MS Final Project

MODEL-BASED CLUSTERING WITH MEASUREMENT ERRORS

May 20, 2014

Wanli Zhang
Oregon State University
Department of Statistics
zhangwa@stat.oregonstate.edu

1. Introduction

Cluster analysis is the identification of natural groupings of observations that share certain characteristics. Classical, heuristic clustering methods, such as hierarchical agglomerative clustering and k-means clustering, are easy to understand and have been successfully implemented. However, despite considerable research in this area, there are few guidelines for answering basic practical questions that arise in cluster analysis, for instance, how many clusters to choose, which clustering method to use, and how to handle outliers and ties. The lack of knowledge on the statistical properties of the clusters also makes it difficult to measure the appropriateness of the clustering result (Fraley and Raftery, 2002).

Major advances in clustering methodology have been made through the introduction of statistical models, among which the most commonly proposed one is the finite mixture model (Johnson and Wichern, 12.5). In finite mixture models, each component probability distribution corresponds to a cluster. In this context, the problems of choosing the number of clusters and the method of clustering are reduced to the problem of choosing an appropriate statistical model.

Model-based clustering has been applied in numerous fields, including data mining, which started from the search for groupings of customers and products in massive retail datasets; document clustering and the analysis of Web use data; gene expression data from microarrays and more recently, from RNA-Seq. For the very last one, see Si et al. (2014) for an example, where the authors clustered genes using mixed Poisson and Negative Binomial (NB) models on RNA-Seq data.

Fraley and Raftery (2002) describes in great detail the procedures of performing model-based cluster analysis. The multivariate normal mixture model is used and different candidate models are acquired by varying constraints on its parameters. The EM algorithm is used to produce the classification of observations for each candidate model. The Bayesian Information Criteria (BIC) is used for model selection. Section 2 discusses this procedure in more detail.

More than often, observations used in clustering are assumed to have been precisely measured, free of any systematic errors. However, there are situations where this assumption is clearly not feasible. For example, in an RNA-Seq experiment, researchers wish to group genes that are consistent in their expression profiles over time, measured by logarithm of fold change at each time point. The log fold changes themselves, usually being functions of regression coefficient estimates, are inaccurately measured due to errors incurred in the estimation process. A cluster analysis without adjusting for these errors may fail to capture the most appropriate grouping of genes, rendering the results less meaningful or even misleading.

In this paper, we extend the model-based clustering methods described in Fraley and

Raftery (2002) so that measurement errors are properly accounted for in the cluster analysis. Inspired by the R package mclust, developed by C. Fraley, A. Raftery and L. Scrucca, we implemented the extended methods in a smaller scale, that is, only for certain cases of covariance matrix structures.

Section 2 includes a review of model-based clustering method developed by Fraley and Raftery (2002). Section 3 introduces the augmented model and the new method that incorporates estimation errors. Section 4 gives settings and results of simulations investigating algorithm performances. Finally, Section 5 summarizes our findings and suggests possibilities for future work.

2. Model-Based Clustering

2.1. Multivariate Normal Mixture Model

Given data $\boldsymbol{y}$ consisting of an independent random sample $\{y_1, ..., y_n\}$, the likelihood of a *mixture model* is as follows:

$$L_{MIX}(\theta_1, ..., \theta_G; \tau_1, ..., \tau_G|\boldsymbol{y}) = \prod_{i=1}^{n} \sum_{k=1}^{G} \tau_k f_k(y_i|\theta_k) \tag{1}$$

where f_k and θ_k are the density and parameters of the kth component in the mixture and τ_k is the probability that an observation belongs to the kth component, subject to $\sum_{k=1}^{G} \tau_k = 1$.

The most commonly used candidate for f_k is the multivariate normal distribution, whose PDF is denoted by ϕ_k and parametrized by mean μ_k and covariance matrix Σ_k as follows:

$$\phi_k\left(y_i|\mu_k, \Sigma_k\right) = \frac{\exp\left\{-\frac{1}{2}(y_i - \mu_k)^T \Sigma_k^{-1}(y_i - \mu_k)\right\}}{\sqrt{\det(2\pi\Sigma_k)}} \tag{2}$$

Data generated by this model will appear as ellipsoidal clusters, each of which corresponds to a component in (1). The kth cluster is centered at μ_k, while its shape, orientation and volume are characterized by Σ_k.

Banfield and Raftery (1993) used eigendecomposition to rewrite Σ_k in the following form:

$$\Sigma_k = \lambda_k D_k A_k D_k^T \tag{3}$$

where D_k is the orthogonal matrix of eigenvectors, A_k is the diagonal matrix of scaled eigenvalues and λ_k is the factor of scaling.

The parametrization above allows us to model the geometric properties of each cluster. In fact, D_k governs the orientation of the kth cluster, A_k its shape, and λ_k its volume.

Treating them as three independent sets of parameters, and allowing them to vary or constraining them to remain the same across clusters, we obtain parsimonious and easily interpreted models which are appropriate to describe various clustering situations.

Celeux and Govaert (1995) describes in detail a total number of fourteen different models acquired using the described method. Fraley et al., (2007) implemented, as part of R library mclust, ten parametrizations for multivariate data plus two for univariate, listed in Table 2.1:

Identifier	Model	Distribution	Volume	Shape	Orientation
`E`		(univariate)	equal		
`V`		(univariate)	variable		
`EII`	λI	Spherical	equal	equal	NA
`VII`	$\lambda_k I$	Spherical	variable	equal	NA
`EEI`	λA	Diagonal	equal	equal	coordinate axes
`VEI`	$\lambda_k A$	Diagonal	variable	equal	coordinate axes
`VVI`	$\lambda_k A_k$	Diagonal	variable	variable	coordinate axes
`EEE`	$\lambda D A D^T$	Ellipsoidal	equal	equal	equal
`EEV`	$\lambda D_k A D_k^T$	Ellipsoidal	equal	equal	variable
`VEV`	$\lambda_k D_k A D_k^T$	Ellipsoidal	variable	equal	variable
`VVV`	$\lambda_k D_k A_k D_k^T$	Ellipsoidal	variable	variable	variable

Table 2.1: Parametrizations of the covariance matrix Σ_k currently available in MCLUST for EM

2.2. The Expectation-Maximization Algorithm for Mixture Models

The EM algorithm (Dempster et al., 1977) is the most commonly used method for obtaining a numerical solution of the MLEs for parameters τ_k and θ_k that maximize (1). Let $x_i = (y_i, z_i)$ be the complete data, where y_i is observed data and $z_i = (z_{i1}, ..., z_{iG})$ is the unobserved group membership, with

$$z_{ik} = \begin{cases} 1 \text{ if } x_i \text{ belongs to group } k \\ 0 \text{ otherwise} \end{cases}$$

The complete data log likelihood is:

$$l(\theta_k, \tau_k, z_{ik}|x) = \sum_{i=1}^{n} \sum_{k=1}^{G} z_{ik} \log[\tau_k f_k(y_i|\theta_k)] \tag{4}$$

The *E step* produces an estimate for the unobserved membership, z_i, with the following:

$$\hat{z}_{ik} \leftarrow \frac{\hat{\tau}_k f_k(y_i|\hat{\theta}_k)}{\sum_{j=1}^{G} \hat{\tau}_j f_j(y_i|\hat{\theta}_j)}$$

while the *M step* maximizes the complete data log likelihood (4) in terms of τ_k and θ_k, with z_{ik} fixed at the values obtained from the E step. To initialize the EM algorithm, we will first perform an initial clustering procedure using hierarchical clustering to acquire an initial classification, i.e. initial values for z_{ik}'s. And then iterate until convergence.

Upon convergence, a classification rule is established in the following fashion: Let z^*_{ik} denote the value of z_{ik} upon convergence, and then assign observation i to group j such that $z^*_{ij} = \max_k(z^*_{ik})$.

For multivariate normal mixture models, the M step involves estimating μ_k, τ_k and Σ_k for $k = 1, ..., G$. Estimators for μ_k and τ_k can be easily found, and they all have analytic solutions:

$$\hat{\tau}_k \leftarrow \frac{n_k}{n}; \quad \hat{\mu}_k \leftarrow \frac{\sum_{i=1}^{n} \hat{z}_{ik} y_i}{n_k}; \quad n_k \leftarrow \sum_{i=1}^{n} \hat{z}_{ik} \tag{5}$$

Computation of the covariance MLE depends on its parametrization. In some situations, such as when we restrict all components to share the same covariance matrix (EEE), or when there are no restrictions at all (VVV), MLEs for the covariance matrices have closed-form solutions. In other cases, an iterative procedure is sometimes required. See Celeux and Govaert (1995) for a more detailed discussion on the estimation procedures.

2.3. Model Selection

For different values of G and different constraints on Σ_k, the problems of choosing an appropriate clustering method and choosing the number of clusters have been reduced to the problem of selecting an appropriate model. Fraley and Raftery (2002) suggests using the BIC as the information criteria, defined as follows:

Suppose that several models, $M_1, ..., M_K$ are considered, given data D, then the *BIC* for model M_k is given by

$$BIC_k = 2 \log p(D|\hat{\theta}_k, M_k) - \nu_k \log(n) \tag{6}$$

where ν_k is the number of independent parameters to be estimated in model M_k, and $p(\cdot)$ denotes the observed likelihood.

In the context of model-based clustering, the BIC of each model is computed from the observed likelihood, upon convergence of the EM algorithm. Model with the largest value of BIC is considered to produce the most appropriate grouping of observations.

2.4. Cluster Analysis

Fraley and Raftery (2002) summarizes the model-based cluster analysis into the following four steps:

1. Determine a *maximum* number of clusters, M, and a set of mixture models to consider, as listed in Table 2.1.
2. Perform hierarchical agglomeration (or some other heuristic procedures) to obtain the corresponding initial classifications up to M groups.
3. Apply the EM algorithm for each model and each number of clusters $2, ..., M$, starting with the classification from Step 2.
4. Compute BIC for the one-cluster case for each model and for the mixture model with the optimal parameters from EM for $2, ..., M$ clusters.

The model as well as the number of clusters will then be determined by selecting the model with the maximum BIC. Parametrizations of the covariance matrix as in Table 2.1 provide us with a good set of candidate models for clustering. With these models, computation can be saved by performing Step 2 for only one model, and using the resulting partitions as initial classification for the EM algorithm with other parametrizations.

3. Incorporating Measurement Error into EM Algorithm

In this section, we propose a model that accounts for measurement errors, and also describe details in its implementation process in R.

3.1. Augmented Model

Firstly, we specify the model that incorporates the measurement error. For each observed value y_i, assume that there exists a latent variable w_i, representing the "true" value of y_i, such that:

$$y_i|w_i \sim N(w_i, E_i)$$
$$w_i|\theta_k \sim N(\mu_k, \Sigma_k)$$

where the measurement error E_i is either known or can be easily estimated, and μ_k and Σ_k are unknown. Notice that in the case where E_i needs to be estimated, the variability of this estimation process is not taken into account, and may serve as a future research topic.

From above, it can be shown that:

$$y_i|\theta_k \sim N(\mu_k, \Sigma_k + E_i)$$

The density for y_i is then:

$$\phi_k^{(a)}(y_i|\mu_k, \Sigma_k, \Sigma_i) = \frac{\exp\left\{-\frac{1}{2}(y_i - \mu_k)^T (\Sigma_k + E_i)^{-1} (y_i - \mu_k)\right\}}{\sqrt{\det[2\pi(\Sigma_k + E_i)]}} \tag{7}$$

Substituting f_k in (1) by $\phi_k^{(a)}$ above, the (observed) likelihood for our augmented model with G components is thus

$$L_{MIX}^{(a)}(\theta_1, ..., \theta_G; \tau_1, ..., \tau_G|\boldsymbol{y}) = \prod_{i=1}^{n} \sum_{k=1}^{G} \tau_k \phi_k^{(a)}(y_i|\theta_k)$$

The EM algorithm is then modified so that the following complete data log-likelihood, instead of (4), is maximized in each M-step:

$$l^{(a)}(\theta_k, \tau_k, z_{ik}|x) = \sum_{i=1}^{n} \sum_{k=1}^{G} z_{ik} \log[\tau_k \phi_k^{(a)}(y_i|\theta_k)] \tag{8}$$

where z_{ik} is defined the same way as in Section 2.2. Notice that MLEs for μ_k and Σ_k no longer have analytic forms, and have to be acquired using numerical methods. See Section A of the Appendix for an (unsuccessful) attempt to reach an analytic solution.

The E-step is modified accordingly as follows:

$$\hat{z}_{ik} = \frac{\hat{\tau}_k \phi_k^{(a)}(y_i|\hat{\theta}_k)}{\sum_{j=1}^{G} \hat{\tau}_j \phi_k^{(a)}(y_i|\hat{\theta}_j)} \tag{9}$$

An initial classification is acquired using model-based hierarchical clustering. Then an M-step is run to obtain parameter estimates, which will in turn be used in E-step to compute the membership probabilities. The two steps are iterated until there is no more (or minimal) increase in the observed likelihood.

3.2. Implementation

During the time available to us, we implemented one part of the algorithm, which is the case when no constraints are imposed on covariance structures, or "VVV" as in

Table 2.1.

To calculate MLEs for μ_k and Σ_k in the M-step, we used Limited-memory BFGS method with box constraints, offered as an option of general purpose optimizer `optim` available in R. The choice of initial values for this algorithm is more complicated than we originally expected. Through trial and error, we summarize the findings into the following situations:

1. When $E_i = E$, there exists a closed-form solution to Σ_k:

$$\hat{\Sigma}_k = \frac{W_k}{n_k} - E \tag{10}$$

where $W_k = \sum_i z_{ik}(y_i - \mu_k)(y_i - \mu_k)^T$ is the within-cluster scatter matrix.

It would then make sense to initiate each M-step with the latest updated $\frac{W_k}{n_k} - E$, so that the results are equivalent to those produced by meVVV, with a difference of E. Through numerical experiments, this equivalence holds for every iteration.

However, in our simulation (see next section), the final covariance estimates upon convergence do not satisfy the above relationship perfectly. We make the observation that the numbers of iterations for the two methods to converge are different, but we are not certain at this point what causes this difference.

2. When E_i's are the same within each cluster, that is, after re-indexing, we have G unique measurement errors $E_1, ..., E_G$, the covariance matrix Σ_k also has a closed-form solution:

$$\hat{\Sigma}_k = \frac{W_k}{n_k} - E_k \tag{11}$$

Notice that (10) is a special case of (11).

In this situation, similar to 1, each M-step is initiated with the latest updated $\frac{W_k}{n_k} - E_k$. Through numerical experiments, we could show that covariance estimates of the two methods are equivalent at each iteration. However, again, we observe discrepancies between the final converged covariance estimates and those produced by meVVV. See the next section for more details.

3. When all E_i's are potentially different, there is no closed-form solution to Σ_k, and in this situation, we simply fixed a single initial value (identity) without updating. We are not certain whether the results are reliable or not. There is much room for improvement and this could serve as an interesting research topic in the future.

4. Simulations

In this section, we investigate the performance of our algorithm in several aspects. Some simulations involve the concept of misclassification rate (MCR), defined as proportion of wrongfully classified objects among all classifications. For all simulations, the true membership matrix is used as the initial classification, in the hope of obtaining a faster convergence. Whenever a comparison is made, we compare our method to function meVVV from library mclust in R, based on which we developed our new method. Also the true error matrices are provided for all runs using the new method. For Tables B.1∼B.4 and Figure B.1, please refer to Appendix B

4.1. Verification of Clustering Results

We simulated normal mixture data with no measurement errors, and did cluster analysis using both our method and function `meVVV` from mclust library. In Table B.1, we can observe that the results are exactly the same, proving that our method is valid at least in this setting.

4.2. Effect of Increasing Error on MCR and Run Time

We simulated mixed multivariate normal data with the same mean, mixing proportion and covariance matrices, and added diagonal error matrices of increasing magnitudes. The goal here is to examine how misclassification rate changes as error goes up.

Setting: 100 observations. Error matrix $E = \epsilon I$, where $\epsilon \in \{0, 5, 10, 15, 20, 30\}$ and I is identity matrix.

From Table 4.1, we can observe that in terms of MCR, our method performs slightly better than `meVVV` when $\epsilon = 15$ and 20; slightly worse when $\epsilon = 10$ and 30; as well as `meVVV` when $\epsilon = 0$ and 5. Notice that each of the MCRs is based on only one cluster analysis.

	MCR	
ϵ	new method	meVVV
0	0.01	0.01
5	0.06	0.06
10	0.10	0.08
15	0.16	0.17
20	0.14	0.19
30	0.26	0.19

Table 4.1: Comparison of MCR for two methods, with varying errors

Table 4.2 demonstrates actual run time when ϵ takes on different values. In general, the running time of our method increases with the value of ϵ in approximately a linear fashion.

ϵ	Time (in seconds)
0	95.08
5	514.57
10	1550.46
15	2069.20
20	3035.61
30	5913.25

Table 4.2: Relationship between running time and magnitude of error

4.3. Effect of Varying Error Structures on MCR

In this simulation, we designed 3 scenarios where the error structures are different in each one.

S1: Identical errors for all observations. This might happen if all measurements are done using one faulty instrument.

S2: Identical errors within each cluster. This might happen if data are from different sources, and a different measuring instrument is used for each source.

S3: Each observation possesses its own error. This is possible if, for example, each observation is a regression coefficient estimate, each with its own estimation error.

Settings for each scenario:

S1: All error matrices are set to be $E = 0.1I$.

S2: Since three clusters are generated, we use E_1, E_2 and E_3 as error matrices for each cluster respectively. Their values are:

$$E_1 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad E_2 = \begin{bmatrix} 0.3 & 0 \\ 0 & 0.3 \end{bmatrix}, \quad E_3 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

S3: Absolute values of randomly generated normal samples are used to construct error matrices for each observation through backwards Cholesky decomposition.

Table 4.3 below displays the misclassification rates of two methods under all three scenarios. Observe that both methods misclassified the same proportion of observations in all cases. In fact, in each scenario, the same observations were mistakenly grouped by both methods. This is not surprising because we do not expect the error matrices, which are relatively small in magnitude, would have much impact on the clustering results.

The MCR in S3 being the smallest of three is a little unexpected, but can be justified by speculating that it could be due to chance that some observations that are misclassified in S1 and S2 are "pulled" back towards their own group, as an effect of the distinct error matrices added. This speculation is further supported by Figure B.1, where observations in group 1 and 2 that were commingled in S1 and S2 have been separated, while a single observation in group 2 is pulled for a great distance towards group 3.

Scenario	MCR	
	new method	meVVV
1	0.06	0.06
2	0.06	0.06
3	0.02	0.02

Table 4.3: Comparison of MCR in three scenarios for two methods

Tables B.2, B.3 and B.4 display the parameter estimates for both our method and meVVV. As mentioned in Section 3.2, results of S1 and S2 are not perfectly equivalent as we anticipated, partly due to the number of iterations till convergence being different for the two methods. Although we have set the tolerance level of our method to be the same as that of meVVV, the number of iterations is still difficult to control.

In Table B.4, observe that neither method gives good estimates for covariance matrices, but mean and proportion estimates are reasonably good. Also notice that variances are greatly overestimated in group 3, but underestimated in group 2. Correspondingly, in Figure B.1, we can observe that group 3 has greater overall spread in this scenario than in the other two, while the spread of group 2 has become smaller.

4.4. Effect of Dimension on Running Time

We simulated datasets with increasing dimensions, and the goal is to investigate how fast the required running time changes with dimension. For each case, we generated samples with size fixed at 10, from 3 clusters with the same mixing proportions. As dimension increases, we randomly adjust the means and covariance matrices of our generating distribution.

As shown in Table 4.4, the order of magnitude of running time increases by 2 when dimension increases from 3 to 4. This observation motivates us to search for ways to speed up our algorithm (e.g. rewrite loops and matrix inversion in C/C++).

Dimension	Time (in seconds)
2	49.79
3	39.90
4	1078.88
5	1009.56

Table 4.4: Relationship between running time of augmented algorithm and dimension of data

4.5. Effect of Sample Size on Running Time

We generated data from the same mixing normal distribution, but with increasing sample sizes. The goal here is to assess how running time changes as sample size increases, hence to be able to estimate beforehand the time required to analyze datasets with a larger size.

Figure 4.5 suggests an approximate linear increase in running time as sample size goes up. If sample size is increased by 1, we expect the actual run time to increase by roughly 40 seconds, on average.

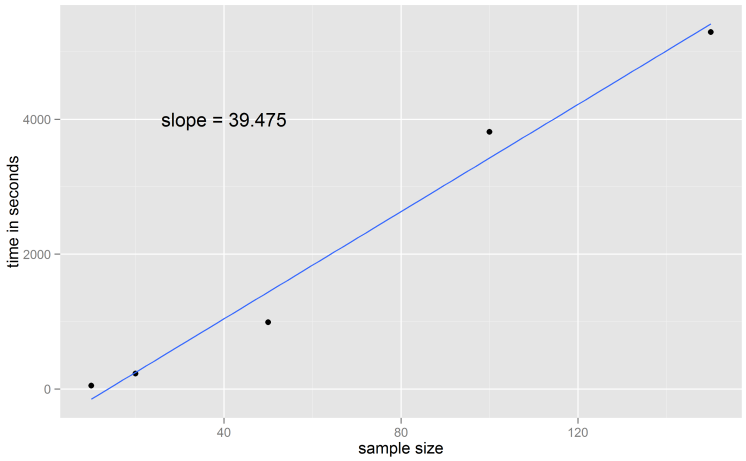

Figure 4.1: Running time vs sample size, with linear fit

5. Conclusions

To account for measurement errors in observations in the context of model-based clustering, we developed a new model and implemented a special case (VVV) with R. We simulated multiple datasets to assess the performance of our new method under different settings.

To recapitulate our findings, first of all, when there are in fact no measurement errors, our method produces exactly the same results as that of meVVV, a function in an existing R library, mclust. When the errors are the same across all observations, or when errors are homogeneous within each cluster, we expect the results, in particular, covariance matrix estimates of the two methods to be equivalent, in the sense that the difference between the estimates is exactly the error matrix. However, simulation results suggest that this equivalence is difficult to acquire, partly due to the difference in the number of iterations required for convergence.

We have also considered the change in actual run time as sample size, magnitude of error, and dimension of data are increased. Based on our simulation, running time increases almost linearly with both sample size and error magnitude (ϵ as in $E = \epsilon I$). Dimension has a huge impact on running time: our method drastically slows down as dimension of data is increased from 3 to 4.

Before we can implement other cases of covariance structures, we should firstly consider improving the efficiency of our method by rewriting parts of the algorithm in C/C++ to try avoiding loops in R. Another issue is the choice of initial values for L-BFGS-B algorithm in the M-step when errors are different for all observations (S3). Unlike S1 and S2, since there are no existing "gold standards" to compare to in this situation, we hope that through trial and error we'll be able to find the appropriate starting values. Finally, as mentioned earlier in Section 3.1, when E_i themselves are estimated, a potentially different model will need to be used to account for the extra variation.

A. Implementation details of augmented M-step

Here we show the difficulty in obtaining a closed-form solution of MLEs for μ_k and Σ_k that maximize the augmented complete data log-likelihood (8).

Substituting the expression (7) into (8), we obtain the following:

$$\begin{aligned} l^{(a)}(\theta_k, \tau_k, z_{ik}|x) = &\sum_i \sum_k z_{ik} \log(\tau_k) - \frac{np}{2} \log(2\pi) - \frac{1}{2} \sum_i \sum_k z_{ik} \log|\Sigma_k + E_i| \\ &- \frac{1}{2} \sum_i \sum_k z_{ik} (x_i - \mu_k)^T (\Sigma_k + E_i)^{-1} (x_i - \mu_k) \end{aligned} \tag{12}$$

Firstly, we find MLE for τ_k using Langrange's multiplier. Since $\sum_i \sum_k z_{ik} \log(\tau_k)$ is the only term involved with τ_k, the objective function is

$$f(\tau_1, ..., \tau_G) = \sum_i \sum_k z_{ik} \log(\tau_k) = \sum_k n_k \log(\tau_k)$$

subject to the constraint

$$g(\tau_1, ..., \tau_G) = \sum_k \tau_k - 1 = 0$$

Let $\Phi(\tau_1, ..., \tau_G, \lambda) = f + \lambda g$. Taking partial derivative with respect to each parameter, we have:

$$\begin{aligned} &\frac{\partial}{\partial \tau_1} \Phi = \frac{n_1}{\tau_1} + \lambda = 0 \\ &\cdots \\ &\frac{\partial}{\partial \tau_G} \Phi = \frac{n_G}{\tau_G} + \lambda = 0 \\ &\frac{\partial}{\partial \lambda} \Phi = \sum_k \tau_k - 1 = 0 \end{aligned}$$

Solving for τ_k and λ, we have:

$$\lambda = -\sum_k n_k = -n, \quad \hat{\tau}_k = -\frac{n_k}{\lambda} = \frac{n_k}{n}$$

which is the same as the result in (5).

Next, we attempt to find $\hat{\mu}_k$. Start by taking partial derivative of $l^{(a)}$ with respect to μ_k. Here we used formula (11.6) from Dwyer (1967). Setting the resulting derivative as zero, we have:

$$\frac{\partial}{\partial \mu_k} l^{(a)} = \sum_i z_{ik} (\Sigma_k + E_i)^{-1} (x_i - \mu_k) = 0$$

$$\Longrightarrow \left[\sum_i z_{ik} (\Sigma_k + E_i)^{-1} \right] \mu_k = B\mu_k = \sum_i z_{ik} (\Sigma_k + E_i)^{-1} x_i$$

If we assume B is positive definite, then there exists a unique solution for μ_k:

$$\hat{\mu}_k = B^{-1} \sum_i z_{ik} (\Sigma_k + E_i)^{-1} x_i = \left[\sum_i z_{ik} (\Sigma_k + E_i)^{-1} \right]^{-1} \sum_i z_{ik} (\Sigma_k + E_i)^{-1} x_i$$

which involves the unknown quantity Σ_k. Notice that when $E_i = 0$, $\hat{\mu}_k$ reduces to the solution in (5).

Finally, we find the MLE for Σ_k. We accomplish this through the profile likelihood method, replacing μ_k in (12) by $\hat{\mu}_k$ above and maximizing the resulting likelihood function. Using formulae (11.7) and (11.8) from Dwyer (1967), the partial derivative of $l^{(a)}$ with respect to Σ_k is:

$$\begin{aligned} \frac{\partial}{\partial \Sigma_k} l^{(a)} = & \frac{1}{2} \sum_i z_{ik} (\Sigma_k + E_i)^{-1} (x_i - \mu_k)(x_i - \hat{\mu}_k)^T (\Sigma_k + E_i)^{-1} \\ & - \frac{1}{2} \sum_i z_{ik} (\Sigma_k + E_i)^{-1} \end{aligned} \tag{13}$$

Equation (13) is generally difficult to solve. In our implementation, we used the Limited-memory BFGS, a quasi-Newton method, to obtain an optimal solution for Σ_k numerically, and then obtain $\hat{\mu}_k$ by substituting in $\hat{\Sigma}_k$.

B. Tables and Graphs

	new method	meVVV	True
$\hat{\mu}$	$\begin{bmatrix} 1.045 & 9.928 & -25.237 \\ 1.038 & -10.021 & 24.574 \end{bmatrix}$	$\begin{bmatrix} 1.045 & 9.928 & -25.237 \\ 1.038 & -10.021 & 24.574 \end{bmatrix}$	$\begin{bmatrix} 1 & 10 & -25 \\ 1 & -10 & 25 \end{bmatrix}$
$\hat{\tau}$	$(0.3, 0.3, 0.4)$	$(0.3, 0.3, 0.4)$	$(0.3, 0.3, 0.4)$
$\hat{\Sigma}_1$	$\begin{bmatrix} 3.073 & -0.041 \\ -0.041 & 2.423 \end{bmatrix}$	$\begin{bmatrix} 3.073 & -0.041 \\ -0.041 & 2.423 \end{bmatrix}$	$\begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$
$\hat{\Sigma}_2$	$\begin{bmatrix} 4.008 & 0.360 \\ 0.360 & 2.557 \end{bmatrix}$	$\begin{bmatrix} 4.008 & 0.360 \\ 0.360 & 2.557 \end{bmatrix}$	$\begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}$
$\hat{\Sigma}_3$	$\begin{bmatrix} 6.631 & 0.107 \\ 0.107 & 6.300 \end{bmatrix}$	$\begin{bmatrix} 6.631 & 0.107 \\ 0.107 & 6.300 \end{bmatrix}$	$\begin{bmatrix} 7 & 0 \\ 0 & 7 \end{bmatrix}$
loglik	-532.1471	-532.1471	

Table B.1: Comparison of results in error-free setting.

	our method	meVVV	True
$\hat{\mu}$	$\begin{bmatrix} 1.784 & 12.934 & -25.360 \\ -0.488 & -10.973 & 23.351 \end{bmatrix}$	$\begin{bmatrix} 1.690 & 12.866 & -25.360 \\ -0.371 & -10.959 & 23.351 \end{bmatrix}$	$\begin{bmatrix} 1 & 10 & -25 \\ 1 & -10 & 25 \end{bmatrix}$
$\hat{\tau}$	$(0.383, 0.217, 0.400)$	$(0.378, 0.222, 0.400)$	$(0.3, 0.3, 0.4)$
$\hat{\Sigma}_1$	$\begin{bmatrix} 14.487 & -13.480 \\ -13.480 & 27.689 \end{bmatrix}$	$\begin{bmatrix} 13.999 & -12.713 \\ -12.713 & 26.833 \end{bmatrix}$	$\begin{bmatrix} 15 & -2 \\ -2 & 15 \end{bmatrix}$
$\hat{\Sigma}_2$	$\begin{bmatrix} 5.780 & 4.714 \\ 4.714 & 29.737 \end{bmatrix}$	$\begin{bmatrix} 6.019 & 4.570 \\ 4.570 & 29.455 \end{bmatrix}$	$\begin{bmatrix} 23 & 3 \\ 3 & 23 \end{bmatrix}$
$\hat{\Sigma}_3$	$\begin{bmatrix} 11.449 & 0.334 \\ 0.334 & 19.189 \end{bmatrix}$	$\begin{bmatrix} 11.549 & 0.334 \\ 0.334 & 19.289 \end{bmatrix}$	$\begin{bmatrix} 31 & -4 \\ -4 & 31 \end{bmatrix}$
loglik	-325.909	-325.915	
iterations	30	14	

Table B.2: [Scenario 1] Comparison of parameter estimates. Error matrix $E = 0.1I$.

	our method	meVVV	True
$\hat{\mu}$	$\begin{bmatrix} 1.791 & 12.954 & -25.364 \\ -0.482 & -10.989 & 23.340 \end{bmatrix}$	$\begin{bmatrix} 1.685 & 12.879 & -25.364 \\ -0.368 & -10.966 & 23.340 \end{bmatrix}$	$\begin{bmatrix} 1 & 10 & -25 \\ 1 & -10 & 25 \end{bmatrix}$
$\hat{\tau}$	$(0.383, 0.217, 0.400)$	$(0.378, 0.222, 0.400)$	$(0.3, 0.3, 0.4)$
$\hat{\Sigma}_1$	$\begin{bmatrix} 14.524 & -13.466 \\ -13.466 & 27.544 \end{bmatrix}$	$\begin{bmatrix} 13.983 & -12.675 \\ -12.675 & 26.798 \end{bmatrix}$	$\begin{bmatrix} 15 & -2 \\ -2 & 15 \end{bmatrix}$
$\hat{\Sigma}_2$	$\begin{bmatrix} 5.618 & 4.770 \\ 4.770 & 29.821 \end{bmatrix}$	$\begin{bmatrix} 6.066 & 4.592 \\ 4.592 & 29.712 \end{bmatrix}$	$\begin{bmatrix} 23 & 3 \\ 3 & 23 \end{bmatrix}$
$\hat{\Sigma}_3$	$\begin{bmatrix} 11.201 & 0.364 \\ 0.364 & 19.042 \end{bmatrix}$	$\begin{bmatrix} 11.701 & 0.364 \\ 0.364 & 19.542 \end{bmatrix}$	$\begin{bmatrix} 31 & -4 \\ -4 & 31 \end{bmatrix}$
loglik	-326.269	-326.278	
iterations	31	14	

Table B.3: [Scenario 2] Comparison of parameter estimates. Error matrices E_1, E_2 and E_3 are defined as before.

	our method	meVVV	True
$\hat{\mu}$	$\begin{bmatrix} 0.230 & 7.270 & -24.596 \\ 0.407 & -12.098 & 24.727 \end{bmatrix}$	$\begin{bmatrix} 0.712 & 7.302 & -24.596 \\ 0.642 & -12.167 & 24.681 \end{bmatrix}$	$\begin{bmatrix} 1 & 10 & -25 \\ 1 & -10 & 25 \end{bmatrix}$
$\hat{\tau}$	$(0.302, 0.278, 0.420)$	$(0.302, 0.278, 0.420)$	$(0.3, 0.3, 0.4)$
$\hat{\Sigma}_1$	$\begin{bmatrix} 9.317 & -5.442 \\ -5.442 & 8.272 \end{bmatrix}$	$\begin{bmatrix} 11.464 & -2.055 \\ -2.055 & 12.113 \end{bmatrix}$	$\begin{bmatrix} 15 & -2 \\ -2 & 15 \end{bmatrix}$
$\hat{\Sigma}_2$	$\begin{bmatrix} 11.398 & 1.511 \\ 1.511 & 6.473 \end{bmatrix}$	$\begin{bmatrix} 11.404 & 2.436 \\ 2.436 & 8.005 \end{bmatrix}$	$\begin{bmatrix} 23 & 3 \\ 3 & 23 \end{bmatrix}$
$\hat{\Sigma}_3$	$\begin{bmatrix} 43.984 & 4.413 \\ 4.413 & 42.856 \end{bmatrix}$	$\begin{bmatrix} 46.028 & 4.739 \\ 4.739 & 46.192 \end{bmatrix}$	$\begin{bmatrix} 31 & -4 \\ -4 & 31 \end{bmatrix}$
loglik	-343.227	-343.497	
iterations	23	22	

Table B.4: [Scenario 3] Comparison of parameter estimates. All observations have different errors.

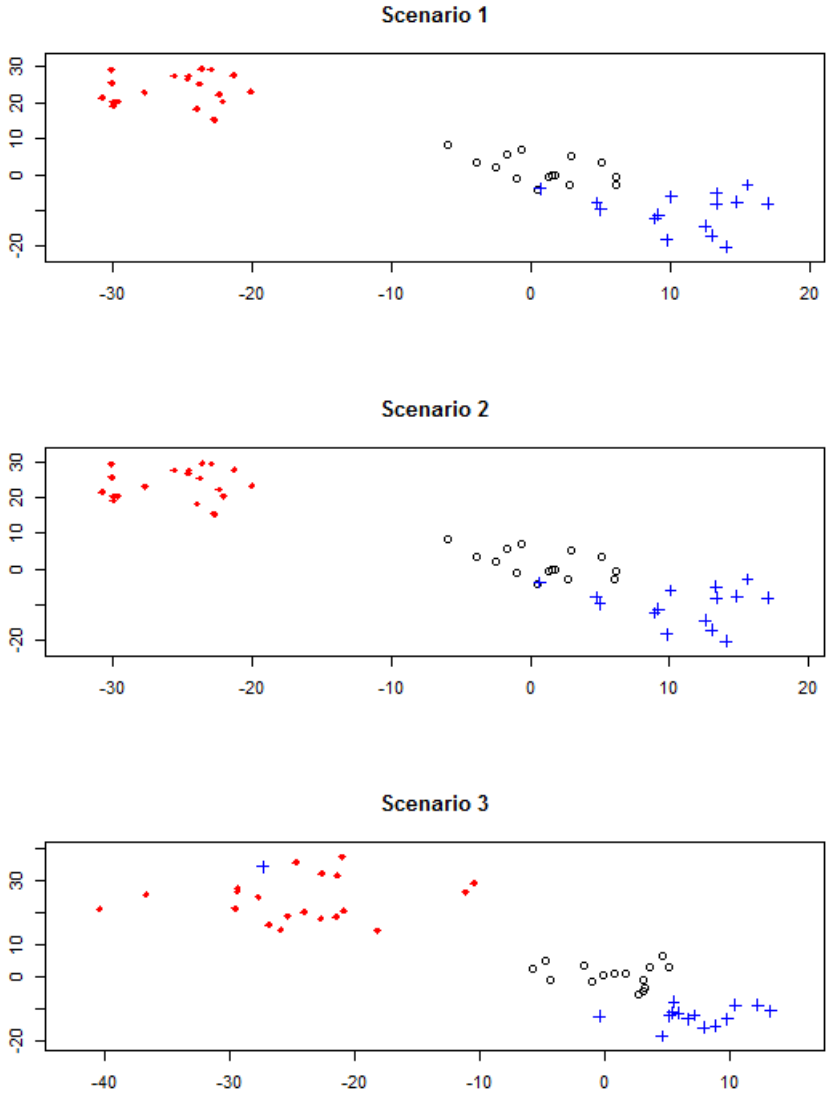

Figure B.1: Sample scatter plots for all 3 scenarios

References

[1] Banfiled, J. D., and A. E. Raftery (1993): "Model-based Gaussian and non-Gaussian clustering," *Biometrics*, 49, 803-821

[2] Dempster, A. P., N. M. Laird and D. B. Rubin (1977): "Maximum likelihood for incomplete data via the EM algorithm (with discussion)," *Journal of the Royal Statistical Society*, Ser. B, 39, 1-38

[3] Fraley, C. and A. E. Raftery (2002): "Model-based clustering, discriminant analysis and density estimation," *Journal of the American Statistical Association*, 97, 611-631

[4] Fraley, C. and A. E. Raftery (2007): "Model-based methods of classification: using the mclust software in chemometrics," *Journal of Statistical Software*, 18

[5] Celeux, G. and G. Govaert (1995): "Gaussian parsimonious clustering models," *Patter Recognition*, 28, 781-793

[6] Johnson, R. A. and D. W. Wichern (2013): *Applied Multivariate Statistical Analysis*, Delhi, PHI Learning Private Limited

[7] Si, Y., P. Liu, P. Li and T. P. Brutnell (2014): "Model-based clustering for RNA-seq data," *Bioinformatics*, 30, 197-205