A Novel Distributed Particle Swarm Optimization Algorithm for the Optimal Power Flow Problem

Nicolo Gionfra, Guillaume Sandou, Houria Siguerdidjane, Philippe Loevenbruck, Damien Faille

A Novel Distributed Particle Swarm Optimization Algorithm for the Optimal Power Flow Problem

N. Gionfra[1], G. Sandou[1], H. Siguerdidjane[1], P. Loevenbruck[2], and D. Faille[3]

***Abstract*— The distributed optimal power flow problem is addressed. No assumptions on the problem cost function, and network topology are needed to solve the optimization problem. A distributed particle swarm optimization algorithm is proposed, based on Deb's rule to handle hard constraints. Moreover, the approach enables to treat a class of distributed optimization problems in which the agents share a common optimization variable. Under mild communication assumptions, agents are only required to know local variables, cost function, and constraints to solve a common optimization problem. A simulation example is provided, based on a 5-bus electric grid.**

I. Introduction

Optimal power flow (OPF) is a well-known engineering problem that has a central role in the power dispatching and planning of the electric grid. Since its first formulation [1], the problem has been widely studied and a great number of centralized algorithms have been proposed to solve it (see for instance [2] and references therein). Such research effort is motivated by the complexity of the problem itself, which happens to be highly nonconvex, mainly due to the power flow equations. Nonetheless, nowadays OPF is still object of great attention. Distributed generation, free electricity market as well as the increasing penetration of renewable energy sources in the grid require the OPF solutions to adapt to a definitely complexified electric network. In particular, the requirements for scalability and efficiency make centralized solutions no longer tractable and justify the research for distributed ones [3].

In this regard, numerous solutions are available in the literature. Generally speaking, the latter usually split the problem in two steps. Firstly, it is reduced to a convex one, either by approximation or exact relaxation. Secondly, the obtained optimization problem is distributed among the agents, i.e. the buses of the grid. This step usually capitalizes on alternating direction method of multipliers because it proved good convergence and scalability properties (e.g. [4], [5], [6]). Moreover, typically, the optimization problem is formulated in a distributed way via the introduction of an additional constraint called *consistency* or *consensus* constraint, which is responsible for the consistency, or agreement, of the variables which are shared among the agents of the network (e.g. [5], [6], [3]). Considerable attention has been devoted to exact convex relaxation methods (e.g. [5], [4]) because they enable to find the OPF global optimum in polynomial time. Unfortunately they are restricted to special network classes, such as radial ones. Other approaches (e.g. [7], [6]), even if removing the assumptions on the network topology, are based on some convex approximations of the original OPF. In [6], for instance, the authors rely on convexification of nonconvex constraints. Thus, neither optimality of the solution nor feasibility of the original problem are guaranteed. In [3] a consensus based distributed OPF is proposed by employing available convex optimization tools. In addition the consensus constraint is taken into account via the method of the penalty function. Thus, it inherits the problem of distortion of the original objective function [8].

In this paper, as in [6], we aim at scalability of the solution and refer to an OPF problem formulation which is as general as possible. To accomplish such goals we capitalize on particle swarm optimization (PSO) algorithm as it showed good performance for the *centralized* OPF problem (e.g. [9], [10], [11]). In particular we suggest a new *distributed* version of the aforementioned algorithm, leading mainly to a two-fold contribution. On the one hand, to the authors' knowledge, PSO was never applied to solve OPF in a distributed manner. On the other hand, the proposed algorithm has a general formulation that makes it applicable to a class of distributed optimization problems for which such solution is not available yet. For instance, related works to distributed PSO (DPSO) can be found in [12], and [13]. Unfortunately these approaches cannot be applied to OPF because they solve a problem in which the agents do not share common variables. Approaches in [14] and, [15] are based on considering each agent as a PSO particle of a common optimization problem. On the one hand this in turns implies that each agent has knowledge of the common cost function, which is typically not the case in optimization problems such as the OPF one. On the other hand, each agent decides on the whole optimization variable, which could be unnecessary, and computationally demanding for the problem addressed in our work. Eventually, while [12] makes use of a perturbed primal-dual method, and [13] capitalizes on projection functions, we choose to handle constraints by employing *Deb's rule* [8], by following the centralized PSO proposed by [16].

[1]N. Gionfra, G. Sandou, and H. Siguerdidjane, are with Laboratoire des Signaux et Systèmes (L2S, CentraleSupélec, Université Paris-Saclay), 3 rue Joliot Curie, 91192 Gif-sur-Yvette, France. {Nicolo.Gionfra; Guillaume.Sandou; Houria.Siguerdidjane}@centralesupelec.fr

[2]P. Loevenbruck with EDF R&D, Department EFESE, 7 boulevard Gaspard Monge, 91120 Palaiseau, France. philippe.loevenbruck@edf.fr

[3]D. Faille with EDF R&D, Department STEP, 6 quai Watier, 78401 Chatou, France. damien.faille@edf.fr

The remainder of this paper is organized as follows. In Section II the optimization problem is stated in a general way. The main result is presented in Section III, where the main features of DPSO algorithm are detailed. Section IV is concerned with the self-configuration of a finite-time average consensus, which is needed in the proposed DPSO algorithm. OPF and its formulation allowing a problem solution via DPSO is addressed in Section V. We carry out a simulation example to test the effectiveness of the approach in Section VI. The paper ends with conclusions, and future perspectives in Section VII.

II. Problem Statement

Let us provide a general formulation of the optimization problem. The specific case of OPF is derived in Section V. Consider a group of N agents, each of which disposes of a private control variable $x_i \in \mathbb{R}^{n_i}$, $i = 1, \cdots, N$, private cost function, inequality and equality constraints. By *private* we mean that the latter are only known by the associated agent i. We provide the following useful definition.

Definition 1: Agent i is *physically* coupled to agent j if at least one among its private cost function, inequality and equality constraints depend on agent j private control variable. Agent j is thus said to be a *physical neighbor* of agent i.

We then consider a graph $\mathscr{G}_p = (\mathscr{V}_p, \mathscr{E}_p)$ that keeps track of the physical relations among the agents. In particular $\mathscr{V}_p = \{1, \ldots, N\}$ is the set of the agents, i.e. the nodes of the graph, and $\mathscr{E}_p \subseteq \mathscr{V}_p \times \mathscr{V}_p$ is the set of edges among them, where the edge $(i, j) \in \mathscr{E}_p$ if and only if agent j is a *physical neighbor* of agent i. Note that according to the given definition of physical neighbor, $\mathscr{G}_p$ generally defines a *digraph*. The set of physical neighbors of agent i is defined as $\mathscr{N}_i^p \triangleq \left\{ j \in \mathscr{V}_p : (i, j) \in \mathscr{E}_p \right\}$. By defining $\boldsymbol{x}_{ij} \triangleq \left\{ x_j \in \mathbb{R}^{n_j} : j \in \mathscr{N}_i^p \right\}$ as the set of physical neighbors variables of agent i, we are able to represent the generic agent i cost function, inequality, and equality constraints respectively as $f_i(x_i, \boldsymbol{x}_{ij})$, $g_i(x_i, \boldsymbol{x}_{ij}) \leq \mathbf{0}$, and $h_i(x_i, \boldsymbol{x}_{ij}) = \mathbf{0}$, where $f_i : \mathbb{R}^{n_i + \sum_{j \in \mathscr{N}_i^p} n_j} \to \mathbb{R}$, $g_i : \mathbb{R}^{n_i + \sum_{j \in \mathscr{N}_i^p} n_j} \to \mathbb{R}^{m_i}$, $h_i : \mathbb{R}^{n_i + \sum_{j \in \mathscr{N}_i^p} n_j} \to \mathbb{R}^{q_i}$, and where $\mathbf{0}$ is a vector of proper dimension with all zero entries. Note that no assumptions were made concerning these functions. We are now able to state the optimization problem as

$$\begin{aligned} \min_{\boldsymbol{x} \triangleq [x_1 \cdots x_N]^\top} F(\boldsymbol{x}) \triangleq & \min_{\{x_i, i=1,\cdots,N\}} \sum_{i=1}^{N} f_i(x_i, \boldsymbol{x}_{ij}) \\ \text{subject to} \quad & g_i(x_i, \boldsymbol{x}_{ij}) \leq \mathbf{0}, \quad i = 1, \cdots, N \\ & h_i(x_i, \boldsymbol{x}_{ij}) = \mathbf{0}, \quad i = 1, \cdots, N \end{aligned} \tag{1}$$

Thus, agents have to *cooperatively* minimize a common cost function $F : \mathbb{R}^{\sum_i^N n_i} \to \mathbb{R}$ while sharing the common optimization variable $\boldsymbol{x}$. In the next section we provide a distributed way to solve (1).

III. Distributed PSO

Before providing the DPSO algorithm we need to clarify how we intend to handle constraints in (1), and how we need to set the distributed communication among the agents.

A. Constraints Handling

As known, one of the major difficulties when employing PSO to solve a constrained optimization problem is the lack of an explicit method to direct the optimum search towards the feasible region [16]. Having made no constraints assumption we choose a *generic* method to handle them. In particular, as previously stated, we make use of Deb's rule, which belongs to the penalty function approaches but it does not require any penalty parameter. Moreover, the aforementioned rule allows avoiding any cost function *distortion* that may occur when incorporating the constraints in the problem via penalty functions. Deb's rule consists of a tournament selection in which, when comparing two solutions of (1), the following criteria is adopted [8]: (i) any feasible solution is preferred to any infeasible solution; (ii) among two feasible solutions, the one having better objective function value is preferred; (iii) among two infeasible solutions, the one having smaller constraint violation is preferred. In addition, equality constraints are handled via a transformation into inequality constraints with the introduction of a positive threshold $\boldsymbol{\varepsilon} \in \mathbb{R}^+$ such that $g_{i+N}(x_i, \boldsymbol{x}_{ij}) \triangleq |h_i(x_i, \boldsymbol{x}_{ij})| - \boldsymbol{\varepsilon} \leq \mathbf{0}$, $i = 1, \cdots, N$, where $\boldsymbol{\varepsilon}$ is a vector of proper dimension with all its entries equal to ε.

PSO is a population based algorithm in which a given number of particles in the search space can evaluate the cost function. In order to implement Deb's rule, though, we consider the following modified cost function, called *fitness* function, which particles need to evaluate to decide on their search direction [8].

$$\tilde{F}(\boldsymbol{x}) \triangleq \begin{cases} F(\boldsymbol{x}) & \text{if } g_i(\boldsymbol{x}) \leq \mathbf{0} \quad i = 1, \cdots, 2N \\ f_{max} + \sum_{i=1}^{N} \left(\sum_{k=1}^{m_i} \chi\left(g_{i,k}(\boldsymbol{x})\right) + \sum_{k=1}^{q_i} \chi\left(g_{i+N,k}(\boldsymbol{x})\right) \right) & \text{otherwise} \end{cases} \tag{2}$$

being $\chi : \mathbb{R} \to \mathbb{R}$ the function

$$\chi(y) = \begin{cases} y & \text{if } y > 0 \\ 0 & \text{otherwise} \end{cases}$$

and where $g_{i,k}(\boldsymbol{x})$, $g_{i+N,k}(\boldsymbol{x})$ are respectively the k-th component of $g_i(\boldsymbol{x})$, and $g_{i+N}(\boldsymbol{x})$. Constraint violation is evaluated via the *sum of violated constraints* appearing in the second equation of (2). f_{max} is the maximum value of F among the available feasible solutions. If some information about the common cost function is known a priori, for ease of implementation, f_{max} can be set as a sufficiently high positive value.

B. Communication Settings

Assumption 1: There exists a communication graph $\mathscr{G}_c^1 = (\mathscr{V}_c^1, \mathscr{E}_c^1)$ such that $\mathscr{G}_c^1 = \mathscr{G}_p$.

This basically means that each agent being a physical neighbor of another one can *directly* communicate to it. We now consider the *undirected* graph associated to it: $\bar{\mathscr{G}}_c^1 = (\mathscr{V}_c^1, \bar{\mathscr{E}}_c^1)$, where $\mathscr{V}_c^1 = \mathscr{V}_c^1$, and $\bar{\mathscr{E}}_c^1$ is such that if $(i,j) \in \mathscr{E}_c^1$ then $(i,j),(j,i) \in \bar{\mathscr{E}}_c^1$. $\bar{\mathscr{G}}_c^1$ is not necessarily connected. Thus, it can be divided in M connected undirected subgraphs $\bar{\mathscr{G}}_{c,m}^1$, $m = 1,\cdots,M$, such that $\cup_{m=1}^M \bar{\mathscr{G}}_{c,m}^1 = \bar{\mathscr{G}}_c^1$, and $\bar{\mathscr{G}}_{c,m}^1 \cap \bar{\mathscr{G}}_{c,r}^1 = (\emptyset,\emptyset)$, $\forall m,r = 1,\cdots,M$, $m \neq r$.

Assumption 2: There exist M undirected connected graphs $\bar{\mathscr{G}}_{c,m}^2$, $m = 1,\cdots,M$ such that $\bar{\mathscr{G}}_{c,m}^2 \cap \bar{\mathscr{G}}_{c,r}^2 = (\emptyset,\emptyset)$, $\forall m,r = 1,\cdots,M$, $m \neq r$, and $\bar{\mathscr{G}}_c^2 \triangleq \cup_{m=1}^M \bar{\mathscr{G}}_{c,m}^2 = (\bar{\mathscr{V}}_c^2, \bar{\mathscr{E}}_c^2)$ where $\bar{\mathscr{V}}_c^2 = \bar{\mathscr{V}}_c^1$.

The latter assumption simply states that the communication among the M groups of agents defined via $\bar{\mathscr{G}}_{c,m}^1$, $m = 1,\cdots,M$, are not required to communicate via $\bar{\mathscr{G}}_c^1$ itself, as long as the considered communication $\bar{\mathscr{G}}_c^2$ keeps the same M groups of agents connected within each group. A graph example is reported in Fig. 1.

Remark 1: For ease of implementation one can simply consider only *one* communication graph $\mathscr{G}_c = (\mathscr{V}_c, \mathscr{E}_c)$ such that $\mathscr{V}_c = \mathscr{V}_c^1$, $\mathscr{E}_c^1 \subseteq \mathscr{E}_c$, and there exist M subgraphs $\mathscr{G}_{c,m}$, $m = 1,\cdots,M$ such that $\cup_{m=1}^M \mathscr{G}_{c,m} = \mathscr{G}_c$, $\mathscr{G}_{c,m} \cap \mathscr{G}_{c,r} = (\emptyset,\emptyset)$, $\forall m,r = 1,\cdots,M$, $m \neq r$.

Remark 2: The considered graphs of communication keep two different groups of agents disconnected whenever there is no physical coupling among them. If the disconnected groups are M, it basically implies that (1) can be split in M separate optimization subproblems, whose solutions can be found separately with no inter-group communication.

C. Variables Settings

Each agent has N_p particles associated to its private variable, $x_{i,p} \in \mathbb{R}^{n_i}$, $p = 1,\cdots,N_p$. Moreover, to each of them we associate a variable that keeps memory of the personal best position visited in the search space, named $b_{i,p} \in \mathbb{R}^{n_i}$, $p = 1,\cdots,N_p$. Each $x_{i,p}$ (resp. $b_{i,p}$) has a *spy* copy in each agent for which it is a physical neighbor, $x_{i,p}^j$ (resp. $b_{i,p}^j$) $\in \mathbb{R}^{n_i} \forall\, j : i \in \mathscr{N}_j^p$, $p = 1,\cdots,N_p$. The role of the spy variables is to allow evaluation of the private cost functions, and constraints. In addition, each agent particle (resp. personal best) has its personal estimate of the *average* common cost function, and *average* sum of violated constraints evaluated in the optimization variable $\boldsymbol{x}_p = [x_{1,p}\cdots x_{N,p}]^\top$ (resp. $\boldsymbol{b}_p \triangleq [b_{1,p}\cdots b_{N,p}]^\top$), i.e composed by all agents particles (resp. personal bests) having the same p index. Respectively we name the former $h_{i,p}^1 \triangleq {}^1\!/_N \sum_{l=1}^N f_l(x_{l,p}, \boldsymbol{x}_{lj,p})$ $\left(\text{resp. } h_{i,p}^{b,1} \triangleq {}^1\!/_N \sum_{l=1}^N f_l(b_{l,p}, \boldsymbol{b}_{lj,p})\right)$, the latter

$$h_{i,p}^2 \triangleq \frac{1}{N}\sum_{l=1}^N \left(\frac{1}{m_l}\sum_{k=1}^{m_l} \chi\left(g_{l,k}(x_{l,p}, \boldsymbol{x}_{lj,p})\right) + \frac{1}{q_l}\sum_{k=1}^{q_l} \chi\left(g_{l+N,k}(x_{l,p}, \boldsymbol{x}_{lj,p})\right) \right)$$

resp. $h_{i,p}^{b,2} \triangleq$

$$\frac{1}{N}\sum_{l=1}^N \left(\frac{1}{m_l}\sum_{k=1}^{m_l} \chi\left(g_{l,k}(b_{l,p}, \boldsymbol{b}_{lj,p})\right) + \frac{1}{q_l}\sum_{k=1}^{q_l} \chi\left(g_{l+N,k}(b_{l,p}, \boldsymbol{b}_{lj,p})\right) \right)$$

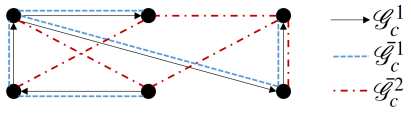

Fig. 1. Communication graphs example.

for $i = 1,\cdots,N$, $p = 1,\cdots,N_p$, and we named $\boldsymbol{x}_{lj,p} \triangleq \left\{x_{j,p}^l \in \mathbb{R}^{n_j} : j \in \mathscr{N}_l^p\right\}$ $\left(\text{resp. } \boldsymbol{b}_{lj,p} \triangleq \left\{b_{j,p}^l \in \mathbb{R}^{n_j} : j \in \mathscr{N}_l^p\right\}\right)$. We keep index i of $h_{i,p}^1$, $h_{i,p}^2$, (resp. $h_{i,p}^{b,1}$, $h_{i,p}^{b,2}$) to stress that agent i estimate of the average value can have a small difference with respect to the other agents one. Furthermore, each particle is endowed with a speed value $v_{i,p}$, and it has access to the global best position, $g_{i,p}$, among a subset of particles S_p^i, i.e. the subset of particles from which $x_{i,p}$ can retrieve information about their personal best. We additionally require such subset to be the same for each particle among the agents having same p index. Thus we can drop its index i: $S_p^i = S_p$, $i = 1,\cdots,N$. Note that S_p can be simply chosen as the set of all agent particles, so that $g_i = g_{i,p}$ would be agent i best visited position. Nonetheless it might be convenient to choose S_p as a smaller set of particles to prevent the algorithm from premature convergence.

D. PSO Update Law

PSO equations determine the particle motion in the search space in order to find the global optimum. The latter is a well-known system of stochastic difference equations. For ease of notation, at iteration k of the algorithm we note $x_{i,p} \triangleq x_{i,p}(k)$, and $x_{i,p}^+ \triangleq x_{i,p}(k+1)$. Same notations hold for $v_{i,p}$, $b_{i,p}$, and $g_{i,p}$. PSO equations, for each particle, are given by

$$\begin{cases} v_{i,p}^+ = \omega v_{i,p} + \phi_{1_{i,p}}(g_{i,p} - x_{i,p}) + \phi_{2_{i,p}}(b_{i,p} - x_{i,p}) \\ x_{i,p}^+ = x_{i,p} + v_{i,p}^+ \end{cases} \tag{3}$$

$$(b_{i,p}^+, h_{i,p}^b) = \begin{cases} (x_{i,p}^+, h_{i,p}^+) & \text{if } h_{i,p}^+ < h_{i,p}^b \\ (b_{i,p}, h_{i,p}^b) & \text{otherwise} \end{cases} \tag{4}$$

$$g_{i,p}^+ = \arg \min_{\left\{b_{i,p}^+ : p \in S_p\right\}} \{h_{i,p}^b\} \tag{5}$$

where

$$h_{i,p} \triangleq h_{i,p}(k) = \begin{cases} h_{i,p}^1 & \text{if } h_{i,p}^2 = 0 \\ f_{max} + h_{i,p}^2 & \text{otherwise} \end{cases}$$

and $h_{i,p}^+ = h_{i,p}(k+1)$. A similar definition holds for $h_{i,p}^b$, $h_{i,p}^{b,1}$, $h_{i,p}^{b,2}$. Equation (4) is the decision step of the algorithm in which the new particles are compared to the according personal best in terms of average fitness function evaluation. $\phi_{1_{i,p}} \triangleq \phi_{1_{i,p}}(k) \sim \mathrm{U}(0,c_1)$, $\phi_{2_{i,p}} \triangleq \phi_{2_{i,p}}(k) \sim \mathrm{U}(0,c_2)$, i.e. they are two aleatory variables with uniform distribution of probability in the respective intervals $[0,c_1]$, and $[0,c_2]$, where $c_1, c_2 \in \mathbb{R}^+$. Exhaustive works in the literature show how to set parameters c_1, c_2, and ω, called inertial factor, in order to ensure convergence of the algorithm, (e.g. [17]). Here, *convergence* is meant as $\forall i = 1,\cdots,N$: $\lim_{k\to\infty} \|x_{i,p}(k) -$

$x_{i,t}(k)\| = 0$, $\forall p, t = 1, \cdots, N_p$, as convergence to the global optimum is not guaranteed.

Remark 3: In this paper we additionally consider lower and upper bounds of the variables of the form: $\underline{x}_i \leq x_i \leq \bar{x}_i$, $\underline{x}_i, \bar{x}_i \in \mathbb{R}^{n_i}$, $i = 1, \cdots, N$, by limiting the maximal speed of each particle, and forcing $x_{i,p}$ to stay within the aforementioned bounds. This is simply obtained by adding the following equations to (3)

$$v^+_{i,p} = \max\{\min\{v^+_{i,p}, \bar{v}_i\}, -\bar{v}_i\} \tag{6}$$

$$x^+_{i,p} = \max\{\min\{x^+_{i,p}, \bar{x}_i\}, \underline{x}_i\} \tag{7}$$

where we place respectively (6) right after the first equation in (3), and (7) after the second equation in (3), and where $\bar{v}_i \triangleq 1/2(\bar{x}_i - \underline{x}_i)$.

E. Diversity and Dynamic Tolerance

For the case of OPF we consider two additional features of PSO algorithm that enhance the optimality of the solution. These are descripted in [16]. The reader may refer to it for further details. The first one consists in keeping *diversity* of the flock of particles, which prevents from premature convergence. This is obtained by applying two *perturbation* operators to the particle personal bests $b_{i,p}$, namely the C-Perturbation, and the M-Perturbation, which will not be described in this paper. However, in order to reduce the communication burden we consider a unique perturbation, obtained by sequential composition of the aforementioned ones, and we name it CM-Perturbation. Eventually, we employ an additional constraint handling technique, named *dynamic tolerance*. It consists in letting the threshold ε, defined in Subsection III-A, be iteration-varying, typically linearly decreasing from an initial value $\bar{\varepsilon}$ to a final target value $\underline{\varepsilon}$.

F. Distributed Algorithm

Before providing the overall algorithm we need to introduce a distributed subroutine that allows the evaluation of the average common cost function and sum of constraints, and that has to be run each time that a decision step, as in (4), has to be performed. Whenever the latter is required, we refer to Algorithm 1 to evaluate the problem functions in the generic point $\hat{\boldsymbol{x}}_p \triangleq [\hat{x}_{1,p} \cdots \hat{x}_{N,p}]^\top$, introduced to be consistent with the notation. Via the aforementioned subroutine, each of its component $\hat{x}_{i,p}$ is associated with the estimation of the average fitness function $\hat{h}_{i,p}$. It is important to notice that its value is dependent on the chosen value of equality constraints threshold ε. In general, when the chosen point to be evaluated is one of the particles we refer to the notation $h_{i,p}$ to indicate the corresponding function estimation. In all the other cases we introduce a superscript to the associated function evaluation with the same name of the considered point, for instance $b_{i,p} \leftrightarrow h^b_{i,p}$.

The overall distributed algorithm to be run by each agent on each of its particles $x_{i,p}$ is shown in Algorithm 2, where iterations are performed until a prescribed number `max_iter`, and where we introduced r, and s, respectively an aleatory variable $r \sim \mathrm{U}(0,1)$, and a linearly decreasing

Algorithm 1 Subroutine for distributed average fitness function evaluation

Input: i component of $\hat{\boldsymbol{x}}_p$
Output: average fitness function value: $\hat{h}_{i,p}$

$\hat{x}^j_{i,p} = \hat{x}_{i,p} \forall j : i \in \mathscr{N}^p_j$, via $\mathscr{G}^1_c$
wait physical neighbors to update $\hat{x}^i_{j,p}$, via $\mathscr{G}^1_c$
$\hat{h}^1_{i,p} = f_i(\hat{x}_{i,p}, \hat{\boldsymbol{x}}_{ij,p})$
$\hat{h}^2_{i,p} = \frac{1}{m_i} \sum_{k=1}^{m_i} \chi\left(g_{i,k}(\hat{x}_{i,p}, \hat{\boldsymbol{x}}_{ij,p})\right) + \frac{1}{q_i} \sum_{k=1}^{q_i} \chi\left(g_{i+N,k}(\hat{x}_{i,p}, \hat{\boldsymbol{x}}_{ij,p}\right)$
run *average consensus* on variables $\hat{h}^1_{i,p}$, $p = 1, \cdots, N_p$, and on $\hat{h}^2_{i,p}$, $p = 1, \cdots, N_p$, via $\mathscr{G}^2_c$ (see Section IV)
if $(\hat{h}^2_{i,p} = 0)$ **then**
  $\hat{h}_{i,p} = \hat{h}^1_{i,p}$
else
  $\hat{h}_{i,p} = f_{max} + \hat{h}^2_{i,p}$
end if
return $\hat{h}_{i,p}$

Algorithm 2 Distributed PSO

Output: global best among S_p: $g_{i,p}$
Initialization :
randomly initialize $x_{i,p} \in [\underline{x}_i, \bar{x}_i]$, $v_{i,p} \in [-\bar{v}_i, \bar{v}_i]$
$b_{i,p} = x_{i,p}$
set $\varepsilon = \bar{\varepsilon}$
run Algorithm 1 on $x_{i,p}$
perform (5), and set $g_{i,p} = g^+_{i,p}$
LOOP Process
for $k = 1$ to `max_iter` **do**
  perform (3)
  run Algorithm 1 on $x^+_{i,p}$
  perform (4)
  if $(r(k) < s(k))$ **then**
    $t_{i,p} =$ CM-Perturbation$(b^+_{i,p})$
    run Algorithm 1 on $t_{i,p}$
    if $(h^t_{i,p} < h^b_{i,p})$ **then**
      $\left(b^+_{i,p}, h^b_{i,p}\right) = \left(t_{i,p}, h^t_{i,p}\right)$
    end if
  end if
  set $\varepsilon = \varepsilon(k)$
  run Algorithm 1 on $b^+_{i,p}$
  if $(h^+_{i,p} < h^b_{i,p})$ **then**
    $\left(b^+_{i,p}, h^b_{i,p}\right) = \left(x^+_{i,p}, h^+_{i,p}\right)$
  end if
  perform (5)
  set $x_{i,p} = x^+_{i,p}$, $b_{i,p} = b^+_{i,p}$, and $g_{i,p} = g^+_{i,p}$
end for
return $g_{i,p}$

value from 1 to 0. In Algorithm 2, lines $10-16$ are used to keep diversity of the flock. In particular $t_{i,p}$ is a *temporary* value obtained by perturbation of the new computed personal best $b_{i,p}^{+}$. Lines $17-21$ are only necessary if ε is iteration-varying. Indeed if ε decreases at each algorithm step, then the personal bests need to be reevaluated to see how they *fit* the new, ε-dependent, fitness function. If not performed, a $b_{i,p}$ computed at the beginning of the iterations has higher probability to have better fitness function value with respect to one computed afterwards. Because of the structure of the proposed algorithm we are able to conclude the following

Proposition 1: Consider the *centralized* PSO algorithm obtained by substitution of Algorithm 1 with direct computation of $\hat{h}_{i,p}$. If it converges, so Algorithm 2 does. In addition, they converge with same performance.

Remark 4: It is important to note that thanks to Assumption 1 we are able to update the spy variables with one communication step (line 1 in Algorithm 1). Moreover we are able to avoid adding any consensus or consistency constraint to (1).

Remark 5: Proposition 1 is essentially verified thanks to the average consensus step in Algorithm 1, and to assumption 1. The latter can be relaxed by requiring that each agent that is a physical neighbor to another one has at least a communication path to it. Each of the agents belonging to this path is required to have spy variables of the mentioned physical neighbor. Then, for instance, line 1 of Algorithm 1 can be modified by considering a leader-follower consensus algorithm to allow the update of the spy variables.

IV. Finite-Time Average Consensus

The main communication burden is due to the need for performing the average consensus in Algorithm 1 to evaluate the problem functions. Unfortunately this step cannot be approximated as a one-step communication as shown in [13], for the accuracy of consensus would not be sufficient to perform decision steps on particles, as (4). Nonetheless we employ a *finite-time* average consensus algorithm to reduce the communication burden as much as possible. In addition, being interested in a whole-distributed solution, we capitalize on a distributed self-configuration of the aforementioned algorithm. Thus, in this section we aim at providing the configuration approach to tune the average consensus appearing in step 5 of Algorithm 1. The following has to be performed once, and offline.

A. Self-configuration

We consider the self-configuration problem for the generic undirected graph $\bar{\mathscr{G}}_{c,m}^{2} = (\bar{\mathscr{V}}_{c,m}^{2}, \bar{\mathscr{E}}_{c,m}^{2})$, associated to the m-subproblem of (1), defined in Subsection III-B. We rename it $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ for ease of notation, where $\mathscr{V} = 1, \cdots, N_m$. Let us state the following graph features.

- The distance between two nodes $(i,j) \in \mathscr{V}$: $dist(i,j)$ is the length of the shortest path between i, and j.
- The eccentricity of a node $i \in \mathscr{V}$: $ecc(i) \triangleq \max_{j \in \mathscr{V}} dist(i,j)$.
- The radius of $\mathscr{G}$: $r(\mathscr{G}) \triangleq \min_{i \in \mathscr{V}} ecc(i)$.
- The diameter of $\mathscr{G}$: $d(\mathscr{G}) \triangleq \max_{i \in \mathscr{V}} ecc(i)$.
- The neighborhood of agent i: $\mathscr{N}_i = \{ j \in \mathscr{V} : (i,j) \in \mathscr{E} \}$.

The state at time t associated to each agent in $\mathscr{V}$ is $h_i(t) \in \mathbb{R}$, $i = 1, \cdots, N_m$. The overall network state is $\boldsymbol{h} \triangleq [h_1 \cdots h_{N_m}] \in \mathbb{R}^{N_m}$. Each agent updates its own state according to

$$h_i(t) = w_{ii}^{t} h_i(t-1) + \sum_{j \in \mathscr{N}_i} w_{ij}^{t} h_j(t-1) \tag{8}$$

where w_{ii}^{t}, $\left\{ w_{ij}^{t} : j \in \mathscr{N}_i \right\} \in \mathbb{R}$, $i = 1, \cdots, N_m$, $\forall t > 0$, are the weights at time t to be chosen. Equation (8) in matrix form can be written as $\boldsymbol{h}(t) = W_t \boldsymbol{h}(t-1)$. It is well-known that finite-time average consensus in D steps can be solved if we are able to find D matrices W_t, $t = 1, \ldots, D$ such that the following is verified

$$\prod_{t=D}^{1} W_t = \frac{1}{N_m} \mathbf{1}\mathbf{1}^{\top} \tag{9}$$

where $\mathbf{1} \in \mathbb{R}^{N_m}$ has all its entries equal to 1. In general D is known to be a value in $[d(\mathscr{G}), 2r(\mathscr{G})]$. In this paper we refer to the work of [18], where the authors suggest a gradient back-propagation method to solve the factorization problem (9) in a distributed way. Please refer to the mentioned reference for details. This is done by solving the following optimization problem

$$\min_{\{W_1, \cdots, W_D\}} \sum_{i=1}^{N_m} \sum_{q=1}^{Q} \left(h_{i,q}(D) - y_q \right)^2 \tag{10}$$

where $h_{i,q}(D)$ is obtained via iteration of (8), and $\left\{ h_{i,q}(0), y_q \right\}$, $i = 1, \cdots, N_m$, $q = 1, \cdots, Q$ are the chosen learning sequences. Nonetheless, being the approach gradient-based, and the optimization problem (10) nonconvex, the solution is strongly dependent on the algorithm initialization of matrices W_t. We notice that (10) can be written as (1), with no constraints, by taking the agent i private cost function as $\sum_{q=1}^{Q} \left(h_{i,q}(D) - y_q \right)^2$, and common cost function as the objective function of (10). Thus, we propose to use Algorithm 2 to provide an initial point to the algorithm shown in [18], as it could help finding a better solution. In this self-configuration step, clearly, average consensus in Algorithm 1 cannot be performed via the finite-time algorithm, which is the aim of this stage. For this reason we consider the following update law for $\hat{h}_{i,p}$ variables at step 5 in Algorithm 1.

$$\hat{h}_{i,p}^{+} = \varphi_{ii} \hat{h}_{i,p} + \sum_{j \in \mathscr{N}_i} \varphi_{ij} \hat{h}_{j,p} \tag{11}$$

$$\varphi_{ij} = \begin{cases} \dfrac{1}{\max\{d_i, d_j\} + 1} & \text{if } (i,j) \in \mathscr{E} \\ 1 - \displaystyle\sum_{j \in \mathscr{N}_i} \dfrac{1}{\max\{d_i, d_j\} + 1} & \text{if } i = j \end{cases} \tag{12}$$

where (12) are Metropolis-Hastings weights, and $d_i \triangleq |\mathscr{N}_i|$ is the degree of node i in $\mathscr{G}$. Equation (11) only requires each agent to know little information about its neighbors, and it has to be run for a sufficient number of steps in order to ensure proper convergence of Algorithm 2. In addition, for the particular problem of (10), there is no need to use spy

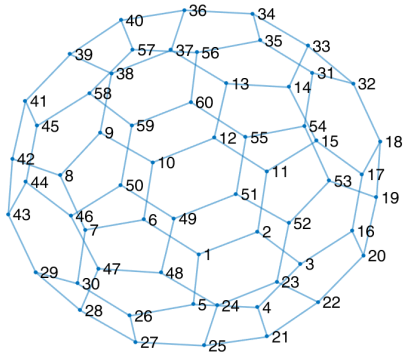

Fig. 2. Bucky communication graph.

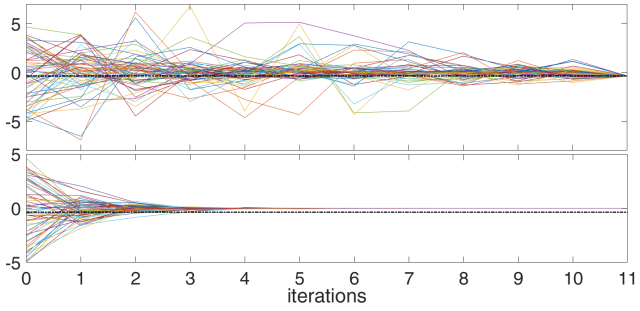

Fig. 3. Finite-time average consensus. (*up*): combined DPSO-gradient back-propagation solution; (*down*): gradient back-propagation solution.

variables neither. Agents do share common variables, but this link is kept implicit in the computation of $h_{i,q}(D)$, appearing in (10), via (8).

B. Example

We consider MATLAB®*bucky* graph in Fig. 2, to show the effectiveness of the approach. This graph has radius, and diameter $r = d = 9$, and a total number of agents equal to 60. Good results were found for a value of $D = 11$, i.e. 11 steps are needed to find the average consensus on the network with an acceptable small error. Fig. 3 shows the results of the combined DPSO and gradient back-propagation algorithm versus the solution provided by the only gradient back-propagation, where the dashed-dotted black line represents the average of the initial network state $1/N_m \sum_{i=1}^{N_m} h_i(0)$. From this example it is clear that DPSO not only can enhance the optimality of the solution of (10), but it also helps finding one when gradient back-propagation fails. Indeed, in this example, weights found via gradient back-propagation do not let convergence to the average consensus value (black dashed-dotted line) as it shows a steady state constant error. On the other hand, this is attained by employing the algorithm initialization provided by DPSO.

V. OPF Model and Problem Formulation

OPF is concerned with minimizing a given cost function while satisfying the electric grid constraints. These are mainly given by the limits on the voltage value, typically allowed in the interval $[0.95, 1.05] p.u.$, and by the power flow equations

$$\begin{aligned} P_i &= U_i \sum_{j \in \mathcal{N}_i^p} U_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j) \right) \\ Q_i &= U_i \sum_{j \in \mathcal{N}_i^p} U_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j) \right) \end{aligned} \tag{13}$$

which show how active power P, and reactive power Q at every bus i of the considered electric grid are function of its own voltage values of amplitude U_i, and phase θ_i, and of its physical neighbors voltages $(U_j, \theta_j) : j \in \mathcal{N}_i^p$. G_{ij}, B_{ij} are parameters depending on the impedance of the line between buses i, and j. If we consider (U_i, θ_i) to be the private variable of bus i, then it is easy to see that (13) are already written in the form of the constraint functions of (1). Among the possible cost functions usually employed for OPF, in this paper we choose to minimize the sum of power losses in the transmission lines

$$\min_{\{(U_i,\theta_i), i=1,\cdots,N\}} \frac{1}{2} \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i^p} \left(P_{loss}^{ij} + Q_{loss}^{ij} \right), \text{ where} \tag{14}$$

$$\begin{aligned} P_{loss}^{ij} &= g_{ij} \left(U_i^2 + U_j^2 - 2U_i U_j \cos(\theta_i - \theta_j) \right) \\ Q_{loss}^{ij} &= -b_{ij}^{sh} \left(U_i^2 + U_j^2 \right) - b_{ij} \left(U_i^2 + U_j^2 - 2U_i U_j \cos(\theta_i - \theta_j) \right) \end{aligned} \tag{15}$$

where N is the number of buses, g_{ij}, b_{ij}, b_{ij}^{sh} are other parameters depending on the impedance of the line between i, and j, and $g_{ii} = b_{ii} = b_{ii}^{sh} = 0$, $i = \cdots, N$. The reader may refer to [19] for a detailed explanation of (13),(15). In the sequel we neglect the shunt susceptance b_{ij}^{sh}, for it is usually verified that $|b_{ij}^{sh}| \ll |b_{ij}|$. Note that considering a *weighted* sum of the kind of $P_{loss}^{ij} + \alpha Q_{loss}^{ij}$, $\alpha \in \mathbb{R}^+$, would not change the solution of the optimization problem, as P_{loss}^{ij}, and Q_{loss}^{ij} are basically the same function (being g_{ij}, $-b_{ij} > 0$). For the same reason one could only consider P_{loss}^{ij}, or Q_{loss}^{ij} and obtain the same solution. We can restate (14),(15), allowing the formulation of the problem in the form of (1), by considering the following

$$\min_{\{(U_i,\theta_i), i=1,\cdots,N\}} \sum_{i=1}^{N} \left(P_{loss}^{i} + Q_{loss}^{i} \right), \text{ where} \tag{16}$$

$$\begin{aligned} P_{loss}^{i} &\triangleq \sum_{j \in \mathcal{N}_i^p} \left(g_{ij} U_i^2 - g_{ij} U_i U_j \cos(\theta_i - \theta_j) \right) \\ Q_{loss}^{i} &\triangleq \sum_{j \in \mathcal{N}_i^p} \left(-b_{ij} U_i^2 + b_{ij} U_i U_j \cos(\theta_i - \theta_j) \right) \end{aligned} \tag{17}$$

Clearly (16),(17) are equivalent to (14),(15). We are now able to formulate the chosen OPF problem. Consider an electric grid of N buses, N_G of which are generators, here considered PU buses, N_L are loads, here considered PQ buses, and one is a $U\theta$ bus, used to balance generation, load and losses. Each bus has the private cost function: $P_{loss}^{i} + Q_{loss}^{i}$. Thus, the common cost function is the objective function of (16). Moreover they all have their voltage amplitude constrained in the previously mentioned allowed interval. This kind of constraint is treated via (6),(7). The $U\theta$ bus also gives its θ as a reference to the grid, and it is simply set as 0°.

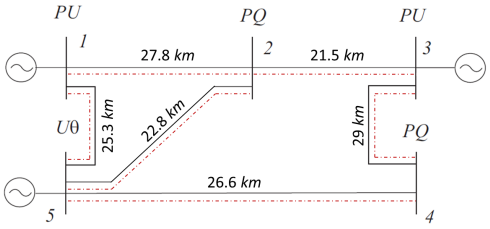

Fig. 4. 5-bus electric grid, and its communication graph $\mathscr{G}_c$.

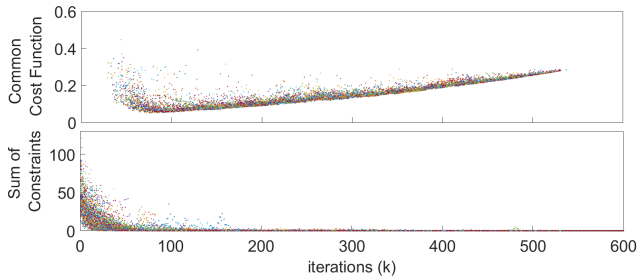

Fig. 5. Particles cost function, and sum of constraints during iterations of Algorithm 2.

Every generator bus i presents an equality constraint on P_i, set as $\bar{P}_i$, which is a problem data, via the first equation of (13), and an inequality constraint on Q_i, $Q_{i_{min}} \leq Q_i \leq Q_{i_{max}}$, employing the second equation of (13). Eventually, every load bus j presents two equality constraints given by $\bar{P}_j$, and $\bar{Q}_j$, data of the problem, via (13). Overall, each bus of the network presents two generalized nonconvex private inequality constraints. Thus the optimization problem is

$$\begin{gathered}(16),(17) \text{ subject to} \\ \text{for } i = 1, \cdots, N \\ 0.95 \leq U_i \leq 1.05 \\ \text{via (13)}: \begin{cases} P_i = \bar{P}_i,\ Q_i = \bar{Q}_i & \text{if } i \text{ is a } PQ \text{ bus} \\ P_i = \bar{P}_i,\ Q_{i_{min}} \leq Q_i \leq Q_{i_{max}} & \text{if } i \text{ is a } PU \text{ bus} \end{cases}\end{gathered}$$

VI. Simulations

We illustrate the performance of the proposed algorithm on a 5-bus grid shown in Fig. 4, which is a modified scheme of a grid example in [19]. The chosen communication graph (red dashed-dotted lines) simply traces the electric lines among the buses. The considered grid has nominal voltage equal to 63 kV, and the grid cables impedance are set to be $R = 0.15$ Ω/km, and $X = 0.21$ Ω/km. Bus 5 is the $U\theta$ bus, and it can be considered as a source substation. The generators are prescribed to produce $\bar{P}_1 = 50$ MW, $\bar{P}_3 = 70$ MW, and load buses require $\bar{P}_2 = 60$ MW, $\bar{P}_4 = 85$ MW. As far as the reactive power is concerned, load buses are operated at constant power factor, $\cos\varphi$, equal to 0.97. Generators have to keep the reactive power within given limits with respect to their active power production. Indeed the reactive power supplied above a Q/P ratio of $(\tan\varphi)_{max}$ or absorbed under $(\tan\varphi)_{min}$ is subject to the application of a charge. In France, for instance, it amounts to 16.3 €/$MVARh$ for electric grids whose nominal voltage is in $[50,130]$ kV. In particular $(\tan\varphi)_{max}$, $(\tan\varphi)_{min}$ are set respectively as 0.4, and -0.35. According to the TURPE 4[1], the cost for the active power losses is set to 53 €/MWh.

As far as the DPSO set-up is concerned, we chose $N_p = 70$ particles for each bus, and $\texttt{max\_iter} = 600$. Having normalized the power and voltage values, we considered a threshold linearly decreasing from $\bar{\varepsilon} = 1$ to $\underline{\varepsilon} = 10^{-5}$ until the 90% of the iterations. For the considered graph, finite-time average consensus algorithm to be run in Algorithm 1, only requires 2 steps to be solved. Fig. 5 shows the particles behavior during the iterations. A particle is assigned with a sum of constraints value whenever it is not feasible, and with a common cost function value otherwise. At the beginning, being randomly initialized, they are very likely to be infeasible. Nonetheless they reach feasibility quite quickly since ε is still enough large. As iterations grow, ε becomes smaller, thus, the common cost function respecting constraints is updated, and this explains its increasing pace from about iteration 100. At the very end all particles move according to their sum of constraints value since ε reaches a very small value. However, this fact does not have to be interpreted as nonconvergence. Indeed, at this point, the particles would only adjust their position in a small neighborhood of their last feasible visited position. By naming g_{best} the best among all $g_{i,p}$ found at the end of the iterations, this fact is confirmed by a maximum convergence quadratic error: $\max_{i,p} \|x_{i,p}(\texttt{max\_iter}) - g_{best}\|^2 \simeq 3 \cdot 10^{-7}$, which shows convergence of the algorithm. Optimal variables are shown in Table I. Equality constraints are presented in Table II. This shows that good performance is achieved since the ideal ratio $\bar{P}/P_{dpso}$, and $\bar{Q}/Q_{dpso}$ should be 1. The total active power losses are 5.95 MW, and the total reactive power losses are 8.34 $MVAR$. Eventually, in order to evaluate the optimality of the given solution, we compare it with the solution to the OPF problem having another common cost function. For instance, let us consider $min_{U_i} \sum_{i=1}^{N} (U_i - 1)^2$, i.e. we want to keep voltages at every bus as close as possible to 1 $p.u.$. In this case, the active power losses amount to 6.57 MW, and reactive ones to 9.19 $MVAR$, which shows the effectiveness of the optimal solution of Table I.

TABLE I
Optimal Variables

bus n°	1	2	3	4	5
U (p.u.)	1.0457	1.0090	1.0338	0.9668	1.0227
θ (°)	1.4467	-0.4259	0.7902	-2.4702	0

TABLE II
Equality Constraints

bus n°	1	2	3	4
$\bar{P}/P_{dpso}$	1.000016	1.000003	1.000021	0.999996
$\bar{Q}/Q_{dpso}$	–	1.000026	–	0.999872

[1]Tarifs d'Utilisation du Réseau Public de Distribution d'Électricité

VII. Conclusions

A novel distributed approach to solve OPF was presented. Being based on a population algorithm, no assumptions were made concerning either the cost function of the optimization problem, and the network topology. Moreover the presented algorithm enables to solve general distributed optimization problems that can be written in the form of (1). Potentiality of the algorithm was also shown when solving a distributed factorization problem to tune the finite-time average consensus algorithm.

The main drawback of the approach is due to the communication burden, in turns due to the need for performing a finite-time average consensus algorithm for each step of DPSO. This basically raises the time for convergence especially for communication graph having high diameter, and radius. We are currently investigating an algorithm modification to avoid performing the average consensus step.

Acknowledgment

This study has been carried out in the RISEGrid Institute (www.supelec.fr/342p38091/risegrid-en.html), joint program between CentraleSupélec and EDF ('Electricité de France') on smarter electric grids.

References

[1] J. Carpentier, "Contribution to the economic dispatch problem," *Bulletin de la Societe Francoise des Electriciens*, vol. 3, no. 8, pp. 431–447, 1962.

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

[3] J. Liu, M. Benosman, and A. Raghunathan, "Consensus-based distributed optimal power flow algorithm," in *Innovative Smart Grid Technologies Conference (ISGT), 2015 IEEE Power & Energy Society*. IEEE, 2015, pp. 1–5.

[4] E. Dall'Anese, H. Zhu, and G. B. Giannakis, "Distributed optimal power flow for smart microgrids," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1464–1475, 2013.

[5] Q. Peng and S. H. Low, "Distributed optimal power flow algorithm for balanced radial distribution networks," *arXiv preprint arXiv:1404.0700*, 2014.

[6] S. Magnússon, P. C. Weeraddana, and C. Fischione, "A distributed approach for the optimal power-flow problem based on admm and sequential convex approximations," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 3, pp. 238–253, 2015.

[7] A. X. Sun, D. T. Phan, and S. Ghosh, "Fully decentralized ac optimal power flow algorithms," in *Power and Energy Society General Meeting (PES), 2013 IEEE*. IEEE, 2013, pp. 1–5.

[8] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer methods in applied mechanics and engineering*, vol. 186, no. 2, pp. 311–338, 2000.

[9] M. Abido, "Optimal power flow using particle swarm optimization," *International Journal of Electrical Power & Energy Systems*, vol. 24, no. 7, pp. 563–571, 2002.

[10] P. Wannakarn, S. Khamsawang, S. Pothiya, and S. Jiriwibhakorn, "Optimal power flow problem solved by using distributed sobol particle swarm optimization," in *Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON), 2010 International Conference on*. IEEE, 2010, pp. 445–449.

[11] M. Syai'in, K. L. Lian, N.-C. Yang, and T.-H. Chen, "A distribution power flow using particle swarm optimization," in *Power and Energy Society General Meeting, 2012 IEEE*. IEEE, 2012, pp. 1–7.

[12] Y. Wakasa and S. Yamasaki, "Distributed particle swarm optimization based on primal-dual decomposition architectures," in *Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications*, vol. 2015. The ISCIE Symposium on Stochastic Systems Theory and Its Applications, 2015, pp. 97–101.

[13] Y. Wakasa and S. Nakaya, "Distributed particle swarm optimization using an average consensus algorithm," in *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*. IEEE, 2015, pp. 2661–2666.

[14] J. M. Hereford, "A distributed particle swarm optimization algorithm for swarm robotic applications," in *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*. IEEE, 2006, pp. 1678–1685.

[15] S. B. Akat and V. Gazi, "Decentralized asynchronous particle swarm optimization," in *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE*. IEEE, 2008, pp. 1–8.

[16] A. H. Aguirre, A. M. Zavala, E. V. Diharce, and S. B. Rionda, "Copso: Constrained optimization via pso algorithm," *Center for Research in Mathematics (CIMAT). Technical report No. I-07-04/22-02-2007*, 2007.

[17] G. Sandou and G. Duc, "Using particle swarm optimization for reduced order h synthesis," *IFAC Proceedings Volumes*, vol. 42, no. 2, pp. 46–51, 2009.

[18] T. M. Dung, T. Alain, and Y. Kibangou, "Distributed design of finite-time average consensus protocols," *IFAC Proceedings Volumes*, vol. 46, no. 27, pp. 227–233, 2013.

[19] G. Andersson, "Modelling and analysis of electric power systems," *ETH Zurich, september*, 2008.