Egyptian Informatics Journal (2010) **11**, 49–57

Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

Using general regression with local tuning for learning mixture models from incomplete data sets

Ahmed R. Abas

Department of Computer Science, Faculty of Computers and Informatics, Zagazig University, Zagazig, Egypt

Received 4 May 2010; accepted 24 June 2010
Available online 2 November 2010

KEYWORDS

Finite mixture models;
Expectation Maximization;
Missing value;
Incomplete data set;
Locally tuned general regression

Abstract Finite mixture models is a pattern recognition technique that is used for fitting complex data distributions. Parameters of this mixture models are usually determined via the Expectation Maximization (EM) algorithm. A modified version of the EM algorithm is proposed earlier to handle data sets with missing values. This algorithm is affected by the occurrence of outliers in the data, the overlap among classes in the data space and the bias in generating the data from its classes. In addition, it only works well when the missing value rate is low. In this paper, a new algorithm is proposed to overcome these problems. A comparison study shows the superiority of the new algorithm over the modified EM algorithm and other algorithms commonly used in the literature.

1. Introduction

Finite mixture models (FMM) is a semi-parametric method for density estimation and pattern recognition [1]. It has the advantage of the analytic simplicity of the parametric methods and the advantage of the flexibility to model complex data distributions of the non-parametric methods [2]. Parameters of FMM are usually estimated by the Expectation Maximization (EM) algorithm [3]. It is shown that the EM algorithm can estimate parameters of a multivariate normal distribution from a data set with missing values [4]. The EM algorithm is modified to estimate parameters of a mixture of multivariate normal distributions using incomplete data set [5]. The modified EM algorithm is used in clustering data sets that contain missing values and a number of categorical features [6,7]. Also, it is used in learning parameters of the radial basis function networks used in classifying data sets that contain missing values [8]. When there is sufficiently large number of observed values, this algorithm outperforms the EM algorithm combined with either the unconditional mean imputation [9,10], or the conditional mean imputation [11], of the missing values according to the classification performance of the resulting FMM. Also, this algorithm outperforms the EM algorithm combined with deleting feature vectors that contain missing values [11], according to the classification performance of the resulting FMM.

E-mail address: arabas@zu.edu.eg

1110-8665 © 2010 Faculty of Computers and Information, Cairo University. Production and hosting by Elsevier B.V. All rights reserved.

Peer review under responsibility of Faculty of Computers and Information, Cairo University.
doi:10.1016/j.eij.2010.10.001

Production and hosting by Elsevier

However, the outcome of the modified EM algorithm is influenced by several modelling assumptions such as the number of components and the probability distribution function of each component [11,12]. For example, the modified EM algorithm produces an inaccurate estimation of both FMM parameters and missing values in the input data set when initialised with an FMM that has too few components, each of which has a probability distribution function that does not fit well any of the clusters of the input data set. In addition, this algorithm has a poor performance when the size of the data set is small [11]. It is shown that better results can be obtained by imputing missing values using the distribution of the input feature vectors rather than using a priori probability distribution function used in the FMM [11]. The modified EM algorithm [5] is slightly changed to reduce computational burden during the EM iterations by incorporating two types of auxiliary binary indicator matrices corresponding to the observed and unobserved components of each datum [13]. The resulting EM algorithm is compared with a novel Data Augmentation (DA) computational algorithm for learning normal mixture models when the data are missing at random [13]. Experimental results show that DA imputation exhibits considerable promising accuracy in the prediction of missing values when compared to the EM imputation, especially as the missing rate increases. However, both algorithms impute missing values using mixture model parameters and hence their imputations are sensitive to the prior information about density functions of mixture components and the size of data that are fully observed. A supervised classification method, called robust mixture discriminant analysis (RMDA), is proposed to deal with label noised data [14]. The proposed algorithm uses only fully observed data to learn mixture model parameters and then uses this mixture model to estimate labels and detect noisy ones. However, imputations made by this algorithm are sensitive to prior information about density functions of mixture components, the size of data that are fully observed and assumptions such as all the uncertain labels are in one feature.

In this paper, a new algorithm is proposed to overcome problems of the modified EM algorithm [5,13,14]. The proposed algorithm is less sensitive to the modelling assumptions than the modified EM algorithm in estimating missing values. In addition, it is less affected by learning problems than the EM algorithm in cases such as the occurrence of outliers in the data set and the overlap among data classes.

2. The general regression neural network

The general regression neural network (referred to as GRNN in the rest of this paper) implements a non-parametric algorithm for density estimation and general (non-linear) regression [15]. This algorithm has a better regression performance than conventional non-linear regression techniques, particularly with a sparse data set that has a small number of feature vectors and a large number of features. This is because the regression surface resulting from the GRNN is defined everywhere in the data space [15]. For example, when the input data set is sparsely distributed, the smoothing parameter (window size) used in the GRNN can be large in order to estimate the regression value at any new point with similar accuracy to the other points. On the other hand, conventional non-linear regression techniques tend to overfit this data set, and therefore there is no assurance that the error in estimating the regression value for any new point will be small. Similar algorithms to the GRNN are used to train neural networks, in which the hidden neurons use Gaussian density functions, using data sets that contain missing values [16,17]. These algorithms are used for regression and classification and they exhibit better performances than neural networks that are trained using either only the complete portion of the input data set or the total data set after estimating its missing values using the unconditional mean imputation method.

Suppose that $\mathscr{R} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\mathrm{N}\}$ is a data set that contains N feature vectors each of which is a vector in d-feature space such that $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{i(\text{d-1})}, y_i]^T$, where the first (d-1)-features are complete and the d-th feature contains missing values. The GRNN algorithm imputes the missing value in $\mathbf{x}_i$ by computing the weighted average of all the observed values on the d-th feature such that each observed value is weighted exponentially according to its Euclidean distance from $\mathbf{x}_i$ in the fully observed subspace.

$$\hat{y}_i(\mathbf{x}_i) = \frac{\sum_{j=1}^{\mathrm{n}_o} y_j \exp\left(-\frac{D_j^2}{2\sigma^2}\right)}{\sum_{j=1}^{\mathrm{n}_o} \exp\left(-\frac{D_j^2}{2\sigma^2}\right)} \tag{1}$$

where n_o is the number of feature vectors that are fully observed, $D_j^2 = \sum_{h=1}^{\mathrm{d}-1} (\mathbf{x}_{ih} - \mathbf{x}_{jh})^2$ is the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$, and σ is the smoothing parameter for the d-th feature. To identify the optimum value of σ, the GRNN uses the leave-one-out cross validation method. The optimum σ is then used in imputing all the missing values on the d-th feature.

However, as a non-linear regression technique the GRNN algorithm produces good results when the data set has a large number of features that are strongly correlated [18–20]. Also, it requires the number of missing values to be small to accelerate the algorithm and to produce unbiased imputations [18,4,15,20]. Therefore, this algorithm is inappropriate for a small data set, which has a small number of feature vectors and a small number of features, especially when several features contain missing values and the correlations between these features and the other features in the data set are weak.

In this paper, the GRNN algorithm is used to fill in the missing values in a certain data set to be used by the EM algorithm for learning parameters of the FMM. The resulting algorithm is referred to as the GREM algorithm in the rest of this paper and it is used in the comparison study below to investigate the performance of the proposed algorithm against other algorithms.

3. The modified EM algorithm

The EM algorithm is modified to estimate FMM parameters from a data set with missing values [5]. This algorithm is referred to as the MEM algorithm in this paper. In this algorithm, the missing values in the input data set and some other statistics are estimated in the E-step from the parameters of each model component. These values together with the observed values in the data set are then used in the M-step to estimate the FMM parameters. This means that in the E-step multiple imputations are obtained for each missing value in the input data set from the different components in the FMM and then these imputations are used in the M-step to estimate the FMM parameters. The following paragraphs give a more detailed explanation of the MEM algorithm.

Suppose that the data set $\mathscr{R} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\mathrm{N}\}$ consists of N feature vectors each of which is a vector in d-feature space such that each feature vector $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{i\mathrm{d}}]^T$. This data set is assumed to be generated from a FMM of K multivariate normal distributions with unknown mixing coefficients $P(c)$, where $\sum_{c=1}^{\mathrm{K}} P(c) = 1$, and $0 \leqslant P(c) \leqslant 1$. Let the probability density of the feature vector $\mathbf{x}_i$, which is fully observed, given the kth component in the FMM be $p(\mathbf{x}_i|\boldsymbol{\theta}_k) = N(\mathbf{x}_i; \boldsymbol{\mu}_k, \Sigma_k)$, where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are the mean and the covariance matrix of this component. The total density of $\mathbf{x}_i$ from the FMM is then computed as $p(\mathbf{x}_i) = \sum_{c=1}^{\mathrm{K}} P(c)p(\mathbf{x}_i|\boldsymbol{\theta}_c)$. In fitting the FMM, there are two types of missing values that have to be considered; the first type is the values of the cluster membership vector for each feature vector $\mathbf{z}_i = [z_{i1}, z_{i2}, \ldots, z_{i\mathrm{K}}]^T$; the second type is the missing values in the different features of $\mathscr{R}$. To represent the second type let each feature vector in $\mathscr{R}$ be rewritten as $\mathbf{x}_i = (\mathbf{x}_i^o, \mathbf{x}_i^m)$, where o and m superscripts denote the observed and the missing values in this feature vector, respectively.

In the E-step of the MEM algorithm, all $\mathbf{z}_i$s are approximated by the posterior probabilities using the observed values for each feature vector $\mathbf{x}_i$ and the parameters of each component in the FMM as follows.

$$\hat{z}_{ic} = \frac{\widehat{P}(c)p(\mathbf{x}_i^o|\boldsymbol{\theta}_c)}{\sum_{j=1}^{\mathrm{K}} \widehat{P}(j)p(\mathbf{x}_i|\boldsymbol{\theta}_j)} \tag{2}$$

Let the mean and the covariance matrix for each component in the FMM be partitioned in the same manner as the feature vector $\mathbf{x}_i$ with o and m superscripts denoting the components corresponding to observed and missing features for this feature vector.

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}^o \\ \boldsymbol{\mu}^m \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}^{oo} & \boldsymbol{\Sigma}^{om} \\ \boldsymbol{\Sigma}^{mo} & \boldsymbol{\Sigma}^{mm} \end{pmatrix} \tag{3}$$

Then the expectation of the missing values in the feature vector $\mathbf{x}_i$ given the parameters of the component c is computed as follows.

$$E(\mathbf{x}_i^m|\mathbf{x}_i^o, \boldsymbol{\theta}_c) = \boldsymbol{\mu}_c^m + \boldsymbol{\Sigma}_c^{mo}\boldsymbol{\Sigma}_c^{oo^{-1}}(\mathbf{x}_i - \boldsymbol{\mu}_c^o) \tag{4}$$

Also, the residual covariances of the estimated values for this feature vector are computed as follows.

$$\mathrm{Cov}(\mathbf{x}_i^m\mathbf{x}_i^{m^T}|\mathbf{x}_i^o, \boldsymbol{\theta}_c) = \boldsymbol{\Sigma}_c^{mm} - \boldsymbol{\Sigma}_c^{mo}\boldsymbol{\Sigma}_c^{oo^{-1}}\boldsymbol{\Sigma}_c^{mo^T} \tag{5}$$

Also, some statistical moments necessary for each model component in the M-step are computed as follows.

$$E(z_{ic}x_{ij}|\mathbf{x}_i^o, \boldsymbol{\theta}_c) = \begin{cases} \hat{z}_{ic}x_{ij} & x_{ij} \in \mathbf{x}_i^o \\ \hat{z}_{ic}E(x_{ij}|\mathbf{x}_i^o, \boldsymbol{\theta}_c) & x_{ij} \in \mathbf{x}_i^m \end{cases} \tag{6}$$

$$E(z_{ic}x_{ij}x_{ij'}|\mathbf{x}_i^o, \boldsymbol{\theta}_c) = \begin{cases} \hat{z}_{ic}x_{ij}x_{ij'} & x_{ij}, x_{ij'} \in \mathbf{x}_i^o \\ \hat{z}_{ic}x_{ij}E(x_{ij'}|\mathbf{x}_i^o, \boldsymbol{\theta}_c) & x_{ij'} \in \mathbf{x}_i^m \\ \hat{z}_{ic}E(x_{ij}|\mathbf{x}_i^o, \boldsymbol{\theta}_c)x_{ij'} & x_{ij} \in \mathbf{x}_i^m \\ \hat{z}_{ic}[E(x_{ij}|\mathbf{x}_i^o, \boldsymbol{\theta}_c)E(x_{ij'}|\mathbf{x}_i^o, \boldsymbol{\theta}_c) & \\ +\mathrm{Cov}(x_{ij}, x_{ij'}|\mathbf{x}_i^o, \boldsymbol{\theta}_c)] & x_{ij}, x_{ij'} \in \mathbf{x}_i^m \end{cases} \tag{7}$$

where $i, j, j' = 1, 2, \ldots, \mathrm{d}$.

In the M-step of the MEM algorithm, the new estimates of the FMM parameters are computed as follows.

$$\widehat{P}(c) = \frac{1}{\mathrm{N}}\sum_{j=1}^{\mathrm{N}} \hat{z}_{jc} \tag{8}$$

$$\hat{\boldsymbol{\mu}}_c = \frac{1}{\mathrm{N}\widehat{P}(c)} E\left(\sum_{j=1}^{\mathrm{N}} z_{jc}\mathbf{x}_j|\mathbf{x}_j^o, \boldsymbol{\theta}_c\right) \tag{9}$$

$$\hat{\boldsymbol{\Sigma}}_c = \frac{1}{\mathrm{N}\widehat{P}(c)} E\left(\sum_{j=1}^{\mathrm{N}} z_{jc}\mathbf{x}_j\mathbf{x}_j^T|\mathbf{x}_j^o, \boldsymbol{\theta}_c\right) - \hat{\boldsymbol{\mu}}_c\hat{\boldsymbol{\mu}}_c^T \tag{10}$$

Both of the E-step and the M-step are repeated until the algorithm converges to a local maximum of the likelihood function.

4. Comparing the MEM algorithm with the GREM algorithm in unsupervised learning of FMM parameters

The same notation that is introduced in Section 3 is used in this section. To simplify the comparison, it is assumed that the data set $\mathscr{R}$ is generated from a mixture model whose components are compact and well separated in the data space. Therefore, each feature vector in the data set belongs to only one component in the FMM to which its posterior probability is 1. In addition, it is assumed that each component in the FMM that needs to be learned from the data set $\mathscr{R}$ has a spherical Gaussian distribution. Finally, it is assumed that there is sufficiently large number of observed values in the data set $\mathscr{R}$ to learn the FMM parameters accurately.

In the MEM algorithm, the missing values in the feature vector $\mathbf{x}_i$ are imputed, given the parameters of the component c in the FMM, as shown in Eq. (4). Since all components in the FMM have spherical shapes then all the values in $\boldsymbol{\Sigma}_c^{mo}$ are zeros. Therefore, Eq. (4) can be rewritten as follows.

$$E(\mathbf{x}_i^m|\mathbf{x}_i^o, \boldsymbol{\theta}_c) = \boldsymbol{\mu}_c^m \tag{11}$$

Since the posterior probability of each feature vector in $\mathscr{R}$ is 1 for precisely one component in the FMM then Eq. (9) can be rewritten as follows.

$$\boldsymbol{\mu}_c^m = \frac{1}{\mathrm{N}_c} E\left(\sum_{j=1}^{\mathrm{N_c}} \mathbf{x}_j^m|\mathbf{x}_j^o, \boldsymbol{\theta}_c\right) \tag{12}$$

where N_c is the number of feature vectors that belong to the component c.

On the other hand, the GREM algorithm imputes each missing value $\widehat{x}_{iq}(\mathbf{x}_i^o)$ in the feature vector $\mathbf{x}_i$ according to Eq. (1). This equation can be rewritten as follows:

$$\hat{x}_{iq}(\mathbf{x}_i^o) = \frac{\sum_{h=1}^{\mathrm{K}}\sum_{j=1}^{n_o^{\mathrm{h}}} x_{jq}\exp\left(-\frac{D_j^2}{2\sigma_q^2}\right)}{\sum_{h=1}^{\mathrm{K}}\sum_{j=1}^{n_o^{\mathrm{h}}} \exp\left(-\frac{D_j^2}{2\sigma_q^2}\right)} \tag{13}$$

where n_o^h is the number of the feature vectors that belong to the component h in the FMM, and they are fully observed on both the fully observed subspace for the feature vector $\mathbf{x}_i$ and the qth feature. D_j^2 is the squared Euclidean distance on the observed subspace of the $\mathbf{x}_i$ between this feature vector and the feature vector $\mathbf{x}_j$ that belongs to component h.

Let the correlations between different pairs of features of $\mathscr{R}$ be strong such that feature vectors that belong to one cluster are near to each other in all subspaces of the data space. Since $\mathscr{R}$ is generated from a mixture of compact and well-separated clusters in the data space then, in all subspaces of $\mathscr{R}$ the distances between feature vectors that belong to the same cluster are small compared with the distances between feature vectors that belong to different clusters. Therefore, the exponential weight in Eq. (13) of a certain feature vector $\mathbf{x}_j$ is approximately one, if $\mathbf{x}_j$ and $\mathbf{x}_i$ belong to the same cluster, and is zero otherwise. Then Eq. (13) can be rewritten as follows.

$$\hat{x}_{iq}(\mathbf{x}_i^o) \cong \frac{1}{n_o^c}\sum_{j=1}^{n_o^c} x_{jq} = \hat{\mu}_{cq} \tag{14}$$

Eq. (14) is then generalized as follows.

$$\hat{\mathbf{x}}_i^m(\mathbf{x}_i^o) \cong \hat{\boldsymbol{\mu}}_c^m \tag{15}$$

Eqs. (11) and (15) show that the GREM algorithm and the MEM algorithm are similar in their estimation of the missing values when there are strong correlations between different pairs of features of $\mathscr{R}$.

Let the correlations between different pairs of features of $\mathscr{R}$ be weak such that feature vectors that belong to different clusters can be near to each other in all subspaces of $\mathscr{R}$. Then, the estimation of the missing values in the GREM algorithm, which is produced by Eq. (13), is approximately the average of all the observed values on the qth feature. This affects the learning of the FMM parameters such that the resulting components will be overlapping in the whole space. Therefore using this mixture model for clustering cannot produce compact or well-separated clusters. On the other hand, the estimation of the missing values in the MEM algorithm, which is produced by Eq. (11), results in different imputations for each missing value from the parameters of the different components in the FMM. When used in estimating parameters of each component in the FMM, these different imputations preserve the features of these components that are learned from the fully observed values in $\mathscr{R}$. Therefore, using the resulting FMM for clustering produces compact and well-separated clusters. In general, when the correlations between different pairs of features of $\mathscr{R}$ are weak the MEM algorithm outperforms the GREM algorithm with respect to the clustering performance of the resulting FMM.

5. The proposed algorithm for learning FMM parameters

It is argued in the previous section that the MEM algorithm is less sensitive to the correlations between different pairs of features of the data set than the GREM algorithm. However, the performance of the MEM algorithm in estimating FMM parameters is severely distorted when the size of the input data set is small [11,21]. For example, when the input data set has a small size and it contains a number of outlier feature vectors, an overlap among its clusters, or a large difference in the sizes of its clusters the estimates produced by the MEM algorithm for the FMM parameters are largely biased and inaccurate.

In this section, a new algorithm is proposed to overcome problems of both the GREM algorithm and the MEM algorithm when dealing with a small incomplete data set that may contain outliers, overlapped clusters, or a large difference in the sizes of its clusters. These problems are the sensitivity to global correlations between features of the input data set, and the dependence on FMM parameters learned from this data set. The proposed algorithm is referred to as the Locally-tuned General Regression with the Expectation Maximization (LGREM) algorithm.

5.1. Description of the LGREM algorithm

In this section, the same notation that is introduced in Section 3 is used.

Step 1: Linearly scale the values of each feature in the input data set to make them lie in the interval [0,1]. This process removes the effect of different unit scales on different features, and therefore it is essential for finding the true cluster structure of this data set and the contribution of each feature in the creation of this structure. A comparison of several methods of data normalization for cluster analysis has shown the superiority of the linear scaling method over many other normalization methods before applying clustering algorithms in terms of the resulting cluster separation and error condition [22,23]. In addition, determine the optimum smoothing parameter σ for each incomplete feature in the data set using the leave-one-out cross validation method. The group of fully observed features used in determining σ for a certain feature consists of both the complete features and the features that have smaller missing rates than this feature. The feature vectors that are used in the leave-one-out cross validation method should be observed in the entire fully observed feature group.

Step 2: As the EM algorithm converges toward the nearest local maximum of the likelihood function to the starting point [24], initialise the EM algorithm randomly several times (20 times in our experiments) and select the FMM corresponding to the maximum log-likelihood function after convergence as the best model for the input data set.

Step 3: In the E-step, compute the following quantities for each model component c in the FMM.

The posterior probabilities vector $\mathbf{z}_i$ for all feature vectors in the data set $\mathscr{R}$.

$$\hat{z}_{ic} = \frac{\widehat{P}(c)p(\mathbf{x}_i^o|\boldsymbol{\theta}_c)}{\sum_{j=1}^{\mathrm{K}}\widehat{P}(j)p(\mathbf{x}_i^o|\boldsymbol{\theta}_j)} \tag{16}$$

- The estimates of the missing values in $\mathscr{R}$ starting with those values in the feature that has the minimum missing rate.

$$E\left(\widehat{x}_{iq}^c|\mathbf{x}_i^0, \mathbf{R}\right) = \frac{\sum_{k=1}^{n_0} x_{kq}\exp\left(\frac{D^2}{2\sigma_q^2}\right)r_{kc}}{\sum_{k=1}^{n_0}\exp\left(\frac{D^2}{2\sigma_q^2}\right)r_{kc}} \tag{17}$$

where $\mathbf{R} = \{r_{kc}\}$ is the matrix of memberships for each feature vector $\mathbf{x}_k$ to each component c in the FMM such that r_{kc} is either one, if $\widehat{z}_{kc}\widehat{z}_{kt}$ for all $t \neq c$, or zero otherwise, n_o is the number of feature vectors that are observed on both of the observed subspace for the feature vector $\mathbf{x}_i$ and the qth feature, and D_k^2 is the squared Euclidean distance between feature vectors $\mathbf{x}_k$ and $\mathbf{x}_i$ on the observed subspace of the feature vector $\mathbf{x}_i$.

After estimating its missing values, the qth feature is added to the group of fully observed features and this new group is then used in estimating the missing values in the next feature that has the minimum missing rate.

- The necessary statistics for the M-step.

$$E(z_{ic}x_{iq}|\mathbf{x}_i^o,\mathbf{R}) = \begin{cases} \hat{z}_{ic}x_{iq} & x_{iq} \in \mathbf{x}_i^o \\ \hat{z}_{ic}E(x_{iq}^c|\mathbf{x}_i^o,\mathbf{R}) & x_{iq} \in \mathbf{x}_i^m \end{cases} \tag{18}$$

$$E(z_{ic}x_{iq}x_{iq'}|\mathbf{x}_i^o,\mathbf{R}) = \begin{cases} \hat{z}_{ic}x_{iq}x_{iq'} & x_{iq}, x_{iq'} \in \mathbf{x}_i^o \\ \hat{z}_{ic}x_{iq}E(x_{iq'}^c|\mathbf{x}_i^o,\mathbf{R}) & x_{iq'} \in \mathbf{x}_i^m \\ \hat{z}_{ic}E(x_{iq}^c|\mathbf{x}_i^o,\mathbf{R})x_{iq'} & x_{iq} \in \mathbf{x}_i^m \\ \hat{z}_{ic}E(x_{iq}^c|\mathbf{x}_i^o,\mathbf{R})E(x_{iq'}^c|\mathbf{x}_i^o,\mathbf{R}) & x_{iq}, x_{iq'} \in \mathbf{x}_i^m \end{cases} \tag{19}$$

where $i, q, q' = 1, 2, \ldots, \mathrm{d}$, E is the expectation operator.

Step 4: In the M-step, compute parameters of each component c in the FMM.

$$\widehat{P}(c) = \frac{1}{\mathrm{N}}\sum_{j=1}^{\mathrm{N}} \hat{z}_{jc} \tag{20}$$

$$\widehat{\mu}_c = \frac{1}{\mathrm{N}\,\widehat{P}(c)} E\left(\sum_{j=1}^{\mathrm{N}} z_{jc}\mathbf{x}_j|\mathbf{x}_j^o,\mathbf{R}\right) \tag{21}$$

$$\widehat{\boldsymbol{\Sigma}}_c = \frac{1}{\mathrm{N}\,\widehat{P}(c)} E\left(\sum_{j=1}^{\mathrm{N}} z_{jc}\mathbf{x}_j\mathbf{x}_j^T|\mathbf{x}_j^o,\mathbf{R}\right) - \widehat{\boldsymbol{\mu}}_c\widehat{\boldsymbol{\mu}}_c^T \tag{22}$$

Step 5: After convergence, save the resulting FMM and the total data log-likelihood. Since the EM algorithm is a strict gradient algorithm, it converges slowly to the nearest local maximum of the likelihood function to the starting point [25]. Therefore, the convergence criterion of the EM algorithm used in our experiments is identical to the one used by Hunt and Jorgensen [7], which is to cease iterating when the difference in the log-likelihood function between iterations (t) and (t-10) is less than or equal 10^{-10}. The use of this criterion does not affect the speed of the algorithm so much because the main interest of this algorithm is to handle small data sets.

Step 6: Repeat *Step 2–5* several times and then select the best FMM that corresponds to the maximum data log-likelihood. This repetition is necessary to reduce the sensitivity to the initialisation of the EM algorithm. In our experiments, the number of repetitions is chosen arbitrarily to be 20.

Step 7: Use the best FMM for clustering feature vectors in the input data set according to Bayes decision rule such that each fully observed feature vector $\mathbf{x}$ is assigned to a certain component i if $P(i|\mathbf{x}) > P(j|\mathbf{x})$ for all $j{\neq}i$, where $P(i|\mathbf{x})$ is the probability that $\mathbf{x}$ is generated from the component i.

Step 8: Estimate missing values in the data set as explained in *Step 3*.

The LGREM algorithm uses local tuning of the general regression (see Eq. (17)) to produce multiple imputations for each missing value in the data set. Each imputation is obtained via a non-linear multivariate regression using a group of fully observed feature vectors belonging to one of the FMM components. These groups are obtained using Bayes decision rule. This helps the algorithm to be insensitive to global correlations between features of the input data set, and therefore to overcome the limitation of the GREM algorithm (see Eq. (13)) with weakly correlated data. In addition, it helps the algorithm to be less dependent of FMM parameters, and therefore to overcome the limitation of the MEM algorithm (see Eq. (4)) with small data sets, which may contain outliers, overlapped clusters, or large differences in the sizes of their clusters.

6. Experiments and results

The LGREM algorithm is evaluated and compared with a number of commonly used algorithms in the literature in unsupervised learning of FMM parameters using data sets with missing values. These algorithms are the MEM algorithm, the GREM algorithm and the common EM algorithm after estimating the missing values in the input data set using the unconditional mean imputation method (MENEM) and the nearest neighbour imputation method (NNEM). In the MENEM algorithm, the missing values in each feature are replaced with the mean of the observed values in that feature. In the NNEM algorithm, each missing value in a certain feature vector is replaced with the observed value in the same feature that is in the closest feature vector according to the Euclidean distance in the subspace that is composed of the fully observed features. Six data sets are used in this comparison study. The missing values are randomly placed with different missing rates in two features of each data set. These features are selected such that the visual separation among data classes is maximum. The mechanism of the occurrence of the missing values in all data sets is missing completely at random (MCAR) [4]. These data sets are described as follows.

6.1. The first and the second data sets

These data sets are artificial data sets, each of which contains 150 feature vectors (rows) generated with equal probabilities from three 4D-Gaussian distributions. The mean vectors of these distributions are $\boldsymbol{\mu}_1 = [2 \quad 2 \quad 2 \quad 2]^T$, $\boldsymbol{\mu}_2 = [4 \quad 4 \quad 4 \quad 4]^T$, and $\boldsymbol{\mu}_3 = [6 \quad 6 \quad 6 \quad 6]^T$ for the first data set, and $\boldsymbol{\mu}_1 = [2 \quad 2 \quad 2 \quad 2]^T$, $\boldsymbol{\mu}_2 = [2 \quad 2 \quad 6 \quad 2]^T$, and $\boldsymbol{\mu}_3 = [2 \quad 2 \quad 2 \quad 6]^T$ for the second data set. Each one of these Gaussian distributions has a covariance matrix $\boldsymbol{\Sigma} = \mathbf{I}_4$, where $\mathbf{I}_4$ is the identity matrix of order four. The difference between both data sets is the correlations between data features. In the first data set, all features are strongly correlated, while in the second data set all features are too weakly correlated.

6.2. The third and the fourth data sets

These data sets are similar to the first and the second data sets respectively besides an outlier feature vector is added to each one of them. The outlier feature vector in the third data set is $\left[\max(\mathbf{d}_1), \min(\mathbf{d}_2), \frac{\max(\mathbf{d}_3)}{2}, \frac{\max(\mathbf{d}_4)}{2}\right]^T$, while the outlier feature vector in the fourth data set is $\left[\frac{\max(\mathbf{d}_1)}{2}, \frac{\max(\mathbf{d}_2)}{2}, \max(\mathbf{d}_3), \max(\mathbf{d}_4)\right]^T$, where $\mathbf{d}_i$, $i = 1{:}4$ are the features of every data set.

6.3. The fifth data set

This data set is the Iris data set [26], which contains 150 feature vectors each of which is a vector in four-feature space. These

Table 1 Comparing different pairs of algorithms using the Student's paired *t*-test statistic with the first data set.

Missing (%)	T(GREM,MEM)		T(LGREM,GREM)		T(LGREM,MEM)		T(LGREM,NNEM)		T(LGREM,MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.34	1	0.34	−1	1	0	0.05	−2.236	0.02	−2.764
(10%, 20%)	0.17	1.5	0.17	−1.5	1	0	0.01	−3.207	1.8E−06	−10.834
(15%, 30%)	0.17	1.5	0.10	−1.809	0.34	−1	0.001	−4.714	3.5E−10	−28.900
(20%, 40%)	0.34	0.997	0.32	−1.061	0.17	−1.5	0.01	−3.498	6.5E−10	−26.943
(25%, 50%)	0.13	1.681	0.12	−1.71	0.22	−1.309	0.003	−4.118	3.4E−09	−22.373

Table 2 Comparing different pairs of algorithms using the Student's paired *t*-test statistic with the second data set.

Missing%	T(GREM,MEM)		T(LGREM,GREM)		T(LGREM,MEM)		T(LGREM,NNEM)		T(LGREM,MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.0003	5.741	0.0004	−5.428	0.09	1.922	0.51	0.681	0.56	0.605
(10%, 20%)	2.7E−09	22.955	4.5E−09	−21.652	0.55	0.620	0.55	−0.616	0.003	−3.982
(15%, 30%)	0.001	5.210	1.5E−08	−18.826	0.09	−1.926	0.06	−2.152	2.4E−07	−13.754
(20%, 40%)	0.03	2.495	0.002	−4.279	0.23	−1.300	0.07	−2.098	0.001	−4.926
(25%, 50%)	0.88	0.152	0.02	−2.784	0.03	−2.508	0.76	−0.315	0.002	−4.253

Table 3 Comparing different pairs of algorithms using the Student's paired *t*-test statistic with the third data set.

Missing (%)	T(GREM,MEM)		T(LGREM,GREM)		T(LGREM,MEM)		T(LGREM,NNEM)		T(LGREM,MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.34	1	0.34	−1	1	0	0.05	−2.236	0.03	−2.535
(10%, 20%)	0.59	0.557	0.59	−0.557	1	0	0.02	−2.689	5.4E−05	−7.149
(15%, 30%)	1	0	0.59	−0.557	0.34	−1	0.001	−5.014	1.7E−09	−24.247
(20%, 40%)	0.34	0.998	0.34	−0.998	0.34	−1	0.001	−4.743	1.9E−10	−30.992
(25%, 50%)	0.04	2.357	0.04	−2.404	0.28	−1.152	0.003	−4.129	4.2E−08	−16.809

feature vectors represent three classes each of which has 50 feature vectors belonging to it. Two of these classes are overlapping in the data space. Correlations between different pairs of features of this data set are moderate. When each one of these five data sets is used, the missing values are put in the third and in the fourth data features. In addition, each one of the FMMs learned by all algorithms compared in this study consists of three Gaussian components that have non-restricted covariance matrices.

6.4. The sixth data set

This data set is the Pima Indians Diabetes data set.[1] It contains 768 feature vectors each of which is a vector in eight-feature space. These feature vectors represent two classes; the first class has 500 feature vectors belonging to it; the second class has 268 feature vectors belonging to it. These classes are largely overlapping. Correlations between different pairs of features of this data set are too weak. The missing values are put in the second and in the fifth features of the data set. Each one of the FMMs learned by all algorithms compared in this study consists of two Gaussian components that have non-restricted covariance matrices.

The evaluation criterion used in this study to compare the different algorithms is the Mis-Classification Error (MCE). It is computed by comparing the clustering results, obtained using Bayes decision rule, of the learned FMM with the true classification of the data feature vectors, assuming each class is represented by a component in the FMM. Components of the FMM are allocated to different data classes such that the total number of misclassified feature vectors in the data set is minimum. Let the number of feature vectors belonging to class i be N_i, from which N_i^m feature vectors are not clustered into the component that represents this class in the FMM. Then the MCE for class i is computed as $\text{MCE}_{class_i} = \frac{N_i^m}{N_i}$. Assuming the data set is generated from K classes the total MCE is the average of all the class-MCEs and it is computed as $\text{MCE}_T = \frac{1}{\text{K}}\left(\sum_{i=1}^{\text{K}} \text{MCE}_{class_i}\right)$.

Tables 1–5 show comparisons of different pairs of the algorithms using the Student's paired *t*-test statistic with each one of the first five data sets. The *P*-value is the significance and the *T*-value is the *t*-statistic. This test examines the statistical significance of the difference in performance of pairs of

[1] Available at: <http://www.ics.uci.edu/~mlearn/MLSummary.html>

Table 4 Comparing different pairs of algorithms using the Student's paired *t*-test statistic with the fourth data set.

Missing (%)	T(GREM,MEM)		T(LGREM,GREM)		T(LGREM,MEM)		T(LGREM,NNEM)		T(LGREM,MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.002	4.432	9.7E−06	−8.865	0.22	−1.309	0.72	0.375	1.4E−05	−8.486
(10%, 20%)	0.0003	5.604	5.9E−08	−16.150	0.13	−1.683	0.15	−1.585	9.3E−05	−6.657
(15%, 30%)	0.34	0.999	4.7E−06	−9.669	0.004	−3.838	0.12	−1.706	4.9E−08	−16.521
(20%, 40%)	0.88	−0.159	9.7E−05	−6.620	0.001	−5.229	0.05	−2.251	1.3E−05	−8.573
(25%, 50%)	0.002	−4.247	0.01	−3.044	0.002	−4.250	0.42	−0.846	0.001	−4.993

Table 5 Comparing different pairs of algorithms using the Student's paired *t*-test statistic with the Iris data set.

Missing (%)	T(GREM,MEM)		T(LGREM,GREM)		T(LGREM,MEM)		T(LGREM,NNEM)		T(LGREM,MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.002	−4.180	0.89	−0.145	0.001	−4.708	0.81	−0.250	0.02	−2.814
(10%, 20%)	0.08	−1.954	0.04	−2.432	1.2E−04	−6.465	0.08	−1.969	4.2E−06	−9.803
(15%, 30%)	0.04	−2.440	0.34	−1.016	0.005	−3.630	0.26	−1.196	0.001	−5.117
(20%, 40%)	0.06	−2.172	0.21	−1.337	0.001	−4.719	0.03	−2.508	1.6E−04	−6.212
(25%, 50%)	0.82	−0.235	0.003	−4.053	0.002	−4.387	0.99	−0.019	3.1E−04	−5.654

Table 6 The misclassification errors of the FMM learned by the EM algorithm using the Pima data set without missing values.

Data	MCE
Class 1	0.426
Class 2	0.511
Total	0.469

algorithms using their total MCEs obtained from ten different experiments. In each experiment, a different group of feature vectors is randomly selected to contain missing values. The results of this test are shown for each pair of percentages of missing values in the third and in the fourth features of each one of the first five data sets. The shaded cells in each table represent the cases in which the difference in performance of certain pairs of algorithms are statistically significant according to the 5% significance level. Table 6 shows both class and total MCEs when the Pima data set is used with no missing values. Table 7 shows a comparison of different pairs of the

Table 7 Comparing different pairs of algorithms using the Student's paired *t*-test statistic with the Pima data set.

Missing (%)	T(GREM,MEM)		T(LGREM,GREM)		T(LGREM,MEM)		T(LGREM,NNEM)		T(LGREM,MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.004	−3.878	0.009	3.305	0.685	−0.420	0.151	−1.570	0.098	1.847
(10%, 20%)	0.331	−1.026	0.092	1.887	0.198	1.391	0.297	−1.107	0.185	1.437
(15%, 30%)	0.573	−0.585	0.559	0.607	0.996	0.005	0.059	−2.158	0.316	1.061
(20%, 40%)	0.337	1.013	0.027	2.640	0.023	2.731	0.729	−0.358	0.217	1.327
(25%, 50%)	0.225	1.301	0.067	2.078	0.008	3.415	0.035	−2.478	0.286	1.134

Table 8 Comparing both class-MCEs of each algorithm using the Student's paired *t*-test statistic with the Pima data set.

Missing (%)	GREM		MEM		LGREM		NNEM		MENEM	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.204	1.369	0.334	−1.020	0.100	−1.831	4.0E−06	−9.852	0.415	−0.855
(10%, 20%)	0.977	0.030	0.693	0.407	0.251	−1.228	0.004	−3.775	0.160	−1.533
(15%, 30%)	0.640	0.484	0.643	0.479	0.900	−0.129	0.009	−3.348	0.385	−0.913
(20%, 40%)	0.020	2.824	5.8E−10	27.26	0.750	−0.329	0.047	−2.295	0.048	−2.283
(25%, 50%)	0.911	−0.116	0.001	5.102	0.217	1.328	0.017	−2.932	0.005	−3.704

Table 9 Comparing different pairs of algorithms using the Student's paired *t*-test statistic with the Pima data set, while the clustering results are used instead of the true classification of the data set.

Missing (%)	T(GREM,MEM)		T(LGREM,GREM)		T(LGREM,MEM)		T(LGREM,NNEM)		T(LGREM,MENEM)	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.002	4.156	0.005	−3.714	0.888	0.144	0.260	1.202	0.047	−2.299
(10%, 20%)	0.049	2.280	0.001	−4.660	0.029	−2.590	0.718	0.372	0.005	−3.650
(15%, 30%)	0.216	1.329	0.022	−2.764	0.145	−1.594	0.451	0.789	0.0003	−5.719
(20%, 40%)	0.064	2.112	0.010	−3.259	0.182	−1.446	0.170	1.491	0.010	−3.251
(25%, 50%)	0.162	1.525	0.050	−2.261	0.635	−0.492	0.022	2.761	0.024	−2.722

Table 10 Comparing both class-MCEs of each algorithm using the Student's paired *t*-test statistic with the Pima data set, while the clustering results are used instead of the true classification of the data set.

Missing (%)	GREM		MEM		LGREM		NNEM		MENEM	
	P	*T*	*P*	*T*	*P*	*T*	*P*	*T*	*P*	*T*
(5%, 10%)	0.003	−4.018	0.252	−1.224	0.001	−4.815	3.4E−04	−5.578	0.032	−2.540
(10%, 20%)	0.001	−4.533	0.457	−0.778	0.232	−1.282	1.9E−04	−6.041	0.019	−2.855
(15%, 30%)	0.003	−4.132	0.769	−0.303	0.734	−0.351	0.032	−2.534	0.040	−2.401
(20%, 40%)	0.002	−4.260	2.0E−06	10.74	0.527	−0.659	0.006	−3.619	0.009	−3.336
(25%, 50%)	0.016	−2.966	0.019	2.840	0.191	−1.414	0.013	−3.073	0.001	−4.873

algorithms using the Student's paired *t*-test statistic with the Pima data set when different pairs of percentages of missing values occur in both of the second and the fifth features. The algorithms are compared using their total MCEs obtained from ten different experiments. Table 8 shows a comparison of both class-MCEs of each algorithm using the Student's paired *t*-test statistic with the Pima data set. This test examines the statistical significance of the bias of each algorithm in estimating the missing values due to the difference in class sizes using pairs of the class-MCEs of the same algorithm obtained from ten different experiments. The shaded cells represent the cases in which the bias of a certain algorithm is statistically significant according to the 5% significance level. Since both classes of the Pima data set are largely overlapping (see Table 6), and their sizes are very different, biased estimation of the missing values can affect the learning of the FMM parameters such that low MCE can be obtained. Therefore, it is decided to remove the effect of the overlap between classes on the learning of the FMM parameters. This is achieved by using the clustering results, obtained from the FMM trained by the EM algorithm using the Pima data set without missing values, instead of the true classification of this data set in computing the MCE. This will clearly show the effect of the estimation of the missing values on the learning of the FMM parameters. The new results of the algorithms compared are shown in Tables 9 and 10.

7. Discussion of results

Tables 1–5 show that the performance of the LGREM algorithm outperforms (*T*-values are negative and *P*-values are less than or equal to 0.05) the performances of the other algorithms, especially when there are weak correlations between data features, few outliers in the data set, or overlap among data classes in the data space. In these cases, local tuning of the general regression used in the LGREM algorithm is less dependent on the overall correlations than the general regression used in the GREM algorithm. Therefore, the LGREM algorithm outperforms the GREM algorithm when features of the data are too weakly correlated (see the results with the second and the fourth data sets in Tables 2 and 4). In addition, estimating the missing values from the observed feature vectors belonging to each cluster used in the LGREM algorithm is less dependent on the FMM parameters than the maximum likelihood estimation used in the MEM algorithm. Therefore, the LGREM algorithm outperforms the MEM algorithm when there are few outliers in the data set, or overlap among data classes in the data space (see the results with the fourth and the Iris data sets in Tables 4 and 5). This requires each mixture component or cluster in the data space to contain at least one feature vector that is fully observed.

Table 6 shows that the Pima data set has two classes that are largely overlapping. Because of this overlap and the large difference in size between both classes the algorithms that produce biased estimation of the missing values produce less total MCE than the algorithms that produce unbiased estimations. This is shown in Tables 7 and 8. Although Table 8 shows that the LGREM algorithm is unbiased in all pairs of percentages of the missing values Table 7 shows that it produces higher total MCE than other algorithms. Using clustering results of the FMM, learned from the Pima data set with no missing values, instead of the true classification in computing the MCE removes the effect of the class overlap and shows only the effect of the estimation of the missing values on the learning of the FMM parameters. Therefore, Table 9 shows that the LGREM and the NNEM algorithms are statistically similar and they have the smallest total MCEs among all algorithms in all pairs of percentages of the missing values. Table 10 shows that the LGREM algorithm is superior over the NNEM algorithm because it is unbiased in its estimation of the missing values due to the large difference in class sizes.

In general, local tuning of the general regression causes the LGREM algorithm to outperform the GREM algorithm in the case of too weak global correlations between different pairs of data features. This means that the LGREM algorithm is insensitive to global correlations between features of the input data set. In addition, this feature causes the LGREM algorithm to outperform the MEM algorithm in cases such as the occurrence of some outlier feature vectors or the overlap among data classes in the data space. This conclusion agrees with the results shown in [27] regarding the preference of the imputation techniques that use the pair-wise relations between data features to those techniques that do not. Finally, when the sizes of the data classes are largely different (i.e., unbalanced data) the LGREM algorithm is superior over the other algorithms compared with as it produces the most accurate and unbiased estimation of the missing values and the parameters of the FMM, which is used for clustering the input data set. This is because the imputation of the missing values in the LGREM algorithm is both local and independent of FMM parameters.

8. Conclusions

In this paper, the GREM and the MEM algorithms are analysed and their strengths and weaknesses are explained. This analysis leads to the proposal of the LGREM algorithm to overcome problems of both algorithms. A comparison study shows the superiority of the LGREM algorithm over several algorithms commonly used in the literature including the MEM algorithm and the GREM algorithm in unsupervised learning of FMM parameters using data sets with missing values. The LGREM algorithm produces the most accurate and unbiased estimation of the missing values and the best estimation of the FMM parameters. This is shown clearly when few outliers occurs in the data set, data classes are overlapped in the data space, or when data classes have large differences in their sizes.

References

[1] McLachlan G, Peel D. Finite mixture models. New York: Wiley; 2000.

[2] Tråvén H. A neural network approach to statistical pattern classification by semiparametric estimation of probability density functions. J IEEE Trans Neural Netw 1991;2(3):366–77.

[3] Dempster A, Laird N, Rubin D. Maximum likelihood from incomplete data via the EM algorithm (with discussion). J R Stat Soc 1977;B(39):1–38.

[4] Little R, Rubin D. Statistical analysis with missing data. New York: John Wiley & Sons; 1987.

[5] Ghahramani Z, Jordan M. Supervised learning from incomplete data via an EM approach. In: Cowan J, Tesauro G, Alspector J, editors. Advances in neural information processing systems. San Francisco, CA, USA: Morgan Kaufmann Publishers; 1994. p. 120–7.

[6] Hunt L. Clustering using finite mixture models. Ph.D. thesis. New Zealand: Department of Statistics, University of Waikato; 1996.

[7] Hunt L, Jorgensen M. Mixture model clustering for mixed data with missing information. J Comput Stat Data Anal 2003;41:429–40.

[8] Dybowski R. Classification of incomplete feature vectors by radial basis function networks. J Pattern Recognit Lett 1998;19: 1257–64.

[9] Morris A, Cooke M. Some solutions to the missing feature problem in data classification, with application to noise robust ASR. In: Proceeding of the international conference on acoustics speech and signal processing, Seattle, WA, USA; 1998. p. 737–40.

[10] Domma F, Ingrassia S. Mixture models for maximum likelihood estimation from incomplete values. In: Borra S, Rocci R, Vichi M, Schader M, editors. Studies in classification, data analysis and knowledge organization. Berlin: Springer-Verlag; 2001. p. 201–8.

[11] Yoon S, Lee S. Training algorithm with incomplete data for feed-forward neural networks. J Neural Process Lett 1999;10:171–9.

[12] Troyanskaya O, Cantor M, Sherlock G, Brown P, Hastie T, Tibshirani R, Botstein D, Altman R. Missing value estimation methods for DNA microarrays. J Bioinform 2001;17(6):520–5.

[13] Lin TI, Lee JC, Ho HJ. On fast supervised learning for normal mixture models with missing information. J Pattern Recognit 2006;39:1177–87.

[14] Bouveyron C, Girard S. Robust supervised classification with mixture models: learning from data with uncertain labels. J Pattern Recognit 2009;42:2649–58.

[15] Specht D. A general regression neural network. J IEEE Trans Neural Netw 1991;2(6):568–76.

[16] Tresp V, Ahmad S, Neuneier R. Training neural networks with deficient data. In: Cowan J, Tesauro G, Alspector J, editors. Advances in neural information processing systems, vol. 6. San Mateo, CA: Morgan Kaufman; 1994. p. 128–35.

[17] Tresp V, Neuneier R, Ahmad S. Efficient methods for dealing with missing data in supervised learning. In: Tesauro G, Touretzky D, Leen T, editors. Advances in neural information processing systems, vol. 7. Cambridge, MA: MIT Press; 1995. p. 689–96.

[18] Raymond M. Missing data in evaluation research. J Eval Health Prof 1986;9:395–420.

[19] Raymond M, Roberts D. A comparison of methods for treating incomplete data in selection research. J Educ Psychol Meas 1987;47:13–26.

[20] Zhou X, Wang X, Dougherty E. Missing-value estimation using linear and non-linear regression with Bayesian gene selection. J Bioinform 2003;19(17):2302–7.

[21] Guo P, Chen C, Lyu M. Cluster number selection for a small set of samples using the Bayesian Ying–Yang model. J IEEE Trans Neural Netw 2002;13(3):757–63.

[22] Milligan GW, Cooper MC. A study of standardization of variables in cluster analysis. J Classif 1988;5:181–204.

[23] Schaffer CM, Green PE. An empirical comparison of variable standardization methods in cluster analysis. J Multivariate Behav Res 1996;31(2):149–67.

[24] Vlassis N, Likas A. A greedy EM algorithm for Gaussian mixture learning. J Neural Process Lett 2002;15:77–87.

[25] Yin H, Allinson NM. Comparison of a Bayesian SOM with the EM algorithm for Gaussian mixtures. In: Proceeding of workshop on self-organising maps (WSOM'97); 1997. p. 118–23.

[26] Fisher R. The use of multiple measurements in taxonomic problems. Annu Eugenics 1936;7:179–88.

[27] Huisman M. Imputation of missing item responses: some simple techniques. J Qual Quant 2000;34:331–51.