

基于分解协调模型的无功优化算法对比分析

针对现有基于分解协调模型的无功优化算法,为明确其特点差异及优劣,在求解思路、寻优方式、通信协调模式和每次迭代所需的数据通信量等方面对其进行了详细的分析对比。以IEEE 118节点系统4种不同分区情况和IEEE 300节点系统为例,对寻优方式相同的邻近中心算法和全分邻近中心算法在计算中所用的系数矩阵以及现有算法的通信协调模式和计算结果进行了比较分析。

明 娇 李 智/国网德阳供电公司
张安安/西南石油大学电气信息学院

0 引言

无功优化是保证电力系统安全稳定和经济运行的重要手段。电网是相互联系、规模庞大的整体,集中式无功优化存在的计算速度慢和数据传输瓶颈等问题使其不能用于在线实时控制;各分层分区控制中心独立的无功优化可满足局部在线实时控制要求,但不能求得全网最优解,因此有众多学者在研究如何解决该问题。文献[1-3]基于协同进化的智能优化算法实现了无功优化的并行分解计算。文献[4-6]将传统的等值方法用于外部关联网络的等值,并在此基础上实现了全网无功优化的分解协调计算。文献[7-8]基于矩阵解耦或矩阵结构实现了无功优化分解协调计算。文献[9]实现了基于内点法的无功优化分解协调计算,基于分解协调模型的无功优化算法通过求解根据电网分解思想建立的无功优化分解协调模型,实现了无功优化分解协调计算。文献[10-14]将不同的数学原理或方法用于无功优化分解协调模型的求

解,都有各自的特点,但目前尚未有文献对这些求解算法进行综合对比。

为了明确现有基于分解协调模型的无功优化算法的特点差异及优劣,从而为无功优化分解协调模型求解算法的后续研究提供参考,本文从求解思路、寻优方式、通信协调模式和数据通信量等方面对其进行了综合对比,并通过算例对其进行了比较分析。

1 无功优化分解协调模型

基于电网分解思想^[15],文献[10-11]提出包含 Δ -变量的多分区无功优化分解协调模型为

$$\min_{(x_i, x_{bi}) \in X_i^f} f(x) = \sum_{i=1}^n f_i(x_i, x_{bi}) \quad (1)$$

$$\text{s. t. } \phi(x_b, \Delta\delta) = (x_{bij} + v_{ij}\Delta\delta_i) - (x_{bji} + v_{ji}\Delta\delta_j) = 0$$

式中, $i=1, \dots, n, j=i+1, \dots, n$, n 为分区个数; x_i 、 x_{bi} 分别为分区I的内部变量和边界变量; X_i^f 为分区I所有变量的可行性约束集; $x_{bij} = (P_{bij} \ Q_{bij} \ U_{bij} \ \theta_{bij})^T$ 和 $x_{bji} = (-P_{bji} \ -Q_{bji} \ U_{bji} \ \theta_{bji})^T$ 为分区I与分区J间边界节点分别在分区I和分区J内注入有功、注入无功、电压幅值和电压相角构成的向量; $v_{ij} = v_{ji} = (0 \ 0 \ 0 \ 1)^T$, 维数与 x_{bij} 和 x_{bji} 一致; $\Delta\delta_i$ 、 $\Delta\delta_j$ 分别为分区I和分区J的平衡节点相对于全网平衡节点的电压相角差。

2 基于分解协调模型的无功优化算法

2.1 求解思路

对于类似式(1)的带约束优化问



明 娇/工程师

关键词/Keywords

无功优化 ·
分解协调 ·
辅助问题原理 ·
分块坐标下降 ·
邻近中心算法 ·

国家自然科学基金(51107107)资助项目;中国博士后基金面上项目(2014M562335)资助项目。

题, 可通过经典拉格朗日松弛方法或增广拉格朗日松弛方法将其转化为鞍点问题进行求解。但是, 经典拉格朗日松弛方法的对偶函数可能不可微, 并且可能由于存在对偶间隙而无法求得可行解, 因此常用增广拉格朗日松弛方法求解。然而, 增广拉格朗日松弛方法引进的二次项是不可分的, 从而不能直接对优化问题进行分解计算。

文献[10-11]和文献[12]分别采用辅助问题原理 (Auxiliary Problem Principle, APP) 和分块坐标下降 (Block Coordinate Descent, BCD) 思想解决增广拉格朗日函数的不可分问题并对其进行求解, 从而实现无功优化分解协调模型的求解。文献[13]通过邻近函数构造平滑同时可分的拉格朗日函数, 并基于估计序列理念通过最优梯度更新拉格朗日乘子, 最终实现了基于邻近中心算法 (Proximal Center Algorithm, PCM) 的模型求解算法。文献[14]在常规邻近中心算法的基础上, 针对式 (1) 约束的特殊结构, 提出了可实现所有步骤分解计算的全分邻近中心算法 (Full Decomposition Proximal Center Algorithm, FDPCM) 用于其求解。各算法的求解思路如图1所示。

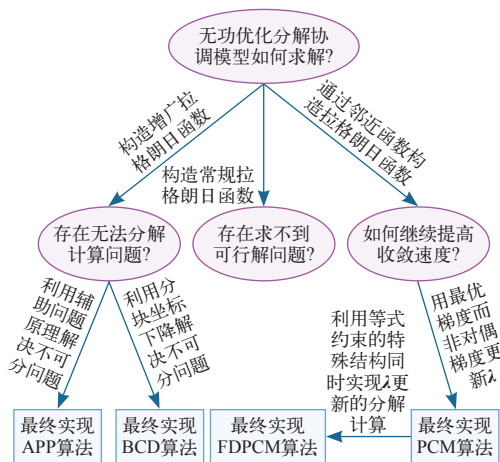


图1 各种算法的求解思路

2.2 寻优方式

由于上述算法采用的原理或算法不同, 造就在迭代计算过程中寻找最优解的方式不同, 这种差异体现在对原始目标函数的分解计算公式不同

上。其中, PCM算法和FDPCM算法的原理相同, 两者对原始目标函数的分解计算公式相同, 只是更新拉格朗日乘子的计算方式不同, 因此其寻优方式是相同的。由于无功优化分解协调模型较复杂, 为便于直观、清晰地对比几种算法的寻优方式, 以文献[12]中的带约束可分解简单优化问题为例进行分析。

$$\begin{aligned} \min_{\mathbf{x}_1 \in X_1^f, \mathbf{x}_2 \in X_2^f} f(\mathbf{x}) &= f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) \\ \text{s. t. } &\mathbf{x}_1 - \mathbf{x}_2 = 0 \end{aligned} \quad (2)$$

根据文献[10-14]可知, APP算法、BCD算法和PCM算法对式 (2) 的分解计算公式分别为

$$\begin{cases} \mathbf{x}_1^{k+1} = \arg \min_{\mathbf{x}_1 \in X_1^f} \{f_1(\mathbf{x}_1) + \lambda^{kT} \mathbf{x}_1 + c(\mathbf{x}_1^k - \mathbf{x}_2^k)^T \mathbf{x}_1 + \frac{b}{2} \|\mathbf{x}_1 - \mathbf{x}_1^k\|^2\} \\ \mathbf{x}_2^{k+1} = \arg \min_{\mathbf{x}_2 \in X_2^f} \{f_2(\mathbf{x}_2) - \lambda^{kT} \mathbf{x}_2 - c(\mathbf{x}_1^k - \mathbf{x}_2^k)^T \mathbf{x}_2 + \frac{b}{2} \|\mathbf{x}_2 - \mathbf{x}_2^k\|^2\} \end{cases} \quad (3)$$

$$\begin{cases} \mathbf{x}_1^{k+1} = \arg \min_{\mathbf{x}_1 \in X_1^f} \{f_1(\mathbf{x}_1) + \lambda^{kT} \mathbf{x}_1 + \frac{c}{2} \|\mathbf{x}_1 - \mathbf{x}_2^k\|^2\} \\ \mathbf{x}_2^{k+1} = \arg \min_{\mathbf{x}_2 \in X_2^f} \{f_2(\mathbf{x}_2) - \lambda^{kT} \mathbf{x}_2 + \frac{c}{2} \|\mathbf{x}_1^k - \mathbf{x}_2\|^2\} \end{cases} \quad (4)$$

$$\begin{cases} \mathbf{x}_1^{k+1} = \arg \min_{\mathbf{x}_1 \in X_1^f} \{f_1(\mathbf{x}_1) + \lambda^{kT} \mathbf{x}_1 + \frac{c}{2} \|\mathbf{x}_1 - \mathbf{x}_{1_0}\|^2\} \\ \mathbf{x}_2^{k+1} = \arg \min_{\mathbf{x}_2 \in X_2^f} \{f_2(\mathbf{x}_2) - \lambda^{kT} \mathbf{x}_2 + \frac{c}{2} \|\mathbf{x}_2 - \mathbf{x}_{2_0}\|^2\} \end{cases} \quad (5)$$

式中, $\|\cdot\|$ 为向量的欧式范数; $\mathbf{x}_{1_0}$ 、 $\mathbf{x}_{2_0}$ 分别为 $\mathbf{x}_1$ 、 $\mathbf{x}_2$ 的初始值。由于

$$\begin{aligned} &c(\mathbf{x}_1^k - \mathbf{x}_2^k)^T \mathbf{x}_1 + \frac{b}{2} \|\mathbf{x}_1 - \mathbf{x}_1^k\|^2 \\ &= c(\mathbf{x}_1^k)^T \mathbf{x}_1 - c(\mathbf{x}_2^k)^T \mathbf{x}_1 + \frac{b}{2} \|\mathbf{x}_1\|^2 - b(\mathbf{x}_1)^T \mathbf{x}_1^k + \frac{b}{2} \|\mathbf{x}_1^k\|^2 + \\ &\quad \frac{c}{2} \|\mathbf{x}_1\|^2 - \frac{c}{2} \|\mathbf{x}_1\|^2 + \frac{c}{2} \|\mathbf{x}_1^k\|^2 - \frac{c}{2} \|\mathbf{x}_1^k\|^2 + \frac{c}{2} \|\mathbf{x}_2\|^2 - \frac{c}{2} \|\mathbf{x}_2\|^2 \\ &= \frac{c}{2} \|\mathbf{x}_1 - \mathbf{x}_2^k\|^2 + \frac{b-c}{2} \|\mathbf{x}_1 - \mathbf{x}_1^k\|^2 + \frac{c}{2} \|\mathbf{x}_1^k\|^2 - \frac{c}{2} \|\mathbf{x}_2^k\|^2 \end{aligned} \quad (6)$$

从而式 (3) 中 $\mathbf{x}_1^{k+1}$ 的计算可以等价变换为

$$\mathbf{x}_1^{k+1} = \arg \min_{\mathbf{x}_1 \in X_1'} \left\{ f_1(\mathbf{x}_1) + \lambda^{kT} \mathbf{x}_1 + \frac{c}{2} \|\mathbf{x}_1 - \mathbf{x}_2^k\|^2 + \frac{b-c}{2} \|\mathbf{x}_1 - \mathbf{x}_1^k\|^2 + \frac{c}{2} \|\mathbf{x}_1^k\|^2 - \frac{c}{2} \|\mathbf{x}_2^k\|^2 \right\} \quad (7)$$

忽略式(7)中的常数项 $(c/2)\|\mathbf{x}_1^k\|^2$ 和 $(c/2)\|\mathbf{x}_2^k\|^2$, 最终式(3)即APP算法的分解计算公式可表示为

$$\begin{cases} \mathbf{x}_1^{k+1} = \arg \min_{\mathbf{x}_1 \in X_1'} \left\{ f_1(\mathbf{x}_1) + \lambda^{kT} \mathbf{x}_1 + \frac{c}{2} \|\mathbf{x}_1 - \mathbf{x}_2^k\|^2 + \frac{b-c}{2} \|\mathbf{x}_1 - \mathbf{x}_1^k\|^2 \right\} \\ \mathbf{x}_2^{k+1} = \arg \min_{\mathbf{x}_2 \in X_2'} \left\{ f_2(\mathbf{x}_2) - \lambda^{kT} \mathbf{x}_2 + \frac{c}{2} \|\mathbf{x}_2 - \mathbf{x}_1^k\|^2 + \frac{b-c}{2} \|\mathbf{x}_2 - \mathbf{x}_2^k\|^2 \right\} \end{cases} \quad (8)$$

根据式(4)、式(5)和式(8)可以看出, 3种算法的迭代公式都可看作是在常规拉格朗日算法的基础上增加了可分解的二次惩罚项, 其中, BCD算法在计算 $\mathbf{x}_1^{k+1}$ 、 $\mathbf{x}_2^{k+1}$ 时受到了与关联变量上次迭代值偏离程度的惩罚, PCM算法在计算 $\mathbf{x}_1^{k+1}$ 、 $\mathbf{x}_2^{k+1}$ 时受到了与自身初始值偏离程度的惩罚, APP算法在计算 $\mathbf{x}_1^{k+1}$ 、 $\mathbf{x}_2^{k+1}$ 时除了都受到与关联变量上次迭代值偏离程度的惩罚外, 还都受到了与自身上次迭代值偏离程度的惩罚。通过对比可以看出, 由于所受约束不同, 最终各种算法在每次迭代中变量更新移动的方向也不同, 如图2所示。因此, 几种算法寻优过程中变量更新移动轨迹及收

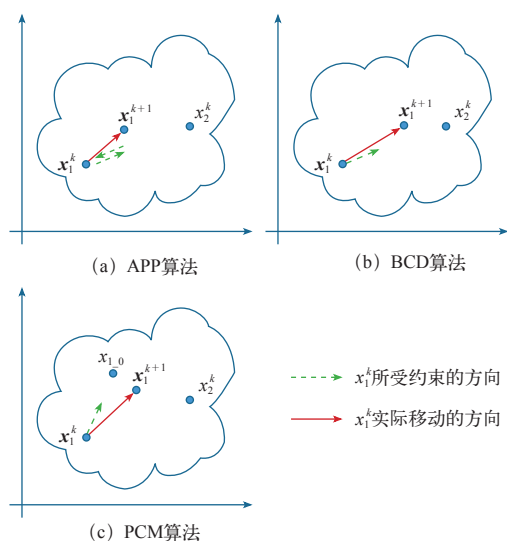
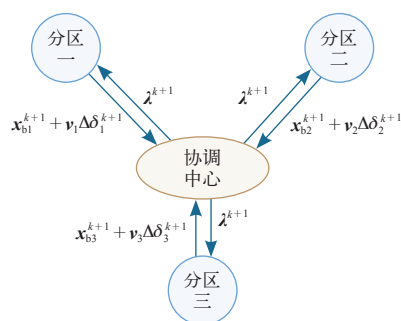


图2 $\mathbf{x}_1^k$ 更新移动方向示意图

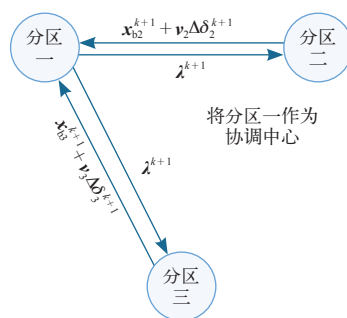
敛速度因所受约束的不同而不同。

2.3 通信协调模式

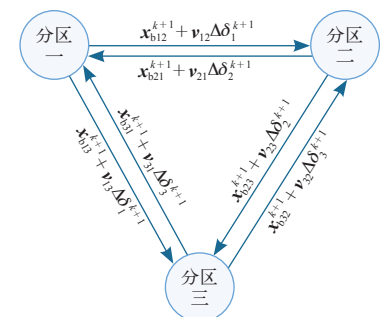
每种算法的通信协调模式与其计算过程中需要交换数据的方式有关。通过几种算法的计算步骤^[10-14]可以看出, PCM算法需要一个协调中心, 每个分区需要向协调中心发送经 Δ -变量修正后的边界节点信息, 并从协调中心接收返回的拉格朗日乘子最新值。而APP算法、BCD算法和FDPCM算法只需要在相邻分区之间交换经 Δ -变量修正后的边界节点信息, 不需要通信交换拉格朗日乘子信息。以一个三分区互联电网为例, 几种算法的通信协调模式如图3所示。



(a) PCM算法的一种模式



(b) PCM算法的另一种模式



(c) APP算法、BCD算法和FDPCM算法的模式

图3 几种算法的通信协调模式

2.4 每次迭代所需的数据通信量

几种算法在每次迭代过程中的通信量与其需要交换数据的内容有关。对于含有 n 个分区、 N 条联接线的网络，PCM算法每次迭代过程的通信量为 $(N \times 4 \times 2 + N \times 4 \times n)$ 个浮点型数字，当将某个分区作为协调中心时，通信量会因该分区不需要与自身交换数据而略少一些；APP算法、BCD算法和FDPCM算法每次迭代过程的通信量为 $N \times 4 \times 2$ 个浮点型数字。总的来说，APP算法、BCD算法和FDPCM算法不但通信协调模式相同，而且每次迭代过程中的通信量也相同，而PCM算法的通信协调模式和每次迭代过程中的通信量则都与前者不同。

3 算例分析

本文仍以IEEE 118节点系统4种不同分区情况和IEEE 300节点系统为例，对寻优方式相同的PCM算法和FDPCM算法计算中所用的系数矩阵进行对比，并对4种算法的通信协调模式及计算结果进行对比，计算中仿真环境以及所有模型和参数信息与文献[13-14]相同。

3.1 PCM算法和FDPCM算法的系数矩阵

各个算例情况下两种算法所用系数矩阵的异同情况如表1所示。限于篇幅，本文仅以IEEE 118节点系统四分区1和IEEE 300节点系统为例，给出具体的系数矩阵，即

$$\left\{ \begin{aligned} A_1 &= \begin{pmatrix} E^{(12)} & 0 \\ 0 & E^{(4)} \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, & A_2 &= \begin{pmatrix} -E^{(12)} & 0 \\ 0 & 0 \\ 0 & E^{(12)} \\ 0 & 0 \end{pmatrix} \\ A_3 &= \begin{pmatrix} 0 & 0 & 0 \\ -E^{(4)} & 0 & 0 \\ 0 & -E^{(12)} & 0 \\ 0 & 0 & E^{(16)} \end{pmatrix}, & A_4 &= \begin{pmatrix} 0 \\ 0 \\ 0 \\ -E^{(16)} \end{pmatrix} \\ B_1 &= \begin{pmatrix} E^{(12)} & 0 \\ 0 & E^{(4)} \end{pmatrix}, & B_2 &= \begin{pmatrix} -E^{(12)} & 0 \\ 0 & E^{(12)} \end{pmatrix} \\ B_3 &= \begin{pmatrix} -E^{(4)} & 0 & 0 \\ 0 & -E^{(12)} & 0 \\ 0 & 0 & E^{(16)} \end{pmatrix}, & B_4 &= -E^{(16)} \end{aligned} \right. \quad (9)$$

表1 系数矩阵的异同情况

系统情况	系数矩阵
IEEE 118节点系统二分区	相同
IEEE 118节点系统三分区	不同
IEEE 118节点系统四分区1	不同
IEEE 118节点系统四分区2	不同
IEEE 300节点系统	相同

$$\left\{ \begin{aligned} A_1 = B_1 &= \begin{pmatrix} E^{(12)} & 0 & 0 \\ 0 & E^{(28)} & 0 \\ 0 & 0 & E^{(4)} \end{pmatrix} \\ A_2 = B_2 &= -E^{(12)} \\ A_3 = B_3 &= -E^{(28)} \\ A_4 = B_4 &= -E^{(4)} \end{aligned} \right. \quad (10)$$

可以看出，两种算法所用系数矩阵的差异与分区数目无关，而与各个分区之间的关联关系有关。不管分区数目是多少，两种算法所用的系数矩阵都有可能相同。同样的分区数情况下，两种算法所用的系数矩阵可能会因各分区之间关联关系的不同而不同。将PCM的系数矩阵中整行矩阵块都为零矩阵的行去掉后就是FDPCM的系数矩阵。当PCM的系数矩阵中不包含整行矩阵块都为零矩阵的行时，两种算法的系数矩阵完全相同。

3.2 通信协调模式

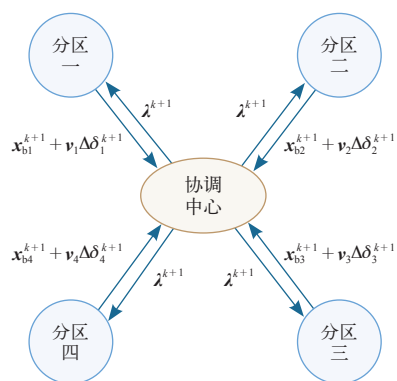
各个算例情况下的通信通道数量及通信量情况如表2所示。限于篇幅，仅以IEEE 118节点系统四分区1为例给出了具体的通信协调模式，如图4所示。

表2 通信通道数及每次迭代的数据通信量

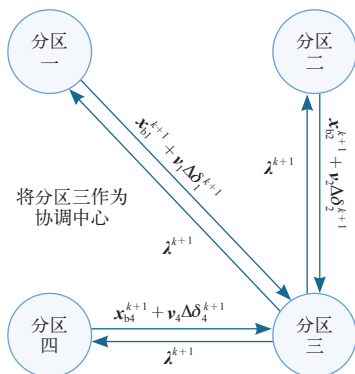
系统情况	通信通道数及通信量		
	PCM-a	PCM-b	其他3种算法
IEEE 118节点系统二分区	2, 64	1, 32	2, 32
IEEE 118节点系统三分区	3, 140	2, 88	3, 56
IEEE 118节点系统四分区1	4, 264	3, 188	4, 56
IEEE 118节点系统四分区2	4, 312	3, 236	4, 208
IEEE 300节点系统	4, 264	3, 176	3, 88

通过以上数据可以看出：

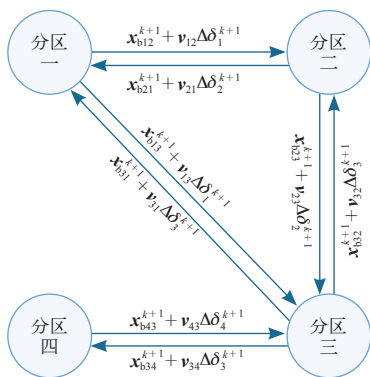
1) PCM算法在单独设立协调中心时需要建立的通信通道数等于分区数目，当将其中一个分区作为协调中心时，需要建立的通信通道数可以减



(a) PCM算法的一种模式



(b) PCM算法的另一种模式



(c) 其他三种算法的模式

图4 IEEE 118节点系统四分区1算例的通信协调模式

少一个。其他3种算法需要建立的通信通道数取决于各个分区之间的关联关系，并且理论上大于等于PCM将其中一个分区作为协调中心时需要建立的通信通道数。当然，其他3种算法是在相邻分区之间建立通信通道，而PCM算法是在分区与协调中心之间建立通信通道，因此理论上前者的距离应该小于后者。

2) PCM算法在单独设立协调中心时每次迭代

过程所需的通信数据量必定大于其他3种算法的，而当将其中一个分区作为协调中心时每次迭代过程所需的通信数据量会减少一些，具体减少的量等于与协调中心所在分区有关联的联络线数与全网所有联络线数的和的4倍，但最终的数据通信量仍然是大于等于其他3种算法的。

3.3 优化计算结果

表3给出了分别采用4种算法对各种不同分区系统进行分解协调无功优化的计算结果，可以看出：

1) PCM算法和FDPCM算法在各种不同算例情况下求得的优化网损和迭代次数都完全相同，说明后者只是在前者基础上改变了计算模式，但没有改变寻优方式和收敛特性。

2) 4种算法在各种不同系统情况下的优化网损几乎完全相同，但收敛迭代次数差别较大。其中，APP算法所需的迭代次数最多，PCM算法和FDPCM算法所需的迭代次数最少。

表3 仿真计算结果

系统 情况	优化网损(pu)				迭代次数/次			
	APP	BCD	PCM	FDPCM	APP	BCD	PCA	FDPCA
IEEE 118 二分区	1.238	1.2377	1.238	1.238	11	8	6	6
IEEE 118 三分区	1.237	1.237	1.237	1.237	18	13	9	9
IEEE 118 四分区1	1.237	1.237	1.237	1.237	29	21	14	14
IEEE 118 四分区2	1.238	1.239	1.239	1.239	32	23	15	15
IEEE 300 节点系统	3.779	3.779	3.779	3.779	25	20	14	14

4 结束语

通过特点分析和算例结果对比，明确了现有基于分解协调模型的无功优化算法的特点差异，具体如下。

1) 4种算法的求解思路不完全相同。APP算法和BCD算法分别采用辅助问题原理和分块坐标下降思想解决增广拉格朗日函数的不可分问题并对其进行求解。PCM算法和FDPCM算法通过邻近函数重新构造平滑同时可分的拉格朗日函数并对其

进行求解,并且后者在前者基础上实现了全部过程的分解计算。

2) 在寻优方式方面,除PCM算法和FDPCM算法完全相同外,几种算法有相同点也有不同点。相同点是变量在更新计算时都以二次惩罚项的方式受到了与某些值偏离程度的约束,不同点是偏离的约束值不同。

3) 在通信协调模式方面,APP算法、BCD算法和FDPCM算法的模式相同,是在相邻分区之间进行通信,而PCM算法是在所有分区和协调中心之间进行通信,并且后者每次迭代的数据通信量大于等于前者。

4) 4种算法都能求得最优解,但收敛迭代次数差别较大。其中,APP算法所需的迭代次数最多,PCM算法和FDPCM算法所需的迭代次数最少并且相同。

5) 与PCM算法相比,FDPCM算法只是在前者基础上改变了计算模式,但没有改变寻优方式和收敛特性。两种算法所用系数矩阵的差异与各个分区之间的关联关系有关。将PCM的系数矩阵中整行矩阵块都为零矩阵的行去掉后就是FDPCM的系数矩阵。

参考文献

- [1] 王建学,王锡凡,陈皓勇,等.基于协同进化法的电力系统无功优化[J].中国电机工程学报,2004,24(9):124-129.
- [2] 赵波,郭创新,张鹏翔,等.基于分布式协同粒子群优化算法的电力系统无功优化[J].中国电机工程学报,2005,25(21):1-7.
- [3] Liang C H, Chung C Y, Wong K P, et al. Parallel optimal reactive power flow based on cooperative co-evolutionary differential evolution and power system decomposition[J]. IEEE Transactions on Power Systems, 2007, 22(1): 249-257.
- [4] 赵维兴,刘明波,孙斌,等.基于诺顿等值的多区域系统无功优化分解协调算法[J].电网技术,2009,33(11):44-48,54.
- [5] 刘志文,刘明波.基于Ward等值的多区域无功优化分解协调算法[J].电力系统自动化,2010,34(14):63-69.
- [6] 刘志文,刘明波,夏文波.采用两种外部等值技术的多区域无功优化并行算法比较[J].电力系统保护与控制,2012,40(17):56-61,66.
- [7] 赵维兴,刘明波.基于近似牛顿方向的多区域无功优化解耦算法[J].中国电机工程学报,2007,27(25):18-24.
- [8] 赵维兴,刘明波,缪楠林,等.基于对角加边模型的多区域无功优化分解算法[J].电力系统自动化,2008,32(4):25-29,40.
- [9] Wei Y, Wen Lili, Li W, et al. Decomposition-coordination interior point method and its application to multi-area optimal reactive power flow[J]. Electrical Power and Energy Systems, 2011, 33(1): 55-60.
- [10] 程新功,厉吉文,曹立霞,等.基于电网分区的多目标分布式并行无功优化研究[J].中国电机工程学报,2003,23(10):110-113.
- [11] 刘宝英,杨仁刚.采用辅助问题原理的多分区并行无功优化算法[J].中国电机工程学报,2009,29(7):47-51.
- [12] 李智,杨洪耕.基于分块坐标下降思想的并行无功优化分解协调算法[J].电网技术,2013,37(1):178-182.
- [13] 李智,杨洪耕.基于邻近中心算法的无功优化分解协调计算[J].电力自动化设备,2012,32(12):33-37.
- [14] 李智,杨洪耕.一种用于分解协调无功优化的全分邻近中心算法[J].中国电机工程学报,2013,33(1):77-83.
- [15] Balho H K, Ross B. Coarse-grained distributed optimal power flow[J]. IEEE Transactions on Power Systems, 1997, 12(2): 932-939.

(收稿日期:2016-06-07) **EA**