Fully Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier Method

Linfeng Yang† , Tingting Zhang*, Guo Chen*, Zhenrong Zhang* ,**, Jiangyao Luo* and Shanshan Pan§**

Abstract – Based on the requirements and characteristics of multi-zone autonomous decision-making in modern power system, fully distributed computing methods are needed to optimize the economic dispatch (ED) problem coordination of multi-regional power system on the basis of constructing decomposition and interaction mechanism. In this paper, four fully distributed methods based on alternating direction method of multipliers (ADMM) are used for solving the ED problem in distributed manner. By duplicating variables, the 2-block classical ADMM can be directly used to solve ED problem fully distributed. The second method is employing ADMM to solve the dual problem of ED in fully distributed manner. N-block methods based on ADMM including Alternating Direction Method with Gaussian back substitution (ADM_G) and Exchange ADMM (E_ADMM) are employed also. These two methods all can solve ED problem in distributed manner. However, the former one cannot be carried out in parallel. In this paper, four fully distributed methods solve the ED problem in distributed collaborative manner. And we also discussed the difference of four algorithms from the aspects of algorithm convergence, calculation speed and parameter change. Some simulation results are reported to test the performance of these distributed algorithms in serial and parallel.

Keywords: ADMM, Economic dispatch, Fully distributed, Dual, Parallel.

1. Introduction

The economic dispatch (ED) model of power system is the basic problem of the optimal operation of the power system. It means to optimize the output of each unit to
minimize the total operating cost of the generator set system. A novel modified particle swarm optimization [13], minimize the total operating cost of the generator set under the condition of satisfying the system load demand and the running condition of the unit [1]. Most of the realtime optimal scheduling procedures adopted by the current national power companies are based on the classical ED mathematical model.

To date, a number of approaches have been proposed for ED. These methods mainly include two categories, the traditional classical optimization algorithm and the intelligent optimization algorithm. The traditional optimi zation algorithms include quadratic programming [2], linear programming [3], dynamic programming [4], Lagrangian relaxation [5, 6], and priority list [7]. The intelligent optimization algorithm mainly includes artificial neural network algorithm [8, 9], genetic algorithm [10],

§ College of Electrical Engineering, Guangxi University, China.

Received: November 10, 2017; Accepted: February 2, 2018

improved method [12-14]. A hybrid algorithm ACO– ABC–HS [12] combines the framework of Ant Colony Optimization (ACO), Artificial Bee Colony (ABC) and Harmonic Search (HS) algorithms to find the optimized solution for the problem of ED for a multi-generator which includes advantages of bacterial foraging and PSO for constrained dynamic ED problem. The modified PSO has better balance between local and global search abilities and it can avoid local minima quickly. The integrated genetic algorithm [14] is implemented in both parallel and cluster structures.

particle swarm optimization (PSO) [11], and some hybrid or with the characteristics of the power system to achieve With the development of renewable energy, the structure of power grid is more and more complex. Centralized ED is clearly not suitable for modern power grid. Most of the above algorithms only focus on how to quickly solve the problem from the perspective of computing speed. And they did not take into account the following questions: First, parallel computing should not mean a simple task of multi-machine allocation [15]. But it should be combined with the characteristics of the power system to achieve Second, in the above parallel algorithms, at least one host needs to understand the entire network of mathematical models and detailed parameters, which needs to collect all the data of the whole system [14]. For this manner, data communication is the bottleneck and the participants' private parameters exposure is the mainly concerned problem. Finally, the development of the electricity market

This is an Open-Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/ licenses/by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

[†] Corresponding Author: School of Computer Electronics and

Information, Guangxi University, China. (ylf@gxu.edu.cn)

^{*} School of Computer Electronics and Information, Guangxi University,
China.
Example: Communication and Naturals

Guangxi Key Laboratory of Multimedia Communication and Network Technology, China.

^{***} School of Electrical Engineering and Telecommunications, University of New South Wales, Australia.

Copyright ⓒ The Korean Institute of Electrical Engineers

makes the models and data within the divisions of the grid a commercial secret, and the data exchange between regions becomes impossible and unnecessary [17, 18]. Based on the above reasons, the fully distributed parallel processing technology in the power system still is a problem worth a further study. In the fully distributed method, all participators can keep their device characteristics secret, and each agent solves a subproblem with limited information communication. The distributed method ADMM is very suitable for distributed convex optimization, and in particular to large-scale problems [19]. At present, many ADMM variations have been derived. Distributed be equal to the system requirements. computing of power system needs a kind of parallel algorithm which is suited to large power grid layered distribution control structure, quickly applied to the realtime control, conform to the development direction of power market. The ADMM can meet this requirement. ADMM for formulating and solving a decentralized unit commitment problem is presented in [20] which provide a significant benefit for computational speed. In [21], an optimal power flow scheme based on ADMM is proposed, which proves the expansibility and convergence of the algorithm.

Aiming at the characteristics of multi-zone autonomous decision-making in modern power system, based on the construction of decomposition and interaction mechanism, four kinds of distributed methods based on ADMM are employed to realize distributed cooperative ED of power system. These algorithms have the following three advantages: 1) Distributed ED methods have better confidentiality. 2) Distributed ED methods are more scalable and flexible. 3) Distributed ED methods are more robust than centralized ED when loss the certain local where $P_{i,t}$ and $P_{i,t-1}$ are the power output of unit i in controllers. The major contributions are summarized as follows: 1) Four algorithms all decouple sub-problems for each unit. This means that the parameters of each unit can be kept in only one computing node. They are fully distributed to meet the needs of modern industrial privacy protection. 2) We use four fully distributed algorithms to solve ED problem. Among these methods, which are ADMM, E_ADMM, D_ADMM, and ADM_G, the first three methods can be implemented in master-slave distributed and parallel schema. And the master-node can be deployed in regional center; other computing nodes can be deployed (or installed) on each unit. Most computations for each unit can be done in each node in parallel. 3) The simulation results show that E_ADMM and D_ADMM can obtain high-quality solutions in reasonable times and D ADMM algorithm has better parallel performance, ADMM has the best convergence for finding a normal quality solution, and this method is suited to solve large scale ED problems when real-time solutions are needed.

2. Formulation for ED

The objective of the ED problem is to minimize the total

operation cost:

angyao Luo and Shanshan Pan

\nost:

\n
$$
\min F = \sum_{i=1}^{N} \sum_{t=1}^{T} \left(a_i P_{i,t}^2 + b_i P_{i,t} + c_i \right) \tag{1}
$$
\nthe total fuel cost of the units, *N* is the total units in the system, $P_{i,t}$ is the power output of period *t*, *a*, *b*, and *c*, are the cost coefficients of

and Shanshan Pan
 $\sum_{i=1}^{T} \left(a_i P_{i,t}^2 + b_i P_{i,t} + c_i \right)$ (1)

fuel cost of the units, N is the total

e system, $P_{i,t}$ is the power output of
 b_i , and c_i are the cost coefficients of *i t min* $F = \sum_{i=1}^{N} \sum_{t=1}^{T} \left(a_i P_{i,t}^2 + b_i P_{i,t} + c_i \right)$ (1)

the total fuel cost of the units, N is the total

units in the system, $P_{i,t}$ is the power output of

rively.

the cost coefficients of

eectively. *uo and Shanshan Pan*
 $\sum_{i=1}^{N} \sum_{t=1}^{T} (a_i P_{i,t}^2 + b_i P_{i,t} + c_i)$ (1)

I fuel cost of the units, *N* is the total

the system, $P_{i,t}$ is the power output of
 a_i , b_i , and c_i are the cost coefficients of where F is the total fuel cost of the units, N is the total ng Zhang, Jiangyao Luo and Shanshan Pan

operation cost:
 $min F = \sum_{i=1}^{N} \sum_{t=1}^{T} (a_i P_{i,t}^2 + b_i P_{i,t} + c_i)$ (1)

where *F* is the total fuel cost of the units, *N* is the total

number of units in the system, $P_{i,t}$ is the number of units in the system, $P_{i,t}$ is the power output of unit *i* in period *t*, a_i , b_i , and c_i are the cost coefficients of unit *i* , respectively. *min* $F = \sum_{i=1}^{N} \sum_{t=1}^{T} (a_i P_{i,t}^2 + b_i P_{i,t} + c_i)$ (1)

where *F* is the total fuel cost of the units, *N* is the total

number of units in the system, $P_{i,t}$ is the power output of

unit *i* in period *t*, a_i , b_i , $\int_{1}^{1} (a_{i}P_{i,t}^{2} + b_{i}P_{i,t} + c_{i})$ (1)

cost of the units, *N* is the total

ystem, $P_{i,t}$ is the power output of

and c_{i} are the cost coefficients of

ED model are:

traints: the sum of all units must

quirement *P*
 P
 P
 *P*_{*A*} *P*_{*A*} *+ b_tP*_{*A*} *+ c_i P*_{*A*} *i* s the total system, $P_{i,t}$ is the power output of *c_i* and *c_i* are the cost coefficients of ED model are:

ED model are:

ET model are:

ET mod where *F* is the total fuel cost of the units, *N* is the total
number of units in the system, $P_{i,t}$ is the power output of
unit *i* in period *t*, a_i , b_i , and c_i are the cost coefficients of
unit *i*, respectivel $\sum_{i=1}^{N} \sum_{t=1}^{T} \left(a_i P_{i,t}^2 + b_i P_{i,t} + c_i \right)$ (1)

all fuel cost of the units, *N* is the total

the system, *P_{i,t}* is the power output of
 a_i , *b*, , and *c*_i are the cost coefficients of
 r,
 f the ED model

The constraints of the ED model are:

1) Power balance constraints: the sum of all units must

$$
\sum_{i=1}^{N} P_{i,t} = P_{D,t} \tag{2}
$$

where $P_{D,t}$ is the system load demand in period t.

2) Unit generation limits: the power output of each unit must be limited to one upper and lower.

$$
P_i^{\min} \le P_{i,t} \le P_i^{\max} \tag{3}
$$

where P_i^{min} , P_i^{max} are the maximum power output and minimum power output of unit *i* , respectively.

3) Ramp rate limits: the power output of a unit can't change by more than a certain value over a period of time.

$$
P_{i,t} - P_{i,t-1} \leq UR_i \tag{4}
$$

$$
P_{i\ t-1} - P_{i\ t} \le D R_i \tag{5}
$$

P P_i P_i P_i P_i P_j $P_{i,t}$ = *P_{D,t}* (2)
 P_{i-1} = *P_{D,t}* (2)
 P *i* $P_{i,t} = P_{D,t}$ (2)
 p stem load demand in period *t* .
 n limits: the power output of each unit

ne upper and lower.
 $P_i^{min} \leq P_{i,t}$ *F* the ED model are:

constraints: the sum of all units must

m requirements.
 $\sum_{i=1}^{N} P_{i,t} = P_{D,t}$ (2)

system load demand in period t.

n limits: the power output of each unit

ne upper and lower.
 $P_i^{min} \leq P_{i,t} \leq$ 2) Unit generation limits: the power output of each unit
must be limited to one upper and lower.
 $P_i^{min} \le P_{i,t} \le P_i^{max}$ (3)
where P_i^{min} , P_i^{max} are the maximum power output and
minimum power output of unit *i*, respecti and *P_i* $P_i^{min} \leq P_i$, $\leq P_i^{max}$ (3)
 $P_i^{min} \leq P_{i,t} \leq P_i^{max}$ (3)
 P_i^{max} are the maximum power output and

wer output of unit *i*, respectively.

rate limits: the power output of a unit can't

ore than a certain valu period *t* and $t-1$. UR_i , DR_i are the ramp up and down limits of unit *i*, respectively.

Then, the ED problem model is:

$$
\min F = \sum_{i=1}^{N} \sum_{t=1}^{T} (a_i P_{i,t}^2 + b_i P_{i,t} + c_i)
$$
\n
$$
s.t. \begin{cases}\n\sum_{i=1}^{N} P_{i,t} = P_{\text{D},t} \\
P_i^{\min} \le P_{i,t} \le P_i^{\max} \\
P_{i,t} - P_{i,t-1} \le UR_i \\
P_{i,t-1} - P_{i,t} \le DR_i\n\end{cases} (6)
$$

3. Theoretical Basis of ADMM

ADMM can be traced back to the 1950s, and it was developed rapidly in the 1970s [19], which was first proposed by Gabay and Mercier. In recent years, due to fast processing performance and good convergence performance, the algorithm has attracted much attention in the field of large-scale data analysis and processing, such as statistical learning, speech recognition, image processing, etc.

ADMM is an important method to solve the convex optimization problem with separable structure. It is generally used to solve the convex optimization problem with equality constraints as follows:

$$
\min f(x) + g(z)
$$

s.t. $Ax + Bz = c$ (7)

Where $x \in R^n$, $z \in R^m$ are variables, $A \in R^{p \times n}$, $B \in R^{p \times m}$, $P_1^{k+1} := \tilde{P}_1^k$. The $f(\cdot)$ and $g(\cdot)$ are convex functions.

To solve the problem (7) by ADMM, we form the augmented Lagrangian

$$
L_{\rho}(x, z, y) = f(x) + g(z) + y^{T} (Ax + Bz - c)
$$

+ $(\rho / 2)||Ax + Bz - c||^{2}$ (8) 4.1 ADMM for fu

where *y* is the Lagrange multiplier, $\rho > 0$ is called the penalty parameter, " $\|\cdot\|$ " denotes " $\|\cdot\|_2$ ".

Then the unscaled form ADMM consists of the iterations as follows:

$$
x^{k+1} := \underset{x}{\operatorname{argmin}} L_p\left(x, z^k, y^k\right) \tag{9}
$$
Then the objective

$$
z^{k+1} := \underset{z}{\operatorname{argmin}} L_{\rho}\left(x^{k+1}, z, y^k\right) \tag{10}
$$

$$
y^{k+1} \coloneqq y^k + \rho \left(Ax^{k+1} + Bz^{k+1} - c \right) \tag{11}
$$
ADMM form as

Let $u = (1/\rho)y$ which called as the scaled dual variable, then $(9)-(11)$ can be equivalently expressed as a scaled form: $+(p/2)||Ax + Bz - c||^2$

here *y* is the Lagrange multiplier, $p > 0$ is called the

nearby parameter, "|-|||" denotes "||-|||,"

Then the unscaled form ADMM consists of the iterations

stellows:
 $I_x(x) = \begin{cases} 0 & \text{if } x \in \mathbb{Z} \\ 1$ here y is the Lagrange multiplier, $\rho > 0$ is called the

national γ cannots with $\left|\frac{1}{2}\right|$, γ .

Then the unscaled form ADMM consists of the iterations

Then the unscaled form ADMM consists of the iterations
 x

$$
x^{k+1} := \underset{x}{\text{argmin}} \bigg(f(x) + (\rho/2) \|Ax + Bz^{k} - c + u^{k}\|^2 \bigg) \tag{12}
$$

$$
z^{k+1} := \underset{z}{\text{argmin}} \bigg(g(z) + (\rho/2) \|Ax^{k+1} + Bz - c + u^k\|^2 \bigg) \tag{13}
$$

$$
u^{k+1} := u^k + Ax^{k+1} + Bz^{k+1} - c \tag{14}
$$

Then the unscaled form ADMM consists of the iterations

1 $I_x(x) = \begin{cases} 0 & \text{if } x \in \mathbb{X} \\ +\infty & \text{if } x \notin \mathbb{X} \end{cases}$
 $x^{k+1} := argmin L_\rho\left(x, x^k, y^k\right)$
 $z^{k+1} := argmin L_\rho\left(x^{k+1}, z, y^k\right)$
 $y^{k+1} := y^k + \rho\left(dx^{k+1} + Bz^{k+1} - c\right)$

11) two Under two assumptions, *L_x*-8(x, z^k, y^k) (9)

Then, the objective function can be decomposed in the special state of the s $x^{k+1} := argmin L_{\rho}\left(x, z^k, y^k\right)$
 $z^{k+1} := argmin L_{\rho}\left(x^{k+1}, z, y^k\right)$
 $z^{k+1} := y^k + \rho\left(dx^{k+1} + Bz^{k+1} - c\right)$ (10) Then, the objective function, i.e., the model (6) $y^{k+1} := y^k + \rho\left(dx^{k+1} + Bz^{k+1} - c\right)$ (11) ADMM form as

Let $u = (1$ the objective function of the iterates approaches the $z^{k+1} := argmin L_{\rho}\left(x^{k+1}, z, y^k\right)$
 $\Rightarrow x^{k+1} := y^k + \rho\left(Ax^{k+1} + Bz^{k+1} - c\right)$

Let $u = (1/\rho)y$ which called as the scaled dual

Let $u = (1/\rho)y$ which called as the scaled dual
 $\Rightarrow x^k = \frac{1}{2}$ and $\left(f(x) + (p/2)\right) || Ax + Bz^k - c + u^k ||^2$
 $*$, where p^* is the optimal value of (7).

4. Methodology for 4 Distributed ED Methods based on ADMM

For the sake of convenience, throughout the paper the following notations are used for the ED problem descripted $x^{k+1} = argmin\left\{f(x) + (\rho/2)\right\} |Ax + Bz^2 - c + n^k|^2\right\}$ (12)
 $x^{k+1} = argmin\left\{f(x) + f_A(x^{k+1} + Bz + 1 - c + 1)^k\right\}$ (13)
 $u^{k+1} = u^k + Ax^{k+1} + Bz^{k+1} - c$

User two assumptions, $I_{y=0}(x, z, y)$ has a saddle

Dende row assumptions, $I_{y=0}(x, z, y$ 1^{J} (1) memme) $D \perp D$ N $(0,1)$ **n** $\begin{bmatrix} n & n \end{bmatrix}$ *i* $g(x) + (p/2) \left\| A x^{k+1} + B z - c + u^k \right\|^2$ (13)
 $\frac{1}{2}$ *i* $A x^{k+1} + B z^{k+1} - c$
 $\frac{1}{2}$ *i* $A x^{k+1} + B z^{k+1} - c$

(14)
 $\frac{1}{2}$ $\frac{1}{2}$ $\left\{ \int_{-2}^{2} (z) + (p_1/2) \right\| P^{k+1} - z + u^k \right\}$

(14)
 $\frac{1}{2}$ **i** $\left\{ \int_{-2$

1780 │ J Electr Eng Technol.2018; 13(5): 1778-1790

It Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier Method

olve the convex optimization problem

The constraints (3)(4)(5) can be fully decoupled

to the unit, and then define the set *s to the convex optimization problem* $X_t = \{P | (3)(4)(5) \}$ and then define the set P_3 , *i* = (*P*) (3)(4)(5) and the fully decoupled to the unit, and then define the set P_3 , *i* = (*P*) (3)(4)(5)); Constraint (2) can The constraints $(3)(4)(5)$ can be fully decoupled according sed on Alternating Direction Multiplier Method
The constraints (3)(4)(5) can be fully decoupled according
to the unit, and then define the set ρ_3 , $i = 1,..., N$;
 $\chi_1 = {P | (3)(4)(5)}$; Constraint (2) can be fully decoupled
ac *net also alternating Direction Multiplier Method*

1*x* the constraints (3)(4)(5) can be fully decoupled according

2*x*₁ = { P |(3)(4)(5)}; Constraint (2) can be fully decoupled

1*x*₁ = { P |(3)(4)(5)}; Constraint according to the time period, then define the set

Fully Distributed Economic Dispatching Methods Based on Alternating Direction M
generally used to solve the convex optimization problem
The constraints (3)(4)(5) c;
with equality constraints as follows:
 $\begin{aligned}\n &\text{in } f(x) + g$ *Fully Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier Method

generally used to solve the convex optimization problem
* $x_1 = {P \mid (3)(4)(5)}$ *; can be fully decoupled according
 x_1 = {P \mid (3)(4 Fully Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier*

d to solve the convex optimization problem
 $\begin{array}{ll}\n & x_1 = {P | (3)(4)(5)}$ can be ful

constraints as follows:

to the unit, and then Fully Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier Method

sed to solve the convex optimization problem

to the unit, and then define the set p_5

min $f(x) + g(z)$

min $f(x) + g(z)$

min *Fully Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier Method

IJy used to solve the convex optimization problem

The constraints (3)(4)(5) can be fully decoupled according

quality cons ibuted Economic Dispatching Methods Based on Alternating Direction Multiplier Method
 <i>c* convex optimization problem
 i c convex in the constraints (3)(4)(5) can be fully decoupled according
 *x*₁ = {*P*|(3)(4)(Fully Distributed Economic Disputching Methods Based on Alternating Direction Multiplier Method

solve the convex optimization problem

the unit, and then define the set P_3 , $i = 1, ..., N$;

the unit, and then define the se stributed Economic Dispatching Methods Based on Alternating Direction Multiplier Method

the convex optimization problem The constraints (3)(4)(5) can be fully decoupled accord

is a follows:
 $f(x)+g(z)$
 $f(x)+g(z)$
 $f(x)+g(z)$ min $f(x)+g(z)$
 $x_1 = {P | (3) (4) (5)}$; Constraint (2) can be fully c
 $s.t. Ax + Bz = c$
 $f(7)$ $x_2 = {P | (2)}$; the time period, then define
 R^n , $z \in R^m$ are variables, $A \in R^{p \times n}$, we note that, with the above notations, all
 $P_$ min $f(x) + g(z)$
 *x*₁ = {*P* |(3)(4)(5)}; Constraint (2) can be fully de
 x, *x*, + *B* = c
 x $z = (P | (3)(4)(5))$; Constraint (2) can be fully de
 x, $x + Bz = c$

(7) $x^2 = (P | (2))$.

We note that, with the above notations, *st.* $Ax + Bz = c$ (7) $\chi_2 = {P | (2)}$.
 R^a , $\chi_3 = R^m$ are variables, $A \in R^{p \times n}$, we note that, with the above notations, all

parameters of each unit *i* are included in $f_i(P_i^{k+1}) = \overline{P_i}^{k+1}$. The $f(\cdot)$ and $g(\cdot)$ a *z* $f(x + Bz = c$
 z $f(\frac{1}{2})$ 1 R_1^{R+1} , $R_2^* = R_1^{R+m}$ are variables, $A \in R^{P \times n}$, parameters of each unit *i* are included in $f_i(P_i)$ and $f_i(P_i)$ and $g(\cdot)$ are convex $\lambda_i \in R^{P \times n}$. In order to solve the ED problem in distributed problem in d *x* ed on Alternating Direction Multiplier Method

The constraints (3)(4)(5) can be fully decoupled according

o the unit, and then define the set ρ_3 , $i = 1,..., N$;
 $\chi_1 = \{P | (3)(4)(5)\}$; Constraint (2) can be fully decou sed on Alternating Direction Multiplier Method

The constraints (3)(4)(5) can be fully decoupled according

to the unit, and then define the set ρ_3 , $i = 1,..., N$;
 $\chi_1 = \{P | (3)(4)(5) \}$; Constraint (2) can be fully decoup $\chi_{1,i}$. In order to solve the ED problem in distributed manner, our paper aims to decouple the ED to be each sub problem corresponding to each unit *i* , and each sub sed on Alternating Direction Multiplier Method
The constraints (3)(4)(5) can be fully decoupled according
to the unit, and then define the set ρ_3 , $i = 1, ..., N$;
 $\chi_1 = {P | (3)(4)(5)}$; Constraint (2) can be fully decoupled
ac each unit can keeping its information secret. *i*ne the set ρ_3 , $i = 1,..., N$;
 int (2) can be fully decoupled according
 int (2) can be fully decoupled

beriod, then define the set

above notations, all private

are included in $f_i(P_i)$ and
 e ED problem in dis *If* $S(3)(4)(3)$ can be finy decoupled according
and then define the set p_3 , $i = 1,..., N$;
 $f(5)$; Constraint (2) can be fully decoupled
the time period, then define the set
 $\frac{1}{2}$.
and, with the above notations, all pr a define the set ρ_3 , $i = 1,..., N$;
onstraint (2) can be fully decoupled
me period, then define the set
if i are included in $f_i(P_i)$ and
we the ED problem in distributed
s to decouple the ED to be each sub-
g to each u solve the ED problem in distributed
alimis to decouple the ED to be each sub-
diding to each unit *i*, and each sub-
eeds the data of $f_i(P_i)$ and $\chi_{1,i}$, i.e.,
ing its information secret.
Illy distributed ED
sic ADMM, w

2 **4.1 ADMM for fully distributed ED**

To use the classic ADMM, we introduce the auxiliary variable *z* and the indicator function of set χ , defined as,

$$
I_{\chi}(x) = \begin{cases} 0 & \text{if } x \in \chi \\ +\infty & \text{if } x \notin \chi \end{cases}
$$
 (15)

 (x^{k+1}, z, y^k) (10) two blocks with no overlapping variables by introducing Then, the objective function can be decomposed into indicator function, i.e., the model (6) can be rewritten in ADMM form as $I_{\chi}(x) = \begin{cases} 0 & \text{if } x \in \chi \\ +\infty & \text{if } x \notin \chi \end{cases}$ (15)
objective function can be decomposed into
ith no overlapping variables by introducing
tion, i.e., the model (6) can be rewritten in
as
 $\min f(P) + I_{\chi_1}(P) + I_{\chi_2}(z)$
s.t *nding* to each unit *i*, and each sub-

rededs the data of $f_i(P_i)$ and $\chi_{1,i}$, i.e.,
 i.e., *i.e.*, *i.e. PHY* and *Z*_{1*i*}, *i.e.*,
 PHY distributed ED

sic ADMM, we introduce the auxiliary
 χ) = $\begin{cases} 0 & \text$

$$
\min f(P) + I_{\chi_1}(P) + I_{\chi_2}(z) \ns.t. P - z = 0
$$
\n(16)

 (16) are the following: The iterations of the scaled form of ADMM for model

$$
L_p(x,z,y) = f(x)+g(z)+y + (ax+sz-c)
$$
\n
$$
+ (\rho/2)||x + Bz-c||^2
$$
\n
$$
= \int_{z^{k+1}}^{\infty} \frac{1}{z} \int_{z=0}^{\infty} \frac{1}{z
$$

$$
u^{k+1} := u^k + \left(P^{k+1} - z^{k+1} \right) \tag{19}
$$

P-update (17) can be completely decoupled according to each unit *i* , which is equivalent to solving the following sub-problems:

$$
\min f_i(P_i) + (\rho_1 / 2) \|P_i - z_i^k + u_i^k\|^2
$$

s.t. $P_i \in \chi_{1,i}$ (20)

 $+(p/2)\left||Ax^{k+1} + Bz - c + u^k\right||^2$ (13)
 $e^{k+1} = \operatorname{argmin}\left(f(P) + I_n(P) + (\rho_1/2)\left||P - z^k + u^k\right||^2\right)$ (17)
 $e^{k+1} = e$ (14)
 $\int e^{k+1} = e^{k+1} \cdot e^{k+1} = e^{k+1} \cdot e^{k+1}$ (18)
 $\int e^{k+1} = e^{k+1} \cdot e^{k+1} = e^{k+1} \cdot e^{k+1}$ (18)
 $\int e^{k+1} = e^{k+1$ It can be seen that the sub-problems (20) can be performed in parallel. Each sub-problem only requires the relevant parameters of the unit *i* . Each sub-problem is small in size and contains only *T* variables, i.e., *Pⁱ* .

The *z*-update step (18) is equivalent to solving the following simple quadratic programming problem:

$$
\min \frac{\rho_1}{2} \|z - P^{k+1} - u^k\|^2
$$
Then, we obtain t
s.t. $z \in \chi_2$ (21)

We are now in a position to give the classical ADMM for solving the problem of ED in full details.

Algorithm 1 ADMM for Distributed ED Initialization: $z^0 = 0$, $u^0 = 0$, $M > 0$, $\rho_1 > 0$, $\varepsilon_1^{\text{pri}} > 0$, $\varepsilon_1^{\text{dual}} > 0$. *Linfeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shan

min* $\frac{\rho_1}{2} ||z - P^{k+1} - u^k||^2$ *Then, we obtain the dual problem with* $s.t.z \in \chi_2$ *(21) max <i>h*(
 st.z $\in \chi_2$ (21) max *h*(
 stributation P-update:
for each unit $i = 1,..., N$: (in parallel) *Linfeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shanshan Pan*
 $\min \frac{\rho_1}{2} \|z - p^{k+1} - u^k\|^2$

Then, we obtain the dual problem of (22) as
 $s.t. z \in \chi_2$

(21) $\max h(y)$

now in a position to give th get P_i^{k+1} by solving problem (20). **end** *z* -update: get z^{k+1} by solving (21). *u* -update: (19) $r_1^k = P^k - z^k$, $\min \frac{\rho_1}{2} \|z - P^{k+1} - u^k\|^2$
 Example 18. $f(x) = \frac{\rho_1}{2} \int_0^x z - P^{k+1} - u^k \Big|_0^x$

Then, we obtain the dual problem
 snr and snr $f(y)$
 snr and snr snr snr snr snr snr $\text{$ $s_1^k = \rho_1 (z^k - z^{k+1}),$ **orithm 1** ADMM for Distributed ED

zation: $z^0 = 0$, $u^0 = 0$, $M > 0$, $\rho_1 > 0$,
 0 , $\varepsilon_1^{\text{dual}} > 0$.
 $= 0,..., M$

-update:
 for each unit $i = 1,..., N$: (in parallel)

get P_i^{k+1} by solving problem (20).
 end
 \begin $\lim_{t \to 0} \frac{|H|}{2} = P^{k+1} - u^k$
 $x \cdot z \in \chi_2$ (21) max $h(y)$

are now in a position to give the classical ADMM
 $x \cdot h(z)$

are mow in a position to give the classical ADMM
 $x \cdot h(z)$
 $x \cdot h(z)$
 $x \cdot h(z)$
 $x \cdot h(z)$
 $x \cdot h(z)$
 min $\frac{p_1}{2} \|z - P^{k+1} - u^k\|$

s.*t.z* $\in \chi_2$ (21) max $h(y)$

now in a position to give the classical ADMM This problem can be rewritted

the problem of ED in full details.
 ifthm 1 ADMM for Distributed ED

tion: $z^$ **if** $r_1^k \leq \varepsilon_1^{\text{pri}}$ and $s_1^k \leq \varepsilon_1^{\text{dual}}$ **break**. $x_i^2 = 0$, $M > 0$, $\rho_1 > 0$,
 $x_i^2 = 0$, $M > 0$, $\rho_1 > 0$,
 $x_i^2 = 1,..., N$: (in parallel)
 y solving problem (20).
 $y^{k+1} = argmin_{y} \left\{ \left\langle \sum_{i=1}^{N} y^{k+1} \right\rangle : argmin_{z_i} \left\{ \left\langle \sum_{i=1}^{N} z_i^{k+1} \right\rangle : argmin_{z_i} \left\{ -h_i(z_i) \right\} \right\} \$ **end** For solving the problem of ED in full details.

For solving the problem of ED in full details.
 Algorithm 1 ADMM for Distributed ED

Initialization: $z^0 = 0$, $u^0 = 0$, $M > 0$, $P_1 > 0$,
 $\varepsilon_1^{pi} > 0$, $\varepsilon_1^{dual} > 0$.

The constraint (2) in model (6) can be fully decoupled according to the time period. But in order to implement totally distributed ED solving, we must decoupled all constraints to be subproblems according to each unit.

We rewrite the model (6) as

if
$$
r_1^k \leq \varepsilon_1^{pri}
$$
 and $s_1^k \leq \varepsilon_1^{dual}$ break.
\nend
\nreturn $f(P)$ and P.
\n**4.2 ADMM solve the ED dual problem**
\nThe constraint (2) in model (6) can be fully decoupled
\naccording to the time period. But in order to implement
\ntotally distributed ED solving, we must decoupled all
\nWe rewrite the model (6) as
\n
$$
\text{where there is a nontrivial vector, } r_1 = \frac{1}{N} \sum_{i=1}^{N} z_i^k - \frac{1}{N\rho_2} \sum_{i=1}^{N} p_i^k
$$
\n
$$
\text{by substituting the expression for } h_i(z_i) \text{ into } z_i - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by the two subproblems according to each unit.}
$$
\n
$$
\text{by the two subproblems according to each unit.}
$$
\n
$$
\text{by the two subproblems according to each unit.}
$$
\n
$$
\text{by substituting the expression for } h_i(z_i) \text{ into } z_i - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by substituting the expression for } h_i(z_i) \text{ into } z_i - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by substituting the expression for } h_i(z_i) \text{ into } z_i - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by substituting the expression for } h_i(z_i) \text{ into } z_i - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by substituting the expression for } h_i(z_i) \text{ into } z_i - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by the above separable convex programming}
$$
\n
$$
\text{by } \left\{ -f_i(P_i) - \left\langle z_i, P_i - \frac{1}{N} P_D \right\rangle \right\} - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by } \left\{ -f_i(P_i) - \left\langle z_i, P_i - \frac{1}{N} P_D \right\rangle \right\} - \frac{1}{N} \varepsilon_i
$$
\n
$$
\text{by } \v
$$

For solving the above separable convex programming problem (22), we use a decomposition algorithm presented in [22]. Here, we note that, according to the practical significance of problem (22), Slater condition holds for the problem (22) . So, strong duality holds for (22) .

Let
$$
h_i(y) = \inf_{P_i \in \chi_{i,j}} \left\{ f_i(P_i) + \left\langle y, P_i - \frac{1}{N} P_D \right\rangle \right\}
$$
, where y

is Lagrangian multiplier vector associated with equality Lagrangian dual function of (22) is

$$
h(y) = \sum_{i=1}^{N} h_i(y)
$$
 (23)

Then, we obtain the dual problem of (22) as

$$
\max h(y) \tag{24}
$$

*infeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shanshan Pan
* $\frac{\rho_1}{2} \|z - P^{k+1} - u^k\|^2$ *

<i>s.t.z* = χ_2 (21) max *h*(*y*)

osition to give the classical ADMM This problem can be rewritten to b *nd Shanshan Pan*

dual problem of (22) as

max $h(y)$ (24)

oe rewritten to be a minimization

riables z_i and a common global This problem can be rewritten to be a minimization problem with local variables z_i and a common global variable *y* :

log Zhang, Jiangyao Luo and Shanshan Pan

\nThen, we obtain the dual problem of (22) as

\nmax
$$
h(y)
$$
 (24)

\nThis problem can be rewritten to be a minimization

\nproblem with local variables z_i and a common global

\nvariable y :

\n
$$
\min\left\{-\sum_{i=1}^{N} h_i(z_i)\right\}
$$

\n $s.t. y - z_i = 0$, $i = 1, \dots, N$.

\n(25)

\nPartitioning the multiplier vector $p = (p_1, \dots, p_N)$ and giving $p_2 > 0$, we may write the ADMM for (25) as follows:

\n
$$
y^{k+1} := \underset{y}{argmin} \left\{\langle \sum_{i=1}^{N} p_i^k, y \rangle + (\rho_2 / 2) \sum_{i=1}^{N} \left\|y - z_i^k\right\|^2\right\}
$$
 (26)

\n $z_i^{k+1} := \underset{z_i}{argmin} \left\{-h_i(z_i) - \left\langle p_i^k, z_i\right\rangle + (\rho_2 / 2) \left\|y^{k+1} - z_i\right\|^2\right\}$ (27)

\n $p_i^{k+1} := p_i^k + p_2\left(y^{k+1} - z_i^{k+1}\right)$ (28)

\nThe y -update problem (26) is minimizing a convex

\nquadratic function of y , we can get the minimize y from the

\noptimality condition, i.e.,

\n
$$
y = \sum_{i=1}^{N} \left\{\sum_{i=1}^{N} p_i^k, y_i - 1\right\} \sum_{i=1}^{N} \left\{\sum_{i=1}^{N} p_i^k, y_i - 1\right\} \sum_{i=1}^{N} p_i^k
$$

giving $\rho_2 > 0$, we may write the ADMM for (25) as follows:

$$
\min \left\{ -\sum_{i=1}^{n} h_i(z_i) \right\}
$$
\ns.t. $y - z_i = 0$, $i = 1, \dots, N$. (25)
\nPartitioning the multiplier vector $p = (p_1, \dots, p_N)$ and
\niving $p_2 > 0$, we may write the ADMM for (25) as
\nallows:
\n
$$
x^{k+1} := \underset{y}{\operatorname{argmin}} \left\{ \left\langle \sum_{i=1}^{N} p_i^k, y \right\rangle + \left(p_2 / 2 \right) \sum_{i=1}^{N} \left\| y - z_i^k \right\|^2 \right\} \quad (26)
$$
\n
$$
x^{k+1} := \underset{z_i}{\operatorname{argmin}} \left\{ -h_i(z_i) - \left\langle p_i^k, z_i \right\rangle + \left(p_2 / 2 \right) \left\| y^{k+1} - z_i \right\|^2 \right\} \quad (27)
$$
\n
$$
y^{k+1} := p_i^k + p_2 \left(y^{k+1} - z_i^{k+1} \right) \quad (28)
$$
\nThe *y*-update problem (26) is minimizing a convex
\nquadratic function of *y*, we can get the minimize *y* from the
\nptimality condition, i.e.,
\n
$$
y^{k+1} = \frac{1}{N} \sum_{i=1}^{N} z_i^k - \frac{1}{N \rho_2} \sum_{i=1}^{N} p_i^k \quad (29)
$$
\nBy substituting the expression for $h_i(z_i)$ into z_i -
\nupdate problem (27), we obtain that
\n
$$
\inf_{z_i} \left\{ -\inf_{p \in \mathcal{P}_i} \left\{ f_i(P_i) + \left\langle z_i, P_i - \frac{1}{N} P_D \right\rangle \right\} -
$$

$$
z_i^{k+1} := \underset{z_i}{\text{argmin}} \left\{ -h_i(z_i) - \left\langle p_i^k, z_i \right\rangle + (\rho_2 / 2) \| y^{k+1} - z_i \|^2 \right\} \tag{27}
$$

$$
p_i^{k+1} := p_i^k + \rho_2 \left(y^{k+1} - z_i^{k+1} \right) \tag{28}
$$

The *y*-update problem (26) is minimizing a convex quadratic function of *y*, we can get the minimize *y* from the optimality condition, i.e., $y > +(\rho_2 / 2) \sum_{i=1}^{\infty} ||y - z_i^k||^2$ (26)
 $p_i^k, z_i \rangle + (\rho_2 / 2) ||y^{k+1} - z_i||^2$ (27)
 (28)

1 (26) is minimizing a convex

e can get the minimize y from the
 $\sum_{i=1}^{r} z_i^k - \frac{1}{N\rho_2} \sum_{i=1}^{N} p_i^k$ (29)

xpression for h $\langle y \rangle + (\rho_2 / 2) \sum_{i=1}^n ||y - z_i^k||^2$ (26)
 $\langle p_i^k, z_i \rangle + (\rho_2 / 2) ||y^{k+1} - z_i||^2$ (27)
 $\lim_{i} \frac{k+1}{i}$ (28)
 $\lim_{i \to i} (26)$ is minimizing a convex

we can get the minimize y from the
 $\sum_{i=1}^N z_i^k - \frac{1}{N \rho_2} \sum_{i=1}^N p_i^k$

$$
y^{k+1} = \frac{1}{N} \sum_{i=1}^{N} z_i^k - \frac{1}{N\rho_2} \sum_{i=1}^{N} p_i^k
$$
 (29)

Longing Using Tanging Zkong Guo Chea, Zkornong Zkang, Jangou a t so not Stanokar Pon
\nmin
$$
\frac{n}{2} \left[z - p^{k+1} - u^k \right]^2
$$

\nThen, we obtain the dual problem of (22) as
\n $\frac{n}{2} \left[z - p^{k+1} - u^k \right]^2$
\nthen, we obtain the dual problem of (22) as
\n $\frac{n}{2} \left[z - p^{k+1} - u^k \right]^2$
\nHence now in a position to give the classical ADMM
\n $\frac{1}{2} \left[0, \frac{v^2}{1 - v^2} - 0,$

For any fixed P_i , the minimum on the right-hand side of (30) is uniquely attained by

$$
= \sum_{i=1}^{N} h_i(y) \qquad (23) \qquad z_i = y^{k+1} + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \qquad (31)
$$

http://www.jeet.or.kr **│ 1781**

We may thus substitute (31) into the function on the right-hand side of (30) to eliminate the variables z_i . As a result, we obtain

Fully Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier Method
\nWe may thus substitute (31) into the function on the
\nresult, we obtain
\nresult, we obtain
\n
$$
P_i^* = \arg \max_{P_i \in \chi_i} \{-f_i(P_i) - \langle y^{k+1} + \frac{1}{p_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \}
$$

\n $P_i^* + P_i = \frac{1}{N} P_D \rangle + (p_2 / 2) \left| \frac{1}{p_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \right|^2$
\n $= \arg \min_{P_i \in \chi_i} \{f_i(P_i) + \frac{p_2}{2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \right|^2$
\n $= \arg \min_{P_i \in \chi_i} \{f_i(P_i) + \frac{p_2}{2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \}$
\n $= \arg \min_{P_i \in \chi_i} \{f_i(P_i) + \frac{p_2}{2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \}$
\n $= \arg \min_{P_i \in \chi_i} \{f_i(P_i) + \frac{p_2}{2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \}$
\n $= \arg \min_{P_i \in \chi_i} \{f_i(P_i) + \frac{p_2}{2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \}$
\n $= \arg \min_{P_i \in \chi_i} \{f_i(P_i) + \frac{p_2}{2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \}$
\n $= \frac{1}{N} \sum_{j \neq i} \{p_i^k - p_j \} \left\{ \sum_{j=1}^N P_j^k - P_D \right\}$
\n $= \frac{1}{N} \sum_{j=1}^N \{p_i^k - p_j \} \left\{ \sum_{j=1}^N P_j^k - P_D \right\}$
\n $= \frac{1}{N} \sum_{j=1}^N \{p_i^k - p_j \} \left\{ \sum_{j=1}^N P_j^k - P_D \right\}$
\n

Therefore, if a solution P_i^* of problem (32) is found, we can determine z_i by (31).

Note that, by the separability of problem (25), the

Initialization: $z_i^0 = 0$, $p_i^0 = 0$, $M > 0$, $p_i > 0$, $\varepsilon_2^{\text{feasible}} > 0.$ *x* argmin $h_i^k + P_i - \frac{1}{N} P_D > + (p_2 / 2) \left| \frac{1}{p_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \right|^2$
 k argmin $h_i = x_i$, $\left\{ f_i(P_i) + \frac{p_2}{2} \left(y^{k+1} + \frac{1}{p_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \right)^2 \right\}$
 *k*⁴⁺¹ = $x^k - p_3 \left(\sum_{j=1}^k |P_i|^2 + \frac{1}{N} P_D \$ (29) $\min_{P_i \in \chi_i} \left\{ f_i(P_i) + \frac{P_2}{2} \left(y^{k+1} + \frac{1}{p_2} \left(p_i^k + P_i - \frac{1}{N} P_0 \right) \right)^2 \right\}$
 $\text{However, if a solution } P_j^* \text{ of problem (32) is found, not necessarily convergent if no further, if the one of the number z_i by (31).
\ndetermine z_i by χ_i by the separability of problem (25), the probability of probability z_i is given by (26) of variables $z = (z_1, \dots, z_N)$ and (28) of variables $z =$$ (32) and (28) of variables $z = (z_1, ..., z_N)$ and substitution proposed in [2]
 $\vec{w}^k := (\tilde{P}_2^k, ..., \tilde{P}_N^k, \tilde{\lambda}^k) = ($
 $\vec{w}^2 = 0$, $p_i^0 = 0$, $M > 0$, $\rho_2 > 0$,
 $\vec{w}^k := (\tilde{P}_2^k, ..., \tilde{P}_N^k, \tilde{\lambda}^k) = ($
 $\vec{w}^k := (\tilde{P}_2^k$ $\rho_2\begin{pmatrix}r_l & l & N & D\end{pmatrix}$ $\qquad \qquad \begin{bmatrix} \rho_2 D_2^T D_2 & 0\end{bmatrix}$ *k_k* $\begin{bmatrix} x_{k+1} & x_{k+1} \end{bmatrix}$ (32) and convergent if no further conditions
 k in the scheme (34)-(3
 k in the scheme (34)-(3
 k in the scheme (34)-(3
 k in the scheme (34) and convergent if no further conditi *i* α , μ_1 *i* μ_2 \cdots *i* α) and the scheme (34)-(35)
 i α , if a solution P_i^* of problem (32) is found, problem (33) and proced on the model (33), even though its efficiency lemmine z_i by (31).
 + $\begin{vmatrix}\n\frac{1}{2}x + y + z & \frac{1}{2}y + z & \frac{1}{2}z \\
\frac{1}{2}y + z & \frac{1}{2}z & \frac{1}{2}z & \frac{1}{2}z \\
\frac{1}{2}y + z & \frac{1}{2}z & \frac{1}{2}z + z & \frac{1}{2}z \\
\frac{1}{2}y + z & \frac{1}{2}z & \frac{1}{2}z + z & \frac{1}{2}z\n\end{vmatrix}\n\end{vmatrix}$

However, [23] points out that the scheme (3 After each round of ADM

12 D_ADMM for Distributed ED
 $\vec{w}^k := (\tilde{P}_2^k, ..., \tilde{P}_N^k; \tilde{\lambda}^k) = (P_2^k, ..., P_N^k; \tilde{\lambda}^k) = (P_2^k, ..., P_N^$ $p_i^{k+1} \coloneqq p_i^k + \rho_2(y^{k+1} - z_i^{k+1}),$; if a solution P_j^* of problem (32) is found, the notation the model (33), even though its efficien

mime *z_i* by (31),

the separability of problem (25), the

substitution proposed in [26] to guarantee the Gaussia **i** e, if a solution P_i^* of problem (32) is found, the model (33), even the philip is efficience the priorid equilibulity of problem (25), the parametic point is efficience that P_i by (31), the separability of proble refore, if a solution P_i^* of problem (32) is found, because the model (33), even though its electromine z_i by (31),

the separation by the separation proposed on the model (33), even though its electromine z_i by (3 **c** that, by the separability of problem (25), the $\{24, 25\}$. Here, we introduce the set of ADMM of problem (25), the substitution roposed in [26] to get $z = (z_1, \dots, z_N)$ and substitution roposed in [26] to $\lambda_1, \dots, \lambda_N$ Noting and (28) of variables $z = (z_1, ..., z_N)$ and substitution proposed in [26] to the separability of problem (25), the substitution proposed in [26] to $\vec{r}^k = (p_2^k; ..., p_N^k)$ can be performed in parallel for of ADMM wit **if** $\varepsilon < \varepsilon_2^{\text{feasible}}$ **break**. platics (27) and (28) of variables $z = (z_1, \dots, z_N)$ and
 $= 1, \dots, N$.
 Algorithm 2 D_ADMM for Distributed ED
 $z_1^{(x_1, ..., x_N)}$
 Algorithm 2 D_ADMM for Distributed ED
 $z_2^{(x_2, ..., x_N)}$
 Algorithm 2 D_ADMM for Distributed **return** *f P*() and *^P* . *p*₂ $(p_i^x + P_i - \frac{1}{N}P_D)$ even the exgensively, D_0 , D_1
 $M = \begin{bmatrix} p_2D_2^TD_2 & \cdots & p_1D_N^TD_N & 0 \\ p_2D_N^TD_2 & \cdots & p_1D_N^TD_N & 0 \\ \vdots & \vdots & \ddots & \vdots \\ p_2D_N^TD_2 & \cdots & p_1D_N^TD_N & 0 \\ 0 & \cdots & 0 & \frac{1}{p_5}E^2 \end{bmatrix}$

reak.

 if t $M = \begin{bmatrix} b & b \\ c_1 & b_2 \end{bmatrix}$, $M = \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$, $M = \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$, $\begin{bmatrix} b_2 & b_3 & b_4 \end{bmatrix}$, $\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$, $B_2^{2 \sinh k}$ break.
 B_1^{2} is identify matrix with dimension matching vector
 ^{s:} $\begin{vmatrix}\n\vdots & \ddots & \vdots & \vdots \\
p_1^* & p_2(p_1^{k+1} - z_1^{k+1})\n\end{vmatrix}$, $M = \begin{vmatrix}\n\vdots & \ddots & \vdots & \vdots \\
p_3D_N^T D_2 & \cdots & p_3D_N^T D_N & 0 \\
0 & \cdots & 0 & \frac{1}{\rho_3} E^{\lambda}\n\end{vmatrix}$, E^{λ} is identify matrix with dimension matrix P) and *P*.
 F

4.3 Alternating direction method with gaussian back substitution

It can be seen that the model (6) is the linearly constrained separable convex minimization problem. And the objective function is decomposed into *N* blocks, then, the model (6) can be rewritten as:

*E*² is identify matrix with dimension matching vector
$$
\lambda
$$
.
\n*P*) and *P*.
\n**ii direction method with gaussian back**
\n**iii** *D_i* is the coefficient matrix of P_i in constraint (2), for
\n**iv in** P_i is the coefficient matrix of P_i in constraint (2), for
\n**iv in** P_i is the coefficient matrix of P_i in constraint (2), for
\n**iv in** P_i is the coefficient matrix of P_i in constraint (2), for
\n**iv in** P_i is the coefficient matrix of P_i in constraint (2), for
\n**iv in** P_i is the coefficient matrix of P_i in constraint (2), for
\n**iv in** P_i is a identity matrix, and
\n**iv in** P_i is a identity matrix, and
\n**iv in** P_i is a identity matrix. So the
\n**in** $\sum_{i=1}^{N} f_i(P_i)$
\n**in**

Inspired by the efficiency of ADMM, a natural idea for solving (33) is to directly extend the two blocks ADMM scheme descripted in section 3 for (33) with *N* blocks:

l Economic Dispatching Methods Based on Alternating Direction Multiplier Method
\n) into the function on the
\n matter the variables
$$
z_i
$$
. As a
\n solving (33) is to directly extend the two blocks ADMM
\n scheme described in section 3 for (33) with *N* blocks:
\n
$$
y^{k+1} + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right),
$$
\n
$$
P_i^{k+1} := \arg \min_{p_i} \{ f_i(P_i) + \frac{\rho_3}{2} || (\sum_{j=1}^{i-1} P_j^k + P_i + (\sum_{j=1}^{k} P_D) \|^2 \}
$$
\n
$$
\sum_{j=i+1}^{N} P_j^k - P_D) - \frac{1}{\rho_3} \chi^k ||^2 | P_i \in \chi_{1,i} \}, i = 1, \dots, N
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N} P_D \right) \Big|^2
$$
\n
$$
1 + \frac{1}{\rho_2} \left(p_i^k + P_i - \frac{1}{N
$$

$$
\lambda^{k+1} = \lambda^k - \rho_3 \left(\sum_{j=1}^N P_j^k - P_D \right) \tag{35}
$$

phomic Dispatching Methods Based on Alternating Direction Multiplier Method

to the function on the

1 Inspired by the efficiency of ADMM, a natural idea for

the variables z_i . As a

solving (33) is to directly extend t notion on the Inspired by the efficiency of ADMM, a natural idea for
bles z_i . As a solving (33) is to directly extend the two blocks ADMM
scheme descripted in section 3 for (33) with N blocks:
 $+P_i - \frac{1}{N}P_D$), P_i^{k+1} However, [23] points out that the scheme (34)-(35) is not necessarily convergent if no further conditions are posed on the model (33), even though its efficiency has been verified empirically by some recent applications (see [24, 25]). Here, we introduce the Gaussian back substitution proposed in [26] to guarantee the convergence of ADMM with *N* blocks. $|I_i = \chi_{1,i} f, i-1, \dots, N$ (34)
 $\sum_{j=1}^{N} P_j^k - P_D$ (35)

that the scheme (34)-(35) is

f no further conditions are

en though its efficiency has

ome recent applications (see

oduce the Gaussian back

co guarantee the conve heme descripted in section 3 for (33) with N blocks:
 ${}^{k+1} := \arg \min_{p_i} \{f_i(P_i) + \frac{p_i}{2} || (\sum_{j=1}^{i-1} P_j^k + P_i + \sum_{j=1}^{N} P_j^k - P_D) - \frac{1}{p_3} \lambda^k ||^2 | P_i \in \chi_{1,i}, i = 1, \dots, N$ (34)
 $\lambda^{k+1} = \lambda^k - \rho_3 \left(\sum_{j=1}^{N} P_j^k - P_D \right)$ (35)

Ho oired by the efficiency of ADMM, a natural idea for

g (33) is to directly extend the two blocks ADMM

de descripted in section 3 for (33) with N blocks:

= $\arg \min_{p_i} \{f_i(P_i) + \frac{\rho_i}{2} || (\sum_{j=1}^{i-1} P_j^k + P_i + \frac{N}{\rho_3} + P_j^k - P_D$ $\chi^{k+1} = \chi^{k} - \rho_3 \left(\sum_{j=1}^{N} P_j^k - P_D \right)$ (34)
 $\chi^{k+1} = \chi^{k} - \rho_3 \left(\sum_{j=1}^{N} P_j^k - P_D \right)$ (35)
 r, [23] points out that the scheme (34)-(35) is

r, [23] points out that the scheme (34)-(35) is

acarily convergent $j^{(-1)}D^{(-1)} = \frac{1}{\rho_3} \times \left[|l_1 \in \chi_{1,i}, i, i-1, \cdots, i \rangle \right]$ (34)
 $\chi^{k+1} = \chi^k - \rho_3 \left(\sum_{j=1}^N P_j^k - P_D \right)$ (35)

[23] points out that the scheme (34)-(35) is

inv convergent if no further conditions are

model (33), even th From Such that the sensitive convergent if no further conditions are

oconvergent if no further conditions are

odel (33), even though its efficiency has

piprically by some recent applications (see

re, we introduce the *ρ* **c** *n <i>n n <i>n n n <i>n n <i>n n <i>n n* **b** *p n n n n n n n n n n n n n n n* *****n n n n n n n n n n n n n n n n n n n* wevel, [23] points out due to scheme (547-35) is

lecessarily convergent if no further conditions are

on the model (33), even though its efficiency has

verified empirically by some recent applications (see

25]). Here, necessarily convergent if no further conditions are

necessarily convergent if no further conditions are

necessarily complicated (33), even through its efficiency has

necessarily by some recent applications (see

25]). is an ignoring that the scheme (34)-(35) is

signify convergent if no further conditions are

the model (33), even though its efficiency has

led empirically by some recent applications (see

. Here, we introduce the Gaus

After each round of ADMM iteration, i.e. (34)-(35), let

$$
\tilde{w}^k := (\tilde{P}_2^k; \dots; \tilde{P}_N^k; \tilde{\lambda}^k) = (P_2^{k+1}; \dots; P_N^{k+1}; \lambda^{k+1}) = w^{k+1}, (36)
$$

then the Gaussian back substitution step obtains new and corrected w^{k+1} by solving following equation,

$$
H^{-1}M^{T}\left(w^{k+1}-w^{k}\right)=\sigma\left(\tilde{w}^{k}-w^{k}\right),\qquad(37)
$$

where $\sigma \in (0,1)$, and [26] recommend $\sigma \in (0.5,1)$ (or even more aggressively, $\sigma = 1$); (37)
 $\equiv (0.5,1)$ (or

g vector λ .

raint (2), for

matrix, and
 $E \dots E \quad 0$
 $\vdots \quad \vdots \quad \vdots$
 $0 \quad \dots E \quad 0$
 $0 \quad \dots 0 \quad E^{\lambda}$

trix. So the

e solved by

2-4, 25J). Fictc, we introduce the Gaussian back
substitution proposed in [26] to guarantee the convergence
of ADMM with N blocks.
After each round of ADMM iteration, i.e. (34)-(35), let

$$
\tilde{w}^k := (\tilde{P}_2^{k};...; \tilde{P}_N^{k}; \tilde{x}^k) = (P_2^{k+1};...; P_N^{k+1}; \tilde{x}^{k+1}) = w^{k+1}
$$
, (36)
then the Gaussian back substitution step obtains new and
corrected w^{k+1} by solving following equation,
 $H^{-1}M^T(w^{k+1} - w^k) = \sigma(\tilde{w}^k - w^k)$, (37)
where $\sigma \in (0,1)$, and [26] recommend $\sigma \in (0.5,1)$ (or
even more aggressively, $\sigma = 1$);
 $M = \begin{bmatrix} \rho_3 D_2^T D_2 & 0 & \cdots & 0 \\ \rho_3 D_N^T D_2 & \cdots & \rho_3 D_N^T D_N & 0 \\ 0 & \cdots & 0 & \frac{1}{\rho_3} E^{\lambda} \\ 0 & \cdots & 0 & \frac{1}{\rho_3} E^{\lambda} \end{bmatrix}$,
 E^{λ} is identify matrix with dimension matching vector λ .
 $H = diag(\rho_3 D_2^T D_2, ..., \rho_3 D_N^T D_N, \frac{1}{\rho_3} E^{\lambda})$,
 D_i is the coefficient matrix of P_i in constraint (2), for
our ED problem, D_i actually is an identity matrix, and
then it is easy to verify that $H^{-1}M^T = \begin{bmatrix} E & \cdots & E & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & E & 0 \end{bmatrix}$

,

 $0 \mid$

 E^{λ} |

λ

$$
H = diag\bigg(\rho_3 D_2^T D_2, \dots, \rho_3 D_N^T D_N, \frac{1}{\rho_3} E^{\lambda}\bigg),
$$

 D_i is the coefficient matrix of P_i in constraint (2), for our ED problem, D_i actually is an identity matrix, and

then it is easy to verify that
$$
H^{-1}M^T = \begin{bmatrix} E & \cdots & E & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & E & 0 \\ 0 & \cdots & 0 & E^{\lambda} \end{bmatrix}
$$

 (P_i) using back-substitution algorithm [27] just as shown in the *P*^{*x*} $\left[\begin{array}{ccc} M = \begin{bmatrix} \rho_3 D_N^t D_2 \dots \rho_3 D_N^t D_N & 0 \\ 0 & \dots & 0 & \frac{1}{\rho_5} E^{\frac{1}{2}} \end{bmatrix} \right]$
 Preak.
 P i identify matrix with dimension matching vector
 P *E* λ is identify matrix with dimension matching vec (37)
 $\in (0.5,1)$ (or

raint (2), for

raint (2), for

matrix, and
 $E \dots E \quad 0$
 $\vdots \quad \vdots \quad \vdots \quad \vdots$
 $0 \quad \dots E \quad 0$
 $0 \quad \dots 0 \quad E^{\lambda}$

trix. So the

se solved by

shown in the

k substitution $\in (0.5,1)$ (or
raint (2), for
matrix, and
 $E \dots E \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$
 $\in E$ o $\begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$
 $\in E$ o $\begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$
atrix. So the solved by
shown in the
k substitution
disadvantage 1,
 $\in (0.5,1)$ (or
 $\in (0.5,1)$ (or
 E and (2) , for
 $x = 0$
 $\vdots \therefore \vdots \vdots$
 $x = 0$
 $\vdots \therefore \vdots \vdots$
 $x = 0$
 x $\left(\tilde{w}^k - w^k\right)$, (37)

mmend $\sigma \in (0.5,1)$ (or
 $\left.\frac{1}{\rho_3}E^{\lambda}\right]$,
 $\left.\begin{matrix}\nP_i & \text{in constraint (2), for} \\
P_i & \text{in constraint (2), for} \\
P_i & \text{in constraint (2), for} \\
\text{an identity matrix, and} \\
H^{-1}M^T = \begin{bmatrix} E & \dots & E & 0 \\
\vdots & \ddots & \vdots & \vdots \\
0 & \dots & E & 0 \\
0 & \dots & 0 & E^{\lambda}\n\end{bmatrix}$

e block mat in, (37)
 $\mathbf{r} \in (0.5,1)$ (or
 $\mathbf{r} \in (0.5,1)$ (or
 $\mathbf{r} \in (0.5,1)$ (or
 $\mathbf{r} \in (0.5,1)$ and
 $\mathbf{r} \in (0.5,1)$
 $\mathbf{$ *k*), (37)
 $\sigma \in (0.5,1)$ (or

ching vector λ .

ching vector λ .

mstraint (2), for

ity matrix, and
 $\begin{bmatrix} E & \cdots & E & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & E & 0 \\ 0 & \cdots & 0 & E^{\lambda} \end{bmatrix}$

matrix. So the

n be solved by

as shown in F \in (0.5,1) (or

straint (2), for

straint (2), for

y matrix, and
 $\begin{bmatrix} E & \cdots & E & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & E & 0 \\ 0 & \cdots & 0 & E^{\lambda} \end{bmatrix}$

natrix. So the

be solved by

s shown in the

ck substitution

a disadvantage which is nearly a upper triangle block matrix. So the Gaussian back substitution step (37) can be solved by following Algorithm 3.

of ADM_G is that the blocks are updated one after another, which is not amenable for parallelization.

Algorithm 3 ADM_G for Distributed ED

Initialization: $P_i^0 = 0$, $\lambda^0 = 0$, $M > 0$, $\rho_3 > 0$, *Linfeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shanshan*

ADM_G is that the blocks are updated one after another,

iich is not amenable for parallelization.

 Algorithm 3 ADM_G for Distribute $f(P) = 0$, $\sigma \in (0,1)$, $\varepsilon_1^{\text{feasible}} > 0$. *Linfeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shans*
 f ADM_G is that the blocks are updated one after another,

thich is not amenable for parallelization.
 Algorithm 3 ADM_G for Distribute **ADMM step** (prediction step)**:** *Linfeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shanshan P*
 G is that the blocks are updated one after another,
 for $\sinh M$ of $\cosh M$ of $\cosh M$ of $\cosh M$ and $\sinh M$ of $\sinh M$ of $\sinh M$ of $\$ obtain \widetilde{P}_i^k by solving (34) **end** obtain $\tilde{\lambda}^k$ by using (35) **Gaussian back substitution step** (correction step): $\lambda^1 := \lambda^k + \sigma\left(\tilde{\lambda}^k - \lambda^k\right)$ Linfeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shanshan Pan
 G is that the blocks are updated one after another,

not amenable for parallelization.
 ifthm 3 ADM_G for Distributed ED

ation: $P_N^k := \lambda^k + \sigma \left(\tilde{P}_N^k - P_N^k \right)$ *k* are updated one after another,
 k for Distributed ED
 k for Distributed ED
 k $P_t^{0} := 0$, $\sigma \in (0,1)$, $e_t^{\text{feasible}} > 0$.
 $\sigma = (0,1)$, $e_t^{\text{feasible}} > 0$.
 $\sigma = (0,1)$, $e_t^{\text{feasible}} > 0$.
 b P_t^k to production step):
 for $i = N-1$, $N-2$, ..., 2
Here the x -update (42) can be carried out in parallel, $P_i^{k+1} := P_i^k + \sigma \left\{ \left(\tilde{P}_i^k - P_i^k \right) - \left(\tilde{P}_{i+1}^k - P_{i+1}^k \right) \right\}$ amenable for parallelization.
 (a)
 (a) enable for parallelization.
 $x \cdot t \sum_{i} x_{i} = 0, i = 1,...N,$ (41)
 $\overrightarrow{B} \cdot \overrightarrow{B} = 0, \quad \lambda^{0} = 0, M > 0, \quad \rho_{3} > 0,$
 $y' (x_{i}) = a_{i}x_{i}^{2} + (b_{i} + 2a_{i} \frac{1}{N}P_{0})x_{i} + a_{i} (\frac{1}{N}P_{0})^{2} + b_{i} \frac{1}{N}P_{0} + c_{i}.$
 $y = 0, \quad \lambda^{0} = 0, M >$ are updated one after another,

allelization.
 $s.t.\sum_{i}^{N} x_{i} = 0, i = 1,...N,$
 $\begin{aligned}\n &\text{Distribute H.D.} \\
 &\text{where} \\
 &= 0, M > 0, \quad \rho_{3} > 0, \\
 &\text{where} \\
 &\text{else } > 0.\n \end{aligned}\n \quad \text{where} \\
 &\text{where} \\
 &\text{where} \\
 &\text{where} \\
 &\text{where} \\
 &\text{where} \\
 &\text{where} \\
 &\text{where$ **end** $P_1^{k+1} := \tilde{P}_1^k$ **EVALUATION CONSUMPLY 10**

alian: $P_i^k = 0$, $P_i^k = -1, ..., N$
 EVALUATION CONSUMPLY $P_i^k = 1 = P_i^k$
 EVALUATION CONSUMPLY $P_i^k = -P_i^k - \sigma\left(\left[\frac{P_i^k - P_i^k}{P_i^k - P_i^k}\right]\right)$
 EVALUATION CONSUMPLY $P_i^k = 1 = P_i^k$
 EVALUATION C $1 - p^k \parallel \parallel \parallel_2 k + 1 \parallel_2 k \parallel \parallel$ $\max_i \left\{ \left\| P_i^{\kappa+1} - P_i^{\kappa} \right\|_1 \right\} \left\| \lambda^{\kappa+1} - \lambda^{\kappa} \right\|_1$ Algorith sing (35)

solution step (correction step):
 $\vec{a}^k - \lambda^k$
 $\vec{a}^k - \lambda^k$
 $\vec{b}^k - \lambda^k$
 35)

on step (correction step):
 $u^{k+1} = u^k + \overline{x}^{k+1}$,

()

()

where $\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$ is the average of x
 $\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$ is the average of x
 $\overline{y} = -P_t^k - P_{t+1}^k$

with the subvectors x_i , σ ∈ (0,1), $\varepsilon_i^{\text{feasible}} > 0$.

S (xi) = $u_i x_i$ (not like x_i) = $u_i x_i$ (not like x_i) = $u_i x_i$ (not like x_i).

IThe exchange problem an be solved via ADM

in \tilde{R}_i^k by solving (34)

and \tilde{R}_i^k by solving *x*_i^{k+1} := arg min_{x_i} { $g^t(x_i) + \frac{p_4}{2} || x_i$

ing (35)
 itution step (correction step):
 $x_i^{k+1} := ax + \frac{x}{i} + x^{k+1}$,
 $x_k^{k+1} := ax^k + \frac{x}{i} + x^{k+1}$,
 $y - 2, ..., 2$
 $y - 2, ..., 2$
 $\left\{ \left(\tilde{P}_i^k - P_i^k \right) - \left(\tilde{P}_{i+1}^k$ by solving (34)
 i i π^{i} $\pi^{$ using (35)
 bstitution step (correction step):
 $\tilde{\lambda}^k - \lambda^k$)
 $N-2, ..., 2$
 $\sigma \{(\tilde{P}_i^k - P_i^k) - (\tilde{P}_{i+1}^k - P_{i+1}^k)\}$
 $\sigma \{(\tilde{P}_i^k - P_i^k) - (\tilde{P}_{i+1}^k - P_{i+1}^k)\}$
 $\left\{\n\begin{array}{ccc}\n\mu^k + 1 - \mu^k & \mu^k \\
\sigma^k + \sigma^k & \mu^k & \$ *A* = $\frac{1}{2}$, *M* > 0, *P*₃ > 0,
 *P*₄ = $\frac{1}{2}$, *P*_{*P*} ($\frac{1}{2}$, *P*_{*P*} + *A*<sub>($\frac{1}{2}$, *P*_{*P*}) $\frac{1}{2}$
 *P*₁ = $\frac{1}{2}$, *P*₂ = $\frac{1}{2}$, *P*₂ = $\frac{1}{2}$, *P*₂ = $\frac{1}{2}$, *P*₂ = $\frac{$ The exchange problem can be solved

dediction step):
 Problem can be solved
 $x_i^{k+1} = \arg \min_{x_i} \{g'(x_i) + \frac{\rho_i}{2} || x_i - x_i^k \}$

using (35)
 P $u^k ||^2 | x_i \in \chi_{3,i} \} = 1, \dots, N$
 P $\left(\lambda^k - \lambda^k\right)$
 $\left(\lambda^k - \lambda^k\right)$
 $\left(\lambda^k - \$ 1), $p_3 > 0$, $p_3 > 0$, $g'(x_i) = a_i x_i^2 + (b_i + 2a_i \frac{1}{N} P_D) x_i + a_i (\frac{1}{N} P_D) + b_i \frac{1}{N} P_D + c_i$.

The exchange problem can be solved via ADMM, the iterations of the scaled form of ADMM [19]:

1)

1) $x_i^{k+1} = \arg \min_{x_i} \{g'(x_i) + \frac{\rho_i$ $\varepsilon = \max \left\{ \frac{\left(\frac{1}{\epsilon}\right)^{n}}{\left(\frac{1}{\epsilon}\right)^{n}} \right\}, \frac{\left(\frac{1}{\epsilon}\right)^{n}}{\left(\frac{1}{\epsilon}\right)^{n}} \right\},$ $P_t^0 := 0$, $A^0 = 0$, $M > 0$, $\rho_3 > 0$,
 $g'(x_i) = a_i x_i^2 + (b_i + 2a_i \frac{1}{N} P_D)x_i + a_i \left(\frac{1}{N} P_D\right)^2 + b_i \frac{1}{N} P_D + c_i$.
 M The exchange problem can be solved via ADMM, the
 M The exchange problem can be solved via ADMM, the
 M 0, $\sigma \in (0,1)$, $\varepsilon_1^{\text{times}}$ is α_2^{times} = 0.
 α_3^{times} is α_4^{times} is α_5^{times} is α_6^{times} is α_7^{times} is α_8^{times} (prediction step):
 Since (prediction step):
 Since (prediction step):
 The exchange problem can be solved via ADMM, the
 iterations of the scaled form of ADMM [19]:

1..., N :

1..., \overline{X}^k by solving (34)
 \overline{X}^k by solving (34)
 \overline{X}^k by solving (34)
 \overline{X}^k by solving **if** $\varepsilon \leq \varepsilon_1^{\text{feasible}}$ **break ADMM** step (prediction step):

for $i = 1,..., N$:

obtain \tilde{P}_i^k by solving (34)

and

obtain \tilde{Z}_i^k by solving (35)
 Caussian back substitution step (correction step):
 $\lambda^{k+1} = \lambda^k + \sigma(\tilde{P}_N^k - P_N^k)$
 $P_N^k = \lambda$ obtain \overline{P}_i^k by solving (34)
 columin \overline{A}_i^k by using (35)

columin \overline{A}_i^k by using (35)
 $\mu^k \parallel \begin{vmatrix} 1 & k \\ k & k \end{vmatrix}$ $\begin{vmatrix} 1 & k \\ k & k \end{vmatrix}$ $\begin{vmatrix} 1 & k \\ k & k \end{vmatrix}$ $\begin{vmatrix} 1 & k \\ k & k \end{vmatrix}$ **consisinn back**

The sequential nature of ADM_G motivates us to consider using a scheme that updates all the *N* blocks consider using a scheme that updates all the N blocks
 $s_2^k = -\rho_4(\overline{x}^k - \overline{x}^{k-1}, \dots, \overline{x}^k - \overline{x}^{k-1})$.

in parallel. By substituting $x_i = P_i - \frac{1}{N}P_D$ into problem

if $r_2^k \leq \varepsilon_2^{\text{pri}}$ and $s_2^k \leq \varepsilon_2^{\text{d$

$$
P_i^{min} - \frac{1}{N} P_{D,t} \le x_{i,t} \le P_i^{max} - \frac{1}{N} P_{D,t},
$$
\n(38)

$$
x_{i,t} - x_{i,t-1} \leq UR_i \tag{39}
$$

$$
x_{i,t-1} - x_{i,t} \le DR_i \,, \tag{40}
$$

Similar to χ_1 and $\chi_{1,i}$, we define the set $\chi_{3,i} = \{x_i \mid (38)(39)(40)\}, i = 1,..., N$ and $x_{i,t-1} = 0x_i$, (39)
 $i = x_{i,t} \leq DR_i$, (40) We are best
 $x_{i,i}$, we define the set
 $x_{i,i}$, we define the set
 $x_{i,i}$, we define the set
 $x_{i,i}$, we additionally the set
 $x_{i,i} = 1,..., N$ and
 $x_{i} = 1,..., N$ and
 $x_{i} = 2x_{$

$$
\mathop{\mathit{min}} \, \sum_{i=1}^N g^i_{x_i \in \chi_{3,i}} \left(x_i \right)
$$

no Luo and Shanshan Pan
\ns.t.
$$
\sum_{i}^{N} x_i = 0, i = 1,...N,
$$
 (41)
\n $b_i + 2a_i \frac{1}{N} P_D) x_i + a_i \left(\frac{1}{N} P_D\right)^2 + b_i \frac{1}{N} P_D + c_i.$
\n2 problem can be solved via ADMM the

where

ung Zhang, Jiangyao Luo and Shanshan Pan

\n
$$
s.t. \sum_{i}^{N} x_{i} = 0, i = 1, \ldots N,
$$
\n(41)

\nwhere

\n
$$
g^{i}(x_{i}) = a_{i}x_{i}^{2} + (b_{i} + 2a_{i}\frac{1}{N}P_{D})x_{i} + a_{i}\left(\frac{1}{N}P_{D}\right)^{2} + b_{i}\frac{1}{N}P_{D} + c_{i}.
$$
\nThe exchange problem can be solved via ADMM, the iterations of the scaled form of ADMM [19]:

\n
$$
x_{i}^{k+1} := \arg\min_{x_{i}} \{g^{i}(x_{i}) + \frac{\rho_{4}}{2} || x_{i} - x_{i}^{k} + \frac{1}{N} + \frac{
$$

The exchange problem can be solved via ADMM, the iterations of the scaled form of ADMM [19]:

ng Zhang, Jiangyao Luo and Shanshan Pan
\n
$$
s.t.\sum_{i}^{N} x_{i} = 0, i = 1,...N,
$$
\n(41)
\nwhere
\n
$$
g^{i}(x_{i}) = a_{i}x_{i}^{2} + (b_{i} + 2a_{i}\frac{1}{N}P_{D})x_{i} + a_{i} \left(\frac{1}{N}P_{D}\right)^{2} + b_{i} \frac{1}{N}P_{D} + c_{i}.
$$
\nThe exchange problem can be solved via ADMM, the
\nterations of the scaled form of ADMM [19]:
\n
$$
x_{i}^{k+1} := \arg \min_{x_{i}} \{g^{i}(x_{i}) + \frac{\rho_{4}}{2} || x_{i} - x_{i}^{k} + \frac{1}{N} + \frac{1}{N}u^{k} ||^{2} | x_{i} \in \chi_{3,i}\}i = 1,...,N
$$
\n(42)
\n
$$
u^{k+1} := u^{k} + \overline{x}^{k+1},
$$
\n(43)
\nwhere $\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_{i}$ is the average of x_{1}, \dots, x_{N} .
\nHere the *x*-update (42) can be carried out in parallel,
\nwith the subvectors x_{i} updated by *N* separate

$$
u^{k+1} := u^k + \overline{x}^{k+1}, \tag{43}
$$

where
$$
\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i
$$
 is the average of x_1, \dots, x_N .

 $\left\{ \left(\tilde{P}_i^k - P_i^k \right) - \left(\tilde{P}_{i+1}^k - P_{i+1}^k \right) \right\}$ with the subvectors x_i updated by *N* separate *x.t.* $\sum_{i}^{N} x_{i} = 0, i = 1,...N,$ (41)
 $= a_{i}x_{i}^{2} + (b_{i} + 2a_{i} \frac{1}{N}P_{D})x_{i} + a_{i} \left(\frac{1}{N}P_{D}\right)^{2} + b_{i} \frac{1}{N}P_{D} + c_{i}.$

exchange problem can be solved via ADMM, the

so of the scaled form of ADMM [19]:
 x_{i+1}^{k+1} (41)
 $\binom{p}{D}^2 + b_i \frac{1}{N} P_D + c_i$.

d via ADMM, the

[19]:
 $\cdot x_i^k + \overline{x}^k +$

(42)

(43)

(43)

..., x_N .

ied out in parallel,

by N separate

all the x_i^{k+1} , and

rs handing the x_i *xx*. $\sum_{i}^{N} x_{i} = 0, i = 1,...N,$ (41)
 arrow $(x_{i}) = a_{i}x_{i}^{2} + (b_{i} + 2a_{i} \frac{1}{N}P_{D})x_{i} + a_{i} \left(\frac{1}{N}P_{D}\right)^{2} + b_{i} \frac{1}{N}P_{D} + c_{i}$ **.

The exchange problem can be solved via ADMM, the

rations of the scaled form of ADM** minimizations. The *u*-update gathers all the x_i^{k+1} , and broadcasts u^{k+1} back to the processors handing the x_i updates.

The exchange problem can be solved via ADMM, the

iterations of the scaled form of ADMM [19]:
 $x_i^{k+1} = \arg \min_{x_i} \{g^i(x_i) + \frac{\rho_i}{2} || x_i - x_i^k + x_k^{k+1} + \frac{1}{2} || x_i^k - x_i^k + x_i^{k+1} + \cdots \}$

correction step):
 $u^{k+1} ||^2 |x_i = \chi_{3,i} \$ *i*, $P_t^* = P_t^* \left| \left| \frac{1}{h} \left\{ \frac{z^{k+1} - z^k}{k!} - \frac{z^k}{k!} - \frac{z^k}{k!} \right\} \right|$ Here the *x* -update (42) can be carried out in parall

with the subvectors x_i updated by *N* separe

broadcasts u^{k+1} back to the proce \overline{N} P_D into problem **if** $r_2^k \leq \varepsilon_2^{\text{pri}}$ and $s_2^k \leq \varepsilon_2^{\text{dual}}$ **break**. ..., 2
 $\left.\begin{array}{lll}\n & \text{Here the } x \text{ -update (42) can be carried out in }\n\\
 & \text{with the subvectors } x_i \text{ updated by } N \text{ is } n\text{imimizations. The } u \text{ -update patterns all the } x_i^k \text{ in this, } x_i^k + 1 \text{ back to the processors handling}\n\end{array}\right\}$
 $\left.\begin{array}{lll}\n\left|\frac{k}{k}\right| & k^{k+1} - \lambda^k \left|\frac{k}{k}\right| & k^{k+1} - \lambda^k \left|\frac{k}{k}\right| & k^{k+1} - \lambda^k \left|\frac{k}{k}\right| & k$ end
 $F_1^{k+1} := \tilde{P}_1^k$
 $\varepsilon = \max \left\{ \frac{\left| m x_1 \right| \left| P_i^{k+1} - P_i^k \right| \right|}{\left| P_i^1 - P_i^0 \right| \right\}} \cdot \left\{ \frac{2(1-x_1)^{k+1}}{k! \left| 2(1-x_1^0) \right| \right\}} \right\}$,
 $\text{Hence, the first term } f(P) \text{ and } P$.

If $\varepsilon \le \varepsilon_1^{\text{double}}$ break.

end
 $\text{Hence, the second term } f(P) \text{ and$ = \overline{R}_i^{k}
 $\times \left\{ \frac{\arg x_i}{\left\| P_i^{k+1} - P_i^k \right\|_1} \right\}, \left\{ \frac{\lambda^{k+1} - \lambda^k}{\left\| P_i^k - P_i^k \right\|_1} \right\},$
 $\left\{ \frac{\text{Algorithm 4 E ADMM for Distributed ED}{\left\| P_i^k - P_i^0 \right\|_1} \right\},$
 $\left\{ \frac{\text{Algorithm 4 E ADMM for Distributed ED}{\left\| P_i^k - P_i^0 \right\|_1} \right\},$
 $\left\{ \frac{\text{Algorithm 4 E ADMM for Distributed ED}{\left\|$ *iax*, $\left\{\left\|P_i^{k+1} - P_i^k\right\|_1\right\}$, $\left\{\left\|Z_i^{k+1} - Z_i^0\right\|_1\right\}$, $\left\{\left\|Z_i^{k+1}$ *i* $\left\|P_i^{1} - P_i^{0}\right\|_1$
 i $\left\|A^1 - A^0\right\|_1$
 i initialization: $x_i^0 = 0$, $u^0 = 0$, $M > 0$,
 p.
 p.
 p.
 i $e^{pi i} > 0$, $e^{4\sin 30}$, $e^{4\sin 30}$
 ii e^{42}
 ii (42)
 iii $i = 1,..., N$ (parallel)
 ii end
 orientary $f(P)$ on P .
 for $k = 0, ..., M$
 for $k = 1, ..., N$ (parallel)
 14 Exchange ADMM algorithm
 end
 if $r_2^k \leq r_2^{m_1} - x^{k+1}, \dots, r_2^{k+1} = x^{k+1}$

onside **Exchange ADMM algorithm**
 Solution $\begin{array}{ccc}\n\text{for each unit } i = 1,..., N : (\text{para})\n\end{array}$
 Solution
 x = *x*₁*x* = *x*₁*x* = *x*₁*x* = *x*₂*x* = *x*²*x* + *x*²*x* + *x*²*x* + *x*²*x* + *x*²*x* 1 nature of ADM_G motivates us to

scheme that updates all the *N* blocks

abstituting $x_i = P_i - \frac{1}{N}P_D$ into problem

abstituting $x_i = P_i - \frac{1}{N}P_D$ into problem

ats (3)(4)(5) become as follows:

and
 $P_{D,i} \le x_{i,i} \le P_i^{max$ **Algorithm 4** E**_**ADMM for Distributed ED Initialization: $x_i^0 := 0$, $u^0 = 0$, $M > 0$, $\rho_4 > 0$, $\varepsilon^{\text{pri}} > 0$, $\varepsilon^{\text{dual}} > 0$. $x_i^{k+1} := \arg \min_{x_i} \{g^i(x_i) + \frac{\rho_i}{2} || x_i - x_i^k + \frac{\pi^k}{2} + \frac{\mu^k}{2} || x_i - x_i^k + \frac{\pi^k}{2} + \frac{\mu^k}{2} || x_i - x_i^k + \frac{\pi^k}{2} + \cdots$ (42)
 $u^{k+1} := u^k + \bar{x}^{k+1}$, (43)

there $\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$ is the average of x_1, \dots, x_N .

Here th \mathbf{r}_i = arg mm_{x_i} { $g(x_i) + \frac{1}{2} ||x_i - x_i + x +$
 $u^k ||^2 |x_i \in \chi_{3,i}$ } i = 1, ..., N (42)
 $u^{k+1} := u^k + \overline{x}^{k+1}$, (43)
 $\overline{x} = \frac{1}{N} \sum_{i=1}^N x_i$ is the average of x_1, \dots, x_N .
 \overline{x} e the x -update (42) can be ca (42) **end** (43) $\left(x_1^{n+1} - x^{n+1}, \cdots, x_N^{n+1} - x^{n+1} \right),$ ions. The u -update gathers all the x_i^{k+1} , and
 u^{k+1} back to the processors handing the x_i
 ithm 4 E_ADMM for Distributed ED

tion: $x_i^0 = 0$, $u^0 = 0$, $M > 0$, $\rho_4 > 0$,
 ε dual > 0 .
 $= 0,..., M$

each u $\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$ is the average of x_1, \dots, x_N .

the *x*-update (42) can be carried out in parallel,

he subvectors x_i updated by *N* separate

zations. The *u*-update gathers all the x_i^{k+1} , and

sts u^{k $\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$ is the average of x_1, \dots, x_N .

the *x*-update (42) can be carried out in parallel,

the subvectors x_i updated by *N* separate

izations. The *u*-update gathers all the x_i^{k+1} , and

asts $=\frac{1}{N}\sum_{i=1}^{N} x_i$ is the average of x_1, \dots, x_N .

the x -update (42) can be carried out in parallel,

e subvectors x_i updated by N separate

titions. The u -update gathers all the x_i^{k+1} , and
 rithm 4 E_ADMM IM for Distributed ED

, $u^0 = 0$, $M > 0$, $\rho_4 > 0$,

..., N : (parallel)
 \ldots, N : $\left(\text{parallel}\right)$
 $\ldots, x_N^{k+1} - \overline{x}^{k+1}$),
 $\ldots, \overline{x}^k - \overline{x}^{k-1}$). $\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$ is the average of x_1, \dots, x_N .

the x -update (42) can be carried out in parallel,

the subvectors x_i updated by N separate

izations. The u -update gathers all the x_i^{k+1} , and

asts $=\frac{1}{N}\sum_{i=1}^{N} x_i$ is the average of x_1, \dots, x_N .
 $\sum_{i=1}^{N} x_i$ is the average of x_1, \dots, x_N .
 $\sum_{i=1}^{N} x_i$ where the explored by N separate

tions. The u -update gathers all the x_i^{k+1} , and
 u^{k+1} back to M for Distributed ED
 $u^0 = 0$, $M > 0$, $\rho_4 > 0$,

..., N :(parallel)

..., $x_N^{k+1} - \overline{x}^{k+1}$,
 \vdots , \vdots , $\overline{x}^k - \overline{x}^{k-1}$.,
 $\frac{1}{2} \le \varepsilon_2^{\text{dual}}$ break. **end** alization: $x_i^{\circ} := 0$, $u^{\circ} = 0$, $M > 0$, $\rho_4 > 0$,
 > 0 , $\varepsilon^{\text{dual}} > 0$.
 $k = 0,..., M$

for each unit $i = 1,..., N$: (parallel)

(42)

end

(43)
 $r_2^{k+1} = (x_1^{k+1} - \overline{x}^{k+1}, ..., x_N^{k+1} - \overline{x}^{k+1}),$
 $s_2^k = -\rho_4 (\overline{x}^k - \overline$ \mathbf{D} , *k* condersts. The α -update gathers and the x_i , and
 k condersts u^{k+1} back to the processors handing the x_i
 Algorithm 4 E_ADMM for Distributed ED

Initialization: $x_i^0 = 0$, $u^0 = 0$, $M > 0$, $\rho_4 > 0$,
 N Fractions. The ^{*n*} -place gamers an the *x_i*, and

products:
 Algorithm 4 E_ADMM for Distributed ED

Initialization: $x_i^0 = 0$, $u^0 = 0$, $M > 0$, $\rho_4 > 0$,
 $\varepsilon^{pri} > 0$, $\varepsilon^{dual} > 0$.
 for each unit $i = 1,..., N$: (p

can be extended and applied to the case that DC power flow network constraints being considered.

5. Numerical Results and Analysis

 (x_i) and (x_i) $\sum_{i=1}^{i} g_{x_i \in \chi_{3,i}}^i(x_i)$ fully distributed algorithms for solving the ED problem, in this section, we use eight kinds of unit characteristic data In order to facilitate the comparison of four kinds of fully distributed algorithms for solving the ED problem, in

No.		Total							
		$\overline{2}$	3	$\overline{4}$	5	6	7	8	units
1	12	11	θ	$\mathbf{0}$	1	$\overline{4}$	θ	$\mathbf{0}$	28
$\overline{2}$	13	15	$\overline{2}$	$\mathbf{0}$	$\overline{4}$	$\mathbf{0}$	$\mathbf{0}$	1	35
3	15	13	$\overline{2}$	6	3			3	44
4	15	11	$\mathbf{0}$	1	$\overline{4}$	5	6	3	45
5	15	13	3	7	5	3	$\overline{2}$	1	49
6	10	10	$\overline{2}$	5	7	5	6	5	50
7	17	16	1	3	1	7	2	4	51
8	17	10	6	5	\overline{c}	1	3	7	51
9	12	17	4	7	5	$\overline{2}$	θ	5	52
10	13	12	5	7	$\overline{2}$	5	4	6	54
11	46	45	8	$\mathbf{0}$	5	$\mathbf{0}$	12	16	132
12	40	54	14	8	3		9	13	156
13	50	41	19	11	4	$\overline{4}$	12	15	156
14	51	58	17	19	16		2	1	165
15	43	46	17	15	13	15	6	12	167

Table 1. Number of units per problem case

as the basic unit data, and the combination is 15 cases. The data of eight units and load demands can be found in [28] and [29].The specific combination is shown in Table 1, and the total number of time periods are $T = 24$. All the ³⁵⁰⁰ algorithms in this paper run the software environment are MATLAB R2014a, the computer processor for the Intel (R) Core (TM) i7-4790 CPU 3.60GHz, memory is 8.0GB. The quadratic programming problem (QP problem) is calculated using CPLEX12.6.2. Unless otherwise specified, we used the parameter values $M = 10^3$, $\sigma = 0.5$. as the basic unit data, and the combination is 15 cases. The

data of eight units and load demands can be found in [28]

and [29].The specific combination is shown in Table 1,

algorithms in this paper run the software en

algorithms in this paper run the software environment are
\nMATLAB R2014a, the computer processor for the Intel
\n(R) Core (TM) i7-4790 CPU 3.60GHz, memory is 8.0GB.
\nThe convergence analysis is the parameter values of the DMS (invariance) is given by
\n
$$
\varepsilon_1^{feasible} = 0.001, \quad \varepsilon_2^{feasible} = 0.01, \quad \varepsilon_1^{abs} = 10^{-3}, \quad \varepsilon_1^{rel} = 10^{-2},
$$
\n
$$
\varepsilon_1^{pri} = \sqrt{NT} \varepsilon_1^{abs} + \varepsilon_1^{rel} \text{ max } \left\{ \left\| \left(P_1^{k+1}; \dots; P_N^{k+1} \right) \right\|, \left\| - z^{k+1} \right\| \right\},
$$
\n
$$
\varepsilon_1^{dual} = \sqrt{NT} \varepsilon_1^{abs} + \varepsilon_1^{rel} \text{ max } \left\{ \left\| \left(P_1^{k+1}; \dots; P_N^{k+1} \right) \right\|, \left\| - z^{k+1} \right\| \right\},
$$
\n
$$
\varepsilon_2^{abs} = 10^{-4}, \quad \varepsilon_2^{rel} = 10^{-5}, \quad \varepsilon_2^{pri} = \sqrt{NT} \varepsilon_2^{abs} + \varepsilon_2^{rel} \text{ max } \left\{ \left\| \left(x_1^{k+1}; \dots; x_n^{k+1} \right) \right\| \right\},
$$
\n
$$
\varepsilon_2^{dual} = \sqrt{NT} \varepsilon_2^{abs} + \varepsilon_2^{rel} \text{ max } \left\{ \left\| \left(x_1^{k+1}; \dots; x_n^{k+1} \right) \right\| \right\},
$$
\n
$$
\varepsilon_2^{dual} = \sqrt{NT} \varepsilon_2^{abs} + \varepsilon_2^{rel} \text{ max } \left\{ \left\| \left(x_1^{k+1}; \dots; x_n^{k+1} \right) \right\| \right\},
$$
\n
$$
\varepsilon_2^{dual} = \sqrt{NT} \varepsilon_2^{abs} + \varepsilon_2^{rel} \text{ max } \left\{ \left\| \left(x_1^{k+1}; \dots; x_n^{k+1} \right) \right\| \right\},
$$

The convergence of the ADM-G for solving multi-block problem is proved in [26]. The convergences of the other three fully distributed ED methods can be obtained according to the convergence of the classical ADMM descripted in Section 3 of this paper. The primal residual and the dual residual, which are the features of ADMM convergence, also converge to 0 (can be seen in the following simulation results). And the convergence of the classical ADMM can be found in [19, 30]. Because that *f*^{pri} = $\sqrt{NT} \varepsilon_1^{abs} + \varepsilon_1^{red} \max \{ \left\| (P_1^{k+1}; \dots, P_n^{k+1}) \right\|, \left\| - \varepsilon^{k+1} \right\| \}$,
 f^{dual} = $\sqrt{NT} \varepsilon_1^{abs} + \varepsilon_1^{red} \left\| (\rho_1 u_1^{k+1}, \dots, \rho_1 u_n^{k+1}) \right\|, \varepsilon_2^{abs} = 10^{-4}$,
 f f g^{dual} = $\sqrt{NT} \varepsilon_2^{obs} + \$ (6) are linear, the convergences of these fully distributed ED methods can be theoretically guaranteed also. solving our ED cases when 1 3 4 *^ρ* , , *^ρ ^ρ* are 0.125 and ² *^ρ* is 10.

speed. In our simulations, these fully distributed algorithms have the best performances in convergence speed while

Fig. 1. Comparison of convergence speed for four methods

Fig. 3. ADMM dual residual values

Now, we report some numerical results about convergence the Case 1 in Table 1 while iteration number is $M = 80$. Fig. 1 shows the results of four algorithms for solving The black line in this Figure 1 labeled "Cplex" represents the final optimal value which obtained by using CPLEX to directly solve model (6) with default setting. As can be seen in this figure, after about 40 iterations, four methods obtain nearly equal optimal values. All these values are very close to the optimal values reported by CPLEX. And all these methods obtain nearly same solutions. However, their performances in convergence speed are different. The ADMM, E_ADMM, and D_ADMM have the similar which is the nearest to $f(P)_{Cplex}$, however, Fig. 4 shows performances and can reach the real optimal value after
about 10 iterations ADM G is the slowest one which need that ADMM convergence very slowly for this $\rho_1 = 0.01$. about 10 iterations. ADM_G is the slowest one, which need about 40 iterations. *Linfeng Yang, Tingting Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and*

seen in this figure, after about 40 iterations, four methods

So the operalistic potential values are also the operal of the original values are

Fig. 2 and Fig. 3 respectively report the primal residual $s_1^k = P^k - z^k$ and dual residual $s_1^k = \rho_1(z^k - z^{k+1})$ of the calculated values classic ADMM in the algorithm 1 while solving Case 1. It can be seen that these two values converge to 0 as the number of iterations increases. Fig. 1, 2, and 3 illustrate the At the same time, we know the sooner the constraint is convergence procedure of classical ADMM for solving
satisfied as the penalty parameter increases. On the contrary, Case 1.

5.2 Stability analysis

The stabilities of four fully distributed algorithms, which are based on ADMM, are influenced, more or less, by the setting of penalty parameter. In order to facilitate the comparison of the influence of penalty parameter variation on the stability of the algorithms, we solve the same test case with different methods by setting different penalty parameters.

Fig. 4 gives the comparison of different penalty parameters ρ_1 for algorithm 1 (ADMM) while solving Case 1. In this figure, the line labeled "Cplex" represents the optimal value obtained by CPLEX, which can be denoted as $f(P)_{Cplex}$. Furthermore, we denote the final *F*² = *P*⁻ = *C*² and dual residual s₁^c = *P*₁(*C*² = ⁻z²^{*C*}) of the calculated value of the conductor of the calculate of the conductor of the singlificantly less the number of iterations increases. we can denote $e_{ADMM} = \frac{f(P)_{ADMM} - f(P)_{Cplex}}{f(P)_{Culer}}$ which represents thms, we solve the same test

a by setting different penalty

aarison of different penalty
 $A = 0.5$
 $=\frac{f(P)_{ADMM}-f(P)_{Cplex}}{f(P)_{Cplex}}$ which represents **EXERIMING THE CONSTRAINER CONSULTER ANCHE SERVIDE DIMM, are influenced more or less, by the performances while set
ality parameter. In order to facilitate the the influence of penalty parameter variation
of the algorithm** the relative error of obtained final objective value for

ADMM. Fig. 5 gives the comparison of different penalty parameters ρ_1 for algorithm 1 in the final objective values.

Fig. 4. Convergence comparison of ADMM with different ρ_1

 $-z^{k+1}$) of the calculated value and the CPLEX calculation is about g *Zhang, Guo Chen, Zhenrong Zhang, Jiangyao Luo and Shanshan Pan*

all those values are

assume preset tolerances for all different parameters in our

eded by CPLEX. And simulations, but the algorithm stability is siffe As can be seen in Fig. 4, ADMM can converge with the same preset tolerances for all different parameters in our simulations, but the algorithm stability is different. See Fig. 4 and Fig. 5 simultaneously. Fig. 5 shows that when $\rho_1 = 0.01$, ADMM obtains the best final objective value mg Zhang, Jiangyao Luo and Shanshan Pan
As can be seen in Fig. 4, ADMM can converge with the
same preset tolerances for all different parameters in our
simulations, but the algorithm stability is different. See
Fig. 4 and When the penalty parameter is 0.125, the convergence rate is fastest and the percentage difference between the 0.02% which is a little more than the e_{ADMM} for $\rho_1 =$ 0.01 but is significantly less than the e_{ADMM} for $\rho_1 > 1$. the more easily meet the objective function requirements.
In addition, we do not give the detailed results for the other two methods, D_ADMM and E_ADMM because that, in our simulations, they have almost the similar numerical performances while setting different penalty parameters.

Fig. 5. Comparison of for ADMM with different ρ_1

Fig. 6. Convergence comparison of ADM_G with different ρ_3

Fig. 6 gives the comparison of different ρ_3 for algorithm 3 (ADM_G) while solving Case 1. Fig. 7 gives the comparison of different penalty parameters ρ_3 for Fully Distributed Economic Dispatching Methods Based on Altern
Fig. 6 gives the comparison of different ρ_3 for of CPLEX t
algorithm 3 (ADM G) while solving Case 1. Fig. 7 gives results are s
the comparison of differen *eADMM* . It can be seen in Fig. 6, ADM_G can converge with the same preset tolerances for all different penalty parameters. But, different to ADMM, the objective value convergence procedures of ADM_G for different penalty parameters are very similar, and the convergence speed is a little slow for the smaller penalty parameters. Now, let's only 3 cases' " e_{D_ADMM} " that are 0.01%). E_ADMM look at Fig. 7. It can be seen that ADM_G obtain the best Fig. 6 gives the comparison of different ρ_3 for of CPLEX to solve the 15 calls algorithm 3 (ADM G) while solving Case 1. Fig. 7 gives results are shown in Table 2. Hold algorithm 3 (ADM G) while solving Case 1. Fig. 7 parameter is 0.125. When the penalty parameter is 0.01 or e_{EADMM} that are 0.01%). ADMM consumes the least Fig. 6 gives the comparison of different ρ_3 for of CPLEX to solve the 15 cases of algorithm 3 (ADM_G) while solving Case 1. Fig. 7 gives results are shown in Table 2. In the comparison of different penalty parameters have $f(P)_{ADM-G} < f(P)_{Cplex}$. Actually, AMD_G can push e_{ADM-G} to close to 0 by improving the tolerance **EXERENT CONSUMPTED ASSEM CONSUMPTED THEORY AND CONSUMPTED TO THE CONSUMPTED THEORY OF THEORY CONSUMPTED THEORY CONSUMPTED THEORY CONSUMPTED THEORY CONSUMPTED THEORY STATE every similar, and the convergence speed is a CPU** *Fully Distributed Economic Dispatching Methods Based on Alternating Direction Multiplier Methods in Space of Tablem Space of Tablem Space of Tablem Space of Tablem Space of GPLEX to solve the 15 cases of Tablem Space of* argonality can be different penalty parameters P_3 for the comparison of different penalty defined as erro
 e P_{ADMM} . It can be seen in Fig. 6, ADM G can converge mum

with the same preset tolerances for all differen $\varepsilon_1^{feasible}$ **.**

5.3 Calculation speed comparison

We use four fully distributed algorithms and the solver

different ρ_3

of CPLEX to solve the 15 cases of Table 1. The simulation results are shown in Table 2. In this table, column "time" represents the execution time; " $e_X < 0$ " represents relative error of objective value for method " *X*"; " *K*" reports the number of iterations.

As can be seen in Table 2, among four distributed methods, D_ADMM has the best performance in our simulations. D_ADMM consumes the second less average CPU times and achieves the best objective values (there are sed on Alternating Direction Multiplier Method
of CPLEX to solve the 15 cases of Table 1. The simulation
results are shown in Table 2. In this table, column "time"
represents the execution time; " $e_X < 0$ " represents relat consumes the third less average CPU times and achieves the second better objective values (there are 4 cases' of CPLEX to solve the 15 cases of Table 1. The simulation
results are shown in Table 2. In this table, column "time"
represents the execution time; " $e_X < 0$ " represents relative
error of objective value for method " X "; average times but achieves the significant worse objective values. As for ADM-G, this method uses the most times, but achieves the worst objective values. In Algorithm 3, we see that the algorithm have a correction step at each iteration. Through the correction, we ensure that the value of the variables change within a certain range after each iteration, so we can see that the objective function value of the ADM_G changes as shown in Fig. 1 and Fig. 6. These cause the convergence of the algorithm to slow down. In Cro Units and admitted state of the store of the score of the score of the second better objective values (there are 4 cases['] ${}^eP_{L}$ *ADMM* ["] that are 0.01%). E_ADMM consumes the third less average CPU times and ac 6 using the ADMM algorithm, which indicates that the relative error between the result of the ADMM algorithm and the solver of Cplex is larger. The main reason for this methods, D_ADMM has the best performance in our
simulations. D_ADMM consumes the second less average
CPU times and achieves the best objective values (there are
only 3 cases' ${}^* \epsilon_{D_ADMM}$ " that are 0.01%). E_ADMM
consume similar for the 6th test case and the other test cases, but ADMM meets the requirement of accuracy with notable fewer iterations while solving the 6th test case.

5.4 Parallel performance comparison

As we discussed in Section 4 when we describing these distributed algorithms, ADMM, D_ADMM, and E_ADMM can be carried out in parallel. And all those steps which can be executed parallel have been marked in the descriptions

Table 2. Comparison of four fully distributed methods in performances

No.	Cplex		ADMM				D ADMM				ADM G				E ADMM		
	$f(P)(\$)$	$f(P)(\$)$	time(s)	e_{ADMM}	K	$f(P)(\$)$	time(s)	e_D admm	K	$f(P)(\$)$	time(s)	e_{ADM} G	K	$f(P)(\$)$		time(s) eE ADMM K	
	3918748	3919626	2.78	0.02%	38	3918749	3.34	0.00%	49	3920229	4.52	0.04%	-64	3918742	3.95	0.00% 58	
2	4930182	4932364	3.74	0.04%	40	4930257	4.10	0.00%	48	4934050	6.79	0.08%	75	4930238	5.24	0.00% 59	
\mathcal{F}	5333629	5336358	3.95	0.05%	35	5334089	4.82	0.01%	46	5344252	9.52	0.20%	-86	5333887	6.45	0.00% 60	
	5043949	5042868	3.59	$-0.02%$	34	5044086	5.25	0.00%	52	5046557	9.42	0.05%	-81	5043929	6.58	0.00% 64	
\mathcal{D}	5618227	5621163	4.14	0.05%	34	5618476	6.06	0.00%	51	5630697	10.76	0.22%	87	5618520	7.32	0.01% 62	
6	4692242	4706561	1.89	0.31%	16	4692315	8.99	0.00%	81	4700552	11.06	0.18%	92	4692160	9.57	0.00% 83	
	6094415	6096244	4.47	0.03%	36	6094509	5.88	0.00%	49	6101022	11.55	0.11%	88	6094494	7.43	0.00% 62	
8	5487161	5488912	3.85	0.03%	31	5487336	7.60	0.00%	63	5498977	11.81	0.22%	94	5487368	9.13	0.00% 76	
9	5882288	5884095	3.90	0.03%	30	5882110	7.09	0.00%	58	5899410	11.64	0.29%	-89	5882243	7.85	0.00% 63	
10	5398276	5399975	4.10	0.03%	32	5398578	6.77	0.01%	55	5413215	12.94	0.28%	98	5398535	8.41	0.00% 67	
	11 16550901	16558734	12.36	0.05%	37	16551041	15.54	0.00%	50	16564824	67.05	0.08%	199	16551194	21.38	0.00% 55	
	12 18062626	18068295	12.79	0.03%	33	18063188	18.09	0.00%	50	18098600	92.92	0.20%	232	18065206	20.12	0.01% 40	
	13 17792083	17799154	13.34	0.04%	34	17792777	20.78	0.00%	57	17828363	93.06	0.20%	232	17793204	29.74	0.01% 63	
	14 20809395	20820371	15.61	0.05%	36	20810941	18.59	0.01%	46	20860861	105.66	0.25%	237	20810925	29.26	0.01% 55	
	15 18225919	18231444	13.16	0.03%	32	18226198	25.97	0.00%	-67	18269991	102.38	0.24%	234	18225882	36.10	0.00% 73	

No.	worker 1	2 workers		4 workers			
	time(s)	time(s)	$p_{\rm spu}$	time(s)	$p_{\rm spu}$		
1	2.78	2.29	1.21	1.99	1.4		
$\overline{2}$	3.74	2.76	1.36	2.4	1.56		
3	3.95	2.83	1.4	2.48	1.59		
$\overline{4}$	3.59	2.69	1.33	2.3	1.56		
5	4.14	2.87	1.44	2.46	1.68		
6	1.89	1.35	1.4	1.15	1.64		
7	4.47	3.11	1.44	2.56	1.75		
8	3.85	2.69	1.43	2.24	1.72		
9	3.9	2.7	1.44	2.23	1.75		
10	4.1	2.95	1.39	2.42	1.69		
11	12.36	8.3	1.49	6.09	2.03		
12	12.79	8.52	1.5	6.52	1.96		
13	13.34	8.81	1.51	6.53	2.04		
14	15.61	10.24	1.52	7.52	2.08		
15	13.16	8.8	1.5	6.51	2.02		

Table 3. Results of parallel ADMM in a local computer

Table 4. Results of parallel D_ADMM in a local computer

No.	1 works	2works		4works				
	time(s)	time(s)	$p_{\rm spu}$	time(s)	$p_{\rm{spu}}$			
1	3.34	2.79	1.2	2.39	1.4			
$\overline{2}$	4.1	3.06	1.34	2.6	1.58			
3	4.82	3.47	1.39	2.81	1.72			
$\overline{4}$	5.25	3.96	1.33	3.21	1.64			
5	6.06	4.19	1.45	3.31	1.83			
6	8.99	6.46	1.39	5.24	1.72			
7	5.88	4.01	1.47	3.19	1.84			
8	7.6	5.09	1.49	4.17	1.82			
9	7.09	4.8	1.48	3.88	1.83			
10	6.77	4.57	1.48	3.76	1.8			
11	15.54	8.96	1.73	6.73	2.31			
12	18.09	10.05	1.8	7.7	2.35			
13	20.78	11.49	1.81	8.71	2.39			
14	18.59	10.26	1.81	7.76	2.4			
15	25.97	14.66	1.77	10.83	2.4			

Table 5. Results of parallel E_ADMM in a local computer

First, we implement the parallel versions for these three algorithms in Matlab with Parallel Computing Toolbox

Table 6. Results of parallel ADMM in distributed computers

Table 7. Results of parallel D_ADMM in distributed computers

No.	2	workers	workers 4				
	time(s)	p_{spu}	time(s)	p_{spu}			
1	4.92	0.68	3.46	0.97			
$\overline{2}$	5.53	0.84	3.78	1.23			
3	6.59	0.77	4.21	1.2			
$\overline{4}$	7.69	0.74	5.45	1.04			
5	7.91	0.78	5.85	1.06			
6	12.8	0.74	9.01	1.05			
7	7.74	0.78	5.7	1.06			
8	10.2	0.77	7.34	1.07			
9	9.59	0.83	7.12	1.12			
10	9.1	0.77	6.79	1.04			
11	15.9	1.02	11.4	1.43			
12	18.4	1.08	13.5	1.47			
13	21.1	1.03	15.5	1.41			
14	18	1.08	12.9	1.5			
15	26.1	1.05	18.9	1.45			

Table 8. Results of parallel E_ADMM in distributed computers

(PCT), and test ADMM, D_ADMM, and E_ADMM with 1 worker, 2 workers, and 4 workers in a local computer, respectively. And the results are listed in Table 3, 4, and 5. In these tables, column "time" represents the execution time; column " p_{spu} " represents the parallel speedup.

It can be seen that, with the increase of case scale, the [3] speedups of all algorithms gradually increase, and the speedups with 4 workers are greater than the ones with 2 workers. At the same time, we can see that the D_ADMM has better parallel performance than the other two algorithms when 2 workers and 4 workers are used.

Next, we use Matlab Distributed Computing Engine (MDCE) services to achieve distributed cluster computing functions. Here, we connect two computers with the same configuration, by turning on one worker and two workers on each computer, respectively, to achieve fully distributed parallel computing. The statistics of three algorithms are listed in Table 6, Table 7, and Table 8.

By comparing with Tables 3, 4 and 5, it is found that using this distributed parallel configuration method will take more time, which is related to computer communication, [7] network delay and so on, resulting in decreased performance.

6. Conclusion

In this paper, we solve the ED problem in four fully distributed manners based on ADMM. The separable objective function and physical constraints for each unit can be decoupled in four fully distributed methods. Except
ADM G the other three fully distributed methods all can [9] ADM G, the other three fully distributed methods all can be carried out in master-slave distributed and parallel schema and can protect privacy for independent power producer. Simulation results show that the D_ADMM and the E_ADMM can obtain high-quality solutions in reasonable times, and the D_ADMM has the better parallel performance, the ADMM can meet the tolerance in the shortest possible times, which suit for solving large-scale ED problems.

Acknowledgements

This work was supported by the Natural Science Foundation of China(51767003, 51407037, 61661004, 61762007, and 61462006), the Guangxi Natural Science Foundation(2016GXNSFDA380019, 2014GXNSFBA1182 74, and 2014GXNSFAA118391), Guangxi Key Laboratory of Power System Optimization and Energy Technology Foundation (15-A-01-11).

References

[1] B.H. Chowdhury and S. Rahman, "A review of recent advances in economic dispatch," *IEEE Trans. Power Systems*, vol. 5, no. 1, pp. 1248-1259, Nov. 1990.

- G.F. Reid and L. Hasdorff, "Economic Dispatch Using Quadratic Programming," *IEEE Trans. PAS*, vol. PAS-92, no. 6, pp. 2015-2023, Nov. 1973.
- C.B. Somuah and N. Khunaizi, "Application of linear programming redispatch technique to dynamic generation allocation," *IEEE Trans. Power Systems,* vol. 5, no. 1, pp. 20-26, Feb. 1990.
- [4] J.G. Waight, F. Albuyeh, and A. Bose, "Scheduling of Generation and Reserve Margin Using Dynamic and Linear Programming," *IEEE Trans. PAS*, vol. PAS- 100, no. 5, pp. 2226-2230, May 1981.
- [5] C. Wang and S. M. Shahidehpour, "Optimal gener ation scheduling with ramping costs," *IEEE Trans. Power Systems,* vol. 10, no.1, pp. 60-67, Feb. 1995.
- [6] T. Guo, M.I. Henwood, and M.V. Ooijen, "An algorithm for combined heat and power economic dispatch," *IEEE Trans. Power Systems*, vol. 11, no. 4, pp. 1778-1784, Nov. 1996.
- R.R. Shoults, S.K. Chang, S. Helmick, and W.M. Grady, "A Practical Approach to Unit Commitment, Economic Dispatch, and Savings Allocation for Multiple-Area Pool Operation with Import/Export Constraints," *IEEE Trans. PAS*, vol. PAS-99, no. 2, pp. 625-635, Mar. 1980. [8] L. Imen, B. Mouhamed, and L. Djamel, "Economic
- dispatch using classical methods and neural net works," *in Proc. of International Conference on Electrical and Electronics Engineering*, Turkey, Nov. 2013.
- [9] A. Mohammadi, M.H. Varahram, and I. Kheirizad, "Online Solving of Economic Dispatch Problem Using Neural Network Approach and Comparing it with Classical Method," *in Proc. of International Conference on Emerging Technologies*, Pakistan, Nov. 2006.
- [10] G.B. Sheble and K. Brittig, "Refined genetic algorithm-economic dispatch example," *IEEE Trans. Power Systems*, vol. 10, no. 1, pp. 117-124, Feb. 1995.
- [11] Z. L. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," *IEEE Trans. Power Systems*, vol. 18, no. 3, pp. 1187-1195, Jul. 2003.
- [12] T. Sen and H.D. Mathur, "A new approach to solve Economic Dispatch problem using a Hybrid ACO– ABC–HS optimization algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 78, pp. 735-744, Jun. 2016.
- [13] A.Y. Saber, "Economic dispatch using particle swarm optimization with bacterial foraging effect," *Inter national Journal of Electrical Power & Energy Systems*, vol. 34, no. 1, pp. 38-46, Jan. 2012.
- [14] C.C. Fung, S.Y. Chow, and K.P. Wong, "Solving the economic dispatch problem with an integrated parallel genetic algorithm," *in Proc. of International Conference on Power System Technology*, Australia, Dec. 2000.
- [15] P. Subbaraj, R. Rengaraj, S. Salivahanan, and T.R. Senthilkumar, "Parallel particle swarm optimization with modified stochastic acceleration factors for solving large scale economic dispatch problem," *International Journal of Electrical Power & Energy Systems*, vol. 32, no. 9, pp. 1014-1023, Nov. 2010.
- [16] F. Guo, C. Wen, and L. Xing, "A distributed algorithm for economic dispatch in a large-scale power system," *in Proc. of International Conference on Control, Automation, Robotics and Vision*, Nov. 2016. [17] H. Yang, D. Yi, J. Zhao, and Z. Dong, "Distributed
- optimal dispatch of virtual power plant via limited communication," *IEEE Trans. Power Systems*, vol. 28, no. 3, pp. 3511-3512, Mar. 2013.
- [18] G. Chen, C. Li, and Z. Dong, "Parallel and Distributed Computation for Dynamical Economic Dispatch," *IEEE Trans. Smart Grid*, vol. 8, no. 2, pp. 1026-1027, Mar. 2017.
- [19] 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, no. 1, pp. 1-122, 2010.
- [20] M.J. Feizollahi, M. Costley, S. Ahmed, and S. Grijalva, "Large-scale decentralized unit commit ment," *International Journal of Electrical Power & Energy Systems*, vol. 73, pp. 97-106, Dec. 2015.
- [21] 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 Trans. on Control of Network Systems*, vol. 2, no. 3, pp. 238-257, Sept. 2015. [22] M. Fukushima, "Application of the alternating
- direction method of multipliers to separable convex programming," *Computational Optimization and Applications*, vol. 1, no. 1, pp. 93-111, Oct. 1992.
- [23] C.H. Chen, B.S. He, Y.Y. Ye, and X.M. Yuan, "The direct extension of ADMM for multiblock convex minimization problems is not necessarily convergent," *Math. Program.*, vol. 155, no. 1-2, pp. 57-79, Jan. 2016.
- [24] Y.G. Peng, A. Ganesh, J. Wright, W.L. Xu, and Y. Ma, "Robust alignment by sparse and low-rank decom position for linearly correlated images," *IEEE Trans. PAMI*, vol. 34, no. 11, pp. 2233-2246, Nov. 2012.
- [25] M. Tao and X.M. Yuan, "Recovering low-rank and sparse components of matrices from incomplete and noisy observations, *SIAM J. Optim*., vol. 21, no. 1, pp. 57-81, Jan. 2011.
- [26] B.S. He, M. Tao, and X.M. Yuan. "Alternating direction method with Gaussian Back substitution for separable convex programming," *SIAM J. Optim.*, vol. 22, no. 2, pp. 313-340, Apr. 2012.
- [27] J.H Mathews and K.K. Fink, *Numerical Methods Using Matlab(4th Edition)*, Prentice-Hall Inc., 2004. [28] J. Ostrowski, M.F. Anjos, and A. Vannelli, "Tight
-

mixed integer linear programming formulations for the unit commitment problem," *IEEE Trans. Power Systems*, vol. 27, no. 1, pp. 39-46, Feb. 2012.

- [29] L.F. Yang, C. Zhang, J.B. Jian, K. Meng, Y. Xu, and Z.Y. Dong, "A novel projected two-binary-variable formulation for unit commitment in power systems. *Applied Energy*, vol. 187, pp. 732-745, Feb. 2017.
- [30] M. Nick, R. Cherkaoui, and M. Paolone, "Optimal siting and sizing of distributed energy storage systems via alternating direction method of multipliers," *International Journal of Electrical Power & Energy Systems*, vol. 72, pp. 33-39, Nov. 2015.

Linfeng Yang received the M.S. degree from Shanghai University, Shanghai, China, in 2005, and the Ph.D. degree in electrical engineering from Guangxi University, Nanning, China, in 2012. Now he is an associate professor in Guangxi University. His research interests include parallel and

distributed computing and optimization in power systems.

Tingting Zhang received the M.S. degree from Guangxi University in 2017. Her research interest is distri buted computing.

Guo Chen received the Ph.D. degree in electrical engineering from the University of Queensland, Brisbane, Australia, in 2010. Now he is an Australian Research Council DECRA fellow and lecturer at School of Electrical Engineering and Telecom munications, the University of New

South Wales. His research interests include optimization and control, complex network, dynamical systems, intelligent algorithms, and their applications in smart grid.

Zhenrong Zhang received the Ph.D degree in Nanyang Technological Uni versity, Singapore, in 2006. Now he is a professor in Guangxi University. His research interests include optimization of network and energy internet.

Jiangyao Luo is a postgraduate of Guangxi University. And his research interest is distributed computing.

Shanshan Pan receive the B.S. degree from Guangxi University in 2012. Now she is pursuing her Ph.D degree in Guangxi University. Her research interest is economic dispatch of power systems.