Foundations and Trends® in Optimization
Vol. 1, No. 2 (2013) 70–122
© 2013 M. Kraning, E. Chu, J. Lavaei, and S. Boyd
DOI: xxx

Dynamic Network Energy Management via Proximal Message Passing

Matt Kraning
Stanford University
mkraning@stanford.edu

Eric Chu
Stanford University
echu508@stanford.edu

Javad Lavaei
Columbia University
lavaei@ee.columbia.edu

Stephen Boyd
Stanford University
boyd@stanford.edu

Contents

Abstract

We consider a network of devices, such as generators, fixed loads, deferrable loads, and storage devices, each with its own dynamic constraints and objective, connected by AC and DC lines. The problem is to minimize the total network objective subject to the device and line constraints over a time horizon. This is a large optimization problem with variables for consumption or generation for each device, power flow for each line, and voltage phase angles at AC buses in each period.

We develop a decentralized method for solving this problem called *proximal message passing*. The method is iterative: At each step, each device exchanges simple messages with its neighbors in the network and then solves its own optimization problem, minimizing its own objective function, augmented by a term determined by the messages it has received. We show that this message passing method converges to a solution when the device objective and constraints are convex. The method is completely decentralized, and needs no global coordination other than synchronizing iterations; the problems to be solved by each device can typically be solved extremely efficiently and in parallel.

The proximal message passing method is fast enough that even a serial implementation can solve substantial problems in reasonable time frames. We report results for several numerical experiments, demonstrating the method's speed and scaling, including the solution of a problem instance with over 30 million variables in 5 minutes for a serial implementation; with decentralized computing, the solve time would be less than one second.

1

Introduction

1.1 Overview

A traditional power grid is operated by solving a number of optimization problems. At the transmission level, these problems include unit commitment, economic dispatch, optimal power flow (OPF), and security-constrained OPF (SC-OPF). At the distribution level, these problems include loss minimization and reactive power compensation. With the exception of the SC-OPF, these optimization problems are static with a modest number of variables (often less than 10000), and are solved on time scales of 5 minutes or more. However, the operation of next generation electric grids (*i.e.*, smart grids) will rely critically on solving large-scale, dynamic optimization problems involving hundreds of thousands of devices jointly optimizing tens to hundreds of millions of variables, on the order of seconds rather than minutes [16, 41]. More precisely, the distribution level of a smart grid will include various types of active dynamic devices, such as distributed generators based on solar and wind, batteries, deferrable loads, curtailable loads, and electric vehicles, whose control and scheduling amount to a very complex power management problem [59, 9].

In this paper, we consider a general problem, which we call the *dynamic optimal power flow problem* (D-OPF), in which dynamic devices are connected by both AC and DC lines, and the goal is to jointly minimize a network objective subject to local constraints on the devices and lines. The network objective is the sum of the objective functions of the devices. These objective functions extend over a given time horizon and encode operating costs such as fuel consumption and constraints such as limits on power generation or consumption. In addition, the objective functions encode dynamic objectives and constraints such as limits on ramp rates for generators or charging and capacity limits for storage devices. The variables for each device consist of its consumption or generation in each time period and can also include local variables which represent internal states of the device over time, such as the state of charge of a storage device.

When all device objective functions and line constraints are convex, D-OPF is a convex optimization problem, which can in principle be solved efficiently [7]. If not all device objective functions are convex, we can solve a relaxed form of the D-OPF which can be used to find good, local solutions to the D-OPF. The optimal value of the relaxed D-OPF also gives a lower bound for the optimal value of the D-OPF which can be used to evaluate the suboptimality of a local solution, or, when the local solution has the same value, as a certificate of global optimality.

For any network, the corresponding D-OPF contains at least as many variables as the number of devices and lines multiplied by the length of the time horizon. For large networks with hundreds of thousands of devices and a time horizon with tens or hundreds of time periods, the extremely large number of variables present in the corresponding D-OPF makes solving it in a centralized fashion computationally impractical, even when all device objective functions are convex.

We propose a decentralized optimization method which efficiently solves the D-OPF by distributing computation across every device in the network. This method, which we call *proximal message passing*, is iterative: At each iteration, every device passes simple messages to its neighbors and then solves an optimization problem that minimizes the sum of its own objective function and a simple regularization term that only depends on the messages it received from its neighbors in the previous iteration. As a result, the only non-local coordination needed between devices for proximal message passing is synchronizing iterations. When all device objective functions are convex, we show that proximal message passing converges to a solution of the D-OPF.

Our algorithm can be used several ways. It can be implemented in a traditional way on a single computer or cluster by collecting all the device constraints and objectives. We will demonstrate this use with an implementation that runs on a single 32-core computer with hyperthreading (64 independent threads). A more interesting use is in a peer-to-peer architecture, in which each device contains its own processor, which carries out the required local dynamic optimization and exchanges messages with its neighbors on the network. In this setting, the devices do not need to divulge their objectives or constraints; they only need to support a simple protocol for interacting with their neighbors. Our algorithm ensures that the network power flows and AC bus phase angles will converge to their optimal values, even though each device has very little information about the rest of the network, and only exchanges limited messages with its immediate neighbors.

Due to recent advances in convex optimization [61, 46, 47], in many cases the optimization problems that each device solves in each iteration of proximal message passing can be executed at millisecond or even microsecond time-scales on inexpensive, embedded processors. Since this execution can happen in parallel across all devices, the entire network can execute proximal message passing at kilohertz rates. We present a series of numerical examples to illustrate this fact by using proximal message passing to solve instances of the D-OPF with over 30 million variables serially in 5 minutes. Using decentralized computing, the solve time would be essentially independent of the size of the network and require just a fraction of a second.

We note that although a primary application for proximal message passing is power management, it can easily be adapted to more general resource allocation and graph-structured optimization problems [51, 2].

1.2 Related work

The use of optimization in power systems dates back to the 1920s and has traditionally concerned the optimal dispatch problem [22], which aims to find the lowest cost method for generating and delivering power to consumers, subject to physical generator constraints. With the advent of computer and communication networks, many different ways to numerically solve this problem have been proposed [62] and more sophisticated variants of optimal dispatch have been introduced, such as OPF, economic dispatch, and dynamic dispatch [12], which extend optimal dispatch to include various reliability and dynamic constraints. For reviews of optimal and economic dispatch as well as general power systems, see [4] and the book and review papers cited above.

When modeling AC power flow, the D-OPF is a dynamic version of the OPF [8], extending the latter to include many more types of devices such as storage units. Recent smart grid research has focused on the ability of storage devices to cut costs and catalyze the consumption of variable and intermittent renewables in the future energy market [23, 44, 13, 48]. With D-OPF, these storage concerns are directly addressed and modeled in the problem formulation with the introduction of a time horizon and coupling constraints between variables across periods.

Distributed optimization methods are naturally applied to power networks given the graph-structured nature of the transmission and distribution networks. There is an extensive literature on distributed optimization methods, dating back to the early 1960s. The prototypical example is dual decomposition [14, 17], which is based on solving the dual problem by a gradient method. In each iteration, all devices optimize their local (primal) variables based on current prices (dual variables). Then the dual variables are updated to account for imbalances in supply and demand, with the goal being to determine prices for which supply equals demand.

Examples of distributed algorithms in the power systems literature include two phase procedures that resemble a single iteration of dual decomposition. In the first phase, dynamic prices are set over a given time horizon (usually hourly over the following 24 hours) by some mechanism (*e.g.*, centrally by an ISO [28, 29], or through information aggregation in a market [57]). In the second phase, these prices allow individual devices to jointly optimize their power flows with minimal (if any) additional coordination over the time horizon. More recently, building on the work of [39], a distributed algorithm was proposed [38] to solve the dual OPF using a standard dual decomposition on subsystems that are maximal cliques of the power network.

Dual decomposition methods are not robust, requiring many technical conditions, such as strict convexity and finiteness of all local cost functions, for both theoretical and practical convergence to optimality. One way to loosen the technical conditions is to use an augmented Lagrangian [25, 49, 5], resulting in the method of multipliers. This subtle change allows the method of multipliers to converge under mild technical conditions, even when the local (convex) cost functions are not strictly convex or necessarily finite. However, this method has the disadvantage of no longer being separable across subsystems. To achieve both separability and robustness for distributed optimization, we can instead use the *alternating direction method of multipliers* (ADMM) [21, 20, 15, 6]. ADMM is very closely related to many other algorithms, and is identical to Douglas-Rachford operator splitting; see, *e.g.*, the discussion in [6, §3.5].

Augmented Lagrangian methods (including ADMM) have previously been applied to the study of power systems with static, single period objective functions on a small number of distributed subsystems, each representing regional power generation and consumption [35]. For an overview of related decomposition methods applied to power flow problems, we direct the reader to [36, 1] and the references therein.

The proximal message passing decentralized power scheduling method is similar in spirit to flow control on a communication network, where each source modulates its sending rate based only on information about the number of un-acknowledged packets; if the network state remains constant, the flows converge to levels that satisfy the constraints and maximize a total utility function [33, 42]. In Internet flow control, this is called *end-point control*, since flows are controlled (mostly) by devices on the edges of the network. A decentralized proximal message passing method is closer to local control, since decision making is based only on interaction with neighbors on the network. Another difference is that the messages our method passes between devices are virtual, and not actual energy flows. (Once converged, of course, they can become actual energy flows.)

1.3 Outline

The rest of this paper is organized as follows. In chapter 2 we give the formal definition of our network model. In chapter 3 we give examples of how to model specific devices such as generators, deferrable loads and energy storage systems in our formal framework. In Chapter 4, we describe the role that convexity plays in the D-OPF and introduce the idea of convex relaxations as a tool to find solutions to the D-OPF in the presence of non-convex device objective functions. In Chapter 5 we derive the proximal message passing equations. In Chapter 6 we present a series of numerical examples, and in Chapter 7 we discuss how our framework can be extended to include use cases we do not explicitly cover in this paper.

2

Network Model

We begin with an abstract definition of our network model and the dynamic optimal power flow problem, and the compact notation we use to describe it. We then give some discussion and an example to illustrate how the model is used to describe a real power network.

2.1 Formal definition and notation

A network consists of a finite set of *terminals* $\mathcal{T}$, a finite set of *devices* $\mathcal{D}$, and a finite set of *nets* $\mathcal{N}$. The sets $\mathcal{D}$ and $\mathcal{N}$ are both partitions of $\mathcal{T}$. Thus, each device and each net has a set of terminals associated with it, and each terminal is associated with exactly one device and exactly one net. Equivalently, a network can be defined as a bipartite graph with one set of vertices given by devices, the other set of vertices given by nets, and edges given by terminals — very similar in nature to 'normal realizations' [19] of graphs in coding theory.

Each terminal $t \in \mathcal{T}$ has a *type*, either AC or DC, corresponding to the type of power that flows through the terminal. The set of terminals can be partitioned by type into the sets $\mathcal{T}^{\mathrm{dc}}$ and $\mathcal{T}^{\mathrm{ac}}$, which represent the set of all terminals of type DC and AC, respectively. A terminal of either type has an associated *power schedule* $p_t = (p_t(1), \ldots, p_t(T)) \in$

$\mathbf{R}^T$, where T is a given time horizon. Here, $p_t(\tau)$ is the amount of power consumed by device d in time period τ through terminal t, where t is associated with d. When $p_t(\tau) < 0$, $-p_t(\tau)$ is the energy generated by device d through terminal t in time period τ. (For AC terminals, p_t is the real power flow; we do not consider reactive power in this paper.) In addition to (real) power schedules, AC terminals $t \in \mathcal{T}^{\mathrm{ac}}$ also have *phase schedules* $\theta_t = (\theta_t(1), \dots, \theta_t(T)) \in \mathbf{R}^T$, which represent the absolute voltage phase angles for terminal t over time. (DC terminals are not associated with voltage phase angles.)

We use a simple method for indexing quantities such as power schedules that are associated with each terminal and vary over time. For devices $d \in \mathcal{D}$, we use 'd' to refer to both the device itself as well as the set of terminals associated with it, *i.e.*, we say $t \in d$ if terminal t is associated with device d. The set of all power schedules associated with device d is denoted by $p_d = \{p_t \mid t \in d\}$, which we can associate with a $|d| \times T$ matrix. We use the same notation for nets as we do for devices. The set of all terminal power schedules is denoted by $p = \{p_t \mid t \in \mathcal{T}\}$, which we can associate with a $|\mathcal{T}| \times T$ matrix. For other quantities that are associated with each terminal (such as phase schedules), we use an identical notation to power schedules, *i.e.*, $\theta_d = \{\theta_t \mid t \in d\}$ is the set of phase schedules associate with device d (with an identical notation for nets), and the set of all phase schedules is denoted by $\theta = \{\theta_t \mid t \in \mathcal{T}\}$.

Each device d contains a set of $|d|$ terminals and has an associated *objective function* $f_d : \mathbf{R}^{|d| \times T} \times \mathbf{R}^{|d^{\mathrm{ac}}| \times T} \to \mathbf{R} \cup \{+\infty\}$, where $d^{\mathrm{ac}} = \{t \mid t \in d \cap \mathcal{T}^{\mathrm{ac}}\}$ is the set of all AC terminals associated with device d, and we set $f_d(p_d, \theta_d) = \infty$ to encode constraints on the power and phase schedules for the device. When $f_d(p_d, \theta_d) < \infty$, we say that (p_d, θ_d) are a set of *realizable* power and phase schedules for device d, and we interpret $f_d(p_d, \theta_d)$ as the cost (or revenue, if negative) to device d for operating according to power schedule p_d and phase schedule θ_d.

Similarly, each net $n \in \mathcal{N}$ contains a set of $|n|$ terminals, all of which are required to have the same type. (We will model AC–DC conversion using devices.) We refer to nets containing AC terminals as *AC nets* and nets containing DC terminals as *DC nets*. Nets are lossless energy carriers which constrain the power schedules (and phase schedules in the case of AC nets) of their constituent terminals: we require *power balance* in each time period, which is represented by the constraints

$$\sum_{t \in n} p_t(\tau) = 0, \quad \tau = 1, \ldots, T, \tag{2.1}$$

for each $n \in \mathcal{N}$. In addition to power balance, each AC net imposes the *phase consistency constraints*

$$\theta_{t_1}(\tau) = \cdots = \theta_{t_{|n|}}(\tau), \quad \tau = 1, \ldots, T, \tag{2.2}$$

where $n = \{t_1, \ldots, t_{|n|}\}$. In other words, in each time period the power flows on each net balance, and all terminals on the same AC net have the same phase.

We define the *average net power imbalance* $\bar{p} : \mathcal{T} \to \mathbf{R}^T$, as

$$\bar{p}_t = \frac{1}{|n|} \sum_{t' \in n} p_{t'}, \tag{2.3}$$

where $t \in n$, *i.e.*, terminal t is associated with net n. In other words, $\bar{p}_t(\tau)$ is the average power schedule of all terminals associated with the same net as terminal t at time τ. We overload this notation for devices by defining $\bar{p}_d = \{\bar{p}_t \mid t \in d\}$. Using an identical notation for nets, we can see that $\bar{p}_n$ simply contains $|n|$ copies of the average net power imbalance for net n. The net power balance constraint for all terminals can be expressed as $\bar{p} = 0$.

For AC terminals, we define the *phase residual* $\tilde{\theta} : \mathcal{T}^{\mathrm{ac}} \to \mathbf{R}^T$ as

$$\tilde{\theta}_t = \theta_t - \frac{1}{|n|} \sum_{t' \in n} \theta_{t'} = \theta_t - \bar{\theta}_t,$$

where $t \in n$ and n is an AC net. In other words, $\tilde{\theta}_t(\tau)$ is the difference between the phase angle of terminal t and the average phase angle of all terminals attached to net n, at time τ. As with the average power imbalance, we overload this notation for devices by defining $\tilde{\theta}_d = \{\tilde{\theta}_t \mid t \in d \cap \mathcal{T}^{\mathrm{ac}}\}$ with a similar notation for nets. The phase consistency constraint for all AC terminals can be expressed as $\tilde{\theta} = 0$.

2.2 Dynamic optimal power flow problem

We say that a set of power and phase schedules $p : \mathcal{T} \to \mathbf{R}^T$, $\theta : \mathcal{T}^{\mathrm{ac}} \to \mathbf{R}^T$ is *feasible* if $f_d(p_d, \theta_d) < \infty$ for all $d \in \mathcal{D}$ (*i.e.*, all devices' power and phase schedules are realizable), and both $\bar{p} = 0$ and $\tilde{\theta} = 0$ (*i.e.*, power balance and phase consistency holds across all nets). We define the network objective as $f(p, \theta) = \sum_{d \in \mathcal{D}} f_d(p_d, \theta_d)$. The *dynamic optimal power flow problem* (D-OPF) is

$$\begin{array}{ll} \text{minimize} & f(p, \theta) \\ \text{subject to} & \bar{p} = 0, \quad \tilde{\theta} = 0, \end{array} \tag{2.4}$$

with variables $p : \mathcal{T} \to \mathbf{R}^T$, $\theta : \mathcal{T}^{\text{ac}} \to \mathbf{R}^T$. We refer to p and θ as *optimal* if they solve (2.4), *i.e.*, globally minimize the objective among all feasible p and θ. We refer to p and θ as *locally optimal* if they are a locally optimal point for (2.4).

Dual variables and locational marginal prices. Suppose p^0 is a set of optimal power schedules, that also minimizes the Lagrangian

$$f(p, \theta) + \sum_{t \in \mathcal{T}} \sum_{\tau=1}^{T} (y_p^0)_t(\tau) \bar{p}_t(\tau),$$

subject to $\tilde{\theta} = 0$, where $y_p^0 : \mathcal{T} \to \mathbf{R}^T$ are the dual variables associated with the power balance constraint $\bar{p} = 0$. (This is actually the partial Lagrangian as we only dualize the power balance constraints, but not the phase consistency constraints.) In this case we call y_p^0 a set of optimal Lagrange multipliers or dual variables. When p^0 is a locally optimal point, which also locally minimizes the Lagrangian, then we refer to y_p^0 as a set of locally optimal Lagrange multipliers.

The dual variables y_p^0 are related to the traditional concept of locational marginal prices $\mathcal{L}^0 : \mathcal{T} \to \mathbf{R}^T$ by rescaling the dual variables associated with each terminal according to the size of its associated net, *i.e.*, $\mathcal{L}_t^0 = |n|(y_p^0)_t$, where $t \in n$. This rescaling is due to the fact that locational marginal prices are the dual variables associated with the constraints in (2.1) rather than their scaled form used in (2.4) [18].

2.3 Discussion

We now describe our model in a less formal manner. Generators, loads, energy storage systems, and other power sources and sinks are modeled as single terminal devices. Transmission lines (or more generally, any wire or set of wires that conveys power), AC-DC converters, and AC phase shifters are modeled as two-terminal devices. Terminals are ports on a device through which power flows, either into or out of the device (or both, at different times, as happens in a storage device). The flow for AC terminals could be, *e.g.*, three phase, two phase, single phase, 230V, or 230kV; the flow for DC terminals could be high voltage DC, 12V DC, or a floating voltage (*e.g.*, the output of a solar panel). We model these real cases with a different type for each mechanism (*e.g.*, two and three phase AC terminals would have distinct types and could not be connected to the same net).

Nets are used to model ideal lossless uncapacitated connections between terminals over which power is transmitted and physical constraints hold (*e.g.*, equal voltages, currents summing to zero); losses, capacities, and more general connection constraints between a set of terminals can be modeled with the addition of a device and individual nets which connect each terminal to the new device. An AC net corresponds to a direct connection between its associated terminals (*e.g.*, a bus); all the terminals' voltage phases must be the same and their power is conserved. A two terminal DC net is just a wired connection of the terminals. A DC net with more than two terminals is a smart power router, which actively chooses how to distribute the incoming and outgoing power flows among its terminals.

The objective function of a device is used to measure the cost (which can be negative, representing revenue) associated with a particular mode of operation, such as a given level of consumption or generation of power. This cost can include the actual direct cost of operating according to the given power schedules, such as a fuel cost, as well as other costs such as CO_2 generation, or costs associated with increased maintenance and decreased system lifetime due to structural fatigue. The objective function can also include local variables other than power and phase schedules, such as the state of charge of a storage device.

Constraints on the power and phase schedules and internal variables for a device are encoded by setting the objective function to $+\infty$ for power and phase schedules that violate the constraints. In many cases, a device's objective function will only take on the values 0 and $+\infty$,

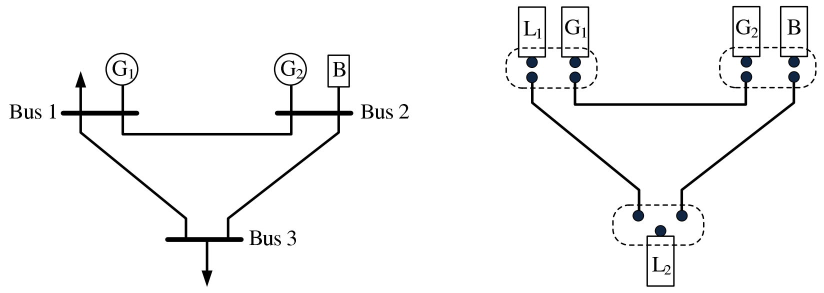

Figure 2.1: A simple network (*left*); its transformation into standard form (*right*).

indicating no local preference among feasible power and phase schedules. Many devices, especially single-terminal devices such as loads or generators, impose no constraints on their AC terminals' phase angles; in other words, these terminals have 'floating' voltage phase angles, which are free to take any value.

2.4 Example

We illustrate how a traditional power network can be recast into our network model in Figure 2.1. The original power network, shown on the left, contains 2 loads, 3 buses, 3 transmission lines, 2 generators, and a single battery storage system. We can transform this small power grid into our model by representing it as a network with 11 terminals, 8 devices, and 3 nets, shown on the right of figure 2.1. Terminals are shown as small filled circles. Single terminal devices, which are used to model loads, generators, and the battery, are shown as boxes. The transmission lines are two terminal devices represented by solid lines. The nets are shown as dashed rounded boxes. Terminals are associated with the device they touch and the net in which they are contained.

The set of terminals can be partitioned by either the devices they are associated with, or the nets in which they are contained. Figure 2.2 shows the network in Figure 2.1 as a bipartite graph, with devices on the left and nets on the right. In this representation, terminals are represented by the edges of the graph.

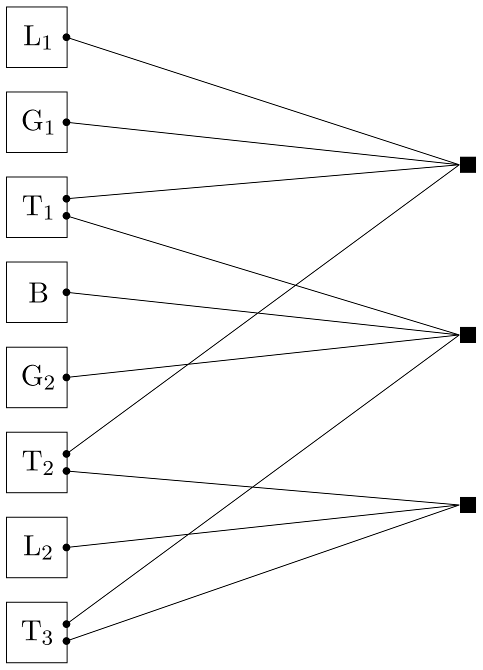

Figure 2.2: The network in Figure 2.1 represented as a bipartite graph. Devices (boxes) are shown on the left with their associated terminals (dots). The terminals are connected to their corresponding nets (solid boxes) on the right.

3

Device Examples

In this chapter we present several examples of how common devices can be modeled in our framework. These examples are intentionally kept simple, but could easily be extended with more refined objectives and constraints. In these examples, it is easier to discuss operational costs and constraints for each device separately. A device's objective function is equal to the device's cost function unless any constraint is violated, in which case we set the objective value to $+\infty$. For all single terminal devices, we describe their objective and constraints in the case of a DC terminal. For AC terminal versions of one terminal devices, the cost functions and constraints are identical to the DC case, and the device imposes no constraints on the phase schedule.

3.1 Generators

A generator is a single-terminal device with power schedule p_{gen}, which generates power over a range, $P^{\min} \leq -p_{\text{gen}} \leq P^{\max}$, and has ramp-rate constraints

$$R^{\min} \leq -Dp_{\text{gen}} \leq R^{\max},$$

which limit the change of power levels from one period to the next. Here, the operator $D \in \mathbf{R}^{(T-1)\times T}$ is the forward difference operator, defined as

$$(Dx)(\tau) = x(\tau + 1) - x(\tau), \quad \tau = 1, \ldots, T - 1.$$

The cost function for a generator has the separable form

$$\psi_{\text{gen}}(p_{\text{gen}}) = \sum_{\tau=1}^{T} \phi_{\text{gen}}(-p_{\text{gen}}(\tau)),$$

where $\phi : \mathbf{R} \to \mathbf{R}$ gives the cost of operating the generator at a given power level over a single time period. This function is typically, but not always, convex and increasing. It could be piecewise linear, or, for example, quadratic:

$$\phi_{\text{gen}}(x) = \alpha x^2 + \beta x,$$

where $\alpha, \beta > 0$.

More sophisticated models of generators allow for them to be switched on or off, with an associated cost each time they are turned on or off. When switched on, the generator operates as described above. When the generator is turned off, it generates no power but can still incur costs for other activities such as idling.

3.2 Transmission lines

DC transmission line. A DC transmission line is a device with two DC terminals with power schedules p_1 and p_2 that transports power across some distance. The line has zero cost function, but the power flows are constrained. The sum $p_1 + p_2$ represents the loss in the line and is always nonnegative. The difference $p_1 - p_2$ can be interpreted as twice the power flow from terminal one to terminal two. A DC transmission line has a maximum flow capacity, given by

$$\frac{|p_1 - p_2|}{2} \le C^{\max},$$

and the constraint

$$p_1 + p_2 - \ell(p_1, p_2) = 0,$$

where $\ell(p_1, p_2) : \mathbf{R}^T \times \mathbf{R}^T \to \mathbf{R}_+^T$ is a loss function.

For a simple model of the line as a series resistance R with average terminal voltage V, we have [4]

$$\ell(p_1, p_2) = \frac{R}{V^2}\left(\frac{p_1 - p_2}{2}\right)^2.$$

A more sophisticated model for the capacity of a DC transmission line includes a dynamic thermal model for the temperature of the line, which (indirectly and dynamically) sets the maximum capacity of the line. A simple model for the temperature at time τ, denoted $\xi(\tau)$, is given by the first order linear dynamics

$$\xi(\tau+1) = \alpha\xi(\tau) + (1-\alpha)\xi^{\text{amb}}(\tau) + \beta(p_1(\tau) + p_2(\tau)),$$

for $\tau = 1, \ldots, T-1$, where $\xi^{\text{amb}}(\tau)$ is the ambient temperature at time τ, and α and β are model parameters that depend on the thermal properties of the line. The capacity is then dynamically modulated by requiring that $\xi \le \xi^{\max}$, where $\xi^{\max}$ is the maximum safe temperature for the line.

AC transmission line. An AC transmission line is a device with two AC terminals, with (real) power schedules p_1 and p_2 and terminal voltage phase angles θ_1 and θ_2, that transmits power across some distance. It has zero cost function, but the power flows and voltage phase angles are constrained. Like a DC transmission line, the sum $p_1 + p_2$ represents the loss in the line, ℓ, and is always nonnegative. The difference $p_1 - p_2$ can be interpreted as twice the power flow from terminal one to terminal two. An AC line has a maximum flow capacity given by

$$\frac{|p_1 - p_2|}{2} \le C^{\max}.$$

(A line temperature based capacity constraint as described for a DC transmission line can also be used for AC transmission lines)

We assume the line is characterized by its (series) admittance $g+ib$, with $g > 0$. (We consider the series admittance model for simplicity; for a more general Π model, similar but more complicated equations can be derived.) Under the common assumption that the voltage magnitude is fixed at V [4], the power and phase schedules satisfy the relations

$$p_1 + p_2 = 2gV^2(1 - \cos(\theta_2 - \theta_1)), \qquad \frac{p_1 - p_2}{2} = bV^2 \sin(\theta_2 - \theta_1),$$

which can be combined to give the relations

$$p_1 + p_2 = \frac{1}{4gV^2}(p_1 + p_2)^2 + \frac{g}{4b^2V^2}(p_1 - p_2)^2,$$

$$\frac{p_1 - p_2}{2} = bV^2 \sin(\theta_2 - \theta_1).$$

Transmission lines are rarely operated with a phase angle difference exceeding 15°, and in this regime, the approximation $\sin(\theta_2 - \theta_1) \approx \theta_2 - \theta_1$ holds within 1%. This approximation, known as the 'DC-OPF approximation' [4], is frequently used in power systems analysis and transforms the second relation above into the relation

$$\frac{p_1 - p_2}{2} = bV^2(\theta_2 - \theta_1). \tag{3.1}$$

Note that the capacity limit constrains $|p_1 - p_2|$, which in turn constrains the phase angle difference $|\theta_2 - \theta_1|$; thus we can guarantee that our small angle approximation is good by imposing a capacity constraint (which is possibly smaller than the true line capacity).

3.3 Converters and interface devices

Inverter. An inverter is a device with a DC terminal, *dc*, and an AC terminal, *ac*, that transforms power from DC to AC and has no cost function. An inverter has a maximum power output $C^{\max}$ and a conversion efficiency $\kappa \in (0, 1]$. It can be represented by the constraints

$$-p_{ac}(\tau) = \kappa p_{dc}(\tau), \quad 0 \le -p_{ac}(\tau) \le C^{\max}, \quad \tau = 1, \ldots, T.$$

The voltage phase angle on the AC terminal, θ_{ac}, is unconstrained.

Rectifier. A rectifier is a device with an AC terminal, *ac*, and a DC terminal, *dc*, that transforms power from AC to DC and has no cost function. A rectifier has a maximum power output $C^{\max}$ and a conversion efficiency $\kappa \in (0, 1]$. It can be represented by the constraints

$$-p_{dc}(\tau) = \kappa p_{ac}(\tau), \quad 0 \le -p_{dc}(\tau) \le C^{\max}, \quad \tau = 1, \ldots, T.$$

The voltage phase angle on the AC terminal, θ_{ac}, is unconstrained.

Phase shifter. A phase shifter is a device with two AC terminals, which is a lossless energy carrier that decouples their phase angles and has zero cost function. A phase shifter enforces the power balance and capacity limit constraints

$$p_1(\tau) + p_2(\tau) = 0, \quad |p_1(\tau)| \leq C^{\max}, \quad \tau = 1, \ldots, T.$$

If the phase shifter can only support power flow in one direction, say, from terminal 1 to terminal 2, then in addition we have the inequalities $p_1(\tau) \geq 0$, $\tau = 1, \ldots, T$. The voltage phase angles θ_1 and θ_2 are unconstrained. (Indeed, this what a phase shifter is meant to do.) When there is no capacity constraint, *i.e.*, $C^{\max} = \infty$, we can think of a phase shifter as a special type of net for AC terminals that enforces power balance, but not voltage phase consistency. (However, we model it as a device, not a net.)

External tie with transaction cost. An external tie is a connection to an external source of power. We represent this as a single terminal device with power schedule p_{ex}. In this case, $p_{\mathrm{ex}}(\tau)_- = \max\{-p_{\mathrm{ex}}(\tau), 0\}$ is the amount of energy pulled from the source, and $p_{\mathrm{ex}}(\tau)_+ = \max\{p_{\mathrm{ex}}(\tau), 0\}$ is the amount of energy delivered to the source, at time τ. We have the constraint $|p_{\mathrm{ex}}(\tau)| \leq E^{\max}(\tau)$, where $E^{\max} \in \mathbf{R}^T$ is the transaction limit.

We suppose that the prices for buying and selling energy are given by $c \pm \gamma$ respectively, where $c(\tau)$ is the midpoint price, and $\gamma(\tau) > 0$ is the difference between the price for buying and selling (*i.e.*, the transaction cost). The cost function is then

$$-(c-\gamma)^T (p_{\mathrm{ex}})_+ + (c+\gamma)^T (p_{\mathrm{ex}})_- = -c^T p_{\mathrm{ex}} + \gamma^T |p_{\mathrm{ex}}|,$$

where $|p_{\mathrm{ex}}|$, $(p_{\mathrm{ex}})_+$, and $(p_{\mathrm{ex}})_-$ are all interpreted elementwise.

3.4 Storage devices

A battery is a single terminal energy storage device with power schedule p_{bat}, which can take in or deliver energy, depending on whether it is charging or discharging. The charging and discharging rates are limited by the constraints $-D^{\max} \leq p_{\mathrm{bat}} \leq C^{\max}$, where $C^{\max} \in \mathbf{R}^T$ and

$D^{\max} \in \mathbf{R}^T$ are the maximum charging and discharging rates. At time τ, the charge level of the battery is given by local variables

$$q(\tau) = q^{\text{init}} + \sum_{t=1}^{\tau} p_{\text{bat}}(t), \quad \tau = 1, \ldots, T,$$

where q^{init} is the initial charge. It has zero cost function and the charge level must not exceed the battery capacity, *i.e.*, $0 \leq q(\tau) \leq Q^{\max}$, $\tau = 1, \ldots, T$. It is common to constrain the terminal battery charge $q(T)$ to be some specified value or to match the initial charge q^{init}.

More sophisticated battery models include (possibly state-dependent) charging and discharging inefficiencies as well as charge leakage [26]. In addition, they can include costs which penalize excessive charge-discharge cycling.

The same general form can be used to model other types of energy storage systems, such as those based on super-capacitors, flywheels, pumped hydro, or compressed air, to name just a few.

3.5 Loads

Fixed load. A fixed energy load is a single terminal device with zero cost function which consists of a desired consumption profile, $l \in \mathbf{R}^T$. This consumption profile must be satisfied in each period, *i.e.*, we have the constraint $p_{\text{load}} = l$.

Thermal load. A thermal load is a single terminal device with power schedule p_{therm} which consists of a heat store (room, cooled water reservoir, refrigerator), with temperature profile $\xi \in \mathbf{R}^T$, which must be kept within minimum and maximum temperature limits, $\xi^{\min} \in \mathbf{R}^T$ and $\xi^{\max} \in \mathbf{R}^T$. The temperature of the heat store evolves as

$$\xi(\tau + 1) = \xi(\tau) + (\mu/c)(\xi^{\text{amb}}(\tau) - \xi(\tau)) - (\eta/c)p_{\text{therm}}(\tau),$$

for $\tau = 1, \ldots, T - 1$ and $\xi(1) = \xi^{\text{init}}$, where $0 \leq p_{\text{therm}} \leq H^{\max}$ is the cooling power consumption profile, $H^{\max} \in \mathbf{R}^T$ is the maximum cooling power, μ is the ambient conduction coefficient, η is the heating/cooling efficiency, c is the heat capacity of the heat store, $\xi^{\text{amb}} \in \mathbf{R}^T$ is the ambient temperature profile, and ξ^{init} is the initial temperature of the heat store. A thermal load has zero cost function.

More sophisticated models [27] include temperature-dependent cooling and heating efficiencies for heat pumps, more complex dynamics of the system whose temperature is being controlled, and additional additive terms in the thermal dynamics, to represent occupancy or other heat sources.

Deferrable load. A deferrable load is a single terminal device with zero cost function that must consume a minimum amount of power over a given interval of time, which is characterized by the constraint $\sum_{\tau=A}^{D} p_{\text{load}}(\tau) \geq E$, where E is the minimum total consumption for the time interval $\tau = A, \ldots, D$. The energy consumption in each time period is constrained by $0 \leq p_{\text{load}} \leq L^{\max}$. In some cases, the load can only be turned on or off in each time period, *i.e.*, $p_{\text{load}}(\tau) \in \{0, L^{\max}\}$ for $\tau = A, \ldots, D$.

Curtailable load. A curtailable load is a single terminal device which does not impose hard constraints on its power requirements, but instead penalizes the shortfall between a desired load profile $l \in \mathbf{R}^T$ and delivered power. In the case of a linear penalty, its cost function is

$$\alpha \mathbf{1}^T (l - p_{\text{load}})_+,$$

where $(z)_+ = \max(0, z)$, $p_{\text{load}} \in \mathbf{R}^T$ is the amount of electricity delivered to the device, $\alpha > 0$ is a penalty parameter, and $\mathbf{1}$ is the vector with all components one. Extensions include time-varying and nonlinear penalties on the energy shortfall.

Electric vehicle. An electric vehicle charging system is an example of a device that combines aspects of a deferable load and a storage device. We model it as a single terminal device with power schedule p_{ev} which has a desired charging profile $c^{\text{des}} \in \mathbf{R}^T$ and can be charged within a time interval $\tau = A, \ldots, D$. To avoid excessive charge cycling, we assume that the electric vehicle battery cannot be discharged back into the grid (in more sophisticated vehicle-to-grid models, this assumption is relaxed), so we have the constraints $0 \leq p_{\text{ev}} \leq C^{\max}$, where $C^{\max} \in \mathbf{R}^T$ is the maximum charging rate. We assume that $c^{\text{des}}(\tau) = 0$ for $\tau = 1, \ldots A-1$, $c^{\text{des}}(\tau) = c^{\text{des}}(D)$ for $\tau = D+1, \ldots, T$, and that the charge level is given by

$$q(\tau) = q^{\text{init}} + \sum_{t=A}^{\tau} p_{\text{ev}}(t),$$

where q^{init} is the initial charge when it is plugged in at time $\tau = A$.

We can model electric vehicle charging as a deferrable load, where we require a given charge level to be achieved at some time. A more realistic model is as a combination of a deferrable and curtailable load, with cost function

$$\alpha \sum_{\tau=A}^{D} (c^{\text{des}}(\tau) - q(\tau))_+,$$

where $\alpha > 0$ is a penalty parameter. Here $c^{\text{des}}(\tau)$ is the desired charge level at time τ, and $c^{\text{des}}(\tau) - q(\tau))_+$ is the shortfall.

4

Convexity

In this chapter we discuss the important issue of convexity, both of devices and the resulting dynamic power flow problem.

4.1 Devices

We call a device *convex* if its objective function is convex. A network is convex if all of its devices are convex. For convex networks, the D-OPF is a convex optimization problem, which means that in principle we can efficiently find a global solution [7]. When the network is not convex, even finding a feasible solution for the D-OPF can become difficult, and finding and certifying a globally optimal solution to the D-OPF is generally intractable. However, special structure in many practical power distribution problems can allow us to guarantee optimality.

In the examples from Chapter 3, the inverter, rectifier, phase shifter, battery, fixed load, thermal load, curtailable load, electric vehicle, and external tie are all convex devices using the constraints and objective functions given. A deferrable load is convex if we drop the constraint that it can only be turned on or off. We discuss the convexity properties of the generator and AC and DC transmission lines next.

4.2 Relaxations

One technique to deal with non-convex networks is to use *convex relaxations*. We use the notation g^{env} to denote the convex envelope [52] of the function g. There are many equivalent definitions for the convex envelope, for example, $g^{\text{env}} = (g^*)^*$, where g^* denotes the convex conjugate of the function g. We can equivalently define g^{env} to be the largest convex lower bound of g. If g is a convex, closed, proper (CCP) function, then $g = g^{\text{env}}$.

The *relaxed dynamic optimal power flow problem* (RD-OPF) is

$$\begin{array}{ll} \text{minimize} & f^{\text{env}}(p, \theta) \\ \text{subject to} & \bar{p} = 0, \quad \tilde{\theta} = 0, \end{array} \tag{4.1}$$

with variables $p : \mathcal{T} \to \mathbf{R}^T$, $\theta : \mathcal{T}^{\text{ac}} \to \mathbf{R}^T$. This is a convex optimization problem, whose optimal value can in principle be computed efficiently, and whose optimal objective value is a lower bound for the optimal objective value of the D-OPF. In some cases, we can guarantee *a priori* that a solution to the RD-OPF will also be a solution to the D-OPF [56, 55] based on a property of the network objective such as monotonicity or unimodularity. Even when the relaxed solution does not satisfy all of the constraints in the unrelaxed problem, it can be used as a starting point to help construct good, local solutions to the unrelaxed problem. The suboptimality of these local solutions can then be bounded by the gap between their network objective and the lower bound provided by the solution to the RD-OPF. If this gap is small for a given local solution, we can guarantee that it is nearly optimal.

Generator. When a generator is modeled as in Chapter 3 and is always powered on, it is a convex device. However, when given the ability to be powered on and off, the generator is no longer convex. In this case, we can relax the generator objective function so that its cost for power production in each time period, given in Figure 4.1, is a convex function. This allows the generator to produce power in the interval $[0, P^{\text{min}}]$.

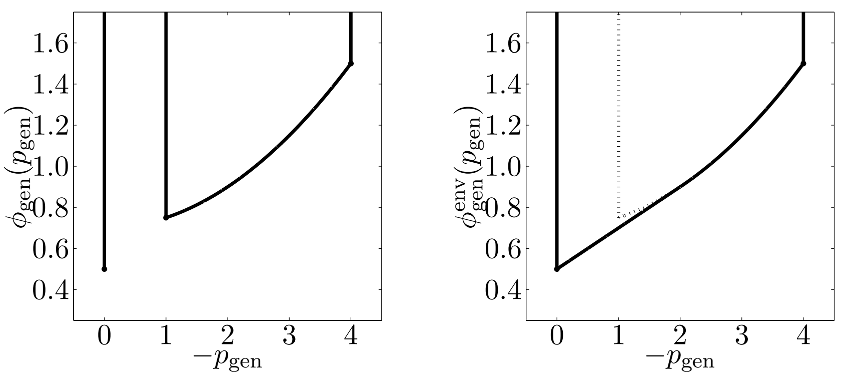

Figure 4.1: *Left*: Cost function for a generator that can be turned off. *Right*: Its convex relaxation.

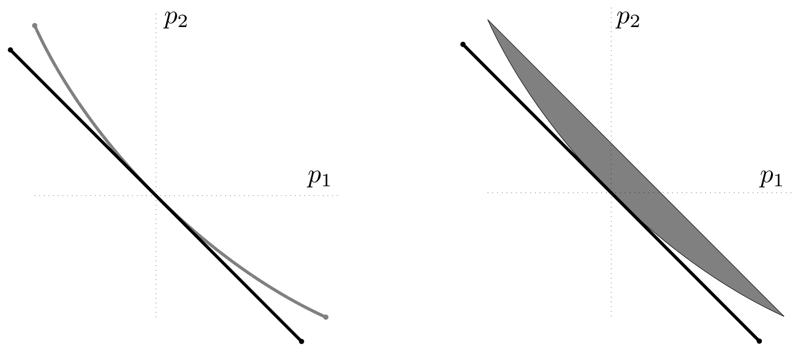

Figure 4.2: *Left*: Feasible sets of a transmission lines with no loss (black) and AC loss (grey). *Right*: Their convex relaxations.

AC and DC transmission lines. In a lossless transmission line (AC or DC), we have $\ell(p_1, p_2) = 0$, and thus the set of feasible power schedules is the line segment

$$L = \{(p_1, p_2) \mid p_1 = -p_2, \quad p_2 \in [-C^{\max}/2, C^{\max}/2]\},$$

as shown in Figure 4.2 in black. When the transmission line has losses, in most cases the loss function ℓ is a convex function of the input and output powers, which leads to a feasible power region like the grey arc in the left part of Figure 4.2.

The feasible set of a relaxed transmission line is given by the convex hull of the original transmission line's constraints. The right side of figure 4.2 shows examples of this for both lossless and lossy transmission lines. Physically, this relaxation gives lossy transmission lines the ability to discard some additional power beyond what is simply lost to heat. Since electricity is generally a valuable commodity in power networks, the transmission lines will generally not throw away any additional power in the optimal solution to the RD-OPF, leading to the power line constraints in the RD-OPF being tight and thus also satisfying the unrelaxed power line constraints in the original D-OPF. As was shown in [40], when the network is a tree, this relaxation is always tight. In addition, when all locational marginal prices are positive and no other non-convexities exist in the network, the tightness of the line constraints in the RD-OPF can be guaranteed in the case of networks that have separate phase shifters on each loop in the networks whose shift parameter can be freely chosen [54].

5

Proximal Message Passing

In this chapter we describe our method for solving D-OPF. We begin by deriving the proximal message passing algorithms assuming that all the device objective functions are convex closed proper (CCP) functions. We then compare the computational and communication requirements of proximal message passing with a centralized solver for the D-OPF. The additional requirements that the functions are closed and proper are technical conditions that are in practice satisfied by any convex function used to model devices. We note that we do not require either finiteness or strict convexity of any device objective function, and that all results apply to networks with arbitrary topologies.

Notation

Whenever we have a set of variables that maps terminals to time periods, $x : \mathcal{T} \to \mathbf{R}^T$ (which we can also associate with a $|\mathcal{T}| \times T$ matrix), we will use the same index, over-line, and tilde notation for the variables x as we do for power schedules p and phase schedules θ. For example, $x_t \in \mathbf{R}^T$ consists of the time period vector of values of x associated with terminal t, $\bar{x}_t = (1/|n|) \sum_{t' \in n} x_{t'}$, where $t \in n$, and $\tilde{x}_t = x_t - \bar{x}_t$, with similar notation for indexing x by devices and nets.

5.1 Derivation

We derive the proximal message passing equations by reformulating the D-OPF using the alternating direction method of multipliers (ADMM) and then simplifying the resulting equations. We refer the reader to [6] for a thorough overview of ADMM.

We first rewrite the D-OPF as

$$\begin{array}{ll} \text{minimize} & \sum_{d\in\mathcal{D}} f_d(p_d,\theta_d) + \sum_{n\in\mathcal{N}}(g_n(z_n)+h_n(\xi_n)) \\ \text{subject to} & p = z, \quad \theta = \xi \end{array} \tag{5.1}$$

with variables $p, z : \mathcal{T} \to \mathbf{R}^T$, and $\theta, \xi : \mathcal{T}^{\text{ac}} \to \mathbf{R}^T$, where $g_n(z_n)$ is the indicator function on the set $\{z_n \mid \bar{z}_n = 0\}$ and $h_n(\xi_n)$ is the indicator function on the set $\{\xi_n \mid \tilde{\xi}_n = 0\}$. We use the notation from [6] and, ignoring a constant, form the augmented Lagrangian

$$\begin{aligned} L_\rho(p,\theta,z,\xi,u,v) = \sum_{d\in\mathcal{D}} f_d(p_d,\theta_d) + \sum_{n\in\mathcal{N}} (g_n(z_n)+h_n(\xi_n)) \\ + (\rho/2)\left(\|p - z + u\|_2^2 + \|\theta - \xi + v\|_2^2\right), \end{aligned} \tag{5.2}$$

with the scaled dual variables $u = y_p/\rho : \mathcal{T} \to \mathbf{R}^T$ and $v = y_\theta/\rho : \mathcal{T}^{\text{ac}} \to \mathbf{R}^T$, which we associate with $|\mathcal{T}| \times T$ and $|\mathcal{T}^{\text{ac}}| \times T$ matrices, respectively, where $y_p : \mathcal{T} \to \mathbf{R}^T$ are the dual variables associated with the power balance constraints and $y_\theta : \mathcal{T}^{\text{ac}} \to \mathbf{R}^T$ are the dual variables associated with the phase consistency constraints. Because devices and nets are each partitions of the terminals, the last two terms of (5.2) can be split across either devices or nets, *i.e.*,

$$\begin{aligned} \|p - z + u\|_2^2 &= \sum_{d\in\mathcal{D}} \|p_d - z_d + u_d\|_2^2 = \sum_{n\in\mathcal{N}} \|p_n - z_n + u_n\|_2^2, \\ \|\theta - \xi + v\|_2^2 &= \sum_{d\in\mathcal{D}} \|\theta_d - \xi_d + v_d\|_2^2 = \sum_{n\in\mathcal{N}} \|\theta_n - \xi_n + v_n\|_2^2. \end{aligned}$$

The resulting ADMM algorithm is then given by the iterations

$$\begin{aligned} (p_d^{k+1}, \theta_d^{k+1}) := \underset{p_d,\theta_d}{\operatorname{argmin}} \Big(f_d(p_d,\theta_d) + (\rho/2)(\|p_d - z_d^k + u_d^k\|_2^2 \\ + \|\theta_d - \xi_d^k + v_d^k\|_2^2)\Big), \quad d \in \mathcal{D}, \end{aligned}$$

$$\begin{array}{rcl}
z_n^{k+1} & := & \underset{z_n}{\operatorname{argmin}} \left(g_n(z_n) + (\rho/2)\|z_n - u_n^k - p_n^{k+1}\|_2^2 \right), \quad n \in \mathcal{N}, \\
\xi_n^{k+1} & := & \underset{\xi_n}{\operatorname{argmin}} \left(h_n(\xi_n) + (\rho/2)\|\xi_n - v_n^k - \theta_n^{k+1}\|_2^2 \right), \quad n \in \mathcal{N}, \\
u_n^{k+1} & := & u_n^k + (p_n^{k+1} - z_n^{k+1}), \quad n \in \mathcal{N}, \\
v_n^{k+1} & := & v_n^k + (\theta_n^{k+1} - \xi_n^{k+1}), \quad n \in \mathcal{N},
\end{array}$$

where the first step is carried out in parallel by all devices, and then the second and third and then fourth and fifth steps are carried out in parallel by all nets.

Since $g_n(z_n)$ and $h_n(\xi_n)$ are simply indicator functions for each net n, the second and third steps of the algorithm can be computed analytically and are given by

$$\begin{array}{rcl}
z_n^{k+1} & := & u_n^k + p_n^{k+1} - \bar{u}_n^k - \bar{p}_n^{k+1}, \\
\xi_n^{k+1} & := & \bar{v}_n^k + \bar{\theta}_n^{k+1},
\end{array}$$

respectively. From the second expression, it is clear that ξ_n^k is simply $|n|$ copies of the same vector for all k.

Substituting these expressions into the u and v updates, the algorithm can be simplified further to yield *proximal message passing*:

1. *Prox schedule updates.*

$$(p_d^{k+1}, \theta_d^{k+1}) := \mathbf{prox}_{f_d,\rho}(p_d^k - \bar{p}_d^k - u_d^k, \bar{\theta}_d^k + \bar{v}_d^{k-1} - v_d^k), \quad d \in \mathcal{D}.$$

2. *Scaled price updates.*

$$\begin{array}{rcl}
u_n^{k+1} & := & u_n^k + \bar{p}_n^{k+1}, \quad n \in \mathcal{N}, \\
v_n^{k+1} & := & \tilde{v}_n^k + \tilde{\theta}_n^{k+1}, \quad n \in \mathcal{N},
\end{array}$$

where the prox function for a function g is given by

$$\mathbf{prox}_{g,\rho}(x) = \underset{y}{\operatorname{argmin}} \left(g(y) + (\rho/2)\|x - y\|_2^2 \right), \tag{5.3}$$

and is guaranteed to exist when g is CCP [52]. If in addition $\bar{v}^0 = 0$ (note that any optimal dual variables $v^\star$ must also satisfy $\bar{v}^\star = 0$), then the v update simplifies to $v_n^{k+1} := v_n^k + \tilde{\theta}_n^{k+1}$ and the second argument of the prox function in the first step simplifies to $\bar{\theta}_d^k - v_d^k$.

The origin of the name 'proximal message passing' should now be clear. In each iteration, every device computes the prox function of its objective function, with an argument that depends on messages passed to it through its terminals by its neighboring nets in the previous iteration ($\bar{p}_d^k$, $\tilde{\theta}_d^k$, u_d^k, and v_d^k). Then, every device passes to its terminals the newly computed power and phase schedules, p_d^{k+1} and θ_d^{k+1}, which are then passed to the terminals' associated nets. Every net computes the prox function of the power balance and phase consistency indicator functions (which corresponds to projecting the power and phase schedules back to feasibility), computes the new average power imbalance, $\bar{p}_n^{k+1}$ and phase residual, $\tilde{\theta}_n^{k+1}$, updates its dual variables, u_n^{k+1} and v_n^{k+1}, and broadcasts these values to its associated terminals' devices. Since $\bar{p}_n^k$ is simply $|n|$ copies of the same vector for all k, all terminals connected to the same net must have the same value for their dual variables associated with power balance throughout the algorithm, *i.e.*, for all values of k, $u_t^k = u_{t'}^k$ whenever $t, t' \in n$ for any $n \in \mathcal{N}$.

As an example, consider the network represented by figures 2.1 and 2.2. The proximal message passing algorithm performs the power and phase schedule updates on the devices (the boxes on the left in Figure 2.2). The devices share the respective power and phase profiles via the terminals, and the nets (the solid boxes on the right) compute the scaled price updates. For any network, the proximal message passing algorithm can be thought of as alternating between devices (on the left) and nets (on the right).

5.2 Convergence

Theory. We now comment on the convergence of proximal message passing. Since proximal message passing is a version of ADMM, all convergence results that hold for ADMM also hold for proximal message passing. In particular, when all devices have CCP objective functions and a feasible solution to the D-OPF exists, the following hold:

1. *Power balance and phase consistency are achieved:* $\bar{p}^k \to 0$ and $\tilde{\theta}^k \to 0$ as $k \to \infty$.

2. *Operation is optimal:* $\sum_{d \in \mathcal{D}} f_d(p_d^k, \theta_d^k) \to f^\star$ as $k \to \infty$.

3. *Optimal prices are found:* $\rho u^k = y_p^k \to y_p^\star$ and $\rho v^k = y_\theta^k \to y_\theta^\star$ as $k \to \infty$.

Here $f^\star$ is the optimal value for the D-OPF, and $y_p^\star$ and $y_\theta^\star$ are optimal dual variables for the power schedule and phase consistency constraints, respectively. The proof of these results (in the more general setting) can be found in [6]. As a result of the third condition, the optimal locational marginal prices $\mathcal{L}^\star$ can be found for each net $n \in \mathcal{N}$ by setting $\mathcal{L}_n^\star = |n|(y_p^\star)_n$.

Stopping criterion. Following [6], we can define primal and dual residuals, which for proximal message passing simplify to

$$r^k = (\bar{p}^k, \tilde{\theta}^k) \quad s^k = \rho((p^k - \bar{p}^k) - (p^{k-1} - \bar{p}^{k-1}), \bar{\theta}^k - \bar{\theta}^{k-1}).$$

We give a simple interpretation of each residual. The primal residual is simply the net power imbalance and phase inconsistency across all nets in the network, which is the original measure of primal feasibility in the D-OPF. The dual residual is equal to the difference between the current and previous iterations of both the difference between power schedules and their average net power as well as the average phase angle on each net. The locational marginal price at each net is determined by the deviation of all associated terminals' power schedule from the average power on that net. As the change in these deviations approaches zero, the corresponding locational marginal prices converge to their optimal values, and all phase angles are consistent across all AC nets.

We can define a simple criterion for terminating proximal message passing when

$$\|r^k\|_2 \le \epsilon^{\text{pri}}, \quad \|s^k\|_2 \le \epsilon^{\text{dual}},$$

where ϵ^{pri} and ϵ^{dual} are, respectively, primal and dual tolerances. We can normalize both of these quantities to network size by the relation

$$\epsilon^{\text{pri}} = \epsilon^{\text{dual}} = \epsilon^{\text{abs}}\sqrt{|\mathcal{T}|T},$$

for some absolute tolerance $\epsilon^{\text{abs}} > 0$.

Choosing a value of ρ. Numerous examples show that the value of ρ can have a strong effect on the rate of convergence of ADMM and proximal message passing. Many good methods for picking ρ in both offline and online fashions are discussed in [6]. We note that unlike other versions of ADMM, the scaling parameter ρ enters very simply into the proximal equations and can thus be modified online without incurring any additional computational penalties, such as having to refactorize a matrix. For devices whose objectives just encode constraints (*i.e.*, only take on the values 0 and $+\infty$), the prox function reduces to projection, and is independent of ρ.

We can modify the proximal message passing algorithm with the addition of a third step

3. *Parameter update and price rescaling.*

$$\begin{aligned}
\rho^{k+1} &:= h(\rho^k, r^k, s^k), \\
u^{k+1} &:= \frac{\rho^k}{\rho^{k+1}} u^{k+1}, \\
v^{k+1} &:= \frac{\rho^k}{\rho^{k+1}} v^{k+1},
\end{aligned}$$

for some function h. We desire to pick an h such that the primal and dual residuals are of similar size throughout the algorithm, *i.e.*, $\rho^k \|r^k\|_2 \approx \|s^k\|_2$ for all k. To accomplish this task, we use a simple proportional-derivative controller to update ρ, choosing h to be

$$h(\rho^k) = \rho^k \exp(\lambda w^k + \mu(w^k - w^{k-1})),$$

where $w^k = \rho^k \|r^k\|_2 / \|s^k\|_2 - 1$ and λ and μ are nonnegative parameters chosen to control the rate of convergence. Typical values of λ and μ are between 10^{-3} and 10^{-1}.

When ρ is updated in such a manner, convergence is sped up in many examples, sometimes dramatically. Although it can be difficult to prove convergence of the resulting algorithm, a standard trick is to assume that ρ is changed in only a finite number of iterations, after which it is held constant for the remainder of the algorithm, thus guaranteeing convergence.

Non-convex case. When one or more of the device objective functions is non-convex, we can no longer guarantee that proximal message passing converges to the optimal value of the D-OPF or even that it converges at all (*i.e.*, reaches a fixed point). Prox functions for non-convex devices must be carefully defined as the set of minimizers in (5.3) is no longer necessarily a singleton. Even then, prox functions of non-convex functions are often intractable to compute.

One solution to these issues is to use proximal message passing to solve the RD-OPF. It is easy to show that $f^{\text{env}}(p, \theta) = \sum_{d \in \mathcal{D}} f_d^{\text{env}}(p_d, \theta_d)$. As a result, we can run proximal message passing using the prox functions of the relaxed device objective functions. Since f_d^{env} is a CCP function for all $d \in \mathcal{D}$, proximal message passing in this case is guaranteed to converge to the optimal value of the RD-OPF and yield the optimal relaxed locational marginal prices.

5.3 Discussion

To compute the proximal messages, devices and nets only require knowledge of who their network neighbors are, the ability to send small vectors of numbers to those neighbors in each iteration, and the ability to store small amounts of state information and efficiently compute prox functions (devices) or projections (nets). As all communication is local and peer-to-peer, proximal message passing supports the ad hoc formation of power networks, such as micro grids, and is self-healing and robust to device failure and unexpected network topology changes.

Due to recent advances in convex optimization [61, 46, 47], many of the prox function calculations that devices must perform can be very efficiently executed at millisecond or microsecond time-scales on inexpensive, embedded processors [30]. Since all devices and all nets can each perform their computations in parallel, the time to execute a single, network wide proximal message passing iteration (ignoring communication overhead) is equal to the sum of the maximum computation time over all devices and the maximum computation time of all nets in the network. As a result, the computation time per iteration is small and essentially independent of the size of the network.

In contrast, solving the D-OPF in a centralized fashion requires complete knowledge of the network topology, sufficient communication bandwidth to centrally aggregate all devices objective function data, and sufficient centralized computational resources to solve the resulting D-OPF. In large, real-world networks, such as the smart grid, all three of these requirements are generally unattainable. Having accurate and timely information on the global connectivity of all devices is infeasible for all but the smallest of dynamic networks. Centrally aggregating all device objective functions would require not only infeasible bandwidth and data storage requirements at the aggregation site, but also the willingness of all devices to expose what could be private and/or proprietary function parameters in their objective functions. Finally, a centralized solution to the D-OPF requires solving an optimization problem with $\Omega(|\mathcal{T}|T)$ variables, which leads to an identical lower bound on the time scaling for a centralized solver, even if problem structure is exploited. As a result, the centralized solver cannot scale to solve the D-OPF on very large networks.

6

Numerical Examples

In this chapter we illustrate the speed and scaling of proximal message passing with a range of numerical examples. In the first two sections, we describe how we generate network instances for our examples. We then describe our implementation, showing how multithreading can exploit problem parallelism and how proximal message passing would scale in a fully peer-to-peer implementation. Lastly, we present our results, and demonstrate how the number of iterations needed for convergence is essentially independent of network size and also significantly decreases when the algorithm is seeded with a reasonable warm-start.

6.1 Network topology

We generate a network instance by first picking the number of nets N. We generate the nets' locations $x_i \in \mathbf{R}^2$, $i = 1, \ldots, N$ by drawing them uniformly at random from $[0, \sqrt{N}]^2$. (These locations will be used to determine network topology.) Next, we introduce transmission lines into the network as follows. We first connect a transmission line between all pairs of nets i and j independently and with probability

$$\gamma(i,j) = \alpha \min(1, d^2/\|x_i - x_j\|_2^2).$$

In this way, when the distance between i and j is smaller than d, they are connected with a fixed probability $\alpha > 0$, and when they are located farther than distance d apart, the probability decays as $1/\|x_i - x_j\|_2^2$. After this process, we add a transmission line between any isolated net and its nearest neighbor. We then introduce transmission lines between distinct connected components by selecting two connected components uniformly at random and then selecting two nets, one inside each component, uniformly at random and connecting them by a transmission line. We continue this process until the network is connected.

For the examples we present, we chose parameter values $d = 0.11$ and $\alpha = 0.8$ as the parameters for generating our network. This results in networks with an average degree of 2.1. Using these parameters, we generated networks with 30 to 100000 nets, which resulted in optimization problems with approximately 10 thousand to 30 million variables.

6.2 Devices

After we generate the network topology described above, we randomly attach a single (one-terminal) device to each net according to the distribution in table 6.1. We also allow the possibility that a net acts as a distributor and has no device attached to it other than transmission lines. About 10% of the transmission lines are DC transmission lines, while the other are AC transmission lines. The models for each device and line in the network are identical to the ones given in Chapter 3, with model parameters chosen in a manner we describe below.

For simplicity, our examples only include networks with the devices listed below. For all devices, the time horizon was chosen to be $T = 96$, corresponding to 15 minute intervals for 24 hour schedules, with the time period $\tau = 1$ corresponding to midnight.

Generator. Generators have the quadratic cost functions given in Chapter 3 and are divided into three types: small, medium, and large. In each case, the generator provides some idling power, so we set $P_{\min} = 0.01$. Small generators have the smallest maximum power output, but the largest ramp rates, while large generators have the largest maximum power output, but the slowest ramp rates. Medium genera-

Device	**Fraction**
None	0.4
Generator	0.4
Curtailable load	0.1
Deferrable load	0.05
Battery	0.05

Table 6.1: Fraction of devices present in the generated networks.

	$P^{\min}$	$P^{\max}$	$R^{\max}$	α	β
Large	0.01	50	3	0.001	0.1
Medium	0.01	20	5	0.005	0.2
Small	0.01	10	10	0.02	1

Table 6.2: Generator parameters.

tors lie in between. Large generators are generally more efficient than small and medium generators which is reflected in their cost function by having smaller values of α and β. Whenever a generator is placed into a network, its type is selected uniformly at random, and its parameters are taken from the appropriate row in table 6.2.

Battery. Parameters for a given instance of a battery are generated by setting $q^{\text{init}} = 0$ and selecting $Q^{\max}$ uniformly at random from the interval $[20, 50]$. The charging and discharging rates are selected to be equal (*i.e.*, $C^{\max} = D^{\max}$) and drawn uniformly at random from the interval $[5, 10]$.

Fixed load. The load profile for a fixed load instance is a sinusoid,

$$l(\tau) = c + a\sin(2\pi(\tau - \phi_0)/T), \quad \tau = 1, \ldots, T,$$

with the amplitude a chosen uniformly at random from the interval $[0.5, 1]$, and the DC term c chosen so that $c = a + u$, where u is chosen uniformly at random from the interval $[0, 0.1]$, which ensures that the load profile remains elementwise positive. The phase shift ϕ_0 is chosen uniformly at random from the interval $[60, 72]$, ensuring that the load profile peaks between the hours of 3pm and 6pm.

Deferrable load. For an instance of a deferrable load, we choose E uniformly at random from the interval $[5, 10]$. The start time index A is chosen uniformly at random from the discrete set $\{1, \dots, (T-9)/2\}$. The end time index D is then chosen uniformly at random over the set $\{A + 9, \dots, T\}$, so that the minimum time window to satisfy the load is 10 time periods (2.5 hours). We set the maximum power so that it requires at least two time periods to satisfy the total energy constraint, *i.e.*, $L^{\max} = 5E/(D - A + 1)$.

Curtailable loads. For an instance of a curtailable load, the desired load l is constant over all time periods with a magnitude chosen uniformly at random from the interval $[5, 15]$. The penalty parameter α is chosen uniformly at random from the interval $[0.1, 0.2]$.

AC transmission line. For an instance of an AC line, we set the voltage magnitude equal to 1 and choose its remaining parameters by first solving the D-OPF with lossless, uncapacitated lines. Using flow values given by the solution to that problem, we set $C^{\max} = \max(30, 10F^{\max})$ for each line, where $F^{\max}$ is equal to the maximum flow (from the lossless solution) along that line over all periods.

We use the loss function for transmission lines with a series admittance $g+ib$ given by (3.1). We choose a maximum phase angle deviation (in degrees) in the interval $[1, 5]$ and a loss of 1 to 3 percent of $C^{\max}$ when transmitting power at maximum capacity. Once the maximum phase angle and the loss are determined, g is chosen to provide the desired loss when operating at maximum phase deviation, while b is chosen so the line operates at maximum capacity when at maximum phase deviation.

DC transmission line. DC transmission lines are handled just like AC transmission lines. We set $R = g/b$, where g and b are chosen using the procedure for the AC transmission line.

6.3 Serial multithreaded implementation

Our D-OPF solver is implemented in C++, with the core proximal message passing equations occupying fewer than 25 lines of C++ (excluding problem setup and class specifications). The code is compiled with `gcc` 4.7.2 on a 32-core, 2.2GHz Intel Xeon processor with 512GB of RAM running the Ubuntu OS. The processor supports hyperthreading, so we have access to 64 independent threads. We used the compiler option `-O3` to leverage full code optimization.

To approximate a fully distributed implementation, we use `gcc`'s implementation of OpenMP (version 3.1) and multithreading to parallelize the computation of the prox functions for the devices. We use 64 threads to solve each example network. Assuming perfect load balancing among the cores, this means that 64 prox functions are being evaluated in parallel. Effectively, we evaluate the prox functions by stepping serially through the devices in blocks of size 64. We do *not* parallelize the computation of the dual updates over the nets since the overhead of spawning threads dominates the vector operations themselves.

The prox functions for fixed loads and curtailable loads are separable over τ and can be computed analytically. For more complex devices, such as a generator, battery, or deferrable load, we compute the prox function using CVXGEN [46]. The prox function for a transmission line is computed by projecting onto the convex hull of the line constraints.

For a given network, we solve the associated D-OPF with an absolute tolerance $\epsilon^{\text{abs}} = 10^{-3}$. This translates to three digits of accuracy in the solution. The CVXGEN solvers used to evaluate the prox operators for some devices have an absolute tolerance of 10^{-8}. We set $\rho = 1$.

6.4 Peer-to-peer implementation

We have not yet created a fully peer-to-peer, bulk synchronous parallel [60, 45] implementation of proximal message passing, but have carefully tracked solve times in our serial implementation in order to facilitate a first order analysis of such a system. In a peer-to-peer implementation, the prox schedule updates occur in parallel across all devices followed by (scaled) price updates occurring in parallel across all nets. As previously

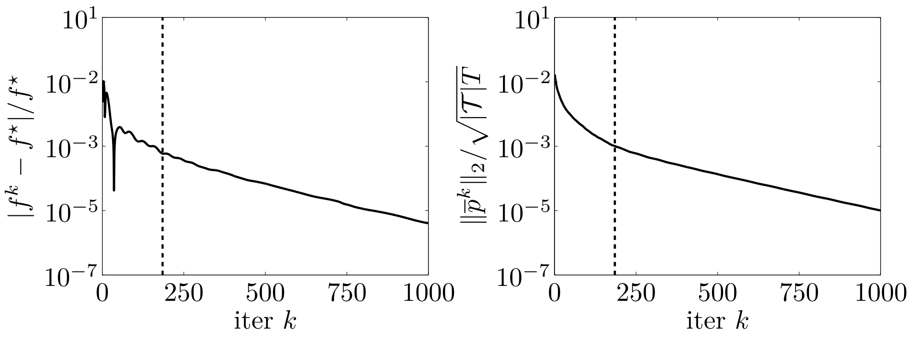

Figure 6.1: The relative suboptimality (*left*) and primal infeasibility (*right*) of proximal message passing on a network instance with $N = 3000$ nets (1 million variables). The dashed line shows when the stopping criterion is satisfied.

mentioned, the computation time per iteration is thus the maximum time, over all devices, to evaluate the prox function of their objective, added to the maximum time across all nets to project their terminal schedules back to feasibility and update their existing price vectors. Since evaluating the prox function for some devices requires solving a convex optimization problem, whereas the price updates only require a small number of vector operations that can be performed as a handful of SIMD instructions, the compute time for the price updates is negligible in comparison to the prox schedule updates. The determining factor in solve time, then, is in evaluating the prox functions for the schedule updates. In our examples, the *maximum* time taken to evaluate any prox function is 1 ms.

6.5 Results

We first consider a single example: a network instance with $N = 3000$ (1 million variables). Figure 6.1 shows that after fewer than 200 iterations of proximal message passing, both the relative suboptimality as well as the average net power imbalance and average phase inconsistency are both less than 10^{-3}. The convergence rates for other network instances over the range of sizes we simulated are similar.

In Figure 6.2, we present average timing results for solving the D-OPF for a family of examples, using our serial implementation, with networks of size $N = 30, 100, 300, 1000, 3000, 10000, 30000$, and 100000. For each network size, we generated and solved 10 network instances to compute average solve times and confidence intervals around those averages. The times were modeled with a log-normal distribution. For network instances with $N = 100000$ nets, the problem has over 30 million variables, which we solve serially using proximal message passing in 5 minutes on average. By fitting a line to the proximal message passing runtimes, we find that our parallel implementation empirically scales as $O(N^{0.996})$, *i.e.*, solve time is linear in problem size.

For a peer-to-peer implementation, the runtime of proximal message passing should be essentially constant, and in particular independent of the size of the network. To solve a problem with $N = 100000$ nets (30 million variables) with approximately 200 iterations of our algorithm then takes only 200 ms. In practice, the actual solve time would clearly be dominated by network communication latencies and actual runtime performance will be determined by how quickly and reliably packets can be delivered [34]. As a result, in a true peer-to-peer implementation, a negligible amount of time is actually spent on computation. However, it goes without saying that many other issues must be addressed with a peer-to-peer protocol, including handling network delays and security.

Figure 6.2 shows *cold start* runtimes for solving the D-OPF. If we have good estimates of the power and phase schedules and dual variables for each terminal, we can use them to *warm start* our D-OPF solver. To show the effect, we randomly convert 5% of the devices into fixed loads and solve a specific instance with $N = 3000$ nets (1 million variables). Let K^{cold} to be the number of iterations needed to solve an instance of this problem. We then uniformly scale the load profiles of each device by separate and independent lognormal random variables. The new profiles, $\hat{l}$, are obtained from the original profiles l via

$$\hat{l} = l \exp(\sigma X),$$

where $X \sim \mathcal{N}(0, 1)$, and $\sigma > 0$ is given. Using the original solution to warm start our solver, we solve the perturbed problem and report the number of iterations K^{warm} needed. Figure 6.3 shows the ratio $K^{\text{warm}}/K^{\text{cold}}$ as we vary σ, showing the significant savings possible with warm-starting even under relatively large perturbations.

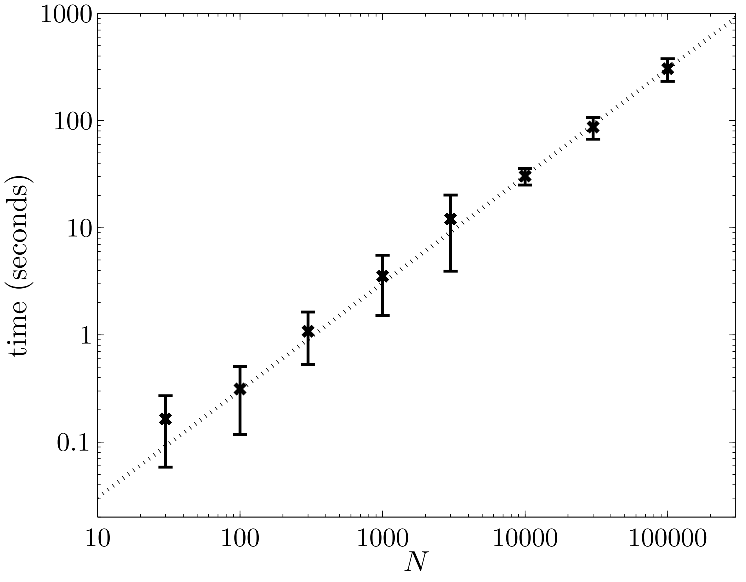

Figure 6.2: Average execution times for a family of networks on 64 threads. Error bars show 95% confidence bounds. The dotted line shows the least-squares fit to the data, resulting in a scaling exponent of 0.996.

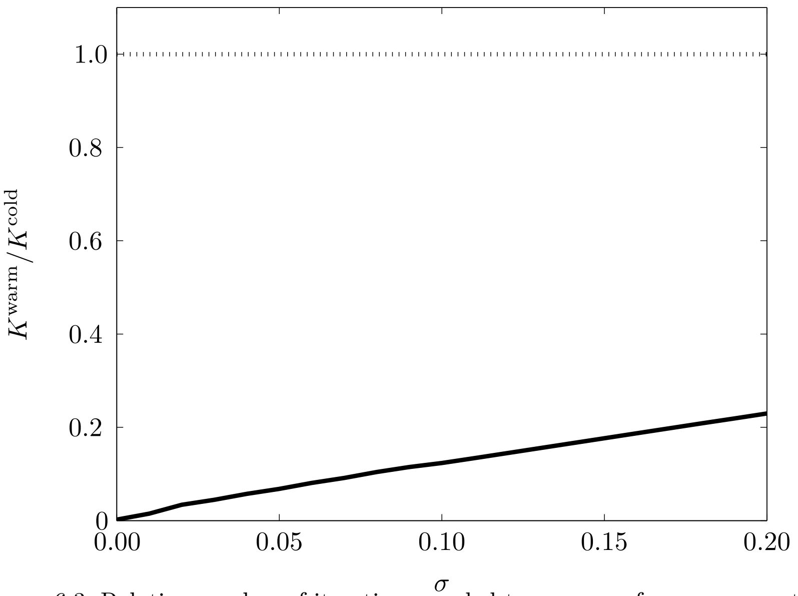

Figure 6.3: Relative number of iterations needed to converge from a warm start for various perturbations of load profiles compared to original number of iterations.

7

Extensions

Here, we give some possible extensions of our model and method.

7.1 Closed-loop control

So far, we have considered only a static energy planning problem, where each device on the network plans power and phase schedules extending T steps into the future and then executes all T steps. This ‘open loop’ control can fail spectacularly, since it will not adjust its schedules in response to external disturbances that were unknown at the original time the schedules were computed.

To alleviate this problem, we propose the use of receding horizon control (RHC) [43, 3, 47] for dynamic network operation. In RHC, at each time step τ, we determine a plan of action over a fixed time horizon T into the future by solving the D-OPF using proximal message passing. The first step of all schedules is then executed, at which point the entire process is repeated, incorporating new measurements, external data, and predictions that have become available.

RHC has been successfully applied in many areas, including chemical process control [50], supply chain management [11], stochastic control in economics and finance [24, 58], and energy storage system operation [37]. While RHC is in general not an optimal controller, it has been shown to achieve good performance in many different domains.

RHC is ideally suited for use with proximal message passing. First, when one time step is executed, we can warm start the next round of proximal message passing with the $T - 1$ schedules and dual variables that were computed, but not otherwise executed, for each device and each net in the previous iteration of RHC. As was shown in the previous section, this can dramatically speed up computation and allow for RHC to operate network-wide at fraction of a second rates. In addition, RHC does not require any stochastic or formal model of future prediction uncertainty. While statistical predictions can be used if they are available, predictions from other sources, such as analysts or markets, are just as simple to integrate into RHC, and are also much easier to come by in many real-world scenarios.

Perhaps the most important synergy between proximal message passing and RHC is that the predictions used by each device need only concern that one device and do not need to include any estimates concerning other devices. This allows for devices to each use their own form of prediction without worrying about what other devices exist or what form of prediction they are using (*e.g.*, even if one generator uses statistical predictions, other devices need not).

The reason for this is that proximal message passing integrates all the device predictions into the final solution — just as they would have been in a centralized solution — but does so through the power and phase schedules and prices that are shared between neighboring devices. In this way, for example, a generator only needs to estimate its cost of producing power at different levels over the time horizon T. It *does not* need to predict any demand itself, as those predictions are passed to it in the form of the power schedule and price messages it receives from its neighbors. Similarly, loads only need to forecast their own future demand and utility of power in each time period. Loads *do not* need to explicitly predict future prices, as those are the result of running proximal message passing over the network.

7.2 Security constrained optimal power flow

In the SC-OPF problem, we determine a set of contingency plans for devices connected on a power network, which tell us the power flows and phase angles each device will operate at under nominal system operation, as well as in a set of specified contingencies or scenarios. The contingencies can correspond, say, to failure or degraded operation of a transmission line or generator, or a substantial change in a load. In each scenario the powers and phases must satisfy the network equations (taking into account any failures for that scenario), and they are also constrained in various ways across the scenarios. Generators and loads, for example, might be constrained to not change their power generation or consumption in any non-nominal scenario.

As a variation on this, we can allow such devices to modify their powers from the nominal operation values, but only over some range or for a set amount of time. The goal is to minimize a composite cost function that includes the cost (and constraints) of nominal operation, as well as those associated with operation in any of the scenarios. Proximal message passing allows us to parallelize the computation of many different (and coupled) scenarios across each device while maintaining decentralized communication across the network.

7.3 Hierarchical models and virtualized devices

The power grid has a natural hierarchy, with generation and transmission occurring at the highest level and residential consumption and distribution occurring at the most granular. Proximal message passing can be easily extended into hierarchical interactions by scheduling messages on different time scales and between systems at similar levels of the hierarchy [10]. By aggregating multiple devices into a virtual device (which themselves may be further aggregated into another virtual device), our framework naturally allows for the formation of composite entities such as virtual power plants and demand response aggregators.

Let $D \subseteq \mathcal{D}$ be a group of devices that are aggregated into a virtual device, which we will also refer to as 'D'. We use the notation that terminal $t \in D$ if there exists a device $d \in D$ such that $t \in d$. The set of terminals $\{t \mid t \in D\}$ can be partitioned into two sets, those terminals whose associated net's terminals are all associated with a device which is part of D, and those who are not. These two sets can be though of as those terminal in D which are purely 'internal' to D, and those terminals which are not, as shown in Figure 7.1. These two sets are given by

$$\begin{aligned} D_{\text{in}} &= \{t \in D \mid \forall t' \in n_t, t' \in D\}, \\ D_{\text{out}} &= \{t \in D \mid \exists t' \in n_t, t' \notin D\}, \end{aligned}$$

respectively, where n_t is defined to be the net associated with terminal t (*i.e.*, $t \in n_t$). We let $(p_{\text{in}}, \theta_{\text{in}})$ and $(p_{\text{out}}, \theta_{\text{out}})$ denote the power and phase schedules associated with the terminals in the sets D_{in} and D_{out}, respectively. Since the power and phase schedules $(p_{\text{in}}, \theta_{\text{in}})$ never directly leave the virtual device, they can be considered as internal variables for the virtual device.

The objective function of the virtual device, $f_D(p_{\text{out}}, \theta_{\text{out}})$, is given by the solution to the following optimization problem

$$\begin{array}{ll} \text{minimize} & \sum_{d \in D} f_d(p_d, \theta_d) \\ \text{subject to} & \bar{p}_{\text{in}} = 0, \quad \tilde{\theta}_{\text{in}} = 0, \end{array}$$

with variables p_d and θ_d for $d \in D$. A sufficient condition for $f_D(p_{\text{out}}, \theta_{\text{out}})$ being a convex function is that all of the virtual device's constituent devices' objective functions are convex functions [7].

By recursively applying proximal message passing at each level of the aggregation hierarchy, we can compute the objective functions for each virtual device. This process can be continued down to the individual device level, at which point the device must compute the prox function for its own objective function as the base case.

These models allow for the computations necessary to operate a smart grid network to be virtualized since the computations specific to each device do not necessarily need to be carried out on the device itself, but can be computed elsewhere (*e.g.*, the servers of a virtual power plant, centrally by an independent system operator, ...), and then transmitted to the device for execution. As a result, hierarchical modeling allows one to smoothly interpolate from completely

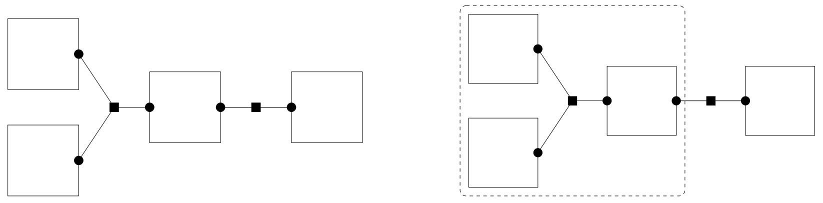

Figure 7.1: *Left:* A simple network with four devices and two nets. *Right:* A hierarchical representation with only 2 devices at the highest level. All terminals connected to the left-most net are internal to the virtual device.

centralized operation of the grid (*i.e.*, all objectives and constraints are gathered in a single location and solved), to a completely decentralized architecture where all communication is peer to peer. At all scales, proximal message passing offers all decision making entities an efficient method to compute optimal power and phase schedules for the devices under their control, while maintaining privacy of their devices' objective functions and constraints.

7.4 Local stopping criteria and ρ updates

The stopping criterion and ρ update method in Chapter 5 currently require global device coordination (via the global primal and dual residuals each iteration). These could be computed in a decentralized fashion by gossip algorithms [53], but this could require many rounds of gossip in between each iteration of proximal message passing, significantly increasing runtime. We are investigating methods to let individual devices or terminals independently choose both the stopping criterion and different values of ρ based only on local information such as the primal and dual residuals of a device and its neighbors.

For dynamic operation another approach is to run proximal message passing continuously, with no stopping criteria. In this mode, devices and nets would exchange messages with each other indefinitely and execute the first step of their schedules at given times (*i.e.*, gate closure), at which point they shift their moving horizon forward one time step and continue to exchange messages.

8

Conclusion

We have presented a fully decentralized method for dynamic network energy management based on message passing between devices. Proximal message passing is simple and highly extensible, relying solely on peer to peer communication between devices that exchange energy. When the resulting network optimization problem is convex, proximal message passing converges to the optimal value and gives optimal power and phase schedules and locational marginal prices. We have presented a parallel implementation that shows the time per iteration and the number of iterations needed for convergence of proximal message passing are essentially independent of the size of the network. As a result, proximal message passing can scale to extremely large networks with almost no increase in solve time.

Acknowledgments

The authors thank Yang Wang and Neal Parikh for extensive discussions on the problem formulation as well as ADMM methods; Yang Wang, Brendan O'Donoghue, Haizi Yu, Haitham Hindi, and Mikael Johansson for discussions on optimal ρ selection and for help with the ρ update method; Steven Low for discussions about end-point based control; and Ed Cazalet, Ram Rajagopal, Ross Baldick, David Chassin, Marija Ilic, Trudie Wang, and Jonathan Yedidia for many helpful comments. We would like to thank Marija Ilic, Le Xie, and Boris Defourny for pointing us to DYNMONDS and other earlier Lagrangian approaches. We are indebted to Misha Chertkov, whose questions on an early version of this paper prodded us to make the concept of AC and DC terminals explicit. Finally, we thank Warren Powell and Hugo Simao for encouraging us to release implementations of these methods.

This research was supported in part by Precourt 1140458-1-WPIAE, by AFOSR grant FA9550-09-1-0704, by NASA grant NNX07AEIIA, and by the DARPA XDATA grant FA8750-12-2-0306.

After this paper was submitted, we became aware of [31] and [32], which apply ADMM to power networks for the purpose of robust state estimation. Our paper is independent of their efforts.

References

[1] R. Baldick, *Applied Optimization: Formulation and Algorithms for Engineering Systems.* Cambridge University Press, 2006.

[2] S. Barman, X. Liu, S. Draper, and B. Recht, “Decomposition methods for large scale LP decoding,” *Submitted, IEEE Transactions on Information Theory*, 2012.

[3] A. Bemporad, “Model predictive control design: New trends and tools,” in *Proceedings of 45th IEEE Conference on Decision and Control*, pp. 6678–6683, 2006.

[4] A. Bergen and V. Vittal, *Power Systems Analysis.* Prentice Hall, 1999.

[5] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods.* Academic Press, 1982.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, 2011.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[8] J. Carpentier, “Contribution to the economic dispatch problem,” *Bull. Soc. Francaise Elect.*, vol. 3, no. 8, pp. 431–447, 1962.

[9] E. A. Chakrabortty and M. D. Ilic, *Control & Optimization Methods for Electric Smart Grids.* Springer, 2012.

[10] M. Chiang, S. Low, A. Calderbank, and J. Doyle, “Layering as optimization decomposition: A mathematical theory of network architectures,” *Proc. of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[11] E. G. Cho, K. A. Thoney, T. J. Hodgson, and R. E. King, “Supply chain planning: Rolling horizon scheduling of multi-factory supply chains,” in *Proceedings of the 35th conference on Winter simulation: driving innovation*, pp. 1409–1416, 2003.

[12] B. H. Chowdhury and S. Rahman, “A review of recent advances in economic dispatch,” *IEEE Transactions on Power Systems*, vol. 5, no. 4, pp. 1248–1259, 1990.

[13] A. O. Converse, “Seasonal energy storage in a renewable energy system,” *Proceedings of the IEEE*, vol. 100, pp. 401–409, Feb 2012.

[14] G. B. Dantzig and P. Wolfe, “Decomposition principle for linear programs,” *Operations Research*, vol. 8, pp. 101–111, 1960.

[15] J. Eckstein, “Parallel alternating direction multiplier decomposition of convex programs,” *Journal of Optimization Theory and Applications*, vol. 80, no. 1, pp. 39–62, 1994.

[16] J. H. Eto and R. J. Thomas, “Computational needs for the next generation electric grid,” in *Department of Energy*, 2011. http://certs.lbl.gov/pdf/lbnl-5105e.pdf.

[17] H. Everett, “Generalized Lagrange multiplier method for solving problems of optimum allocation of resources,” *Operations Research*, vol. 11, no. 3, pp. 399–417, 1963.

[18] A. Eydeland and K. Wolyniec, *Energy and Power Risk Management: New Developments in Modeling, Pricing and Hedging.* Wiley, 2002.

[19] G. Forney, “Codes of graphs: Normal realizations,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 520–548, 2001.

[20] D. Gabay and B. Mercier, “A dual algorithm for the solution of nonlinear variational problems via finite element approximations,” *Computers and Mathematics with Applications*, vol. 2, pp. 17–40, 1976.

[21] R. Glowinski and A. Marrocco, “Sur l’approximation, par elements finis d’ordre un, et la resolution, par penalisation-dualité, d’une classe de problems de Dirichlet non-lineares,” *Revue Française d’Automatique, Informatique, et Recherche Opérationelle*, vol. 9, pp. 41–76, 1975.

[22] H. H. Happ, “Optimal power dispatch — a comprehensive survey,” *IEEE Transactions on Power Apparatus and Systems*, vol. 96, no. 3, pp. 841–854, 1977.

[23] E. K. Hart, E. D. Stoutenburg, and M. Z. Jacobson, “The potential of intermittent renewables to meet electric power demand: current methods and emerging analytical techniques,” *Proceedings of the IEEE*, vol. 100, pp. 322–334, Feb 2012.

[24] F. Herzog, *Strategic Portfolio Management for Long-Term Investments: An Optimal Control Approach.* PhD thesis, ETH, Zurich, 2005.

[25] M. R. Hestenes, “Multiplier and gradient methods,” *Journal of Optimization Theory and Applications*, vol. 4, pp. 302–320, 1969.

[26] K. Heussen, S. Koch, A. Ulbig, and G. Andersson, “Energy storage in power system operation: The power nodes modeling framework,” in *Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES*, pp. 1–8, Oct 2010.

[27] T. Hovgaard, L. Larsen, J. Jorgensen, and S. Boyd, “Nonconvex model predictive control for commercial refrigeration,” *International Journal of Control*, vol. 86, no. 8, pp. 1349–1366, 2013.

[28] M. Ilic, L. Xie, and J.-Y. Joo, "Efficient coordination of wind power and price-responsive demand — Part I: Theoretical foundations," *IEEE Transactions on Power Systems*, vol. 26, pp. 1875–1884, Nov 2011.

[29] M. Ilic, L. Xie, and J.-Y. Joo, "Efficient coordination of wind power and price-responsive demand—part ii: Case studies," *IEEE Transactions on Power Systems*, vol. 26, pp. 1885–1893, Nov 2011.

[30] J. L. Jerez, P. J. Goulart, S. Richter, G. A. Constantinides, E. C. Kerrigan, and M. Morari, "Embedded online optimization for model predictive control at megahertz rates," *Submitted, IEEE Transactions on Automatic Control*, 2013.

[31] V. Kekatos and G. Giannakis, "Joint power system state estimation and breaker status identification," in *Proceedings of the 44th North American Power Symposium*, 2012.

[32] V. Kekatos and G. Giannakis, "Distributed robust power system state estimation," *IEEE Transactions on Power Systems*, 2013.

[33] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[34] A. Kiana and A. Annaswamy, "Wholesale energy market in a smart grid: A discrete-time model and the impact of delays," in *Control and Optimization Methods for Electric Smart Grids*, (A. Chakrabortty and M. Ilic, eds.), pp. 87–110, Springer US, 2012.

[35] B. H. Kim and R. Baldick, "Coarse-grained distributed optimal power flow," *IEEE Transactions on Power Systems*, vol. 12, no. 2, pp. 932–939, 1997.

[36] B. H. Kim and R. Baldick, "A comparison of distributed optimal power flow algorithms," *IEEE Transactions on Power Systems*, vol. 15, no. 2, pp. 599–604, 2000.

[37] M. Kraning, Y. Wang, E. Akuiyibo, and S. Boyd, "Operation and configuration of a storage portfolio via convex optimization," in *Proceedings of the 18th IFAC World Congress*, pp. 10487–10492, 2011.

[38] A. Lam, B. Zhang, and D. Tse, "Distributed algorithms for optimal power flow problem," `http://arxiv.org/abs/1109.5229`, 2011.

[39] J. Lavaei and S. Low, "Zero duality gap in optimal power flow problem," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 92–107, 2012.

[40] J. Lavaei, D. Tse, and B. Zhang, "Geometry of power flows in tree networks," *IEEE Power & Energy Society General Meeting*, 2012.

[41] J. Liang, G. K. Venayagamoorthy, and R. G. Harley, "Wide-area measurement based dynamic stochastic optimal power flow control for smart grids with high variability and uncertainty," *IEEE Transactions on Smart Grid*, vol. 3, pp. 59–69, 2012.

[42] S. H. Low, L. Peterson, and L. Wang, "Understanding tcp vegas: a duality model," in *Proceedings of the 2001 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, (New York, NY, USA), pp. 226–235, ACM, 2001.

[43] J. Maciejowski, *Predictive Control with Constraints.* Prentice Hall, 2002.

[44] S. H. Madaeni, R. Sioshansi, and P. Denholm, "How thermal energy storage enhances the economic viability of concentrating solar power," *Proceedings of the IEEE*, vol. 100, pp. 335–347, Feb 2012.

[45] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: A system for large-scale graph processing," in *Proceedings of the 2010 International Conference on Management of Data*, pp. 135–146, 2010.

[46] J. Mattingley and S. Boyd, "CVXGEN: Automatic convex optimization code generation," `http://cvxgen.com/`, 2012.

[47] J. Mattingley, Y. Wang, and S. Boyd, "Receding horizon control: Automatic generation of high-speed solvers," *IEEE Control Systems Magazine*, vol. 31, pp. 52–65, 2011.

[48] W. F. Pickard, "The history, present state, and future prospects of underground pumped hydro for massive energy storage," *Proceedings of the IEEE*, vol. 100, pp. 473–483, Feb 2012.

[49] M. J. D. Powell, "A method for nonlinear constraints in minimization problems," in *Optimization*, (R. Fletcher, ed.), Academic Press, 1969.

[50] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764, 2003.

[51] P. Ravikumar, A. Agarwal, and M. J. Wainwright, "Message-passing for graph-structured linear programs: Proximal methods and rounding schemes," *Journal of Machine Learning Research*, vol. 11, pp. 1043–1080, 2010.

[52] R. T. Rockafellar, *Convex Analysis.* Princeton University Press, 1970.

[53] D. Shah, "Gossip algorithms," *Foundations and Trends in Networking*, vol. 3, no. 2, pp. 1–125, 2008.

[54] S. Sojoudi and J. Lavaei, "Physics of power networks makes hard problems easy to solve," *To appear, IEEE Power & Energy Society General Meeting*, 2012.

[55] S. Sojoudi and J. Lavaei, "Convexification of generalized network flow problem with application to power systems," Preprint available at `http://www.ee.columbia.edu/~lavaei/Generalized_Net_Flow.pdf`, 2013.

[56] S. Sojoudi and J. Lavaei, "Semidefinite relaxation for nonlinear optimization over graphs with application to power systems," Preprint available at `http://www.ee.columbia.edu/~lavaei/Opt_Over_Graph.pdf`, 2013.

[57] N. Taheri, R. Entriken, and Y. Ye, "A dynamic algorithm for facilitated charging of plug-in electric vehicles," *arxiv:1112:0697*, 2011.

[58] K. T. Talluri and G. J. V. Ryzin, *The Theory and Practice of Revenue Management.* Springer, 2004.

[59] K. Turitsyn, P. Sulc, S. Backhaus, and M. Chertkov, "Options for control of reactive power by distributed photovoltaic generators," *Proceedings of the IEEE*, vol. 99, pp. 1063–1073, 2011.

[60] L. G. Valiant, "A bridging model for parallel computation," *Communications of the ACM*, vol. 33, no. 8, p. 111, 1990.

[61] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, pp. 267–278, 2010.

[62] J. Zhu, *Optimization of Power System Operation*. Wiley-IEEE Press, 2009.