Swarm and Evolutionary Computation 9 (2013) 47–57

Contents lists available at SciVerse ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular Paper

Distributed DOA estimation using clustering of sensor nodes and diffusion PSO algorithm

Trilochan Panigrahi [a,*], Ganapati Panda [b], Bernard Mulgrew [c], Babita Majhi [d]

[a] Department of ECE, National Institute of Science and Technology, Berhampur 761008, India
[b] School of Electrical Sciences, Indian Institute of Technology Bhubaneswar, India
[c] IDCOM, The University of Edinburgh, Edinburgh, UK
[d] Department of Computer Science and Information Technology, Guru Ghasidas Vishwavidyalaya (Central University), Bilaspur 495009, India

ARTICLE INFO

Article history:
Received 3 March 2012
Received in revised form
30 October 2012
Accepted 1 November 2012
Available online 24 November 2012

Keywords:
Array signal processing
DOA estimation
ML estimation
Particle swarm optimization
Diffusion algorithm

ABSTRACT

This paper proposes a distributed DOA estimation technique using clustering of sensor nodes and distributed PSO algorithm. The sensor nodes are suited by clustered to act as random arrays. Each cluster estimates the source bearing by optimizing the Maximum Likelihood (ML) function locally with cooperation of other clusters. During the estimation process each cluster shares its best information obtained by Diffusion Particle Swarm Optimization (DPSO) with other clusters so that the global estimation is achieved. The performance of the proposed technique has been evaluated through simulation study and is compared with that of obtained by the centralized and decentralized MUltiple SIgnal Classification (MUSIC) algorithms and distributed in-network algorithm. The results demonstrate improved performance of the proposed method compared to others. However, the new method exhibits slightly inferior performance compared to the centralized Particle Swarm Optimization-Maximum Likelihood (PSO-ML) algorithm. Further the proposed method offers low communication overheads compared to other methods.

1. Introduction

Accurate Direction of Arrival (DOA) estimation is an important problem for tracking and localizing of sources. In DOA estimation, the outputs from a set of sensors are analyzed to determine the bearings of signals arriving at the sensor array. The maximum likelihood (ML) method is one of the best technique used in source localization problems [1]. The ML function can be formulated from the signal–noise model equation which holds its maxima when the tried angle of arrival is exactly equal to the actual incident angle. Standard iterative approaches have been proposed to solve the resulting non-linear optimization problem. Due to the high computational load of the multivariate non-linear optimization problem required in ML estimation, several high resolution suboptimal techniques have been suggested in the literature which includes the minimum variance method of Capon [2], the MUltiple SIgnal Classification (MUSIC) method of Schmidt [3], Estimation of Signal Parameters via Rotational Invariance Techniques (ESPRIT) [4], etc. In general, the performance of suboptimal techniques is inferior to the ML technique in low signal-to-noise ratio (SNR) or for small number of data samples.

The DOA estimation is not possible by a single node using its own observed data. It needs at least one neighbor to form an array, so that any aforesaid bearing estimation algorithm can be used to estimate the DOA. The interconnection of large number of sensor nodes of a Sensor Network (SN) can constitute a random array to estimate the DOA.

The centralized method provides satisfactory estimation, because this scheme allows the most information present when making the inference. But the conventional centralized processing poses difficulty for large SN as it requires excessive communication overheads to deliver and process the data to the central processor [5]. In an energy constrained SN, the energy expenditure of transmitting all the observed data to the central processor might be too costly by making the method highly energy inefficient.

Attempts have been made to estimate the source location by measuring DOA with an antenna array at each sensor node [6] or by taking a group of sensor subarrays [7–9]. Those process is adopted for the DOA estimate from each subarray of sensors. The existing decentralized methods are fail to achieve the global performance [10]. Hence there is a need of development of distributed (not decentralized) algorithm to achieve the global performance by using less communication overheads.

Distributed estimation algorithms in WSNs have been proposed in the literature to achieve global optimum estimation by

* Corresponding author. Tel.: +91 661 2464455; fax: +91 661 2462999.
E-mail address: trilochanpanigrahi@hotmail.com (T. Panigrahi).

2210-6502/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.swevo.2012.11.001

processing the data locally at every node [11]. In one of the approaches each node in the network receives the updated estimation (instead of observed data) from its neighboring nodes, fuses it by using some rules to make it initial or previous guess and then updates it by using its own observed data [12]. This approach is referred to as the distributed in-network scheme. Based on this distributed approach, the diffusion co-operation concept is employed to estimate the global DOA [13] by solving local ML function. In that approach, it is difficult to maintain the independence of the observed data among the nodes. So it causes a problem for the formulation of global cost function as the sum of all local cost functions. The method is also influenced by the connectivity of the sensor nodes. Therefore in the present investigation clustering based diffusion co-operation is proposed for DOA estimation.

Clustering is a standard approach used in WSNs in order to achieve efficient and scalable performance [14]. In the present approach each cluster makes an arbitrary array with the nodes present inside the cluster and then estimates the DOA by optimizing a local ML function using diffusive mode of cooperation among the clusters. Recently many clustering algorithms are developed in the literature such as automatic shape independent clustering [15], clustering based on *k*-means and gravitational search algorithms [16], clustering using firefly algorithm [17], hybrid approach based on PSO, ACO and *k*-means for cluster analysis [18], etc. But in this proposed work the simple *k*-means clustering is used because the main objective is to address the use of distributed technique for DOA estimation.

Iterative search techniques have been reported in the literature for optimization of ML function such as Alternating Projection-Approximated Maximum Likelihood (AP-AML) [19], fast Expectation-Maximization (EM) and Space-Alternating Generalized Expectation-maximization (SAGE) algorithms [20] and a local search technique like Quasi–Newton methods to estimate different parameters of the source. All these techniques have several limitations because the ML cost function is multi-dimensional in nature. The PSO is a simple optimization algorithm introduced by Eberhart and Kennedy in 1995 [21,22] and has been successfully applied to estimate DOA [23–25]. It has been demonstrated that the PSO provides a competitive or even better results in a faster and cheaper ways, compared to the Genetic Algorithm (GA) [26,23]. Based on the potentiality, the PSO algorithm is selected here and the diffusive distributed version [27] is used for estimation of the DOA. In the literature [28] it has been shown that the comprehensive learning particle swarm optimization (CLPSO) is better for multimodal optimization problem. Thus the distributed version of CLPSO is used here for comparison purpose.

The proposed distributed in-cluster algorithm also reduces the computational burden on individual nodes compared to the distributed in-network algorithm [13]. Because in the proposed approach only the cluster head runs the PSO algorithm instead of running the PSO in each node. The amount of communication can reduced by using different variants of PSO such as adaptive PSO [29] and orthogonal Learning PSO [30], etc., in distributed manner. Further the communications among the cluster is reduced by using block concepts [31]. In this paper, the estimation accuracy in terms of Probability of Resolution (PR) and Root Mean Square Error (RMSE) among the distributed in-cluster, distributed in-network and centralized algorithms are compared.

The remainder of the paper is organized as follows. In Section 2, we discuss the formulation of distributed ML bearing estimation. The detail of distributed particle swarm optimization is given in Section 4. The implementation of distributed PSO for distributed DOA estimation in SN is presented in Section 5. The performance of the proposed algorithms in terms of PR and RMSE is compared with conventional centralized algorithms in Section 6. Finally, Section 7 discusses the conclusions of the paper.

2. Problem formulation

Consider a SN with N sensor nodes distributed over some geographical region to collect data for DOA estimation. These nodes receive the signal from $M(<N)$ number of narrowband far-field sources from direction angles $\boldsymbol{\theta}=[\theta_1,\theta_2,\ldots,\theta_M]^T$. Let all the sensor nodes in the network form an arbitrary array where $\mathbf{a}_g(\theta)$ as a complex $N\times 1$ response vector of the global sensor array in the direction of θ. The global $N\times M$ steering matrix for M number of sources is $\mathbf{A}_g(\boldsymbol{\theta})=[\mathbf{a}_g(\theta_1),\ldots,\mathbf{a}_g(\theta_M)]$ which depends on the position of the nodes in the network. The (n,m)th element of $\mathbf{A}_g(\boldsymbol{\theta})$ is modeled as the complex gain with respect to a reference point of the mth signal at the nth sensor is represented as

$$\mathbf{A}_g(n,m)=\exp\left\{j\frac{2\pi}{\lambda}[x_n\sin\theta_m+y_n\cos\theta_m]\right\},\quad n=1,2,\ldots,N,\quad m=1,2,\ldots,M \tag{1}$$

where λ denotes the wavelength of the signal source and (x_n,y_n) are the Cartesian co-ordinates of the nth node. Then, the complex $N\times 1$ output of sensor nodes $\mathbf{y}_g(i)$ is modeled [32] as

$$\mathbf{y}_g(i)=\mathbf{A}_g(\boldsymbol{\theta})\mathbf{s}(i)+\mathbf{v}_g(i),\quad i=1,2,\ldots,L \tag{2}$$

where $\mathbf{s}(i)$ is the complex $M\times 1$ vector of signal source which is assumed to be stochastic. The unpredicted noise vector $\mathbf{v}_g(i)$ is independent, identically complex, normally distributed with zero mean and covariance matrix $\sigma^2\mathbf{I}_N$, where σ^2 is the noise variance and L is the number of data samples (snapshots). Further, the vectors of signal and noise are assumed to be uncorrelated. The array covariance matrix is given by

$$\mathbf{R}_g=E[\mathbf{y}_g(i)\mathbf{y}_g^{\mathrm{H}}(i)]=\mathbf{A}_g\mathbf{S}\mathbf{A}_g^{\mathrm{H}}+\sigma^2\mathbf{I}_N \tag{3}$$

where $\mathbf{S}=E[\mathbf{s}(i)\mathbf{s}^{\mathrm{H}}(i)]$ is the signal covariance matrix, $[\cdot]^{\mathrm{H}}$ denotes conjugate transpose and $E[\cdot]$ stands for expectation operator.

The problem is to estimate the DOA of the sources $\boldsymbol{\theta}$ from the given observation $\{\mathbf{y}_g(i)\}_1^L$ in distributed manner. Now the question arises which of the bearing estimation algorithms can be used to make it distributed. Of course, the best choice is the ML estimator because of its superior statistical property and the distributed ML for sensor networks is discussed in [33].

2.1. Maximum likelihood estimation

Since the data received at different nodes are spatially uncorrelated, therefore the probability density function for nodes single snap shot is given as

$$p_g(\mathbf{y}_g|\boldsymbol{\theta})=\frac{1}{\det[\pi R_g(\boldsymbol{\theta})]}\exp\{-(\mathbf{y}_g^{\mathrm{H}}R_g^{-1}\mathbf{y}_g)\} \tag{4}$$

where $\mathbf{y}_g$ is an $N\times 1$ complex Gaussian random variable and $\boldsymbol{\theta}$ is an $M\times 1$ non-random unknown vector of unknown bearings to be estimated. The joint probability density function (PDF) $p_g(\mathbf{y}_g(j)|\boldsymbol{\theta})$ for L independent snapshots is defined as

$$p_g(\mathbf{y}_g|\boldsymbol{\theta})=\prod_{i=1}^{L}\frac{1}{\det[\pi R_g(\boldsymbol{\theta})]}\exp\{-(\mathbf{y}_g^{\mathrm{H}}(i)R_{\mathbf{y}_g}^{-1}\mathbf{y}_g(i))\} \tag{5}$$

For bearing estimation, the cost function is a log-likelihood of the form

$$f_g(\boldsymbol{\theta})=\ln\prod_{i=1}^{L}p_g(\mathbf{y}_g(i)|\boldsymbol{\theta})=\sum_i\ln p_g(\mathbf{y}_g(i)|\boldsymbol{\theta}) \tag{6}$$

For convenience, we suppress the dependency of f on the parameter in the notation and let $f(\mathbf{y}(i)|\boldsymbol{\theta})=f(\boldsymbol{\theta})$. The use of simple product of probabilities assumes that the snapshots are independent random events. The log-likelihood expression is convenient because it transforms the product to summation and in addition for Gaussian densities the log function inverts the exponential function.

Further, following [32] the ML estimates of DOA is obtained as

$$f_g(\boldsymbol{\theta}) = \log\left|\mathbf{P}_{\mathbf{A}_g}\hat{\mathbf{R}}_g\mathbf{P}_{\mathbf{A}_g}^{\mathrm{H}} + \frac{\mathbf{P}_{\mathbf{A}_g}^{\perp}}{N-M}\ \mathrm{tr}[\mathbf{P}_{\mathbf{A}_g}^{\perp}\hat{\mathbf{R}}_g]\right| \tag{7}$$

where $\mathrm{tr}[\cdot]$ is the trace of the bracketed matrix; $\mathbf{P}_{\mathbf{A}_g} = \mathbf{A}_g(\mathbf{A}_g^{\mathrm{H}}\mathbf{A}_g)^{-1}\mathbf{A}_g^{\mathrm{H}}$ is the projection of matrix $\mathbf{A}_g$ and $\mathbf{P}_{\mathbf{A}_g}^{\perp} = \mathbf{I} - \mathbf{P}_{\mathbf{A}_g}$ is the orthogonal complementary. The sample covariance matrix $\hat{\mathbf{R}}_g$ is defined as

$$\hat{\mathbf{R}}_g = \frac{1}{L}\sum_{i=1}^{L}\mathbf{y}_g(i)\mathbf{y}_g^{\mathrm{H}}(i) \tag{8}$$

The dependence of $\mathbf{A}_g$ on $\boldsymbol{\theta}$ is suppressed here for notational simplicity.

It is required to obtain a the optimal estimate of $\boldsymbol{\theta}$ that minimizes the global likelihood function

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} f_g(\boldsymbol{\theta}) \tag{9}$$

Note that $\mathbf{y}_g(i)$ is to be maintained constant while estimating $\boldsymbol{\theta}$.

2.2. Local bearing estimation (distributed in-network)

In order to decompose the global estimation problem (9) into a distributed optimization problem, one has to assume that the cost function has an additive structure. The additive properties state that the global cost function with all the sensor nodes data that can be expressed as the sum of individual cost functions computed at individual nodes. According to the array processing theory, a single node cannot do the bearing estimation. It may be possible to form an arbitrary array at individual nodes with their immediate neighbors. In this manner, N number of arbitrary arrays are formed at each node.

2.2.1. Local likelihood function

In the sensor network if two nodes can communicates directly with one another we say that these are connected. A node is always connected to itself. A node k connected to a set of sensor nodes (including itself) is called the neighbor nodes and is denoted as $\mathcal{N}_k$. The number of nodes in the neighbors is known as the degree of the kth node and denoted by n_k. For bearing estimation the standard data model for kth array formed at kth node is

$$\mathbf{y}_k(i) = \mathbf{A}_k(\boldsymbol{\theta})\mathbf{s}(i) + \mathbf{v}_k(i), \quad i = 1,2,\ldots,L \tag{10}$$

where $\mathbf{y}_k(i)$ denotes the data vector of sensor array at kth node; $\mathbf{A}_k(\boldsymbol{\theta})$ is the steering matrix of that particular array and $\mathbf{v}_k(i)$ is the unpredicted noise process.

The joint probability density function (PDF) $p_k(\mathbf{y}(i)|\boldsymbol{\theta})$ for L number of independent snapshots is defined as

$$p_k(\mathbf{y}_k|\boldsymbol{\theta}) = \prod_{i=1}^{L}\frac{1}{\det[\pi R_k]}\ \exp\{-(\mathbf{y}_k^{\mathrm{H}}(i)R_k^{-1}\mathbf{y}_k(i))\} \tag{11}$$

and the cost function in log-likelihood form is

$$f_k(\boldsymbol{\theta}) = \ln\prod_{i=1}^{L}p_k(\mathbf{y}_k(i)|\boldsymbol{\theta}) = \sum_i \ln p_k(\mathbf{y}_k(i)|\boldsymbol{\theta}) \tag{12}$$

where $\mathbf{y}_k(i)$ is the snapshot vector available at node k at time i and $p_k(\mathbf{y}_k(i)|\theta)$ is the likelihood function of observing data for given θ. For distributed bearing estimation an approximation is stated in (13)

$$p_g(\mathbf{y}_g(i)|\boldsymbol{\theta}) \approx \prod_{k=1}^{N}p_k(\mathbf{y}_k(i)|\boldsymbol{\theta}) \tag{13}$$

The snapshots at neighboring nodes share common data sample. Hence the use of product to describe the probability of simultaneously observing all the node snapshot vectors $\{\mathbf{y}(i)\}_k$ at time i is incorrect as the events are not independent. Taking log-likelihood, the cost function becomes

$$f_g(\boldsymbol{\theta}) \approx \sum_{k=1}^{N}\left\{\sum_{i=1}^{L}\ln p_k(\mathbf{y}(i)|\boldsymbol{\theta})\right\} \tag{14}$$

Using (12) the global cost function (14) can be written in additive form as the sum of local cost functions $f_k(\boldsymbol{\theta})$ given as

$$f_g(\boldsymbol{\theta}) \approx \sum_{k=1}^{N}f_k(\mathbf{y}_k(i)|\boldsymbol{\theta}) \tag{15}$$

The simple summation of the form in (15) is identical of the form used in distributed optimization. But the drawback here is the use of approximation to derive (15). To overcome this, in the next section a distributed in-clustering strategy is proposed.

3. Distributed in-clustering DOA estimation

In this section the distributed in-clustering DOA estimation algorithm applicable to sensor network is developed. Let us assume that all the sensor nodes in the SN are divided into N_c number of clusters. Let the set of nodes belonging to jth cluster is denoted as $\mathcal{N}_j^c$ and the number of nodes in jth cluster is represented as n_j^c. Let the average cluster size be n_c.

Now, the sensor nodes belonging to a cluster constitute an arbitrary array. The data model for jth array formed at cluster j is given as

$$\mathbf{y}_j^c(i) = \mathbf{A}_j^c(\boldsymbol{\theta})\mathbf{s}(i) + \mathbf{v}_j^c(i), \quad i = 1,2,\ldots,L \tag{16}$$

where $\mathbf{y}_j^c(i)$ is the data vector of an arbitrary sensor array formed at jth cluster; $\mathbf{A}_j^c(\boldsymbol{\theta})$ denotes the steering matrix of that array and $\mathbf{v}_j^c(i)$ represents the unpredicted noise process at the jth cluster.

The joint PDF $p_c(\mathbf{y}_c(i)|\boldsymbol{\theta})$ for L number of independent snapshots is defined as

$$p_j^c(\mathbf{y}_j^c|\boldsymbol{\theta}) = \prod_{i=1}^{L}\frac{1}{\det[\pi R_{\mathbf{y}_j^c}(\theta)]}\ \exp\{-(\mathbf{y}_j^c(i)^{\mathrm{H}}R_{\mathbf{y}_j^c}^{-1}\mathbf{y}_j^c(i))\} \tag{17}$$

and the cost function of the jth cluster in log-likelihood form is given by

$$f_j^c(\boldsymbol{\theta}) = \ln\prod_{i=1}^{L}p_j^c(\mathbf{y}_c(i)|\boldsymbol{\theta}) = \sum_i \ln p_j^c(\mathbf{y}_j^c(i)|\boldsymbol{\theta}) \tag{18}$$

where $\mathbf{y}_j^c(i)$ is the snapshot vector available at cluster j at time i and $p_j^c(\mathbf{y}_j^c(i)|\theta)$ is the likelihood function of observed data for a given θ.

Now the global joint PDF can be written in terms of individual clusters PDF for distributed bearing estimation as

$$p_g(\mathbf{y}_g(i)|\boldsymbol{\theta}) = \prod_{j=1}^{N_c}p_j^c(\mathbf{y}_j^c(i)|\boldsymbol{\theta}) \tag{19}$$

In the above equation no approximation has been made as a node shares its data to a particular to which the node belongs to. Further, a node always belongs to a specific cluster only. Therefore the data are clearly uncorrelated among clusters. Taking log-likelihood the cost function has an additive form given as

$$f_g(\boldsymbol{\theta}) = \sum_{j=1}^{N_c} \left\{ \sum_{i=1}^{L} \ln p_j^c(\mathbf{y}(i)|\boldsymbol{\theta}) \right\} \tag{20}$$

Using (18) the global cost function can be rewritten as the sum of local cost functions of the corresponding clusters as

$$f_g(\boldsymbol{\theta}) = \sum_{j=1}^{N_c} f_j^c(\mathbf{y}_j^c(i)|\boldsymbol{\theta}) \tag{21}$$

Further, following [32] the ML estimates of DOA at different cluster are

$$f_j^c(\theta) = \log\left| \mathbf{P}_{\mathbf{A}_j^c} \hat{\mathbf{R}}_j^c \mathbf{P}_{\mathbf{A}_j^c}^{\mathrm{H}} + \frac{\mathbf{P}_{\mathbf{A}_j^c}^{\perp}}{N-M} \operatorname{tr}[\mathbf{P}_{\mathbf{A}_j^c}^{\perp} \hat{\mathbf{R}}_j^c] \right| \tag{22}$$

where tr[·] is the trace of the bracketed matrix; $\mathbf{P}_{\mathbf{A}_j^c} = \mathbf{A}_j^c (\mathbf{A}_j^c \mathrm{H} \mathbf{A}_j^c)^{-1} \mathbf{A}_j^c \mathrm{H}$ is the projection of matrix $\mathbf{A}_j^c$ and $\mathbf{P}_{\mathbf{A}_j^c}^{\perp} = \mathbf{I} - \mathbf{P}_{\mathbf{A}_j^c}$ is the orthogonal complementary. The sample covariance matrix $\hat{\mathbf{R}}_j^c$ is defined as

$$\hat{\mathbf{R}}_j^c = \frac{1}{L} \sum_{i=1}^{L} \mathbf{y}_j^c(i) \mathbf{y}_j^c(i)^{\mathrm{H}} \tag{23}$$

In this case the dependence of $\mathbf{A}_j^c$ on $\boldsymbol{\theta}$ is suppressed for notational simplicity.

Now each cluster has local ML function $f_j^c(\boldsymbol{\theta})$ in (22) which is different for different clusters. It is because of the differences in the array response vector for different arrays formed at different clusters. But the difficulty in ML estimation is its multimodal nature which is mostly overcome in PSO based ML solution [23,25]. During the evolution of PSO each node can share their estimated angle of arrival to other clusters. Therefore every cluster can diffuse the neighbor's estimated DOA in order to make faster and accurate global estimation.

3.1. The clustering method

Assuming that the number of clusters to be known, the clustering problem can be summarized as the determination of a set of centers that minimizes the cost function

$$J = \sum_{j=1}^{N_c} \sum_{i \in \mathcal{N}_j^c} \|\mathbf{x}_i - \mathbf{c}_j\|^2 \tag{24}$$

where $\mathbf{x}_i$ is the co-ordinates of the ith node and $\mathbf{c}_j$ is the center associated with the cluster C_j.

One of the simplest algorithm to solve this problem is the k-means algorithm [34]. The grouping of the sensor nodes is done by minimizing the sum of squares of distances between node and the corresponding cluster centroid. k-means clustering is a single cluster membership method where each sensor node can belong to only one cluster and it also assigns every node to a cluster. Initially, k-centroids are chosen randomly from the network. Sensor nodes are then assigned to their nearest cluster, and the centroids are moved to minimise the distance between them and their assigned nodes. This process is repeated until the centroids stop moving. The procedure is summarized in Fig. 1.

However, despite its simplicity and efficacy in many applications, the performance of the k-means algorithm heavily relies on its initializations and it is not possible to guarantee that the algorithm will find the global solution of (24). The k-means algorithm can be run multiple times to reduce this effect. Since the main objective of the proposed work is to make the algorithm distributed, and hence effort has not been given for efficient clustering [15–18].

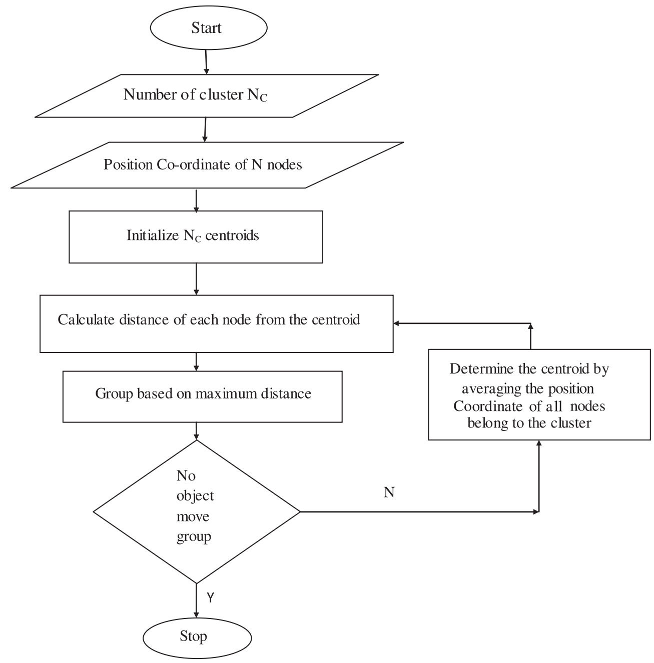

Fig. 1. Flow chart summarizing the method used by k-means clustering.

3.2. Distributed optimization

In distributed optimization methods using consensus algorithm [35], the goal of each cluster is to solve the optimization problem in a cooperative manner. Let $f_j^c(\boldsymbol{\theta})$ represents the local objective function of jth cluster which is known to that particular cluster only. Then the optimization problem is to minimize

$$f_g(\boldsymbol{\theta}) = \sum_{j=1}^{N_c} f_j^c(\boldsymbol{\theta}) \tag{25}$$

subject to $\boldsymbol{\theta} \in \mathbb{R}^M$.

Since the ML function is multimodal, its cost functions are not convex, but shares a common global solution (all provides unbiased bearing estimates). In the vicinity i.e., a small region around the solution, all the functions are convex. Therefore it is possible to initialize an optimization procedure within this region.

The effectiveness of PSO (or CLPSO) algorithm is that in a multimodal problem it can search the region where the global solution lies. Further it enables the individual clusters to interact with other clusters to make the search process faster and accurate. So it needs an algorithm that serves a basic mechanism for distributing the optimization among the clusters.

In this formulation each cluster runs the PSO (or CLPSO) algorithm for optimizing its own cost function and simultaneously shares the best solution to other clusters. In particular, each cluster starts with an initial estimate $\boldsymbol{\theta}_j(0)$ and update its estimate at discrete times $i = 1, 2, \ldots$. Let $\boldsymbol{\theta}_j(i)$ be the bearing vector estimate by jth cluster at time i. While updating sensor node combines its current estimate $\boldsymbol{\theta}_j(i)$ with the estimate $\boldsymbol{\theta}_k(i)$ received from other clusters l. Hence update equation is given as

$$\boldsymbol{\theta}_j^c(i+1) = \sum_{l=1}^{N_c} a_{jl}(i) \boldsymbol{\theta}_l^c(i) \tag{26}$$

where the scalars $a_{jl}(i)$ represents a factor by which the estimates of others clusters $\boldsymbol{\theta}_1(i), \ldots, \boldsymbol{\theta}_{N_c}(i)$ contribute to the lth cluster. These weights are assumed to be same as for static network where the topology as well as clusters remain unchanged with time. In the literature [36] different methods have been proposed to calculate these weights according to the network topology.

4. Distributed Particle Swarm Optimization (DPSO)

Consider a swarm at each cluster having P particles. The particles are represented by an M-dimensional positions (because M number of sources are considered) and associated velocity vectors. Let the position and velocity vectors of the lth particle for jth cluster are denoted as $\mathbf{x}_{lj}^i=[x_{lj1}^i,x_{lj2}^i,\dots,x_{ljM}^i]^T$ and $\mathbf{v}_{lj}^i=[v_{lj1}^i,v_{lj2}^i,\dots,v_{ljM}^i]^T$ respectively. These particles fly through the hyperspace (i.e., $\mathbb{R}^M$) and have two essential information: their own best as well as the global or their neighborhood's best position. Let the *pbest* of the lth particle at the jth cluster be $\mathbf{p}_{lj}^i=[p_{lj1}^i,p_{lj2}^i,\dots,p_{ljM}^i]$ and that of *gbest* be $\mathbf{p}_{gl}^i=[p_{gl1}^i,p_{gl2}^i,\dots,p_{glM}^i]$. In a minimization formulation each particle of the swarm at kth node updates its velocity and position according to [37]

$$\mathbf{v}_{lj}^{i+1}=\omega^i\mathbf{v}_{lj}^i+c_1\mathbf{r}_1^i\odot(\mathbf{p}_{lj}^i-\mathbf{x}_{lj}^i)+c_2\mathbf{r}_2^i\odot(\mathbf{p}_{gl}^i-\mathbf{x}_{lj}^i) \tag{27}$$

$$\mathbf{x}_{lj}^{i+1}=\mathbf{x}_{lj}^i+\mathbf{v}_{lj}^{i+1} \tag{28}$$

where $\odot$ denotes the element-wise product, $l=1,2,\dots,P$, $i=1,2,\dots$, indicates the iteration numbers and $j=1,2,\dots,n_c$ represents the cluster number. The velocity is updated based on the current value scaled by *inertial weight* ω and increased in the direction of *pbest* and *gbest*. The constants c_1 and c_2 are scaling factors that determine the relative pull of the particles' best location *pbest* and global best location *gbest*. These positive constants are sometimes referred to as *cognitive* and *social* parameters respectively [22,38]. Two independent M-dimensional random vectors r_1 and r_2 consisting of independent random numbers uniformly distributed between 0 and 1 are used to stochastically vary the relative pull of *pbest* and *gbest*. In order to simulate the unpredictable components of the natural swarm behavior, these random elements are introduced in the optimization process [38].

In a distributed DOA estimation, the problem is how do different clusters mutually share their *gbest* information with other clusters. For this purpose, diffusion based distributed PSO (DPSO) algorithm is chosen using similar cooperation strategy described in [39]. In DPSO each node updates its particle's position, velocity and *pbest* using its local objective function $f_j^c(\boldsymbol{\theta})$ and shares its *gbest* vector to all other clusters of the network.

At every cluster a consensus mechanism given in (29) is used to fuse the received *gbest* estimates

$$\mathbf{p}_{gj}^{(i-1)}=\sum_{n=1}^{N_c}a_{jn}\mathbf{p}_{gn}^{(i-1)} \tag{29}$$

The combination coefficients $\{a_{jn}\geq 0\}$ are determined from the network topology and known as metropolis weight. The metropolis weight selection procedure is given as [36]

$$a_{kl}=\begin{cases}\dfrac{1}{\max(n_k,n_l)} & \text{if } k\neq l \text{ are linked}\\ 0 & \text{for } k \text{ and } l \text{ are not linked}\\ 1-\sum\limits_{l\in\mathcal{N}_k/k}a_{kl} & \text{for } k=l\end{cases} \tag{30}$$

where n_k and n_l denote the degree for nodes k and l. But in the present case cluster of nodes are cooperated for distributed DOA estimation. Again it is also assumed that the clusters are fully connected which means all the clusters are connected to each other. Therefore the combiner co-efficients are constant and equal to

$$a_{kl}=1/N_c \tag{31}$$

The fused $\mathbf{p}_{gj}^{(i-1)}$ is used in the local optimization process to estimate the global DOA so that it can rapidly respond to change in its neighborhood. Subsequently the position and velocities of all particles at nodes are simultaneously evaluated by taking diffused *gbest* as the previous *gbest* of the node. The computed global best information is exchanged between all participating nodes for further processing. The distributed CLPSO (DCLPSO) algorithm is developed by following the way how the weighted PSO is used in distributed manner. The differences between distributed PSO and CLPSO algorithms are the velocity and weight update methods.

5. Distributed PSO for DOA estimation in sensor network

The formulation of the distributed version of PSO and CLPSO algorithms for clustering based distributed ML optimization to estimate the source DOA's in WSNs is dealt in this section. Instead of nodes, the cluster heads estimate the DOA. To achieve it every cluster head in the network begins by initializing a population of particles in the search space with random positions constrained between 0 and π in each dimension, and random velocities in the range of $0-\pi$. The M-dimensional position vector of the jth cluster of lth particle takes the form $x_{jl}=[\theta_{j1},\dots,\theta_{jM}]$, where θ_j represents the DOAs of jth cluster. A particle position vector is converted to a candidate solution vector in the problem space. The score of the mapped vector evaluated by a likelihood function $f_j^c(\boldsymbol{\theta})$ which is given in (22), represents the fitness value of the corresponding particle.

During the evolution of the algorithm in each iteration each cluster updates its velocity and position of all associated particles and then evaluates *gbest* and shares with its neighbor nodes for diffusion operation. The velocity update of (27) acts as a key to the entire optimization process. Three components typically contribute to the new velocity. The first part refers to the inertial effect of the movement which is just proportional to the old velocity and has a tendency to guide particle to proceed in the same direction. The inertial weight w is considered critical for the convergence behavior of PSO [40]. A larger w facilitates the search in new area and global exploration while a smaller w helps local exploitation in the current search area. In this study, w is selected so that it decreases during the optimization process, thus the PSO tends to have more global search ability at the beginning of the run while having more local search ability towards the end.

Let the maximum and minimum values of w be $w_{\max}$ and $w_{\min}$ respectively. The value of w in ith iteration is

$$w^i=\begin{cases}w_{\max}-\dfrac{w_{\max}-w_{\min}}{rK}(i-1) & \text{if } 1\leq i\leq[rK]\\ w_{\min} & \text{for } [rK]+1\leq i\leq K\end{cases} \tag{32}$$

where $[rK]$ is the number of iterations with time decreasing inertial weight, $0<r<1$ represents ratio, K is the maximum iteration number and $[\cdot]$ is a rounding operator.

The second and third components of the velocity update equation introduce stochastic tendencies to return towards the particle's own as well as the group's best historical positions. Since there is no mechanism for controlling the velocity of a particle, it is necessary to define a maximum velocity to avoid the swarm divergence [41]. The velocity limit which has been applied along each dimension at every node as

$$v_{lj}^i=\begin{cases}V_{\mathrm{MAX}} & \text{if } v_{lj}^i>V_{\mathrm{MAX}}\\ V_{\mathrm{MIN}} & \text{if } v_{lj}^i<V_{\mathrm{MIN}}\end{cases} \tag{33}$$

where $j = 1, \ldots, n_c$. In this work, V_{MAX} is limited to half of the dynamic range. The new particle position is calculated using (28). The iteration is terminated if the specified maximum iteration number K is reached. The final global best position p_{Kg} is taken as the ML estimates of source DOA. The main steps involved in distributed PSO–ML algorithms are outlined in Algorithm 1.

Algorithm 1. Main steps of distributed in-cluster algorithm.

Setup Problem:
- Define WSN with topology.
- Initialize the location of each node.
- Form of clusters using K-means clustering algorithm.
- Choose the cluster head.
- Calculate weighted coefficients for diffusion.
- Set up random array at each cluster by connecting to the nodes present in that cluster.
- Define problem space.
- Define local fitness function at each cluster.
- Select PSO or CLPSO Parameter.

Swarm Initialization at each cluster:
- Random Positions.
- Random velocities.

```
for each iteration do
  for each cluster do
    for each particle do
      Map particle location to solution vector in solution space;
      Evaluate the objective function of current iteration of the
      according to its local ML function (22);
      Update particles best location p_k^i and group best
      location g_k^i according to fitness value;
      Update particles velocity according to (27);
      if velocity exceeds the limits then
        Limit particles velocity using (33);
      end
      Update particle position using (28);
      if particles position out of solution boundaries then
        Clip or adjust particles position;
      end
    end
    Share its gbest to all neighbor nodes;
  end
  Perform the local diffusion according to (29);
  Check termination criterion;
end
```

5.1. Block distributed in-cluster algorithm

Each cluster runs the PSO algorithm locally and shares their best estimated value in each iteration with other clusters so that they may achieve the global performance. In this way the DPSO also converges faster than the PSO algorithm. For this purpose initially the algorithm needs more co-operation to search the region of the global optimum in the local cost function. After finding the global optimum region, the PSO then searches the actual optimum value in the vicinity of it so that the sharing of their best estimated data may not be required in each iteration. Therefore a block distributed in-cluster algorithm (Algorithm 2) is proposed where the individual cluster shares its best estimate up to K_B iterations for local diffusion. After K_B iterations, the clusters share their information in the interval of B iterations until the stopping criterion is satisfied. The main steps involved in this algorithm is presented as Algorithm 2.

Algorithm 2. Main steps of block distributed in-cluster algorithm.

Setup Problem:
As in Algorithm 1
Swarm Initialization at each cluster:
As in Algorithm 1

```
for each iteration do
  for each cluster do
    for each particle do
      These steps are same as in Algorithm 1
    end
    If iteration >= K_B & mod(iteration,B) = 0 then
      Share its gbest to all neighbor nodes;
    end
  end
  Carry out the local diffusion operation according to (29);
  Check termination criterion;
end
```

5.2. Communication overhead of the proposed algorithms

In a sensor network, the communication of information consumes most of the energy [11]. Therefore it is of great importance to minimize the communication requirements among the nodes. In the present case an upper bound of the average number of messages transmitted by a node and the overall communication for the network are determined for the proposed algorithms, and are also compared with those of existing algorithms.

A node communicates directly only with other sensor nodes that are within a small distance known as transmission radius (T_r). In order to avoid communication between sensors within other's communication range, the sensor nodes form a multihop network. In multihop communication the distance between any two nodes is defined as the minimum number of hops between them. Let h_N be the maximum distance between any node in the network and the Fusion Center (FC). Similarly let h_C be the maximum distance between any node in the cluster and the Cluster Head (CH). Let d be the average degree of a node in the network where the degree of a node is the number of nodes that serve as neighbors to it.

The proposed algorithm involves two phases: a data sharing phase and the distributed estimation phase. In fact the clustering is also a part of this algorithm, but we are not considering the amount of communication required for clustering. It is because the position based k-means clustering is used in the case where the nodes need not have to communicate anything. The clustering is carried out by the FC and the cluster information is sent to all the nodes of the network. Thus excluding the clustering phase, the overall communication overheads required are the sum of the number of messages in two aforesaid phases.

In the data sharing phase, each node shares L complex data samples to its CH and the average number of messages unicast from nodes to the CH node can be calculated. In the present case aggregation of message is not used and it provides worst case value. Under these conditions the number of messages transmitted becomes $O(dh_c^3)$ [42]. The overall communication overheads for all clusters of the network during data sharing phase are given by

$$\mathcal{M}_{sharing} = O(N_c L d h_c^3) \tag{34}$$

During the estimation phase, at each iteration every cluster in the network shares M ($M \ll L$) real data samples with all N_c clusters. Let us assume that in the worst case, CH's are present at the center. If one cluster sends data to another neighboring cluster, it needs at most $2h_c$ hops (h_c hops for the cluster sending the data and another h_c hops for the cluster sending to a node). Therefore, total number of messages transmitted by all the clusters in a cycle is $2h_cN_cM$ (assuming that cluster are communicating is $2h_c$ hops). Let I be the number of cycles required to attain the steady state value in the network. Then, the total communication cost for the estimation phase is $\mathcal{M}_{est} = O(N_cMI)$.

Therefore the overall communication overhead for the network using Algorithm 1 is $O(N_cLdh_c^3)+O(2*h_cN_cMI)$. Since the number of clusters is always lesser than the cluster size, the number of hops between clusters can be minimized and sharing the information between the clusters can further be reduced by using block concept (Algorithm 2). Again in estimation phase the number messages are also depends on the number of cycles required to get steady state value. The convergence speed of PSO can be improved if we initialize the PSO by AP-AML [23] or MUSIC algorithm. Therefore the total number of messages needed for Algorithm 1 is

$$\mathcal{M}_{alg1} = O(N_cLdh_c^3) \tag{35}$$

The number of messages needed for distributed CLPSO is same as in distributed PSO algorithm. In case of distributed in-network algorithm each node shares its information in every cycle. Hence the number of messages required for distributed in-network algorithm (Algorithm 3 developed in Ref. [13]) is given as

$$\mathcal{M}_{alg3} = O(NMId) \tag{36}$$

In centralized estimation, each node in the network sends L number of data samples to the FC. In the worst case scenario where message aggregation is not used, the message unicast from a leaf node in the spanning tree, is forwarded h_N times till it reaches the FC. Therefore, the overall communication overhead for the network in the centralized case is

$$\mathcal{M}_{cent} = O(NLdh_N^3) \tag{37}$$

In general, a more number of snapshots are required for the centralized MUSIC algorithm say $L' > L$. The communication overhead for this algorithm is computed as

$$\mathcal{M}_{MUSIC} = O(NL'dh_N^3) \tag{38}$$

Thus the centralized MUSIC algorithm needs more communication than the centralized PSO-ML algorithm. The overall comparison between all the algorithms is given in Table 1. The value of h_c is small because it may approximately equal to $\lceil h_N/N_C \rceil$. Therefore the saving in overall communication overhead is $O(NMI/N_cLh_c^3)$ over distributed in-network algorithm. Specially for large network the ratio N/N_c is large, hence the saving of communication overhead is large. The distributed in-network algorithm needs less communication overhead compared to other centralized algorithms. Therefore it may be conclude that the proposed distributed in-cluster algorithm uses less communication overhead compared to others.

6. Simulation results

In this section an illustration is given to show the advantages of distributed in-clustering algorithm over distributed in-network and also to examine the performance of the proposed algorithm against decentralized MUSIC, centralized MUSIC, PSO-ML and CLPSO-ML algorithms. The performances of these estimators are evaluated in term of two key parameters: the Root Mean Square Error (RMSE) and the Probability of Resolution (PR). The RMSE is defined as

$$\mathrm{RMSE} = \sqrt{\frac{1}{MK}\sum_{i=1}^{K}\sum_{m=1}^{M}\{\hat{\theta}_m(i)-\theta_m(i)\}^2} \tag{39}$$

where $\hat{\theta}_m(i)$ is the estimate of the mth DOA achieved in the ith run, and $\theta_m(i)$ is the true value of the mth source. The PR is the ability to resolve closely spaced sources. Two sources are said to be resolved in an experiment if both DOA estimation errors are less than the half of their angular separation. All the algorithms have been simulated using MATLAB for 100 Monte Carlo (MC) for each point of the plot on an Intel Core™ Duo, 2.8 GHz, 4 GB RAM PC. The PSO parameters are chosen for the experiments are: the constants $c_1 = c_2 = 2$, maximum number of iteration $K=100$, maximum velocity limit $V_{max} = 0.5*\pi$, the range of time varying inertial weight is $w_{max} = 0.9$, $w_{min} = 0.4$ and $r=0.5$. These values can be tuned for better optimization performance. The PSO algorithm starts with random initialization at diffusion center for centralized estimation and at cluster heads for distributed in-clustering algorithm respectively. The algorithm is stopped when maximum number of iterations is reached. The parameters for CLPSO algorithm are taken from the literature [28].

6.1. Example

Consider a SN having $N=24$ (number of identical sensor nodes) distributed randomly in an area of 20×20. It is assumed that the sensor nodes are aware of their position co-ordinates. Then the topology of the network is developed on the basis of transmission radius of the nodes which is shown in Fig. 2. Two nodes are connected if they all lie in their transmission range. Then the k-means clustering algorithm is used to make the

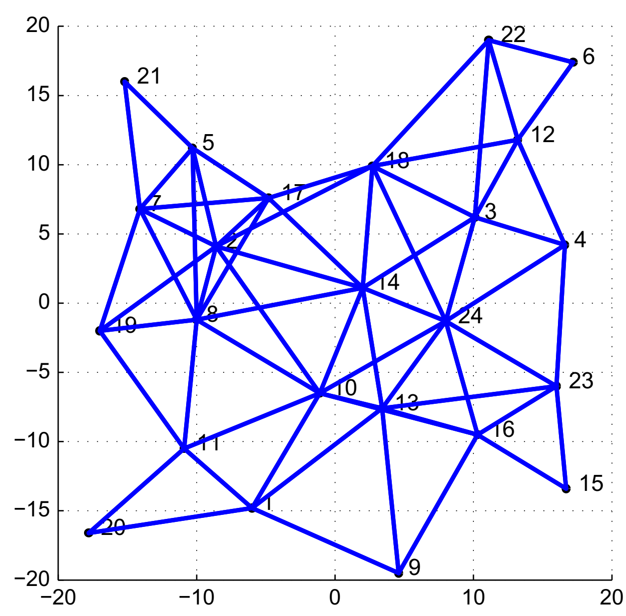

Fig. 2. Network topology used in Example 1.

Table 1
Comparison of communication overheads for different algorithms.

Algorithms	Distributed in-cluster algorithm	Distributed in-network algorithm	Centralized PSO-ML/ CLPSO-ML	Centralized MUSIC
Overall communication overhead	$O(N_cLdh_c^3)$	$O(NMId)$	$O(NLdh_N^3)$	$O(NL'dh_N^3)$

networks into three numbers of non-overlapping clusters. The clusters are shown in Fig. 3. In Fig. 3 the red, green and magenta colored nodes belongs to Cluster 1, 2 and 3 respectively. The nodes 8, 18 and 16 serve as the respective Cluster Heads (CH). The variation of cluster head and the objective function with respect to iteration are given in Figs. 4 and 5 respectively. These performances are not fixed and varies if repeat the experiment. The given performance is for the case where we get nearly equal number of nodes in each cluster.

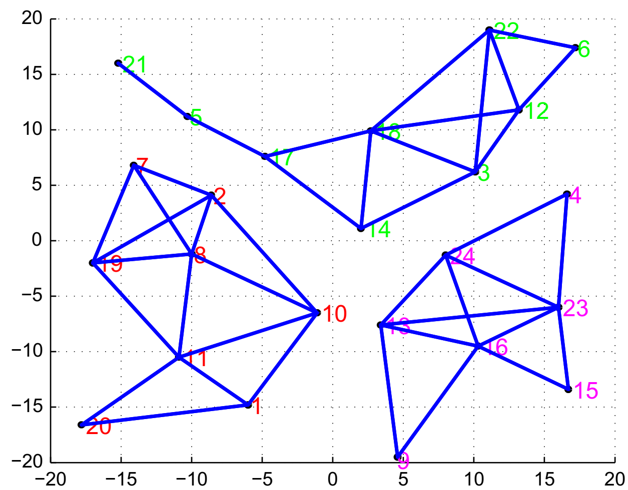

Fig. 3. Clusters used in Example 1. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

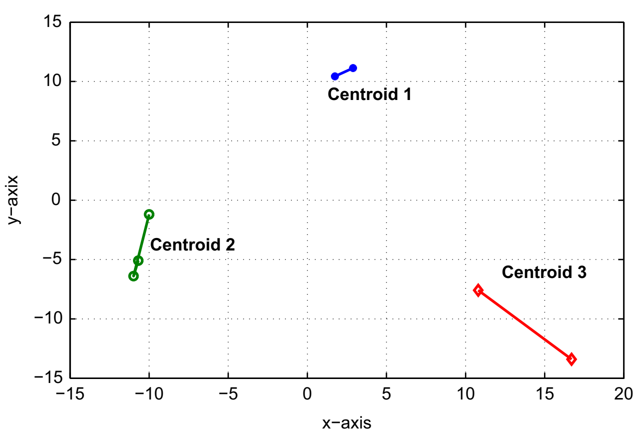

Fig. 4. Variation of cluster heads with iteration in Example 1.

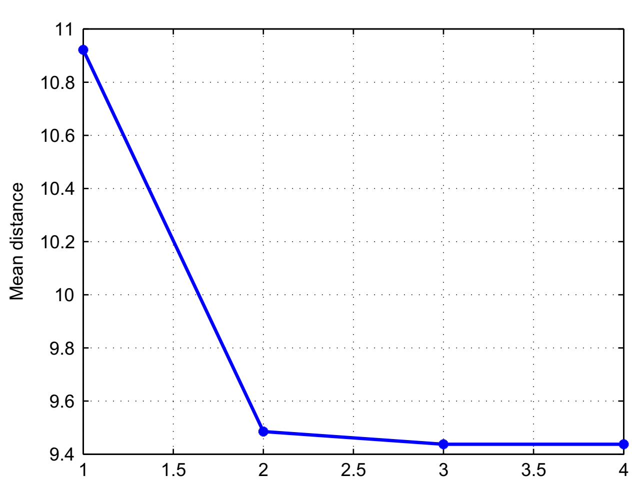

Fig. 5. Variation of objective function with iteration in Example 1.

It is assumed that two uncorrelated equal signals power impinge on the distributed network at 130° and 138°. The number of snapshots $L=20$ and 60 is taken for PSO-ML, CLPSO-ML and MUSIC algorithms respectively. Higher number of snapshots for MUSIC compared to that of ML is used because minimum number of snapshots required for MUSIC to get satisfactory performance is equal to twice the number of sensors where as the ML estimator can provide similar performance even by using fewer snapshots. The functions $f_g(\boldsymbol{\theta})$ and $f_k^c(\boldsymbol{\theta})$ are computed for global and local estimation using the PSO or CLPSO. Fig. 6 shows the estimated RMSE obtained by using distributed in-clustering algorithm using PSO (Algorithm 1A), distributed in-clustering algorithm using CLPSO (Algorithm 1B), block distributed in-clustering algorithm (Algorithm 2), distributed in-network algorithm (Algorithm 3) [13], centralized PSO-ML, CLPSO-ML and MUSIC and decentralized MUSIC. The performance of all those algorithms is compared with the corresponding Cramer-Rao Lower Bound (CRLB) which is based on stochastic signal assumption [32]. Fig. 7 depicts the probability of resolution obtained using these methods. The SNR is varied from −20 dB to 20 dB with step size of 1 dB.

It is evident from Figs. 6 and 7, that the centralized PSO-ML and CLPSO-ML provide better performance over all other algorithms by demonstrating lower RMSE and higher PR. The RMSE is asymptotically attain the CRLB at around −7 dB SNR. The accurate DOA estimation is because of use of the ML function (which is

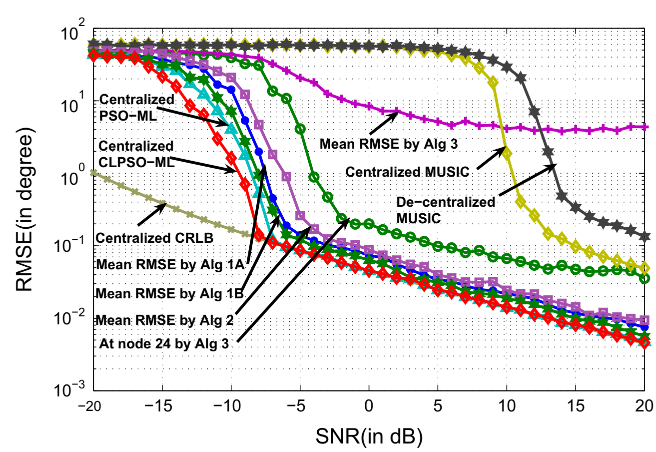

Fig. 6. RMSE vs. SNR plots in DOA estimation by centralized PSO-ML and CLPSO-ML, centralized and decentralized MUSIC, proposed distributed in-clustering Algorithms 1A, 1B and 2, distributed in-network Algorithm 3, and theoretical centralized CRLB.

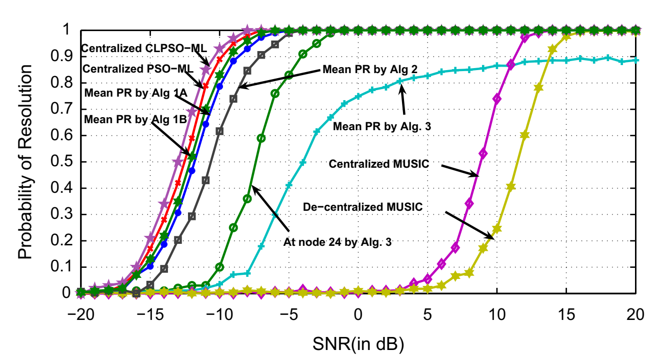

Fig. 7. PR vs. SNR plots in DOA estimation by centralized PSO-ML and CLPSO-ML, centralized and decentralized MUSIC, proposed distributed in-clustering Algorithms 1A, 1B and 2 and distributed in-network Algorithm 3.

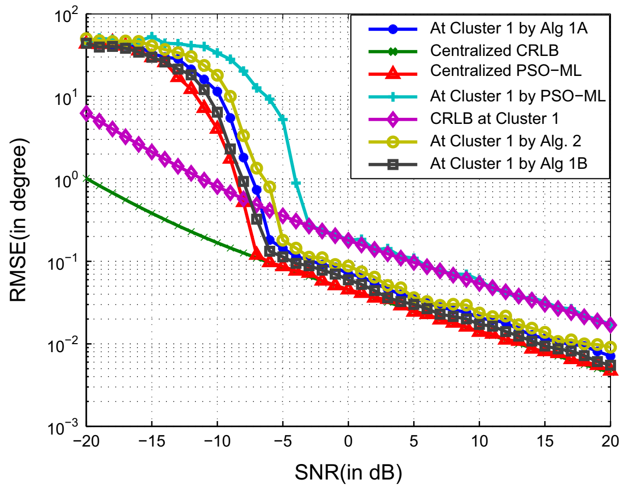

Fig. 8. RMSE vs. SNR plots in DOA estimation by centralized PSO-ML and CLPSO-ML, at Cluster 1 by using Algorithms 1A, 1B and 2, at Cluster 1 by PSO-ML, theoretical CRLB at Cluster 1 and theoretical centralized CRLB.

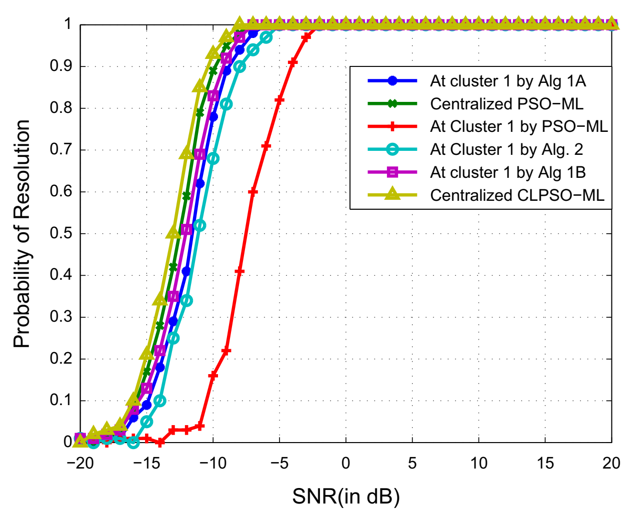

Fig. 9. PR vs. SNR plots in DOA estimation by centralized PSO-ML and CLPSO-ML, at Cluster 1 by using Algorithms 1A, 1B and 2 and at Cluster 1 by PSO-ML and CLPSO-ML.

statistically optimal) and the PSO algorithm (which is robust and reliable for global optimization). The performances of the proposed in-clustering Algorithm 1A, 1B and 2 are closer to global performance (both RMSE and PR). Among these three distributed in-clustering algorithm, the Algorithm 1B is providing better performance. The performance of Algorithm 2 is little less because the clusters are not sharing the best estimated value in each iteration. The performances are not exactly matching with that of global network's performance, but it is much better than the Algorithm 3.

The average performance of Algorithm 3 is inferior to that of Algorithm 1A, 1B and 2. It is known that the DOA estimation performance of an array decreases with the decrease in number of sensors in the network. In case of Algorithm 3, few nodes present in the edge of a network constitutes random array with their immediate neighbors having only three sensor nodes. In such situation the ML-DPSO algorithm fails to estimate two sources DOA using three nodes. On the other hand the nodes which reside in the interior of the network have better connectivity, and hence they yield better performance. The best performance obtained at node 24 (compared to all other nodes present in the network) using Algorithm 3 is given in Figs. 6 and 7. Thus it may be concluded that the proposed in-clustering algorithms are providing better performance compared to all other distributed algorithms. At lower SNRs this performance is better than the MUSIC and the decentralized MUSIC algorithm. These two algorithms show comparable performance at high SNRs.

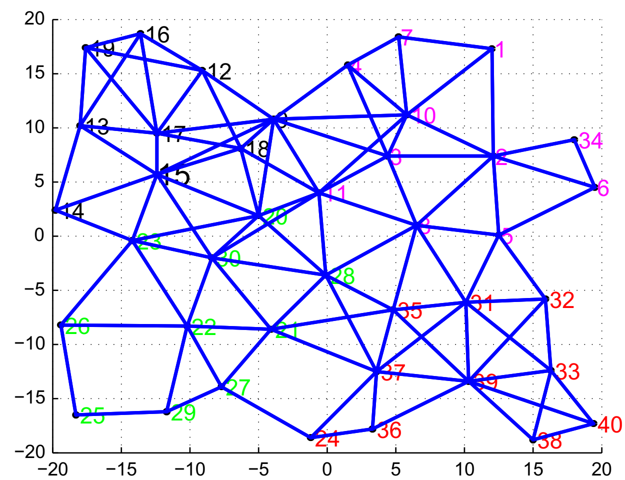

Fig. 10. Network topology and clusters used in Example 2.

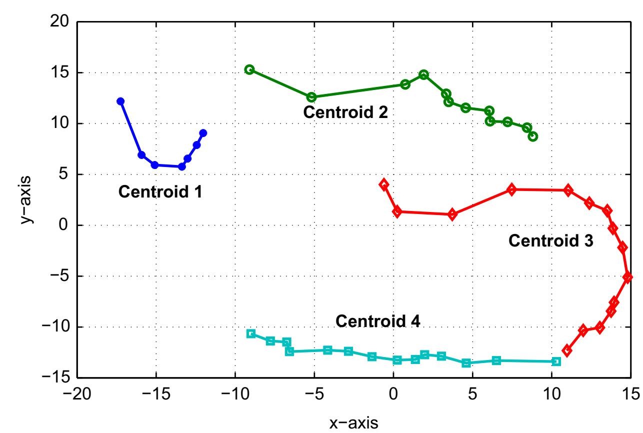

Fig. 11. Variation of cluster heads with iteration in Example 2.

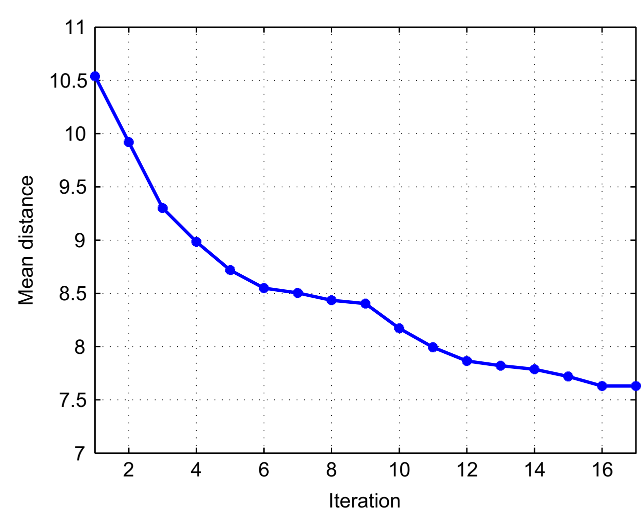

Fig. 12. Variation of objective function with iteration in Example 2.

In order to show the performance improvement of distributed in-clustering estimation over non-cooperating estimation, we have considered cluster 1 (clust1) of the network estimates the same sources DOA by using PSO-ML without any cooperation with other clusters. The performances of clust1 using PSO-ML, CLPSO-ML and Algorithm 1A, 1B and 2 are plotted in Figs. 8 and 9. Since the objective of distributed algorithm is to achieve the centralized performance, the performance of clust1 is compared with the centralized network performances. The optimum performance of an array is the CRLB. In case of no-cooperative estimation, the PSO-ML attains the CRLB asymptotically at around 3 dB. But the same network gives better performance than the optimum theoretical value, if the estimation is co-operative. It is because in distributed estimation by using consensus algorithm for a fully connected network, each agent tries to achieve the global performance. In case of Algorithm 2, the performance is slightly inferior to that of Algorithm 1A and 1B. But the block method saves communication overheads. So when energy is major constraint for designing WSNs, the Algorithm 2 is preferable with slight compromise in the performance.

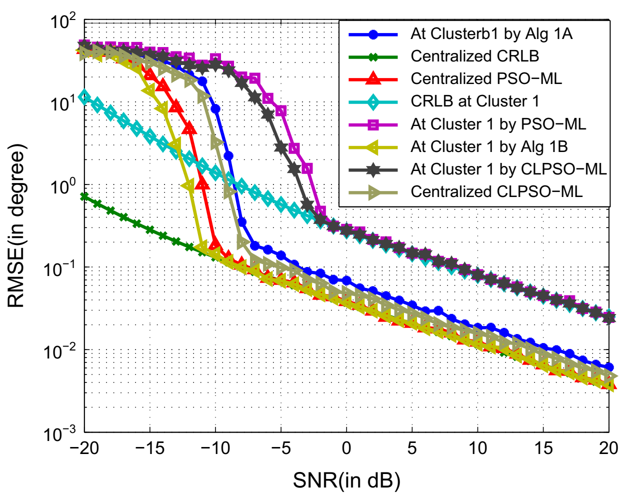

Fig. 13. RMSE vs. SNR plots in DOA estimation by PSO-ML and CLPSO-ML, distributed in-clustering Algorithms 1A and 1B, and theoretical centralized CRLB.

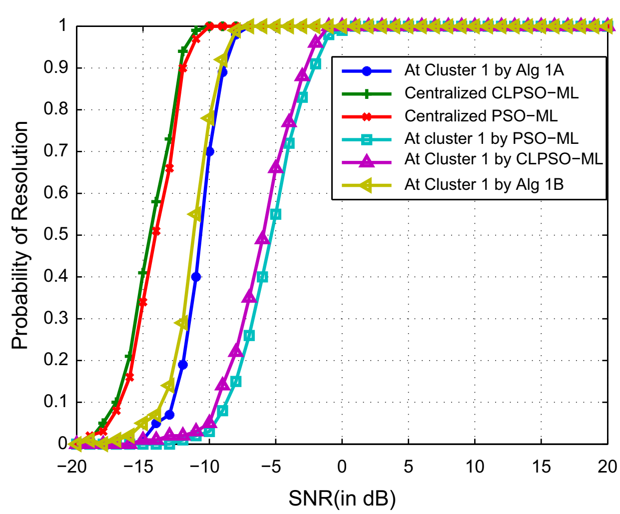

Fig. 14. PR vs. SNR plots in DOA estimation by centralized PSO-ML and CLPSO-ML, proposed distributed in-clustering Algorithms 1A and 1B and distributed in-network Algorithm 3.

6.2. Example 2

In the second example, the proposed technique is examined in the presence of two closely spaced sources by using a large sensor network of $N=40$ nodes. The topology is shown in Fig. 10 having average node degree $d=6$. This value can be reduced by minimizing communication link between nodes (i.e. by using multi hop network) because performance of the algorithm does not depend on connectivity of the nodes unlike distributed in-network algorithm. The clustering operation is carried out by unsupervised k-means algorithm. In this case the network is divided into four clusters having 11, 10, 9 and 10 cluster nodes (the average cluster size is 10) respectively. The nodes numbered in magenta, green, black and red belongs to clusters 1, 2, 3 and 4 respectively. The central nodes 2, 22, 17 and 39 are chosen CH's of clusters 1, 2, 3, and 4 respectively. The variation of cluster head and the objective function with respect to iteration are given in Figs. 11 and 12 respectively.

The distributed SN receives signals from two sources present at DOAs 80° and 82° relative to x-axis. The number of snapshots is 20. In Fig. 13, the RMSE values obtained by global network using PSO-ML, CLPSO-ML, small network at cluster 1 without any co-operation by PSO-ML and distributed in cluster Algorithm 1A and 1B are plotted and compared with the theoretical CRLB. The probability of resolution for the same methods is shown in Fig. 14. As evidents from Figs. 13 and 14 the distributed in-cluster algorithms at cluster 1 outperform the individual cluster without co-operation. But in this distributed estimation the centralized networks performance is not achieved. It is because of the nature of likelihood function. However, if the number of sensors is more, then the minima of the negative likelihood function are clearly distinguishable even at lower SNRs, but it provides better performance by involving high communication overhead. The performance comparison in terms of RMSE is listed in Table 2 at 5 dB of SNR and after 100 iterations. The results of this table demonstrate that due to distributed estimation, nearly 6 dB and 7 dB gain in RMSE is achieved over non-cooperative estimation using DPSO-ML and DCLPSO-ML algorithms respectively. The distributed RMSE is 1–2 dB less than the centralized one. But at the same time substantial saving $O(100)$ of communication overhead is achieved. The CLPSO based DOA estimation algorithms in both centralized and distributed manner provides slightly better performance compared to that of the PSO based algorithms.

7. Conclusion

The paper proposes a distributed *in-cluster* ML bearing estimation in wireless sensor networks using distributed PSO algorithms. In the

Table 2
Comparison of performances for different algorithms.

Algorithms	At cluster 1 using distributed PSO-ML	At cluster 1 using distributed CLPSO-ML	At cluster 1 using PSO-ML without co-operation	Centralized PSO-ML	Centralized CLPSO-ML
RMSE performance in dB	−14.63	−15.62	−8.32	−16.73	−16.78
Overall communication overhead	$O(9.6\times10^3)$	$O(9.6\times10^3)$	$O(9.6\times10^3)$	$O(6\times10^5)$	$O(6\times10^5)$

network the nodes form clusters to estimate the DOA by optimizing the ML function locally in diffusion mode of cooperation among clusters using diffusion PSO algorithms. This approach is independent of the connectivity of sensor nodes unlike distributed in-network algorithm. When the clusters are formed in the network and then estimate the source bearing in distributed manner its performance at lower SNR is nearly same as that computed centrally. However, at high SNR the distributed estimation provides 6–7 dB gain compared to that achieved by small network.

Acknowledgment

Thanks to the authors of the paper [28] for providing the CLPSO code.

References

[1] P. Stoica, K. Sharman, Maximum likelihood methods for direction-of-arrival estimation, IEEE Transactions on Acoustics, Speech and Signal Processing 38 (July (7)) (1990) 1132–1143.
[2] J. Capon, High-resolution frequency-wavenumber spectrum analysis, Proceedings of the IEEE 57 (August (8)) (1969) 1408–1418.
[3] R. Schmidt, Multiple emitter location and signal parameter estimation, IEEE Transactions on Antennas and Propagation 34 (3) (1986) 276–280.
[4] R. Roy, T. Kailath, ESPRIT-estimation of signal parameters via rotational invariance techniques, IEEE Transactions on Acoustics, Speech and Signal Processing 37 (July (7)) (1989) 984–995.
[5] G.J. Pottie, W.J. Kaiser, Wireless integrated network sensors, Communications of the ACM 43 (May) (2000) 51–53.
[6] R.L. Moses, O.L. Moses, D. Krishnamurthy, R. Patterson, A self-localization method for wireless sensor networks, EURASIP Journal on Applied Signal Processing 4 (2002) 348–358.
[7] J.C. Chen, K. Yao, R.E. Hudson, Source localization and beamforming, IEEE Signal Processing Magazine (March) (2002) 30–39.
[8] M. Wax, T. Kailath, Decentralized processing in sensor array, IEEE Transactions on Acoustics, Speech and Signal Processing 33 (October) (1985) 1123–1129.
[9] P. Stoica, A. Nehorai, T. Söderström, Decentralized array processing using the MODE algorithm, Circuits, Systems, and Signal Processing 14 (1) (1995) 17–38.
[10] T. Söderström, P. Stoica, Statistical analysis of decentralized MUSIC, Circuits, Systems, and Signal Processing 11 (4) (1992) 443–454.
[11] D. Estrin, M. Srivastav, Instrumenting the world with wireless sensor networks, in: International Conference on Acoustics, Speech, Signal Processing (ICASSP), May 2001, pp. 2033–2036.
[12] S. Marano, V. Matta, P. Willett, Distributed estimation in large wireless sensor networks via a locally optimum approach, IEEE Transactions on Signal Process. 56 (February (2)) (2008) 748–756.
[13] T. Panigrahi, G. Panda, B. Mulgrew, Novel distributed bearing estimation technique using diffusion PSO algorithm, IET Wireless Sensor Systems, http://dx.doi.org/10.1049/iet-wss.2011.0107, in press.
[14] A. Abbasi, M. Younis, A survey on clustering algorithms for wireless sensor network, Computer Communication 30 (October) (2007) 2826–2841.
[15] A. Chowdhury, S. Das, Automatic shape independent clustering inspired by ant dynamics, Swarm and Evolutionary Computation 3 (0) (2012) 33–45.
[16] A. Hatamlou, S. Abdullah, H. Nezamabadi-pour, A combined approach for clustering based on K-means and gravitational search algorithms, Swarm and Evolutionary Computation, 6 (2012) 47–52, http://dx.doi.org/10.1016/j.swevo.2012.02.003.
[17] J. Senthilnath, S. Omkar, V. Mani, Clustering using firefly algorithm: performance study, Swarm and Evolutionary Computation 1 (3) (2011) 164–171.
[18] T. Niknam, B. Amiri, An efficient hybrid approach based on PSO, ACO and *k*-means for cluster analysis, Applied Soft Computing 10 (1) (2010) 183–197.
[19] I. Ziskind, M. Wax, Maximum likelihood localization of multiple sources by alternating projection, IEEE Transactions on Acoustics, Speech and Signal Processing 36 (October (10)) (1988) 1553–1560.
[20] P.J. Chung, J.F. Böhme, DOA estimation using fast EM and SAGE algorithms, Signal Processing 82 (11) (2002) 1753–1762.
[21] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Neural Networks, 1995. Proceedings of the International Conference on IEEE, vol. 4, November/December 1995, pp. 1942–1948.
[22] Eberhart, Y. Shi, Particle swarm optimization: developments, applications and resources, in: Proceedings of the Congress on Evolutionary Computation 2001, vol. 1, IEEE service center, Piscataway, NJ, Seoul, Korea, 2001, pp. 81–86.
[23] M. Li, Y. Lu, Maximum likelihood DOA estimation in unknown colored noise fields, IEEE Transactions on Aerospace and Electronic Systems 44 (July (3)) (2008) 1079–1090.
[24] T. Panigrahi, D.H. Rao, G. Panda, B. Mulgrew, B. Majhi, Maximum likelihood DOA estimation in distributed wireless sensor network using adaptive particle swarm optimization, in: Proceedings of the ACM International Conference on Communication, Computing and Security (ICCCS2011), February 2011, pp. 134–136.
[25] T. Panigrahi, G. Panda, B. Mulgrew, B. Majhi, Maximum likelihood source localization in wireless sensor network using particle swarm optimization, in: Proceedings of the International Conference on Electronics Systems (ICES), January 2011, pp. 111–115.
[26] M. Li, Y. Lu, A refined genetic algorithm for accurate and reliable DOA estimation with a sensor array, Wireless Personal Communications 43 (2) (2007) 533–547.
[27] B. Majhi, G. Panda, B. Mulgrew, Distributed identification of nonlinear processes using incremental and diffusion type PSO algorithms, in: CEC'09—Proceedings of the Eleventh Conference on Congress on Evolutionary Computation, Piscataway, NJ, USA: IEEE Press, 2009, pp. 2076–2082.
[28] J. Liang, A. Qin, P. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Transactions on Evolutionary Computation 10 (June (3)) (2006) 281–295.
[29] X. Hu, R.C. Eberhart, Adaptive particle swarm optimization: detection and response to dynamic systems, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC2002, 2002, pp. 1666–1670.
[30] Z.-H. Zhan, J. Zhang, Y. Li, Y.-H. Shi, Orthogonal learning particle swarm optimization, IEEE Transactions on Evolutionary Computation 15 (December (6)) (2011) 832–847.
[31] T. Panigrahi, P.M. Pradhan, G. Panda , B. Mulgrew, Block least-mean square algorithm over distributed wireless sensor network, Journal of Computer Networks and Communications, 2012 (2012), Article ID 601287, http://dx.doi.org/10.1155/2012/601287.
[32] H.V. Trees, Optimum Array Processing, Wiley-Inter science Publication, 2002.
[33] D. Blatt, A. Hero, Distributed maximum likelihood estimation for sensor networks, in: International Conference on Acoustics, Speech, and Signal Processing, 2004.
[34] J.A. Hartigan, M.A. Wong, A *K*-means algorithm, Journal of the Royal Statistical Society. Series C (Applied Statistics), 28, (1) (1979), 100-108.
[35] A. Nedic, A. Ozdaglar, Cooperative distributed multi-agent optimization, in: Y. Eldar, D. Palomar (Eds.), Convex Optimization in Signal Processing and Communications, Cambridge University Press, 2008, pp. 340–386.
[36] L. Xiao, S. Boyd, S. Lall, Ascheme for robust distributed sensor fusion based on average consensus, in: Proceedings of the Fourth International Symposium Information Processing in Sensor Networks, Los Angles, CA, April 2005, pp. 63–70.
[37] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of the IEEE C Congress on Evolutionary Computation, 1998, pp. 69–73.
[38] J. Robinson, Y. Rahmat-Samii, Particle swarm optimization in electromagnetics, IEEE Transactions on Antennas and Propagation 52 (February (2)) (2004) 397–407.
[39] C.G. Lopes, A.H. Sayed, Diffusion least-mean squares over adaptive networks: formulation and performance analysis, IEEE Transactions on Signal Processing 56 (July (7)) (2008) 3122–3136.
[40] K.E. Parsopoulos, M.N. Vrahatis, Recent approaches to global optimization problems through particle swarm optimization, Natural Computing 1 (2002) 235–306.
[41] M. Clerc, J. Kennedy, The particle swarm—explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (February (1)) (2002) 58–73.
[42] M. Youssef, A.M. Youssef, M.F. Younis, Overlapping multi-hop clustering for wireless sensor networks, IEEE Transactions on Parallel and Distributed Systems 22 (December) (2009) 1844–1856.