Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 9 (5): 854-864, 2009
ISSN 1812-5654
© 2009 Asian Network for Scientific Information

Distributed Data Clustering Using Expectation Maximization Algorithm

B. Safarinejadian, M.B. Menhaj and M. Karrari
Department of Electrical Engineering, Amirkabir University of Technology, 424 Hafez Ave., Tehran, Iran

Abstract: In this study, a distributed expectation maximization (DEM) algorithm is first introduced in a general form for estimating parameters of a finite mixture of components. This algorithm is used for density estimation and clustering of the data distributed over the nodes of a network. Then, a distributed incremental EM algorithm (DIEM) with a higher convergence rate is proposed. After a full derivation of distributed EM algorithms, convergence of both DEM and DIEM algorithms is studied based on the negative free energy concept. It is shown that these algorithms increase the negative free energy incrementally at each node until reaching the convergence. Finally, the proposed algorithms are applied to cluster analysis of gene-expression data. Simulation results approve that DIEM remarkably outperforms DEM.

Key words: Distributed data mining, EM algorithm, cluster analysis, density estimation, gene-expression data

INTRODUCTION

Recent advances in sensor, high throughput data acquisition and digital information storage technologies have made it possible to acquire, store and process large volumes of data in digital form in a number of domains. For example, biologists are generating gigabytes of genome and protein sequence data at steadily increasing rates. Organizations have begun to capture and store a variety of data about various aspects of their operations (e.g., products, customers and transactions). Complex distributed systems (e.g., computer systems, communication networks, power systems) are equipped with sensors and measurement devices that gather and store, a variety of data that is useful in monitoring, controlling and improving the operation of such systems.

Distributed data mining (DDM) has recently emerged as an extremely important area of research. The objective, here, is to perform data mining tasks (such as association rule mining, clustering, classification) on a distributed database, that is, a database distributed across several sites (nodes) connected by a network. Research in this field aims at mining information from such databases while minimizing the amount of communication between nodes. For example, Wolff and Schuster (2004) presented an algorithm for distributed association rule mining in peer-to-peer systems. Datta *et al.* (2006) extended K-means clustering to the distributed scenario.

The EM (expectation maximization) algorithm (Ordonez and Omiecinski, 2005; McLachlan and Krishnan, 1997; McLachlan and Peel, 2000; Neal and Hinton, 1999; Verbeek *et al.*, 2003), is an important method of density estimation in which some of the variables are assumed to be missing or unobservable. Recently there has been some research on distributed density estimation using the EM algorithm. Nowak (2003) developed a distributed EM algorithm for density estimation in sensor networks assuming that the measurements are statistically modeled by a mixture of Gaussians. Kowalczyk and Vlassis (2005) proposed a gossip-based distributed EM algorithm for Gaussian mixture learning named Newscast EM, in which the E and M steps of the EM algorithm are first performed locally, the global estimate of means and covariances are then obtained through a gossip-based randomized method. Lin *et al.* (2005) has also developed a privacy-preserving distributed EM algorithm for mixture modeling. This method performs clustering on distributed data and meanwhile, controls data sharing and prevents disclosure of individual data items or any results that can be traced to an individual site. All the above methods have assumed the components to be Gaussian. Here, a more general case is considered in which components belong to an exponential family.

Assume that the data set distributed over the nodes of a network can be modeled by a finite mixture model. Here, a general distributed expectation maximization algorithm (DEM) is proposed first to estimate the parameters of this mixture without transferring the nodes data to a central unit. Then, a distributed incremental EM algorithm (DIEM) is developed with a higher convergence rate. Afterwards, convergence of both DEM and DIEM algorithms are studied based on the negative energy concept used in statistical physics. The proposed algorithms are then applied to cluster analysis of gene-expression data which is distributed in a network. The proposed methods can also be used as general distributed

Corresponding Author: Behrooz Safarinejadian, Department of Electrical Engineering, Amirkabir University of Technology, 424 Hafez Ave., Tehran, Iran Tel: +98-9124306341 Fax: +98-21-66419728

data mining algorithms for density estimation and clustering of the data distributed over the nodes of a network.

Problem statement: Consider a network of M nodes and a d-dimensional random vector $Y_m = [Y_m^1, \ldots, Y_m^d]^T$ which corresponds to node m. Each data (observation) $y_m = [y_m^1, \ldots, y_m^d]^T$ of the node *m* is a realization of the random vector Y_m. Assume that distribution of the measurements is represented by a finite mixture of components:

$$p\left(y_m \mid \theta\right) = \sum_{j=1}^{J} \alpha_{m,j} p\left(y_m \mid \varphi_j\right) \tag{1}$$

where, $\alpha_m \equiv \{\alpha_{m,j}\}_{j=1}^{J}$ are the mixture probabilities at node m, φ_j is the set of parameters defining the jth component and J is the number of mixture components. Assume that $\varphi \equiv \{\varphi_j\}_{j=1}^{J}$ and $\theta = \varphi \cup \{\alpha_m\}_{m=1}^{M}$. The mixture probabilities $\{\alpha_{m,j}\}$ may be different at various nodes while the parameters φ_j are the same throughout the network. The set of data points of the mth node is represented by $y_m = \{y_{m,i}\}_{i=1}^{N_m}$ in which N_m is number of observations at node m. It is assumed that observations of each node are independent and identically distributed.

This study describes a distributed algorithm for computing a maximum likelihood estimate; i.e., θ maximizing the log-likelihood function:

$$L(\theta) \equiv \sum_{m=1}^{M} \sum_{i=1}^{N_m} \log\left(\sum_{j=1}^{J} \alpha_{m,j} p\left(y_{m,i} \mid \varphi_j\right)\right) \tag{2}$$

where, $p(y|\varphi)$ denotes the evaluation of an exponential density with parameter vector θ at the point y.

Consider a set of missing variables $Z = (z_{m,\ i})$ corresponding to $Y = \{y_{m,i}\}$. Each $z_{m,i} = [z_{m,i}^1, \ldots, z_{m,i}^J]$ is a binary vector indicating by which component the data $y_{m,i}$ is produced. We would say $y_{m,i}$ is produced by the jth component of the mixture if for all r ≠ j, $z_{m,i}^r = 0$ and $z_{m,i}^j = 1$. Assume that $z_{m,i}$ is a realization of the random vector Z_m. The pair $x_{m,i} = (y_{m,i}, z_{m,i})$ is regarded as the complete data and we write $X = \{Y, Z\}$ in which $X = \{x_{m,i}\}$. The random vector X_m is also defined as $X_m = \{Y_m, Z_m\}$.

Define θ^t, the set of parameters at the t-th iteration of the EM algorithm. Define the conditional expectation:

$$Q\ \left(\theta;\theta^t\right) = E_z\left[\log p(x \mid \theta) \mid y, \theta^t\right] = \sum_z \log p(y, z \mid \theta) p\left(z \mid Y, \theta^t\right) \tag{3}$$

where, $p(x|\theta)$ denotes the joint density of y and z with parameters θ.

EM (expectation maximization) is an iterative algorithm to obtain the maximum likelihood estimate of the finite mixture parameters. At the E step of the EM algorithm, the Q function is calculated and at the M step, the parameter vector θ is estimated such that the Q function is maximized.

The data at each node are assumed to be statistically independent in this study. If the data are (spatially or temporally) correlated, then the simple independent likelihood model can still be employed by interpreting it as a pseudolikelihood (Besg, 1986). Under mild conditions the maximum pseudolikelihood estimates tend to the true maximum likelihood estimates as the number of data tends to infinity.

Distributed EM algorithm: Here, distributed density estimation based on a finite mixture model is described. The EM algorithm is used in a distributed approach in order to estimate the parameters of this model. Here, a general distributed EM algorithm is proposed for estimating the parameters of a finite mixture model whose components belong to the exponential family. Then this algorithm is expressed in the special case of Gaussian mixture model.

A general distributed EM algorithm: Here, a general distributed EM algorithm is developed to estimate the parameters of a finite mixture model. The distributed EM algorithm cycles through the nodes of the network and estimates the parameters θ such that the log-likelihood function represented by Eq. 2 is maximized.

At each node, the local conditional expectation of complete data log-likelihood is defined as:

$$Q_m\left(\theta;\theta_m^t\right) = E_z\left[\log p\left(x_m \mid \theta\right) \mid y_m, \theta_m^t\right] \tag{4}$$

where, θ_m^t is the vector of estimated parameters at node *m* and iteration t and $p(x_m|\theta)$ denotes the probability distribution of the random vector X_m given θ. Total conditional expectation can be written as:

$$Q\left(\theta;\theta_1^t, \ldots \theta_M^t\right) = \sum_{m=1}^{M} Q_m\left(\theta;\theta_m^t\right) \tag{5}$$

If the mixture components belong to the exponential family, calculating Q is reduced to calculating a vector of sufficient statistics that can be incrementally updated. The reason behind this is that with models in the exponential family, the inferential import of the complete data can be represented by a vector of sufficient statistics. Denoting this vector of sufficient statistics as $\tilde{s}(y,z) = \sum_i \tilde{s}_i(y_i, z_i)$, the E step of EM algorithm can be implemented by computing $s^t = E_z[\tilde{s}(y,z)]$ and the M step can be performed by setting θ^t to the θ which maximizes the likelihood function given s^t.

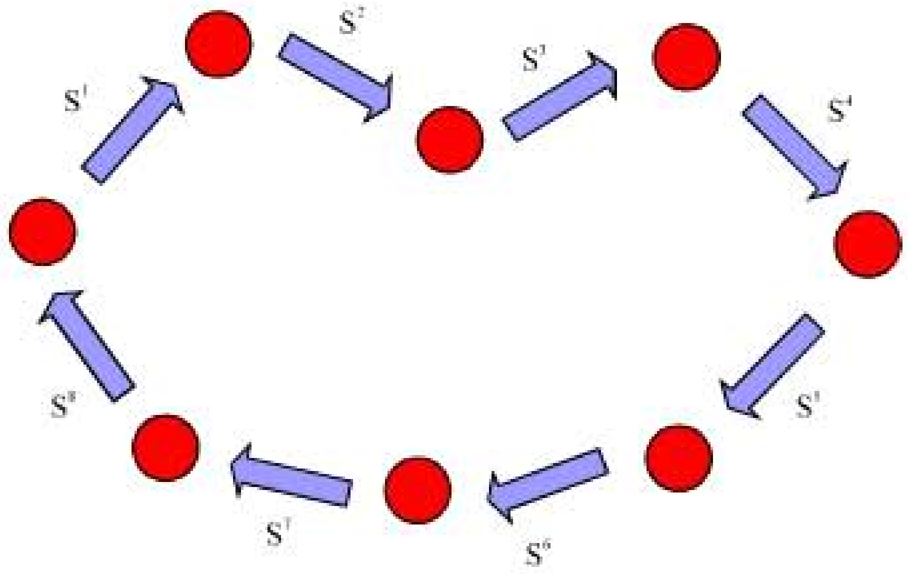

Fig. 1: Communication cycle in a typical network

In the case that the data set is distributed over the nodes of a network, s^t will be given by $s = \sum_m s_m$ in which $s_m = E_z[\tilde{s}_m(y_m, z_m)]$.

The distributed EM algorithm works as follows: The values of parameters that should be estimated are first initialized. The distributed EM algorithm, at each node, first updates conditional expectation of complete data log-likelihood and then estimates the parameters of the finite mixture in order to maximize this expectation. In other words, at each iteration t, node m receives the sufficient statistics s^t from the previous node, calculates the local sufficient statistics using: $s_m = E_z[\tilde{s}_m(y_m, z_m)]$ and updates s^t as:

$$s^t = s^{t-1} + s^t_m - s^{t-1}_m \tag{6}$$

s^t is then transmitted to the next node and this procedure is continued until convergence is reached. Figure 1 shows the communication cycle in a typical network.

The amount of variations of the log-likelihood function is considered as a stopping criterion in this algorithm. If this value is less than a certain threshold ε, the algorithm will stop. Here, after updating the parameters using the data of each node m, the value of local log-likelihood function corresponding to that node is calculated:

$$L^m(\theta^t) \equiv \sum_{i=1}^{N_m} \log\left(\sum_{j=1}^{J} \alpha^t_{m,j}\, p(y_{m,i} \mid \theta^t_j) \right) \tag{7}$$

Whenever, the difference $L^m(\theta^{t+1}) - L^m(\theta^t)$ becomes less than the convergence threshold, the algorithm will stop. Instead of likelihood variations, parameter variations can be also used as another stopping criterion.

Since, scalability is an important feature of distributed algorithms, here scalability of the proposed DEM is analyzed and compared to that of the centralized EM algorithm; in which all nodes send their data to a central unit.

Assuming that N_b is the number of bytes communicated between two nodes per time step, it can be found that the communication in bytes for the centralized method in which all nodes send their data to the center of the network is $\sqrt{M}\left(1+2+\ldots+\sqrt{M}/2\right)N_b = O\left(M^{3/2}\right)$. The worst case in this method is that the centralized unit is not in the center of the network, but is at the end of it. The communication in bytes for such a case is $(M-1+M-2+\ldots+1)N_b = O(M^2)$. Ones the centralized unit receives all data, it can run the standard EM algorithm.

For the proposed DEM, the communication and computation are executed iteratively. The communication cost is related to the number of loops, i.e., the accuracy of the estimated results. By denoting T as the number of loops, the communication in bytes for the DEM is $MN_bT = O(M)$. Therefore, unlike the centralized method the proposed DEM is scalable. The peer-to-peer distributed EM algorithm which will be presented later in this study has even better scalability features.

If the data are possibly correlated, then the DEM algorithm can still be applied with the independent likelihood structure employed here. In that case, the independent likelihood can be interpreted as a pseudolikelihood (Besg, 1986) and under mild conditions the maximum pseudolikelihood estimates tend to the true maximum likelihood estimates as the number of data tends to infinity.

The distributed EM algorithm for a Gaussian mixture model: The distributed EM algorithm proposed by Nowak (2003) is a special case of the general DEM algorithm that was presented in the earlier. Here, the study of Nowak (2003) is reviewed briefly in which components are assumed to be Gaussian. In this case, the function Q_m can be rewritten as:

$$Q_m(\theta;\theta^t_m) = \sum_{i=1}^{N_m} \sum_{j=1}^{J} w^{t+1}_{m,i,j} (\log \alpha_{m,j} + \log N(y_{m,i} \mid \mu_j, \Sigma_j)) \tag{8}$$

in which

$$w^{t+1}_{m,i,j} = \frac{\alpha^t_{m,j} N\left(y_{m,i} \mid \mu^t_{m,j}, \Sigma^t_{m,j}\right)}{\sum_{n=1}^{J} \alpha^t_{m,n} N\left(y_{m,i} \mid \mu^t_{m,n}, \Sigma^t_{m,n}\right)} \tag{9}$$

where, N(y|μ, Σ) denotes the evaluation of a Gaussian density with mean μ and covariance Σ at the point y.

In this case, the sufficient statistics vector is defined as

$$s^t_m = \{w^t_{m,j}, a^t_{m,j}, b^t_{m,j}\}^J_{j=1}$$

in which:

$$w^t_{m,j} = \sum_{i=1}^{N_m} w^t_{m,i,j} \tag{10}$$

$$a^{t}_{m,j} = \sum_{i=1}^{N_m} w^{t}_{m,i,j} y_{m,i} \tag{11}$$

$$b^{t}_{m,j} = \sum_{i=1}^{N_m} w^{t}_{m,i,j} y^{2}_{m,i} \tag{12}$$

In the DEM algorithm the following processing and communication procedure is performed at each node. At iteration t+1, node m receives the value of w^{t}_{j}, a^{t}_{j} and b^{t}_{j} from the earlier node and calculates its local sufficient statistics as:

$$w^{t+1}_{m,i,j} = \frac{\alpha^{t}_{m,j} N\left(y_{m,i} \mid \mu^{t}_{j}, \Sigma^{t}_{j}\right)}{\sum_{n=1}^{J} \alpha^{t}_{m,n} N\left(y_{m,i} \mid \mu^{t}_{n}, \Sigma^{t}_{n}\right)} \tag{13}$$

$$w^{t+1}_{m,j} = \sum_{i=1}^{N_m} w^{t+1}_{m,i,j} \tag{14}$$

Then, the value of sufficient statistics are updated by:

$$w^{t+1}_{j} = w^{t}_{j} + w^{t+1}_{m,j} - w^{t}_{m,j} \tag{15}$$

$$a^{t+1}_{j} = a^{t}_{j} + a^{t+1}_{m,j} - a^{t}_{m,j} \tag{16}$$

$$b^{t+1}_{j} = b^{t}_{j} + b^{t+1}_{m,j} - b^{t}_{m,j} \tag{17}$$

The mean and covariances are calculated as:

$$\mu^{t}_{j} = \frac{a^{t}_{j}}{w^{t}_{j}} \tag{18}$$

$$\Sigma^{t}_{j} = \frac{b^{t}_{j}}{w^{t}_{j}} - \mu^{t}_{j}(\mu^{t}_{j})' \tag{19}$$

And node m updates its mixture probabilities:

$$\alpha^{t+1}_{m,j} = \frac{1}{N_m} \sum_{i=1}^{N_m} w^{t+1}_{m,i,j} \tag{20}$$

At last the updated values of $s^{t} = \{w^{t}_{j}, a^{t}_{j}, b^{t}_{j}\}_{j=1}^{J}$ are sent to the next node and this procedure is repeated.

A distributed incremental EM: After presenting a general view of the EM algorithm, Neal and Hinton (1999) has developed an incremental EM algorithm. Thiesson *et al.* (2001) has also used the incremental EM algorithm for data mining in large data bases. Here first, the incremental EM algorithm is introduced briefly and then a distributed incremental EM (DIEM) algorithm is proposed.

The incremental EM: An incremental EM algorithm attempts to reduce the computational cost by performing partial E-steps. Let $y = \{y_1, ..., y_K\}$ denote a particular partition of the data into mutually disjoint blocks of data cases. The incremental EM algorithm iterates through the blocks in a cyclic way. At each iteration, a partial E-step is performed by updating only a part of the conditional expectation for the complete data log-likelihood (the Q-function) before performing an M-step. A generic cycle of the incremental EM algorithm is shown below.

E-step: Select the data block y_k to update the parameters as follows:

- Compute $Q_k\left(\theta;\theta^{t}\right) = E_{z_k}\left[\log p\left(x \mid \theta\right) \mid y_k, \theta^{t-1}\right]$
- Set $Q_j\left(\theta;\theta^{t}\right) = Q_j\left(\theta;\theta^{t-1}\right)$for $j \neq k$
- Construct $Q\left(\theta;\theta^{t}\right) = \sum_{k} Q_k\left(\theta;\theta^{t}\right)$

M-step: Choose θ^{t+1} as the value that maximizes $Q(\theta; \theta^{t})$.

Notice the way in which the E-step incrementally constructs the Q-function to be maximized. In each iteration, the algorithm only computes a fraction of the Q-function under consideration, namely the Q_k associated with the block of data Y_k. For all other data blocks, the algorithm reuses previously computed contributions to the Q-function. In an efficient implementation, we incrementally update the Q-function by adding the difference between the new and old Q_k components:

$$Q\left(\theta;\theta^{t}\right) = Q\left(\theta;\theta^{t-1}\right) + Q_k\left(\theta;\theta^{t}\right) - Q_k\left(\theta;\theta^{t-1}\right) \tag{21}$$

Note that this algorithm has an additional cost beyond EM, which is the storage of Q_k for all blocks $k = 1, ..., K$. As in the EM algorithm, if the statistical model is a subfamily of an exponential family, then the E-step can be cast as constructing expected sufficient statistics for the statistical model.

The proposed distributed incremental EM: Here, both the aforementioned incremental EM algorithm and partitioning of the measurements of each node is used to establish a distributed incremental EM possessing a faster convergence rate. Basically, measurements of each node m are partitioned into K_m disjoint blocks and then the mixture parameters are estimated using a distributed incremental EM algorithm. Here, the measurements of sensor m are represented by $y_m = \{y_m, ..., y_{m,K_m}\}$ so that each $y_{m,k}$ represents a block of data and K_m is the number of blocks at node m. The distributed incremental EM algorithm proceeds through the nodes cyclically and performs an incremental EM algorithm at each node using the local data at that node and the vector of sufficient statistics received from the earlier node. It should be noted that the procedure given here does not necessarily require a cyclic communication structure. The following processing and communication procedure is performed at each node.

At iteration t+1, node m receives $s^t = \{w_j^t, a_j^t, b_j^t\}_{j=1}^J$ from the earlier node and calculates $w_{m,k,j}^{t+1}$ using the block of data $y_{m,k}$:

$$w_{m,k,j,i}^{t+1} = \frac{\alpha_{m,j}^t N(y_{m,k,i} \mid \mu_j^t, \Sigma_j^t)}{\sum_{n=1}^J \alpha_{m,n}^t N(y_{m,n,i} \mid \mu_n^t, \Sigma_n^t)} \qquad (22)$$

$$w_{m,k,j}^t = \sum_{i=1}^{N_{m,k}} w_{m,k,j,i}^t \qquad (23)$$

Note that the index k represents a block of data and we have: $y_{m,k} = \{y_{m,k,i}\}_{i=1}^{N_{m,k}}$. The values of $a_{m,k,j}^t$ and $b_{m,k,j}^t$ corresponding to the block of data $Y_{m,k}$ are also calculated:

$$a_{m,k,j}^t = \sum_{i=1}^{N_{m,k}} w_{m,k,j,i}^t \, y_{m,k,i} \qquad (24)$$

$$b_{m,k,j}^t = \sum_{i=1}^{N_{m,k}} w_{m,k,j,i}^t \, y_{m,k,i}^2 \qquad (25)$$

Then using the following incremental relations and the vector of sufficient statistics obtained for the block of data $y_{m,k}$ the values of $w_{m,j}^{t+1}$, $a_{m,j}^{t+1}$ and $a_{m,j}^{t+1}$ are updated:

$$w_j^{t+1} = w_j^t + w_{m,k,j}^{t+1} - w_{m,k,j}^t \qquad (26)$$

$$a_j^{t+1} = a_j^t + a_{m,k,j}^{t+1} - a_{m,k,j}^t \qquad (27)$$

$$b_j^{t+1} = b_j^t + b_{m,k,j}^{t+1} - b_{m,k,j}^t \qquad (28)$$

Since, we use incremental EM, here the values of $w_{m,k,j}^t$, $a_{m,k,j}^t$ and $b_{m,k,j}^t$ at the earlier iteration should be saved. At this node the following means and covariances are calculated:

$$\mu_j^t = \frac{a_j^t}{w_j^t} \qquad (29)$$

$$\Sigma_j^t = \frac{b_j^t}{w_j^t} - \mu_j^t (\mu_j^t)' \qquad (30)$$

And the mixture probabilities of node m are updated:

$$\alpha_{m,j}^t = \frac{w_{m,k,j}^{t+1}}{N_{m,k}} \qquad (31)$$

In the sequel, this procedure is continued for other data blocks of node m. After processing all K_m data blocks, the updated values of $\{w_j^t, a_j^t, b_j^t\}$ are sent to the next node and this procedure is repeated. Figure 1 shows the communication procedure in a typical sensor network.

Convergence analysis: The convergence behavior of standard EM in the Gaussian mixture case has been examined by Thiesson *et al.* (2001) and Xu and Jordan (1996). Ma *et al.* (2000) has also shown that the incremental EM algorithm converges to a fixed point. Usually, the EM fixed points are points of local maxima of the log likelihood, although saddle points are also possible. The standard EM algorithm converges linearly in general and can display super linear convergence for well separated Gaussian mixtures.

Here, the results of Neal and Hinton (1999) and Thiesson *et al.* (2001) are used to prove the convergence of DEM and also DIEM algorithms.

The EM algorithm performs maximum likelihood estimation for a set of data in which some variables cannot be observed. In Neal and Hinton (1999), a function F is introduced which resembles negative free energy and it is shown that the M step maximizes this function with respect to the model parameters and the E step maximizes it with respect to the distribution over the unobserved variables. Here, the function F is used to analyze the convergence behavior of the DEM and DIEM algorithms. It will be shown that in these algorithms, each node improves the local part of F corresponding to itself and leaves the other parts unchanged. If data sets of different nodes are assumed to be independent, F is the sum of local F's, denoted as F_m, of all nodes of the network: $F = \sum_{m=1}^M F_m$. The goal here is to show that F monotonically increases and eventually converges to its maximum value.

Convergence analysis of the DEM algorithm: As mentioned before, the M step of the EM algorithm maximizes the Q function:

$$\theta^{t+1} = \arg\max_{\theta} Q(\theta; \theta^t) \qquad (32)$$

in which:

$$Q(\theta; \theta^t) = E_z\left[\log p(y, z \mid \theta) \mid y, \theta^t\right] = \sum_z \log p(y, z \mid \theta) p(z \mid y, \theta^t) \qquad (33)$$

In the distributed EM algorithm, assuming that data sets at different nodes are independent, the Q function can be written as a sum of local Q functions.

$$Q(\theta;\theta_1^t,\ldots,\theta_M^t)=\sum_{m=1}^{M}Q_m(\theta;\theta_m^t) \tag{34}$$

θ_m is the vector of estimated parameters at node m and:

$$\begin{aligned}Q_m(\theta;\theta_m^t)&=E_{z_m}[\log p(x_m\mid\theta)\mid y_m,\theta_m^t]\\&=\sum_{z_m}\log p(x_m\mid\theta)p(z_m\mid y_m,\theta_m^t)\end{aligned} \tag{35}$$

Finally, the update equation of the distributed EM algorithm can be written as:

$$\theta^{t+1}=\arg\max_{\theta}Q\left(\theta;\theta_1^t,\ldots,\theta_M^t\right) \tag{36}$$

Here, convergence of DEM is proved based on the negative free energy concept. Assume that function F is defined as:

$$F(\tilde{p},\theta)=E_z[\log p(y,z\mid\theta)]+H(\tilde{p}) \tag{37}$$

In which $H(\tilde{p})=-E_z[\log\tilde{p}(z)]$ is the entropy of $\tilde{p}(z)=p(z\mid y,\theta)$. It has been shown by Neal and Hinton (1999) that E and M steps of the EM algorithm increase $F(\tilde{p},\theta)$ monotonically and finally the algorithm converges to $(\tilde{p}^*,\theta^*)$, where, θ^* is the local maximum (or saddle point) of the log likelihood function. The F function represents the negative free energy initially introduced in statistical physics.

If the data set of different nodes are independent, the joint probability density function of Y and Z can be written as $p(y,z\mid\theta)=\prod_m p(y_m,z_m\mid\theta_m)$. Based on the independence assumption we also have $\tilde{p}=\prod_m\tilde{p}_m(y_m)$. Therefore, F can be written as $F(\tilde{P},\theta)=\sum_m F_m(\tilde{p}_m,\theta_m)$, in which:

$$F_m\left(\tilde{p}_m,\theta_m^t\right)=E_{z_m}\left[\log p\left(x_m\mid\theta_m^t\right)\right]+H(\tilde{p}_m) \tag{38}$$

The vector of estimated parameters at node m is denoted by θ_m, $\tilde{p}_m$ is the conditional probability density function of z_m given y_m and θ, i.e., $\tilde{p}_m=p(z_m\mid y_m,\theta)$ and $H(\tilde{p}_m)$ is the entropy of $\tilde{p}_m$ defined as $H(\tilde{p}_m)=-E_{\tilde{p}_m}[\log\tilde{p}_m]$.

Since, each node has its own estimated parameters, the function F can be described as:

$$F(\tilde{p},\theta_1^t,\ldots\theta_M^t)=\sum_{m=1}^{M}F_m(\tilde{p}_m,\theta_m^t) \tag{39}$$

To complete the convergence proof, we proceed as follows. At each node m, the E step maximizes the value of $F_m(\tilde{p}_m,\theta_m)$ with respect to $\tilde{p}_m$ by setting $\tilde{p}_m=p(z_m\mid y_m,\theta_m^t)$, while the values of $F_j(\tilde{p}_j,\theta_j^t), j\neq m$ keep unchanged. Therefore, since the values of other F_j's are fixed, the total F increases. Hence, at each E step, the value of F is increased by improving $\tilde{p}_m$.

It is easy to show that the M step will also increase the function F. As it was mentioned before, if the $\tilde{p}_m$'s are assumed to be fixed, the M step updates θ by maximizing the following Q function:

$$\theta^{t+1}=\arg\max_{\theta}Q\left(\theta;\theta_1^t,\ldots\theta_M^t,y\right) \tag{40}$$

The sum of log likelihood expectations in Eq. 38 is the Q function.

$$Q(\theta;\theta_1^t,\ldots\theta_M^t)=\sum_{m=1}^{M}Q_m(\theta;\theta_m^t)=\sum_{m}(E_{z_m}[\log p\left(x_m\mid\theta\right)\mid y_m,\theta_m^t]) \tag{41}$$

Since, the second term of Eq. 38 is the entropy and it is independent of θ, maximizing the Q function at M step is equivalent to maximizing the F function. Therefore, the distributed EM is a nondecreasing algorithm that incrementally improves the value of *F* at each node.

In theorem 2 of Neal and Hinton (1999), it has been shown that if $F(\tilde{p},\theta)$ has a local maximum at $\tilde{p}^*$ and θ^*, then $L(\theta)$ has a local maximum at θ^* as well. Similarly, if F has a global maximum at $\tilde{p}^*$ and θ^*, then L has a global maximum at θ^*. Regarding this theorem, because (1) $F(\tilde{p},\theta)$ represents an upper bound of $F(\tilde{p},\theta)$ and (2) DEM is a nondecreasing algorithm, the function F will eventually converge to its maximum at $\tilde{p}^*$ and θ^*. Consequently, $L(\theta)$ will converge to its maximum at θ^* and DEM is indeed a convergent algorithm. In the next section, this convergence analysis is extended to improve convergence of the distributed incremental EM algorithm.

Convergence analysis of the DIEM algorithm: The M step of DIEM can be rewritten as:

$$\theta_m^{t+1}=\arg\max_{\theta}Q(\theta;\theta_1^t,\ldots,\theta_m^t,\ldots,\theta_M^t) \tag{42}$$

where, θ_M^t is the set of estimated parameters at node m based on the earlier data block and θ_m^{t+1} is the set of estimated parameters using the next block of data. Like DEM, the function Q is defined as:

$$Q\ (\theta;\theta_1^t,\ldots,\theta_M^t)=\sum_{m=1}^{M}Q_m\ (\theta;\theta_m^t) \tag{43}$$

Where:

$$Q_m\ (\theta;\theta^t_m) = \sum_k Q_{m,k}\ (\theta;\theta^t_m) \tag{44}$$

and

$$\begin{aligned} Q_{m,k}\ (\theta;\theta^t_m) &= E_{z_{m,k}}\left[\log p\ (x_{m,k} \mid \theta)\right] \\ &= \sum_{z_{m,k}} \log p\ (x_{m,k} \mid \theta)\ p\ (z_{m,k} \mid y_{m,k}, \theta^t_m) \end{aligned} \tag{45}$$

We also have:

$$F\ (\tilde{p}, \theta^t_1, \ldots \theta^t_M) = \sum_{m=1}^{M} F_m\ (\tilde{p}_m,\ \theta^t_m) \tag{46}$$

Where:

$$\begin{aligned} F_m\ (\tilde{p}_m, \theta^t_m) &= \sum_{k=1}^{K} F_{m,k}\ (\tilde{p}_{m,k}, \theta^t_m) \\ &= \sum_{k=1}^{K} (E_{z_{m,k}}\ [\log p\ (x_{m,k} \mid \theta^t_m)] + H(\tilde{p}_{m,k})) \end{aligned} \tag{47}$$

The DIEM algorithm, at each node m, increases the total F function by processing each block's data. The E step maximizes $F_{m,k}\ (\tilde{p}_{m,k}, \theta^t_m)$ with respect to $\tilde{p}_{m,k}$ by setting $\tilde{p}_{m,k} = p(x_{m,k} \mid y_{m,k}, \theta^t_m)$ and keeping $F_{m,j}\left(\tilde{p}_{m,j}, \theta^t_m\right), j \neq k$ unchanged. This makes the function F_m nondecreasing. Consequently, since the value of $F_n\left(\tilde{p}_n, \theta^t_n\right), n \neq m$ does not change, the total F function will be increased by performing the E step.

It is easy to show that the M step also increases the F function. As mentioned before, if $\tilde{p}_{m,k}$'s are assumed to be fixed, the M step updates θ by maximizing the following Q function:

$$\theta^{t+1}_m = \arg\max_\theta Q\ (\theta;\theta^t_1, \ldots, \theta^t_m, \ldots, \theta^t_M) \tag{48}$$

The first term of Eq. 47, i.e., expected log likelihood of the complete data, is the $Q_{m,k}$ function. Since, the second term of this equation is the entropy of $\tilde{p}_{m,k}$ and independent of θ, maximizing the Q function at the M step, is equivalent to maximizing the F function. Therefore, both E and M steps of the DIEM algorithm incrementally increase the value of F at each node until the convergence is reached. This proves the nondecreasing property of the DIEM algorithm.

Note that as in the DEM convergence analysis, since we have shown that the DIEM is a nondecreasing algorithm, after some iterations the function F will converge to its maximum at $\tilde{p}^*$ and θ*; hence L(θ) will converge to its maximum at θ*. Consequently, DIEM represents a convergent algorithm so that at each node it increases the value of F until it is maximized at θ* based on the assumption that $F(\tilde{p}^*, \theta^*)$ is a maximum or upper bound of $F(\tilde{p}, \theta)$.

Applying distributed EM to cluster analysis of gene-expression data: DNA microarray technology has now made it possible to simultaneously monitor the expression level of thousands of genes during important biological processes and across collections of related samples. Elucidating the patterns hidden in gene expression data offers a tremendous opportunity for an enhanced understanding of functional genomics. However, the large number of genes and the complexity of biological networks greatly increase the challenge of comprehending and interpreting the resulting mass of data, which often consists of millions of measurements. A first step toward addressing this challenge is the use of clustering techniques, which is essential in the data mining process to reveal natural structures and identify interesting patterns in the underlying data. Cluster analysis seeks to partition a given data set into groups based on specified features so that the data points within a group are more similar to each other than the points in different groups. A very rich literature on cluster analysis has developed over the past three decades. Many conventional clustering algorithms have been adapted or directly applied to gene expression data and also new algorithms have recently been proposed specifically aiming at gene expression data.

Model-based clustering approaches (McLachlan *et al*., 2002; Yeung *et al*., 2001; Fraley and Raftery, 1998; Ghosh and Chinnaiyan, 2002; Jiang *et al*., 2004) provide a statistical framework to model the cluster structure of gene expression data. The data set is assumed to come from a finite mixture of underlying probability distributions, with each component corresponding to a different cluster. The parameters of these components are usually estimated by the EM algorithm. When the EM algorithm is converged, each data object is assigned to the component (cluster) with the maximum conditional probability.

However, microarray data deposited in the public domain, demand decentralized access to these data (Stratowa, 2003; Chernyi *et al*., 2004). Since, the corresponding datasets have already been cleaned and validated, an obvious choice is their storage in a distributed data warehouse. Powerful data mining techniques can then be applied to discover hidden patterns and to extract knowledge from microarray data. Considering the ever-increasing amount of microarray data and the increasing computing requirements for large-scale data mining and analysis, using efficient distributed data clustering algorithms with reasonable computational cost for microarray data analysis is inevitable.

In the case that the data set is distributed in a network, centralized EM algorithm is not applicable. Here, the proposed DEM and DIEM algorithms are applied to cluster analysis of gene-expression data. In this study, distributed EM is capable of performing clustering on extremely large or geographically distributed set of gene expression data. Here, the performance of distributed EM and DIEM algorithms are compared with each other.

Although the use of Gaussian components to simulate data is clearly not ideal, the Gaussian model has been shown to be a reasonably good approximation for suitably normalized real data (Yeung *et al*., 2001). Here, the case where the data are generated by two types of samples is considered and both univariate and multivariate (two dimensions) Gaussian components are treated.

Here, a two dimensional microarray data is first considered to evaluate performance of the proposed DEM and DIEM algorithms to estimate parameters of the mixture model by which the data is produced. Convergence rate and computational cost of these algorithms are also studied. A multivariate data model is considered next to evaluate the proposed methods classification performance. From a biologist's perspective what matters most in the context of clustering is whether the algorithm classifies the microarray data correctly or not. The aim of using artificial data is to provide a framework in which the prediction accuracy of the model based clustering approaches is studied. The focus is on correct data clustering implied by misclassification rates. Finally, convergence rate of DIEM and DEM algorithms are compared based on different K values and various gene-expression data dimensions.

Here, a two dimensional microarray data set simulated from a two component Gaussian mixture model is considered first. A network of 100 nodes is used in this study. Each node contains 1000 data points that are partitioned into 10 disjoint blocks of data. True and estimated parameters of the components are shown in Table 1 and 2, respectively. As it is seen, good estimates of the true values have been obtained. The values offered in these tables are the mean value of the estimated parameters at all nodes of the network.

Table 1: True mean and covariance matrices

	True distribution	
Component	Mean vector	Covariance matrix
1	[0,0]	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
2	[-0.2, -0.2]	$\begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}$

Table 2: Fitted mean and covariance matrices

	Estimated distribution			
	DEM algorithm		DIEM algorithm	
Component	Mean vector	Covariance matrix	Mean vector	Covariance matrix
1	[0.015, -0.001]	$\begin{bmatrix} 1.017 & -0.001 \\ -0.001 & 1.012 \end{bmatrix}$	[0.013, -0.001]	$\begin{bmatrix} 1.006 & -0.001 \\ -0.001 & 1.002 \end{bmatrix}$
2	[-0.200, -0.200]	$\begin{bmatrix} 0.010 & 0.000 \\ 0.000 & 0.010 \end{bmatrix}$	[-0.200, -0.200]	$\begin{bmatrix} 0.010 & 0.000 \\ 0.000 & 0.010 \end{bmatrix}$

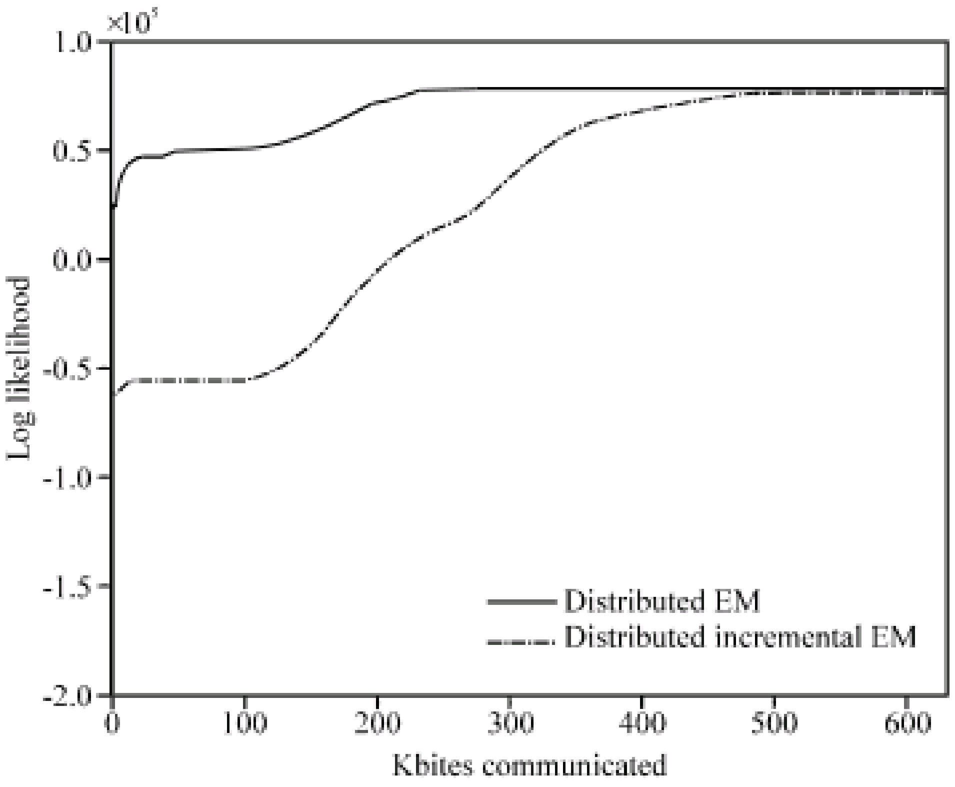

Fig. 2: Log-likelihood values of DIEM and DEM algorithms

Figure 2 shows log-likelihood values of DIEM and DEM algorithms as a function of number of transmitted bits. Number of transmitted bits corresponds to number of messages passed between nodes. As it is seen in Fig. 2, convergence rate of DIEM is much faster than that of DEM. Here, the convergence threshold is assumed to be ε = 0.1. At this simulation, the DIEM algorithm has converged after 724 iterations while DEM has reached the same log-likelihood value after 367 iterations. Computational cost of DIEM is also considerably less than that of the DIEM algorithm. DEM has converged in 153.12 sec while the DIEM has converged in 79.45 sec. Other simulations have shown similar results. Present experiments are performed on a 1.86 GHz dual-core Intel CPU with enough random access memory (RAM) to avoid paging.

Note that when the algorithms are converged, all nodes of the network have relatively the same values of estimated Gaussian mixture parameters which can be reached using any node of the network.

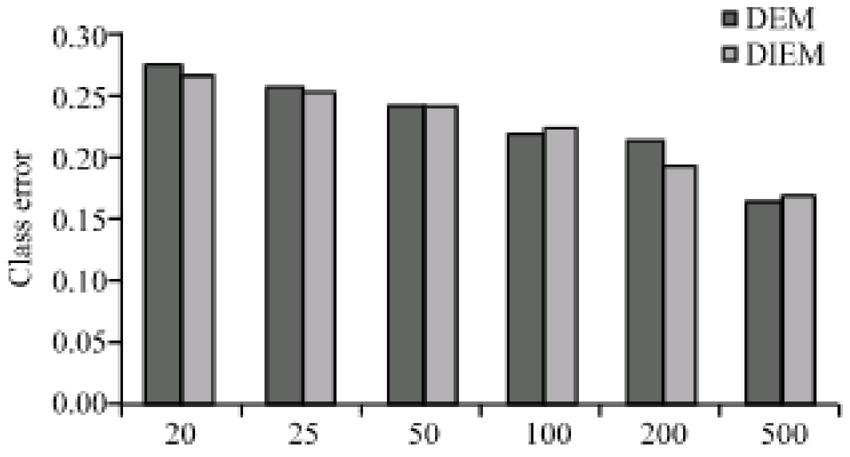

Fig. 3: Average misclassification rates versus sample size of each node (Std = 0.75)

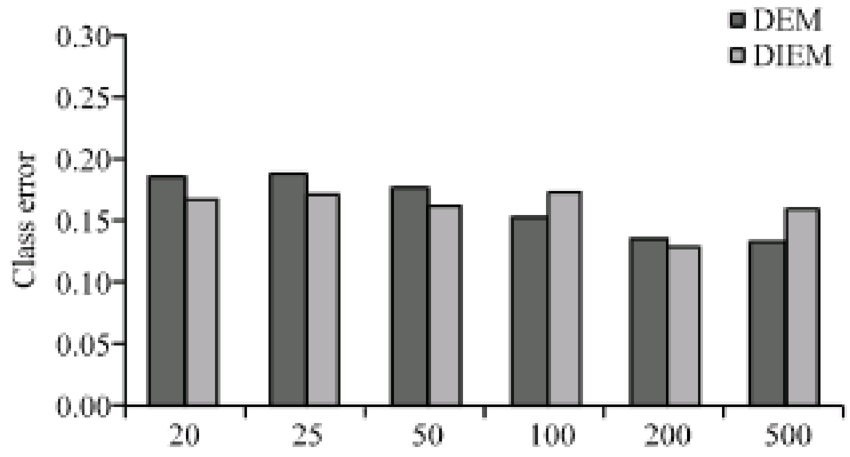

Fig. 4: Average misclassification rates versus sample size of each node (Std = 0.5)

In the next simulation, a multivariate model is considered in which the underlying clusters have the same covariance structure. Here, a model is considered in which clusters are spherical. This data set is analyzed for a range of different degrees of separation of the clusters, known as 'c-separation', as defined by Dasgupta (1999). Three different cases was considered in which Gaussian components are c-separated with c<2 (linearly separable), two-separated (almost linearly separable) and c-separated with c<2 (overlapping). The synthetic data was generated by fixing, without loss of generality, the centers to be at (0,0) and (1,1) and considering a range of different standard deviations (SD = 0.75, 0.5, 0.25). Thus, a model with SD = 0.75 corresponds to c = 4/3(<2), which indeed indicates the case of overlapping clusters (Dasgupta, 1999). Finally, we consider a wide range of sample sizes at each node that are typical for the current and future microarray studies.

A network of 20 nodes is considered in this study. To ensure robustness of the proposed methods, all simulated data were randomly generated 5 times and the resulting misclassification rates recorded. Misclassification rates obtained using the DEM and DIEM algorithms for the above mentioned three cases are shown in Fig. 3, 4 and 5. These figures show that both DEM and DIEM algorithms possess small misclassification rates. In other words, these algorithms were able to cluster the gene-expression data efficiently using the proposed distributed clustering methods.

In the next simulation, performance of DIEM is studied based on different K values and various gene-expression data dimensions. A network with 100 nodes (M = 100) is considered in which each node has 1000 data observations (N_m = 1000). The observations are generated from two Gaussian components. Here, we consider the case in which observations of different nodes do not come evenly from the two components. In the first 40 nodes, 30% of observations come from the first Gaussian component and other 70% of observations come from the other Gaussian component. In the next 30 nodes, observations come evenly from the two components. In the last 30 nodes, 70% of observations come from the first component and other 30% of observations come from the second Gaussian component.

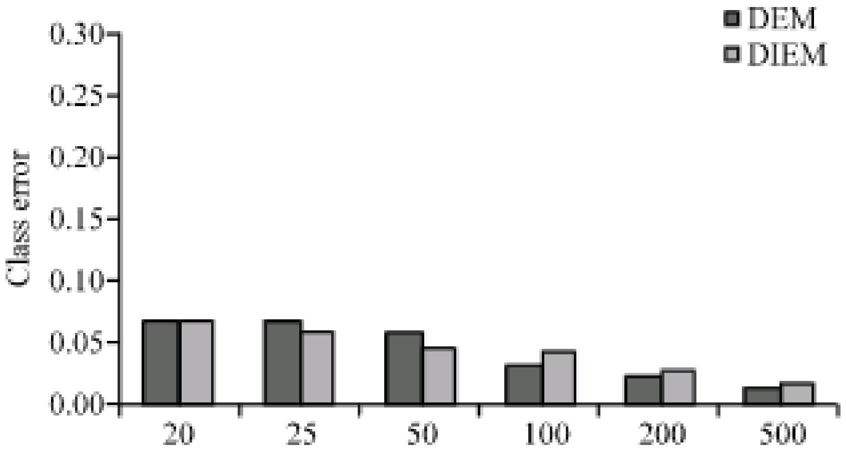

Fig. 5: Average misclassification rates versus sample size of each node (Std = 0.25)

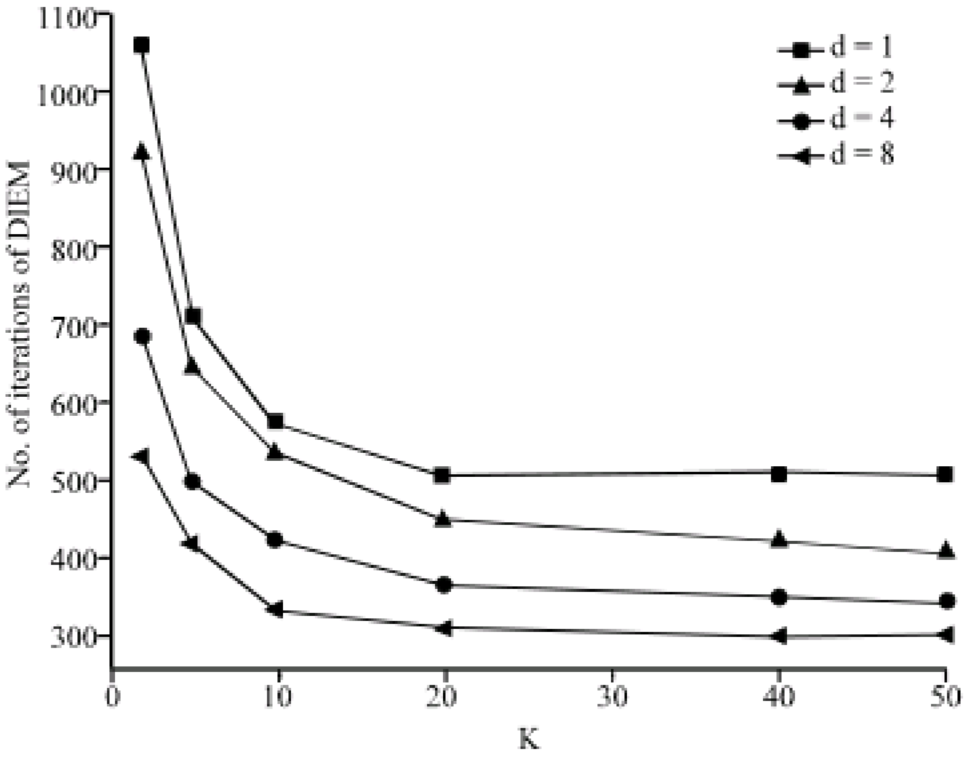

Fig. 6: No. of iterations of the DIEM algorithm for four data sets and different K values

Figure 6 shows number of iterations of the DIEM algorithm required to reach convergence as a function of number of data blocks (denoted by K). Each curve in Fig. 6 represents a particular data set with dimension d for d = 1, 2, 4, 8. As it is seen, by increasing the number of data blocks, number of iterations will decrease. In other

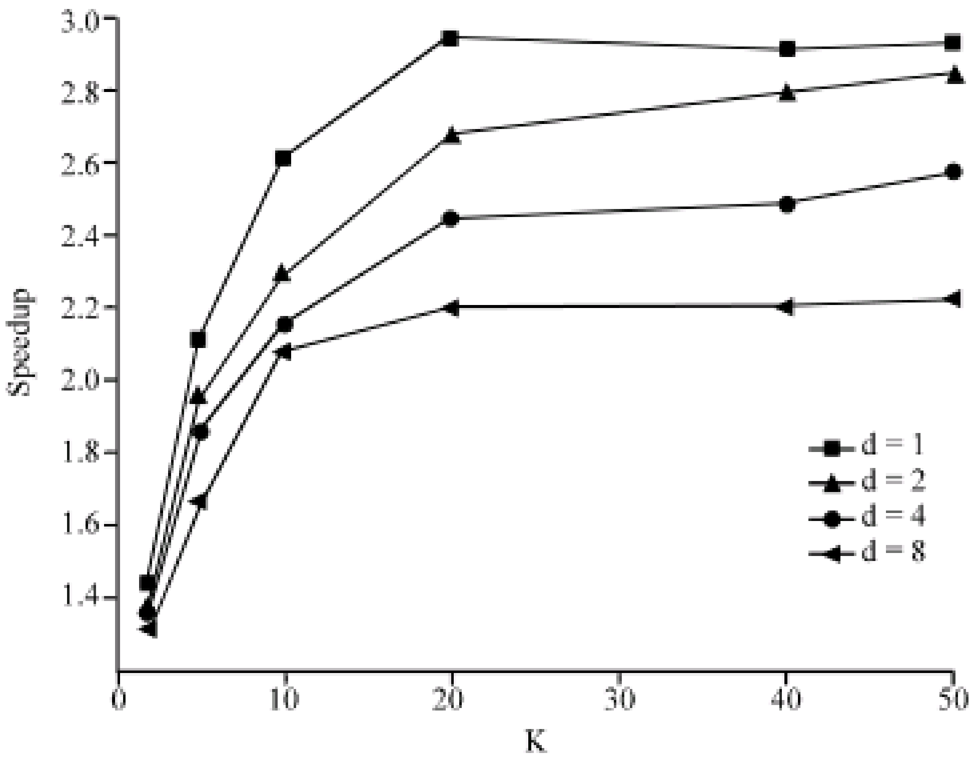

Fig. 7: Speedup factor obtained for different K values

words, increasing number of data blocks results in increasing the convergence rate. This result is valid for all of data sets.

In order to compare computational cost of the algorithms as well as their convergence rate, a speedup factor is defined here. A speedup factor is computed as the elapsed time for the DIEM algorithm to reach convergence divided by the elapsed time for DEM to reach convergence. Thus, a speedup factor greater than 1 means the algorithm improves performance. In Fig. 7, the speedup factor is shown for earlier data sets and different data blocks. As it is seen, increasing the value of K results in the speedup factor increasing. The results shown in these Fig. 6, 7 are the average value of the results obtained through 5 different runs of the DIEM algorithm.

CONCLUSION

In this study, a distributed incremental EM algorithm was proposed for density estimation and clustering of data distributed over the nodes of a network. A general distributed EM algorithm was first introduced. A distributed incremental EM algorithm was then proposed with a faster convergence rate. In this method, the data set of each node is partitioned into disjoint blocks of data and partial E-steps are performed on these blocks. Convergence of both DEM and DIEM was also analyzed based on the negative free energy concept. It was shown that these algorithms increase the negative free energy incrementally at each node until reaching the convergence.

As future study, lazy EM algorithm (Thiesson *et al.*, 2001) can be used to improve the DEM algorithm and reduce the computation cost at each node. Another field of research is how to choose initial values of the mixture parameters. In the proposed methods, initial values of the parameters are chosen randomly. Distributed k-means clustering may be used to choose more proper values for these parameters.

In the algorithms proposed here, E-step of the EM algorithm is performed in a cyclic distributed approach. Other noncyclic structures may also be used. For instance, methods have been developed for gossip-based randomized distributed sum or average calculation (Kempe *et al.*, 2003; Mehyar *et al.*, 2007). These methods may be used to develop distributed EM algorithms in which the E-step is performed in a different communication structure. These items are currently under investigation and will be reported later.

REFERENCES

Besg, 1986. On the statistical analysis of Dirty Pictures. J. R. Stat. Soc. Series B, 48: 259-302.

Chernyi, A.A., K.A. Trushkin, V.A. Bokovoy, A.K. Yanovski and N.V. Tverdokhlebov *et al.*, 2004. A system for distributed storage and analysis of genome information. Mol. Biol., 38: 89-93.

Dasgupta, S., 1999. Learning mixtures of gaussians. Proceedings of the 40th Annual Symposium on Foundations of Computer Science, Oct. 17-19, IEEE Computer Society, New York, pp: 634-644.

Datta, S., K. Bhaduri, C. Giannella, R. Wolff and H. Kargupta, 2006. Distributed data mining in peer-to-peer networks. IEEE Internet Comput., 10: 18-26.

Fraley, C. and A.E. Raftery, 1998. How Many Clusters? Which Clustering Method? Answers via model-based cluster analysis. Comput. J., 41: 578-588.

Ghosh, D. and A.M. Chinnaiyan, 2002. Mixture modeling of gene expression data from microarray experiments. Bioinformatics, 18: 275-286.

Jiang, D., C. Tang and A. Zhang, 2004. Cluster analysis for gene expression data: A survey. IEEE T Knowl. Data Eng., 16: 1370-1386.

Kempe, D., A. Dobra and J. Gehrke, 2003. Gossip-based computation of aggregate information. Gossip-based computation of aggregate information. Oct. 11-14, Cambridge, MA., USA., pp: 482-491.

Kowalczyk, W. and N. Vlassis, 2005. Newscast EM. Advances in Neural Information Processing Systems, 17, MIT Press. http://books.nips.cc/papers/files/nips17/NIPS2004_0460.pdf.

Lin, X., C. Clifton and M. Zhu, 2005. Privacy-preserving clustering with distributed EM mixture modeling. Knowl. Inform. Syst., 8: 68-81.

Ma, J., L. Xu and M.I. Jordan, 2000. Asymptotic convergence rate of the EM algorithm for Gaussian Mixtures. Neural Comput., 12: 2881-2907.

McLachlan, G.J. and T. Krishnan, 1997. The EM Algorithm and Extensions. 1st Edn., Wiley, New York, ISBN: 9780471201700.

McLachlan, G. and D. Peel, 2000. Finite Mixture Models. 1st Edn., Wiley, New York, ISBN: 0471006262.

McLachlan, G.J., R.W. Bean and D. Peel, 2002. A mixture model-based approach to the clustering of microarray expression data. Bioinformatics, 18: 413-422.

Mehyar, M., D. Spanos, J. Pongsajapan, S.H. Low and R.M. Murray, 2007. Asynchronous distributed averaging on communication networks. IEEE/ACM Trans. Network., 15: 512-520.

Neal, R. and G. Hinton, 1999. A View of the EM Algorithm that Justifies Incremental, Sparse and Other Variants. In: Learning in Graphical Models, Michael I. Jordan (Ed.). MIT Press, Cambridge, MA., ISBN: 0-262-60032-3, pp: 355-368.

Nowak, R.D., 2003. Distributed EM algorithms for density estimation and clustering in sensor networks. IEEE Trans. Signal Process., 51: 2245-2253.

Ordonez, C. and E. Omiecinski, 2005. Accelerating EM clustering to find high-quality solutions. Knowl. Inform. Syst., 7: 135-157.

Stratowa, C., 2003. Distributed storage and analysis of microarray data in the terabyte range: An alternative to bioconductor. Proceeding of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003), Mar. 20-22, Vienna, Austria, pp: 1-21.

Thiesson, B., C. Meek and D. Heckerman, 2001. Accelerating EM for large databases. Mach. Learn., 45: 279-299.

Verbeek, J.J., N. Vlassis and J.R.J. Nunnink, 2003. A variational EM algorithm for large-scale mixture modeling. Proceedings of 8th Annual Conference the Advanced School of Computing and Imaging, (ASCI'03), Heijen, The Netherlands, pp: 1-7.

Wolff, R. and A. Schuster, 2004. Association rule mining in peer-to-peer systems. IEEE Trans. Syst. Man Cybernet. Part B., 34: 2426-2438.

Xu, L. and M.I. Jordan, 1996. On convergence properties of the EM algorithm for Gaussian Mixtures. Neural Comput., 8: 129-151.

Yeung, K.Y., C. Fraley, A. Murua, A.E. Raftery and W.L. Ruzz, 2001. Model-based clustering and data transformation for gene expression data. Bioinformatics, 17: 977-987.