Research Report

Department of Statistics

No. 2008:4

Implementation of the MBCA Matlab Program for Model-Based Cluster Analysis

Jessica Franzén

Department of Statistics, Stockholm University, SE-106 91 Stockholm, Sweden

Implementation of the MBCA Matlab Program for Model-Based Cluster Analysis

Jessica Franzén*
Department of Statistics
University of Stockholm

April 2008

Abstract

This guide describes and explains the software package *Model-Based Cluster Analysis* (MBCA), which is written in Matlab. The programs estimates the parameters of a multivariate mixture model of normal distributions and clusters the observations. Full posterior distributions are obtained using the Gibbs sampler. An introduction is given to the theory of model-based clustering and to Bayesian inference. Instructions are presented on how to enter data and prior specifications into the program. Special programs in this package take care of deviant observations, handle missing data, and perform longitudinal cluster analysis.

Keywords: Cluster analysis, Clustering, Classification, Mixture model, Model-based, Gaussian, Bayesian inference, MCMC, Gibbs sampler, Matlab, Deviant group, Longitudinal, Missing data, Multiple imputation

*The support from the Swedish Research Council (Grant no 2005-2003) are gratefully acknowledged.

1 Introduction

Classification or clustering of data is one of the most important techniques of multivariate analysis. Many software packages contain prewritten programs for handling deterministic cluster analysis. Model-based cluster analysis is a good alternative for finding group patterns in data. It has become increasingly preferred over traditional deterministic clustering due to its flexibility. Clustering based on probability models has certain advantages for handling overlapping groups and groups of different sizes and shapes. The estimation method, however, relies on an iteration procedure which is not straightforward to implement and the choice of prewritten programs is limited. The MBCA program is written for the model-based clustering approach. The program consists of five variants that can be used in standard and non-standard situations. We give an introduction to the theory and also practical guidance to the program.

The MBCA program is written in Matlab, and Bayesian inference is applied in the program. Standard techniques such as ML-estimation are sometimes used for the model-based approach, often with the EM-algorithm. MCLUST and MIXMOD are two existing programs written for this purpose. Biernacki et al. (2005) give an introduction to MIXMOD, and Fraley and Raftery (2007), (2006), and (2003) do the same for the MCLUST software. MCLUST is available with an R or S-PLUS language interface. MIXMOD is interfaced with SCILAB and Matlab. The EM algorithm is advanced in the sense of allowing for different sizes, shapes, and orientations of the clusters. Still, it comes with some limitations that we can overcome with the Bayesian approach. The MCMC technique used will eventually reach the target distribution, even if it takes some time. The maximum likelihood estimator runs the risk of getting stuck in a local maximum, if present. In addition, the method only gives point estimates and produces no estimates regarding the uncertainty of the parameters. The Bayesian approach generates point estimates of all variables as well as associated uncertainty in the form of the whole posterior distribution. Moreover, the method generates posterior predictive probabilities for a single observation's being derived from all the different distributions (groups) in the model. A comprehensive explanation of Bayesian analysis is given in Bernardo and Smith (2000) and in Gelman et al. (2004), and MCMC methods can be studied in Gamerman and Lopez (2006)

WINBUGS is a widely used software package for MCMC computations for a wide variety of Bayesian models, including normal mixtures. The program is very flexible, but because of that it is not straighforward to use. The user have to do some own coding which requires previous knowledge about Bayesian inference and the program itself. Discussions on how to use WINBUGS is found in Schollnik (2001), Fryback et al. (2001), and Woodworth (2004, Appendix B). The MBCA program is not nearly as comprehensive as WINBUGS, but is instead much more user friendly. Without much previous knowledge, one may execute the MCMC simulations for the basic case with a mixture of J multivariate normal distributions as well as for a few special situations.

The MBCA package is available on www.statistics.su.se/forskning/MBCA. The package contains five programs written for special features and two programs for graphical presentations of the results.

- The first program handles the basic case of grouping data into J clusters.
- Outliers or deviant observations may interfere in a negative way when clustering data. The second program handles these observations by adding an extra cluster into the solution. This group consists of observations which do not fit into one of the general group patterns.
- Missing data is almost inevitable in real-life data. The model-based clustering approach can easily and effectively be extended to handle data with item non-response. In Program 3, multiple imputation is carried out as a step in the algorithm.
- The next issue is longitudinal studies. Program 4 gives the possibility of clustering data from two or three repeated measurements. Changes in cluster divisions over time and transition patterns between clusters at different time points may be analyzed.
- Repeated measurements are especially exposed to missing data. Program 5 combines the longitudinal clustering with missing data. All observations from one or more time points may even be missing.
- Two programs are included for graphical presentations of the results. Iteration plots and histograms over the estimated parameters can be obtained. The program Graph1.m handles cross sectional data while Graph2.m handles longitudinal data.

In the MBCA program, data is assumed to be generated from a mixture model of multivariate normal distributions. Each distribution represents a cluster with its specific group parameters. The programs do not come with limitations on the number of variables or clusters. In Section 2, a short presentation of the theory is given. The mixture model is presented and, in a Bayesian manner, prior distribution and posterior derivations are given. The section also includes a brief description of the MCMC estimation technique. A more complete description of the theory and also a number of applications for the different features of the program can be found in Franzén (2006), (2007), (2008a), and (2008b). Section 3 gives instructions on how to use each of the five programs. It gives guidance on how to make the model- and prior specifications. Finally, in Section 4, a number of practical considerations and possible challenges faced when using the program are explained.

2 Mixture Model

In MBCA, the n multivariate observations $\mathbf{y} = \{\mathbf{y}_1, ..., \mathbf{y}_n\}$ are assumed to be independent observations of a mixture distribution with density

$$f(\mathbf{y}_i \,|\boldsymbol{\theta}) = \sum_{j=1}^{J} \omega_j f_j(\mathbf{y}_i \,|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \qquad i = 1, ..., n$$

where ω_j $\{j = 1, ..., J\}$ are the mixing proportions which satisfy $0 < \omega_j < 1$ and $\sum_{j=1}^{J} \omega_j = 1$. The density $f_j(\mathbf{y}_i \,|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ denotes a multivariate normal distribution with mean vector $\boldsymbol{\mu}_j$ and covariance matrix $\boldsymbol{\Sigma}_j$. Each of these J densities corresponds to a cluster with specific characteristics described by its parameters.

The unknown parameters to be estimated are thus $(\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_J, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_J, \omega_1, ..., \omega_J)$. We also introduce a classification vector $\mathbf{V} = (v_1, ..., v_n)$, where $v_i = j$ implies that observation $\mathbf{y}_i$ is classified into Cluster j. The classification vector is regarded as an unknown parameter, and the marginal of its posterior are the cluster probabilities for single observations.

2.1 Prior Distributions

In a Bayesian analysis each parameter of the model follows a distribution. The prior opinion on a parameter is described by its prior distribution. We use conjugate priors for the parameters of the mixture model according to Lavine and West (1992). When there are no prior opinions, a vague prior can be used within this class of conjugate priors.

The prior distribution for $\boldsymbol{\Sigma}_j$ is the inverse Wishart distribution

$$\boldsymbol{\Sigma}_j \sim W^{-1}\left(m_j, \boldsymbol{\psi}_j\right)$$

with m_j degrees of freedom and scale matrix $\boldsymbol{\psi}_j$.

No limitations are put on variability between clusters, i.e. we allow for each cluster to have its own specific covariance matrix in terms of volume, shape and orientation. This makes it possible to work with cases where one cluster (or more) may have a distinguishing characteristic in terms of large variance.

The prior distribution for $\boldsymbol{\mu}_j$ is the multivariate normal distribution with known covariance matrix $\boldsymbol{\Sigma}_j/\tau_j$ for some precision parameters τ_j. That is,

$$\boldsymbol{\mu}_j \,|\boldsymbol{\Sigma}_j \sim N_M\left(\boldsymbol{\xi}_j, \boldsymbol{\Sigma}_j/\tau_j\right)$$

The conjugate prior distribution for $\boldsymbol{\Omega} = (\omega_1, ..., \omega_J)$ is a multivariate generalization of the beta distribution, known as the Dirichlet distribution

$$(\omega_1, ..., \omega_J) \sim D(\alpha_1, ..., \alpha_J)$$

The prior distribution can be seen as a probability (or density) function describing the uncertainty before the data is observed. The prior belief, specified here by the location and precision parameters m_j, $\boldsymbol{\psi}_j$, $\boldsymbol{\xi}_j$, τ_j, and α_j $\{j = 1, ..., J\}$, can vary between persons according to their knowledge and experience. With an uninformative prior the posterior distribution is almost completely determined by data. In Section 3 there is an explanation of how to specify the priors in accordance with ones choice.

2.2 Posterior Derivations

The likelihood from data, together with the priors described in the previous section, generates the posterior distribution for each parameter. The transformation from prior to posterior is given by Bayes theorem, which says that the posterior distribution of the parameters, $\boldsymbol{\theta}$, is proportional to the prior information times the information from data, i.e. the likelihood function.

$$\begin{aligned} Posterior &\propto Prior \times Likelihood\ of\ data \\ \pi(\boldsymbol{\theta}|data) &\propto \pi(\boldsymbol{\theta})\ \times p(data|\boldsymbol{\theta}) \end{aligned}$$

The posterior distributions is in this program given by a set of conditional distributions. The posterior distribution of $\boldsymbol{\Sigma}_j$ is the inverse Wishart distribution conditional on $\mathbf{y}$ and $\mathbf{V}$,

$$\boldsymbol{\Sigma}_j\,|\mathbf{y}, \mathbf{V} \sim W^{-1}\left(n_{j+}m_j, \boldsymbol{\psi}_j + \boldsymbol{\Lambda}_j + \frac{n_j\tau_j}{n_j+\tau_j}(\overline{\mathbf{y}}_j - \boldsymbol{\xi}_j)(\overline{\mathbf{y}}_j - \boldsymbol{\xi}_j)^t\right)$$
$$\text{where } \boldsymbol{\Lambda}_j = \sum_{i\in j}(\mathbf{y}_i - \overline{\mathbf{y}}_j)(\mathbf{y}_i - \overline{\mathbf{y}}_j)^t$$

The degrees of freedom equal the sum of the prior degrees of freedom m_j, and the number of observations in Cluster j, n_j. The scale matrix has three components - the prior opinion of $\boldsymbol{\Sigma}_j$, namely $\boldsymbol{\psi}_j$, the sum of squares $\boldsymbol{\Lambda}_j$, and the deviation between prior and estimated mean values.

The posterior distribution for $\boldsymbol{\mu}_j$ is the multivariate normal which is expressed conditional on $\mathbf{y}$, $\boldsymbol{\Sigma}_j$, and $\mathbf{V}$, namely:

$$\boldsymbol{\mu}_j\,|\mathbf{y}, \boldsymbol{\Sigma}_j, \mathbf{V} \sim N_M\left(\overline{\boldsymbol{\xi}}_j, \boldsymbol{\Sigma}_j/(\tau_j + n_j)\right)$$

$$\text{where } \overline{\boldsymbol{\xi}}_j = \frac{\tau_j\boldsymbol{\xi}_j + n_j\overline{\mathbf{y}}_j}{(n_j + \tau_j)}$$

The mean vector $\overline{\boldsymbol{\xi}}_j$ in the posterior distribution is a weighted sum of the prior- and, by data, estimated mean values.

The posterior distribution of the probability vector $\mathbf{\Omega}$ conditional on $\mathbf{V}$ is the Dirichlet distribution

$$(\omega_1, ..., \omega_J \left| \mathbf{V} \right.) \sim D\left(\alpha_1 + \sum_{i=1}^{n} I\left(v_i = 1\right), ..., \alpha_J + \sum_{i=1}^{n} I\left(v_i = J\right)\right)$$

The prior specification $\alpha_1, ..., \alpha_J$, and the number of objects classified into each Cluster j described by $\sum_{i=1}^{n} I\left(v_i = j\right)$, are the updated parameters in the posterior of $\mathbf{\Omega}$.

The posterior probability t_{ij} for observation $\mathbf{y}_i$ to belong to Cluster j is calculated according to Bayes theorem conditionally on $\mathbf{y}$, $\boldsymbol{\mu}_j$, and $\mathbf{\Sigma}_j$

$$t_{ij} \left| \boldsymbol{\mu}_j, \mathbf{\Sigma}_j, \mathbf{\Omega} \right. = \frac{\omega_j f\left(\mathbf{y}_i \left| \boldsymbol{\mu}_j \mathbf{\Sigma}_j \right.\right)}{\sum\limits_{j=1}^{J} \omega_j f\left(\mathbf{y}_i \left| \boldsymbol{\mu}_j \mathbf{\Sigma}_j \right.\right)} \qquad i = 1, ..., n$$

The probabilities are the basis for the simulation of the classification vector $\mathbf{V}$.

2.3 Parameter Estimation through the Gibbs sampler

The Gibbs sample algorithm (Geman and Geman, 1984) is used to estimate the model parameters $\mathbf{\Sigma}_j$, $\boldsymbol{\mu}_j$, $\mathbf{\Omega}$, and the classification vector $\mathbf{V}$. The Gibbs sampler works by iteratively drawing samples from the full conditional posterior distributions of the parameters in the model, as presented in the previous section. A parameter value simulated from its posterior distribution in one iteration step is used as a conditional value in the next step. Replicating the process, consisting of steps 1 to 4 below, allows for an approximate random sample to be drawn from the joint posterior density.

1. New values for $\mathbf{\Sigma}_j$, $j = 1, ..., J$, are simulated from the inverse Wishart posterior distributions, conditional on $\mathbf{y}$ and the previous $\mathbf{V}$.

2. New values for $\boldsymbol{\mu}_j$, $j = 1, ..., J$, are simulated from the multivariate normal posterior distributions, conditional on $\mathbf{y}$ and the previous values of $\mathbf{\Sigma}_j$ and $\mathbf{V}$. The new covariance matrices simulated in step 1 are considered as known in step 2.

3. A new vector probability $\mathbf{\Omega}$ is simulated from the Dirichlet posterior distribution, conditional on the previous $\mathbf{V}$.

4. In the last step, new classification variables v_i are simulated according to their posterior probabilities t_{ij}, conditional on the new $\boldsymbol{\mu}$, $\mathbf{\Sigma}$, and $\mathbf{\Omega}$. The element $v_i = j$ with probability t_{ij}, independent of all other $v_{i'}$ $i' \neq i$.

3 Programs

The programs run in a Matlab environment. Versions 7.1 or later are recommended. Previous versions have a fault in one of Matlab's own m-files, which may overestimate the covariances. For effective simulations, the recommended computer capacity is an Intel Core 2 Duo processor with at least 2 GHz and 2 GB RAM, or corresponding. Instructions on how to use the programs are given in the following steps.

1. **Download the programs**

   Download the Matlab programs from www.statistics.su.se/forskning/MBCA into one catalogue without changing their names or formats. There are a total of six files, one for each program described below and two for graphical presentations.

2. **Create a data matrix**

   Open Matlab and the command window will appear. Before running any of the programs, the data matrix Y has to be specified in the command window. The data matrix Y has to be of size $K \times n$, where K is the dimension of data and n the number of observations. Each observation in Y is then represented by a column and each variable by a row. For small data materials, the matrix can be typed directly in the Matlab command window. For an imaginary data set with 4 observations in 3 dimensions type:

   ```
   >> Y=[2.6 1.4 3.8 4.5;4.5 1.2 6.9 4.5;6.3 4.5 1.1 2.5]
   ```

   which Matlab writes as:

   ```
   Y =
   2.6000 1.4000 3.8000 4.5000
   4.5000 1.2000 6.9000 4.5000
   6.3000 4.5000 1.1000 2.5000
   ```

   Most data sets are too big to be typed manually. If data is stored in Excel, one may fetch data by the Matlab command

   ```
   >> Y = xlsread('filename')
   ```

   Data in Excel is often in the format of columns representing variables and rows representing observations, i.e. the opposite of the matrix Y. This is easily put right by transposing the matrix;

   ```
   >> Y = Y'
   ```

Other alternatives to *xlsread* for other data forms than Excel are *dlmread*, *wk1read*, *cdfread*, and *textread*. For information on these options, type *help* followed by the desired alternative in the command window, or use the Matlab help menu.

For Programs 4 and 5, where we work with longitudinal data, data is specified in one matrix for each time point. Y1 contains data from time point 1, Y2 from time point 2, and, if present, Y3 from time point 3. Specifications for these data matrices are performed in the same way as above. Note that the number of observations must be the same for all time points, but dimensions on data may be different. This means the number of columns in Y1, Y2, and Y3 have to be the same, but the number of rows may differ. The same column must correspond to the same individual at all time points. If an individual is not measured at a certain time point, enter NaN in that column.

3. **Start the program**

   To start the desired program type its name in the command window. Depending on the current directory in Matlab, it may be enough just to print the file name. If the current directory, which shows if cd is typed in the command window, is set to another location, the whole pathname must be specified. Alternatively, one may change the current directory by typing cd('directory'), where directory is the pathname.

4. **Model and prior specifications**

   When the program is started, model and possibly prior specifications are typed directly in the command window according to instructions that appear on the screen. Necessary entries include the number of clusters, iterations, and burn-in iterations. For Program 2, it also includes specification of the possible outcomes for the deviant cluster. After making model specifications, one has the choice of using default prior specifications or making customized specifications. Default prior values are prespecified in the program. One may, however, change these specifications to other values by typing 1 when the question appears on the screen. If 1 is typed, a number of prior specifications that need to be made appear in turn on the screen. Instructions on how to make these specifications are given in the following subsections.

   If 0 is typed, default values are used in the analysis. The default priors are rather vague but center around the mean and covariance for the whole data set. It should be said, that it is opposite to the Bayesian idea when using the data in the prior specifications. However, we make this moderate overstep to simplify for the user. At the same time we reduce the strength of the mean and covariance priors by putting low values on the other prior parameters for the mean vectors and covariance matrices. The degrees of freedom m_j, equal 10, and the precision parameters τ_j equal 1 for all clusters. Default priors for the cluster probabilities α_j is 5 for all clusters. In Program 2, α_j is 5 for all non-deviant clusters and 1 for the last deviant group, reflecting the prior belief of a smaller deviant cluster. For Programs 3 and 4, the β_j specifications for the transitions matrices are all set to 5.

5. **Running Time**

   After the specifications are made in the command window, the iteration process starts. This might take a considerable amount of time depending on the number of iterations, the extent of the data material, and the program used. Running time for Program 1 with 6 clusters, 7 variables, 100 000 iterations, and 1 000 observations was a little over 3 hours on a computer with 3 GHz and an Intel Core 2 Duo E6850 processor with 3 GB RAM. Running time for Program 4, with the same number of iterations and observations, and with 4 clusters in 3 dimensions at Time 1 and 3 clusters in 4 dimensions at Time 2, was almost 5.5 hours on the same computer. 100 000 iterations are usually considered a long iteration chain.

6. **Results**

   Estimation results are automatically presented in the command window after the program is executed. MEAN are the mean estimates where each column represents one cluster. PROB shows all the cluster probability estimates, and COV1, COV2,..., COVJ are the covariance estimates for the J clusters. To receive the cluster probabilities of all n objects, write

   ```
   >> CLUSTERPROB
   ```

   in the Matlab command window.

   Each row in CLUSTERPROB shows cluster probabilities for one observation. The columns represent the J clusters in the same order as the mean and covariance estimates are presented. When we have more than one time point, i.e. in Programs 4 and 5, the name of the estimation results are followed by a number corresponding to the time. MEAN1 are for example the mean estimates at Time 1 and COV12 is the covariance matrix for Cluster 2 at Time 1. Cluster probabilities at Time 2 for example are received by typing CLUSTERPROB2.

7. **Save the results**

   Save the results under "Save Workspace As" in the file menu. When opening the workspace again, the results will not automatically appear on the screen. You have to call each estimate by its name given above. Before running a new program, make sure to clear the previous data by typing

   ```
   >> clear
   ```

   and

   ```
   >> clc
   ```

8. **Graphical presentations of the results**.

   Iteration plots and histograms over estimated parameters may be obtained after the program is run. Iteration plots are useful when checking the convergence. If the method works properly, the iterations should generate "white noise" around the estimated mean value. The iterations, after the specified burn-in period, underlie the histogram which gives a visual representation of the posterior distribution of the estimated value. The figures below show an example of an iteration plot and corresponding histograms for three mean variables from one cluster. The convergence for this example was short, and the burn-in period of 5 000 iterations was much longer than necessary.

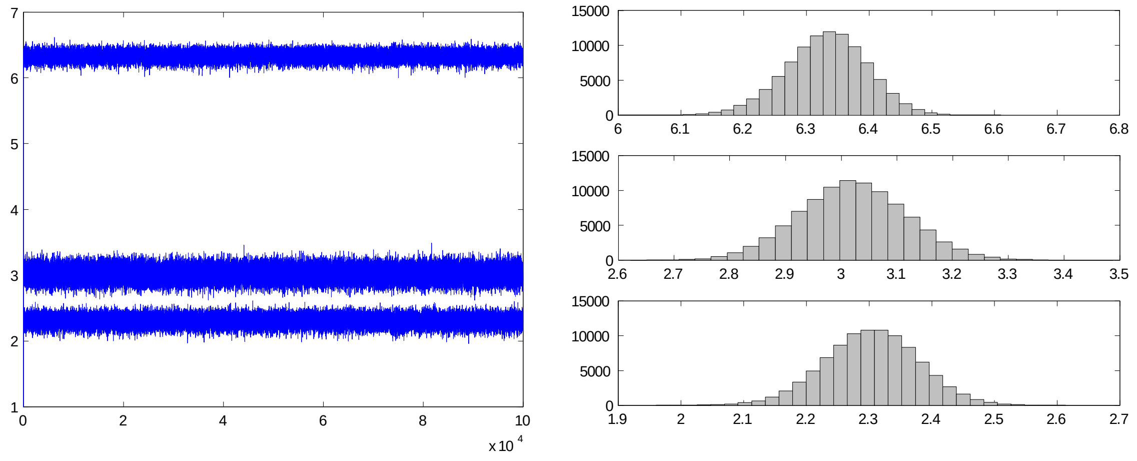

FIGURE 1: Left graph - Iteration plot for 3 mean variables. Right graph - Histograms for the same variables created from the last 95 000 iterations (100 000 minus a burn-in of 5 000).

To obtain iteration plots and/or histograms for the estimates, open the program Graph1.m or Graph2.m through the file menu. The first program handles cross sectional data and is used after running Program 1, 2, or 3. The second program handles longitudinal data and is used after running Program 4 or 5. When opening the suitable program, one will then enter the editor where several sections are prepared for different plots and histograms. To obtain a specific graph, copy the corresponding section and simply paste it into the Matlab command window. Before copying, minor specifications need to be made as, for example, which cluster the graph shall illustrate. The first section in the Graph1.m program is shown below. When pasted into the command window, this section plots the iterations for one probability estimate. If another cluster than 1 is desired, change $j = 1$ to the number of the desired cluster.

%ITERATION PLOT FOR CLUSTER PROBABILITIES
%Before copying, enter j for the desired cluster $j = 1, ..., J$
$j = 1$;

$plot(Theta(:, j))$

The iteration plots show all generated values, including the burn-in iterations in the beginning. This way one can study how long it takes for the chain to converge. The histograms are plotted without the burn-in iterations, to give a picture of the true posterior distribution.

3.1 Program 1 - Clustering of J Groups

The first program handles the basic case where data is to be clustered in J different groups. The program is the foundation for the extended and modified programs to follow in the next sections. For a demonstration and application of this program see Franzén (2006). Below are instructions on how to enter model and prior specification into the program.

Model specifications

- The program asks for the number of clusters J. Specify and press enter.
- The program asks for the number of iterations T. The larger the number of T the better the estimates, but keep in mind that a large number of iterations may demand a lot of computer time, memory, and capacity.
- The program asks for the number of iterations F to discard in the beginning , i.e. the burn-in period. More on this in Section 5.1.
- The program asks if one wants to use default priors or not. For default values, type 0. If customized prior specifications are wanted, type 1. The program will then ask for new prior specifications. Below are instructions for each step.

Prior specifications for μ

- **Instruction 1**. Specify the precision parameters for each cluster in vector form, i.e. $[\tau_1\ \tau_2\ ...\ \tau_J]$. The length of the vector is equal to the number of clusters J.
- **Instruction 2**. Specify the mean of the prior beliefs of the mean values $\boldsymbol{\xi}_j$ in matrix form. The size of the matrix has to be $K \times J$. Rows represent variables and columns represent clusters. Each column in the matrix represents the vector $\boldsymbol{\xi}_j$ for Cluster j. For example

  [1 2 3;1 2 3;1 2 3;1 2 3]

  generates a matrix with values equal to 1 in column 1, 2 in column 2, and 3 in column 3. This corresponds to a prior belief of 1 for all variables in Cluster 1, and 2 for all variables in Cluster 2, and 3 for all variables in Cluster 3.

The cluster means are expected to be around the selected values in $\boldsymbol{\xi}_j$, $j = 1, ..., J$. A small value of the precision parameters τ_j gives less weight to the prior means and larger variance in the posterior distributions, compared to higher values. The precision of the prior opinion corresponds to having observed τ_j individuals that are known to come from that cluster. The choice $\tau_j = 0$, corresponds to having no information at all.

Prior specifications for Σ

- **Instruction 3**. Specify the degrees of freedom for each cluster in vector form i.e. $[m_1\ m_2\ ...\ m_J]$. The length of the vector is equal to the number of clusters J.
- **Instruction 4**. Specify the prior belief of the covariance matrices Σ_j for each cluster in matrix form. The program asks for one covariance matrix at a time, starting with the covariance for Cluster 1. The size of the matrix has to be $K \times K$. If, for example, one wants to use the identity matrix I as the prior covariance, type $eye(K)$. If another value a is desired instead of 1 in the diagonal, simply write $a * eye(K)$. If other values than 0 are desired for the non-diagonal values, i.e. the covariances, each matrix has to be typed out in its complete form. For example, if $K = 3$, [1.2 0.5 0.5;0.5 2 0.5;0.5 0.5 3] generates the prior covariance matrix

  ```
  1.2000 0.5000 0.5000
  0.5000 2.0000 0.5000
  0.5000 0.5000 3.0000
  ```

  Observe that $\boldsymbol{\psi}_j = m_j \Sigma_j$; but we specify m_j and Σ_j separately and leave it to the program to calculate $\boldsymbol{\psi}_j$. Σ_j should reflect the actual prior belief of the covariance matrix. The strength of our prior belief for $\boldsymbol{\Sigma}_j$ is adjusted with m_j. Our best prior guess of $\boldsymbol{\Sigma}_j$ would thus be $\boldsymbol{\psi}_j / m_j$, and the knowledge of the variance corresponds to the knowledge obtained from m_j individuals. The choice $m_j = 0$, corresponds to no prior knowledge.

Prior specifications for Ω

- **Instruction 5**. Specify the prior beliefs of the cluster proportions in vector form, i.e. $[\alpha_1\ \alpha_2\ ...\ \alpha_J]$. The length of the vector is equal to the number of clusters J.

The relative sizes of the Dirichlet parameters α_j describe the expected proportions between groups, and the sum of the α_j's is a measure of the strength of the prior distribution. The prior distribution is mathematically equivalent to a likelihood resulting from $\sum_{j=1}^{J} (\alpha_j - 1)$ observations with $\alpha_j - 1$ observations of the j:th group.

3.2 Program 2 - Clustering with Deviant Observation

Usually, outlier or deviant observations are simply ignored in the analysis, or, more preferably, removed from the data set prior to the analysis. In this program we allow for deviant observations within the model. The mixture model is extended with one deviant cluster where the observations are assumed to follow a uniform distribution $f_0(\mathbf{y})$ over the whole sample space.

$$f(\mathbf{y}_i\,|\boldsymbol{\theta}) = \sum_{j=1}^{J} \omega_j f_j(\mathbf{y}_i\,|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) + \; p_0 f_0(\mathbf{y})$$

The cluster probabilities satisfy $\sum_{j=1}^{J} \omega_j + \omega_0 = 1$. Theory and application related to this program may be found in Franzén (2007).

The model and prior specifications in Program 1 also apply in this program. One additional entry concerning the deviant cluster needs to be made in the model specifications.

- **Instruction 1**. The program asks for the possible outcomes of the deviant cluster. For discrete data: Give the number of possible outcomes for each variable in vector form for example [10 5 10] means that the first variable, among a total of three, may attain 10 possible values, the second 5 and the last 10. For continuous data: Give instead the interval length of each variable's range.

Priors for the J non-deviant clusters are specified in the same way as they are for the J clusters in Program 1, with one exception. No prior specifications are made on the mean vector and covariance matrix of the deviant cluster, since estimates for this cluster would be uninformative. The size of the deviant cluster, is, however of great interest. Therefore, the vector specifying the cluster proportions will now be of length $J + 1$.

- **Instruction 2**. The program asks for the prior beliefs of the cluster proportions in vector form, i.e. $[\alpha_1\ \alpha_2\ ...\ \alpha_{J+1}]$. The vector is now of length $J + 1$ where the last value corresponds to the deviant cluster. The prior specifications on the last value are usually lower than the rest of the α parameters since we normally expect this deviant cluster to be smaller than the others.

After the program is executed, one may in addition to the automatically presented results obtain information on the observations in the deviant cluster.

>> DEVOBS
Shows the values of those observations where cluster probabilities are the highest for the deviant cluster.

>> PLACE

Shows which observation numbers these observations have.

Instead of assuming a uniform distribution for the deviant cluster, one may assume a normal distribution with a much larger variance than the rest of the clusters. In that case, Program 1 can be used to model the existence of a deviant group. This is simply done by specifying large values on the prior variances in $\mathbf{\Sigma}_j$ for the cluster corresponding to the deviant group.

3.3 Program 3 - Clustering with Missing Data

Missing values are handled as an extra step in the iteration process. Missing values for an observation are replaced by values generated from the normal distribution of which the observation is a member at that iteration step.

The missing variables are denoted NaN in the program. To change numeric values representing missing values (for example 99) in the data matrix Y to NaN, write in the command window

```
>> Y(Y==99)=NaN
```

No additional entries from Program 1 need to be made. The model and prior specifications are specified in the same way. The prior default values of the mean and covariance are now only based on the observations with a complete variable set.

3.4 Program 4 - Longitudinal Clustering

This program clusters data collected at 2 or 3 consecutive time points. At each time point t, data $\mathbf{y}_i^{(t)}$ $\{i = 1, ..., n\}$ is assumed to come from a mixture model of multivariate normal distributions

$$f\left(\mathbf{y}_i^{(t)}\right) = \sum_{j=1}^{J^{(t)}} \omega_j^{(t)} f_j^{(t)} \left(\mathbf{y}_i^{(t)} \middle| \boldsymbol{\mu}_j^{(t)}, \mathbf{\Sigma}_j^{(t)}\right) \qquad i = 1, ..., n$$

where $\omega_j^{(t)}$ is the proportion of objects belonging to Cluster j at Time t and $f_j^{(t)}$ is a multivariate normal density. $J^{(t)}$ denotes the number of clusters at Time t. The mixture model theory is the same as when clustering cross-sectional data. The allocation of objects is however done in a longitudinal manner. An object's classification is determined simultaneously for all time points. Information from all occasions is taken into consideration when determining an object's development pattern. We introduce the transition matrix $\mathbf{Q}_t$, which consists of transition probabilities from clusters at Time t to clusters at Time $t + 1$. Given a cluster membership at Time t corresponding to one row in $\mathbf{Q}_t$, the columns in $\mathbf{Q}_t$ give transition probabilities to all possible clusters at Time $t + 1$. In addition to the cross sectional study one may study transition patterns between time points and also see how cluster structures change. The model allows for the number of clusters and/or the number of variables to differ between time points.

The prior distribution for each row in the transition matrix $\mathbf{Q}_t$ is the Dirichlet distribution

$$\mathbf{Q}_t(j^{(t)},\cdot) \sim Dir(\beta_1^{(t)},...,\beta_{J^{(t)}}^{(t)})$$

where the β parameters have functions equivalent to the α parameters in the Dirichlet distribution for the cluster probabilities.

The posterior distributions for each row in $\mathbf{Q}_t$ is

$$\mathbf{Q}_t(j^{(t)},\cdot)\left|\mathbf{V}^{(t)} \sim Dir\left(\beta_1^{(t)} + n^{(t)}\left(j^{(t)},1\right),...,\beta_{J^{(t)}}^{(t)} + n^{(t)}\left(j^{(t)},J^{(t+1)}\right)\right)\right.$$

where $n^{(t)}(j^{(t)},j^{(t+1)})$ counts the number of transitions from Cluster $j^{(t)}$ to Cluster $j^{(t+1)}$ between Times t and $t+1$ and $\beta_1^{(t)} ... \beta_{J^{(t)}}^{(t)}$ are the parameters from the prior Dirichlet distribution.

For more information on the theory of longitudinal clustering and applications of this particular program see Franzén (2008a).

Except for the addition of the transition matrices $\mathbf{Q}_t$, the model- and prior specifications do not differ much from Program 1. The same specifications have to be made, but now for more than one time point. We specify changes and additions from Program 1 below.

Model specifications

- The program asks for the number of time points, i.e. 2 or 3.
- The program asks for the number of clusters at each time point in vector form, i.e. $\left[J^{(1)}\ J^{(2)}\ J^{(3)}\right]$ if we have data from 3 time points, or else $\left[J^{(1)}\ J^{(2)}\ \right]$.

Prior specifications for μ

- **Instruction 1**. Specify the precision parameters for each cluster in vector form, i.e. $\left[\tau_1^{(t)}\ \tau_2^{(t)} ...\ \tau_J^{(t)}\right]$. The specification is repeated for $t = 1,...,T$.
- **Instruction 2**. Specify the prior beliefs of the mean values $\boldsymbol{\xi}_j^{(t)}$ in matrix form. The size of the matrix has to be $K^{(t)} \times J^{(t)}$. Rows represent variables and columns represent clusters, both at Time t. Each column in the matrix corresponds to the vector $\boldsymbol{\xi}_j^{(t)}$ for Cluster j. Either one types out the whole matrix as shown in Program 1, or if the same value is desired within the same

matrix we simplify and type $0 * ones(D(1), J(1))$. This results in a matrix in the right size (at Time 1) with zeros on all places. Replace the zero when the prior belief is of another magnitude. The specification is repeated for $t = 1, ..., T$.

Prior specifications for Σ

- **Instruction 3**. Specify the degrees of freedom for each cluster in vector form i.e. $\left[m_1^{(t)}\ m_2^{(t)} ...\ m_J^{(t)}\right]$. The specification is repeated for $t = 1, ..., T$.

- **Instruction 4**. Specify the prior belief of the covariance matrices $\Sigma_j^{(t)}$ at Time t, in the same way as in Program 1. The size of the matrix has to be $K^{(t)} \times K^{(t)}$. The specification is repeated for $t = 1, ..., T$.

Prior specifications for Ω

- **Instruction 5**. Specify the prior beliefs of the cluster proportions for the clusters at Time t in vector form, i.e. $\left[\alpha_1^{(1)}\ \alpha_2^{(1)} ...\ \alpha_J^{(1)}\right]$. No specifications are needed for Times 2 and 3 since these probabilities are a direct consequence of the cluster probabilities at Time 1 and the transition matrices specified in the next steps.

Prior specifications for $\mathbf{Q}$

- **Instruction 6**. Specify the prior beliefs of the transition probabilities between Times t and $t + 1$ in matrix form. Note that the number of rows corresponds to the number of clusters at Time t and the number of columns to the number of clusters at Time $t + 1$. The size of the matrix between Time 1 and 2 is $J^{(1)} \times J^{(2)}$ and between Time 2 and 3 (if there is a third point) $J^{(2)} \times J^{(3)}$. Each row is specified unconditional of any other rows. As for the α parameters, the relative sizes of the $\beta_j^{(t)}$ in one row describe the expected proportions between groups, and the sum of the $\beta_j^{(t)}$'s in one row is a measure of the strength of the prior distribution. If there are three time points, the matrix specification is repeated once, for transition between Times 2 and 3.

3.5 Program 5 - Longitudinal Clustering with Missing Values

Longitudinal data in several dimensions are in particular subject to incompleteness. Deleting observations with one or more missing variables at one or more time points may drastically reduce the data set and worsen the result. In the same way as in Program 3, multiple imputation is performed as a step in the iteration process. This time the method is applied to longitudinal data. Franzén (2008b) presents the theory and applies it to simulated and real data.

The entries are the same as in Program 4. Like Program 3, the missing values in the data matrices $Y1$, $Y2$, and possibly $Y3$ have to be encoded NaN.

4 Practical Issues

4.1 Start values

The iteration process successively updates the values in the Markov chain. To get the process started we need a set of start values for all parameters. Start values in the MBCA programs are decided by default by doing a preliminary clustering by the k-means clustering method. The values could be settled in an easier way, for example through a qualified guess or neutral values. The gain from using a more defined method is that the start values probably become closer to their target values and therefore make the Markov chain converge faster. Generally it is best to try several starting points in the state space. If they lead to noticeably different posterior estimates the Markov-chain has not yet converged. The opposite condition, i.e. if one starts at different starting points and ends up in the same region, does not guarantee that the chain has reach its stationary distribution. It may be stuck in a local maximum and will need more iteration runs to eventually find its way out. This means that, within reason, as many iterations T as possible should be chosen.

Changes of the default start values are not straightforward but can be done in any of the Programs P1.m to P5.m. Lines 138-141 in Program P1.m, for example, look like this:

```
M(:,:,1)=M0(:,1:J);
V(1,:)=V0;
Theta(1,:)=Theta0;
Sigma(:,:,1)=Sigma0(:,1:K*J);
```

To change start values, the expressions to the right of the equal signs are in turn replaced by:

- A $K \times J$ matrix where each column represents the start values for one cluster. For example, type $zeros(K, J)$ if all starting mean values are to be 0.
- A $1 \times n$ vector where each value represents the cluster belonging for that corresponding observation. This may be a long vector if n is large, and therefore be time-consuming to type. One may then leave the line unchanged, which means the cluster classifications generated by the k-means are valid.

- A $1 \times J$ vector where each value is the cluster probability for each cluster. The sum of all values has to be 1. For example, type $(1/J) * ones(1, J)$ for equal size of all start values.

- A $K \times (J \cdot K)$ matrix where the first K columns represent the covariance matrix for Cluster 1, the next K columns Cluster 2 and so on. For example, type $repmat(eye(K), 1, J)$ for J identity covariance matrices in a row or $repmat(a * eye(K), 1, J)$ if the value a is desired in the diagonal instead of 1.

The same lines are found on lines 167-170 in Program P2.m and lines 153-156 in Program P3.m. When we are dealing with data from more than one time point, we have to change start values for all time points. The lines in Program P4.m are found on lines 386-394 for Time 1 and 2 and on lines 424-427 if there are three times. For Program P5.m the lines are 424-432 and 462-465.

4.2 Burn-in Period

Usually it takes a number of iteration rounds before the algorithm converges to the desired limiting distribution. The length of the burn-in period, during which the generated values are not representative of the posterior distribution, must be decided. The slower the chain is to converge the longer the burn-in period has to be. Even when starting the chain in the target area, there is no guarantee the burn-in period is unimportant. It will always take some time for the Markov chain to forget its starting position.

There is no guaranteed way to decide the length of the burn-in period, which we denote F in the programs. There are methods of approximation, but we settle for a visual inspection of the iteration output. By studying an iteration plot, one would most often get an idea how many iterations are needed before the chain seems to have reached its stationary distribution. Iteration plots of all the cluster proportions in one graph usually give a good indication. Figure 2 shows the iteration plot for the cluster proportions for a mixture of 4 groups. The burn-in period for these estimates, in this particular run, consists of about 400 iterations. The burn-in period in the next run may be much longer. It is always better to exaggerate the length of the burn-in period than the opposite. The only disadvantage with a longer burn-in period than necessary would be that useful generated values are discarded.

4.3 Label Switching

So-called *label switching* is a well known problem when taking a Bayesian approach to clustering using mixture models. Label switching is the name for the event when

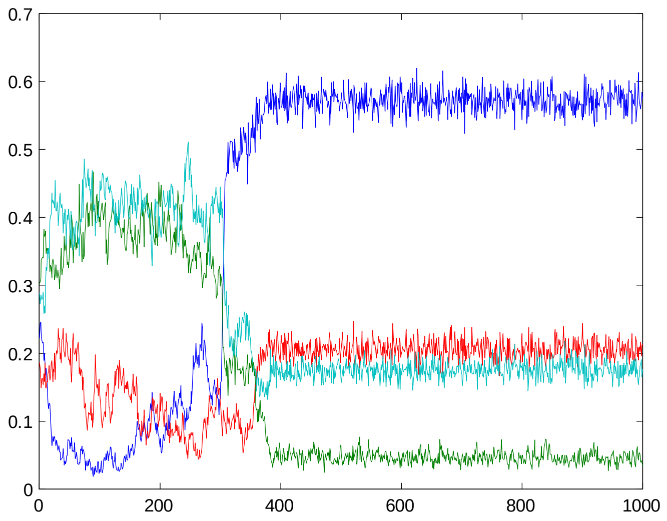

FIGURE 2: Illustration of the burn-in period. For these iterations convergence was reached after about 400 iterations.

Clusters j and $j\prime$ change places during the iteration process. This phenomenon arises because the likelihood

$$L\left(\boldsymbol{\theta}\left|\mathbf{y}\right.\right)=\prod_{j=1}^{n}\left[\omega_1 f\left(y_i\left|\boldsymbol{\mu}_1\boldsymbol{\Sigma}_1\right.\right)+...+\omega_J f\left(y_i\left|\boldsymbol{\mu}_J\boldsymbol{\Sigma}_J\right.\right)\right]$$

is the same for all permutations of clusters. This means the parameters in the model are not *identifiable* by a specific cluster number. If we have no prior information that distinguishes the components of the mixture, i.e. if the priors are the same for all permutations of $\boldsymbol{\theta}$, then the posterior distribution will be similarly symmetric. The same prior distribution for all components of the mixture is usually the case if one has no real prior information about the components.

Label switching can often be detected by studying the iteration plots.

A common solution to the label switching problem is to introduce some identifiability constraints on the parameter space such as $\omega_1 > \omega_2 > .. > \omega_J$ or $\mu_1 > \mu_2 > ... > \mu_J$. The first constraint, where the cluster sizes are ordered, can be included in the programs. The constraints are prepared for by an inactive line in each program. To activate, simply remove the %-sign on rows 188, 220, 257, 561, or 767 depending on which program among P1.m to P5.m is being used. It should be said that this is not a guaranty for eliminating label switching. However, when experiencing label switching, this measure should at least be tried. Stephens (2000) gives an explanation and proposals for other solutions to this particular problem.

4.4 Other Problems

The programs in the MBCA package are prepared for a number of problems and deviations that may occur in the simulation process. However, it is not possible to account for every possible situation that may occur for all types of data set. The program may be interrupted for some reason other than when the user has made a wrong entry. When this happens, one should try to run the program again and see if an odd situation was created by chance. In that case it would probably not be repeated in another run. The whole idea with MCMC simulation is to base the inference on randomness. This is an effective method but may also create unexpected situations.

References

Bernardo, J. M. and Smith, A. F. M. (2000). *Bayesian Theory*, Chichester: John Wiley and Sons.

Biernacki, C., Celeux, G., Govaert, G., and Langrognet, F. (2006). "Model-Based Cluster and Discriminant Analysis with the MIXMOD Software", *Computational Statistics & Data Analysis*,.5, 2, 587-600.

Fraley, C. and Raftery, A. E. (2003). "Enhanced Model-Based Clustering, Density Estimation, and Discriminant Analysis Software: MCLUST", *Journal of Classification*, 20: 263-286.

Fraley, C. and Raftery, A. E. (2006). "MCLUST Version 3 for R: Normal Mixtures Modeling and Model-Based Clustering", *Technical Report no 504*, Department of Statistics, University of Washington.

Fraley, C. and Raftery, A. E. (2007). "Model-based Methods of Classification: Using the MCLUST Software in Chemometrics", *Journal of Statistical Software*, Vol 18, Issue 6.

Franzén, J. (2006). "Bayesian Inference for a Mixture Model using the Gibbs Sampler," Research Report 2006:1, Department of Statistics, Stockholm University.

Franzén, J. (2007). "Classification with the Possibility of a Deviant Group - An Application to Twelve-Year-Old Children," Included Paper in this Thesis.

Franzén, J. (2008a). "Successive Clustering of Longitudinal Data - A Bayesian Approach", Research Report 2008:2, Department of Statistics, Stockholm University.

Franzén, J. (2008b). "Longitudinal, Model-Based Clustering with Missing Data", Research Report 2008:3, Department of Statistics, Stockholm University.

Fryback, D., Stout, N. and Rosenberg, M. (2001). "An Elementary Introduction to Bayesian Computing using WINBUGS", *International Journal of Technology Assessment in Health Care*, 17, 96-113.

Gamerman, D. and Lopes, H. F. (2006). *Markov Chain Monte Carlo - Stochastic Simulation for Bayesian Inference*, second edition. Boka Raton: Chapman & Hall.

Gelman, A., Carlin, J. B., Stern, H. S. and Rubin, D. B. (2004). *Bayesian Data Analysis*, Boca Raton, Chapman & Hall.

Geman, S. and Geman, D. (1984), "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6, 721-741.

Lavine, M. and West, M. (1992), "A Bayesian Method for Classification and Discrimination". *Canadian Journal of Statistics*, 20, 451-461.

Scollnik, D. (2001). "Actuarial Modeling with MCMC and BUGS*"*, *North American Actuarial Journal*, 5, 96-124.

Stephens, M. (2000). "Dealing with Label-switching in Mixture Models," *Journal of the Royal Statistical Society*, Serie B, vol 62, 795-810.

Woodworth, G. (2004). *Biostatistics: A Bayesian Introduction*, Chichester: John Wiley& Sons.