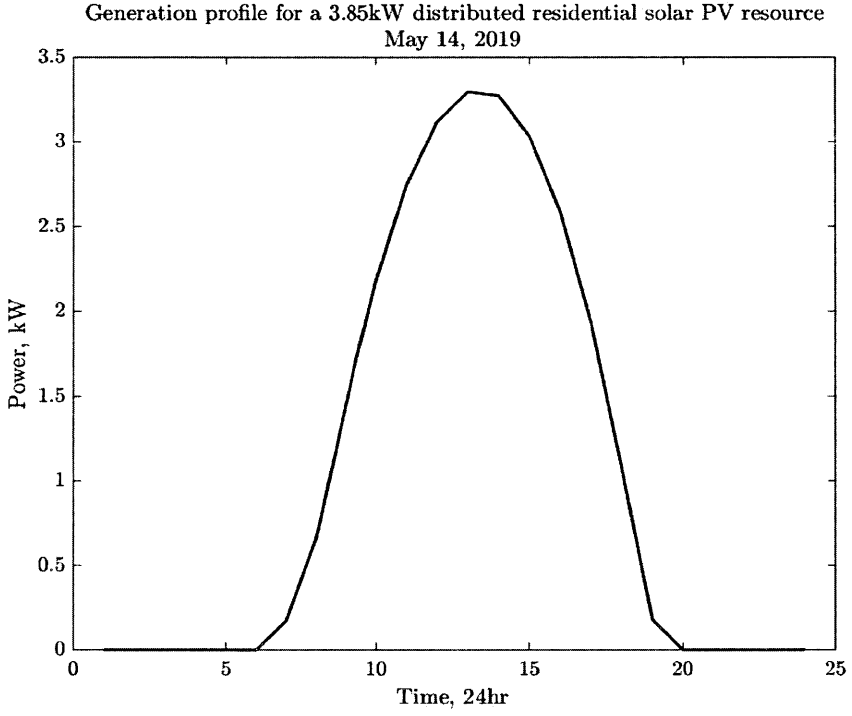


Figure B-4: Solar PV generation profile for small distributed residential resource, obtained from NREL's SAM tool.

Time (hr)	Power (kW)
00	0
01	0
02	0
03	0
04	0
05	0
06	0.207588
07	0.796325
08	1.75089
09	2.62495
10	3.29244
11	3.74114
12	3.95503
13	3.92816
14	3.63557
15	3.09874
16	2.31529
17	1.29041
18	0.21557
19	0
20	0
21	0
22	0
23	0

Table B.23: Generation profile for a 3.85kW distributed residential solar PV resource

B.7.2 Large scale distributed residential PV

Generation profile for a 11.55kW distributed residential solar PV resource (small resource), obtained from NREL's System Advisory Model (SAM) tool, queried for May 14. This model is used for residential resources larger than 10kW size. Resource data for Phoenix, AZ, using the SunPower SPR-X21-335 module, and SMA America (SB3800TL-US-22, 240V) inverter. The DC to AC ratio is set to the default of 1.2. The desired array size is set to 12kWdc, giving a total AC capacity of 11.550kWac, and uses 3 inverters. All other parameters are left unchanged in the SAM setup.

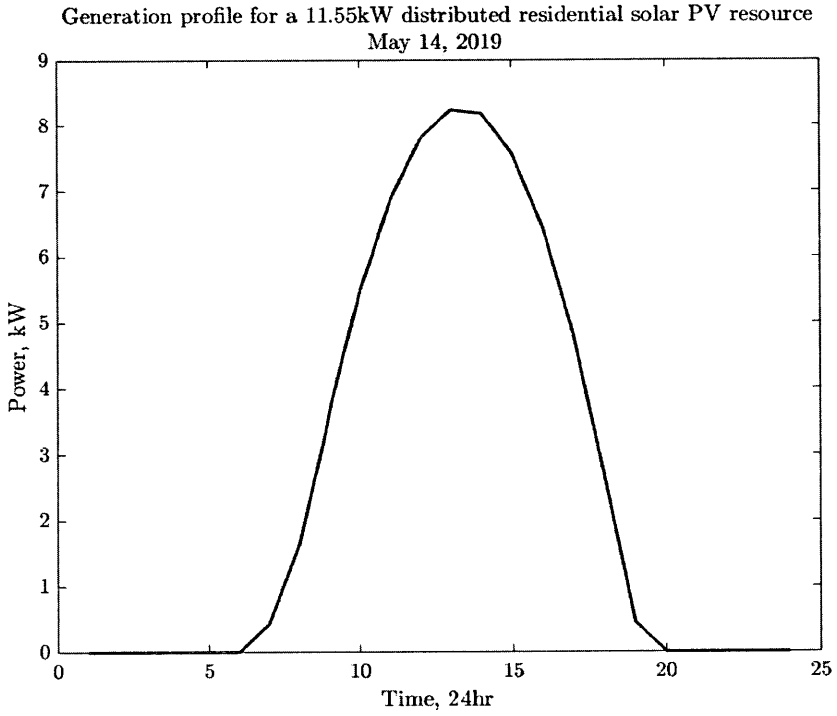


Figure B-5: Solar PV generation profile for large distributed residential resource, obtained from NREL's SAM tool.

Time (hr)	Power (kW)
00	0
01	0
02	0
03	0
04	0
05	0
06	0.518969
07	1.99081
08	4.37724
09	6.56236
10	8.23361
11	9.35284
12	9.88759
13	9.8204
14	9.08893
15	7.74686
16	5.78823
17	3.22603
18	0.538925
19	0
20	0
21	0
22	0
23	0

Table B.24: Generation profile for a 11.55kW distributed residential solar PV resource

B.7.3 Distributed commercial PV

Generation profile for a 185kW distributed commercial solar PV resource, obtained from NREL's System Advisory Model (SAM) tool, queried for May 14. Resource data is for Phoenix, AZ, using the SunPower SPR-E19-310-COM module, and SMA America (STP 60-US-10, 400V) inverter. The DC to AC ratio is set to the default of 1.2. The desired array size is set to 220kWdc, giving a total AC capacity of 179.580kWac. All other parameters are left unchanged in the SAM setup.

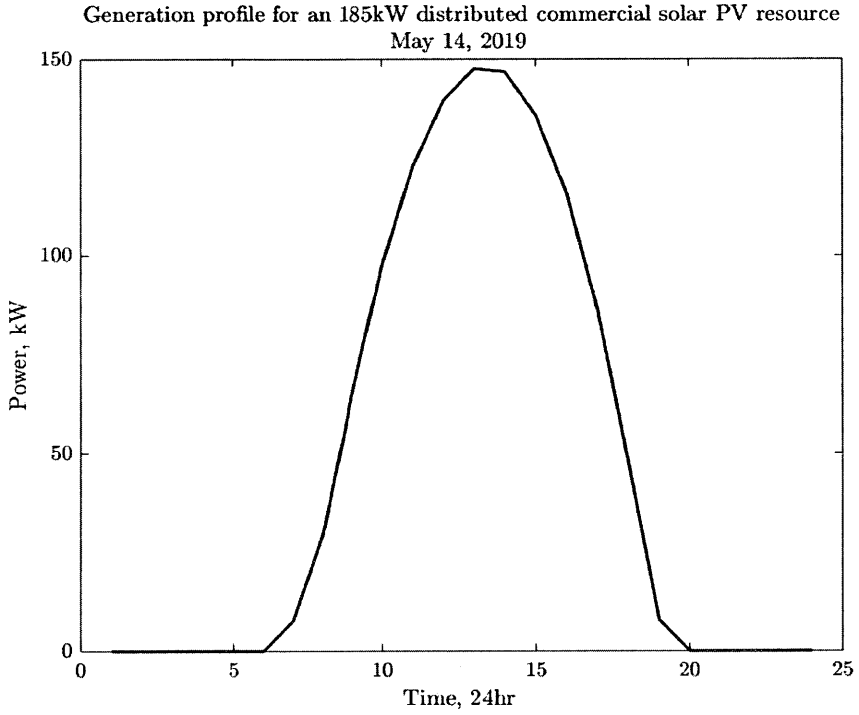


Figure B-6: Solar PV generation profile for distributed commercial resource, obtained from NREL's SAM tool.

Time (hr)	Power (kW)
00	0
01	0
02	0
03	0
04	0
05	0
06	9.23617
07	35.5421
08	78.1584
09	117.068
10	147.029
11	167.309
12	177.106
13	176.132
14	162.819
15	138.732
16	103.488
17	57.5034
18	9.51637
19	0
20	0
21	0
22	0
23	0

Table B.25: Generation profile for an 185kW distributed commercial solar PV resource

B.7.4 Wind Farm

Generation profile for a 600kW wind farm resource, obtained from NREL's System Advisory Model (SAM) tool, queried for May 14.

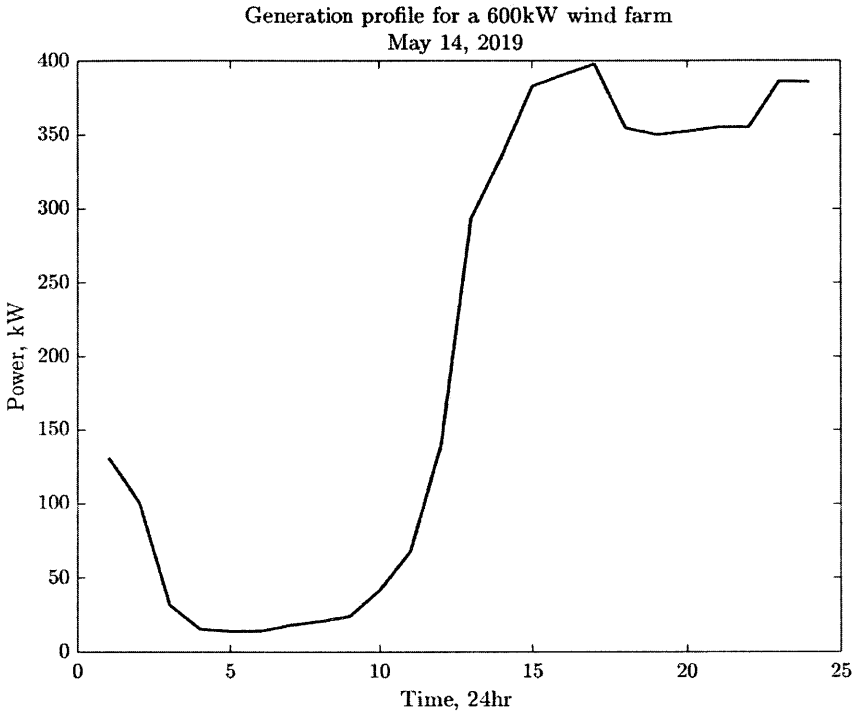


Figure B-7: Wind generation profile for wind farm, obtained from NREL's SAM tool.

Time (hr)	Power (kW)
00	157.248
01	120.554
02	38.0912
03	18.6316
04	16.78
05	16.9781
06	21.9101
07	24.9154
08	29.0332
09	50.1929
10	81.6055
11	167.594
12	352.588
13	402.553
14	459.247
15	468.504
16	477.276
17	425.394
18	420.121
19	422.718
20	426.086
21	426.088
22	462.448
23	462.949

Table B.26: Generation profile for a 600kW wind farm

Appendix C

MATLAB Code

C.1 Main Scripts

masterSolver.m

```
%% Script to determine best PAC parameters for implementation

% fnc = {1: solve with PAC, 2: parameter testing for PAC}
% testParam = {"L","gamma","rho"}
% gridID = {"3Node","13Node","123Node"}

function masterSolver(gridID,dataConfigCode,T,K,initCondStatus,fnc,testParam)

    % Default Values
    FontSize    = 12;                % Default Line Size
    LineWdith   = 1.5;               % Default Line Width
    MarkerSize  = 16;                % Default Line Width

    % General plotting settings
    set(0,'defaultAxesFontSize',FontSize);
    set(0,'defaultTextFontSize',FontSize);
    set(0,'defaultTextInterpreter','latex');
    set(0,'defaultLegendInterpreter','latex')
```

```
    set(0,'defaultAxesTickLabelInterpreter','latex');
    set(0,'defaultLineLinewidth',LineWdith);
    set(0,'defaultLineMarkerSize',MarkerSize);
    % set(0,'defaultFigureWindowStyle','docked')

    kInit = 2; eps = 1e-3;
    runSetup = 1; runPAC = 0;
    runCIOPF = 1; runADMM = 0;
    saveInd = 1; % to save fig and data needed for MOPF graphing
    % setup and solve the optimization problem
    solveOPF;

end
```

solveOPF.m

```
dbstop if error

%% Setup problem
if runSetup
    pathname = './networkdata/';
    filename = strcat('Network',gridID,'T',int2str(T),'NEW.mat'); % append NEW for the (

    if isfile(strcat(pathname,filename))
        load(strcat(pathname,filename))
    else
        fprintf('\n\t ---------------------- ERROR NETWORK FILE NOT FOUND\n');
        return;
    end

    % get problem data, in atomized form
    nodeData = strcat("[Hj,ej,ineqcon,limits,costs,rengenInds,battInds,Sindsj,Sinds_new,

    eval(nodeData);
```

```
% update vars for storage (decomposition already updated)
if ~isempty(battInds)

    Lj = cellfun(@numel, Lset); % size of owned vars of each atom
    Tj = cellfun(@numel, Tset); % size of each atom
    aInds = cumsum(Tj);
    numT = sum(Tj); % total number of atomized vars
    muInds = cumsum(Cj);

    % run update on B, Bj1, Bj2, nuInds (recreate)
    Oidx=[]; % rows of B
    for i=1:size(Oset,2)
        O_tmp=Oset{i}';
        atom=ones(size(O_tmp))*i;
        Oidx=[Oidx;[O_tmp, atom]];
    end

    Tidx=[]; % columns of B
    for i=1:size(Tset,2)
        T_tmp=Tset{i}';
        atom=ones(size(T_tmp))*i;
        Tidx=[Tidx;[T_tmp, atom]];
    end
    uniqueVars = unique(Oidx(:,1));
    rowgroup = arrayfun(@(x) find(Oidx(:,1)==x),uniqueVars,'UniformOutput',false);
    colgroup = arrayfun(@(x) find(Tidx(:,1)==x),uniqueVars,'UniformOutput',false);

    vals = [];
    rowidx = [];
    colidx = [];
    for i=1:size(uniqueVars,1)
        var = uniqueVars(i); % copied var
        row = rowgroup{i};
        col = colgroup{i};
        atomO = Oidx(row,2); % atoms with copy
        atomT = Tidx(col,2); % atoms needing var
```

```
    owner = setdiff(atomT,atomO); % find owner of var (in T, not in O)
    owneridx = col(atomT == owner); % find index in T of owner of var (in T, not
    copieridx = col(atomT ~= owner); % find index in T of copy of var (only in (

    rowidx = [rowidx,reshape(repmat(row', 2, 1),1,[])];
    colidx = [colidx,reshape([ones(1,size(copieridx,1))*owneridx; copieridx'],1,
    vals = [vals,repmat([-1,1],1,size(copieridx,1))];
    % a(row,column)
    % Oidx are rows, Tidx are columns
end

B = sparse(rowidx,colidx,vals,size(Oidx,1),size(Tidx,1));

% Construct Bj (this is B_(j,-) matrix)
Bj={}; Bj{N}={};
nuInds=zeros(1,N);
totalidx_1 = 0;
% build B_(-,j) which is needed for Lagrangian
Bj2={}; Bj2{N}={};
totalidx_2 = 0;
for j = 1:N
    % B_(j,-)
    numrows = size(Oset{j},2);
    Bj{j} = B(totalidx_1+1:(totalidx_1+numrows),:);
    totalidx_1 = totalidx_1 + numrows;
    nuInds(j) = totalidx_1;
    % B_(-,j)
    numcols = size(Tset{j},2);
    Bj2{j} = B(:,totalidx_2+1:(totalidx_2+numcols));
    totalidx_2 = totalidx_2 + numcols;
end

% build actuation projection matrix, PhiAct (P and Q are actuation vars)
indsAct = (13:18); % to get P and Q from every aj | will need to update for MOP
indsActrep = reshape(repmat(indsAct,T,1)',1,[]);
indsAct2rep = reshape(repmat((0:2),2*T,1)',1,[]); % 3 phases per var
```

```
        indsActAll = indsActrep + indsAct2rep;
        colsAct = reshape(repmat([0,aInds(1:N-1)],2*3*T,1),1,[]) + repmat(indsActAll,1,1
        PhiAct = sparse(1:2*3*N*T,colsAct,ones(1,2*3*N*T),2*3*N*T,numT); % 2*3*N*T

    end

    P_max = limits{10};
    Vr_min = limits{5}; Vr_max = limits{6};
    Vi_min = limits{7}; Vi_max = limits{8};
    xi = 50; % weighting for electrical losses & price of economics

    Ibranchr_max = ones(3*M*T,1);
    Ibranchi_max = ones(3*M*T,1);

    % get generated atomized objective functions
    [objfncs,globalObjfnc,L] = getObj_economic(N,T,costs,P_max);
%     [objfncs,globalObjfnc,L] = getObj(N,T,Vr_max,Vi_max);
%     [objfncs,globalObjfnc,L] = getObj_socialwelfare(N,M,T,xi,costs,Zij,RMatj,numChild
end

% create projection mats (memory issue for large problems so need to create)
allL = horzcat(Lset{:})'; % all owned vars stacked
skip = cumsum(Tj);

cols = [];
for j = 1:N
    if j == 1
        cols = vertcat(cols,(1:Lj(j))');
    else
        cols = vertcat(cols,((1:Lj(j))+skip(j-1))');
    end
end

PhiL = sparse(allL,cols,ones(1,size(allL,1)),numTotalVars,numT); % y = PhiL*a
invPhiL = sparse(1:numT,horzcat(Tset{:}),ones(1,numT),numT,numTotalVars); % a = invPhiL
G_tilda = blkdiag(Gj{:}); % total G_tilda
```

```
IrIndsAll = (1:3*N*T); % Ir is 1st set of vars
IiIndsAll = 3*1*N*T + (1:3*N*T); % Ii is 2nd set of vars
VrIndsAll = 3*2*N*T + (1:3*N*T); % Vr is 3rd set of vars
ViIndsAll = 3*3*N*T + (1:3*N*T); % Vi is 4th set of vars
PIndsAll = 3*4*N*T + (1:3*N*T); % P is 5th set of vars
QIndsAll = 3*5*N*T + (1:3*N*T); % Q is 6th set of vars
MCEaIndsAll = 3*6*N*T + (1:3*N*T); % a is 7th set of vars
MCEbIndsAll = 3*7*N*T + (1:3*N*T); % b is 8th set of vars
MCEcIndsAll = 3*8*N*T + (1:3*N*T); % c is 9th set of vars
MCEdIndsAll = 3*9*N*T + (1:3*N*T); % d is 10th set of vars

if ~isempty(battInds)
    % create Pdis and Pcharge indices
    sizeNewVars = (numTotalVars - 3*N*numVars*T)/2;
    PdisIndsAll = 3*N*numVars*T + (1:sizeNewVars);
    PchargeIndsAll = 3*N*numVars*T + sizeNewVars + (1:sizeNewVars);
else
    PdisIndsAll = [];
    PchargeIndsAll = [];
end

%% Run centrally if needed
if runCIOPF
    % run CIOPF

    % run ACIOPF
    H_tilda = blkdiag(Hj{:});
    e_tilda = vertcat(ej{:});

    % G_tilda = blkdiag(Gj{:}); % total G_tilda
    [aCIOPF, yCIOPF, muEq, muIneq, nuEq, error] = aciopfmin(N,M,T,numTotalVars,numT,glol

    if error == 1
        fprintf('\n\t ---------- ERROR IN ACIOPF\n');
        return;
```

```
    end
end

genSaveResFig(yCIOPF, N, M, T, rengenInds, battInds, Sindsj, Sinds_new, limits, saveInd,


if runPAC
    % determine function and setup PAC parameters
    if runSetup
        alpha = 0.5;
        if runPAC
            timePAC = zeros(N,K); % for timing

            % optimally determine parameters
            eigsPAC = eig(G_tilda'*G_tilda + B'*B);
            sigMinPAC = min(eigsPAC(eigsPAC > 1e-3));
            sigMaxPAC = max(eigsPAC);
            % primal
            gammaPAC = (2*alpha*L)/(2*sigMaxPAC + sigMinPAC);%(2*alpha*L+2)/(sigMinPAC),
            rhoPAC = 1/sqrt(gammaPAC*sigMaxPAC);
            gammaHPAC = gammaPAC;

            % ergodic
%             gammaPACErg = 4*Phi^2/(sigMinPAC*norm(Piy*xROPFMat(:,1),2)^2);
%             rhoPACErg = 1/sqrt(gammaPACErg*sigMaxPAC);
%             xiPACErg = sqrt(sigMinPAC/sigMaxPAC)/(2*Phi*norm(Piy*xROPFMat(:,1),2));
        end
    end

    switch fnc
        case 1
            % run PAC script normally for the test cases
            [aPAC,yPAC,muPAC,nuPAC,timePAC,totaltimePAC,aPACCVX,muPACCVX,nuPACCVX] = so
                runSetup,initCondStatus,N,M,K,kInit,eps,...
                numT,numTotalVars,numCVs,numTotalnG,Tj,muInds,nuInds,aInds,...
                G_tilda,B,Gj,Bj,Bj2,Hj,ej,PhiL,invPhiL,objfncs,...
```

```
        rhoPAC,gammaPAC,gammaPAC,aCIOPF,muEq,nuEq);

    fprintf('\n\t ------------------------------------------------------------');
    fprintf('\n\t Time elapsed: \n');
    fprintf(sprintf('\n\t Total time elapsed: %d for %d iterations',totaltimePA(

    fprintf('\n\t ------------------ Plotting Metrics\n');
    metrics = calculateMetrics(K,N,globalObjfnc,PhiL,PhiAct,G_tilda,B,aPAC,runC:
    plotMetrics(K,metrics,1,0,'');

case 2
    % run test for optimal PAC parameters
    rangeParam = optParamPAC(testParam,[L,gammaPAC,rhoPAC],alpha,sigMaxPAC,sigM:
    Lrange = rangeParam{1}; gammarange = rangeParam{2}; rhorange = rangeParam{3
    iters = size(Lrange,2);

    paramTestRes = {};
    paramTestRes{iters} = {};
    timeTestRes = zeros(1,iters);
    maxFeasTest = zeros(1,iters);
    maxConsisTest = zeros(1,iters);
    endFeasTest = zeros(1,iters);
    endConsisTest = zeros(1,iters);

    timeNow = datestr(now,'yyyymmddTHHMMSS'); % (ISO 8601) standard

    filepath = strcat('./resprocessing/simresults/',timeNow,'_paramTesting_',gr:
    mkdir(filepath)

    for i = 1:iters
        Ltest = Lrange(i);
        gammatest = gammarange(i);
        rhotest = rhorange(i);


        [aPAC,yPAC,muPAC,nuPAC,timePAC,totaltimePAC,aPACCVX,muPACCVX,nuPACCVX] =
```

```
                runSetup,initCondStatus,N,M,K,kInit,eps,...
                numT,numTotalVars,numCVs,numTotalnG,Tj,muInds,nuInds,aInds,...
                G_tilda,B,Gj,Bj,Bj2,Hj,ej,PhiL,invPhiL,objfncs,...
                rhotest,gammatest,gammatest,aCIOPF,muEq,nuEq);

            metrics = calculateMetrics(K,N,globalObjfnc,PhiL,PhiAct,G_tilda,B,aPAC,

            [maxFeas,maxFeasInd] = max(metrics(4,:));
            [maxConsis,maxConsisInd] = max(metrics(5,:));

            paramTestRes{i} = {Ltest,gammatest,rhotest,totaltimePAC,aPAC(:,end),...
                maxFeas,maxFeasInd,maxConsis,maxConsisInd,metrics(4,end),metrics(5,
            timeTestRes(i) = totaltimePAC;
            maxFeasTest(i) = maxFeas;
            maxConsisTest(i) = maxConsis;
            endFeasTest(i) = metrics(4,end);
            endConsisTest(i) = metrics(5,end);

            plotMetrics(K,metrics,0,1,strcat(filepath,'/',timeNow,'_','iter',int2st

            % save aPAC ending value, totaltimePAC, maximum distance to
            % {feasibility, consistancy}, current distance
            % to {feasibility, consistency}

        end

        filename = strcat(timeNow,'_paramTesting_',gridID);
%         writetable(paramTestRes,strcat(filepath,filename))

        save(strcat(filepath,'/',filename),'paramTestRes','rangeParam','timeTestRes

    otherwise
        fprintf('\n\t ---------------- ERROR: Unknown fnc\n');
end

end
```

```
% fprintf('\n\t ------------------------------------------------------------------');
% fprintf('\n\t Time elapsed: \n');
% fprintf(sprintf('\n\t Total time elapsed: %d for %d iterations',totaltimePAC/60,K));
%
%
% fprintf('\n\t ------------------- Plotting Metrics\n');
% metrics = calculateMetrics(k,globalObjfnc,PhiL,aPAC,runCIOPF,aCIOPF);
% plotMetrics(K,metrics);
%
% % process results to check things and produce relevant graphs
% fprintf('\n\t ------------------- Plotting Results\n');
% Apac = aPAC(:,k_);
% Ypac = PhiL*Apac;
% graphRes(N,Apac,Ypac,limits)


function rangeParam = optParamPAC(testParam,optParams,alpha,sigMax,sigMin,N)
% optParams order: [L, gamma, rho]
    L = optParams(1); gamma = optParams(2); rho = optParams(3);

%         gammaPAC = (2*alpha*L)/(2*sigMaxPAC + sigMinPAC);%(2*alpha*L+2)/(sigMinPAC);
%         rhoPAC = 1/sqrt(gammaPAC*sigMaxPAC);

    iters = 100;
    switch testParam
        case "L"
            Llow = 0.1;
            Lupp = 4160*3*N*100;
            Lrange = linspace(Llow,Lupp,iters);
            gammarange = (2*alpha*Lrange)/(2*sigMax + sigMin);
            rhorange = 1./sqrt(gammarange*sigMax);
%             gammarange = gamma*ones(size(Lrange));
%             rhorange = rho*ones(size(Lrange));
        case "gamma"
```

```
            gammalow = gamma/10;
            gammaupp = gamma*10;
            gammarange = linspace(gammalow,gammaupp,iters);
            Lrange = L*ones(1,iters);
%             rhorange = 1/sqrt(gammaPAC*sigMaxPAC);
            rhorange = rho*ones(1,iters);
        case "rho"
            rholow = rho/10;
            rhoupp = rho*10;
            rhorange = linspace(rholow,rhoupp,iters);
            Lrange = L*ones(1,iters);
            gammarange = gamma*ones(1,iters);
        otherwise
            fprintf('\n\t ---------- ERROR: Incorrect testing parameter\n');
    end

    rangeParam = {Lrange, gammarange, rhorange};

end

function [aPAC,yPAC,muPAC,nuPAC,timePAC,totaltimePAC,aPACCVX,muPACCVX,nuPACCVX] = solveI
            N,M,K,kInit,eps,numT,numTotalVars,numCVs,numTotalnG,Tj,muInds,nuInds,aInds,
            G_tilda,B,Gj,Bj,Bj2,Hj,ej,PhiL,invPhiL,objfncs,rhoPAC,gammaPAC,gammaHPAC,aC

    if runSetup
        % initial conditions
        y0 = zeros(numTotalVars,1); % GLOBAL VAR
        muEq0 = zeros(numTotalnG,1); % equality constraints
        nuEq0 = zeros(numCVs,1); % numCVs number of var copies

        switch initCondStatus
            case 1 % random initialization
                % for initial testing, just use zero :(
                a0 = invPhiL*y0; % atomized var
                % duals must maintain feasibility
                muEq0 = rhoPAC*gammaPAC*G_tilda*a0;
```

```
        nuEq0 = rhoPAC*gammaPAC*B*a0;

    case 2 % optimal central
        a0 = aCIOPF;
        muEq0 = -muEq;
        nuEq0 = -nuEq;
        y0 = PhiL*a0;

    case 3 % perturbed optimal
        eps = 1e-2;
        a0 = aCIOPF + 2*eps.*(rand(numT,1)-0.5);
        % duals must maintain feasibility
        muEq0 = rhoPAC*gammaPAC*G_tilda*a0;
        nuEq0 = rhoPAC*gammaPAC*B*a0;
        y0 = PhiL*a0;
end

% initialize vectors
aPAC = zeros(numT,K);
aPAC(:,1) = a0;
yPAC = zeros(numTotalVars,K);
yPAC(:,1) = y0;
muPAC = zeros(numTotalnG,K);
muPAC(:,1) = muEq0;
nuPAC = zeros(numCVs,K);
nuPAC(:,1) = nuEq0;

% initialize vectors for CVX (for testing)
aPACCVX = zeros(numT,K);
aPACCVX(:,1) = a0;
yPACCVX = zeros(numTotalVars,K);
yPACCVX(:,1) = y0;
muPACCVX = zeros(numTotalnG,K);
muPACCVX(:,1) = muEq0;
nuPACCVX = zeros(numCVs,K);
nuPACCVX(:,1) = nuEq0;
```

```
    timePAC = zeros(N,K);

end

fprintf('\n\t ---------- Finished setup\n');

% initialize models in Gurobi
fprintf('\n\t ---------- Initializing PAC\n');
[models,newTj] = initAtomsGur(objfncs,Tj,Hj,ej,rhoPAC);

mujInds = {}; mujInds{N} = {};
nujInds = {}; nujInds{N} = {};
ajInds = {}; ajInds{N} = {};
for j = 1:N
    if j == 1
        midx = 1; nidx = 1; aidx = 1;
    else
        midx = muInds(j-1)+1;
        nidx = nuInds(j-1)+1;
        aidx = aInds(j-1)+1;
    end
    mujInds{j} = midx:muInds(j);
    nujInds{j} = nidx:nuInds(j);
    ajInds{j} = aidx:aInds(j);
end

tstart = clock;
for k=kInit:K

% run PAC update
    [aPAC(:,k),muPAC(:,k),nuPAC(:,k),timePAC(:,k),aPACCVX(:,k),muPACCVX(:,k),nuPACC
        updatePACOPFg(models,aPAC(:,k-1),muPAC(:,k-1),nuPAC(:,k-1),...
        objfncs,Tj,Gj,Bj,Bj2,Hj,ej,mujInds,nujInds,ajInds,rhoPAC,gammaPAC,gammaHPAC,
        aPACCVX(:,k-1),muPACCVX(:,k-1),nuPACCVX(:,k-1));
```

```
        yPAC(:,k) = PhiL*aPAC(:,k);

        % print progress as necessary
        if rem(k,500)==0
            fprintf(sprintf('\n\t [PAC: Time %d/%d] Ran PAC',k,K));
        end
    end
    tend = clock;

    totaltimePAC = etime(tend,tstart);
end


function metrics = calculateMetrics(K,N,globalObjfnc,PhiL,PhiAct,G_tilda,B,aPAC,runCIOPI
    % set metrics for getting convergence
    metrics = zeros(9,K);

    if runCIOPF % Set metrics
        metrics(1,:) = (globalObjfnc(PhiL*aCIOPF)).*ones(1,K); % central cost
        metrics(2,1) = globalObjfnc(PhiL*aPAC(:,1)); % cost
        metrics(6,1) = globalObjfnc(PhiL*aPACCVX(:,1)); % cost
        metrics(3,1) = norm(PhiL*(aCIOPF - aPAC(:,1)),2); % dist opt cost
        metrics(7,1) = norm(PhiAct*(aCIOPF - aPACCVX(:,1)),2); % dist opt act
    end

    for k = 2:K
        metrics(2,k) = globalObjfnc(PhiL*aPAC(:,k));
        % distance to feasibility
        metrics(4,k) = norm(G_tilda*aPAC(:,k),2);
        % distance to consistency
        metrics(5,k) = norm(B*aPAC(:,k),2);

        metrics(6,k) = globalObjfnc(PhiL*aPACCVX(:,k));
        % distance to feasibility
        metrics(8,k) = norm(G_tilda*aPACCVX(:,k),2);
        % distance to consistency
```

```
        metrics(9,k) = norm(B*aPACCVX(:,k),2);

        if runCIOPF
            % distance to optimal actuation
            metrics(3,k) = norm(PhiAct*(aCIOPF - aPAC(:,k)),2);
            metrics(7,k) = norm(PhiAct*(aCIOPF - aPACCVX(:,k)),2);
        end
    end
end

function plotMetrics(K,metrics,visibility,saveInd,filepath)
    % plot the PAC convergence profiles
    kVec = 1:K;

    figure('visible',visibility);
%     figure;
    subplot(2,2,1);
    plot(kVec, metrics(1,kVec),'k'); hold on;
    plot(kVec, metrics(2,kVec),'b'); hold on;
%     plot(kVec, metrics(6,kVec),'g--'); hold on;
    title(sprintf('Costs')); legend('CIOPF','PAC');

    subplot(2,2,2);
    plot(kVec, metrics(3,kVec),'b'); hold on;
%     plot(kVec, metrics(7,kVec),'g--'); hold on;
    title(sprintf('Distance to Optimal Actuation')); legend('PAC');

    subplot(2,2,3);
    plot(kVec, metrics(4,kVec),'b'); hold on;
%     plot(kVec, metrics(8,kVec),'g--'); hold on;
    title(sprintf('Distance to Feasibility'));

    subplot(2,2,4);
    plot(kVec, metrics(5,kVec),'b'); hold on;
%     plot(kVec, metrics(9,kVec),'g'); hold on;
    title(sprintf('Distance to Consistency'));
```

```
    if saveInd
        if isempty(filepath)
            fprintf('\n\t ---------- ERROR: Empty filepath for PAC figure\n');
        else
            filename = strcat(filepath,'PACconvergence.fig');
            savefig(filename)
        end
    end

    if ~visibility % close figure that isn't displayed
        close(gcf);
    end
end


function genSaveResFig(y, N, M, T, rengenInds, battInds, Sindsj, Sinds_new, limits, save
    % generate graph of dispatch schedule from OPF solution
    generateResGraphs(y, N, M, T, rengenInds, battInds,Sindsj,Sinds_new, limits)

    timeNow = datestr(now,'yyyymmddTHHMMSS'); % (ISO 8601) standard
    filepath = './resprocessing/simresults/MOPF/';
    filename = strcat(timeNow,gridID,'_T',int2str(T),'_config',int2str(dataConfigCode)),
    if saveInd
        if isempty(filepath)
            fprintf('\n\t ---------- ERROR: Empty filepath for MOPF res figure\n')
        else
            f = strcat(filepath,filename);
            savefig(strcat(f,'.fig'))
            save(strcat(f,'.mat'),'y','N','M','T','rengenInds','battInds','Sindsj','Sin
        end
    end

    close(gcf);
end
```

```
function graphRes(N,aPAC,yPAC,limits,battInds)

    IrIndsAll = (1:3*N*T); % Ir is 1st set of vars
    IiIndsAll = 3*1*N*T + (1:3*N*T); % Ii is 2nd set of vars
    VrIndsAll = 3*2*N*T + (1:3*N*T); % Vr is 3rd set of vars
    ViIndsAll = 3*3*N*T + (1:3*N*T); % Vi is 4th set of vars
    PIndsAll = 3*4*N*T + (1:3*N*T); % P is 5th set of vars
    QIndsAll = 3*5*N*T + (1:3*N*T); % Q is 6th set of vars
    MCEaIndsAll = 3*6*N*T + (1:3*N*T); % a is 7th set of vars
    MCEbIndsAll = 3*7*N*T + (1:3*N*T); % b is 8th set of vars
    MCEcIndsAll = 3*8*N*T + (1:3*N*T); % c is 9th set of vars
    MCEdIndsAll = 3*9*N*T + (1:3*N*T); % d is 10th set of vars

    if ~isempty(battInds)
        % create Pdis and Pcharge indices
        sizeNewVars = (numTotalVars - 3*N*numVars*T)/2;
        PdisIndsAll = 3*N*numVars*T + (1:sizeNewVars);
        PchargeIndsAll = 3*N*numVars*T + sizeNewVars + (1:sizeNewVars);
    else
        PdisIndsAll = [];
        PchargeIndsAll = [];
    end

    Ir_min = limits{1}; Ir_max = limits{2};
    Ii_min = limits{3}; Ii_max = limits{4};
    Vr_min = limits{5}; Vr_max = limits{6};
    Vi_min = limits{7}; Vi_max = limits{8};
    P_min = limits{9}; P_max = limits{10};
    Q_min = limits{11}; Q_max = limits{12};

    resP = yPAC(PIndsAll);
    resQ = yPAC(QIndsAll);
    resV = yPAC(VrIndsAll) + 1i*yPAC(ViIndsAll);

    figure;
    subplot(3,1,1); hold on;
```

```
scatter(1:N*T,resP(1:3:end,:),'*','k');
scatter(1:N*T,[0;P_max(4:3:end,:)],'o','k'); % feeder has max unbounded
scatter(1:N*T,P_min(1:3:end,:),'o','k');
hline = refline([0 0]);
hline.Color = 'b';
hline.LineWidth = 0.5;
xlim([0 N+1]);
title('Phase A');
ylabel('Power, P [pu]');
subplot(3,1,2); hold on;
scatter(1:N*T,resP(2:3:end,:),'*','k');
scatter(1:N*T,[0;P_max(5:3:end,:)],'o','k'); % feeder has max unbounded
scatter(1:N*T,P_min(2:3:end,:),'o','k');
hline = refline([0 0]);
hline.Color = 'b';
hline.LineWidth = 0.5;
xlim([0 N+1]);
title('Phase B');
ylabel('Power, P [pu]');
subplot(3,1,3); hold on;
scatter(1:N*T,resP(3:3:end,:),'*','k');
scatter(1:N*T,[0;P_max(6:3:end,:)],'o','k'); % feeder has max unbounded
scatter(1:N*T,P_min(3:3:end,:),'o','k');
hline = refline([0 0]);
hline.Color = 'b';
hline.LineWidth = 0.5;
xlim([0 N+1]);
title('Phase C');
ylabel('Power, P [pu]');
xlabel('Node');

% Q
figure;
subplot(3,1,1); hold on;
scatter(1:N,resQ(1:3:end,:),'*','k');
scatter(1:N,[0;Q_max(4:3:end,:)],'o','k'); % feeder has max unbounded
```

```
scatter(1:N,Q_min(1:3:end,:),'o','k');
hline = refline([0 0]);
hline.Color = 'b';
hline.LineWidth = 0.5;
xlim([0 N+1]);
title('Phase A');
ylabel('Reactive Power, Q [pu]');
subplot(3,1,2); hold on;
scatter(1:N,resQ(2:3:end,:),'*','k');
scatter(1:N,[0;Q_max(5:3:end,:)],'o','k'); % feeder has max unbounded
scatter(1:N,Q_min(2:3:end,:),'o','k');
hline = refline([0 0]);
hline.Color = 'b';
hline.LineWidth = 0.5;
xlim([0 N+1]);
title('Phase B');
ylabel('Reactive Power, Q [pu]');
subplot(3,1,3); hold on;
scatter(1:N,resQ(3:3:end,:),'*','k');
scatter(1:N,[0;Q_max(6:3:end,:)],'o','k'); % feeder has max unbounded
scatter(1:N,Q_min(3:3:end,:),'o','k');
hline = refline([0 0]);
hline.Color = 'b';
hline.LineWidth = 0.5;
xlim([0 N+1]);
title('Phase C');
ylabel('Reactive Power, Q [pu]');
xlabel('Node');


% plot voltages
V_min = [ones(3,1); ones((N-1)*3,1).*0.95];
V_max = [ones(3,1); ones((N-1)*3,1).*1.05];

figure;
subplot(3,1,1); hold on;
```

```
    plot(1:N,V_min(1:3:end),'k');
    plot(1:N,V_max(1:3:end),'k');
    plot(1:N,abs(resV(1:3:end)),'b');
    % scatter(1:N,abs(resV(1:3:end)),'.','k');
    xlim([0 N+1]);
    subplot(3,1,2); hold on;
    plot(1:N,V_min(2:3:end),'k');
    plot(1:N,V_max(2:3:end),'k');
    plot(1:N,abs(resV(2:3:end)),'b');
    % scatter(1:N,abs(resV(2:3:end)),'.','k');
    xlim([0 N+1]);
    subplot(3,1,3); hold on;
    plot(1:N,V_min(3:3:end),'k');
    plot(1:N,V_max(3:3:end),'k');
    plot(1:N,abs(resV(3:3:end)),'b');
    % scatter(1:N,abs(resV(3:3:end)),'.','k');
    xlim([0 N+1]);
end
```

C.2 Atomized Central Solver

aciopf.m

```
function [ aOpt, yOpt, muEq, muIneq, nuEq, error ] = aciopfmin(N,M,T,numTotalVars,numT,(
    error = 0;

    % yalmip
    a = sdpvar(numT,1);
    y = sdpvar(numTotalVars,1);

    c1 = [y == PhiL*a]; % definition
    c2 = [G_tilda*a == zeros(size(G_tilda,1),1)]; % muEq
    c3 = [H_tilda*a <= e_tilda]; % muIneq
```

```
    c4 = [B*a == zeros(size(B,1),1)]; % nuEq

    ops = sdpsettings('solver','gurobi');

    optimize([c1;c2;c3;c4],globalObjfnc(y),ops);

    aOpt = value(a);
    yOpt = value(y);
    muEq = dual(c2);
    muIneq = dual(c3);
    nuEq = dual(c4);

end
```

C.3 Implementation of PAC Algorithm

initAtomsGur.m

```
function [models, newNj] = initAtomsGur(objfncs,Tj,Hj,ej,rho)
    N = length(Tj);

    models{1} = {}; models{N} = {};
    newNj{1} = 0; newNj{N} = 0;
    for j = 1:N
        NTj = Tj(j); % variable length for node j
        objfnc = objfncs{j};

        % WITH YALMIP
        % Define variables
        aj = sdpvar(NTj,1,'full');
        % Define constraints
        constraints = [Hj{j}*aj <= ej{j}];
        % Update with constant quad function
        % note: linear and constant obj values change every iteration
```

```
        objfnc = @(aj,j_,len) objfnc(aj,j_,len) + (1/(2*rho))*(aj)'*(aj);
        % set YALMIP and solver options
        options = sdpsettings('solver','gurobi');
        modelj = export(constraints, objfnc(aj,j,NTj), options);

        models{j} = modelj;
        newNj{j} = size(modelj.obj,1);
    end
end
```

updatePACOPFg.m

```
function [aPACNew,muPACNew,nuPACNew,timeNew,aPACNewCVX,muPACNewCVX,nuPACNewCVX] =...
    updatePACOPFg(models,aPACOld,muPACOld,nuPACOld,objfncs,Tj,Gj,Bj,Bj2,Hj,ej,...
    muInds,nuInds,aInds,rho,gamma,gammaH,aPACOldCVX,muPACOldCVX,nuPACOldCVX)

    aPACNew = aPACOld; % yPACNew: y[k+1]
    muPACNew = muPACOld; % muPACNew: mu[k+1]
    nuPACNew = nuPACOld; % nuPACNew: nu[k+1]
    muBarPACOld = muPACOld; % muBarPACNew: muBar[k+1]
    nuBarPACOld = nuPACOld; % nuBarPACNew: nuBar[k+1]

    % for performance testing:
%     aPACNewCVX = aPACOldCVX; % yPACNew: y[k+1]
%     muPACNewCVX = muPACOldCVX; % muPACNew: mu[k+1]
%     nuPACNewCVX = nuPACOldCVX; % nuPACNew: nu[k+1]
%     muBarPACOldCVX = muPACOldCVX; % muBarPACNew: muBar[k+1]
%     nuBarPACOldCVX = nuPACOldCVX; % nuBarPACNew: nuBar[k+1]

    N = length(Tj);
    timeNew = zeros(N,1);

    % index vars for parallel computing
    parvars = {}; parvars{N} = {};

    % Price Predictions
```

```
for j = 1:N
    mujInds = muInds{j};
    nujInds = nuInds{j};
    ajInds = aInds{j};

    ajPACOld = aPACOld(ajInds);
    muBarPACOld(mujInds) = muPACOld(mujInds) + gammaH*rho*Gj{j}*ajPACOld;
    nuBarPACOld(nujInds) = nuPACOld(nujInds) + gammaH*rho*Bj{j}*aPACOld;

    ajPACOldCVX = aPACOldCVX(ajInds);
    muBarPACOldCVX(mujInds) = muPACOldCVX(mujInds) + gammaH*rho*Gj{j}*ajPACOldCVX;
    nuBarPACOldCVX(nujInds) = nuPACOldCVX(nujInds) + gammaH*rho*Bj{j}*aPACOldCVX;

    parvars{j} = {muBarPACOld(mujInds), ajPACOld, Gj{j}, Bj2{j}, Tj(j)}; % length(a
end

aPACNewVals = {}; aPACNewVals{N} = {};

% Update Primal Variables
parfor j = 1:N
    muBarj = parvars{j}{1};
    ajOld = parvars{j}{2};
    Gjatom = parvars{j}{3};
    Bj2atom = parvars{j}{4};
    NTj = parvars{j}{5};

    model = models{j};

    % update matrices for objective cost
    cLinj = model.obj';
    cConstj = model.objcon;

    % update cost function matrices
    cConstj = cConstj + 1/(2*rho)*dot(ajOld,ajOld); % proximal term
    cLinj(1:NTj) = cLinj(1:NTj) - (1/rho)*ajOld'; % proximal term
    cLinj(1:NTj) = cLinj(1:NTj) + muBarj'*Gjatom; % local dual var term
```

```
        cLinj(1:NTj) = cLinj(1:NTj) + nuBarPACOld'*Bj2atom;

        tic;
        [modelG, resG] = atomicGurobiOPT(model, full(cLinj), cConstj);
        timeNew(j) = toc;

        if strcmp(resG.status, 'OPTIMAL')
            augA = resG.x;
            aPACNewVals{j} = augA(1:NTj,:);
        else
          fprintf('PAC  Optimization returned status: %s\n', resG.status);
          aPACNewVals{j} = ajOld; % keep same old values
        end

        % for testing:
%         % augment objective for Lagrangian
%         objfncj = @(aj,j_,len) objfncj(aj,j_,len) + muBarj'*Gjatom*aj;
%         objfncj = @(aj,j_,len) objfncj(aj,j_,len) + (1/(2*rho))*(aj-ajOld)'*(aj-ajOld)
%         objfncj = @(aj,j_,len) objfncj(aj,j_,len) + nuBarPACOld'*Bj2atom*aj;
%
% %         tic;
%         cvx_begin quiet
%             variable aj(NTj);
%             minimize (objfncj(aj,j,NTj));
%             subject to
%                 Hj_atom*aj <= ej_atom;
%         cvx_end
% %         timeNew(j) = toc;
%
%         if sum(isnan(aj)) > 0
%             fprintf('[ropfmin.m] ERROR! UN SOLVABLE!');
%         else
%             aPACNewVals{j} = aj;
%         end

    end
```

```
    % added for parallel implementation
    for j = 1:N
        ajInds = aInds{j};
        ajPACNew = aPACNewVals{j};
        if size(ajInds') ~= size(ajPACNew)
            fprintf('** INCORRECT DIMENSIONS. ERROR. **');
        end
        aPACNew(ajInds) = ajPACNew;
    end

    % Update Dual Variables
    for j = 1:N
        mujInds = muInds{j};
        nujInds = nuInds{j};
        ajInds = aInds{j};

        muPACNew(mujInds) = muPACOld(mujInds) + gamma*rho*Gj{j}*aPACNewVals{j};
        nuPACNew(nujInds) = nuPACOld(nujInds) + gamma*rho*Bj{j}*aPACNew;

    end


    % For testing: check against CVX implementation
    % Update Primal Variables
%     for j = 1:N
%         ej_atom = ej{j};
%         Hj_atom = Hj{j};
%         objfncj = objfncs{j};
%
%         ajPACOldCVX = aPACOldCVX(aInds{j});
%         muBarj = muBarPACOldCVX(muInds{j});
%
%         % update Lagrangian
%         objfncj = @(aj,j_,len) objfncj(aj,j_,len) + (1/(2*rho))*(aj-ajPACOldCVX)'*(aj-
%         objfncj = @(aj,j_,len) objfncj(aj,j_,len) + muBarj'*Gj{j}*aj;
```

```
%         objfncj = @(aj,j_,len) objfncj(aj,j_,len) + nuBarPACOld'*Bj2{j}*aj;
%
%         cvx_begin quiet
%             cvx_solver sedumi
% %             cvx_precision high ;
%             variable aj(Tj(j));
%             minimize (objfncj(aj,j,Tj(j)));
%             subject to
%                 Hj_atom*aj <= ej_atom;
%         cvx_end
%         if sum(isnan(aj)) > 0
%             fprintf('[cvxPAC] ERROR! UN SOLVABLE!');
%         else
%             aPACNewCVX(aInds{j}) = aj; % yj;
%         end
%     end
%
%   % Update Dual Variables, CVX
%     for j = 1:N
%         mujInds = muInds{j};
%         nujInds = nuInds{j};
%         ajInds = aInds{j};
%
%         muPACNewCVX(mujInds) = muPACOldCVX(mujInds) + gamma*rho*Gj{j}*aPACNewCVX(ajIn
%         nuPACNewCVX(nujInds) = nuPACOldCVX(nujInds) + gamma*rho*Bj{j}*aPACNewCVX;
%     end


end
```

atomicGurobiOPT.m

```
function [model,result] = atomicGurobiOPT(model, cLinj, cConstj)
    % set objective function
    model.obj = cLinj';
    model.objcon = cConstj;
```

```
    % set some parameters
    params = struct('OutputFlag',0);
    result = gurobi(model,params);
end
```

C.4 Atomization for PAC

setupNetwork.m

```
%% GENERATE MODEL
% General functions

function setupNetwork(networkname,T) % T = num time periods
    % run network
    modelpath = strcat('../models/Network',networkname,'Model.m');
    run(modelpath);

    Y = zeros(3*N,3*N); % same for each time period
    A = zeros(3*M,3*N);
    Zlist = {}; Zlist{M} = {};

    RMatj={}; RMatj{N}={}; % resistance matrices for calculating line losses for atoms
    branchParent = min(arcs(:,1:2),[],2); % gets which node owns the branch flow variab
    % need: RVec*(Ibranchr^2 + Ibranchi^2)

    % create network admittance matrix
    for i=1:size(arcs,1)
        l=arcs(i,1);%line
        r=arcs(i,2);%row
        linelength=arcs(i,3);
        c_val = admittances{arcs(i,4)};
        Y([3*l-2:3*l],[3*r-2:3*r])=-c_val/linelength/YBase;
        Y([3*r-2:3*r],[3*l-2:3*l])=-c_val/linelength/YBase;
```

```
    Y([3*l-2:3*l],[3*l-2:3*l])=Y([3*l-2:3*l],[3*l-2:3*l])+c_val/linelength/YBase;
    Y([3*r-2:3*r],[3*r-2:3*r])=Y([3*r-2:3*r],[3*r-2:3*r])+c_val/linelength/YBase;

    % for impedance model
    z_val = impedances{arcs(i,4)};
    Zlist{i} = z_val*linelength/ZBase;

    % negative incidence matrix for voltage differences
    A([3*i-2:3*i],[3*l-2:3*l]) = eye(3);
    A([3*i-2:3*i],[3*r-2:3*r]) = -eye(3);

    parent = branchParent(i);
    RMatj{parent} = blkdiag(RMatj{parent},real(z_val));
end

Zij = blkdiag(Zlist{:}); % 3 phase, complex

% Generate the PAC decomposition
% variables
Ir=sdpvar(3*N*T,1);
Ii=sdpvar(3*N*T,1);
Vr=sdpvar(3*N*T,1);
Vi=sdpvar(3*N*T,1);
P=sdpvar(3*N*T,1);
Q=sdpvar(3*N*T,1);
a=sdpvar(3*N*T,1);     % MCE vars
b=sdpvar(3*N*T,1);     % MCE vars
c=sdpvar(3*N*T,1);     % MCE vars
d=sdpvar(3*N*T,1);     % MCE vars
Ibranchr=sdpvar(3*M*T,1); % for Z representation
Ibranchi=sdpvar(3*M*T,1); % for Z representation

V=Vr+1i*Vi;
I=Ir+1i*Ii;
Ibranch = Ibranchr + 1i*Ibranchi;
```

```
    F = [];
    for t = 1:T
        inds = 3*N*(t-1)+1:3*N*t;
        indsLine = 3*M*(t-1)+1:3*M*t;
        F = [F;
            Ir(inds) == real(A'*Ibranch(indsLine));
            Ii(inds) == imag(A'*Ibranch(indsLine));
            P(inds) == a(inds) + b(inds);
            Q(inds) == -c(inds) + d(inds);
            real(A*V(inds)) == real(Zij*Ibranch(indsLine));
            imag(A*V(inds)) == imag(Zij*Ibranch(indsLine));
            ];
    end

    % set up physical constraints (G matrix, Gx = 0) using Yalmip
    [model,recoverymodel] = export(F,0,sdpsettings('solver','Gurobi'));

    % get the atomized vectors and matrices
    decomposePAC

    numVars = (3*N*10 + 3*M*2)*T; % update num vars to be total number of variables (gl
    nG = (4*3*N+2*3*M)*T; % update to be total number of equality constraints (global)

    % save file to .mat to load
    filename = strcat('../Network',networkname,'T',int2str(T),'NEW.mat');
%     filename = strcat('../Network',networkname,'.mat');
    save(filename,'SBase','VBase','ZBase','IBase','YBase','Stran','N','M','Neigh','pare
        'Y','Zij','RMatj','A','numVars','nG','nH','Lset','Cset','Oset','Tset','Gj','Hji
        'aInds','numT','Cj','muInds','nuInds','G_tilda','PhiL','invPhiL','B','Bj','Bj2',
% 'G_tilda','PhiL','invPhiL',
end
```

decomposePAC.m

```
numVars = (size(model.obj,1)-3*M*2*T)/N/T/3; % N nodes, 3 phases per variable, M lines,
```

```
% Gx == 0
G=full(model.A);
% G = model.A; % trying without using full format of matrix
nG = (size(G,1)-3*M*2*T)/(3*N*T); % number of equality constraints per node, per phase

% Hx <= e
nH = 28; % number of inequality constraints per node, per phase

%% process G matrix to replace 'rabab' (dummy) set with zeros
G((G>0 & G<1e-25) | (G<0 & G>-1e-25)) = 0;

%% generate decomposition profile
% Set L - set of indices of variables owned by atom
Lset={}; Lset{N}={};
% Set C -- determines which rows of G belong to each atom, j
Cset={}; Cset{N}={};
% Set O -- copied variables required
Oset={}; Oset{N}={};
% Matrix G without primary variables then put the indices in G
Gtmp={}; Gtmp{N}={};
% Set T (union of L and O: all the vars for each atom)
Tset={}; Tset{N}={};
% Constraint submatrix
Gj={}; Gj{N}={}; % inizializzo il set
% Phi_L projection matrix, goes from atomized vars to global var
PhiLmats={}; PhiLmats{N}={};
invPhiLmats={}; invPhiLmats{N}={};
% decompose the matrix of the inequality constraints to obtain the one necessary for the
% create a new decomposition for Ax <= b, (var names: Hx <= e)
% get indices of matrix of inequality constraints for each atom, Ax <= b
Hjidx={}; Hjidx{N}={};


% iterate through nodal vars
for j = 1:N
    % owned vars (nodal vars only, all time periods)
```

```
    ind1 = (0:numVars*T-1)*3*N + 3*(j-1) + 1; % beginning index + phases*node + offset
    ind1rep = reshape(repmat(ind1,3,1),1,[]);
    ind2rep = reshape(repmat((0:2),numVars*T,1)',1,[]); % 3 phases per var
    Lset{j} = ind1rep + ind2rep;

    % rows of matrix G - for single period only
    ind1 = (0:nG-1)*3*N + 3*(j-1) + 1; % beginning index + phases*node + offset of 1
    ind1rep = reshape(repmat(ind1,3,1),1,[]);
    ind2rep = reshape(repmat((0:2),nG,1)',1,[]); % 3 phases per var
    tempInds = ind1rep + ind2rep;
    Cset{j} = tempInds;  %+ ind3rep

    % rows of matrix H (nodal vars only, all time periods)
    ind1 = (0:nH*T-1)*3*N + 3*(j-1) + 1; % beginning index + phases*node + offset of 1
    ind1rep = reshape(repmat(ind1,3,1),1,[]);
    ind2rep = reshape(repmat((0:2),nH*T,1)',1,[]); % 3 phases per var
    idx = ind1rep + ind2rep;
    Hjidx{j} = idx;
end


% iterate through branch vars, account only for single period for now.
Lindstart = numVars*3*N*T; % branch vars come after nodal vars
Cindstart = nG*3*N; % Iij const come after nodal var defn, per period
Hindstart = nH*3*N*T; % Iij inequality constraints come after all others
indRep = 3*M*T;
for m=1:M
    j=arcs(m,1);
    k=arcs(m,2);

    vinds = Lindstart + [3*m-2:3*m];
    cinds = Cindstart + [3*m-2:3*m];
    hinds = Hindstart + [3*m-2:3*m];

    temprep = reshape(repmat((0:T-1),3,1),1,[])*(3*(M));
    vindsrep = reshape(repmat(vinds,T,1)',1,[]);
```

```
    vindsrep = vindsrep + temprep;
    hindsrep = reshape(repmat(hinds,T,1)',1,[]);
    hindsrep = hindsrep + temprep;

    % upstream owns the var, downstream copies the var
    if j < k
        % j is upstream, k is downstream
        Lset{j} = [Lset{j},vindsrep,vindsrep+indRep];
%         Oset{k} = [Oset{k},vinds,vinds+indRep];
        Cset{j} = [Cset{j}, cinds,cinds+indRep/T];
        Hjidx{j} = [Hjidx{j}, hindsrep,hindsrep+indRep];
    elseif j > k
        % k is upstream, j is downstream
        Lset{k} = [Lset{k},vindsrep,vindsrep+indRep];
%         Oset{j} = [Oset{j},vinds];
        Cset{k} = [Cset{k}, cinds,cinds+indRep/T];
        Hjidx{k} = [Hjidx{k}, hindsrep,hindsrep+indRep];
    else
        sprintf('\n\t ---------------- ERROR IN DECOMPOSITION: Branch',m,'\n');
        return;
    end
end

% iterate through atoms again
for j = 1:N
    % update Cset for multiperiod
    Cinds = Cset{j};
    timerep = reshape(repmat((0:T-1),size(Cinds,2),1),1,[])*(nG*3*N+2*3*M); % 3 phases,
    Cinds = repmat(Cset{j},1,T);
    Cset{j} = Cinds + timerep;
%
    % copies of nodal vars needed
    row=Cset{j};
    Gtmp=(G(row,:));
    col=Lset{j};
    Gtmp(:,col) = 0; % set owned indices to 0
```

```
    [nonzeroRow,nonzeroCol] = find(Gtmp); % get needed var copies
    Oset{j} = unique(nonzeroCol)';

    % total set of vars
    Tset{j}=[cell2mat(Lset(j)) cell2mat(Oset(j))];

    % atomic equality constraints
    Gj{j} = G(row,Tset{j});


    % Inequality constraint mat
%     % get rows of matrix H - nodal constraints only
%     ind1 = (0:nH-1)*3*N*T + 3*(j-1) + 1; % beginning index + phases*node + offset of
%     ind1rep = reshape(repmat(ind1,3,1),1,[]);
%     ind2rep = reshape(repmat((0:2),nH*T,1)',1,[]); % 3 phases per var
%     idx = ind1rep + ind2rep;
%     % add rows of matrix H for branch constraints
%     Hindstart = nH*3*N*T;
%     timerep = reshape(repmat((0:T-1),size(idx,2),1),1,[])*Hindstart; % all nodal and
%     idxrep = repmat(idx,1,T);
%     Lset{j} = Linds + timerep;

    % copied nodal vars (fixed for phases)
%     aux_vars = Oset{j} - N*3*2;
%     aux_rows = [(aux_vars + 3*N*4) , (aux_vars + 3*N*6)];
%     Hjidx{j}=[Hjidx{j},aux_rows];
end


Lj = cellfun(@numel, Lset); % size of owned vars of each atom
Oj = cellfun(@numel, Oset); % size of copies of each atom
numCVs = sum(Oj); % total number of copied variables
Tj = cellfun(@numel, Tset); % size of each atom
aInds = cumsum(Tj);
numT = sum(Tj); % total number of atomized vars
Cj = cellfun(@numel, Cset); % num of rows of G owned by each atom
```

```
muInds = cumsum(Cj);
G_tilda = blkdiag(Gj{:}); % total G_tilda


allL = horzcat(Lset{:})'; % all owned vars stacked
skip = cumsum(Tj);

cols = [];
for j = 1:N
    if j == 1
        cols = vertcat(cols,(1:Lj(j))');
    else
        cols = vertcat(cols,((1:Lj(j))+skip(j-1))');
    end
end

PhiL = sparse(allL,cols,ones(1,size(allL,1)),(numVars*3*N+3*M*2)*T,numT);
invPhiL = sparse(1:numT,horzcat(Tset{:}),ones(1,numT),numT,(numVars*3*N+3*M*2)*T);

% Build coordination matrix, B
Oidx=[]; % rows of B
for i=1:size(Oset,2)
    O_tmp=Oset{i}';
    atom=ones(size(O_tmp))*i;
    Oidx=[Oidx;[O_tmp, atom]];
end

Tidx=[]; % columns of B
for i=1:size(Tset,2)
    T_tmp=Tset{i}';
    atom=ones(size(T_tmp))*i;
    Tidx=[Tidx;[T_tmp, atom]];
end

uniqueVars = unique(Oidx(:,1));
rowgroup = arrayfun(@(x) find(Oidx(:,1)==x),uniqueVars,'UniformOutput',false);
```

```
colgroup = arrayfun(@(x) find(Tidx(:,1)==x),uniqueVars,'UniformOutput',false);

% rowidx = 1:size(Oidx,1);
vals = [];
rowidx = [];
colidx = [];
for i=1:size(uniqueVars,1)
    var = uniqueVars(i); % copied var
    row = rowgroup{i};
    col = colgroup{i};
    atomO = Oidx(row,2); % atoms with copy
    atomT = Tidx(col,2); % atoms needing var
    owner = setdiff(atomT,atomO); % find owner of var (in T, not in O)
    owneridx = col(atomT == owner); % find index in T of owner of var (in T, not in O)
    copieridx = col(atomT ~= owner); % find index in T of copy of var (only in O)

    rowidx = [rowidx,reshape(repmat(row', 2, 1),1,[])];
    colidx = [colidx,reshape([ones(1,size(copieridx,1))*owneridx; copieridx'],1,[])];
    vals = [vals,repmat([-1,1],1,size(copieridx,1))];
    % a(row,column)
    % Oidx are rows, Tidx are columns
end

B = sparse(rowidx,colidx,vals,size(Oidx,1),size(Tidx,1));

% Construct Bj (this is B_(j,-) matrix)
Bj={}; Bj{N}={};
nuInds=zeros(1,N);
totalidx_1 = 0;
% build B_(-,j) which is needed for Lagrangian
Bj2={}; Bj2{N}={};
totalidx_2 = 0;
for j = 1:N
    % B_(j,-)
    numrows = size(Oset{j},2);
    Bj{j} = B(totalidx_1+1:(totalidx_1+numrows),:);
```

```
    totalidx_1 = totalidx_1 + numrows;
    nuInds(j) = totalidx_1;
    % B_(-,j)
    numcols = size(Tset{j},2);
    Bj2{j} = B(:,totalidx_2+1:(totalidx_2+numcols));
    totalidx_2 = totalidx_2 + numcols;
end


% build actuation projection matrix, PhiAct (P and Q are actuation vars)
indsAct = (13:18); % to get P and Q from every aj | will need to update for MOPF
indsActrep = reshape(repmat(indsAct,T,1)',1,[]);
indsAct2rep = reshape(repmat((0:2),2*T,1)',1,[]); % 3 phases per var
indsActAll = indsActrep + indsAct2rep;
colsAct = reshape(repmat([0,aInds(1:N-1)],2*3*T,1),1,[]) + repmat(indsActAll,1,N); % 2*
PhiAct = sparse(1:2*3*N*T,colsAct,ones(1,2*3*N*T),2*3*N*T,numT); % 2*3*N*T
```

C.5 OPF Problem Setup

C.5.1 Main Setup

setLimCost.m

```
%% GENERAL FUNCTIONS
% Case data
% Function to assemble variable bounds and costs
% contains the logic around DERs

function [Hj,ej,ineqcon,limits,costs,rengenInds,battInds,Sindsj,Sinds_new,newTotalVars,
    V_min,V_max,delta_min,delta_max,N,M,A,Tset,Lset,Oset,Gj,Cj,nG,Hjidx,SBase,Stran,num

    P_max = zeros(N*3*T,1);
    P_min = zeros(N*3*T,1);
    Q_max = zeros(N*3*T,1);
```

```
Q_min = zeros(N*3*T,1);

% set the first period loads, P and Q
for i = 1:size(loadsP,1)
    row = loadsP(i,:);
    idx = ((row(1)-1)*3 + 1):((row(1)-1)*3 + 3);

    P_max(idx) = P_max(idx) - row(2:4)'; % loads negative
    P_min(idx) = P_min(idx) - row(2:4)'; % assume no DR at load node
end

for i = 1:size(loadsP,1)
    row = loadsQ(i,:);
    idx = ((row(1)-1)*3 + 1):((row(1)-1)*3 + 3);

    Q_max(idx) = Q_max(idx) - row(2:4)'; % loads negative
    Q_min(idx) = Q_min(idx) - row(2:4)'; % assume no DR at load node
end

if T > 1
    % use demand profile for additional time periods, P and Q both
    [mm_dP,mm_dQ] = demandProfiles(demandProfileID,N,T-1);
    dP = reshape(repmat(mm_dP',3,1),[],1); % replicate for 3 phases
    dQ = reshape(repmat(mm_dQ',3,1),[],1); % replicate for 3 phases
    % apply demand profile to time periods for t>1
    P_max(3*N+1:end) = repmat(P_max(1:3*N),T-1,1).*dP;
    P_min(3*N+1:end) = repmat(P_min(1:3*N),T-1,1).*dP;
    Q_max(3*N+1:end) = repmat(Q_max(1:3*N),T-1,1).*dQ;
    Q_min(3*N+1:end) = repmat(Q_min(1:3*N),T-1,1).*dQ;
end

% get feeder P, Q bounds
% S = P^2 + Q^2; P = VrIr + ViIi; Q = -VrIi + ViIr
feedP = 0; feedQ = 0;
if size(loadsP,1) > 0
    maxPLoad = sum(reshape(-P_max,3*N,[]));
```

```
    feedP = maxPLoad*1.5;
end

if size(loadsQ,1) > 0
    maxQLoad = sum(reshape(-Q_max,3*N,[]));
    feedQ = min([sqrt((Stran*SBase)^2 - feedP.^2)',maxQLoad'*1.2],[],2)';
end

cost_aP = zeros(N*3*T,1); % quadratic coefficient for DERs
cost_bP = zeros(N*3*T,1); % linear coefficient for DERs
cost_cP = zeros(N*3*T,1); % constant terms for DERs
cost_bPs = zeros(N*3*T,1); % linear terms for storage (P pos and neg issue)
cost_bQ = zeros(N*3*T,1); % linear coefficient term, relevant for feeder

H2jidx={}; H2jidx{N}={};
Qpfinds = []; Ppfinds = []; pfpos = []; pfneg = [];
H2ind = 1;
Prindsj = {}; Prindsj{N} = {}; % indices for generators with ramp constraints
H6jidx={}; H6jidx{N}={}; % for storage ramp constraints
H6ind = 1;
delPGup = [];
delPGdown = [];
% add controllable DGs
for i = 1:size(gensP,1)
    row = gensP(i,:);
    j = row(1);
    idx = ((j-1)*3 + 1):((j-1)*3 + 3);
    idxT = repmat(idx,1,T)+3*N*reshape(repmat((0:T-1),3,1),1,[]);
    P_max(idxT) = P_max(idxT) + repmat(row(2:4)',T,1);%row(2:4)'; % generators posi
    P_min(idxT) = P_min(idxT) + repmat(row(5:7)',T,1);%row(5:7)';

    % prepare indices for DG capability curve
    if (j ~= 1 && row(8) ~= 0 && row(9) ~= 0)
        Ppfinds = [Ppfinds idx];
        Qpfinds = [Qpfinds idx];
        pfpos = [pfpos row(8)];
```

```
        pfneg = [pfneg row(9)];
        numConst = T; % one per time period; only for pos constraint; other easy to
        H2jidx{j} = [H2ind:H2ind+numConst-1];
        H2ind = H2ind + numConst;
    end

    % prepare indices for ramp constraints
    if (j ~= 1 && row(10) ~= 0 && row(11) ~= 0)
        Prindsj{j} = repmat(idx,1,T) + (3*N)*reshape(repmat((0:T-1),length(idx),1),1
        if size(row)> 11
            Pt0 = sum(row(12:14)); % get initial power injection
        else
            Pt0 = 0; % assume no initial setpoint for the generator
        end

        delPGup = [delPGup row(10)*ones(1,T)]; % replicated for easy vector constra
        delPGdown = [delPGdown -row(11)*ones(1,T)]; % replicated for easy vector co
        delPGup(end-T+1) = delPGup(end-T+1) + Pt0; % adjust first time period
        delPGdown(end-T+1) = delPGdown(end-T+1) + Pt0; % adjust first time period

        % have index only for ramp up constraint; other is easy to calculate
        numConst = T; % one constraint per time period (all phases sum)
        H6jidx{j} = [H6ind:H6ind+numConst-1];
        H6ind = H6ind + numConst;
    end
end
% adjust for multi-period
Ppfinds = repmat(Ppfinds,1,T) + 3*N*reshape(repmat((0:T-1),length(Ppfinds),1),1,[]),
Qpfinds = repmat(Qpfinds,1,T) + 3*N*reshape(repmat((0:T-1),length(Qpfinds),1),1,[]),
pfpos = repmat(pfpos,1,T) + 3*N*reshape(repmat((0:T-1),length(pfpos),1),1,[]);
pfneg = repmat(pfneg,1,T) + 3*N*reshape(repmat((0:T-1),length(pfneg),1),1,[]);

for i = 1:size(gensQ,1)
    row = gensQ(i,:);
    j = row(1);
    idx = ((j-1)*3 + 1):((j-1)*3 + 3);
```

```
    idxT = repmat(idx,1,T)+3*N*reshape(repmat((0:T-1),3,1),1,[]);

    Q_max(idxT) = Q_max(idxT) + repmat(row(2:4)',T,1);%row(2:4)'; % generators posit
    Q_min(idxT) = Q_min(idxT) + repmat(row(5:7)',T,1);%row(5:7)';
end

rengenInds = [];
% add renewable generators
for i = 1:size(rengens,2)
    row = rengens{i};
    j = row{1};
    rengenInds = [rengenInds,j];

    % not all nodes have 3-phase connections
    phases = row{3}; phases = num2cell(convertStringsToChars(phases));
    V = [1 2 3]; S = {'a','b','c'};
    [tf, idx] =ismember(phases,S);
    phaseOut = V(idx(tf));
    numPhases = length(phaseOut);
    idx = (j-1)*3 + phaseOut;
    idxT = repmat(idx,1,T) + reshape(repmat((0:T-1),numPhases,1),1,[])*3*N;

    rengenspecs_i = rengenmodel(row{4},row{2},T);

    if rengenspecs_i == 0
        return;
    end

    mm_gP = reshape(repmat(rengenspecs_i('genP'),1,numPhases)',[],1);
    mm_gQ = reshape(repmat(rengenspecs_i('genQ'),1,numPhases)',[],1);

    P_max(idxT) = P_max(idxT) + mm_gP; % generators positive
    P_min(idxT) = P_min(idxT) + mm_gP.*rengenspecs_i('curtailLimit');
    Q_max(idxT) = Q_max(idxT) + mm_gQ; % generators positive
    Q_min(idxT) = Q_min(idxT) + mm_gQ.*rengenspecs_i('curtailLimit');
end
```

```
% update P and Q for feeder
Pfeed = reshape(repmat(feedP,3,1),[],1);
Qfeed = reshape(repmat(feedQ,3,1),[],1);

feederInds = repmat(1:3,1,T);
timeInds = (0:T-1)*3*N;
reptimeInds = reshape(repmat(timeInds,3,1),1,[]);
repfeederInds = feederInds + reptimeInds;

P_max(repfeederInds) = Pfeed;
Q_max(repfeederInds) = Qfeed;
Q_min(repfeederInds) = -Qfeed; % feeder can provide pos or neg Q

% DR uses percentage of base load as set point
for i = 1:size(demRes,1)
    row = demRes(i,:);
    j = row(1);
    idx = ((j-1)*3 + 1):((j-1)*3 + 3);
    idxT = repmat(idx,1,T)+3*N*reshape(repmat((0:T-1),3,1),1,[]);

    P_max(idxT) = P_max(idxT).*(1-row(2)); % loads are negative so need to change P_
end

% storage considers capacity constraint
Sindsj = {}; Sindsj{N} = {}; % original vars needed for storage, P
Sinds_new = []; % new variables required for storage, P_dis and P_charge
Sinds_new_sizej = {}; Sinds_new_sizej{N} = {};
Sidx = 1; % idx for the new storage vars
etaCAll = []; etaDAll = []; betaSDAll = [];
minCapAll = []; maxCapAll = []; % min and max capacity for device
initCapsAll = []; % initial battery capacity for device
delPup = [];
delPdown = [];
G2jidx={}; G2jidx{N}={}; G2ind = 1; % for added equality constraints
```

```
H3jidx={}; H3jidx{N}={}; % for storage Pdis nonnegativity
H3ind = 1;
H4jidx={}; H4jidx{N}={}; % for storage capacity constraints
H4ind = 1;
H5jidx={}; H5jidx{N}={}; % for storage ramp constraints
H5ind = 1;
battInds = [];
for i = 1:size(storage,1)
    % storage looks like [node battID phases initCharge] --phases 'ab' or 'abc'
    row = storage(i,:);
    j = str2num(row(1));
    battID = row(2);
    battInds = [battInds j];
    phases = row(3); phases = num2cell(convertStringsToChars(phases));
    V = [1 2 3]; S = {'a','b','c'};
    [tf, idx] =ismember(phases,S);
    phaseOut = V(idx(tf));
    numPhases = length(phaseOut);
    idx = (j-1)*3 + phaseOut;
    idxrepT = repmat((j-1)*3+phaseOut,1,T) + reshape(repmat((0:T-1),length(phaseOut)

    % load battery model
    bSpecs = battmodel(battID); % batt models are in W

    % storage costs (linear term only)
    cost_bPs(idxrepT) = bSpecs('wearCost').*ones(numPhases*T,1); % wear cost to inf

    % storage related regular constraints:
    % powercharge <= P <= powerdischarge

    P_max(idxrepT) = P_max(idxrepT) + bSpecs('power'); % discharge power
    P_min(idxrepT) = P_min(idxrepT) - bSpecs('power'); % charge power (is negative)
    Sindsj{j} = repmat(idx,1,T) + (3*N)*reshape(repmat((0:T-1),length(idx),1),1,[]),

    % storage only constraints:
    % P = P_dis - P_charge --- these vars exist only for storage nodes,
```

```
% P_dis >= 0            % per time period (aggregate over phases)
% P_charge >= 0         % per time period (aggregate over phases)
% upper cap constraint
% lower cap constraint

% add 2 new vars per phase, but index only one of them
Sinds_new = [Sinds_new Sidx]; % index for P_dis (P_charge easy to calculate afte
Sinds_new_sizej{j} = numPhases;
Sidx = Sidx + 1;
% for capacity constraints
etaCAll = [etaCAll bSpecs('etaC')];
etaDAll = [etaDAll bSpecs('etaD')];
betaSDAll = [betaSDAll bSpecs('betaSD')];
minCapAll = [minCapAll (bSpecs('capacity')*(1-bSpecs('maxDOD')))*ones(1,T)]; %
maxCapAll = [maxCapAll bSpecs('capacity')*ones(1,T)]; % replicated for easy vect
initCapsAll = [initCapsAll str2double(row(4))*ones(1,T)]; % replicated for easy
% for ramp constraints
Pt0 = str2double(row(5:numPhases+4)); % get initial power injection
delPup = [delPup bSpecs('rampUp')*ones(1,T)]; % replicated for easy vector const
delPdown = [delPdown -bSpecs('rampDown')*ones(1,T)]; % replicated for easy vect
delPup(end-T+1) = delPup(end-T+1) + sum(Pt0); % adjust first time period
delPdown(end-T+1) = delPdown(end-T+1) + sum(Pt0); % adjust first time period

% new inequality constraints: non-negativity of P_dis P_charge,
% upper bound for both as well
% two constraint per time period for each P_dis, P_charge
% have index only for P_dis >= 0; the others are easy to calculate
numConst = 1;
H3jidx{j} = [H3ind:H3ind+numConst-1];
H3ind = H3ind + numConst;

% new inequality constraints: capacity
% have index only for upper bound on capacity; other is easy to calculate
numConst = T; % one constraint per time period (all phases sum)
H4jidx{j} = [H4ind:H4ind+numConst-1];
H4ind = H4ind + numConst;
```

```
    % new inequality constraints: ramp for battery
    % have index only for ramp up constraint; other is easy to calculate
    numConst = T; % one constraint per time period (all phases sum)
    H5jidx{j} = [H5ind:H5ind+numConst-1];
    H5ind = H5ind + numConst;

    % new equality constraints: defn of P, done in order
    % need numPhases: relating Pdis and Pcharge to P (P is per phase)
    G2jidx{j} = [G2ind:G2ind+numPhases-1];
    G2ind = G2ind + numPhases;
end

% adjust for multiperiod
if ~ isempty(storage)
    Sinds_new = repmat(Sinds_new,1,T) + max(Sinds_new)*reshape(repmat((0:T-1),lengt
end

% change to per unit analysis
P_max = P_max./SBase;
P_min = P_min./SBase;
Q_max = Q_max./SBase;
Q_min = Q_min./SBase;
minCapAll = minCapAll./SBase;
maxCapAll = maxCapAll./SBase;
initCapsAll = initCapsAll./SBase;
delPup = delPup./SBase;
delPdown = delPdown./SBase;
delPGup = delPGup./SBase;
delPGdown = delPGdown./SBase;

% get feeder costs (estimates of market data for LMPs)
mm_p = retrieveLMPs(T);
mm_q = mm_p.*0.1; % Q price is 10% of P price (random setting)

cost_bP(repfeederInds) = mm_p;
```

```
    cost_bQ(repfeederInds) = mm_q;

    % convert costs from $/kWh -> $/pu
    costs = {cost_aP*SBase^2,cost_bP*SBase,cost_cP,cost_bPs*SBase,cost_bQ*SBase};

    % get current limits from pre-processing
    [Ir_min,Ir_max,Ii_min,Ii_max,Vr_min,Vr_max,Vi_min,Vi_max] = ...
        genLimits(N,T,V_min,V_max,delta_min,delta_max,P_min,P_max,Q_min,Q_max,Y,battInds

    limits{1} = Ir_min; limits{2} = Ir_max;
    limits{3} = Ii_min; limits{4} = Ii_max;
    limits{5} = Vr_min; limits{6} = Vr_max;
    limits{7} = Vi_min; limits{8} = Vi_max;
    limits{9} = P_min; limits{10} = P_max;
    limits{11} = Q_min; limits{12} = Q_max;

    % get atomized inequality matrices
    [Hj,ej,ineqcon,battInds,newTotalVars,newTotalnG,Tset,Lset,Gj,Cj] = ineqCon(N,M,A,T,
        pfpos,pfneg,betaSDAll,etaCAll,etaDAll,maxCapAll,minCapAll,initCapsAll,delPup,de
        G2jidx,Hjidx,H2jidx,H3jidx,H4jidx,H5jidx,H6jidx,numVars,nG,Y,Zij,costs); % Pind

end

function cost_P = retrieveLMPs(T)

    dateid = 'May142019';
    pathname = strcat('./networkdata/dataprofiles/',dateid,'/');
    filename = strcat('isonedata_',dateid,'.mat');

    if isfile(strcat(pathname,filename))
        load(strcat(pathname,filename),'mm_p','units'); % $/kWh for ISONE LMP
    else
        fprintf('\n\t ---------------- ERROR PRICE DATA FILE NOT FOUND\n');
        return;
    end
```

```
    % assume all profile data is needed per hour
    % profile data is every 5 mins from ISONE -> convert to hourly
    n = 12;

    mm_p_reshaped = reshape(mm_p,n,[]);
    mm_p_hourly = mean(mm_p_reshaped)';
    mm_p_hourly = reshape(repmat(mm_p_hourly',3,1),[],1); % duplicate for all phases

    cost_P = mm_p_hourly(1:3*T); % 3 phases, T timesteps
end

% Limit Pre-processing to determine tighter bounds for MCE
function [Ir_min,Ir_max,Ii_min,Ii_max,Vr_min,Vr_max,Vi_min,Vi_max] = ...
    genLimits(N,T,V_min,V_max,de_min,de_max,P_min,P_max,Q_min,Q_max,Y,battInds,INDSTOPR(

    % V_min and V_max are scalars right now
    es1 = V_max.*(cos(de_max)+1i*sin(de_max));
    es2 = V_max.*(cos(de_min)+1i*sin(de_min));
    es3 = V_max.*(cos(de_max)+1i*sin(de_min));
    es4 = V_max.*(cos(de_min)+1i*sin(de_max));
    es5 = V_min.*(cos(de_max)+1i*sin(de_max));
    es6 = V_min.*(cos(de_min)+1i*sin(de_min));
    es7 = V_min.*(cos(de_max)+1i*sin(de_min));
    es8 = V_min.*(cos(de_min)+1i*sin(de_max));
    esAll = [es1; es2; es3; es4; es5; es6; es7; es8];

    Vr_min = min(real(esAll));
    Vr_max = max(real(esAll));
    Vi_min = min(imag(esAll));
    Vi_max = max(imag(esAll));

    % single period
    Vr_min_t1 = [ones(3,1);Vr_min.*ones(3*(N-1),1)];
    Vr_max_t1 = [ones(3,1);Vr_max.*ones(3*(N-1),1)];
    Vi_min_t1 = [zeros(3,1);Vi_min.*ones(3*(N-1),1)];
    Vi_max_t1 = [zeros(3,1);Vi_max.*ones(3*(N-1),1)];
```

```
Vr_min = repmat(Vr_min_t1,T,1);
Vr_max = repmat(Vr_max_t1,T,1);
Vi_min = repmat(Vi_min_t1,T,1);
Vi_max = repmat(Vi_max_t1,T,1);

% optimization variables
Ir = sdpvar(1,1);
Ii = sdpvar(1,1);
Vr = sdpvar(1,1);
Vi = sdpvar(1,1);
P = Vr.*Ir+Vi.*Ii; % definition
Q = -Vr.*Ii+Vi.*Ir; % definition

% initialize constraints
Ir_max = zeros(N*3*T,1);
Ir_min = zeros(N*3*T,1);
Ii_max = zeros(N*3*T,1);
Ii_min = zeros(N*3*T,1);

feederInds = repmat(1:3,1,T);
timeInds = (0:T-1)*3*N;
reptimeInds = reshape(repmat(timeInds,3,1),1,[]);
repfeederInds = feederInds + reptimeInds;
% need to get current limits based on all the powers
% run for every node (excluding feeder), every phase, every time period
for j = 4:N*3*T
    % check if in feeder indss
    if ismember(j,repfeederInds)
        continue
    end

    % Ir_max
    F=[
        P_min(j) <= P <= P_max(j),...
        Q_min(j) <= Q <= Q_max(j),...
```

```
    Vr_min(j) <= Vr <= Vr_max(j),...
    Vi_min(j) <= Vi <= Vi_max(j),...
    ];
obj=Ir;

ops = sdpsettings('solver','ipopt','verbose',0);
optimize(F,-obj,ops);
Ir_max(j) = value(Ir); % just in case

%Ir_min
F=[
    P_min(j,1) <= P <= P_max(j,1),...
    Q_min(j,1) <= Q <= Q_max(j,1),...
    Vr_min(j) <= Vr <= Vr_max(j),...
    Vi_min(j) <= Vi <= Vi_max(j),...
    ];
obj=Ir;

ops = sdpsettings('solver','ipopt','verbose',0);
optimize(F,obj,ops);
Ir_min(j) = value(Ir); % just in case

%Ii_max
F=[
    P_min(j,1) <= P <= P_max(j,1),...
    Q_min(j,1) <= Q <= Q_max(j,1),...
    Vr_min(j) <= Vr <= Vr_max(j),...
    Vi_min(j) <= Vi <= Vi_max(j),...
    ];
obj=Ii;

ops = sdpsettings('solver','ipopt','verbose',0);
optimize(F,-obj,ops);
Ii_max(j) = value(Ii);

%Ir_min
```

```
        F=[
            P_min(j,1) <= P <=P_max(j,1),...
            Q_min(j,1) <= Q <= Q_max(j,1),...
            Vr_min(j) <= Vr <= Vr_max(j),...
            Vi_min(j) <= Vi <= Vi_max(j),...
            ];
        obj=Ii;

        ops = sdpsettings('solver','ipopt','verbose',0);
        optimize(F,obj,ops);
        Ii_min(j) = value(Ii);

    end

    % Feeder limits determined by feeder power and transformer rating
    % S = P^2 + Q^2; P = VrIr + ViIi; Q = -VrIi + ViIr; Vi = 0
    Ir_min(repfeederInds) = P_min(repfeederInds)./Vr_max(repfeederInds); % note Pmin dep
    Ir_max(repfeederInds) = P_max(repfeederInds)./Vr_min(repfeederInds);
    Ii_min(repfeederInds) = Q_min(repfeederInds)./Vr_min(repfeederInds); % note: Qmin =
    Ii_max(repfeederInds) = Q_max(repfeederInds)./Vr_min(repfeederInds);

    % STORAGE - SET Ii BOUNDS TO ZERO; ONLY CONSIDERING REAL POWER
    if ~isempty(battInds)
        for i=1:length(battInds)
            battID = battInds(i);
            idx = (battID-1)*3 + (1:3);
            idxrepT = repmat(idx,1,T) + reshape(repmat((0:T-1),3,1),1,[])*3*N;
            Ii_min(idxrepT) = zeros(3*T,1);
            Ii_max(idxrepT) = zeros(3*T,1);
        end
    end

%     save('currentLims_123_NEW','Ir_min','Ir_max','Ii_min','Ii_max','Vi_min','Vi_max',
end
```

```
% Assemble inequality constraints matrices and vector
% Hx <= e so Hj*a <= ej
function [Hj,ej,ineqcon,battInds,newTotalVars,newTotalnG,Tset,Lset,Gj,Cj] = ineqCon(N,M,
    pfpos,pfneg,betaSDAll,etaCAll,etaDAll,maxCapAll,minCapAll,initCapsAll,delPup,delPdo
    G2jidx,Hjidx,H2jidx,H3jidx,H4jidx,H5jidx,H6jidx,numVars,nG,Y,Zij,costs)
% Pinds, Qinds (after limits) from master

    % constraints (currentx4, ampacityx2, voltagex4, powerx4, MCEx4x4)
    Ir_min = limits{1}; Ir_max = limits{2};
    Ii_min = limits{3}; Ii_max = limits{4};
    Vr_min = limits{5}; Vr_max = limits{6};
    Vi_min = limits{7}; Vi_max = limits{8};
    P_min = limits{9}; P_max = limits{10};
    Q_min = limits{11}; Q_max = limits{12};

    % initialize yalmip variables
    Ir=sdpvar(3*N*T,1);
    Ii=sdpvar(3*N*T,1);
    Vr=sdpvar(3*N*T,1);
    Vi=sdpvar(3*N*T,1);
    P=sdpvar(3*N*T,1);
    Q=sdpvar(3*N*T,1);
    a=sdpvar(3*N*T,1);
    b=sdpvar(3*N*T,1);
    c=sdpvar(3*N*T,1);
    d=sdpvar(3*N*T,1);
    Ibranchr=sdpvar(3*M*T,1); % for Z representation
    Ibranchi=sdpvar(3*M*T,1); % for Z representation
    Pdis = sdpvar(length(Sinds_new),1); % discharge power
    Pcharge = sdpvar(length(Sinds_new),1); % charge power

    V=Vr+1i*Vi;
    I=Ir+1i*Ii;
    Ibranch = Ibranchr + 1i*Ibranchi;

    % primary inequality constraints
```

```
    ineqcon = [ % current limits
        Ir_min<=Ir,...
        Ir<=Ir_max,...
        Ii_min<=Ii,...
        Ii<=Ii_max,...
%         % voltage limits
        Vr_min<=Vr,...
        Vr<=Vr_max,...
        Vi_min<=Vi,...
        Vi<=Vi_max,...
%         % power limits
        P_min<=P,...
        P<=P_max,...
        Q_min<=Q,...
        Q<=Q_max,...
        % MCE for a
        a>=Vr_min.*Ir+Vr.*Ir_min-Vr_min.*Ir_min,...
        a>=Vr_max.*Ir+Vr.*Ir_max-Vr_max.*Ir_max,...
        a<=Vr_max.*Ir+Vr.*Ir_min-Vr_max.*Ir_min,...
        a<=Vr_min.*Ir+Vr.*Ir_max-Vr_min.*Ir_max,...
        % MCE for b
        b>=Vi_min.*Ii+Vi.*Ii_min-Vi_min.*Ii_min,...
        b>=Vi_max.*Ii+Vi.*Ii_max-Vi_max.*Ii_max,...
        b<=Vi_max.*Ii+Vi.*Ii_min-Vi_max.*Ii_min,...
        b<=Vi_min.*Ii+Vi.*Ii_max-Vi_min.*Ii_max,...
        % MCE for c
        c>=Vr_min.*Ii+Vr.*Ii_min-Vr_min.*Ii_min,...
        c>=Vr_max.*Ii+Vr.*Ii_max-Vr_max.*Ii_max,...
        c<=Vr_max.*Ii+Vr.*Ii_min-Vr_max.*Ii_min,...
        c<=Vr_min.*Ii+Vr.*Ii_max-Vr_min.*Ii_max,...
        % MCE for dY*v
        d>=Vi_min.*Ir+Vi.*Ir_min-Vi_min.*Ir_min,...
        d>=Vi_max.*Ir+Vi.*Ir_max-Vi_max.*Ir_max,...
        d<=Vi_max.*Ir+Vi.*Ir_min-Vi_max.*Ir_min,...
        d<=Vi_min.*Ir+Vi.*Ir_max-Vi_min.*Ir_max,
        Ibranchr <= 1.5, % dummy bounds
```

```
    Ibranchi <= 1.5 % dummy bounds
    ];

% secondary inequality constraints for pf, already multi-period treated
ineqconDER = [
    sum(P(reshape(Ppfinds,3,[])'),2).* tan(acos(pfneg')) <= sum(Q(reshape(Qpfinds,3,
    sum(Q(reshape(Qpfinds,3,[])'),2) <= sum(P(reshape(Ppfinds,3,[])'),2).* tan(acos
    ];


% storage capacity constraints
battInds = find(~cellfun(@isempty,Sinds_new_sizej)); % nodes which have batteries
numbatts = length(battInds);
n = length(Sinds_new)/T;

Sinds = [Sindsj{:}];  % unpack into a vector
if isa(Sinds,'cell')
    Sinds = [Sinds{:}]; % need to unpack more
end

eqcon = [];
Sidx = 0;
if numbatts > 0
    for i = 1:numbatts
        batt = battInds(i);
        nphases = Sinds_new_sizej{batt};
        Pidx = Sinds(Sidx+(1:nphases*T));
        PSidx = i:numbatts:numbatts*T;
        newEqconEx = (reshape(repmat((1./repmat(etaDAll',T,1).*Pdis(PSidx) - repmat
%         newEqconEx = (reshape(repmat((repmat(-ones(size(etaCAll))',T,1).*Pcharge(
        eqcon = [eqcon;P(Pidx) == newEqconEx];

        Sidx = Sidx + nphases*T;
    end
end
```

```
% storage constraints => 2*numphases + 2 (non-negativity and capacity)
ineqconBattdefn = [
    Pdis >= 0;
%     Pdis == 0; % DEBUGGING
    Pcharge >= 0;
    Pdis <= max([abs(delPup),abs(delPdown)])*ones(length(Sinds_new),1); % add const
    Pcharge <= max([abs(delPup),abs(delPdown)])*ones(length(Sinds_new),1); % add co
%     Pcharge >= -max([abs(delPup),abs(delPdown)])*ones(length(Sinds_new),1); % DEB
    ];

ineqconExps = []; % variable side of capacity expr
ineqconVal = []; % constant side of capacity expr
ineqrampExps = []; % variable side of ramp expr

%   Sind_new_idx = 1;
for i = 1:numbatts
    battID = battInds(i);
    nphases = Sinds_new_sizej{battID};
    PSid = i;
    betaSD = betaSDAll(i);
    etaC = etaCAll(i);
    etaD = etaDAll(i);

    % for capacity
    ineqconExp = etaC*Pcharge(PSid) - 1./etaD*Pdis(PSid);
%     ineqconExp = Pcharge(PSid); % DEBUGGING
    PSidx = i:numbatts:numbatts*T; % indices for tau=2 to tau=T
    ineqconExps = [ineqconExps; ineqconExp]; % for tau=1

    for tau = 2:T
        PSid = PSidx(tau);
%         ineqconExp = ineqconExp*(1-betaSD) + Pcharge(PSid); % DEBUGGING
        ineqconExp = ineqconExp*(1-betaSD) + etaC*Pcharge(PSid) - 1./etaD*Pdis(PSid
        ineqconExps = [ineqconExps; ineqconExp];
    end
```

```
    % ramp constraints for batteries
    Sindsbatt = Sindsj{battID};
    Sindsbatt_idx = 1:nphases;
    idx_new = Sindsbatt(Sindsbatt_idx); % for time period 1
    ineqrampExp = sum(P(idx_new)); % for t = 1 % -Pt0
    ineqrampExps = [ineqrampExp]; % for tau=1
    idx_old = idx_new;
    for tau = 2:T
        Sindsbatt_idx = nphases*(tau-1)+1:nphases*tau;
        idx_new = Sindsbatt(Sindsbatt_idx);
        ineqrampExp = sum(P(idx_new)-P(idx_old));
        ineqrampExps = [ineqrampExps; ineqrampExp];
        idx_old = idx_new;
    end

    ineqconVal(T*(i-1)+1:T*i) = (1-betaSD).^(1:T);
end

% assemble capacity constraints
ineqconBattCap = [ineqconExps <= (maxCapAll - ineqconVal.*initCapsAll)'
    ineqconExps >= (minCapAll - ineqconVal.*initCapsAll)'];

% assemble ramp constraints for batteries
ineqconBattRamp = [ineqrampExps <= delPup'
    ineqrampExps >= delPdown'];

% ramp constraints for generators
PGrampInds = find(~cellfun(@isempty,Prindsj)); % nodes which have DGs with ramp con
numPGr = length(PGrampInds);
ineqrampGExps = [];
for i = 1:numPGr
    pgrampID = PGrampInds(i);
    Prinds = Prindsj{pgrampID};
    Pramp_idx = 1:3;
    idx_new = Prinds(Pramp_idx); % for time period 1
    ineqrampGExp = sum(P(idx_new)); % for t = 1 % -Pt0
```

```
    ineqrampGExps = [ineqrampGExp]; % for tau=1
    idx_old = idx_new;
    for tau = 2:T
        Pramp_idx = 3*(tau-1)+1:3*tau;
        idx_new = Prinds(Pramp_idx);
        ineqrampGExp = sum(P(idx_new)-P(idx_old));
        ineqrampGExps = [ineqrampGExps; ineqrampGExp];
        idx_old = idx_new;
    end
end

% assemble ramp constraints for DGs
ineqconDGRamp = [ineqrampGExps <= delPGup'
    ineqrampGExps >= delPGdown'];

[model,~] = export([ineqcon,ineqconDER,ineqconBattdefn,ineqconBattCap,ineqconBattRa
    0,sdpsettings('solver','Gurobi'));

% Hx <= e
H=model.A;
e=model.rhs;
sense=model.sense;

Hj={}; Hj{N}={};
ej={}; ej{N}={};
DGpfInds = find(~cellfun(@isempty,H2jidx)); % nodes with generators with pf limits

% new equality constraints
Lstart = numVars;
numPdis = length(Sinds_new); % nphases*T for Pdis only
newTotalVars = numVars + numPdis*2; % added Pdis and Pcharge for batt

G2indstart = find(sense=='=',1) - 1; % index starts at 1 for G2jidx
Hindstart = find(sense~='=',1) - 1; % index starts at 1 for Hjidx
numG2cons = cellfun(@max,G2jidx(~cellfun(@isempty,G2jidx))); % get index to repeat
if isempty(numG2cons)
```

```
    numG2cons = 0;
end

newTotalnG = nG + numG2cons;

% new inequality constraints
if ~isempty(DGpfInds) % have P-Q-pf constraints
    numH2cons = max(cellfun(@max,H2jidx(~cellfun(@isempty,H2jidx))));
else
    numH2cons = 0;
end

if numbatts > 0 % have storage constraints
    numH3cons = max(cellfun(@max,H3jidx(~cellfun(@isempty,H3jidx))));
    numH4cons = max(cellfun(@max,H4jidx(~cellfun(@isempty,H4jidx))));
    numH5cons = max(cellfun(@max,H5jidx(~cellfun(@isempty,H5jidx))));
else
    numH3cons = 0;
    numH4cons = 0;
    numH5cons = 0;
end

if numPGr > 0 % battery ramp constraints
    numH6cons = max(cellfun(@max,H6jidx(~cellfun(@isempty,H6jidx))));
else
    numH6cons = 0;
end

H2indstart = max(cellfun(@max,Hjidx)) + Hindstart; % start for P-Q-pf constraints
H3indstart = numH2cons*2 + H2indstart; % start for P_dis >= 0
H4indstart = numH3cons*4*T + H3indstart; % start for batt cap constaints
H5indstart = numH4cons*2 + H4indstart; % start for batt ramp constaints
H6indstart = numH5cons*2 + H5indstart; % start for gen ramp constaints

for j = 1:N
    elems = Hjidx{j} + Hindstart; % primary inequality constraints
```

```
% append the elements for P-Q-pf constraints
if ismember(j,DGpfInds)
    H2idx1 = H2jidx{j};
    H2idx2 = H2idx1 + numH2cons; % already account for time periods
    elems = [elems H2indstart+[H2idx1 H2idx2]];
end

if ismember(j,battInds)
    % storage exists at the node
    nphase = Sinds_new_sizej{j};

    % begin update of Gj by first inserting new cols for owned vars Pdis
    % Pcharge (for T periods)
    oldGj = Gj{j};
    [rowsGj,~] = size(oldGj);
    newGj = [oldGj(:,1:length(Lset{j})) zeros(rowsGj,2*T) oldGj(:,length(Lset{j

    % update Lset
    Lidx1 = Lstart + (1:T); % generate for Pdis
    Lidx2 = Lstart + numPdis + (1:T); % generate for Pcharge
    Lset{j} = [Lset{j}, Lidx1, Lidx2];
    Lstart = Lstart + T; % repetitions of time taken care of separately


    % update Tset
    Tset{j} = [cell2mat(Lset(j)) cell2mat(Oset(j))];

    % append new rows to end of Gj
    Gidx = repmat(G2jidx{j},1,T) + numG2cons*reshape(repmat(0:T-1,length(G2jidx
    newGj(rowsGj+1:rowsGj+nphase*T, 1:length(Tset{j})) = H(G2indstart + Gidx, T:
    Gj{j} = newGj;

    % update muInds
    Cj(j) = Cj(j) + T; % number of rows of Gj owned by j
end
```

```
    % storage...
    if ismember(j,battInds)
        % get elements for Pdis Pcharge nonnegativity
        H3idx1 = repmat(H3jidx{j},1,T) + numH3cons*reshape(repmat(0:T-1,length(H3ji
        H3idx2 = H3idx1 + numH3cons*T;  % Pcharge >= 0
        H3idx3 = H3idx2 + numH3cons*T;  % Pcharge <= max
        H3idx4 = H3idx3 + numH3cons*T;  % Pcharge <= max

        % get elements for capacity constraints
        H4idx1 = H4jidx{j};
        H4idx2 = H4idx1 + numH4cons; % already account for time periods

        % get elements for batt ramp constraints
        H5idx1 = H5jidx{j};
        H5idx2 = H5idx1 + numH5cons; % already account for time periods

        % assemble
        elems = [elems H3indstart+[H3idx1 H3idx2 H3idx3 H3idx4] H4indstart+[H4idx1 I
    end

    if ismember(j,PGrampInds)
        % get elements for batt ramp constraints
        H6idx1 = H6jidx{j};
        H6idx2 = H6idx1 + numH6cons; % already account for time periods
        elems = [elems H6indstart+[H6idx1 H6idx2]];
    end

    % atomic inequality constraints
    Hj{j} = H(elems,Tset{j}); % already augmented with P_dis and P_charge
    ej{j} = e(elems,:);
end



% FOR TESTING CENTRAL MODEL
```

```
    F = [];
    for t = 1:T
        inds = 3*N*(t-1)+1:3*N*t;
        indsLine = 3*M*(t-1)+1:3*M*t;
        F = [F;
            Ir(inds) == real(A'*Ibranch(indsLine));
            Ii(inds) == imag(A'*Ibranch(indsLine));
            P(inds) == a(inds) + b(inds);
            Q(inds) == -c(inds) + d(inds);
            A*V(inds) == Zij*Ibranch(indsLine);
            ];
    end

    ops = sdpsettings('solver','gurobi');
%     [~,globalObjfnc,~] = getObj(N,T,Vr_max,Vi_max);
    [~,globalObjfnc,~] = getObj_economic(N,T,costs,P_max);
%     optimize([F,ineqcon],0,ops); % ,ineqconDER,ineqcon
%     optimize([F,ineqcon],globalObjfnc([Ir;Ii;Vr;Vi;P;Q;a;b;c;d;Ibranchr;Ibranchi]),op
%     yCENTRAL = [value(Ir);value(Ii);value(Vr);value(Vi);value(P);value(Q);value(a);va

    optimize([F,ineqcon,ineqconDER,ineqconBattdefn,ineqconBattCap,ineqconBattRamp,ineqc
    yCENTRAL_storage = [value(Ir);value(Ii);value(Vr);value(Vi);value(P);value(Q);value

end
```

C.5.2 DER Models

rengenmodel.m

```
% Models for renewable generators, returning quantities in p.u., percentage,
% and $

% System Nameplate Capacity              [kW]
% Lower production limit                 [kW]
% Ramp rates (up and down)               [kW/h]
```

```
% Curtail limit                          [%]
% Capacity Factor                        [%]
% Capital cost of system                 [$/kW]
% Installation/ancillary costs          [$/kW]
% Annual O & M costs                     [$/kW/year]
% Replacement cost                       [$/kW]
% Generation profiles, hourly            [kW]

function specs = rengenmodel(rengenID,capacity,T)
    if strcmp(rengenID, "solar_distres_small")
        specs = rengenID_SAM_solar_distres_small(capacity,T);
    elseif strcmp(rengenID, "solar_distres_large")
        specs = rengenID_SAM_solar_distres_large(capacity,T);
    elseif strcmp(rengenID, "solar_distcomm")
        specs = rengenID_SAM_solar_distcomm(capacity,T);
    elseif strcmp(rengenID, "wind_farm")
        specs = rengenID_SAM_wind_farm(capacity,T);
    elseif strcmp(rengenID, "IITChinaSolar")
        specs = rengenID_IITChinaSolar();
    elseif strcmp(rengenID, "IITChinaWind")
        specs = rengenID_IITChinaWind();
    else
        fprintf('\n\t ---------------- ERROR UNKNOWN REN GEN ID\n');
        specs = 0;
    end
end

function [specs,genFlag] = rengenID_SAM_solar_distres_small(capacity,T)
    % distributed residential solar, valid for capacities upto 10kW, 1 inverter
    % capacity in kW

    dataCapacity = 3.85; % initial capacity data of 3.85 kW;

    maxGen = capacity;
    minGen = 0;
    rampUp = 0.8*capacity; % given in kW
```

```
    rampDown = 0.8*capacity; % given in kW
    curtailLimit = 1;
    capacityFactor = 0.2; % https://atb.nrel.gov/electricity/2017/index.html?t=sr

    % financial information
    sysOM = 30; % https://atb.nrel.gov/electricity/2017/index.html?t=sr upwards of $40/
    sysInstall = 6900/dataCapacity; % $6100+800, SAM
    sysCap = 5800/dataCapacity; % capital cost of system, $5800, SAM
    sysReplace = 3000/dataCapacity; % inverter replacements can cost upwards of 2500, t

    genID = 'solar_distres_small'; genFlag = 0;
    [genP,genQ] = generationProfiles(genID,T);

    keySet = {'maxGen', 'minGen', 'rampUp', 'rampDown', 'curtailLimit', 'capacityFactor
    valueSet = {maxGen, minGen, rampUp, rampDown, curtailLimit, capacityFactor, genP.*c
    specs = containers.Map(keySet, valueSet);
end

function [specs,genFlag] = rengenID_SAM_solar_distres_large(capacity,T)
    % distributed residential solar, valid for capacities above 10kW, 3 inverters
    % capacity in kW

    dataCapacity = 11.89; % initial capacity data of 11.89 kW;

    maxGen = capacity;
    minGen = 0;
    rampUp = 0.8*capacity; % given in kW
    rampDown = 0.8*capacity; % given in kW
    curtailLimit = 1;
    capacityFactor = 0.2; % https://atb.nrel.gov/electricity/2017/index.html?t=sr

    % financial information
    sysOM = 30; % https://atb.nrel.gov/electricity/2017/index.html?t=sr upwards of $40/
    sysInstall = 17200/dataCapacity; % $3500+11700+2000, SAM
    sysCap = 14400/dataCapacity; % capital cost of system, $14400, SAM
    sysReplace = 9000/dataCapacity; % inverter replacements can cost upwards of 2500, t
```

```
    genID = 'solar_distres_small'; genFlag = 0;
    [genP,genQ] = generationProfiles(genID,T);

    keySet = {'maxGen', 'minGen', 'rampUp', 'rampDown', 'curtailLimit', 'capacityFactor
    valueSet = {maxGen, minGen, rampUp, rampDown, curtailLimit, capacityFactor, genP.*ca
    specs = containers.Map(keySet, valueSet);
end


function [specs,genFlag] = rengenID_SAM_solar_distcomm(capacity,T)
    % distributed commercial solar
    % capacity in kW

    dataCapacity = 185; % initial capacity data of 185 kW;

    maxGen = capacity;
    minGen = 0;
    rampUp = 0.8*capacity; % given in kW
    rampDown = 0.8*capacity; % given in kW
    curtailLimit = 1;
    capacityFactor = 0.2; % https://atb.nrel.gov/electricity/2017/index.html?t=sr

    % financial information
    sysOM = 30; % https://atb.nrel.gov/electricity/2017/index.html?t=sr upwards of $40/k
    sysInstall = 217000/dataCapacity; % $30200+151100+35700, SAM
    sysCap = 162000/dataCapacity; % capital cost of system, $75500+21500+65000, SAM
    sysReplace = 12000/dataCapacity; % inverter replacements can cost upwards of 2500, t

    genID = 'solar_distcomm'; genFlag = 0;
    [genP,genQ] = generationProfiles(genID,T);

    keySet = {'maxGen', 'minGen', 'rampUp', 'rampDown', 'curtailLimit', 'capacityFactor
    valueSet = {maxGen, minGen, rampUp, rampDown, curtailLimit, capacityFactor, genP.*ca
    specs = containers.Map(keySet, valueSet);
end
```

```
function [specs,genFlag] = rengenID_SAM_wind_farm(capacity,T)
    % commerical wind, farm
    % capacity in kW

    dataCapacity = 600; % kW

    maxGen = capacity;
    minGen = 0;
    rampUp = 0.8*capacity; % given in kW
    rampDown = 0.8*capacity; % given in kW
    curtailLimit = 1;
    capacityFactor = 0.4; % https://atb.nrel.gov/electricity/2017/index.html?t=lw

    % financial information
    sysOM = 51; % https://atb.nrel.gov/electricity/2017/index.html?t=lw, for TRG6 becau
    sysInstall = 0; % capacity + install lumped together
    sysCap = 1640; % https://atb.nrel.gov/electricity/2017/index.html?t=lw, for TRG6 be
    sysReplace = 0; % unknown

    genID = 'wind_farm'; genFlag = 0;
    [genP,genQ] = generationProfiles(genID,T);

    keySet = {'maxGen', 'minGen', 'rampUp', 'rampDown', 'curtailLimit', 'capacityFactor
    valueSet = {maxGen, minGen, rampUp, rampDown, curtailLimit, capacityFactor, genP.*c
    specs = containers.Map(keySet, valueSet);
end

function specs = rengenID_IITChinaSolar()
    capacity = 1000;
    maxGen = capacity;
    minGen = 0;
    rampUp = maxGen*10;
    rampDown = maxGen*10;
    curtailLimit = 1; % 1 = curtail 100% ability
    capacityFactor = 0.2;
```

```
% financial information
sysOM = 100;
sysInstall = 1200;
sysCap = 5000; % capital cost of system
sysReplace = 0;

%     keySet = {'capacity','maxGen', 'minGen', 'rampUp', 'rampDown', 'curtailLimit', 'sy
%     valueSet = {capacity, maxGen*1000, minGen*1000, rampUp*1000, rampDown*1000, curta
%     specs = containers.Map(keySet, valueSet);

genProfile = [
    0
    0
    0
    0
    0
    0
    0.1
    0.5
    0.7
    0.9
    1
    1
    1
    0.89
    0.76
    0.65
    0.23
    0
    0
    0
    0
    0
    0
    0]; % data given in MW, hourly
```

```
    pf = 1;

    keySet = {'maxGen', 'minGen', 'rampUp', 'rampDown', 'curtailLimit', 'capacityFactor
    valueSet = {maxGen, minGen, rampUp, rampDown, curtailLimit, capacityFactor, genProf
    specs = containers.Map(keySet, valueSet);
end

function specs = rengenID_IITChinaWind()
    capacity = 3000;
    maxGen = capacity;
    minGen = 0;
    rampUp = maxGen*10;
    rampDown = maxGen*10;
    curtailLimit = 1; % 1 = curtail 100% ability
    capacityFactor = 0.4;

    % financial information
    sysOM = 100;
    sysInstall = 1200;
    sysCap = 5000; % capital cost of system
    sysReplace = 0;

    genProfile = [
        0.75
        1.5
        1.9
        2.1
        2.2
        1.45
        0
        0
        0.8
        0
        1.5
        0
```

```
        1.4
        0.2
        0
        0.45
        0.4
        1.05
        1.2
        2.5
        2.8
        2.7
        2.75
        2.75]; % data given in MW, hourly

    pf = 1;

    keySet = {'maxGen', 'minGen', 'rampUp', 'rampDown', 'curtailLimit', 'capacityFactor
    valueSet = {maxGen, minGen, rampUp, rampDown, curtailLimit, capacityFactor, genProf
    specs = containers.Map(keySet, valueSet);
end
```

battmodel.m

```
% Generic battery models to use, returning quantities in p.u., percentage,
% and $

% Capacity                              [kWh]
% Rated power                           [kW]
% Maximum depth of discharge            [%]
% Efficiency (charge and discharge)     [%]
% Self discharge rate                   [kW/h/kW]
% Ramp rates (charge and discharge)     [kW/h]
% Number of discharge cycles            [count]
% Capital cost of system                [$/kW]
% Installation/ancillary costs          [$/kW]
% Annual O & M costs                    [$/MWh/year]
% Replacement cost                      [$/kW]
```

```
% Energy throughput                    [kWh]
% Battery wear cost                    [$/kWh]

function specs = battmodel(battID)
    if strcmp(battID, "1")
        specs = battID_1();
    elseif strcmp(battID, "IIT16")
        specs = battID_IIT16();
    elseif strcmp(battID, "IIT17")
        specs = battID_IIT17();
    elseif strcmp(battID, "IITChina")
        specs = battID_IITChina();
    else
        specs = 0;
    end
end

function specs = battID_1()
    capacity = 1000; % in kW % 1000, 500, 2000
    power = 700; % in kW % 700, 300
    maxDOD = 0.85; % depth of discharge

    etaC = 0.95; % charge efficiency
    etaD = 0.95; % discharge efficiency
    betaSD = 0.021; % self discharge rate  % see paper: Optimal Operation of Distributi
%     betaSD = 0; etaC = 1; etaD = 1;
    % in kW
%     rampUp = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Systems
%     rampDown = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Syste

    rampUp = 600; % 600, 300
    rampDown = 600; % 600, 300

    cycleLife = 5000;

    % financial information
```

```
    battOM = 100;
    battInstall = 1200;
    battCap = 5000;
    battReplace = 1000; % $/kW

    energythru = cycleLife*capacity*maxDOD*2; % 80% DOD, 2 trips/cycle
    wearCost = battReplace/(energythru*sqrt(etaC*etaD)); % ref: Homer energy

    keySet = {'capacity', 'power', 'maxDOD', 'etaC', 'etaD', 'betaSD', 'rampUp', 'rampD
    valueSet = {capacity, power, maxDOD, etaC, etaD, betaSD, rampUp, rampDown, cycleLif
    specs = containers.Map(keySet, valueSet);
end


function specs = battID_IIT16()
    capacity = 880;
    power = 330;
    maxDOD = 0.85; % depth of discharge

    etaC = 0.95; % charge efficiency
    etaD = 0.95; % discharge efficiency
    betaSD = 0.021; % self discharge rate  % see paper: Optimal Operation of Distributi

    rampUp = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Systems i
    rampDown = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Systems

    cycleLife = 10000;

    % financial information
    battOM = 100;
    battInstall = 1200;
    battCap = 5000;
    battReplace = 1000;  % $/kW

    energythru = cycleLife*capacity*0.8*2; % 80% DOD, 2 trips/cycle
    wearCost = battReplace/(energythru*sqrt(etaC*etaD)); % ref: Homer energy
```

```
    keySet = {'capacity', 'power', 'maxDOD', 'etaC', 'etaD', 'betaSD', 'rampUp', 'rampD
    valueSet = {capacity, power, maxDOD, etaC, etaD, betaSD, rampUp, rampDown, cycleLif
    specs = containers.Map(keySet, valueSet);
end


function specs = battID_IIT17()
    capacity = 960;
    power = 880;
    maxDOD = 0.85; % depth of discharge

    etaC = 0.95; % charge efficiency
    etaD = 0.95; % discharge efficiency
    betaSD = 0.021; % self discharge rate  % see paper: Optimal Operation of Distributi

    rampUp = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Systems i
    rampDown = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Systems

    cycleLife = 10000;

    % financial information
    battOM = 100;
    battInstall = 1200;
    battCap = 5000;
    battReplace = 1000; % $/kW

    energythru = cycleLife*capacity*0.8*2; % 80% DOD, 2 trips/cycle
    wearCost = battReplace/(energythru*sqrt(etaC*etaD)); % ref: Homer energy

    keySet = {'capacity', 'power', 'maxDOD', 'etaC', 'etaD', 'betaSD', 'rampUp', 'rampD
    valueSet = {capacity, power, maxDOD, etaC, etaD, betaSD, rampUp, rampDown, cycleLif
    specs = containers.Map(keySet, valueSet);

end
```

```
function specs = battID_IITChina()
    capacity = 3000;
    power = 500;
    maxDOD = 0.9; % depth of discharge

    etaC = 0.95; % charge efficiency
    etaD = 0.95; % discharge efficiency
%     betaSD = 0.021; % self discharge rate  % see paper: Optimal Operation of Distribut
    betaSD = 0; % VRFlow batts

    rampUp = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Systems i
    rampDown = 300; % see paper: Optimal Allocation of Dispersed Energy Storage Systems

    cycleLife = 15000; % VRFlow batts

    % financial information
    battOM = 100;
    battInstall = 0;
    battCap = 900; % $/kWh
    battReplacePerc = 0.15; % replacement cost is 15% of original cost
    battReplace = battReplacePerc*battCap; % $/kWh

    energythru = cycleLife*capacity*0.9*2; % 90% DOD, 2 trips/cycle
    wearCost = battReplace*capacity/(energythru*sqrt(etaC*etaD)); % ref: Homer energy, 

    keySet = {'capacity', 'power', 'maxDOD', 'etaC', 'etaD', 'betaSD', 'rampUp', 'rampD
    valueSet = {capacity, power, maxDOD, etaC, etaD, betaSD, rampUp, rampDown, cycleLif
    specs = containers.Map(keySet, valueSet);
end
```

C.5.3 Objective Functions

getObj.m

```
function [objfncs,globalObjfnc,L] = getObj(N,T,Vr_max,Vi_max)
```

```
    objfncs={}; objfncs{N}={};
    for j = 1:N
        VrInds = 3*2*T + (1:3*T); % Vr is 3rd variable,  3*(j-1)

        ViInds = 3*3*T + (1:3*T); % Vi is 4th variable,  3*(j-1)
        % volt-var control: (Vr-1)^2 + (Vi-0)^2
        objfncs{j} = @(aj,j_,len) sum((aj(VrInds)-ones(3*T,1)).^2 + (aj(ViInds)-zeros(3*
    end

    globalObjfnc = globalObj(N,T);
    L = getObjL(N,T,Vr_max,Vi_max);
end

% yOrder: [Ir Ii Vr Vi P Q a b c d] (global) CIY
% aOrder: [Ir Ii Vr Vi P Q a b c d | Vr_tilde Vi_tilde] (atomic) CIY
% yOrder: [Ir Ii Vr Vi P Q a b c d {Ibranchr Ibranchi}_down Pdis Pcharge] (global) CIZ
% aOrder: [Ir Ii Vr Vi P Q a b c d {Ibranchr Ibranchi}_down Pdis Pcharge | {Vr_tilde Vi_
% y multi-period: [{{Ir}_N}t1 ... {{Ir}_N}tT , {{Ii}_N}t1 ... {{Ii}_N}tT .... ]

function val = globalObj(N,T)
    VrInds = 3*2*N*T + (1:3*N*T);
    ViInds = 3*3*N*T + (1:3*N*T);
    % volt-var control: (Vr-1)^2 + (Vi-0)^2
    val = @(y) sum((y(VrInds)-ones(3*N*T,1)).^2 + (y(ViInds)-zeros(3*N*T,1)).^2);
end

% L should be at least as large as the derivative
function L = getObjL(N,T,Vr_max,Vi_max)
    % (Vr^2 - 2*Vr + 1) + Vi^2 ---> d/dy = 2Vr + 2Vi - 2(size of Vr)
    L = sum((2*Vr_max + 2*Vi_max)) - 2*3*N*T;
end
```

getObj_economic.m

```
function [objfncs,globalObjfnc,L] = getObj_economic(N,T,costs,P_max)
    objfncs={}; objfncs{N}={};
```

```
    cost_aP = costs{1};
    cost_bP = costs{2};
    cost_cP = costs{3};
    cost_bPs = costs{4};
    cost_bQ = costs{5};

    for j = 1:N
        PInds = 3*4*T + (1:3*T); % P is 5th variable
        QInds = 3*5*T + (1:3*T); % Q is 6th variable

        costIdx = ((j-1)*3 + 1):((j-1)*3 + 3);
        costIdxrep = repmat(costIdx,1,T)+3*N*reshape(repmat((0:T-1),3,1),1,[]);
        objfncs{j} = @(aj,j_,len) cost_aP(costIdxrep)'*aj(PInds).^2 + cost_bP(costIdxrep
            sum(cost_cP(costIdxrep)) + abs(cost_bPs(costIdxrep)'*aj(PInds)) + cost_bQ(co
    end

    globalObjfnc = globalObj(N,T,costs);

    L = getObjL(N,T,costs,P_max);
end

% yOrder: [Ir Ii Vr Vi P Q a b c d] (global) CIY
% aOrder: [Ir Ii Vr Vi P Q a b c d | Vr_tilde Vi_tilde] (atomic) CIY
% yOrder: [Ir Ii Vr Vi P Q a b c d {Ibranchr Ibranchi}_down Pdis Pcharge] (global) CIZ
% aOrder: [Ir Ii Vr Vi P Q a b c d {Ibranchr Ibranchi}_down Pdis Pcharge | {Vr_tilde Vi_
% y multi-period: [{{Ir}_N}t1 ... {{Ir}_N}tT , {{Ii}_N}t1 ... {{Ii}_N}tT .... ]

function val = globalObj(N,T,costs)
    cost_aP = costs{1};
    cost_bP = costs{2};
    cost_cP = costs{3};
    cost_bPs = costs{4};
    cost_bQ = costs{5};

    PInds = 3*4*N*T + (1:3*N*T);
    QInds = 3*5*N*T + (1:3*N*T);
```

```
    val = @(y) cost_aP'*y(PInds).^2 + cost_bP'*y(PInds) + sum(cost_cP) + abs(cost_bPs'*y
end

% L should be at least as large as the derivative
function L = getObjL(N,T,costs,P_max)
    % (Vr^2 - 2*Vr + 1) + Vi^2 ---> d/dy = 2Vr + 2Vi - 2(size of Vr)
    cost_aP = costs{1};
    cost_bP = costs{2};
    cost_cP = costs{3};
    cost_bPs = costs{4};
    cost_bQ = costs{5};

    L = 2.*cost_aP'*P_max + sum(cost_bP + cost_bPs + cost_bQ);
end
```

getObj_socialwelfare.m

```
function [objfncs,globalObjfnc,L] = getObj_socialwelfare(N,M,T,xi,costs,Zij,RMatj,numCh
    objfncs={}; objfncs{N}={};

    cost_aP = costs{1};
    cost_bP = costs{2};
    cost_cP = costs{3};
    cost_bPs = costs{4};
    cost_bQ = costs{5};

    % node owns downstream power flows

    % electrical losses: R*|Ir + jIi|^2 = R*(Ir^2 + Ii^2)
    % cost of electricity from feeder: LMP_P*P(feeder) + LMP_Q*Q(feeder)
    % assume all phases have same LMP

    % electrical losses and costs
    for j = 1:N
        numChildj = numChildren(j);
```

```
        if numChildj > 0
            IbranchrInds = 3*10*T + (1:3*T*numChildj);
            IbranchiInds = (3*10 + 3*numChildj)*T + (1:3*T*numChildj);

            objfncs{j} = @(aj,j_,len) xi*sum(RMatj{j}*(aj(IbranchrInds).^2 + aj(Ibranch:
        else
            % no children node, no downstream power flows
            objfncs{j} = @(aj,j_,len) 0;
        end

        PInds = 3*4*T + (1:3*T); % P is 5th variable
        QInds = 3*5*T + (1:3*T); % Q is 6th variable
        idx = (j-1)*3 + (1:3);
        idxT = repmat(idx,1,T) + reshape(repmat((0:T-1),3,1),1,[])*3*N;

        objfncs{1} = @(aj,j_,len) objfncs{1}(aj,j_,len) + ...
            cost_aP(idxT)'*aj(PInds).^2 + cost_bP(idxT)'*aj(PInds) + sum(cost_cP(idxT))
    end

    RMat = real(Zij);
    globalObjfnc = globalObj(N,M,T,xi,costs,RMat);
    L = getObjL(N,T,xi,costs,RMat,Ibranchr_max,Ibranchi_max,P_max);
end

% yOrder: [Ir Ii Vr Vi P Q a b c d] (global) CIY
% aOrder: [Ir Ii Vr Vi P Q a b c d | Vr_tilde Vi_tilde] (atomic) CIY
% yOrder: [Ir Ii Vr Vi P Q a b c d {Ibranchr Ibranchi}_down Pdis Pcharge] (global) CIZ
% aOrder: [Ir Ii Vr Vi P Q a b c d {Ibranchr Ibranchi}_down Pdis Pcharge | {Vr_tilde Vi_
% y multi-period: [{{Ir}_N}t1 ... {{Ir}_N}tT , {{Ii}_N}t1 ... {{Ii}_N}tT .... ]

function val = globalObj(N,M,T,xi,costs,RMat)
    cost_aP = costs{1};
    cost_bP = costs{2};
    cost_cP = costs{3};
    cost_bPs = costs{4};
```

```
    cost_bQ = costs{5};

    IbranchrInds = 3*10*N*T + (1:3*M*T);
    IbranchiInds = (3*10*N + 3*M)*T + (1:3*M*T);

    PInds = 3*4*N*T + (1:3*N*T);
    QInds = 3*5*N*T + (1:3*N*T);

    RMatT = repmat({RMat},1,T);

    % volt-var control: (Vr-1)^2 + (Vi-0)^2
    % electrical losses: R*|Ir + jIi|^2 = R*(Ir^2 + Ii^2)
    % cost of electricity from feeder: LMP_P*P(feeder) + LMP_Q*Q(feeder)
    % assume all phases have same LMP
    val = @(y) xi*sum(blkdiag(RMatT{:})*(y(IbranchrInds).^2 + y(IbranchiInds).^2)) + ..
        cost_aP'*y(PInds).^2 + cost_bP'*y(PInds) + sum(cost_cP) + abs(cost_bPs'*y(PInds)

end

% L should be at least as large as the derivative
 function L = getObjL(N,T,xi,costs,RMat,Ibranchr_max,Ibranchi_max,P_max)
    % (Vr^2 - 2*Vr + 1) + Vi^2 ---> d/dy = 2Vr + 2Vi - 2(size of Vr)
    % electrical losses: R*|Ir + jIi|^2 = R*(Ir^2 + Ii^2)
    % ---> d/dy = R*(2Ibranchr + 2Ibranchi)
    % cost of electricity from feeder: LMP_P*P(feeder) + LMP_Q*Q(feeder)
    % ---> d/dy = LMP_P(size of Pfeed) + LMP_Q(size of Qfeed)

    cost_aP = costs{1};
    cost_bP = costs{2};
    cost_cP = costs{3};
    cost_bPs = costs{4};
    cost_bQ = costs{5};

    RMatT = repmat({RMat},1,T);

    L = xi*sum(blkdiag(RMatT{:})*(Ibranchr_max + Ibranchi_max)*2) + 2.*cost_aP'*P_max +
```

```
end
```

C.6 Result Processing

generateResGraphs.m

```
function generateResGraphs( y, N, M, T, rengenInds, battInds,Sindsj,Sinds_new, limits)
    % eventually include some electricity cost....
    % Include ramp rates on limits for generators

    IrIndsAll = (1:3*N*T); % Ir is 1st set of vars
    IiIndsAll = 3*1*N*T + (1:3*N*T); % Ii is 2nd set of vars
    VrIndsAll = 3*2*N*T + (1:3*N*T); % Vr is 3rd set of vars
    ViIndsAll = 3*3*N*T + (1:3*N*T); % Vi is 4th set of vars
    PIndsAll = 3*4*N*T + (1:3*N*T); % P is 5th set of vars
    QIndsAll = 3*5*N*T + (1:3*N*T); % Q is 6th set of vars
    MCEaIndsAll = 3*6*N*T + (1:3*N*T); % a is 7th set of vars
    MCEbIndsAll = 3*7*N*T + (1:3*N*T); % b is 8th set of vars
    MCEcIndsAll = 3*8*N*T + (1:3*N*T); % c is 9th set of vars
    MCEdIndsAll = 3*9*N*T + (1:3*N*T); % d is 10th set of vars


    Ir_min = limits{1}; Ir_max = limits{2};
    Ii_min = limits{3}; Ii_max = limits{4};
    Vr_min = limits{5}; Vr_max = limits{6};
    Vi_min = limits{7}; Vi_max = limits{8};
    P_min = limits{9}; P_max = limits{10};
    Q_min = limits{11}; Q_max = limits{12};

    time = (1:T);
    periodend = T;
    plotGenNLoad(N, T, y(PIndsAll), y(QIndsAll), time, rengenInds, battInds, Sindsj, P_I

end
```

```
function plotGenNLoad(N, T, P, Q, time, rengenInds, battInds, Sindsj, P_min)
% N, pG, qG, qL, pL, sNet, demand, time, genNodes, rengenNodes, battNodes)

    % GENERATION
    % get production from feeder
    feedinds = [1:3*N:3*N*T];
    repPhase1 = reshape(repmat(feedinds,3,1),1,[]);
    repPhase2 = reshape(repmat((0:2),T,1)',1,[]);
    feedindsrep = repPhase1 + repPhase2;
    feederP = sum(reshape(P(feedindsrep),[],T));
    feederQ = sum(reshape(Q(feedindsrep),[],T));

    % get production by rengens
    if ~isempty(rengenInds)
        repPhase1 = reshape(repmat(3*(rengenInds-1),3,1),[],1);
        repPhase2 = reshape(repmat((1:3),size(rengenInds,2),1)',1,[])';
        repT1 = repmat(repPhase1 + repPhase2,T,1);
        repT2 = reshape(repmat(3*N*(0:T-1),size(repPhase1,1),1),[],1);
        rengensP = sum(reshape(P(repT1 + repT2),[],T));
        rengensQ = sum(reshape(Q(repT1 + repT2),[],T));
    else
        rengensP = zeros(1,T);
        rengensQ = zeros(1,T);
    end

%     % all other gens
%     gensP = sum(pG(genNodes(2:end),:),1);

    % STORAGE
    % get action of storage across entire network (net storage)
    if ~isempty(battInds)
        indsStorage = [Sindsj{:}];

        if isa(indsStorage,'cell')
            indsStorage = [indsStorage{:}]; % need to unpack more
```

```
    end

    indsStorageT = reshape(sort(indsStorage),[],T); % each column is the indices fo
    netStorageP = sum(P(indsStorageT));
    netStorageQ = sum(Q(indsStorageT));
else
    netStorageP = zeros(1,T);
    netStorageQ = zeros(1,T);
end

% LOADS
notStorageNotGen = setdiff((2:N),[battInds,rengenInds]);
repPhase1 = reshape(repmat(3*(notStorageNotGen-1),3,1),[],1);
repPhase2 = reshape(repmat((1:3),size(notStorageNotGen,2),1)',1,[])';
repT1 = repmat(repPhase1 + repPhase2,T,1);
repT2 = reshape(repmat(3*N*(0:T-1),size(repPhase1,1),1),[],1);
loadInds = repT1 + repT2;

P_minT = reshape(P_min(loadInds),[],T); % each column is for time period t (T colum
demandP = sum(P_minT.*(P_minT<0)); % loads are negative; get net loads per period t

P_load = P(loadInds);
P_loadT = reshape(P_load,[],T); % each column is for time period t (T columns)
loadsServicedP = sum(P_loadT);

Q_load = Q(loadInds);
Q_loadT = reshape(Q_load,[],T); % each column is for time period t (T columns)
loadsServicedQ = sum(Q_loadT);

loadDR = abs(demandP) - abs(loadsServicedP); % amt of demand response, total across
loadDRpercent = (abs(demandP) - abs(loadsServicedP))./abs(demandP)*100;

figure, hold on;
subplot(5,1,1)
if size(time,2)==1
    % hack to allow for single stacked bar
```

```
    bar([1;nan], [[feederP;netStorageP.*(netStorageP>0);rengensP]'; nan(1,3)], 'sta
else
    bar([feederP;netStorageP.*(netStorageP>0);rengensP]','stacked');
end
xticks((time(1):1:time(end)));
xlabel('Time Period (hr)');
ylabel('Real power in pu');
legend('Imported Electricity','Storage Generation (Discharge)','Renewable Generatio
%   legend('Imported Electricity','Nonrenewable Generation','Renewable Generation','St

subplot(5,1,2)
if size(time,2)==1
    % hack to allow for single stacked bar
    bar([1;nan], [[feederQ;netStorageQ.*(netStorageQ>0);rengensQ]'; nan(1,3)], 'sta
else
    bar([feederQ;netStorageQ.*(netStorageQ>0);rengensQ]','stacked');
end
xticks((time(1):1:time(end)));
xlabel('Time Period (hr)');
ylabel('Reactive power in pu');
legend('Imported Electricity','Storage Generation (Discharge)','Renewable Generatio
%   legend('Imported Electricity','Nonrenewable Generation','Renewable Generation');

subplot(5,1,3)
if size(time,2)==1
    % hack to allow for single stacked bar
    bar([1;nan], [[loadsServicedP;netStorageP.*(netStorageP<0)]'; nan(1,2)], 'stack
else
    bar([loadsServicedP;netStorageP.*(netStorageP<0)]', 'stacked');
end
xticks((time(1):1:time(end)));
xlabel('Time Period (hr)');
ylabel('Real power in pu');
legend('System Load','Storage Load (Charging)');

subplot(5,1,4)
```

```
    if size(time,2)==1
        % hack to allow for single stacked bar
        bar([1;nan], [[loadsServicedQ;netStorageQ.*(netStorageQ<0)]'; nan(1,2)], 'stacke
    else
        bar([loadsServicedQ;netStorageQ.*(netStorageQ<0)]', 'stacked');
    end
    xticks((time(1):1:time(end)));
    xlabel('Time Period (hr)');
    ylabel('Reactive Power in pu');
    legend('System Load','Storage Load (Charging)');

    subplot(5,1,5)
    if size(time,2)==1
        % hack to allow for single stacked bar
        scatter(time,loadDRpercent)
    else
        scatter(time,loadDRpercent)
    end
    xlim([0,time(end)]);
    xticks((time(1):1:time(end)));
    xlabel('Time Period (hr)');
    ylabel('Percentage of Forecasted Load');
    legend('Load Curtailed through DR');
end
```

C.7 Network Data and Data Profiles

All data files are not shown for brevity. Only the data files for the 3 node network are shown for completeness of understanding code structure and dependencies. **Network3NodeModel.m**

```
%% GRID NETWORK
% Details of network configuration and topology
% Constant and do not change
```

```
% Define base values
SBase=5e3; % kVA
VBase=4160; % V
ZBase=VBase^2/(SBase)/1e3; % Ohm
YBase = 1/ZBase; % Siemens
IBase=SBase*1e3/(sqrt(3)*VBase);

% Grid topology - set neighbours of each node
N = 3;

arcs = [
    1   2   0.0568  1
    2   3   0.0568  2
];

M = size(arcs,1);

% Grid topology - set neighbours of each node
Graph_network=graph(arcs(:,1),arcs(:,2));
adj=full(adjacency(Graph_network));
[i,j]=find(adj);
Neigh = accumarray(i,j,[size(adj,1), 1],@(x) {sort(x).'});
parents = cellfun(@(neighbours,node)neighbours(neighbours<node),Neigh,num2cell((1:N)'),
children = cellfun(@(neighbours,node)neighbours(neighbours>node),Neigh,num2cell((1:N)'),
numParents = cellfun(@numel,parents);
numChildren = cellfun(@numel,children);

% PCC transformer rating
Stran = 5e3/SBase; % kVA

% admittances of line configs
c_601 = [0.4338 - 1.2502i  -0.1840 + 0.4622i  -0.1008 + 0.3455i
    -0.1840 + 0.4622i   0.3798 - 1.1847i  -0.0478 + 0.2639i
    -0.1008 + 0.3455i  -0.0478 + 0.2639i   0.3359 - 1.1176i];
```

```
c_602 = [0.5270 - 0.6756i  -0.1416 + 0.1426i  -0.2026 + 0.1600i
    -0.1416 + 0.1426i   0.4648 - 0.6689i  -0.1030 + 0.1309i
    -0.2026 + 0.1600i  -0.1030 + 0.1309i   0.4914 - 0.6710i];

admittances = {c_601,c_602};

%     % HIGHER R
% z_601 = [ 0.9+1.0179i 0.45+0.5017i    0.45+0.4236i
%           0.45+0.5017i    0.99+1.0478i    0.45+0.3849i
%           0.45+0.4236i    0.45+0.3849i  0.99+1.0348i];
% z_602 = [2.1+1.1814i     0.45+0.4236i 0.45+0.5017i
%          0.45+0.4236i     2.1+1.1983i 0.45+0.3849i
%          0.45+0.5017i     0.45+0.3849i  2.1+1.2112i];

     % ORIGINAL
z_601 = [ 0.3465+1.0179i    0.1560+0.5017i  0.1580+0.4236i
          0.1560+0.5017i    0.3375+1.0478i  0.1535+0.3849i
          0.1580+0.4236i    0.1535+0.3849i  0.3414+1.0348i];
z_602 = [0.7526+1.1814i     0.1580+0.4236i  0.1560+0.5017i
         0.1580+0.4236i     0.7475+1.1983i  0.1535+0.3849i
         0.1560+0.5017i     0.1535+0.3849i  0.7436+1.2112i];

impedances = {z_601,z_602};
```

Network3NodeData.m

```
%% NETWORK SPECIFIC
% 3 NODE NETWORK

classdef Network3NodeData
    methods(Static)
        % Select the desired setup
        function [Hj,ej,ineqcon,limits,costs,rengenInds,battInds,Sindsj,Sinds_new,numTot
            V_min = 0.95; V_max = 1.05;
            delta_min = -0.01; delta_max = 0.01;
            switch configCode
```

```
                case 1
                    [loadsP,loadsQ,demRes,gensP,gensQ,rengens,storage,demandProfileID] =
                case 2
                    [loadsP,loadsQ,demRes,gensP,gensQ,rengens,storage,demandProfileID] =
                case 3
                    [loadsP,loadsQ,demRes,gensP,gensQ,rengens,storage,demandProfileID] =
                otherwise % default
                    fprintf('\n\t ------------------- Default Network Configuration Used\n
                    [loadsP,loadsQ,demRes,gensP,gensQ,rengens,storage,demandProfileID] =
            end

            % run pre-processing and assemble all the bounds - general function
            [Hj,ej,ineqcon,limits,costs,rengenInds,battInds,Sindsj,Sinds_new,numTotalVa
                V_min,V_max,delta_min,delta_max,N,M,A,Tset,Lset,Oset,Gj,Cj,nG,Hjidx,SBa
        end
    end
end


function [loadsP,loadsQ,demRes,gensP,gensQ,rengens,storage,demandProfileID] = configCod
    demandProfileID = 'ISONE';

    % [node phaseA phaseB phaseC]  for P and Q respectively
    loadsP = [
        2   200    300    500
        3   200    500    100]; % [node load]

    loadsQ = [
        2   100 150 300
        3   80 350 20]; % [node load]

    % demand response capabilities, percentage of DR enabled
    % DR will apply to loads "equally" on ALL phases
    demRes = [];
    % [node maxGenA maxGenB maxGenC minGenA minGenB minGenC pfpos pfneg rampup rampdown
    gensP = [];
```

```
    % [node maxGenA maxGenB maxGenC minGenA minGenB minGenC]
    gensQ = [];
    % {{node, capacitykW, phases, rengenID},{...}}
    rengens = {};
    % [node battID phases initCharge initPstate] --phases 'ab' or 'abc'
    % note: for 2 phase storage, initPstate = [x y 0], regardless of which phases are c
    storage = [];

end

function [loadsP,loadsQ,demRes,gensP,gensQ,rengens,storage,demandProfileID] = configCod
    demandProfileID = 'ISONE';

    loadsP = [
        2   200    300    500]; % [node load]

    loadsQ = [
        2   100 150 300]; % [node load]

    demRes = []; % demand response capabilities

    % [node maxGenA maxGenB maxGenC minGenA minGenB minGenC pfpos pfneg rampup rampdown
    gensP = [];
%     gensP = [3   100 100 100 0   0   0   0.95    -0.95 60 45];

    gensQ = [];

    %{{node, capacitykW, phases, rengenID},{...}}
    rengens = {};
    %     rengens = {{3,10,'ac','solar_distres_small'}};

    % [node battID phases initCharge initPstate] --phases 'ab' or 'abc'
    % note: for 2 phase storage, initPstate = [x y 0], regardless of which phases are c
    storage = [3 "1" 'abc' 150 0 0 0];
%     storage = []; % 500 0 0 0
```

```
end

function [loadsP,loadsQ,demRes,gensP,gensQ,rengens,storage,demandProfileID] = configCod
    demandProfileID = 'ISONE';

    loadsP = [
        2   200    300    500
        3   200    500    100]; % [node load]

    loadsQ = [
        2   100 150 300
        3   80 350 20]; % [node load]

    demRes = [
        2   0.7];

    gensP = [];
    gensQ = [];
    rengens = {};
    storage = [];

end
```

generationProfiles.m

```
function [mm_gP,mm_gQ] = generationProfiles(generationID,T)
    % mm_gP - normalized generation profile for P, for T hours
    % mm_gQ - normalized generation profile for Q, for T hours
    pathname = './networkdata/dataprofiles/';

    switch generationID
        case 'solar_distres_small'
            pf = 1; % fixed power factor for the inverter
            DCtoAC = 1.2; % fixed DC to AC conversion ratio
            filename = 'PV_distres_pheonixAZ_3850W.csv';
```

```
        startDay = 'May 14, 12:00 am';
    case 'solar_distres_large'
        pf = 1; % fixed power factor for the inverter
        DCtoAC = 1.2; % fixed DC to AC conversion ratio
        filename = 'PV_distres_pheonixAZ_11893W.csv';
        startDay = 'May 14, 12:00 am';
    case 'solar_distcomm'
        pf = 1; % fixed power factor for the inverter
        DCtoAC = 1.2; % fixed DC to AC conversion ratio
        filename = 'PV_distcomm_pheonixAZ_185kW.csv';
        startDay = 'May 14, 12:00 am';
    case 'wind_farm'
        pf = 1; % fixed power factor for the inverter
        DCtoAC = 1.2; % fixed DC to AC conversion ratio
        filename = 'Wind_farm_600kW.csv';
        startDay = '05/14/2018 00:00'; % % May 14
end

if isfile(strcat(pathname,filename))
    genProfile = readtable(strcat(pathname,filename));
else
    fprintf('\n\t ---------- ERROR GEN PROFILE FILE NOT FOUND\n');
    return;
end

if ~strcmp(generationID,'wind_farm')
    startInd = find(strcmp(genProfile.TimeStamp, startDay));
else
    startInd = find(genProfile.TimeStamp == datetime(startDay));
end

if ~isempty(startInd)
    if startInd+T-1 > size(genProfile,1)
        fprintf('\n\t ---------- ERROR GEN TIME EXCEEDS FORECAST\n');
        return;
    end
```

```
        mm_gPT = table2array(genProfile(startInd:startInd+T-1,2))/DCtoAC;
        mm_gQT = mm_gPT.*tan(acos(pf));
    else
        fprintf('\n\t ---------------- WARNING GEN TIME NOT FOUND\n');
        mm_gPT = zeros(T,1);
        mm_gQT = zeros(T,1);
    end

    mm_gP = rescale(mm_gPT);
    mm_gQ = rescale(mm_gQT);
end
```

demandProfiles.m

```
function [mm_dP,mm_dQ] = demandProfiles(demandID,N,T)
    % mm_dP - demand profile for P
    % mm_dQ - demand profile for Q
    % N is number of nodes in network (so each node gets a unique profile)

   nominalP = ones(N,1); nominalQ = ones(N,1); % temp, (may not need)
    switch demandID
        case 'ISONE'
            [mm_dP,mm_dQ] = ISONE(N,T);
        case 'normal'
            [mm_dP,mm_dQ] = normalD(N,T,nominalP,nominalQ);
        case 'normal_pf'
            [mm_dP,mm_dQ] = normalD_pf(N,T,nominalP,nominalQ);
        case 'random'
            [mm_dP,mm_dQ] = randomD(N,T);
    end
end

function [mm_dPT,mm_dQT] = ISONE(N,T)

    dateid = 'May142019';
    pathname = strcat('./networkdata/dataprofiles/',dateid,'/');
```

```
    filename = strcat('isonedata_',dateid,'.mat');

    if isfile(strcat(pathname,filename))
        load(strcat(pathname,filename),'mm_d','units')
    else
        fprintf('\n\t ---------------- ERROR DEMAND PROFILE FILE NOT FOUND\n');
        return;
    end

    % assume all profile data is needed per hour
    % profile data is every 5 mins from ISONE -> convert to hourly
    n = 12;

    mm_d_reshaped = reshape(mm_d,n,[]);
    mm_d_hourly = mean(mm_d_reshaped);
    mm_d_hourly = repmat(mm_d_hourly,N,1); % duplicate for all nodes

    % perturb demand profile data to generate variation between nodes
    sigmaP = 0.1;
    rP = normrnd(0,sigmaP,[N,24]); % need N by T variations
    mm_dP_dup = mm_d_hourly.*(1+rP); % N by T
    % smooth the data to give a more realistic demand profile
    mm_dP = smoothdata(smoothdata(smoothdata(mm_dP_dup',1,'gaussian'),2),'gaussian')';

    % generate some variability so P and Q don't follow identical profiles
    sigmaQ = 0.01;
    rQ = normrnd(0,sigmaQ,[N,24]); % need N by T variations
    mm_dQ = mm_dP.*(1+rQ); % N by T

    % adjust to only return the desired hours
    mm_dPT = mm_dP(:,1:T);
    mm_dQT = mm_dQ(:,1:T);
end

function [mm_dP,mm_dQ] = normalD(N,T,nominalP,nominalQ)
    % random, normally distributed demand profiles
```

```
    mm_dP = round(abs(normrnd(1,0.3,[N*T,1])),2); % more variation in P?
    mm_dQ = round(abs(normrnd(1,0.1,[N*T,1])),2); % less variation in Q?

    % check any 0's in the profile, set to a nominal value
    % can't have any negative values
    nominalP = repmat(nominalP,T,1);
    nominalQ = repmat(nominalQ,T,1);
    mm_dP(mm_dP <= 0) = nominalP(mm_dP <= 0);
    mm_dQ(mm_dQ <= 0) = nominalQ(mm_dQ <= 0);
end

function [mm_dP,mm_dQ] = normalD_pf(N,T,nominalP,nominalQ)
    % random, normally distributed demand profiles
    mm_dP = round(abs(normrnd(1,0.3,[N*T,1])),2); % more variation in P?
    mm_pf = round(abs(normrnd(0.55,0.05,[N*T,1])),2); % use pf
    mm_dQ = mm_dP.*(1-mm_pf)./mm_pf;

    % check any 0's in the profile, set to a nominal value
    % can't have any negative values
    nominalP = repmat(nominalP,T,1);
    nominalQ = repmat(nominalQ,T,1);
    mm_dP(mm_dP <= 0) = nominalP(mm_dP <= 0);
    mm_dQ(mm_dQ <= 0) = nominalQ(mm_dQ <= 0);
end

function [mm_dP,mm_dQ] = randomD(N,T)
    % truly random demand profiles
    mm_dP = rand(N,T); % more variation in P?
    mm_dQ = rand(N,T); % less variation in Q? or should we set as a pf range of the P l

end
```

Bibliography

[1] Jeff St. John. Distributed energy poised for âĂŸexplosive growthâĂŹ on the us grid. June 2018.

[2] US Environmental Protection Agency (EPA). U.s. electricity grid markets. https://www.epa.gov/greenpower/us-electricity-grid-markets, 2017. Accessed: 2020-01-06.

[3] PJM. Market for electricity. https://learn.pjm.com/electricity-basics/market-for-electricity.aspx, 2018. Accessed: 2020-01-06.

[4] PJM Demand Side Response Operations. 2018 distributed energy resources (der) that participate in pjm markets as demand response. https://www.pjm.com/-/media/markets-ops/demand-response/2018-der-annual-report.ashx?la=en, 2019. Accessed: 2020-01-06.

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

[6] M. Farivar and S. Low. Branch flow model: Relaxations and convexification part ii. *IEEE Transactions on Power Systems*, 28(3):2565–2572, 2013.

[7] L. Gan, N. Li, U. Topcu, and S. H. Low. Exact convex relaxation of optimal power flow in radial networks. *IEEE Transactions on Automatic Control*, 60(1):72–87, Jan 2015.

[8] G. Ferro, M. Robba, D. D'Achiardi, R. Haider, and A.M. Annaswamy. A distributed approach to the optimal power flow problem for unbalanced and mesh networks. In *In 21st IFAC World Congress, 2020*, 2019 (submitted).

[9] F.C. Schweppe, M.C. Caramanis, R.D. Tabors, and R.E Bohn. *Spot Pricing of Electricity*. Kluwer Academic Publishers, 1998.

[10] Pu Huang, Jayant Kalagnanam, Ramesh Natarajan, Don Hammerstrom, Ron Melton, Mayank Sharma, and Ron Ambrosio. Analytics and transactive control design for the pacific northwest smart grid demonstration project. *First IEEE International Conference on Smart Grid Communications*, Oct 2010.

[11] B. Stott, J. Jardim, and O. Alsac. Dc power flow revisited. *IEEE Transactions on Power Systems*, 24(3):1290–1300, 2009.

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

[13] M. Baran and F.Wu. Optimal capacitor placement on radial distribution systems. *IEEE Transactions on Power Delivery*, 4(1):725–734, 1989.

[14] A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automated Control*, 54(1):48–61, 2009.

[15] A. Nedic, A. Ozdaglar, and P. A. Parrilo. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automated Control*, 55(4):922–938, 2010.

[16] M. Gurbuzbalaban, A. Ozdaglar, and P. A. Parrilo. Convergence rate of incremental gradient and newton methods. *arXiv preprint:1510.08562*, pages 922–938, 2015.

[17] M. Gurbuzbalaban, A. Ozdaglar, and P. A. Parrilo. A globally convergent incremental newton method. *Mathematical Programming*, 151(1):283–313, 2015.

[18] M. Gurbuzbalaban, A. Ozdaglar, and P. A. Parrilo. Why random reshuffling beats stochastic gradient descent. *arXiv preprint: 1510.08560*, 2015.

[19] M. Gurbuzbalaban, A. Ozdaglar, and P. A. Parrilo. On the convergence rate of incremental aggregated gradient algorithms. *SIAM Journal on Optimization*, 27(2):1035–1048, 2017.

[20] J. Romvary, G. Ferro, and A.M. Annaswamy. A distributed proximal atomic coordination algorithm. *IEEE Transactions on Automatic Control*, 2019 (submitted).

[21] J. Romvary. *A proximal atomic coordination algorithm for distributed optimization in distribution grids.* PhD thesis, Massachusetts Institute of Technology, 2018.

[22] R. Haider, S. Baros, Y. Wasa, J. Romvary, K. Uchida, and A.M. Anaswamy. Towards a retail market for distribution grids. *IEEE Transactions on Smart Grids*, 2019 (under review). Available at http://aaclab.mit.edu/resources/TowardsARetailMarket$_r$*evision.pdf*.

[23] Daniel K. Molzahn, Florian Dörfler, Henrik Sandberg, Steven H. Low, Sambuddha Chakrabarti, Ross Baldick, and Javad Lavaei. A survey of distributed optimization and control algorithms for electric power systems. *IEEE Transactions on Smart Grid*, 8(6):2941–2962, 2017.

[24] Emiliano Dell'anese, Hao Zhu, and Georgios B. Giannakis. Distributed optimal power flow for smart microgrids. *IEEE Transactions on Smart Grid*, 4(3):1464âĂŞ1475, 2013.

[25] Brett A. Robbins and Alejandro D. Dominguez-Garcia. Optimal reactive power dispatch for voltage regulation in unbalanced distribution systems. *IEEE Transactions on Power Systems*, 31(4):2903âĂŞ2913, 2016.

[26] Garth P. Mccormick. Computability of global solutions to factorable nonconvex programs: Part i âĂŤ convex underestimating problems. *Mathematical Programming*, 10(1):147âĂŞ175, 1976.

[27] Garth P. Mccormick. Computability of global solutions to factorable nonconvex programs: Part i âĂŤ convex underestimating problems. *Mathematical Programming*, 10(1):147âĂŞ175, 1976.

[28] Ipopt documentation.

[29] Giulio Ferro, Michela Robba, Mansueto Rossi, and Anuradha M. Annaswamy. A new convex relaxation of the optimal power flow problem for distribution grids with unbalanced structure and mesh topology. 2020 (unpublished).

[30] The Mathworks, Inc., Natick, Massachusetts. *MATLAB version 9.5.0.713579 (R2018b)*, 2018.

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

[32] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2019.

[33] S. Gill and G. W. Ault. Dynamic optimal power flow for active distribution networks. *IEEE Transactions on Power Systems*, 29(1):121–131, Jan. 2014.

[34] L. H. Macedo, J. F. Franco, M. J. Rider, and R. Romero. Optimal operation of distribution networks considering energy storage devices. *IEEE Transactions on Smart Grid*, 6(6):2825–2836, Nov. 2015.

[35] M. Nick, R. Cherkaoui, and M. Paolone. Optimal allocation of dispersed energy storage systems in active distribution networks for energy balance and grid support. *IEEE Transactions on Power Systems*, 29(5):2300–2310, Sept. 2014.

[36] Homer. Homer pro 3.13. https://www.homerenergy.com/products/pro/docs/latest/battery_minimum_state_of_charge.html, 2019. Accessed: 2020-01-07.

[37] ISO-NE. Pricing reports. https://www.iso-ne.com/isoexpress/web/reports/pricing/-/tree/lmps-rt-five-minute-final, 2019.

[38] ISO-NE. Energy, load, and demand reports. https://www.iso-ne.com/isoexpress/web/reports/load-and-demand/-/tree/dmnd-five-minute-sys, 2019.

[39] National Renewable Energy Laboratory (NREL). Annual technology baseline. 2017. Accessed: 2020-01-10.

[40] NERC. About nerc. https://www.nerc.com/AboutNERC/Pages/default.aspx, 2017. Accessed: 2020-01-05.

[41] FERC. What ferc does. https://www.ferc.gov/about/ferc-does.asp, 2018. Accessed: 2020-01-05.

[42] US Energy Information Administration (EIA). Us electric system is made up of interconnections and balancing authorities. https://www.eia.gov/todayinenergy/detail.php?id=27152, 2016. Accessed: 2020-01-05.

[43] ISONE. Faqs: Day-ahead energy marketâĂŤcommitment, scheduling, and dispatch. https://www.iso-ne.com/participate/support/faq/da-market-commitment, 2019. Accessed: 2020-01-06.

[44] CAISO. Market processes and products. http://www.caiso.com/market/Pages/MarketProcesses.aspx, 2019. Accessed: 2020-01-06.

[45] Jerry Webb. Indiana utility regulatory commission: Ancillary services markets. https://pubs.naruc.org/pub.cfm?id=5375186A-2354-D714-51D9-83D16B591C58, 2007. Accessed: 2020-01-06.

[46] ISO New England. Iso new england pricing reports. https://www.iso-ne.com/isoexpress/web/reports/pricing/-/tree/ancillary, 2018.

[47] P. Maloney. FERC order opens 'floodgates' for energy storage in wholesale markets. https://www.utilitydive.com/news/ferc-order-opens-floodgates-for-energy-storage-in-wholesale-markets/517326/, Feb. 2018.

[48] Farrokh Rahimi and Ali Ipakchi. Using a transactive energy framework. *IEEE Electrification Magazine*, Dec 2016.

[49] M. Yazdanian and A. Mehrizi-Sani. Distributed Control Techniques in Microgrids. *IEEE Transactions on Smart Grid*, 5(6):2901–2909, November 2014.

[50] K. E. Antoniadou-Plytaria, I.N. Kouveliotis-Lysikatos, P. S. Georgilakis, and N. D. Hatziargyriou. Distributed and Decentralized Voltage Control of Smart Distribution Networks: Models, Methods, and Future Research. *IEEE Transactions on Smart Grid*, 8(6):2999–3008, November 2017.

[51] Tao Chen, Qais Alsafasfeh, Hajir Pourbabak, and Wencong Su. The next-generation us retail electricity market with customers and prosumers âĂŞ a bibliographical survey. *Energies*, 2018.

[52] S. D. Manshadi and M. E. Khodayar. A Hierarchical Electricity Market Structure for the Smart Grid Paradigm. *IEEE Transactions on Smart Grid*, 7(4):1866–1875, Jul. 2016.

[53] M. Hofling, F. Heimgartner, B. Litfinski, and M. Menth. A Perspective on the Future Retail Energy Market. In *Proceedings of the International Workshops SOCNET 2014 and FGENET 2014*, pages 87–95, 2014.

[54] S. Parhizi, A. Khodaei, and S. Bahramirad. Distribution Market Clearing and Settlement. In *Proceedings of the IEEE Power and Energy Society General Meeting*, Jul. 2016.

[55] L. Bai, J. Wang, C. Wang, C. Chen, and F. F. Li. Distribution Locational Marginal Pricing (DLMP) for Congestion Management and Voltage Support. *IEEE Transactions on Power Systems*, Oct. 2017.

[56] Yahya K. Renani, Mehdi Ehsan, and Mohammad Shahidehpour. Optimal transactive market operations with distribution system operators. *IEEE Transactions on Smart Grids*, 9(6), Nov 2018.

[57] J. Wei, Y. Zhang, F. Sahriatzadeh, and A. K. Srivastava. Dlmp using three-phase current injection opf with renewables and demand response. *IET Renewable Power Generation*, 13(7):1160–1167, July 2019.

[58] J. Wei, L. Corson, and A. K. Srivastava. Three-phase optimal power flow based distribution locational marginal pricing and associated price stability. *IEEE Power and Energy Society General Meeting*, July 2015.

[59] C-H Lo and N. Ansari. Decentralized Controls and Communications for Autonomous Distribution Networks in Smart Grid. *IEEE Transactions on Smart Grid*, 4(1):66–77, March 2013.

[60] Z. Wang, B. Chen, J. Wang, and J. Kim. Decentralized Energy Management System for Networked Microgrids in Grid-Connected and Islanded Modes. *IEEE Transactions on Smart Grid*, 7(2):1097–1105, March 2016.

[61] N. Li. A Market Mechanism for Electric Distribution Networks. In *Proceedings of the 54th Conference on Decision and Control (CDC)*, Dec. 2015.

[62] Thomas Morstyn, Alexander Teytelboym, and Malcolm D. McCulloch. Designing decentralized markets for distribution system flexibility. *IEEE Transactions on Power Systems*, 34(3), May 2019.

[63] Claudia Pop, Tudor Cioara, Marcel Antal, Ionut Anghel, Ioan Salomie, and Massimo Bertoncini. Blockchain based decentralized management of demand response programs in smart energy grids. *Sensors*, 2018.

[64] Ni Zhang, Yu Yan, and Shengyao Xu. Game-theory-based electricity market clearing mechanisms for an open and transactive distribution grid. In *2015 IEEE Power & Energy Society General Meeting*, July 2015.

[65] M. Caramanis, E. Ntakou, W. W. Hogan, A. Chakrabortty, and J. Schoene. Co-Optimization of Power and Reserves in Dynamic T and D Power Markets With Nondispatchable Renewable Generation and Distributed Energy Resources. *Proceedings of the IEEE*, 104(4):807–836, Apr. 2016.

[66] Kuo Feng Hong Zhou, Zhi-Wei Liu, and Dandan Hu. Retail market pricing design in smart distribution networks considering wholesale market price uncertainty. In *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, Nov 2017.

[67] Hajir Pourbabak, Tao Chen, and Wencong Su. Consensus-based distributed control for economic operation of distribution grid with multiple consumers and prosumers. In *2016 IEEE Power and Energy Society General Meeting (PESGM)*, July 2016.

[68] Navid Rahbari-Asr, Unnati Ojha, Ziang Zhang, and Mo-Yuen Chow. Incremental welfare consensus algorithm for cooperative distributed generation/demand response in smart grid. *IEEE Transactions on Smart Grid*, Nov 2014.

[69] Soummya Kar, Gabriela Hug, Javad Mohammadi, and Moura Jose M.F. Distributed state estimation and energy management in smart grids: A consensus+innovations approach. *IEEE Journal of Selected Topics in Signal Processing*, Dec 2014.

[70] W. Shi, X. Xie, Chi-Cheng Chu, and R. Gadh. Distributed Optimal Energy Management in Microgrids. *IEEE Transactions on Smart Grid*, 6(3):1137–1146, May 2015.

[71] Y. Zhang, N. Gatsis, and G. B. Giannakis. Robust Energy Management for Microgrids With High-Penetration Renewables. *IEEE Transactions on Sustainable Energy*, 4(4):944–953, October 2013.

[72] H. S. V. S. K. Nunna and S. Doolla. Multiagent-Based Distributed-Energy-Resource Management for Intelligent Microgrids. *IEEE Transactions on Industrial Electronics*, 60(4):1678–1687, April 2013.

[73] Y. Xu and Z. Li. Distributed Optimal Resource Management Based on the Consensus Algorithm in a Microgrid. *IEEE Transactions on Industrial Electronics*, 62(4):2584–2592, April 2015.

[74] Qiuye Sun, Renke Han, Huaguang Zhang, Jianguo Zhou, and Josep M Guerrero. A multiagent-based consensus algorithm for distributed coordinated control of distribued generators in the energy internet. *IEEE Transactions on Smart Grid*, Nov 2015.

[75] J. W. Simpson-Porco, Q. Shafiee, F. Dörfler, J. C. Vasquez, J. M. Guerrero, and F. Bullo. Secondary Frequency and Voltage Control of Islanded Microgrids via Distributed Averaging. *IEEE Transactions on Industrial Electronics*, 62(11):7025–7038, November 2015.

[76] A. Makhdoumi and A. Ozdaglar. Convergence rate of distributed admm over networks. *IEEE Transactions on Automatic Control*, 52(10):5082–5095, 2017.

[77] Y. Zhang, M. Hong, E. Dall'Anese, S. Dhople, and Z. Xu. Distributed Controllers Seeking AC Optimal Power Flow Solutions Using ADMM. *IEEE Transactions on Smart Grid*, Feb. 2017.

[78] P. Kansal and A. Bose. Bandwidth and latency requirements for smart transmission grid applications. *IEEE Transactions on Smart Grid*, 2(3), Sept. 2012.

[79] Y-J Kim, M. Thottan, V. Kolesnikov, and W Lee. A secure decentralized data-centric information infrastructure for smart grid. *IEEE Communications Magazine*, 48(11):58–65, Nov. 2010.

[80] M. Kuzlu, M. Pipattanasomporn, and S. Rahman. Communication network requirements for major smart grid applications in han, nan, and wan. *Computer Networks*, 67:74–88, 2014.

[81] Y. Hayashi et al. Versatile Modeling Platform for Cooperative Energy Management Systems in Smart Cities. *Proceedings of the IEEE*, 106(4):594–612, Mar. 2018.

[82] Tokyo Electric Power Company. https://www4.tepco.co.jp/en/forecast/html/index-e.html.

[83] Japan Electric Power eXchange. http://www.jepx.org/english/index.html.

[84] OVO Energy. Average electricity prices around the world: $/kwh. https://www.ovoenergy.com/guides/energy-guides/average-electricity-prices-kwh.html, 2019.

[85] Tokyo Electric Power Company. Electricity rate plans. https://www7.tepco.co.jp/ep/rates/electricbill-e.html, 2019.

[86] IEEE Power and Energy Society. Ieee pes amps dsas test feeder working group: Resources. Technical report.

[87] Helena Gerard, Enrique Israel Rivero Puente, and Daan Six. Coordination between transmission and distribution system operators in the electricity sector: A conceptual framework. *Utilities Policy*, 50:40–48, 2018.

[88] G. L. Barbose, N. R. Darghouth, B. Hoen, and R. H. Wiser. Income trends of residential pv adopters: An analysis of household-level income estimates. *LBNL Energy Technologies Area*, Apr 2018.

[89] Scott Burger, Ian Schneider, Audun Botterud, and Ignacio PÃl'rez-Arriaga. Consumers, prosumers, prosumagers: How service innovations will disrupt the utility business model. 2019.

[90] *Standard models for Japanese power system.* IEEJ Tech. Report, no. 754 (in Japanese), 1999.