arXiv:2305.01032v2 [math.OC] 5 Aug 2024

采用谱惩罚参数选择的电力系统分布式优化径向划分

MEHDI KARIMI

Abstract. 本文针对最优潮流（OPF）优化问题，提出了基于智能划分的基于分组的分布式优化（DO）算法。本文引入电力网络图的径向划分，将大规模问题系统地拆分为更易处理的子问题，并可利用凸松弛等方法高效求解。本文引入谱参数选择，将其作为基于分组的分布式优化中的关键超参数选择步骤。我们创建了 DiCARP 软件包，该软件包使用 Python 基于 Pyomo 优化软件包实现。针对不同电力网络实例的数值结果表明，与基于组件的 DO 相比，我们设计的算法在迭代次数更少的情况下，为测试问题返回了更精确的解。我们的结果证实，对于网络上的大规模优化问题，智能划分和参数选择具有重要意义。

1. Introduction

随着光伏（PV）发电、具备车辆到电网能力的车辆以及电池储能等新能源的利用日益增加，现代电力网络正以前所未有的速度发生变化。电力网络运行中出现的问题变得如此复杂且规模庞大，以至于传统的集中式算法可能已不再适用。对于如此复杂且不断扩大的问题，单台机器上的集中式计算难以实现。另一方面，将所有信息集中到一个集中式控制器中，需要庞大的通信基础设施，还可能引发网络安全问题。分布式和并行计算 [4, 21, 27, 41]是应对此类情况的一种解决方案：将主要计算问题划分为多个子问题，并由各个计算代理分别处理。分布式优化（DO）

日期：2024年8月7日。Mehdi Karimi：伊利诺伊州立大学数学系，伊利诺伊州诺默尔，邮编61761。（电子邮件：**mkarim3@ilstu.edu**）。作者的研究得到美国国家科学基金会（NSF）CMMI-2347120号拨款的资助。

算法，尤其是交替方向乘子法（ADMM）[7]，已在电力系统领域得到广泛研究，相关综述论文[32, 28, 42, 37] 及其参考文献中有诸多实例。然而，尽管该领域已有数百篇论文，分布式算法仍未得到广泛的实际应用。分布式优化算法的主要问题是收敛缓慢 和参数选择困难，本文将对此进行探讨。

并非所有问题都可以分解并以分布式方式处理。对于具有这种有利结构的问题，例如电力网络运行中的许多问题，计算代理会相互协调和通信，以解决主要问题。电力网络运行中的主要优化问题称为最优潮流（OPF） [12, 13, 48, 15, 11, 38], ，这是对任何在物理约束和工程限制条件下优化目标函数（例如发电成本）的问题的统称。最优潮流涵盖电力系统运行中长期和短期决策问题的各种公式化。

本文的主要贡献之一是展示分区对分布式优化算法收敛的重要性。对于许多问题，将其划分 为子问题有多种方式，而这种分区对分布式算法性能的影响通常尚不明确。文献中的许多论文采用基于组件的 [30], 方法，其中网络的每个组件都被视为一个计算代理。在这种设置下，大规模问题的子问题数量可能会影响分布式算法的收敛性。将基于组件的分布式算法应用于包含数千个组件的智能电网，会在收敛和实现方面带来重大问题。相比之下，我们将把问题划分为子问题的算法称为基于组的 算法 [17, 16, 22, 35]。 [17, 16] 中的分区基于一种启发式算法，该算法利用图的谱聚类，将所谓亲和度较大的节点归为一组。 [22] 通过数值实验比较了图论中的一些分区技术，并得出结论：它们的性能相当接近。作者表明，恰当的分区能够缩短运行时间。同一研究团队在后续论文 [35] 中指出，恰当的问题分区会对无功功率调度问题的收敛速度产生重大影响。 [46] 中的作者提出了一种针对直流最优潮流问题的、基于交替方向乘子法并采用分区的方法，并通过数值实验表明，分区能够减少迭代次数。本文进一步拓展了分区的用途，不仅仅是减少子问题数量，并提出

智能划分的理念，即为子问题设定一些理想的结构。智能划分的一个典型示例，是将一个非凸问题拆分为若干子问题；这些子问题虽然仍是非凸的，却可以进行精确凸松弛。与使用通用求解器相比，这种方法能够更高效地求解子问题。我们提出一种基于网络底层图的系统性径向划分 方法，使每个子问题所对应的图都构成一棵树。这种分区既减少了迭代次数，也使生成的子问题不仅规模更小，而且比原始问题更易处理。

本文的第二项主要贡献是提出了一种用于基于分组的分布式优化的自适应参数选择方法。这些参数会显著影响分布式优化算法的收敛性[30]，目前已有多种方法，例如自适应 [19, 44] 和机器学习（ML）辅助的 [47]参数选择。我们提出了一种简单的分布式一致性算法（DiCA），并将其与谱惩罚参数选择相结合，从而以更少的迭代次数获得比基于能力的算法所报告结果更准确的解。这些算法使用优化建模软件包 Pyomo[18, 8]，在 Python 中实现，并以软件包 DiCARP [20] 的形式公开提供。利用这段代码，我们研究了分区对分布式算法的迭代次数、收敛性和参数选择的影响。

本文的主要**贡献**如下：

- 提出面向分布式优化的智能划分概念及一种径向划分原型。设计一种用于径向划分的图算法。
- 设计一种用于最优潮流问题的自适应分组式分布式优化方法，并采用谱惩罚参数选择。
- 介绍一个用于在 Python 中实现算法的开源软件包 DiCARP，该软件包使用 Pyomo 优化软件包。实现数值示例以展示其性能。

1.1. 符号说明。 小写字母（如 v）表示标量变量，其粗体形式（如 $\boldsymbol{v}$）表示变量向量。索引 i 和 j 用于表示母线和线路，k 表示子问题，t 表示迭代次数。例如，$p^{k,}{}_{ij}{}^{t}$ 表示区域 k 中线路 ij 在第 t次迭代时的有功潮流。由于我们主要使用电力网络的底层图，因此节点/母线和边/线路这两组术语可以互换使用。i 是虚数单位，$\Re\{\cdot\}$，$\Im\{\cdot\}$ 分别表示实部算子和虚部算子，$\cdot^*$ 表示复共轭算子，$\overline{\cdot}$，$\underline{\cdot}$ 分别表示上界算子和下界算子。

2. Group-based Consensus Optimization

在最高层级的形式下，许多优化问题都可以写成

$$\min \sum_{i=1}^{N} f_i(x), \quad \text{s.t.} \quad x \in D, \tag{1}$$

其中，$D \subseteq \mathbb{R}^n$ ($\mathbb{R}^n$ 是维度为 n 的欧几里得向量空间），且对所有 i，$f_i : \mathbb{R}^n \to \mathbb{R}$ 。D 是由物理和工程约束（例如基尔霍夫定律和欧姆定律）定义的可行域。最优潮流问题以及电力系统中的许多其他问题具有三个重要特征，使其非常适合采用分布式算法：（i）目标函数高度分离，在许多情况下，每个 f_i 都代表网络中的一个单独组件；（ii）对于每个 i，只有 x 的某些分量参与 $f_i(x)$ 的参数；（iii）参与 $f_i(x)$ 的 x 分量也出现在定义 D 的约束中一个近乎明确的子集中。令 A_i 为 x 中参与 f_i 的分量子集，$\mathcal{A}_i$ 为提取 x 分量并返回一个向量的线性变换。然后，通过添加一些辅助变量，该优化问题可写为

$$\begin{aligned} \min \quad & \textstyle\sum_{i=1}^{N} f_i(z_i) \\ \text{s.t.} \quad & z_i = \mathcal{A}_i(x), \quad z_i \in D_i(x), \quad i \in \{1, \ldots, N\}. \end{aligned} \tag{2}$$

$D_i(x)$ 是由涉及 A_i 的一组约束所定义的集合，并且是 x 的函数，因为这些约束并未完全解耦。公式 (2) 是我们基于分组的 分布式优化的形式，其中有 N 个组，组 i 中的变量由子集 A_i 标记。实现这种分组有多种方法，而我们工作的主要创新在于以智能方式执行分组。基于分组的分布式优化方法不同于基于组件的 分布式优化算法，后者不进行分组，每个组件都作为一个计算代理。鉴于现代电力网络由数千个组件组成，管理如此多子问题对分布式算法性能的影响尚未得到充分理解。该框架可以视为本文 Boyd 等人讨论的受约束共识 优化，其中 [7], 将问题转化为适用于分布式方法的形式。[7] 是一篇关于交替方向乘子法（ADMM）的综述性文章；ADMM 是文献中占主导地位的分布式算法。ADMM 以及许多其他分布式算法都基于优化的对偶性 理论，并使用拉格朗日 对偶函数，我们将在后文介绍该函数。

3. Optimal Power Flow Problem

最优潮流是一个优化问题，其优化函数通常具有线性或二次目标函数，并受欧姆定律和基尔霍夫定律等物理约束与运行约束的限制。将网络中的母线和线路分别视为图 G 的节点和边，其中节点集为 $\mathcal{V}$，边集为 $\mathcal{L}$。对于每个边集 $\mathcal{L}$，我们将 $\mathcal{L}^t$ 定义为所有满足 $(j,\ i)$ 使得 $(i,\ j) \in \mathcal{L}$ 的集合。设 $\mathcal{G}$ 为由若干对 $(g,\ i)$ 组成的集合，其中发电机 g 与母线 i 相连。定义最优潮流问题所需的网络量列于表 1 中。假设 $f_i(P_i^g)$ 是与 $(g,\ i) \in \mathcal{G}$ 相关联的目标函数，它可以是如下形式的二次发电成本函数：$f_i(p_i^g) := c_{2,\ i}^g(p_i^g)^2 + c_{1,\ i}^g p_i^g + c_{0,\ i}^g$。于是，我们将优化问题 $OPF(G)$ 定义为如下 [30]:

$$\min \quad \sum_{(g,i)\in\mathcal{G}} f_i(p_i^g)$$

$$\text{(3a)} \quad \text{s.t.} \quad \underline{p}_i^g \le p_i^g \le \overline{p}_i^g; \quad \underline{q}_i^g \le q_i^g \le \overline{q}_i^g \qquad (g,i) \in \mathcal{G}$$

$$\text{(3b)} \quad \underline{v}_i \le v_i \le \overline{v}_i, \qquad i \in \mathcal{V}$$

$$\text{(3c)} \quad \underline{\theta}_{ij} \le \theta_i - \theta_j \le \overline{\theta}_{ij}, \qquad (i,j) \in \mathcal{L}$$

$$\text{(3d)} \quad \sum_{(g,i)\in\mathcal{G}} p_i^g - p_i^d = \sum_{j\in\mathcal{V}_i} p_{ij} + g_i^{sh} v_i^2, \qquad i \in \mathcal{V}$$

$$\text{(3e)} \quad \sum_{(g,i)\in\mathcal{G}} q_i^g - q_i^d = \sum_{j\in\mathcal{V}_i} q_{ij} - b_i^{sh} v_i^2, \qquad i \in \mathcal{V}$$

$$\text{(3f)} \quad p_{ij} = g_{ij}^c v_i^2 - g_{ij} v_i v_j \cos(\Delta\theta_{ij}) + b_{ij} v_i v_j \sin(\Delta\theta_{ij}), \quad (i,j) \in \mathcal{L}$$

$$\text{(3g)} \quad q_{ij} = b_{ij}^c v_i^2 - b_{ij} v_i v_j \cos(\Delta\theta_{ij}) - g_{ij} v_i v_j \sin(\Delta\theta_{ij}), \quad (i,j) \in \mathcal{L}$$

$$\text{(3h)} \quad p_{ji} = g_{ji}^c v_i^2 - g_{ji} v_j v_i \cos(\Delta\theta_{ji}) + b_{ji} v_j v_i \sin(\Delta\theta_{ji}), \quad (i,j) \in \mathcal{L}$$

$$\text{(3i)} \quad q_{ji} = b_{ji}^c v_i^2 - b_{ji} v_j v_i \cos(\Delta\theta_{ji}) + g_{ji} v_j v_i \sin(\Delta\theta_{ji}), \quad (i,j) \in \mathcal{L}$$

$$\text{(3j)} \quad \sqrt{p_{ij}^2 + q_{ij}^2} \le \overline{s}_{ij}; \quad \sqrt{p_{ji}^2 + q_{ji}^2} \le \overline{s}_{ij}, \qquad (i,j) \in \mathcal{L}$$

其中 $\Delta\theta_{ij} := \theta_i - \theta_j$, $g_{ij}^c = \Re\left\{\frac{Y_{ij}^* - \mathrm{i}b_{ij}^{ch}/2}{|T_{ij}|^2}\right\}$, $b_{ij}^c = \Im\left\{\frac{Y_{ij}^* - \mathrm{i}b_{ij}^{ch}/2}{|T_{ij}|^2}\right\}$, $g_{ij} = \Re\left\{\frac{Y_{ij}^*}{T_{ij}}\right\}$, $b_{ij} = \Im\left\{\frac{Y_{ij}^*}{T_{ij}}\right\}$, $g_{ji}^c = \Re\left\{Y_{ji}^* - \mathrm{i}\overline{b_{ji}^{ch}/2}\right\}$, $b_{ji}^c = \Im\left\{Y_{ji}^* - \mathrm{i}\overline{b_{ji}^{ch}/2}\right\}$, $g_{ji} = \Re\left\{\frac{Y_{ji}^*}{T_{ji}^*}\right\}$ 和 $b_{ji} = \Im\left\{\frac{Y_{ji}^*}{T_{ji}^*}\right\}$。约束 (3d)-(3e) 是潮流方程，(3f)-(3i) 来自基尔霍夫定律，(3j) 是线路潮流限值。

TABLE 1. 潮流符号

Y_{ij}	线路 ij 在 π-模型中的串联导纳（标幺值）
$v_i \angle \theta_i$	母线 i 处的复相量电压
p_{ij}, q_{ij}	沿线路 ij的有功、无功潮流
p_i^g, q_i^g	母线 i 上发电机 g 的有功功率发电、无功功率发电
p_i^d, q_i^d	母线 i 上的有功功率需求、无功功率需求
$\bar{s}_{ij}$	线路 ij 的视在功率限值
$\overline{\theta}_{ij}, \underline{\theta}_{ij}$	差值的上限、下限 母线 i 和 j的电压相角
T_{ij}	移相变压器的复变比
b_i^{sh}, g_i^{sh}	母线 i处的并联电纳、电导
b_{ij}^{ch}	π模型中线路 ij 的充电电纳

4. RADIAL PARTITIONING

最优潮流问题是非凸的，难以求解，而且通常是 NP难问题 [5, 24]。求解最优潮流问题已采用多种数值方法，包括牛顿法、线性规划和内点法 [48, 29]。分布式算法将大规模最优潮流问题划分为较小的子问题；然而，这些子问题各自具有相似的结构，只是规模较小，因此仍可能难以求解。文献中的大多数论文都需要在分布式算法中使用通用求解器来求解部分子问题。

针对最优潮流的松弛与近似已得到广泛研究（例如，参见 [34] 的全面综述）。二阶锥规划（SOCP）和半定规划（SDP）等凸松弛 [3]尤其具有吸引力，因为凸优化领域的新进展使我们能够高效、快速且高精度地求解许多类别的优化问题。SDP 以及一般而言的凸松弛有一个重要缺点：其解不一定是最优潮流问题的最优解（存在最优性差距），甚至可能不可行。多位研究者探讨了 SOCP 或 SDP 松弛对最优潮流问题精确的充分条件 [23, 25, 26, 1, 33]。

对于辐射状网络，其底层图为树，在一些温和条件下，SCOP 和 SDP 松弛的精确性已得到证明 [14, 43, 10]。这正是本文提出径向分解的主要动机。通过对具有难解最优潮流问题的一般图进行径向划分，每个子问题都是辐射状网络，或接近辐射状网络，因而可以通过通用求解器或凸松弛更高效地求解。

辐射状网络，并且可以通过通用求解器或凸松弛更高效地求解。

将图划分为树可以从不同的角度进行 [9, 39] 。我们将径向划分 定义为对节点集进行划分，使得每个分区诱导出一棵树。一个简单的径向划分是将每个节点分别置于一个独立的集合中。对于我们的应用，一个值得探讨的问题是：如何找到具有最少分区数量的径向划分？ 据我们所知，这一问题尚未得到解答，仍是一个有趣的开放性问题。在下文中，我们提出了贪心算法 1，该算法使用蛮力法进行分区，并且在我们研究的问题中表现良好。

算法 1 贪心径向划分

INPUT: 一个非空连通图 G, $k = 1$
START: **定义子图节点集** $\mathcal{V}_k = \{\}$，**一个空栈** $S = [\ ]$。
随机选择一个节点 v **在** G**中**。
$\mathcal{V}_k \leftarrow \{v\}$
将 v **的所有邻居节点推入** S **，并将** v **设为它们的父节点**。
WHILE （S 非空）：
    从 S**中弹出元素** u
    如果（ u **没有非父邻居位于** $\mathcal{V}_k$）：
        $\mathcal{V}_k \leftarrow \mathcal{V}_k \cup \{u\}$
    将 u **中不在** $\mathcal{V}_k$ **中的所有邻居节点加入** S
保存 $\mathcal{V}_k$ **为一个子图**
$G \leftarrow G \backslash \mathcal{V}_k$
IF（G **非空**）：
    $k \leftarrow k + 1$，**转到起始位置**
END: **返回所有已保存的子图**

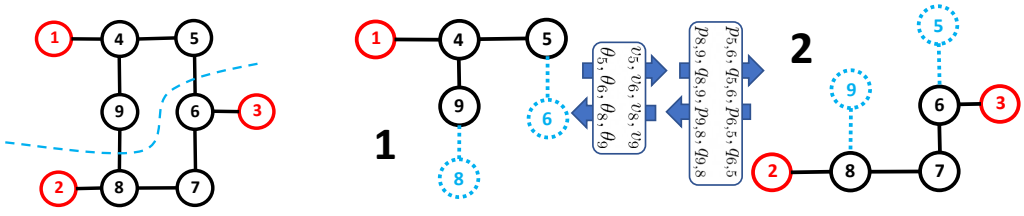
定理 4.1。 算法 *1* 将一个连通图划分为若干子图，每个子图都诱导出一棵树。

证明。 对于加入集合 $\mathcal{V}_k$ 的每个节点，其尚未位于 $\mathcal{V}_k$ 中的邻居节点都会被推入 S。反设 WHILE 循环结束后的 $\mathcal{V}_k$ 所诱导的子图包含一个环。显然，当 $\mathcal{V}_k$ 只包含一个节点时，该子图是一棵树。考虑 WHILE 循环执行过程中 $\mathcal{V}_k$ 首次诱导出环的时刻。只有在新加入的 u 除了其父节点之外，在 $\mathcal{V}_k$ 中还有一个邻居节点时才会发生这种情况，这与 WHILE 循环内的 IF 条件矛盾。□

TABLE 2. 不同问题算例的径向划分区域数量

Case	9	14	39	89	118	300
区域数量	2	3	7	10	23	36

表2展示了我们针对 MATPOWER 库 [49] 中的问题所用贪心算法生成的子问题数量。图 1 展示了问题 case9 的图的径向划分。



(A) (B)

FIGURE 1. (a) MATPOWER 库中问题算例 case9 的图。红色节点为含发电机的母线。虚线表示将图径向划分为两个分区。(b) 两个子图的节点集；二者均构成树。虚线和节点同时出现在两个区域中。所示变量是每个区域在 DiCA 中需要发送给另一个区域的变量。

5. DiCA

在本节中，我们假设给定了一个最优潮流问题，并且已经对网络的图进行了划分，从而得到 P 子问题[1]。我们定义 $\mathcal{V}_k$ 为区域 k的节点集。假设 $N(\mathcal{V}_k)$ 是 $\mathcal{V}_k$ 与 $\mathcal{V}_k$中各节点的所有邻居节点的并集。对于一个节点集 $\mathcal{V}_k$，我们还将 $G(\mathcal{V}_k)$ 定义为由节点$\mathcal{V}_k$诱导的 G 的子图。对于 DiCA 算法，每个代理都求解一个优化问题，其约束集与图 G_k 的最优潮流问题具有相同的约束：$= G(N(\mathcal{V}_k))$，但目标函数经过了修改。

我们定义 $\mathcal{V}_k$、 $\mathcal{L}_k$和 $\mathcal{G}_k$ 分别为 G_k的节点集、线路集和发电机集。对于出现在多个 $G(N(\mathcal{V}_k))$ 中的每个节点或线路，我们为其分配一个辅助变量 β，作为该变量的参考值。我们还为每个区域中的该变量分配一个对偶变量 y^k 。例如，在图 1-(b) 中，有四个节点和四条线路出现在

[1]我们的 DiCA 算法适用于任何分区，而不仅仅是径向划分。

两个区域。对于节点 5，我们为区域 1 定义辅助变量 β_{v_5} 和 β_{θ_5}，以及对偶变量$y^1_{v_5}$ 和 $y^1_{\theta_5}$；为区域 2 定义 $y^2_{v_5}$ 和 $y^2_{\theta_5}$。对于每个节点 i，我们将 $\mathcal{M}_i$ 定义为由所有满足 G_k 包含 i的 k 构成的¯集合。类似地，对于每条线路 $e = ij$，我们将 $\mathcal{M}_e$ 定义为由所有满足 G_k 包含 e的 k 构成的集合¯。对于每个满足 $|\mathcal{M}_i| > 1$的节点 i，每个区域都保留一份相应变量的副本。例如，在图 1-(b) 中，对于 $i = 9$，区域 1 有一个变量 v^1_9，区域 2 有一个变量 v^2_9。对于每条满足 $|\mathcal{M}_e| > 1$的线路 e，情况也类似。

设 $\boldsymbol{v}$、$\boldsymbol{\theta}$、$\boldsymbol{p}$、$\boldsymbol{q}$ 和 $\boldsymbol{p}^G$ 为向量，分别包含给定区域的 v_i、θ_i、p_{ij}、q_{ij} 和 p^g_i；设 $\boldsymbol{w}$ 为将这五个向量堆叠而成的向量。换言之，$\boldsymbol{w}$ 是子问题中所有变量构成的向量。类似地，我们定义 $\boldsymbol{\beta}$ 和 $\boldsymbol{y}$。那么，区域 k 的 拉格朗日对偶 为

$$L_k(\boldsymbol{w}, \boldsymbol{\beta}, \boldsymbol{y}) := \sum_{(g,i)\in\bar{\mathcal{G}_k}} f_i(p^g_i) + \sum_{i\in\bar{\mathcal{V}_k}, |\mathcal{M}_i|>1} \sum_{x\in\{v,\theta\}} y_{x_i}(x_i - \beta_{x_i}) + \frac{\rho^k_{x_i}}{2}(x_i - \beta_{x_i})^2$$
$$+ \sum_{ij\in\bar{\mathcal{L}_k}\cup\bar{\mathcal{L}_k}^t, |\mathcal{M}_{ij}|>1} \sum_{x\in\{p,q\}} y_{x_{ij}}(x_{ij} - \beta_{x_{ij}}) + \frac{\rho^k_{x_{ij}}}{2}(x_{ij} - \beta_{x_{ij}})^2$$

算法 2 描述了 DiCA 算法。在第一阶段（也是计算量最大的阶段），我们以 L_k 为目标函数，对每个区域求解最优潮流，以更新 $\boldsymbol{w}$。在第二和第三阶段，算法分别通过简单的代数运算更新 $\boldsymbol{\beta}$ 和 $\boldsymbol{y}$。第四阶段用于更新惩罚参数。如图所示，变量 β 没有区域索引 k，因为这些变量独立于各区域，并且是局部副本 同意 与之相等的参考变量。

该算法是对 [7]- 第 7 章中所述共识交替方向乘子法的简单推广，数值表现颇具前景。与文献中许多用于最优潮流的分布式优化算法相比，该算法的结构更简单，从而简化了关键的参数调优。对于停止准则，每个代理都会检查自己的原始残差和对偶残差是否小于给定的 ϵ。这比需要由集中式单元计算停止准则的情况更加实用。对于每个代理 k，在迭代 t 时，这些残差向量的计算如下，它们是.....的推广

算法 2 分布式共识算法（DiCA）

INPUT: 图 G 被划分为 P 个子图的START: 最优潮流问题。设置 $t=0$。初始化 $\boldsymbol{\beta}^0$、 $\boldsymbol{y}^{k,0}$s 和 $\boldsymbol{\rho}^{k,0}$。WHILE （未满足停止准则）：

FOR $k=1$、...、 P求解:

$$\begin{aligned}\boldsymbol{w}^{k,t+1} \leftarrow \quad & \operatorname{argmin}_{\boldsymbol{w}} \qquad L_k(\boldsymbol{w}, \boldsymbol{\beta}^t, \boldsymbol{y}^{k,t}) \\ & \text{s.t.} \qquad \text{(3a)-(3j) for } \bar{\mathcal{V}}_k, \bar{\mathcal{L}}_k, \text{ and } \bar{\mathcal{G}}_k\end{aligned}$$

FOR 节点及其 $|\mathcal{M}_i|>1$ ，边及其 $|\mathcal{M}_{ij}|>1$:

$$\begin{aligned}\beta_{x_i}^{t+1} &\leftarrow \tfrac{1}{|\mathcal{M}_i|}\textstyle\sum_{k\in\mathcal{M}_i}(x_i^{k,t+1}+\tfrac{1}{\rho_x}y_{x_i}^{k,t}), \quad x\in\{v,\theta\} \\ \beta_{x_e}^{t+1} &\leftarrow \tfrac{1}{|\mathcal{M}_e|}\textstyle\sum_{k\in\mathcal{M}_e}(x_e^{k,t+1}+\tfrac{1}{\rho_x}y_{x_e}^{k,t}), \quad e=ij,ji, x\in\{p,q\}\end{aligned}$$

FOR 节点及其 $|\mathcal{M}_i|>1$ ，边及其 $|\mathcal{M}_{ij}|>1$:

$$\begin{aligned}y_{x_i}^{k,t+1} &= y_{x_i}^{k,t}+\rho_{x_i}^{k,t}(x_i^{k,t+1}-\beta_{x_i}^{t+1}), \quad x\in\{v,\theta\} \\ y_{x_e}^{k,t+1} &= y_{x_e}^{k,t}+\rho_{x_e}^{k,t}(x_e^{k,t+1}-\beta_{x_e}^{t+1}), \quad e=ij,ji, x\in\{p,q\}\end{aligned}$$

FOR 节点及其 $|\mathcal{M}_i|>1$ ，边及其 $|\mathcal{M}_{ij}|>1$，更新 $\rho^{k,t+1}_{x_i}$和 $\rho^{k,t+1}_{x_e}$。$t\leftarrow t+1$

一般交替方向乘子法方法在 [7]中的残差:

$$\boldsymbol{r}^{k,t}:=\boldsymbol{w}^{k,t}-\boldsymbol{\beta}^{k,t}, \quad \boldsymbol{d}^{k,t}:=\boldsymbol{\beta}_{\boldsymbol{\rho}}^{k,t}-\boldsymbol{\beta}_{\boldsymbol{\rho}}^{k,t-1},$$

$$\boldsymbol{w}^{k,t}:=\begin{bmatrix}\mathbf{v}^{k,t}\\ \boldsymbol{\theta}^{k,t}\\ \boldsymbol{p}^{k,t}\\ \boldsymbol{q}^{k,t}\end{bmatrix}_{|\mathcal{M}|>1} \quad \boldsymbol{\beta}^{k,t}:=\begin{bmatrix}\boldsymbol{\beta}_{\boldsymbol{v}}^t\\ \boldsymbol{\beta}_{\boldsymbol{\theta}}^t\\ \boldsymbol{\beta}_{\boldsymbol{p}}^t\\ \boldsymbol{\beta}_{\boldsymbol{q}}^t\end{bmatrix}_k \quad \boldsymbol{\beta}_{\boldsymbol{\rho}}^{k,t}:=\begin{bmatrix}\rho_v\boldsymbol{\beta}_{\boldsymbol{v}}^t\\ \rho_\theta\boldsymbol{\beta}_{\boldsymbol{\theta}}^t\\ \rho_p\boldsymbol{\beta}_{\boldsymbol{p}}^t\\ \rho_q\boldsymbol{\beta}_{\boldsymbol{q}}^t\end{bmatrix}_k \quad \boldsymbol{y}^{k,t}:=\begin{bmatrix}\boldsymbol{y}_{\boldsymbol{v}}^{k,t}\\ \boldsymbol{y}_{\boldsymbol{\theta}}^{k,t}\\ \boldsymbol{y}_{\boldsymbol{p}}^{k,t}\\ \boldsymbol{y}_{\boldsymbol{q}}^{k,t}\end{bmatrix}.$$

我们使用记号 $[\cdot]_{|\mathcal{M}|>1}$ 来表示，计算残差向量时仅使用节点 i 和线路 e ，且满足 $|\mathcal{M}_i|>1$ 和$|\mathcal{M}_e|>1$ 。此外， $\boldsymbol{\beta}^{k,t}$ 和 $\boldsymbol{\beta}^{k,t}_{\boldsymbol{\rho}}$定义为包含区域 k中涉及的变量 βs。然后，代理 k的停止准则（容差为 ϵ ）如下:

$$\|\boldsymbol{r}^{k,t}\|_2 \le \epsilon\max\{\|\boldsymbol{w}^{k,t}\|_2, \|\boldsymbol{\beta}^{k,t}\|_2\}, \quad \|\boldsymbol{d}^{k,t}\|_2 \le \epsilon\|\boldsymbol{y}^{k,t}\|_2 \tag{4}$$

当所有区域均满足 (4) 时，DiCA 停止。

6. Spectral Penalty Parameter

针对原始交替方向乘子法的谱自适应参数选择方法最初见于 [44] ，随后在 [45]中扩展至共识交替方向乘子法。 [44]中提出的方法基于 Barzilai 和 Borwein [2] 提出的一种经典参数选择方法；在使用*Douglas-Rachdord* 分裂（*DRS*）求解对偶问题时，该方法应用于对偶问题。由于在原始交替方向乘子法中，对偶问题的 DRS 等价于对原问题应用交替方向乘子法，因此我们得到了原问题的参数选择方法。 [31] 中的作者对这些结果进行了调整，以设计用于最优潮流的自适应基于组件的交替方向乘子法算法。本节介绍用于分组算法的谱自适应参数选择方法。对于每个节点 i ，其中 $|\mathcal{M}_i| > 1$、 ρ_{v_i} 和 ρ_{θ_i} 均采用自适应方式更新。同样，对于每条边 e ，其中 $|\mathcal{M}_e| > 1$、 ρ_{p_e} 和 ρ_{q_e} 均采用自适应方式更新。对于变量 x_i，定义 $\bar{y}^{k,}{}_{x_i}{}^{t+1} = \bar{y}^{k,}{}_{x_i}{}^{t} + \rho^{k,}{}_{x_i}{}^{t}(x^{k,}{}_{i}{}^{t+1} - \beta_{x_i}^{t+1})$。然后，定义

$$\alpha_{SD,x_i}^{k,t} = \frac{\sum_{k\in\mathcal{M}_i}(\Delta\bar{y}_{x_i}^{k,t})^2}{\sum_{k\in\mathcal{M}_i}(\Delta\bar{y}_{x_i}^{k,t})(\Delta x_i^{k,t})},\ \alpha_{MG,x_i}^{k,t} = \frac{\sum_{k\in\mathcal{M}_i}(\Delta\bar{y}_{x_i}^{k,t})(\Delta x_i^{k,t})}{\sum_{k\in\mathcal{M}_i}(\Delta x_i^{k,t})^2}, \tag{5}$$

其中，SD 表示最速下降，MG 表示最小梯度 [44], ，分别代表用于计算海森矩阵近似中线性项系数的两种不同方法。与 [45], 类似，我们将这两种估计量结合使用，得到

$$\alpha_{x_i}^{k,t} := \begin{cases} \alpha_{MG,x_i}^{k,t} & \text{if } 2\alpha_{MG,x_i}^{k,t} > \alpha_{SD,x_i}^{k,t} \\ \alpha_{SD,x_i}^{k,t} - \alpha_{MG,x_i}^{k,t}/2 & \text{o.w.} \end{cases} \tag{6}$$

为确保自适应更新的安全性，我们可以检查相关性 n

$$\alpha_{c,x_i}^{k,t} := \frac{\sum_{k\in\mathcal{M}_i}(\Delta\bar{y}_{x_i}^{k,t})(\Delta x_i^{k,t})}{\sqrt{\sum_{k\in\mathcal{M}_i}(\Delta\bar{y}_{x_i}^{k,t})^2\sum_{k\in\mathcal{M}_i}(\Delta x_i^{k,t})^2}}, \tag{7}$$

有正的下界。我们还使用相同的公式定义 $\beta^{k,}{}_{SD,x_i}{}^{t}$, $\beta^{k,}{}_{MG,x_i}{}^{t}$, $\beta^{k,}{}_{x_i}{}^{t}$ 和 $\beta^{k,c}{}_{,x_i}{}^{t}$ ，方法是将 $\Delta\bar{y}^{k,}{}_{x_i}{}^{t}$ 替换为 $\Delta y^{k,}{}_{x_i}{}^{t}$，并将 $\Delta x^{k,}{}_{i}{}^{t}$ 替换为 $\Delta\beta_{x_i}^{t}$。利用上述估计并采取保护措施，最终的自适应公式为

$$\hat{\rho}_{x_i}^{k,t+1} := \begin{cases} \sqrt{\alpha_{x_i}^{k,t}\beta_{x_i}^{k,t}} & \text{if } \alpha_{c,x_i}^{k,t} > \epsilon_c \text{ and } \beta_{c,x_i}^{k,t} > \epsilon_c \\ \alpha_{x_i}^{k,t} & \text{if } \alpha_{c,x_i}^{k,t} > \epsilon_c \text{ and } \beta_{c,x_i}^{k,t} \le \epsilon_c \\ \beta_{x_i}^{k,t} & \text{if } \alpha_{c,x_i}^{k,t} \le \epsilon_c \text{ and } \beta_{c,x_i}^{k,t} > \epsilon_c \\ \hat{\rho}_{x_i}^{k,t} & \text{if } \alpha_{c,x_i}^{k,t} \le \epsilon_c \text{ and } \beta_{c,x_i}^{k,t} \le \epsilon_c \end{cases}$$

$$\rho_{x_i}^{k,t+1} = \max\{\min\{\hat{\rho}_{x_i}^{k,t+1}, U_{x_i}\}, L_{x_i}\}, \quad x \in \{v, \theta\}, \tag{8}$$

其中， U_{x_i} 和 L_{x_i} 分别是惩罚参数的上界和下界。对于每条链路 e，只要满足 $|\mathcal{M}_e| > 1$，我们也可以类似地计算 ρ_{p_e} 和 ρ_{q_e}。

7. Numerical Results

本文为所提出的算法开发了开源软件包 DiCARP [20]。这些算法使用优化建模软件包 Pyomo [18, 8], 和后端求解器 Ipopt [40]以 Python 实现。Ipopt 是一款用于大规模非线性优化的开源软件包。我们的 DiCARP 软件包包含一种可对给定图应用贪心径向划分的方法。我们求解的问题实例来自 MATPOWER 库 [49]。我们提供了一个函数，可将 MATPOWER 算例转换为 Pyomo 的数据文件格式。我们在一台配备 32GB 内存、主频为 1.7 GHz 的第 12 代 Intel Core i7 个人电脑上进行了计算实验。在本节中，我们将 Ipopt 返回的解称为 P_{IPM}，将 DiCA 返回的解称为P_{DiCA}。为比较这两个量，类似于 [30], ，我们定义最优性 GAP 如下：

$$GAP = |P_{IPM} - P_{DiCA}| / P_{IPM} \tag{9}$$

对于算法 2 中惩罚参数的更新，我们采用第 6 节所述的谱参数选择方法。由于该方法具有自适应性，与非自适应 DO 相比，算法对惩罚参数初始值的敏感度较低。数值实验中，对于所有满足 $|\mathcal{M}_i| > 1$ 的节点，我们设置 $\rho^{k,\ {}^{0}_{v_i}} = \rho^{k,\ {}^{0}_{\theta_i}} = 10^4$；对于所有满足 $|\mathcal{M}_e| > 1$ 的边，我们设置$\rho^{k,\ {}^{0}_{p_e}} = \rho^{k,\ {}^{0}_{q_e}} = 10^3$。

关于电力网络的基于组件的 DO，最全面的数值结果分别为：非自适应情形下的 [30]，以及自适应 DO 下的 [31]。表 3 展示了求解 MATPOWER 库中问题的结果，这些结果也曾在 [30] 中报告。如表中所示，基于分组的 DiCA 无需调整参数，便能以少得多的迭代次数返回更准确的解。

表 4 报告了在 [31] 中使用自适应基于组件的 DO 求解较大实例时采用 DiCA 的结果。与自适应基于组件的算法相比，DiCA 算法表现更好。不过，随着径向划分产生的分区数量增加，两者的性能差距会缩小。我们的数值结果与有关分区的有限文献相一致，表明减少子问题的数量可以显著减少迭代次数。我们没有在表格中报告运行时间，因为使用单台机器时，IPM 的速度快得多，

TABLE 3. DiCA 算法在 MATPOWER 算例上的性能

案例迭代		P_{DiCA}	P_{IPM}	GAP	迭代次数 [30]	间隙 [30]
5	248	17551.89	17551.89	4.51 e-9	3911	4.14e-5
6ww	64	3143.97	3143.97	2.12e-8	918	4.94e-5
9	44	5296.68	5296.68	1.13e-08	630	7.25e-5
14	72	8081.52	8081.52	3.53e-08	857	7.88e-5
24	115	63352.20	63352.20	2.38e-08	924	2.05e-06
30	532	576.89	576.89	7.74e-07	2763	1.34e-4
39	342	41864.14	41864.18	1.28 e-08	7468	9.82e-07
57	232	41737.79	41737.78	2.39e-07	1305	4.24e-05
118	215	129660.81	129660.69	9.25e-07	1168	3.66e-6
300	684	719725.54	719725.09	6.25e-07	11755	8.72e-7

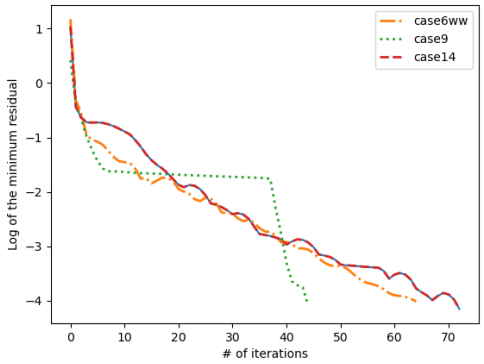
TABLE 4. DiCA 算法在 MATPOWER 算例上的性能，这些算例也使用自适应基于组件的 DO 在 [31] 中求解。

Case	iter	GAP	迭代次数 [31]	间隙 [31]
1354 PEGASE	753	6.75 e-7	1110	6.43e-6
2383 wp	1740	7.81e-7	4070	1.95e-5
2736 sp	1212	5.42e-7	2154	1.33e-6
2746 wp	986	3.21e-6	1872	5.09e-5

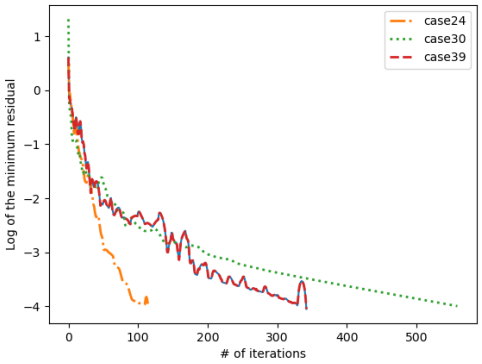
对于实验而言，运行时间并不能提供有意义的信息。只有在数据隐私要求使用多个智能体，或问题规模大到通用求解器（如 Ipopt）无法处理时，才有必要采用分布式优化（DO）。针对超大型现代电力网络的这些问题，将是未来的研究主题。

图2展示了各区域的最小残差随迭代次数变化的曲线。如 [30], 所述，残差进展过程中的振荡是 DO 收敛缓慢的原因之一。可以看出，在几乎所有示例中，采用我们的 DiCA 算法时，曲线中都没有出现振荡。

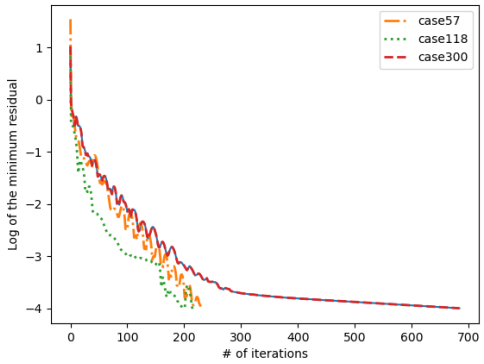
采用径向划分的好处之一是简化子问题的结构。衡量这一点的一种方法，是统计使用通用内点求解器求解问题所需的迭代次数。这种方法之所以可行，是因为在实践中，内点法的迭代次数与问题规模无关，而更多地取决于其结构



(A)



(B)



(C)

FIGURE 2.MATPOWER 库中不同算例的区域最小残差与迭代次数的关系：(a) case6ww、case9 和 case14；(b) case24、case30 和 case39；(c) case57、case118 和 case300。

TABLE 5. Ipopt 的迭代次数

Case	9	14	30	57	118	300
主问题	30	48	69	53	88	138
子问题平均值	25.5	25.3	23.8	25.6	30.3	33.4

。换言之，越接近病态的问题，使用内点法求解时所需的迭代次数越多。表 5 展示了使用 Ipopt 求解某些问题实例时的迭代次数，以及子问题的平均迭代次数，后者显著更少。

8. Conclusion

在这项工作中，我们提出了一种分区方法，旨在减少子问题的数量，并为其施加理想的结构。数值结果表明，在我们提出的径向划分基础上应用 DiCA 算法，并采用谱参数选择，与自适应基于组件的算法相比，能够以更少的迭代次数获得更准确的解。谱参数选择减轻了复杂参数调优的需求。

在径向划分以及更广泛的智能划分方面，仍有许多未解问题。我们观察到，使用通用内点求解器时，子问题的迭代次数平均不到主问题的一半。这表明径向划分简化了子问题的结构。下一步是利用当前关于径向网络凸松弛的研究成果（例如 [14, 43, 10]），证明子问题存在精确凸松弛。另一项未来工作是设计不同于我们所提贪心算法的径向划分方法，以更有效地考虑电力网络结构。数值结果还显示，径向划分所生成的分区数量会随着问题规模增大而增加，这使得超大规模问题再次面临收敛缓慢的问题。一个有趣的开放问题是，如何找到一种智能划分方法，即使对于大型问题实例也能生成极少数量的分区。对于参数选择，下一步值得研究的是不同的自适应参数选择方法或基于机器学习的参数选择方法。

9. 致谢

作者的研究得到了美国国家科学基金会（NSF）拨款 CMMI-2347120 的资助。

References

[1] X. Bai, H. Wei, K. Fujisawa, and Y. Wang，用于最优潮流问题的半定规划，《国际电力与能源系统杂志》，30 (2008)，第 383–392 页。[2] J. Barzilai and J. M. Borwein，两点步长梯度法，《IMA 数值分析杂志》，8 (1988)，第 141–148 页。[3] A. Ben-Tal and A. Nemirovski，《现代凸优化讲义：分析、算法与工程应用》，SIAM，2001。

[4] D. Bertsekas and J. Tsitsiklis，并行与分布式计算：数值方法，*Athena Scientific*，*2015*。[5] D. Bienstock and A. Verma，交流潮流可行性的强 *NP* 难性，《运筹学快报》，*47 (2019)*，第 *494–501* 页。[6] S. Boyd, S. P. Boyd, and L. Vandenberghe，凸优化，剑桥大学出版社，*2004*。[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al.，通过交替方向乘子法实现分布式优化与统计学习，《机器学习基础与趋势》，*3 (2011)*，第 *1–122* 页。[8] M. L. Bynum, G. A. Hackebeil, W. E. Hart, C. D. Laird, B. L. Nicholson, J. D. Siirola, J.-P. Watson, and D. L. Woodruff，*Pyomo*——用于 *Python* 的优化建模，第 *67* 卷，施普林格科学与商业媒体，第三版，*2021*。[9] F. Chung，图的树划分，《离散数学》，*23 (1978)*，第 *23–30* 页。[10] M. Farivar and S. H. Low，支路潮流模型：松弛与凸化——第一部分，*IEEE* 电力系统汇刊，*28 (2013)*，第 *2554–2564* 页。[11] S. Frank, S. Rebennack, et al.，最优潮流入门：理论、公式化与实践示例，科罗拉多矿业学院，技术报告，*(2012)*。[12] S. Frank, I. Steponavice, and S. Rebennack，最优潮流：文献综述 *I*：公式化与确定性方法，《能源系统》，*3 (2012)*，第 *221–258* 页。[13] , 最优潮流：文献综述 *II*：非确定性方法与混合方法，《能源系统》，*3 (2012)*，第 *259–289* 页。[14] L. Gan, N. Li, U. Topcu, and S. H. Low，辐射状网络中最优潮流的精确凸松弛，《*IEEE*自动控制汇刊》，*60 (2014)*，第 *72–87* 页。[15] J. D. Glover, M. S. Sarma, and T. Overbye，电力系统分析与设计，圣智学习，*2012*。[16] J. Guo, G. Hug, and O. K. Tonguz，电力系统分布式优化中的智能划分，*IEEE* 智能电网汇刊，*7 (2015)*，第 *1249–1258* 页。[17] , 大型电力系统采用非凸分布式优化的理由，*IEEE* 电力系统汇刊，*32 (2016)*，第 *3842–3851* 页。[18] W. E. Hart, J.-P. Watson, and D. L. Woodruff，*Pyomo*：使用 *Python* 对数学规划进行建模与求解，《数学规划计算》，*3 (2011)*，第 *219–260* 页。[19] B. He, H. Yang, and S. Wang，用于单调变分不等式的自适应惩罚参数交替方向法，《优化理论与应用杂志》，*106 (2000)*，第 *337–356* 页。[20] M. Karimi，*DiCARP*。[21] A. D. Kshemkalyani and M. Singhal，分布式计算：原理、算法与系统，剑桥大学出版社，*2011*。[22] M. Kyesswa, A. Murray, P. Schmurr, H. Çakmak, U. Kühnapfel, and V. Hagenmeyer，电网划分算法对组合式分布式交流最优潮流与并行动态电网仿真的影响，《*IET*发电、输电与配电》，*14 (2020)*，第 *6133–6141* 页。

[23] J. Lavaei and S. H. Low, 最优潮流问题中的零对偶间隙，*IEEE* 电力系统汇刊，*27*（*2011*），第*92–107*页。[24] K. Lehmann, A. Grastien, and P. Van Hentenryck, 树形网络上的交流可行性是*NP*难问题，*IEEE* 电力系统汇刊，*31*（*2015*），第*798–801*页。[25] S. H. Low, 最优潮流的凸松弛——第一部分：公式化与等价性，*IEEE*网络系统控制汇刊，*1*（*2014*），第*15–27*页。[26] , 最优潮流的凸松弛——第二部分：精确性，*IEEE*网络系统控制汇刊，*1*（*2014*），第*177–189*页。[27] N. A. Lynch, 分布式算法，*Elsevier*，*1996*。[28] A. Maneesha and K. S. Swarup, 交替方向乘子法在智能电网中的应用综述，《可再生与可持续能源评论》，*152*（*2021*），第*111687*页。[29] D. Mehta, D. K. Molzahn, and K. Turitsyn, 潮流方程计算方法的最新进展，载于*2016*年美国控制会议（*ACC*），*IEEE*，*2016*，第*1753–1765*页。[30] S. Mhanna, A. C. Chapman, and G. Verbič, 用于最优潮流问题的基于组件的对偶分解方法，《可持续能源、电网与网络》，*16*（*2018*），第*91–110*页。[31] S. Mhanna, G. Verbič, and A. C. Chapman, 用于分布式交流最优潮流的自适应交替方向乘子法，*IEEE* 电力系统汇刊，*34*（*2019*），第*2025–2035*页。[32] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, andJ. Lavaei, 电力系统分布式优化与控制算法综述，*IEEE* 智能电网汇刊，*8*（*2017*），第*2941–2962*页。[33] D. K. Molzahn and I. A. Hiskens, 最优潮流问题中利用稀疏性的基于矩的松弛，*IEEE* 电力系统汇刊，*30*（*2014*），第*3168–3180*页。[34] ，潮流方程的松弛与近似方法综述，《电能系统基础与趋势》，*4*（*2019*），第*1–221*页。[35] A. Murray, M. Kyesswa, H. Çakmak, and V. Hagenmeyer, 面向无功功率调度分布式优化的最优电网划分，载于*2021*年*IEEE*电力与能源学会创新智能电网技术会议（*ISGT*），*IEEE*，*2021*，第*1–5*页。[36] J. Nocedal and S. J. Wright, 数值优化，*Springer*，*1999*。[37] N. Patari, V. Venkataramanan, A. Srivastava, D. K. Molzahn, N. Li, and A. Annaswamy, 配电系统中的分布式优化：应用场景、局限性与研究需求，*IEEE* 电力系统汇刊，*37*（*2021*），第*3469–3481*页。[38] J. K. Skolfield and A. R. Escobedo, 最优潮流中的运筹学：近期及新兴方法与应用指南，《欧洲运筹学杂志》，（*2021*）。[39] J. A. Telle and A. Proskurowski, 部分*k*树上顶点划分问题的算法，《*SIAM*离散数学杂志》，*10*（*1997*），第*529–550*页。[40] A. Wächter and L. T. Biegler, 大规模非线性规划中内点滤波线搜索算法的实现，《数学规划》，*106*（*2006*），第*25–57*页。[41] H. Wang, H. Li, and B. Zhou, 分布式优化、博弈与学习算法：智能电网系统中的理论与应用，*Springer Nature*，*2021*。[42] Y. Wang, S. Wang, and L. Wu, 面向新兴电力系统运行的分布式优化方法：综述，《电力系统研究》，*144*（*2017*），第*127–135*页。

[43] W. Wei, J. Wang, N. Li, and S. Mei, 径向网络的最优潮流及其变体：一种序列凸优化方法，*IEEE* 智能电网汇刊，*8 (2017)*，第 *2974–2987* 页。[44] Z. Xu, M. Figueiredo, and T. Goldstein, 采用谱惩罚参数选择的自适应交替方向乘子法，载于《人工智能与统计》，*PMLR*，*2017* 年，第 *718–727* 页。[45] Z. Xu, G. Taylor, H. Li, M. A. Figueiredo, X. Yuan, and T. Goldstein, 用于分布式优化的自适应共识交替方向乘子法，载于《国际机器学习大会》，*PMLR*，*2017* 年，第 *3841–3850* 页。[46] L. Yang, J. Luo, Y. Xu, Z. Zhang, and Z. Dong, 一种基于分区的分布式对偶共识交替方向乘子法，用于考虑碳排放交易的直流最优潮流，*IEEE* 工业信息学汇刊，*16 (2019)*，第 *1858–1872* 页。[47] S. Zeng, A. Kody, Y. Kim, K. Kim, and D. K. Molzahn, 一种用于电力系统分布式优化参数选择的强化学习方法，*arXiv* 预印本 *arXiv:2110.11991*，*(2021)*。[48] J. Zhu, 电力系统运行优化，约翰·威立父子出版公司，*2015* 年。[49] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, *MATPOWER*：用于电力系统研究与教育的稳态运行、规划和分析工具，*IEEE* 电力系统汇刊，*26 (2010)*，第 *12–19* 页。