# A Parallel Primal-Dual Interior-Point Method for DC Optimal Power Flow

Ariana Minot, Yue M. Lu, and Na Li
School of Engineering and Applied Sciences
Harvard University, Cambridge, MA, USA
{aminot,yuelu,nali}@seas.harvard.edu

***Abstract*—Optimal power flow (OPF) seeks the operation point of a power system that maximizes a given objective function and satisfies operational and physical constraints. Given real-time operating conditions and the large scale of the power system, it is demanding to develop algorithms that allow for OPF to be decomposed and efficiently solved on parallel computing systems. In our work, we develop a parallel algorithm for applying the primal-dual interior point method to solve OPF. The primal-dual interior point method has a much faster convergence rate than gradient-based algorithms but requires solving a series of large, sparse linear systems. We design efficient parallelized and iterative methods to solve such linear systems which utilize the sparsity structure of the system matrix.**

***Index Terms*—Optimal power flow, distributed optimization, interior-point methods.**

## I. Introduction

Optimal power flow (OPF) solves for the values of power generations and loads that optimize a particular objective, such as minimizing operation costs and maximizing user utility, subject to physical and operational constraints. In order for the grid operator to respond immediately and efficiently to fluctuations in load and generation, especially when there is a large penetration of renewable energy, a solution to OPF needs to be available on the order of seconds or minutes. This demands new approaches to solving OPF, which allow for efficient use of parallel computing systems. Such approaches provide a way for different control areas to coordinate with limited and local communication.

Previous parallel approaches to the OPF problem are mostly first-order (*i.e.* gradient-based) methods. As an example, in [1], the authors develop an algorithm based on primal and dual decomposition techniques as a distributed solution to OPF. In [2]–[4], the ADMM algorithm is used as a distributed semidefinite programming solver, which is shown to have improved convergence results compared to those in [1]. However, ADMM-based approaches usually achieve a sublinear or linear convergence rate [5]. In [6], a fully distributed algorithm for OPF is proposed based on proximal message-passing algorithm, a version of ADMM.

On the other hand, primal-dual interior point (PDIP) methods, which are Newton-based approaches, demonstrate fast convergence behavior on various optimization problems, including OPF [7], [8]. In [9], a decentralized Newton-based approach based on the unlimited point algorithm is presented. The unlimited point algorithm introduces slack variables whereas PDIP methods successively solve a series of problems parameterized by an increasing barrier parameter. In [9], each control area solves a local optimization problem. In contrast, we decompose the global optimization problem across control areas.

PDIP methods usually achieve a superlinear convergence rate [10]. However, the primal-dual interior point method requires solving a large, sparse linear system at each iteration to calculate the search direction for updating the current estimate. For large systems, such as real-scale power networks with thousands of buses, direct inversion can be prohibitive in terms of computation.

To solve these series of large, sparse systems, we propose iterative, distributed methods based on matrix-splitting [11]. We design the matrix-splitting in order to exploit the sparsity and the topology that are inherent to the problem and the power network structure. Iteratively solving for the search direction introduces a set of inner iterations at each outer PDIP iteration. A centralized controller is used to calculate the step length and termination criteria. In this work, we consider the linearized DC power flow model. The method designed in this paper can be extended to the AC power flow model, which will be considered in future work. The main contributions of this work include:

1) A parallel PDIP method for DC-OPF is proposed. Its convergence rate is demonstrated on several systems and shown to scale favorably as the network size grows.
2) Local processing is used to limit communication requirements with the central processor. The search direction, which is the main computational burden, is calculated in a fully distributed way.

*Notations:* We use $v_k$ to denote the $k$th entry of a vector $\boldsymbol{v}$. The $(i, j)$th entry of a matrix $\boldsymbol{M}$ is given by $M_{ij}$. The transpose of a vector or matrix $\boldsymbol{X}$ is denoted by $\boldsymbol{X}^T$. We use $\boldsymbol{M} \succ 0$ to denote that $\boldsymbol{M}$ is positive definite. For a vector-valued function

This research is supported by the Blue Waters sustained-petascale computing project (which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois) and the Harvard Center for Green Buildings and Cities. *Corresponding author: Na Li, nali@seas.harvard.edu.*

$\boldsymbol{f} : \mathbb{R}^m \to \mathbb{R}^n$,

$$D\boldsymbol{f}(\boldsymbol{x}) \overset{\text{def}}{=} \begin{bmatrix} \nabla f_1(\boldsymbol{x})^T \\ \vdots \\ \nabla f_n(\boldsymbol{x})^T \end{bmatrix}.$$

## II. Problem Statement

### A. DC Power Flow Model

We consider a multi-area interconnected power network, denoted by an undirected graph $(\mathcal{N}, \mathcal{L})$ with a set $\mathcal{N} \overset{\text{def}}{=} \{1, 2, \ldots, n\}$ of buses and a set $\mathcal{L} \subseteq \mathcal{N} \times \mathcal{N}$ of transmission lines connecting the buses. Let the total number of transmission lines be $|\mathcal{L}| = L$. Let the power injection profile, $\boldsymbol{p} \in \mathbb{R}^n$, be the value of the net real power injection at every bus. A positive power injection denotes a generation bus, whereas a negative power injection denotes a demand bus. Let $\boldsymbol{f}$ denote the real power flow along every branch in the power network $\boldsymbol{f} \in \mathbb{R}^L$. The real power injection at bus $i$ is given by

$$p_i = \sum_{j \in \mathcal{N}_i} f_{l(i,j)}, \tag{1}$$

where $f_{l(i,j)}$ is the real power flow along the line $l(i,j)$ connecting buses $i$ and $j$. The quantity $f_{l(i,j)}$ is positive if the direction of the flow is towards bus $j$ and negative if it is towards bus $i$. We use the following convention: a line between buses $l(i,j)$ is an ordered pair with $i < j$. The relation between power flows and injections can be written in matrix form

$$\boldsymbol{B}\boldsymbol{f} = \boldsymbol{p}, \tag{2}$$

where

$$B_{il} = \begin{cases} 1 & \text{if power flows on line } l \text{ from bus } i \\ -1 & \text{if power flows on line } l \text{ to bus } i \\ 0 & \text{if bus } i \text{ and line } l \text{ are not connected} \end{cases}$$

The matrix $\boldsymbol{B}$ encodes the network structure.

### B. Mathematical Program Formulation

The optimization problem we study in this paper is to minimize cost subject to line capacity and power injection constraints:

$$\begin{aligned} \min\ & f_0(\boldsymbol{x}) \\ s.t.\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0} \\ & \boldsymbol{g}(\boldsymbol{x}) < \boldsymbol{0}. \end{aligned} \tag{3}$$

The decision variables $\boldsymbol{x}$ are the real power injections, $\boldsymbol{p}$, and real power flows, $\boldsymbol{f}$. We consider a differentiable, convex objective function $f_0(\boldsymbol{x})$. The equality constraints enforcing the relation between power injections and power flows, $\boldsymbol{B}\boldsymbol{f} = \boldsymbol{p}$, can be re-written as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$, where

$$\boldsymbol{A} \overset{\text{def}}{=} [-\boldsymbol{I} \quad \boldsymbol{B}] \tag{4}$$

$$\boldsymbol{x} \overset{\text{def}}{=} [\boldsymbol{p} \quad \boldsymbol{f}]^T \tag{5}$$

The inequality constraints $\boldsymbol{g}(\boldsymbol{x}) < \boldsymbol{0}$ can be written as

$$\boldsymbol{g}(\boldsymbol{x}) \overset{\text{def}}{=} \begin{bmatrix} \boldsymbol{p} - \bar{\boldsymbol{p}} \\ \underline{\boldsymbol{p}} - \boldsymbol{p} \\ \boldsymbol{f} - \bar{\boldsymbol{f}} \\ \underline{\boldsymbol{f}} - \boldsymbol{f} \end{bmatrix} < \boldsymbol{0}, \tag{6}$$

where $\bar{\boldsymbol{p}}$ and $\underline{\boldsymbol{p}}$ are constants representing the upper and lower power injection capacities, respectively. Similarily, $\bar{\boldsymbol{f}}$ and $\underline{\boldsymbol{f}}$ are constants representing the upper and lower line capacities, respectively. The lower bound on the power flow ensures that it cannot be too large in either direction along a line. A key feature of this formulation is that the objective function and inequality constraints are separable. In summary, we have the following formulation of DC-OPF as a linear equality and linear inequality constrained problem,

$$\begin{aligned} & \min\ f_0(\boldsymbol{p}) \\ & s.t. [-\boldsymbol{I} \quad \boldsymbol{B}] \left[\begin{smallmatrix} \boldsymbol{p} \\ \boldsymbol{f} \end{smallmatrix}\right] = \boldsymbol{0} \\ & \begin{bmatrix} \boldsymbol{p} - \bar{\boldsymbol{p}} \\ \underline{\boldsymbol{p}} - \boldsymbol{p} \\ \boldsymbol{f} - \bar{\boldsymbol{f}} \\ \underline{\boldsymbol{f}} - \boldsymbol{f} \end{bmatrix} < \boldsymbol{0} \end{aligned} \tag{7}$$

We propose using primal-dual interior point (PDIP) methods to solve the optimization problem in (7). First, we will provide a basic overview of interior-point algorithms, and then we will present our parallel PDIP algorithm for OPF.

### C. Preliminaries on Interior-Point Methods

Interior-point methods are used to solve optimization problems with inequality constraints and often demonstrate superlinear convergence behavior [10]. To deal with the inequality constraints, interior-point methods solve a series of equality-constrained problems that are a function of an adaptively changing parameter $\gamma > 0$ [12]. As $\gamma$ increases, the inequality constraints are more strictly enforced. For each value of $\gamma$, Newton's method is applied to solve the Karush-Kuhn-Tucker (KKT) equations. For the DC-OPF formulation in (7), the KKT equations are given by

$$\boldsymbol{r}_\gamma(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = \begin{bmatrix} \boldsymbol{r}_{dual} \\ \boldsymbol{r}_{cent} \\ \boldsymbol{r}_{pri} \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} \nabla f_0(\boldsymbol{x}) + D\boldsymbol{g}(\boldsymbol{x})^T \lambda + \boldsymbol{A}^T \boldsymbol{\nu} \\ -\text{diag}(\boldsymbol{\lambda})\boldsymbol{g}(\boldsymbol{x}) - (1/\gamma)\boldsymbol{1} \\ \boldsymbol{A}\boldsymbol{x} \end{bmatrix}, \tag{8}$$

where the dual variables $\boldsymbol{\nu}$ are associated with the equality constraints and the centrality variables $\boldsymbol{\lambda}$ are associated with the inequality constraints. Let

$$\boldsymbol{y} \overset{\text{def}}{=} (\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}),\ \Delta\boldsymbol{y} \overset{\text{def}}{=} (\Delta\boldsymbol{x}, \Delta\boldsymbol{\lambda}, \Delta\boldsymbol{\nu}). \tag{9}$$

Primal-dual interior-point methods utilize Newton's method to solve the set of non-linear equations $\boldsymbol{r}_\gamma(\boldsymbol{y}) = \boldsymbol{0}$ via the following first-order approximation

$$D\boldsymbol{r}_\gamma(\boldsymbol{y'})\Delta\boldsymbol{y} = \boldsymbol{r}_\gamma(\boldsymbol{y'}), \tag{10}$$

where $\boldsymbol{y'}$ is the value of the previous PDIP iterate. The basic outline of an interior-point algorithm is summarized below [12].

## III. Parallel Primal-Dual Interior-Point Method for OPF

Our aim is to develop a parallel primal-dual interior-point (PDIP) algorithm for the OPF formulation in (7) with limited communication. We partition the buses of the power network into $N$ control areas. The decision variables associated with the power injections at the buses in control area $I$ are denoted $\boldsymbol{p}_I$. Consider the decision variable associated with the power flow on line $l(i,j)$. Recall that our convention requires $i <$

**Algorithm 1** Interior-Point Algorithm Overview [12]

```
1: Choose an initial feasible point for inequality constraints.
2: while termination critera unsatisfied do
3:     Set parameter $\gamma$ in terms of current iterate, $\boldsymbol{y}$.
4:     Compute the search direction, $\Delta\boldsymbol{y}$.
5:     Compute the step length, $d$.
6:     Compute the next iterate, $\boldsymbol{y}^+ = \boldsymbol{y} + d\Delta\boldsymbol{y}$.
7:     Evaluate termination criteria to measure feasibility and
   optimality of current solution, $\boldsymbol{y}^+$.
```

$j$. We assign the decision variable $f_{l(i,j)}$ to the control area containing bus $i$. Denote the following sets for control area $I$

$$\begin{aligned} F_I &= \{f_{l(i,j)} \mid i \in I, j \in I\} \\ S_I &= \{f_{l(i,j)} \mid i \in I, j \in J \neq I\} \\ T_I &= \{f_{l(j,i)} \mid i \in I, j \in J \neq I\}. \end{aligned} \tag{11}$$

This partitions the power flows spanning two different control areas into two categories 1) those that are locally available $S_I$ and 2) those that must be communicated $T_I$. These sets are useful for analyzing the communication requirements in later sections. Thus, $\boldsymbol{f}_I = F_I \cup S_I$.

There are three challenges to developing a parallel PDIP algorithm for OPF: 1) solving for the search direction $\Delta\boldsymbol{y}$ in Step 4 of Algorithm 1, 2) calculating the step length $d$ in Step 5, and 3) evaluating the termination criteria in a parallel way in Step 7.

### *A. Distributed Search Direction Calculation*

Calculating the search direction involves solving the large, sparse linear system in (10). For DC-OPF, we have the following simplifications to (8)

$$\boldsymbol{r}_{dual} = \begin{bmatrix} \nabla_{\boldsymbol{p}} f_0 \\ \\ \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} \lambda(\bar{p}_1) - \lambda(\underline{p}_1) \\ \vdots \\ \lambda(\bar{p}_n) - \lambda(\underline{p}_n) \\ \lambda(\bar{f}_1) - \lambda(\underline{f}_1) \\ \vdots \\ \lambda(\bar{f}_L) - \lambda(\underline{f}_L) \end{bmatrix} + \begin{bmatrix} -\boldsymbol{\nu} \\ \\ \boldsymbol{B}^T\boldsymbol{\nu} \end{bmatrix}, \tag{12}$$

where $\lambda(\bar{p}_i)$ and $\lambda(\underline{p}_i)$ are the centrality variables associated with the upper and lower capacity constraints on the power injection at bus $i$, respectively. Similarily, $\lambda(\bar{f}_i)$ and $\lambda(\underline{f}_i)$ are the centrality variables associated with the upper and lower capacity constraints on line $i$, respectively. Furthermore, since $\boldsymbol{g}(\boldsymbol{x})$ is separable, the entries of the vector $\boldsymbol{r}_{cent}$ are separable. We note that except for the term $\boldsymbol{B}^T\boldsymbol{\nu}$ in $\boldsymbol{r}_{dual}$ and the term $\boldsymbol{B}\boldsymbol{f}$ in $\boldsymbol{r}_{pri}$ our formulation yields expressions for the KKT equations that are separable. In later sections, we will show how this facilitates a parallel solution to (8). A reduced system to (10) is formed by eliminating $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{\lambda}$, yielding

$$\begin{aligned} \boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{A}^T\Delta\boldsymbol{\nu} =& \boldsymbol{r}_{pri} - \boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{r}_{dual} \\ & - \boldsymbol{A}\boldsymbol{H}^{-1}D\boldsymbol{g}(\boldsymbol{x})^T \text{diag}(\boldsymbol{g}(\boldsymbol{x}))^{-1}\boldsymbol{r}_{cent} \end{aligned} \tag{13}$$

where for DC-OPF, the matrix $\boldsymbol{H}$ is diagonal with $i$th diagonal entry given by

$$[\boldsymbol{H}]_{ii} = \begin{cases} \frac{\partial^2 f_0}{\partial p_i^2} + \frac{-\lambda(\underline{p}_i)}{\underline{p}_i - p_i} + \frac{-\lambda(\bar{p}_i)}{p_i - \bar{p}_i} & 1 \leq i \leq n \\ \frac{-\lambda(\underline{f}_{i-n})}{-f_{i-n} + \underline{f}_{i-n}} + \frac{-\lambda(\bar{f}_{i-n})}{f_{i-n} - \bar{f}_{i-n}} & n+1 \leq i \leq n+L. \end{cases} \tag{14}$$

$$\text{Let } \boldsymbol{P} \overset{\text{def}}{=} (\boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{A}^T) \tag{15}$$

$$\boldsymbol{w} \overset{\text{def}}{=} \boldsymbol{r}_{pri} - \boldsymbol{A}\boldsymbol{H}^{-1}(\boldsymbol{r}_{dual} + D\boldsymbol{g}(\boldsymbol{x})^T \text{diag}(\boldsymbol{g}(\boldsymbol{x}))^{-1}\boldsymbol{r}_{cent}), \tag{16}$$

where the simplifications for terms comprising $\boldsymbol{w}$ are given in (8) and (12). We now present a fully-distributed method for calculating the solution to

$$\boldsymbol{P}\Delta\boldsymbol{\nu} = \boldsymbol{w}. \tag{17}$$

To achieve this, the decision variables $\boldsymbol{x}$ are permuted and partitioned into control areas,

$$\boldsymbol{x} = [\boldsymbol{x}_1 \; \boldsymbol{x}_2 \; \ldots \; \boldsymbol{x}_N]^T, \tag{18}$$

where $\boldsymbol{x}_I = [\boldsymbol{p}_I \; \boldsymbol{f}_I]^T$. The equality and inequality constraints are permuted according to the reordering of the decision variables in $\boldsymbol{x}$. For simplicity, we allow this abuse in notation for $\boldsymbol{x}$, $\boldsymbol{A}$, and $\boldsymbol{g}(\cdot)$, where the ordering will be assumed to be according to (18) in the sequel. Under our formulation, the system matrix $\boldsymbol{P}$ is a large, sparse system. As an example, the sparsity pattern of $\boldsymbol{P}$ for a 118-bus power system is given in Figure 1. Our aim is to develop an efficient iterative method based on matrix-splitting to exploit this sparsity to solve the reduced system in (17). Consider decomposing the matrix $\boldsymbol{P}$ into the difference of two matrices $\boldsymbol{M}$ and $\boldsymbol{N}$. Provided the spectral radius $\rho(\boldsymbol{M}^{-1}\boldsymbol{N})$ is less than one, let the fixed-point of the sequence

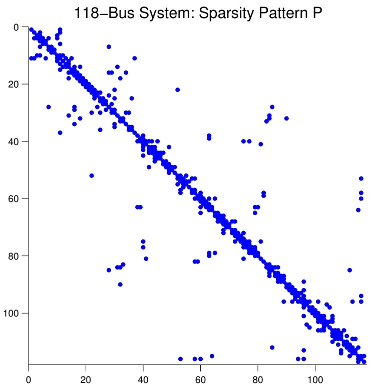


Figure 1. The sparsity of the system matrix $\boldsymbol{P}$ can be used to develop an efficient iterative inversion method.

$$\Delta\boldsymbol{\nu}^{t+1} = \boldsymbol{M}^{-1}\boldsymbol{N}\boldsymbol{\nu}^t + \boldsymbol{M}^{-1}\boldsymbol{w}, \tag{19}$$

be $\Delta\boldsymbol{\nu}^*$, then $\boldsymbol{P}\Delta\boldsymbol{\nu}^* = \boldsymbol{w}$. The splitting (*i.e.* choice of $\boldsymbol{M}$ and $\boldsymbol{N}$) should be designed so that 1) the matrix-splitting iterates converge and 2) the iterates in (19) are easy to calculate in a distributed way.

Without loss of generality, assume the bus ids and line ids are consecutively assigned across control areas. The matrix $\boldsymbol{P}$ can be decomposed into the sum of a block-diagonal matrix, $\boldsymbol{D}$, and a matrix containing the remaining off-diagonal entries $\boldsymbol{E}$. The consecutive ordering of the bus and line indices described above allows for all entries of $\boldsymbol{P}$ corresponding to buses within the same control area to be contained within a single diagonal block. Specifically, let

$$D_{ij} = \begin{cases} P_{ij} & \text{if buses } i \text{ and } j \\ & \text{belong to the same control area ,} \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

$$E_{ij} = \begin{cases} P_{ij} & \text{if buses } i \text{ and } j \\ & \text{belong to different control areas ,} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

yielding $\boldsymbol{P} = \boldsymbol{D} + \boldsymbol{E}$. We use the following matrix-splitting design

$$\boldsymbol{M} = \boldsymbol{D} + \tau \bar{\boldsymbol{E}}, \ \boldsymbol{N} = \tau \bar{\boldsymbol{E}} - \boldsymbol{E}, \tag{22}$$

where $\tau$ is a scalar parameter and $\bar{\boldsymbol{E}}$ is a diagonal matrix whose $i$th diagonal entry equals

$$\bar{E}_{ii} \overset{\text{def}}{=} \sum_{j \neq i} |A_{ij}|. \tag{23}$$

This is a block-Jacobi scheme modified to be diagonally dominant. We have the following proposition which ensures convergence of the matrix-splitting iterates in (19).

**Proposition 1.** *Using the splitting in (22), for $\tau \geq \frac{1}{2}$, the iterative updates in (19) converge.*

The proof is given in Appendix A. Control area *I* has its search direction update given by

$$\Delta \boldsymbol{\nu}_I^{t+1} = \boldsymbol{M}_I^{-1} \left[ \sum_{J \neq I} \boldsymbol{N}_{IJ} \Delta \boldsymbol{\nu}_J^t + \boldsymbol{w}_I \right]. \tag{24}$$

The calculation for the search direction $\Delta \boldsymbol{y}$ concludes by using back-substitution of $\Delta \boldsymbol{\nu}$ to calculate $\Delta \boldsymbol{x}$ and $\Delta \boldsymbol{\lambda}$. The information exchange needed to calculate the search direction in a fully-distributed way is summarized in the following proposition.

**Proposition 2.** *In order to calculate $\Delta \boldsymbol{y}_I^{t+1}$ in a fully distributed way, area I must receive from its neighbors the values of the power flows in the set $T_I$ defined in (11). In addition, for the power flows in $S_I$, consider the set of neighboring buses $\{j\}$ in other control areas. The neighboring control areas must send the values of $\{\boldsymbol{\nu}_j, \Delta \boldsymbol{\nu}_j\}$ corresponding to the power injections at these buses. For the power flows in $l \in T_I$, the neighboring control areas must send the values of $\boldsymbol{r}_{dual}$ corresponding to the decision variables for power flows $l \in T_I$ and the values of $\boldsymbol{r}_{cent}$ corresponding to the inequality constraints on the lower and upper bounds for power flows $l \in T_I$.*

The proof of the above result is provided in Appendix B.

### *B. Parallel Step Length Calculation*

Once the search direction $\Delta \boldsymbol{y}$ has been calculated, the next iterate $\boldsymbol{y}^+$ is produced from the current iterate $\boldsymbol{y}$ by

$$\boldsymbol{y}^+ = \boldsymbol{y} + d \Delta \boldsymbol{y}, \tag{25}$$

where $d$ is the step length (Steps 5 and 6, Algorithm 1). A full Newton step corresponds to a step length of 1. In order to ensure feasibility (*e.g.*, $\boldsymbol{g}(\boldsymbol{x}^+) < \boldsymbol{0}$ and $\boldsymbol{\lambda}^+ \geq \boldsymbol{0}$), the step length is in general less than 1. One common approach to determining the step length is to use a backtracking line search [12]. To facilitate parallelization of the backtracking line search, we introduce local pre-processing schemes that

- Reduce the size of the information sent and the computation to be performed at the central coordinator.
- Allow for increased privacy by disclosing a surjective function of each control area's information to the central coordinator.

Our proposed parallel backtracking line search is presented in Algorithm 2 and based on the centralized one given in [12]. The parameters $\alpha$ and $\beta$ are chose *a priori*.

**Algorithm 2** Parallel Backtracking Line Search Calculation

```
1: Each control area $I$ calculates $d_{max}^I = \min\{1, \min_{i|\lambda_i \in \boldsymbol{y}_I}\{-\lambda_i/\Delta\lambda_i < 0\}\}$ and sends it to the central coordinator, $C$.
2: $C$ calculates $d_{max} := \min_I\{d_{max}^I\}$ and sends it to the control areas.
3: Each area $I$ sets $d_I := d_{max}$.
4: while $\boldsymbol{g}_I(\boldsymbol{x}^+) \geq \boldsymbol{0}$ do $d_I := 0.99 * d_{max}$.
5: Each area $I$ sends $d_I$ to $C$.
6: $C$ calculates $d := \min_I\{d_I\}$ and sends it to control areas.
7: Each area $I$ sets $d_I := d$.
8: while $||[\boldsymbol{r}_\gamma(\boldsymbol{y}^+)]_I||_2 \geq (1-\alpha d_I)||[\boldsymbol{r}_\gamma(\boldsymbol{y})]_I||_2$ do $d_I := \beta d$
9: Each area sends $d_I$ to $C$.
10: $C$ calculates $d := \min_I\{d_I\}$ and sends it to all areas.
```

The convergence of the centralized version of this scheme in a finite number of steps is guaranteed in [12]. The convergence of the parallel scheme follows easily. Beside communication with the central processor, we consider the neighbor-to-neighbor communication of this algorithm. First, we note that $\boldsymbol{g}_I(\boldsymbol{x}^+)$ can be calculated locally at each area without communication. The only information exchange between areas is used to calculate the norm of the residuals in Step 8 of Algorithm 2. The information exchange is summarized in the following proposition.

**Proposition 3.** *To calculate $\boldsymbol{r}_\gamma^I(\boldsymbol{y})$, area I must receive from its neighbors the values of the power flows in the set $T_I$ defined in (11). In addition, for the power flows in $S_I$, consider the set of neighboring buses $\{j\}$ in other control areas. The neighboring control areas must send the values of $\{\nu_j\}$ corresponding to the power injections at these buses.*

The proof is provided in Appendix C. Of note, as shown in Proposition 2, this information was already exchanged in calculating the search direction.

### C. Parallel Termination Checking

Criteria measuring the feasibility and optimality of the new iterate are evaluated to determine whether to terminate the algorithm. Again, we consider how local pre-processing can be done to more effectively parallelize the algorithm. Locally, at PDIP iteration $k$ each area $I$ calculates $\hat{\eta}_I = \boldsymbol{g}(\boldsymbol{x}_I^k)^T \boldsymbol{\lambda}_I^k$ in parallel. This is then sent to the central coordinator, which sums all contributions to calculate $\hat{\eta} = \sum_{I=1}^{N} \hat{\eta}_I$. Each area has already calculated the squared norm of its local residual vectors $||\boldsymbol{r}_{pri,I}||^2$ and $||\boldsymbol{r}_{dual,I}||^2$ during the backtracking line search in Section III-B. These values are sent to the central coordinator, which calculates $||r_{pri}|| = \sqrt{\sum_{I=1}^{N} ||\boldsymbol{r}_{pri,I}||^2}$ and $||r_{dual}|| = \sqrt{\sum_{I=1}^{N} ||\boldsymbol{r}_{dual,I}||^2}$. The central coordinator checks the termination criterion and broadcasts whether or not to terminate to all control areas.

### D. Overview of Parallel PDIP and Analysis of Communication Requirements

We detail the communication requirements for each of the 13 steps in the proposed parallel PDIP-OPF in Algorithm 3.

**Algorithm 3** Parallel PDIP-OPF for Control Area $I$

```
1:  Initialization: Set feasible initial point for $\boldsymbol{y}^0$, and set
    $||r_{pri}||_2, ||r_{dual}||_2$, and $\hat{\eta}$ to $10 * max(\epsilon_{feas}, \epsilon)$.
2:  while $||r_{pri}||_2 > \epsilon_{feas}, ||r_{dual}||_2 > \epsilon_{feas}$, or $\hat{\eta} > \epsilon$ do
3:      Calculate $t^k := \mu m / \hat{\eta}$
4:      Calculate $\boldsymbol{M}_I^{-1} \boldsymbol{w}_I$.
5:      Initialize $\Delta \boldsymbol{\nu}_I^t := \boldsymbol{0}$
6:      while $||\Delta \boldsymbol{\nu}_I^{t+1} - \Delta \boldsymbol{\nu}_I^t|| > \delta$ do
7:          Calculate $\Delta \boldsymbol{\nu}_I^{t+1} = \boldsymbol{M}_I^{-1}[\boldsymbol{w}_I + \sum_{J \neq I} \boldsymbol{N}_{IJ} \Delta \boldsymbol{\nu}_J^t]$
8:      $\Delta \boldsymbol{\nu}_I^k := \Delta \boldsymbol{\nu}_I^{t+1}$
9:      Substitute $\Delta \boldsymbol{\nu}_I^k$ to calculate $\Delta \boldsymbol{x}_I^k$.
10:     Substitute $\Delta \boldsymbol{x}_I^k$ to calculate $\Delta \boldsymbol{\lambda}_I^k$.
11:     Coordinate with central controller to calculate step size
        $d^k$ using backtracking line search.
12:     Calculate next PDIP iterate $\boldsymbol{y}_I^{k+1} = \boldsymbol{y}_I^k + d^k \Delta \boldsymbol{y}_I^k$.
13:     Coordinate with central controller to calculate $\hat{\eta} =$
        $-\boldsymbol{g}(\boldsymbol{x}^{k+1})^T \boldsymbol{\lambda}^{k+1}$.
```

- In Step 1, no communication is required. The $\boldsymbol{y}$ variables are initialized to a point feasible with respect to the inequality constraints. For example, we initialize the power injections, $\boldsymbol{\lambda}$ centrality variables, and $\boldsymbol{\nu}$ dual variables to small positive values, and the power flows are initially set to zero. We assume each control area is given the global initial starting point *a priori*.
- In Steps 2 and 3, no communication is needed. The values for the residuals and surrogate duality gap, $\hat{\eta}$, are either set at initialization or calculated below in the previous iteration. The parameter $\mu$ is a constant, and $m = 2(n + L)$ is the number of inequality constraints.
- A fully distributed scheme for calculating Step 4 and its communication requirements (see Proposition 2) are given in Section III-A. We stress that in the system matrix $\boldsymbol{P} = \boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{A}^T$ only the diagonal matrix $\boldsymbol{H}$ changes at each PDIP iteration, so there is limited computation to recalculate $\boldsymbol{P}$.
- In Steps 5, 6, 8, and 12 all quantities are local, so no communication is needed.
- In Step 7, only neighbor-to-neighbor communication is needed. As shown in the proof of Proposition 2, $\{\boldsymbol{N}_{IJ} | J \neq I\}$ can be calculated locally. The matrix $\boldsymbol{N}_{IJ}$ is non-zero only if control area $I$ contains at least one bus with a neighbor in control area $J$. Only the updates to the dual Newton step, $\{\Delta \boldsymbol{v}_J^t | \boldsymbol{N}_{IJ} \neq \boldsymbol{0}\}$, need to be communicated at each matrix-splitting iteration.
- In Steps 9 and 10, no additional communication is needed as values were communicated at Steps 4 and 7.
- In Step 11, coordination with a central coordinator is required with local processing as described in Algorithm 2. The neighbor-to-neighbor exchange is the same as required in Proposition 2 and needs only to be done once per PDIP iteration.
- In Step 13, local pre-processing in Section III-C is used to reduce the communication and computation with the central processor.

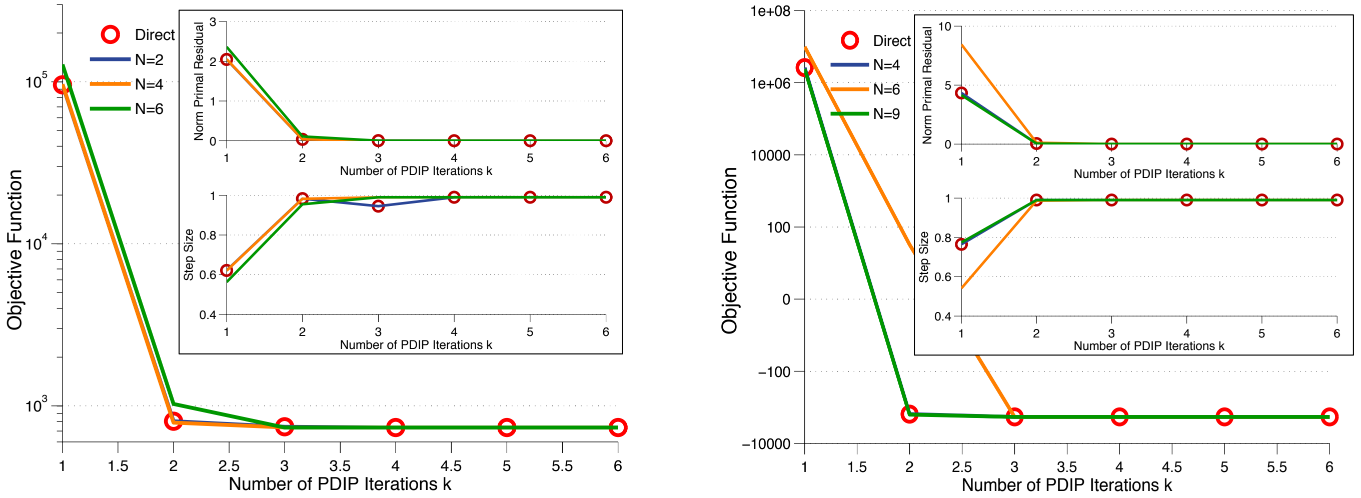
## IV. Numerical Results

In Figure 2, we study the performance of the parallel PDIP algorithm for two different size networks, the IEEE 118-bus and PEGASE 1,354-bus [13], [14]. We use the following quadratic cost function:

$$f_0(\boldsymbol{x}) \overset{\text{def}}{=} \sum_{i=1}^{n} a_i p_i^2 + b_i p_i + c_i. \tag{26}$$

The convergence using direct inversion (*i.e.*, centralized method) to solve system (17) is compared to that using the iterative inversion (*i.e.*, distributed method) under partitionings into different sized control areas. Both the centralized and distributed methods converge within several PDIP iterations. Moreover, as the network size grows, the number of PDIP iterations needed remains constant, and and the degree to which the network is partitioned can tune the computational cost of each iteration. Using more control areas corresponds to a more distributed configuration with comparable performance. In Figure 2, the primal residual converges to zero ensuring feasibility of the equality constraints, and the step size becomes one (*i.e.*, a full Newton step) as the PDIP method converges. The values used for $\mu$, $\alpha$, and $\beta$ are 10, 0.01, and 0.3, respectively. The number of inner matrix-splitting iterations used in the distributed method is determined by the following termination criterion, $|\Delta v^+ - \Delta v| < \delta$, where $\delta = 10^{-10}$. The number of matrix-splitting iterations used per PDIP iteration is given in Table I.

The modified block-Jacobi method allows for a fully distributed calculation of the search direction. However, parallel direct inversion methods for sparse matrices like multifrontal LU decomposition can be an attractive choice when memory and centralized communication is not of high concern. Using



(a) 118-Bus System

(b) 1,354-Bus System

Figure 2. These figures show the performance of the PDIP and proposed parallel PDIP methods for the DC-OPF formulated in Section II-B. The main figures compare the convergence of the objective function for different number of control areas. The inset figures provide various performance measures beyond the objective function.

multifrontal LU decomposition, we achieve a runtime of $6.10 * 10^{-3}$ seconds for solving the search direction on the PEGASE 9,241-bus system. To compare to this method, in (19), the inverse $\boldsymbol{M}_I^{-1}$ is calculated only at the first matrix-splitting iteration and then can be reused, so that all subsequent iterations are reduced to simple matrix-vector multiplications. Different control areas contain a different number of buses. Using control areas with 11, 51, and 101 buses, the local matrix inversion required at the first iteration takes $t_{inv} = 8.87 * 10^{-5}, 4.39 * 10^{-4}$, and $1.2 * 10^{-4}$ seconds, respectively. A single inner iteration of the modified block Jacobi method for the 9,241-bus system takes $t_{loop} = 2.85 * 10^{-5}, 3.09 * 10^{-5}$, and $4.36 * 10^{-5}$ seconds using control areas with 11, 51, and 101 buses, respectively. All runtimes are averaged over 50 trials. Control areas with smaller numbers of buses require less computation time, so increasing the number of control areas makes the algorithm highly scalable. Then, without taking into account communication time, the total runtime for the modified block-Jacobi method is $N_{inner} \times t_{loop} + t_{inv}$ seconds, where $N_{inner}$ is the number of matrix-splitting iterations used. The value for $N_{inner}$ depends on the tolerance $\delta$. In future work, we will further explore the trade-off between the inner-loop tolerance and outer-loop convergence.

## V. Conclusion

We presented a parallel primal-dual interior point method for optimal power flow. The favorable convergence properties of PDIP are demonstrated through numerical experiments. The limited communication requirements are analyzed. In future work, we plan to extend to the AC-OPF problem and implement it using MPI in order to accurately measure runtimes.

TABLE I. Number of Matrix-Splitting Iterations

| PDIP Iteration | n=118 N=2 | n=118 N=4 | n=118 N=6 | n=1354 N=4 | n=1354 N=6 | n=1354 N=9 |
|---|---|---|---|---|---|---|
| 1 | 249 | 629 | 667 | 1580 | 1593 | 1677 |
| 2 | 298 | 724 | 697 | 1981 | 1877 | 1986 |
| 3 | 374 | 941 | 777 | 3309 | 2539 | 2428 |
| 4 | 441 | 1627 | 767 | 4217 | 2622 | 2096 |
| 5 | 589 | 1759 | 609 | 2929 | 2053 | 1645 |
| 6 | 437 | 1021 | 505 | 2117 | 1613 | 1309 |
| 7 | – | – | – | 1668 | 1261 | 1004 |
| 8 | – | – | – | 1262 | 922 | 702 |

## Appendix

### A. Proof of Proposition 1: Convergence of Matrix-Splitting Iterates

*Proof.* The sequence $\{\Delta \boldsymbol{v}^t\}$ in (19) converges to its limit $\Delta \boldsymbol{v}^*$ as $t \rightarrow \infty$ if and only if the spectral radius of the matrix $\boldsymbol{M}^{-1}\boldsymbol{N}$ is strictly less than 1 [11]. Furthermore, if the sequence converges, the limit $\Delta \boldsymbol{v}^*$ is the solution of the system, (i.e., $\boldsymbol{P}\Delta \boldsymbol{v}^* = \boldsymbol{w}$). In order to have the spectral radius $\rho(\boldsymbol{M}^{-1}\boldsymbol{N}) < 1$, it is sufficient to have $\boldsymbol{P} = \boldsymbol{M} - \boldsymbol{N} \succ \boldsymbol{0}$ and $\boldsymbol{M} + \boldsymbol{N} \succ \boldsymbol{0}$ [15]. To show that $\boldsymbol{P} \succ 0$, it is sufficient to show that 1) $\boldsymbol{A}^T$ has full column rank and that 2) $\boldsymbol{H}^{-1}$ has strictly positive diagonal entries. Recall from (7) that

$$\boldsymbol{A}^T = \begin{bmatrix} -\boldsymbol{I} \\ \boldsymbol{B}^T \end{bmatrix}.$$

The matrix $\boldsymbol{A}^T$ has full column rank by construction. Last, we show that $\boldsymbol{H}$ has strictly positive diagonal entries. From

[12], we have

$$\begin{aligned}\boldsymbol{H} =&\nabla^2 f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i \nabla^2 g_i(\boldsymbol{x}) + \\ &\sum_{i=1}^{m} \frac{\lambda_i}{-g_i(\boldsymbol{x})} \nabla g_i(\boldsymbol{x}) \nabla g_i(\boldsymbol{x})^T \qquad (27a)\\ =&\nabla^2 f_0(\boldsymbol{x}) + \sum_{i=1}^{m} \frac{\lambda_i}{-g_i(\boldsymbol{x})} \nabla g_i(\boldsymbol{x}) \nabla g_i(\boldsymbol{x})^T, \qquad (27b)\end{aligned}$$

where $\nabla^2 f_i(\boldsymbol{x})$ is zero since the inequality constraints in (3) are linear. Furthermore, since the inequality constraints are separable and there are two inequality constraints per decision variable,

$$\sum_{i=1}^{m} \nabla g_i(\boldsymbol{x}) \nabla g_i(\boldsymbol{x})^T = 2\boldsymbol{I}_{(n+L)\times(n+L)}.$$

Since each iterate is feasible, $f_i(\boldsymbol{x}) \leq 0$ and $\lambda_i \geq 0$, so we have $-\lambda_i / f_i(\boldsymbol{x}) \geq 0$. Therefore, the matrix given by

$$\sum_{i=1}^{m} \frac{\lambda_i}{-f_i(\boldsymbol{x})} \nabla f_i(\boldsymbol{x}) \nabla f_i(\boldsymbol{x})^T$$

is diagonal with strictly positive values. Since $f_0$ is a strictly convex function separable in $\boldsymbol{p}$, $\nabla^2 f_0(\boldsymbol{x})$ is a diagonal matrix with strictly positive values on the entries corresponding to the power injections and zero values on the entries corresponding to the power flows. We conclude that $\boldsymbol{H}$ is diagonal with strictly positive diagonal entries, which finalizes the proof. Second, $\boldsymbol{M} + \boldsymbol{N}$ is positive definite due to its construction which makes it diagonally dominant. Details, omitted here for space, are provided in [16]. □

### *B. Proof of Proposition 2: Communication Requirements for $\Delta \boldsymbol{y}_I$*

*Proof.* Calculation of $\Delta \boldsymbol{y}$ involves computing the term $\boldsymbol{M}_I^{-1} \boldsymbol{w}_I$. First, we note that the matrix $\boldsymbol{M}_I^{-1}$ can be calculated locally without any communication. To see this, note that area $I$ can calculate $\boldsymbol{M}_I$ and $\{\boldsymbol{N}_{IJ} | J \neq I\}$ as defined in (20)-(22). We have that $P_{ij} = \sum_l [\boldsymbol{H}^{-1}]_{ll} A_{il} A_{jl}$ . The matrix $\boldsymbol{A}$ simply encodes the topology of the network, and assuming a control area knows all of its neighboring buses, the $i$th row of $\boldsymbol{A}$ is known to the control area containing bus $i$. The diagonal matrix $\boldsymbol{H}$ is characterized in (15), and each control area can locally calculate its relevant entries of $\boldsymbol{H}$. We stress that in calculating $\boldsymbol{P}$ at each PDIP iteration, only the matrix $\boldsymbol{H}$ is changing. For $\boldsymbol{w}$ in (16), we see that the information exchange needed for calculating $\boldsymbol{r}_{dual}$ and $\boldsymbol{r}_{cent}$ is the same as that in Proposition 3. We note that $\boldsymbol{H}^{-1}[D\boldsymbol{g}(\boldsymbol{x})^T \text{diag}(\boldsymbol{g}(\boldsymbol{x}))^{-1} \boldsymbol{r}_{cent}]_I$ and $\boldsymbol{H}^{-1} \boldsymbol{r}_{dual}$ require no communication to calculate locally. In our formulation the inequality constraints $\boldsymbol{g}(\boldsymbol{x})$ are linear and separable, so there is no mixing of terms that might require communication. To calculate locally relevant entries of $\boldsymbol{A}\boldsymbol{H}^{-1}[D\boldsymbol{g}(\boldsymbol{x})^T \text{diag}(\boldsymbol{g}(\boldsymbol{x}))^{-1} \boldsymbol{r}_{cent}]_I$ and $\boldsymbol{A}[r_{dual}]_I$, areas need to receive entries in $\boldsymbol{r}_{dual}$ and $\boldsymbol{r}_{cent}$ corresponding to power flows in $T_I$. □

### *C. Proof of Proposition 3: Communication Requirements for $\boldsymbol{r}_\gamma^I(\boldsymbol{y})$*

*Proof.* As noted in equations (8) and (12), the only terms that violate the separability of $\boldsymbol{r}_\gamma(\boldsymbol{y})$ are $\boldsymbol{Bf}$ and $\boldsymbol{B}^T \boldsymbol{\nu}$. Thus, it is sufficient to analyze the communication requirements for calculating these two terms. The relevant entries of the vector $\boldsymbol{Bf}$ to area $I$ correspond to the power injection decision variables local to area $I$. For the power injection at node $i$, calculation of $[\boldsymbol{Bf}]_i$ requires the values of all power flows incident to bus $i$. Therefore, neighbors must send the values of the flows in $T_I$ to area $I$. Similarily, the relevant entries of the vector $\boldsymbol{B}^T \boldsymbol{\nu}$ to area $I$ correspond to the power flows in $F_I \cup S_I$. For the power flow $f_{l(i,j)} \in S_I$, calculation of $[\boldsymbol{B}^T \boldsymbol{\nu}]_l$ requires node $j$ to send the value of its dual variable $\nu(p_j)$ to node $i$. □

## References

[1] A. Lam, B. Zhang, and D. Tse, "Distributed Algorithms for Optimal Power Flow Problem," in *2012 IEEE 51st Annual Conference on Decision and Control (CDC)*, Dec 2012, pp. 430–437.

[2] E. Dall'Anese, H. Zhu, and G. B. Giannakis, "Distributed Optimal Power Flow for Smart Microgrids," *IEEE Trans. Smart Grid*, vol. 4, no. 3, pp. 1464–1475, 2013.

[3] T. Erseghe, "Distributed Optimal Power Flow Using ADMM," *IEEE Trans. Power Systems*, vol. 29, no. 5, pp. 2370–2380, Sept 2014.

[4] S. Magnusson, P. C. Weeraddana, and C. Fischione, "A Distributed Approach for the Optimal Power-Flow Problem Based on ADMM and Sequential Convex Approximations," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 3, pp. 238–253, Sept 2015.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[6] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, "Dynamic Network Energy Management via Proximal Message Passing," *Foundations and Trends in Optimization*, vol. 1, no. 2, 2014.

[7] R. Jabr, A. Coonick, and B. Cory, "A Primal-Dual Interior Point Method for Optimal Power Flow Dispatching," *IEEE Trans. Power Systems*, vol. 17, no. 3, pp. 654–662, Aug. 2002.

[8] F. Capitanescu, D. Ernst, M. Glavic, and L. Wehenkel, "Interior-Point Based Algorithms for the Solution of Optimal Power Flow Problems," *Electric Power Systems Research*, pp. 508–517, Apr. 2007.

[9] G. Hug-Glanzmann and G. Andersson, "Decentralized Optimal Power Flow Control for Overlapping Areas in Power Systems," *IEEE Trans. Power Systems*, vol. 24, no. 1, pp. 327–336, Feb 2009.

[10] S. Wright, *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics, 1997.

[11] R. S. Varga, *Matrix Iterative Analysis*, ser. Springer Series in Computational Mathematics. Berlin, Heidelberg, Paris: Springer Verlag, 2000.

[12] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[13] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "MATPOWER: Steady-State Operations, Planning and Analysis Tools for Power Systems Research and Education," *Power Systems, IEEE Transactions on*, vol. 26, no. 1, pp. 12–19, 2011.

[14] S. Fliscounakis, P. Panciatici, F. Capitanescu, and L. Wehenkel, "Contingency Ranking with respect to Overloads in Very Large Power Systems taking into Account Uncertainty, Preventive and Corrective Actions," *IEEE Trans. Power Systems*, vol. 28, no. 4, pp. 4909–4917, 2013.

[15] R. W. Cottle, J. Pang, and R. Stone, *Linear Complementarity Problem.* Academic Press, 1992.

[16] A. Minot and N. Li, "A Fully Distributed State Estimation Using Matrix Splitting Methods," *American Control Conference*, 2015.