Technische Universität München

Department of Mathematics

Master's Thesis

EM algorithm and its extensions for Gaussian and vine copula mixture models

Ka Wing Ho

Supervisor: Prof. Claudia Czado

Advisor: Prof. Claudia Czado, Özge Sahin

Submission Date: December 15, 2021

I assure the single handed composition of this master's thesis only supported by declared resources.

Garching,

Acknowledgement

I would like to express my gratitude to my thesis supervisor Prof. Claudia Czado, that I have this valuable opportunity to write my master's thesis at the Chair of Mathematical Statistics. In the last six months, we had a lot of meetings and she can always give me useful advice and feedback, so that I can keep working on and improve my thesis. Furthermore, I would like to thank my advisor Özge Sahin very much for all the guidance, her support and help in the programming for simulation. Every time I ask her questions about the thesis through email, she always replies me very soon with useful answers, even in the evening and on non-working days. Last but not least, I thank my family and my friends for their constant support on my thesis and my study in TUM.

Abstract

In the 21th century, the importance of finite mixture models in the statistical analysis of data keeps increasing, so that the number of articles on mixture model applications appearing in the statistical and general scientific literature increases steadily, for example, cluster analysis, unsupervised pattern recognition, speech recognition, medical imaging and other applications.

The expectation and maximization (EM) algorithm is a well-known and convenient way for parameter estimation in mixture models. However, the EM algorithm is an iterative algorithm requiring starting values. Different starting values for the EM algorithm can significantly impact the resulting solution. In addition to initialization strategies, the extensions of EM algorithm with different iteration processes from classic EM algorithm also affect the resulting solution. Furthermore, the majority of finite mixture models cannot capture clusters, which are non-elliptical and asymmetric tail dependencies. Due to the higher flexibility of vine copulas, the vine copula mixture model (VCMM) algorithm proposed by Sahin and Czado [2021] is more suitable for modelling Non-Gaussian multivariate data with clusters.

This thesis aims to study the performance of the classic Gaussian mixture model (GMM) algorithm and the vine copulas mixture model (VCMM) algorithm with different EM algorithms and initialization strategies for clustering data with different characteristics and real data sets. According to our results, clustering Gaussian data with GMM algorithm using different EM algorithms and initialization strategies both have a significant effect on classification rate. For the best fit, we recommend the expectation/conditional maximisation either (ECME) algorithm together with the optimization method Nelder-Mead and initialization by k-means clustering. However, the is not the case in VCMM algorithm. For clustering data with various characteristics and real data sets by VCMM algorithm, different EM algorithms don't affect the performance regarding to classification rate for clustering significantly, but computation time. We found that heuristic based optimization method (Nelder-Maad) is taking more time than with gradient based optimization method (BFGS) in many situations. Moreover, we found that some initialization strategies for VCMM algorithm outperform other strategies for clustering data with different characteristics. The result is summarised as a flow chart in the Figure 3.3.19 and the recommendation of the initialization strategy for data clustering in the Figure 3.3.20 and 3.3.21. Lastly, we show how the VCMM algorithm improves the clustering fit over GMM in two data sets and the performance of the selected models from the Figure 3.3.20 for clustering the two real data sets.

Contents

1 Introduction

This thesis consists of five parts. In Chapter 2, we give the necessary theoretical background, Chapter 3 contains the simulation studies, Chapter 4 does the clustering for real data sets, Chapter 5 concludes.

More specifically, Chapter 2 mainly introduces the mathematical definitions, theory and methods used throughout the thesis. Chapter 2 consists of six subsections. In Section 2.1, we introduce maximum likelihood estimation and discuss in detail the expectation maximization (EM) algorithm and its extensions. In Section 2.2, the method of Lagrange multipliers for solving optimization problems is introduced. Section 2.3 gives an overview of copulas and vine copulas. The formulation for the Gaussian mixture model (GMM) and the vine copulas mixture model (VCMM) and their corresponding steps for EM algorithms are presented in Section 2.4. Section 2.5 describes the characteristics for clusters in model based Gaussian mixtures. Further we discuss modelling issues we encounter some times when we implement the extension of EM algorithm - ECME. Section 2.6 discusses some performance measures used in the simulation study for performance assessment.

Chapter 3 shows the required steps of how the simulation experiments are implemented. There are three subsections in Section 3. In Section 3.1, four data settings following a Gaussian mixture model are generated for testing the performance of GMM algorithm by using different EM algorithms. In Section 3.2, we will use the same data settings as in the Section 3.1 for performance assessment with different initialization strategies. In the last Section 3.3, data sets for nine different settings are generated to assess the performance of the VCMM algorithm, with using different EM algorithms and initialization strategies and the best initialization strategy is suggested for the data with different characteristics.

In Chapter 4, two real data sets with two clusters are classified by GMM and VCMM algorithm, different EM algorithms and initialization strategies for performance assessment.

We will conclude the thesis in Chapter 5.

2 Theoretical Background

In this chapter, we will introduce mathematical concepts, definitions and theories used in the simulation studies.

2.1 Parameter Estimation Techniques

In this section, we will introduce some methods for parameter estimation in statistical modelling. Maximum likelihood estimation is a well-known approach to do so, but it is quite challenging on data in the presence of missing variables, for example, mixture models. Because our paper mainly focus on mixture models, EM algorithm and its extension as the iterative algorithms will be introduced and used for the parameter estimation in this thesis.

2.1.1 Maximum Likelihood Estimation

We begin by assuming that $\boldsymbol{X} \in \mathbb{R}^p$ is the random vector and the corresponding random sample matrix $\mathcal{X} = [\ \boldsymbol{x_1},\ \boldsymbol{x_2},\ ...,\ \boldsymbol{x_n}] \in \mathbb{R}^{p \times n}$ of n independent and identically distributed (i.i.d.) observations being realizations of $\boldsymbol{X}$ is given. Furthermore, it is assumed that $\boldsymbol{X}$ follows cumulative distribution function (cdf) $F(\boldsymbol{x};\ \boldsymbol{\theta})$ with probability density function (pdf) $f(\boldsymbol{x};\ \boldsymbol{\theta})$ with corresponding parameters $\boldsymbol{\theta}$. The parameter $\boldsymbol{\theta} = [\ \boldsymbol{\theta_1},\ \boldsymbol{\theta_2},\ ...,\ \boldsymbol{\theta_k}]^T \in \boldsymbol{\Theta}$ is a vector of unknown parameters needed to be estimated and $\boldsymbol{\Theta}$ denotes the parameter space.

In statistics, **maximum likelihood estimation** (MLE) is one of the common technique to estimate the parameter $\boldsymbol{\theta}$ of the (pdf), by maximizing the likelihood of the realization of the sample. Intuitively, the estimated value of the parameter by the MLE is the one most likely to have produced the sample or observations.

The likelihood function is defined as

$$L(\boldsymbol{\theta} \mid \boldsymbol{x}) = L(\boldsymbol{\theta} \mid \boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}) := \prod_{i=1}^{n} f(\boldsymbol{x_i} \mid \boldsymbol{\theta}),$$

or equivalently the corresponding log-likelihood function given by

$$l(\boldsymbol{\theta} \mid \boldsymbol{x}) = l(\boldsymbol{\theta} \mid \boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}) := \sum_{i=1}^{n} \ln(f(\boldsymbol{x_i} \mid \boldsymbol{\theta})). \tag{2.1.1}$$

Based on the definition of the MLE mentioned above, the estimated parameter $\hat{\boldsymbol{\theta}}_{MLE}$ can be denoted as

$$\hat{\boldsymbol{\theta}}_{MLE} = \arg\max_{\boldsymbol{\theta} \in \Theta} l(\boldsymbol{\theta} \mid \boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}).$$

Since the logarithm is a monotonic function, the maximum of log-likelihood function occurs at the same value of $\boldsymbol{\theta}$ as the maximum of likelihood.

We can obtain the estimated parameter $\hat{\boldsymbol{\theta}}_{MLE}$ by taking the first derivative of the log-likelihood function with respect to $\boldsymbol{\theta}$, and finding out the solution of the following equation by setting the values of the partial derivatives vector to the zero vector.

$$\frac{\partial l(\boldsymbol{\theta} \mid \boldsymbol{x})}{\partial \boldsymbol{\theta}} = \mathbf{0}$$

Because the log-likelihood function leads to a summation structure for which it is easier to take the derivatives, we prefer to maximize the log-likelihood function instead of likelihood function.

Example 2.1.1 (MLE of univariate normal distribution)

The probability density function of univariate normal distribution with mean μ and variance σ^2 is:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$

Suppose that x_1, x_2, ..., x_n represents n random samples (i.i.d) from an univariate normal distribution with mean μ and variance σ^2. The likelihood for the samples with $\boldsymbol{\theta} = (\mu, \sigma^2)$ is given by

$$L(\boldsymbol{\theta} \mid \boldsymbol{x}) = L(\mu, \sigma^2 \mid x_1, x_2, ..., x_n)$$

$$= \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x_i-\mu}{\sigma}\right)^2\right) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2\right)$$

and the corresponding log-likelihood is

$$l(\boldsymbol{\theta} \mid \boldsymbol{x}) = \ln L(\boldsymbol{\theta} \mid \boldsymbol{x}) = -\frac{n}{2}\ln(2\pi) - n\ln\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2. \tag{2.1.2}$$

In order to obtain the estimated mean $\hat{\mu}$, we can maximize the log-likelihood by taking the first derivative with respect to μ and then set to 0.

$$\frac{\partial l}{\partial \mu} = -\frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2 = 0$$

Solving this equation, we get

$$\hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Similarly, to obtain the estimated $\hat{\sigma}$, we take the first derivatives of the log-likelihood with respect to σ and then set to 0. We have

$$\frac{\partial l}{\partial \sigma} = -\frac{n}{\sigma} + \sum_{i=1}^{n} (x_i - \mu)^2 \sigma^{-3} = 0.$$

Solving this equation, we get

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n} (x_i - \hat{\mu})^2}{n}.$$

Maximum likelihood estimation has many advantages and works in many situations. However, the limitation of maximum likelihood estimation is that we assume the data used for parameter estimation is fully observable. In other words, if we have missing data and/or latent variables, which are unobserved or hidden variables influencing other random variables, then computing a maximum likelihood estimate becomes hard. [Murphy, 2012] In Section 2.1.2, another approach for parameter estimation in the presence of latent variables called **expectation-maximization (EM) algorithm** is introduced.

2.1.2 The Classical Expectation-Maximization (EM) Algorithm

The expectation-maximization is a method for deriving an iterative procedure, called EM algorithm, to maximize the likelihood function in the situation of missing data and/or latent variables.

We let the complete data consists of $\boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{z})$, where $\boldsymbol{x}$ denotes the observed but incomplete data and $\boldsymbol{z}$ denotes the unobserved or missing data. The complete-data log likelihood is then denoted by $l(\boldsymbol{\theta}; \boldsymbol{x}, \boldsymbol{z})$, where $\boldsymbol{\theta}$ is the unknown parameter vector for which we wish to find the ML estimate. In the presence of missing data, estimating the parameters by MLE is difficult. Since the complete data $\boldsymbol{y}_{comp}$ is not observed, evaluation and maximization of complete-data log likelihood l are not possible. In this situation, the EM algorithm is an useful method to solve this problem.

In general, the EM algorithm attempts to maximize $l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp})$, which is the log-likelihood function of the complete data, iteratively, by replacing the complete data log likelihood by its conditional expectation given the observed data $\boldsymbol{x}$. Because the iteration repeats an expectation step (E step) followed by a maximization step (M step), that is the reason why it is named expectation-maximization (EM) algorithm and it is defined as follows:

E-step: The E-step of the EM algorithm computes the expected value of $l(\boldsymbol{\theta} \mid \boldsymbol{Y}_{comp} = (\boldsymbol{X}, \boldsymbol{Z}))$ given the observed data $\boldsymbol{x}$, and the current parameter estimate $\boldsymbol{\theta}_{old}$. In particular, we define

$$\begin{aligned} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}) &:= E[l(\boldsymbol{\theta} \mid \boldsymbol{Y}_{comp}) \mid \boldsymbol{\theta}_{old}, \boldsymbol{X} = \boldsymbol{x}] \\ &= \int l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{z}))\ p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})\ d\boldsymbol{z}, \end{aligned}$$

where $p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})$ is the conditional density for the missing data $\boldsymbol{Z}$ given the observed data $\boldsymbol{x}$, and assuming $\boldsymbol{\theta} = \boldsymbol{\theta}_{old}$.

M-step: The M-step of the EM algorithm chooses $\boldsymbol{\theta}_{new}$ to be any value of $\boldsymbol{\theta}$ that maximizes $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$. That is

$$Q(\boldsymbol{\theta}_{new} \mid \boldsymbol{\theta}_{old}) \geq Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}),$$

or equivalently, the $\boldsymbol{\theta}_{new}$ can be denoted as

$$\boldsymbol{\theta}_{new} := \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}).$$

More specifically, on the first iteration, we first let $\boldsymbol{\theta}_{old}$ to be some initial value of $\boldsymbol{\theta}$. After one E and M-step is carried out, we set $\boldsymbol{\theta}_{old} = \boldsymbol{\theta}_{new}$. The two steps will be repeated until the stopping conditions fulfills. Two stopping conditions will be introduced in the following:

Stopping condition 1 : Convergence of parameter estimation $\boldsymbol{\theta}$
The EM algorithm stops if two successive estimate differ smaller by a prespecified ϵ, such that

$$\|\boldsymbol{\theta}_{new} - \boldsymbol{\theta}_{old}\| < \epsilon \tag{2.1.3}$$

Stopping condition 2 : Convergence of log-likelihood $l(\boldsymbol{\theta} \mid \boldsymbol{x})$
The EM algorithm stops when the relative change of the log-likelihood between two successive iterations is smaller than a prespecified ϵ, such that

$$\frac{l(\boldsymbol{\theta}_{new} \mid \boldsymbol{x}) - l(\boldsymbol{\theta}_{old} \mid \boldsymbol{x})}{l(\boldsymbol{\theta}_{old} \mid \boldsymbol{x})} < \epsilon \tag{2.1.4}$$

At that time, we can conclude that $\boldsymbol{\theta}_{new}$ is the optimal estimate of the EM process.

Example 2.1.2 (Missing Data in an Univariate Normal Sample)

Suppose $\boldsymbol{x} := (y_1, ..., y_n)^T$ is n i.i.d. samples from univariate nominal distribution with mean μ and variance σ^2. In Example 2.1.1, the log-likelihood was derived in Equation (2.1.2). However, in this example, we want to perform the EM algorithm instead. To do so, we assume the complete data is $\boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{z}) := (y_1, ..., y_n, y_{n+1}, ..., y_m)^T$, so that $\boldsymbol{z} := (y_{n+1}, ..., y_m)^T$ are missing, where $\boldsymbol{y}_{comp}$ are assumed to be i.i.d.

E-step: The E step of the EM algorithm computes the expected value of $l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{Z}))$ given the observed data $\boldsymbol{x}$ and the current parameter estimate $\boldsymbol{\theta}_{old} = (\mu_{old}, \sigma^2_{old})$. We first compute $l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp})$:

$$\begin{aligned}
l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{z})) &= -\frac{m}{2} \ln(2\pi) - \frac{m}{2} \ln \sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^{m} (y_i - \mu)^2 \\
&= -\frac{m}{2} \ln(2\pi) - \frac{m}{2} \ln \sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^{m} y_i^2 + \frac{\mu}{\sigma^2} \sum_{i=1}^{m} y_i - \frac{m\mu^2}{2\sigma^2} \\
&= -\frac{m}{2} \ln(2\pi) - \frac{m}{2} \ln \sigma^2 - \frac{m\mu^2}{2\sigma^2} - \frac{1}{2\sigma^2} \left(\sum_{i=1}^{n} y_i^2 + \sum_{i=n+1}^{m} y_i^2 \right) \\
&+ \frac{\mu}{\sigma^2} \left(\sum_{i=1}^{n} y_i + \sum_{i=n+1}^{m} y_i \right)
\end{aligned}$$

Since

$$E[Y_i \mid \boldsymbol{x}, \boldsymbol{\theta}_{obs}] = \begin{cases} y_i & i = 1, ..., n \\ \mu_{old} & i = n+1, ..., m \end{cases}$$

$$E[Y_i^2 \mid \boldsymbol{x}, \boldsymbol{\theta}_{obs}] = \begin{cases} y_i^2 & i = 1, ..., n \\ \mu^2_{old} + \sigma^2_{old} & i = n+1, ..., m, \end{cases}$$

we have

$$\begin{aligned}
Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}) &:= E[l(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{Z}) \mid \boldsymbol{x}, \boldsymbol{\theta}_{old}] \\
&= -\frac{m}{2} \ln(2\pi) - \frac{m}{2} \ln \sigma^2 - \frac{m\mu^2}{2\sigma^2} - \frac{1}{2\sigma^2} \sum_{i=1}^{n} y_i^2 - \frac{(m-n)}{2\sigma^2} (\mu^2_{old} + \sigma^2_{old}) \\
&+ \frac{\mu}{\sigma^2} \sum_{i=1}^{n} y_i - \frac{(m-n)\mu_{old}\mu}{\sigma^2}
\end{aligned}$$

We define

$$S_1(\mu_{old}) = (m-n)\mu_{old} + \sum_{i=1}^{n} y_i$$

and

$$S_2(\mu_{old}, \sigma^2_{old}) := (m-n)(\sigma^2_{old} + \mu^2_{old}) + \sum_{i=1}^{n} y_i^2.$$

$$\Rightarrow Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}) = -\frac{m}{2}\ln(2\pi) - \frac{m}{2}\ln\sigma^2 - \frac{m\mu^2}{2\sigma^2} - \frac{1}{2\sigma^2}S_2(\mu_{old}, \sigma^2_{old}) + \frac{\mu}{\sigma^2}S_1(\mu_{old}) \tag{2.1.5}$$

M-step: The M-step of the EM algorithm chooses $\boldsymbol{\theta}_{new}$ to be any value of $\boldsymbol{\theta}$ that maximizes $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$. The $\boldsymbol{\theta}_{new}$ can be denoted as

$$\boldsymbol{\theta}_{new} := \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}). \tag{2.1.6}$$

The M step here, taking the derivatives with respect to μ and σ^2 separately, does not maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$ over the parameter space $\boldsymbol{\Theta}$ jointly, which is the definition of the M step in the classical EM algorithm. Therefore, this method is actually ECM algorithm which will be introduced later. But, at this moment, we regard this as a standard EM algorithm because this is the way how people estimate the parameters by EM algorithm.

To get μ_{new} needed in Equation (2.1.6), take the derivatives of $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$ with respect to μ and set to 0:

$$\begin{aligned}
\frac{S_1(\mu_{old})}{\sigma^2} - \frac{m\mu}{\sigma^2} &= 0 \\
\mu_{new} &= \frac{S_1(\mu_{old})}{m} \\
&= \frac{\sum_{i=1}^{n} y_i + (m-n)\mu_{old}}{m}
\end{aligned} \tag{2.1.7}$$

To get σ^2_{new} needed in Equation (2.1.6), take the derivatives of $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$ with respect to σ^2 and set to 0:

$$\begin{aligned}
-\frac{m}{2\sigma^2} + \frac{1}{2(\sigma^2)^2}S_2(\mu_{old}, \sigma^2_{old}) - \frac{\mu}{(\sigma^2)^2}S_1(\mu_{old}) + \frac{m\mu^2}{2(\sigma^2)^2} &= 0 \\
-m\sigma^2 + S_2(\mu_{old}, \sigma^2_{old}) - 2\mu S_1(\mu_{old}) + m\mu^2 &= 0 \\
-m\sigma^2 + S_2(\mu_{old}, \sigma^2_{old}) - 2m\mu^2 + m\mu^2 &= 0
\end{aligned}$$

$$\begin{aligned}
\Longrightarrow \sigma^2_{new} &= \frac{S_2(\mu_{old}, \sigma^2_{old})}{m} - \mu^2 \\
&= \frac{\sum_{i=1}^{n} y_i^2 + (m-n)(\sigma^2_{old} + \mu^2_{new})}{m} - \mu^2_{new}
\end{aligned} \tag{2.1.8}$$

To illustrate the application of the EM algorithm to this type of problem, we now apply it to the following random data set. Here, there are $m = 15$ observations, but 5 of them are missing which are indicated by NA, thus observed number $n = 10$.

$$\boldsymbol{y} = \{5, 3, 10, 0, -4, 5, 6, 3, -5, 7, \text{NA}, \text{NA}, \text{NA}, \text{NA}, \text{NA}\}$$

We assume that the observations follow normal distribution, so we can apply the expression in Equation (2.1.7) and Equation (2.1.8) to estimate its mean μ and variance σ^2. Also, we choose $\mu^{(0)} = 10$ and $\sigma^{2^{(0)}} = 10$ as the initial values for the iteration and the stopping condition 1 from Equation (2.1.3) is used. Table 2.1.1 shows the convergence results of applying EM to this problem and we can know that, as iteration t increases, the $\mid \mu^{(t)} - \mu^{(t-1)} \mid$ and $\mid \sigma^{2^{(t)}} - \sigma^{2^{(t-1)}} \mid$ in each iteration tend to 0, so that $\mu^{(t)}$ and $\sigma^{2^{(t)}}$ converge. Therefore, we can conclude that, by the EM algorithm, the estimated $\hat{\mu} \approx 3$ and estimated $\hat{\sigma^2} \approx 20.4$ are the optimal estimate of EM process.

Actually, each EM step monotonically increases the likelihood function. This is the reason why repeating EM steps until convergence can maximize the likelihood function. To prove this, the definition of the E-step and M-step can be used. In the M-step, $\boldsymbol{\theta}_{new}$ is chosen to maximizes $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$, that is

$$Q(\boldsymbol{\theta}_{new} \mid \boldsymbol{\theta}_{old}) \geq Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}) \quad \forall \boldsymbol{\theta}.$$

And $\boldsymbol{\theta}$ can be chosen as $\boldsymbol{\theta}_{old}$, such that

$$Q(\boldsymbol{\theta}_{new} \mid \boldsymbol{\theta}_{old}) \geq Q(\boldsymbol{\theta}_{old} \mid \boldsymbol{\theta}_{old}).$$

Table 2.1.1: The EM iteration values for Example 2.1.2

Iteration t	$\mu^{(t)}$	$\mid \mu^{(t)} - \mu^{(t-1)} \mid$	$\sigma^{2(t)}$	$\mid \sigma^{2(t)} - \sigma^{2(t-1)} \mid$
0	10	-	10	-
1	5.333333	4.666667	27.82222	17.82222
2	3.777778	1.555555	24.08395	3.73827
3	3.259259	0.518519	21.76241	2.32154
4	3.08642	0.172839	20.86907	0.89334
5	3.028807	0.057613	20.55802	0.31105
6	3.009602	0.019205	20.45286	0.10516
7	3.003201	0.006401	20.41764	0.03522
8	3.001067	0.002134	20.40588	0.01176
9	3.000356	0.000711	20.40196	0.00392
10	3.000119	0.000237	20.40065	0.00131

In the E-step, we defined $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old}) := \int l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp})\ p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})\ d\boldsymbol{z}$. So, we have

$$\int l(\boldsymbol{\theta}_{new} \mid \boldsymbol{y}_{comp})\ p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})\ d\boldsymbol{z} \geq \int l(\boldsymbol{\theta}_{old} \mid \boldsymbol{y}_{comp})\ p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})\ d\boldsymbol{z},$$

which implies

$$\int [\ l(\boldsymbol{\theta}_{new} \mid \boldsymbol{y}_{comp}) - l(\boldsymbol{\theta}_{old} \mid \boldsymbol{y}_{comp})\ ]\ p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})\ d\boldsymbol{z} \geq 0$$

In the Section 2.1.1, log-likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{x})$ is defined as $\ln[\prod_{i=1}^{n} f(\boldsymbol{x_i} \mid \boldsymbol{\theta})]$. For simplicity, we define $l(\boldsymbol{\theta} \mid \boldsymbol{y})$ as $\ln p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta})$, where $p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta})$ is the probability density function of complete data $\boldsymbol{y}_{comp}$.

$$\int \ln \left[\frac{p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{new})}{p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{old})}\right]\ p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})\ d\boldsymbol{z} \geq 0$$

By $\ln x \leq x - 1$ for all $x \geq 0$ with equality if and only if $x = 1$, we have

$$\int \left[\frac{p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{new})}{p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{old})} - 1\right]\ p(\boldsymbol{z} \mid \boldsymbol{\theta}_{old}, \boldsymbol{x})\ d\boldsymbol{z} \geq 0$$

$$\iff \int \left[\frac{p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{new})}{p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{old})} - 1\right]\ \frac{p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{old})}{p(\boldsymbol{x} \mid \boldsymbol{\theta}_{old})}\ d\boldsymbol{z} \geq 0$$

$$\iff \frac{1}{p(\boldsymbol{x} \mid \boldsymbol{\theta}_{old})} \int [p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{new}) - p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta}_{old})]\ d\boldsymbol{z} \geq 0$$

Since $\int p(\boldsymbol{y}_{comp} \mid \boldsymbol{\theta})\ d\boldsymbol{z} = \int p(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta})\ d\boldsymbol{z} = p(\boldsymbol{x} \mid \boldsymbol{\theta})$,

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}_{new}) - p(\boldsymbol{x} \mid \boldsymbol{\theta}_{old}) \geq 0$$
$$\implies L(\boldsymbol{\theta}_{new} \mid \boldsymbol{x}) \geq L(\boldsymbol{\theta}_{old} \mid \boldsymbol{x})$$

We can see that, in the M-step, finding a $\boldsymbol{\theta}$ by maximizing $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$ over $\boldsymbol{\theta}$ is equivalent to maximize the likelihood function $L(\boldsymbol{\theta} \mid \boldsymbol{x})$ over $\boldsymbol{\theta}$, such that

$$Q(\boldsymbol{\theta}_{new} \mid \boldsymbol{\theta}_{old}) \geq Q(\boldsymbol{\theta}_{old} \mid \boldsymbol{\theta}_{old}) \quad \forall \boldsymbol{\theta} \implies L(\boldsymbol{\theta}_{new} \mid \boldsymbol{x}) \geq L(\boldsymbol{\theta}_{old} \mid \boldsymbol{x}) \tag{2.1.9}$$

This is the reason why the EM algorithm works, such that each iteration increasing the likelihood.

In this section, we have shown that the advantages of the EM algorithm: 1) In the presence of missing data and/or latent variables, the complete-data maximum likelihood estimation is often computationally simple. 2) Its convergence is stable, because of each iteration increasing the likelihood. However, the classic EM algorithm also has its limitations, for example, when the associated complete-data maximum likelihood estimation itself is complicated, EM is less attractive because the M-step is computationally unattractive. [Meng and Rubin, 1993] In the next Section 2.1.3, some extensions of the EM algorithm improving its limitations will be introduced.

2.1.3 Extensions of the EM Algorithm

In this section, further modifications and extensions to the EM algorithm are considered. In particular, we focus on 1. Expectation Conditional Maximization (ECM) algorithm, 2. multi-cycle ECM (MCECM) algorithm and 3. Expectation/Conditional Maximization Either (ECME) algorithm.

1. The Expectation Conditional Maximization (ECM) Algorithm

In the M step of the classical EM algorithm, it is required to choose $\boldsymbol{\theta}_{new}$ which maximizes $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$. In the Example 2.1.2, we want to choose μ, σ^2 together to maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$. Actually, we just can do it numerically because a analytic solution to maximize a function for 2 parameters or more does not exist in general. The M step in the Example 2.1.2, taking the derivatives with respect to μ and σ^2 separately (or fixing the μ to estimate the σ^2), cannot really maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$ over the parameter space $\boldsymbol{\Theta}$, which is the definition of the M step in the classical EM algorithm. Strictly speaking, the method shown in the example is not the classical EM algorithm. Because of the convenience of finding a analytic solution, people do the so-called "EM algorithm" in that way and we still call it "EM algorithm".

Due to the limitation of the classical EM algorithm in the M step, Meng and Rubin [1993] proposed a modification of the EM algorithm called expectation conditional maximization (ECM) algorithm. In the ECM algorithm, the M step is replaced by several computationally simpler conditional maximization (CM) steps. Each CM-step maximizes $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$ found on the preceding E-step subject to the constraints on $\boldsymbol{\theta}$. compared to the classic EM algorithm, ECM algorithm is often faster, simpler and more stable for maximization in the M step, because the CM maximization are over smaller dimensional spaces.

More precisely, the expectation conditional maximization (ECM) algorithm at iteration $t+1$ is defined as follows:

E-step: The E step in the classic EM algorithm and the ECM algorithm are exactly the same. Compute the expected value of $l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{Z}))$ given the observed data $\boldsymbol{x}$, and the current parameter estimate $\boldsymbol{\theta}^{(t)}$:

$$Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) := E[l(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{Z}) \mid \boldsymbol{\theta}^{(t)}, \boldsymbol{x}]$$

CM-steps: The M step in the ECM algorithm is replaced by $S > 1$ steps and each step is called CM step, so that there are S CM steps in each iteration. At the sth CM-step of the $(t+1)$th iteration, $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ is chosen to maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ which is subject to the constraint $g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})$, or equivalently, the $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ is defined as

$$\begin{aligned} \boldsymbol{\theta}^{(t+\frac{s}{S})} &:= \arg\max\nolimits_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) \\ &\qquad \text{subject to } g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})}) \end{aligned}$$

where $G = \{g_s(\boldsymbol{\theta}), s = 1, ..., S\}$ is a set of S preselected functions. Thus, $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ satisfies

$$Q(\boldsymbol{\theta}^{(t+\frac{s}{S})} \mid \boldsymbol{\theta}^{(t)}) \geq Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) \text{ for all } \boldsymbol{\theta} \in \{\boldsymbol{\theta} \in \boldsymbol{\Theta} : g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})\}. \tag{2.1.10}$$

where $\boldsymbol{\Theta}$ is parameter space.

In the final CM-step of the $(t+1)$th iteration, the value $\boldsymbol{\theta}^{(t+\frac{S}{S})} = \boldsymbol{\theta}^{(t+1)}$, is taken to be the input on the $(t+2)$th iteration.

From Equation (2.1.10), we know that

$$\begin{aligned} Q(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) &\geq Q(\boldsymbol{\theta}^{(t+\frac{S-1}{S})} \mid \boldsymbol{\theta}^{(t)}) \\ &\geq Q(\boldsymbol{\theta}^{(t+\frac{S-2}{S})} \mid \boldsymbol{\theta}^{(t)}) \\ &\;\;\vdots \\ &\geq Q(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)}) \end{aligned}$$

As noted before from Equation (2.1.9) in the Section 2.1.2 that each EM step monotonically increases the likelihood function, we have

$$L(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{x}) \geq L(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{x}).$$

We can conclude that the likelihood function monotonically increases in each iteration for the ECM algorithm. Therefore, the ECM algorithm still works like the classical EM algorithm, even faster, more stable.

Now, we are going to show more detail about the implementation of the ECM algorithm. Firstly, we need to decide the number of S CM steps and the order of the parameters that we are going to update in each CM step, but there is no rule to choose it. Normally, S is equal to the number of parameters because ideally we want to update all the parameters once in each iteration. Also, we partition the parameter $\boldsymbol{\theta}$ into subvectors $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_S\}$ in order to decide the order of the parameters which will be updated in each step. According to the sub-vectors $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_S\}$, $\boldsymbol{\theta}_1$ will be updated first by maximization with respect to $\boldsymbol{\theta}_1$, with all other parameters held fixed and so on and so fort. Basically, $G = \{g_s(\boldsymbol{\theta}), s = 1, ..., S\}$ is just chosen for setting other parameters constant as the constraints in an optimization problem in each CM step. More detail will be in the Example 2.1.3.

Example 2.1.3 (Missing Data in an Univariate Normal Sample (ECM))

In the Example 2.1.2, we showed the steps for the missing data in an univaiate normal sample using the classical EM algorithm. Here, we are going to show you the typical steps of the ECM algorithm for this problem and the difference between the classical EM algorithm and the ECM algorithm.

E step of EM, ECM at iteration $t+1$

E step of the ECM algorithm and the EM algorithm are the same. From Equation (2.1.5), we have

$$\begin{aligned} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) &:= E[l(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{Z}) \mid \boldsymbol{\theta}^{(t)}, \boldsymbol{x}] \\ &= -\frac{m}{2}\ln\sigma^2 - \frac{1}{2\sigma^2}S_2^{(t)} + \frac{\mu}{\sigma^2}S_1^{(t)} - \frac{m\mu^2}{2\sigma^2} + \text{constant}, \end{aligned}$$

where $S_1^{(t)} = \sum_{i=1}^n y_i + (m-n)\mu^{(t)}$ and $S_2^{(t)} = \sum_{i=1}^n y_i^2 + (m-n)({\sigma^2}^{(t)} + {\mu^{(t)}}^2)$.

M step of EM at iteration $t+1$

Recalling from Equation (2.1.6), the M-step of the EM algorithm chooses $\boldsymbol{\theta}^{(t+1)} = (\mu^{(t+1)}, {\sigma^2}^{(t+1)})$ that maximizes $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$. That is

$$\boldsymbol{\theta}^{(t+1)} := \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}).$$

As we mentioned, we just can solve this optimization problem numerically when there are more than 1 parameters. Therefore, no analytic solution can be found.

CM step of ECM at iteration $t+1$

Recalling that in normal distribution, there are two parameters with μ and σ^2. Therefore, we just have two ways to partition them because the order is also taking into account, so that we have case $1: \theta_1 = \mu$, $\theta_2 = \sigma^2$ and case $2: \theta_1 = \sigma^2$, $\theta_2 = \mu$. Also, due to $S = 2$, the M step can be replaced by 2 CM steps.

Case 1 : $\theta_1 = \mu$, $\theta_2 = \sigma^2$

CM-step 1: Due to $\theta_1 = \mu$, the CM-step 1 is to choose $\mu^{(t+\frac{1}{2})}$ by maximizing $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ subject to $g_1(\mu, \sigma^2) = g_1(\mu^{(t)}, \sigma^{2^{(t)}})$, where $g_1(\mu, \sigma^2)$ is some selected functions and we will select it a bit later. Therefore, $\mu^{(t+\frac{1}{2})}$ can be denoted as

$$\begin{aligned}\mu^{(t+\frac{1}{2})} &:= \arg\max_\mu Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) \\ &= \arg\max_\mu \left(-\frac{m}{2}\ln\sigma^2 - \frac{1}{2\sigma^2}S_2^{(t)} + \frac{\mu}{\sigma^2}S_1^{(t)} - \frac{m\mu^2}{2\sigma^2} + \text{constant} \right) \\ &\quad \text{subject to } g_1(\mu, \sigma^2) = g_1(\mu^{(t)}, \sigma^{2^{(t)}})\end{aligned}$$

and other parameters keep constant, so that the "updated" variance is denoted as $\sigma^{2^{(t+\frac{1}{2})}} := \sigma^{2^{(t)}}$.

Now, we would like to choose some constraints $g_1(\mu, \sigma^2) = g_1(\mu^{(t)}, \sigma^{2^{(t)}})$ to make the above optimization can be solved easily, at least by taking the derivatives. The easiest way is to set the parameters other than μ to constant, such that $\sigma^2 = \sigma^{2^{(t)}}$. Formally, $g_1(\mu, \sigma^2) = \sigma^2$ and the conditional optimization problem can be written as

$$\mu^{(t+\frac{1}{2})} := \arg\max_\mu \left(-\frac{m}{2}\ln\sigma^{2^{(t)}} - \frac{1}{2\sigma^{2^{(t)}}}{S_2}^{(t)} + \frac{\mu}{\sigma^{2^{(t)}}}{S_1}^{(t)} - \frac{m\mu^2}{2\sigma^{2^{(t)}}} + \text{constant} \right)$$

Taking the derivatives with respect to μ and set to 0, we get

$$\begin{aligned}\mu^{(t+\frac{1}{2})} &= \frac{{S_1}^{(t)}}{m} \\ &= \frac{\sum_{i=1}^n y_i + (m-n)\mu^{(t)}}{m}.\end{aligned}$$

CM-step 2: Similarly, the CM-step 2 is to choose $\sigma^{2^{(t+1)}}$ (or $\sigma^{2^{(t+\frac{2}{2})}}$) by maximizing $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t+\frac{1}{2})})$ subject to $g_2(\mu, \sigma^2) = g_2(\mu^{(t+\frac{1}{2})}, \sigma^{2^{(t+\frac{1}{2})}})$, where $g_2(\mu, \sigma^2) = \mu$, so that parameters other than σ^2 can keep constant. In this case, $\sigma^{2^{(t+1)}}$ can be defined as

$$\sigma^{2^{(t+1)}} := \arg\max_{\sigma^2} \left(-\frac{m}{2}\ln\sigma^2 - \frac{1}{2\sigma^2}{S_2}^{(t+\frac{1}{2})} + \frac{\mu^{(t+\frac{1}{2})}}{\sigma^2}{S_1}^{(t+\frac{1}{2})} - \frac{m\left(\mu^{(t+\frac{1}{2})}\right)^2}{2\sigma^2} + \text{constant} \right).$$

and other parameters keep constant, so that the "updated" mean is denoted as $\mu^{(t+1)} := \mu^{(t+\frac{1}{2})}$. Taking the derivatives with respect to σ^2 and set to 0, we get

$$\begin{aligned}\sigma^{2^{(t+1)}} &= \frac{{S_2}^{(t+\frac{1}{2})}}{m} - \left(\mu^{(t+\frac{1}{2})}\right)^2 \\ &= \frac{\sum_{i=1}^n y_i^2 + (m-n)\left(\sigma^{2^{(t+\frac{1}{2})}} + \left(\mu^{(t+\frac{1}{2})}\right)^2\right)}{m} - \left(\mu^{(t+\frac{1}{2})}\right)^2.\end{aligned}$$

You may notice that the expressions for μ and σ^2 that we found above are the same in the Example 2.1.2, just the notations different. As we mentioned the M steps in that example, fixing the μ to estimate the σ^2, cannot really maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{old})$ over the parameter space $\boldsymbol{\Theta}$, which is the definition of the M step in the classical EM algorithm. Actually, the method used in the Example 2.1.2 is the ECM algorithm.

For $S = 2$, we have 2 different orders to update the parameters. This time, we try to update σ^2 first and then μ.

Case 2 : $\theta_1 = \sigma^2$, $\theta_2 = \mu$

The computational steps are exactly the same as the case 1. Just the order is different. So, the steps are skipped here:

CM-step 1:

$$\begin{aligned}\sigma^{2^{(t+\frac{1}{2})}} &:= \frac{{S_2}^{(t)}}{m} - \left(\mu^{(t)}\right)^2 \\ &= \frac{\sum_{i=1}^n y_i^2 + (m-n)\left(\sigma^{2^{(t)}} + \left(\mu^{(t)}\right)^2\right)}{m} - \left(\mu^{(t)}\right)^2 \\ \mu^{(t+\frac{1}{2})} &:= \mu^{(t)}\end{aligned}$$

CM-step 2:

$$\begin{aligned}\mu^{(t+1)} &:= \frac{{S_1}^{(t+\frac{1}{2})}}{m} \\ &= \frac{\sum_{i=1}^{n} y_i + (m-n)\mu^{(t+\frac{1}{2})}}{m} \\ \sigma^{2(t+1)} &:= \sigma^{2(t+\frac{1}{2})}\end{aligned}$$

In general, assume that there are S parameters in a model. If we just want to update 1 parameter in each CM step, we totally have $S!$ different orders to implement the CM steps. You may be interested in whether the order affects the performance of the parameter estimation. van Dyk and Meng [1997] mentioned that the order that the CM-steps are performed is trivial to change and generally affects how fast the algorithm converges and it is impossible to find an "optimal" order choosing or grouping-choosing rule that will be universally applicable, which means the order of performing CM steps affecting the performance of the parameter estimation.

2. The Multi-cycle Expectation Conditional Expectation (MCECM) Algorithm MCECM algorithm is an extension of ECM algorithm. In the ECM algorithm, E step is just performed once before all the CM steps. But, in the MCECM algorithm, E step is performed before each CM step. Meng and Rubin [1993] mentioned that in some cases, the computation of an E-step may be much cheaper than computation of the CM-steps. Performing an E-step before each CM-step may result in larger increases in likelihood function L per iteration since Q is being updated more often. This is the reason why MCECM may be a better extension of ECM.

Suppose there are S CM steps in each iteration. In MCECM, since E step is performed before each CM step and a cycle is defined by one E step followed by one CM step, where each iteration involves S cycles. This is how multi-cycle Expectation Conditional Expectation is named. More precisely, multi-cycle Expectation Conditional Expectation (MCECM) algorithm at the sth of iteration $t+1$ is defined as follows:

E-step: The E step of the MCECM algorithm in the sth cycle of iteration $t+1$ is to compute the expected value of $l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{Z}))$ given the observed data $\boldsymbol{x}$, and the parameter estimate in the $(s-1)$th cycle $\boldsymbol{\theta}^{(t+\frac{s-1}{S})}$:

$$Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})}) := E[l(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{Z}) \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})}, \boldsymbol{x}]$$

CM-step: The M step of the MCECM algorithm in the sth cycle of iteration $t+1$ is to choose $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ to maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})})$ which is subject to the constraint $g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})$, or equivalently, the $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ is defined as

$$\begin{aligned}\boldsymbol{\theta}^{(t+\frac{s}{S})} &:= \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})}) \\ &\quad \text{subject to } g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})\end{aligned}$$

where $G = \{g_s(\boldsymbol{\theta}), s = 1, ..., S\}$ is a set of S preselected functions. Thus, $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ satisfies

$$Q(\boldsymbol{\theta}^{(t+\frac{s}{S})} \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})}) \geq Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})}) \text{ for all } \boldsymbol{\theta} \in \{\boldsymbol{\theta} \in \boldsymbol{\Theta} : g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})\}. \tag{2.1.11}$$

where $\boldsymbol{\Theta}$ is parameter space.

In the final Sth cycle of the $(t+1)$th iteration, the value $\boldsymbol{\theta}^{(t+\frac{S}{S})} = \boldsymbol{\theta}^{(t+1)}$, is taken to be the input on the $(t+2)$th iteration.

From Equation (2.1.11), we know that in the sth cycle of MCECM,

$$Q(\boldsymbol{\theta}^{(t+\frac{s}{S})} \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})}) \geq Q(\boldsymbol{\theta}^{(t+\frac{s-1}{S})} \mid \boldsymbol{\theta}^{(t+\frac{s-1}{S})}).$$

In the Section 2.1.2 from Equation (2.1.9), we have mentioned that EM algorithm monotonically increases the likelihood function, because $Q(\boldsymbol{\theta}_{new} \mid \boldsymbol{\theta}_{old}) \geq Q(\boldsymbol{\theta}_{old} \mid \boldsymbol{\theta}_{old})$ results in $L(\boldsymbol{\theta}_{new} \mid \boldsymbol{x}) \geq L(\boldsymbol{\theta}_{old} \mid \boldsymbol{x})$. Therefore, this inequality leads to

$$L(\boldsymbol{\theta}^{(t+\frac{s}{S})} \mid \boldsymbol{x}) \geq L(\boldsymbol{\theta}^{(t+\frac{s-1}{S})} \mid \boldsymbol{x}).$$

Hence, the multi cycle ECM algorithm monotonically increases the likelihood function L, after each cycle. Because each iteration consists of S cycles, likelihood function L also increases in each iteration.

3. The Expectation/Conditional Maximisation Either (ECME) ECME is another extension of ECM algorithm proposed by Liu and Rubin [1994]. They found that this extension is nearly always faster the both the EM and ECM algorithms in terms of the number of iterations. The full name of the ECME algorithm is Expectation/Conditional Maximisation Either. McLachlan and Krishnan [2007] explained that the "either" refers to the fact that, some or all of the CM-steps of the ECM algorithm are replaced by steps that conditionally maximize the actual incomplete-data log likelihood function and not the Q function. Therefore, in the ECME algorithm, each CM step either maximizes the conditional expectation of the complete data log likelihood Q or the actual log likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{x})$ from Equation (2.1.1), subject to the same constraints G on θ like in the ECM algorithm.

More precisely, the Expectation/Conditional Maximisation Either (ECME) algorithm at iteration $t+1$ is defined as follows:

E-step: The E step in the classic EM algorithm, the ECM algorithm and the ECME algorithm are exactly the same. Compute the expected value of $l(\boldsymbol{\theta} \mid \boldsymbol{y}_{comp} = (\boldsymbol{x}, \boldsymbol{Z}))$ given the observed data $\boldsymbol{x}$, and the current parameter estimate $\boldsymbol{\theta}^{(t)}$:

$$Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) := E[l(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{Z}) \mid \boldsymbol{\theta}^{(t)}, \boldsymbol{x}]$$

The M step in the ECME algorithm is replaced by $S > 1$ steps. Here, we suppose that, in the first $(S-1)$ steps, the conditional expectation of the complete data log likelihood is maximised and in the last step, the actual log likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{x})$ from Equation (2.1.1) is maximised.

CM-steps with conditional expectation: At the sth CM-step of the $(t+1)$th iteration, $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ is chosen to maximize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ which is subject to the constraint $g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})$, or equivalently, the $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ is defined as

$$\begin{aligned}\boldsymbol{\theta}^{(t+\frac{s}{S})} &:= \arg\max\nolimits_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) \\ &\quad \text{subject to } g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})\end{aligned}$$

where $G = \{g_s(\boldsymbol{\theta}), s = 1, ..., S-1\}$ is a set of $S-1$ preselected functions. Thus, $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ satisfies

$$Q(\boldsymbol{\theta}^{(t+\frac{s}{S})} \mid \boldsymbol{\theta}^{(t)}) \geq Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) \text{ for all } \boldsymbol{\theta} \in \{\boldsymbol{\theta} \in \boldsymbol{\Theta} : g_s(\boldsymbol{\theta}) = g_s(\boldsymbol{\theta}^{(t+\frac{s-1}{S})})\}. \tag{2.1.12}$$

where $\boldsymbol{\Theta}$ is parameter space.

After $(S-1)$ CM steps with conditional expectation have been performed, the actual log likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{x})$ from Equation (2.1.1) will be maximised in the Sth step.

CM-step with actual log likelihood:

$$\begin{aligned}\boldsymbol{\theta}^{(t+\frac{S}{S})} &:= \arg\max\nolimits_{\boldsymbol{\theta}} l(\boldsymbol{\theta} \mid x) \\ &\quad \text{subject to } g_S(\boldsymbol{\theta}) = g_S(\boldsymbol{\theta}^{(t+\frac{S-1}{S})})\end{aligned}$$

where g_S is a preselected function. Thus, $\boldsymbol{\theta}^{(t+\frac{s}{S})}$ satisfies

$$l(\boldsymbol{\theta}^{(t+\frac{S}{S})} \mid x) \geq l(\boldsymbol{\theta} \mid x) \text{ for all } \boldsymbol{\theta} \in \{\boldsymbol{\theta} \in \boldsymbol{\Theta} : g_S(\boldsymbol{\theta}) = g_S(\boldsymbol{\theta}^{(t+\frac{S-1}{S})})\}. \tag{2.1.13}$$

In the final CM-step of the $(t+1)$th iteration, the value $\boldsymbol{\theta}^{(t+\frac{S}{S})} = \boldsymbol{\theta}^{(t+1)}$, is taken to be the input on the $(t+2)$th iteration.

From Equation (2.1.13), we know that

$$\begin{aligned} & l(\boldsymbol{\theta}^{(t+\frac{S}{S})} \mid x) \geq l(\boldsymbol{\theta}^{(t+\frac{S-1}{S})} \mid x) \\ \implies & L(\boldsymbol{\theta}^{(t+\frac{S}{S})} \mid x) \geq L(\boldsymbol{\theta}^{(t+\frac{S-1}{S})} \mid x). \end{aligned} \tag{2.1.14}$$

From Equation (2.1.12), we know that

$$\begin{aligned} \implies Q(\boldsymbol{\theta}^{(t+\frac{S-1}{S})} \mid \boldsymbol{\theta}^{(t)}) &\geq Q(\boldsymbol{\theta}^{(t+\frac{S-2}{S})} \mid \boldsymbol{\theta}^{(t)}) \\ &\geq Q(\boldsymbol{\theta}^{(t+\frac{S-3}{S})} \mid \boldsymbol{\theta}^{(t)}) \\ &\vdots \\ &\geq Q(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)}) \end{aligned}$$

From Equation (2.1.9) in the Section 2.1.2 that each EM step monotonically increases the likelihood function, we have

$$L(\boldsymbol{\theta}^{(t+\frac{S-1}{S})} \mid \boldsymbol{x}) \geq L(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{x}). \tag{2.1.15}$$

Combining Equation (2.1.14) and (2.1.15), we have

$$L(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{x}) \geq L(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{x}).$$

We can conclude that the likelihood function monotonically increases in each iteration for the ECME algorithm.

2.2 Optimization

In this section, the method of Lagrange multipliers for solving optimization problem is introduced. This strategy is needed to solve the analytical solution for mixture model by using EM algorithm in the later section.

For the case of only one constraint and only two choice variables, consider the optimization problem

$$\begin{aligned} &\textbf{maximize over } (x,y) \in \mathbb{R}^2 : m(x,y) \\ &\textbf{subject to}: \; n(x,y) = 0 \end{aligned}$$

The Lagrangian function is defined by

$$\mathcal{L}(x,y,\lambda) = m(x,y) - \lambda n(x,y)$$

where λ is called Lagrange multiplier. The optimization problem can be solved by getting the solution of the equation:

$$\nabla_{x,y,\lambda} \mathcal{L}(x,y,\lambda) = 0$$

Example 2.2.1 (Optimization problem with single constraint)

Suppose we want to maximize $m(x,y) = x+y$ subject to the constraint $x^2+y^2 = 1$. Formally, the optimization problem can be expressed as

$$\begin{aligned} &\textbf{maximize over } (x,y) \in \mathbb{R}^2 : x+y \\ &\textbf{subject to}: x^2+y^2 = 1 \end{aligned}$$

For the method of Lagrange multipliers, the constraint $n(x,y)$ is

$$n(x,y) = x^2+y^2-1 = 0.$$

So, we have the Lagrangian function

$$\begin{aligned} \mathcal{L}(x,y,\lambda) &= m(x,y) - \lambda n(x,y) \\ &= x+y+\lambda(x^2+y^2-1). \end{aligned}$$

Now, we can calculate the gradient:

$$\begin{aligned}\nabla_{x,y,\lambda}\mathcal{L}(x,y,\lambda) &= \left(\frac{\partial \mathcal{L}}{\partial x}, \frac{\partial \mathcal{L}}{\partial y}, \frac{\partial \mathcal{L}}{\partial \lambda}\right) \\ &= (1+2\lambda x, 1+2\lambda y, x^2+y^2-1)\end{aligned}$$

We set the gradient to 0 and we have a system of equations:

$$\begin{cases} 1+2\lambda x = 0 \\ 1+2\lambda y = 0 \\ x^2+y^2-1=0 \end{cases}$$

By the first two equations, we have

$$x = y = -\frac{1}{2\lambda},\ \lambda \neq 0$$

Substituting into the last equation we have:

$$\begin{aligned}&\frac{1}{4\lambda^2} + \frac{1}{4\lambda^2} - 1 = 0 \\ &\implies \lambda = \pm\frac{1}{\sqrt{2}}\end{aligned}$$

So, the stationary points, such that function's derivative is zero, of $\mathcal{L}$ are

$$\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, -\frac{1}{\sqrt{2}}\right) \text{ and } \left(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, \frac{1}{\sqrt{2}}\right).$$

Plugging in the stationary points to the objective function $f(x,y)$ yields

$$f\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right) = \sqrt{2} \quad \text{and} \quad f\left(\frac{-\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right) = -\sqrt{2}.$$

Thus the constrained maximum is $\sqrt{2}$ and the constrained minimum is $-\sqrt{2}$.

In the Figure 2.2.1, this shows you the illustration of the constrained optimization problem. The colourful plane represents function $m(x,y) = x+y$. The colour near warm colour, like red, the value of $m(x,y)$ is higher. The colour near cold colour, like blue, the value of $m(x,y)$ is lower. Also, the black circle on the ground is the function $x^2+y^2=1$. Therefore, the black circle inside the colour plane shows the value of $m(x,y)$ subject to the constraint $x^2+y^2=1$. When the coordinate is $\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}, -\frac{1}{\sqrt{2}}\right)$, the value of $m(x,y)$ reaches the highest value in the black circle and when the coordinate is $\left(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, \frac{1}{\sqrt{2}}\right)$, the value of $m(x,y)$ reaches the lowest value in the black circle.

2.3 Copulas

We know that if (Z_1, Z_2) follows a bivariate Gaussian distribution, Z_1 and Z_2 also follow univariate Gaussian distributions. How about the only if situation ? If Z_1 and Z_2 both follow univariate Gaussian distributions, does (Z_1, Z_2) follows a bivariate Gaussian distribution ? The answer is no, because we don't know the dependency between Z_1 and Z_2. Bivariate Gaussian distribution is a common and symmetric distribution which allows us to have a symmetric dependency, that the value of Z_1 is larger (smaller) and the value of Z_2 has higher chance to get larger (smaller) and vice versa. However, in the real world cases, it is not always reasonable to fit the data with a Gaussian distribution. What if the case of asymmetric dependency or the variables don't follow normal distribution ? Copulas are flexible distribution functions which are more suitable to model data with particular or special distribution, especially for those distributions with asymmetric dependency between variables. Therefore, in this section, we are going to introduce the concept of copulas and vine copulas and the definitions and theories presented will follow Czado [2019]. Also, the related proofs are mostly skipped.

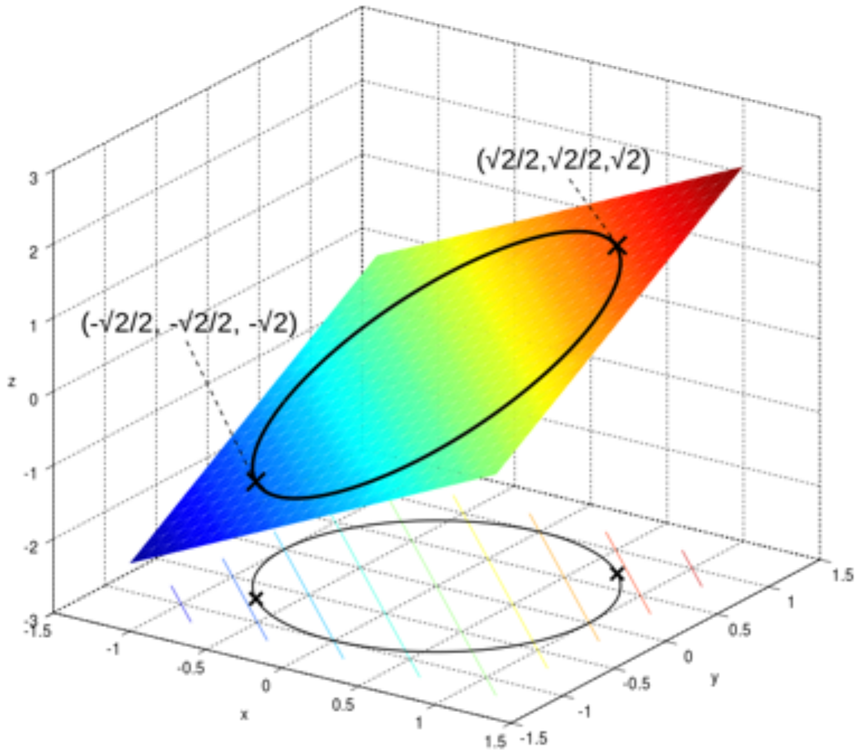


Figure 2.2.1: Illustration of the constrained optimization problem in Example 2.2.1

2.3.1 Concept of Copulas

Copula is a special multivariate distribution function and the marginal distribution of each variable for a Copula is uniform on the interval $[0, 1]$. Here is the definition of the Copula and its density:

Definition 2.3.1 (Copula and copula density)

A d-dimensional copula C is a multivariate distribution function

$$C : [0, 1]^d \to [0, 1]$$

with uniformly distributed marginals and it is given by

$$C(u_1, ..., u_d) := P(U_1 \leq u_1, ..., U_d \leq u_d)$$

where $U_1, U_2, ..., U_d$ are the variables uniformly distributed. The corresponding copula density denote by c can be obtained by partial differentiation, i.e.

$$c(u_1, ..., u_d) := \frac{\partial^d}{\partial u_1 ... \partial u_d} C(u_1, ..., u_d) \text{ for all } \boldsymbol{u} \text{ in } [0, 1]^d$$

Example 2.3.1 (Independence Copula and Comonotonicity copula)

Independence Copula
Let $U_1, U_2, ..., U_d$ be independent and random variables follow uniform distribution on $[0, 1]$. The independence Copula is

$$\begin{aligned} C(u_1, u_2, ..., u_d) &= P(U_1 \leq u_1, ..., U_d \leq u_d) \\ &= P(U_1 \leq u_1) \cdots P(U_d \leq u_d) \\ &= u_1 u_2 \cdots u_d. \end{aligned}$$

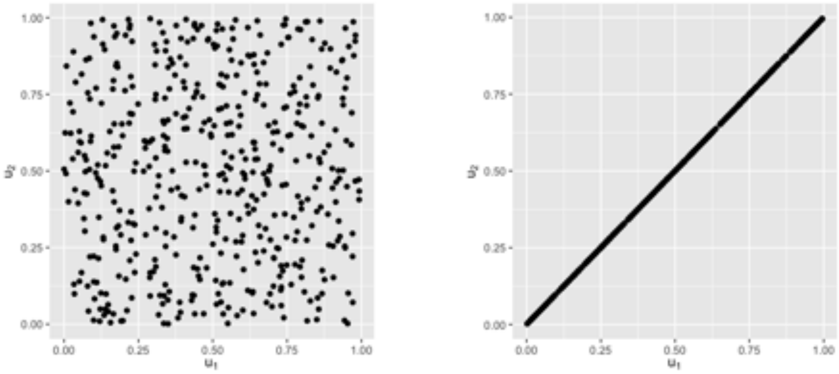


Figure 2.3.1: The scatter plot of the 500 data generated from independence copula (left) and comonotonicity copula (right)

Comonotonicity Copula

Let $U_1, U_2, ..., U_d$ be completely dependent that $U_1 = U_2 = \cdots = U_d$. The comonotonicity copula is

$$\begin{aligned} C(u_1, u_2, ..., u_d) &= P(U_1 \leq u_1, ..., U_d \leq u_d) \\ &= P(U_1 \leq u_1, ..., U_1 \leq u_d) \\ &= P(U_1 \leq \min\{u_1, u_2, ..., u_d\}) \\ &= \min\{u_1, u_2, ..., u_d\}. \end{aligned}$$

The illustations of the bivariate independence copula and bivariate comonotonicity copula are shown in the Figure 2.3.1.

If we want to model the dependence among random variables using Copula, we need to transform the random variables to uniformly distributed first. For this, we use the probability integral transform:

Definition 2.3.2 (Probability integral transform)

Let X be a continuous random variable following distribution function F and x is an observed value of X, then the transformation $u := F(x)$ is called the Probability integral transform (PIT) at x. Furthermore, the random variable $X \sim F$ after probability integral transform follows uniform distribution, such that $U := F(X)$, since

$$P(U \leq u) = P(F(X) \leq u) = P(X \leq F^{-1}(u)) = F(F^{-1}(u)) = u$$

We suppose that $X = (X_1, ..., X_d)^T$ is a d-dimensional continuous random variable and the corresponding marginal distribution functions are F_j, such that $X_j \sim F_j, j = 1, ..., d$. We can construct so called copula data $(U_1, ..., U_d)$ which is uniformly distributed by using probability integral transform, such that

$$(U_1, ..., U_d) = (F_1(X_1), ..., F_d(X_d)).$$

The resulting data is called as in u-scale or copula-scale.

Normally, we don't know the distribution of the data we have. We use the empirical distribution function instead to construct the Copula data:

Definition 2.3.3 (Empirical distribution function)

Let $x_1, ..., x_n$ be an i.i.d. sample from a distribution function F, then the empirical distribution function is defined as

$$\hat{F}(x) := \frac{1}{n+1} \sum_{i=1}^{n} 1_{\{x_i \leq x\}} \text{ for all } x,$$

where $\hat{F}$ is also called estimated marginal distributions.

Assume that we have the data $\{(x_{i1}, x_{i2}, i = 1, .., n)\}$. We can transform the data to the copula data by using the estimated marginal distributions $\hat{F}_j, j = 1, 2$:

$$(u_{i1}, u_{i2}) := (\hat{F}_1(x_{i1}), \hat{F}_2(x_{i2})) \text{ for } i = 1, ..., n. \tag{2.3.1}$$

Now, we are going to show an important theory called Sklar's Theorem by Sklar [1959] that we can construct the multivariate distribution of $X_1, X_2, ..., X_d$ by the marginal distributions $F_j(x)$ and the copula C.

Definition 2.3.4 (Sklar's Theorem)

Let X be a d-dimensional random vector with joint distribution function F and marginal distribution function F_i, $i = 1, ..., d$, then the joint distribution function can be expressed as

$$F(x_1, ..., x_d) = C(F_1(x_1), ..., F_d(x_d))$$

with associated density or probability mass function

$$f(x_1, ..., x_d) = c(F_1(x_1), ..., F_d(x_d))f_1(x_1)...f_d(x_d)$$

for some d-dimensional copula C with copula density c. For absolutely continuous distributions, the copula C is unique. The inverse also holds: the copula corresponding to a multivariate distribution function F with marginal distribution functions F_i for $i = 1, ..., d$ can be expressed as

$$C(u_1, ..., u_d) = F(F_1^{-1}(u_1), ..., F_d^{-1}(u_d)) \tag{2.3.2}$$

and its copula density or probability mass function is determined by

$$c(u_1, ..., u_d) = \frac{f(F_1^{-1}(u_1), ..., F_d^{-1}(u_d))}{f_1(F_1^{-1}(u_1)...f_d(F_d^{-1}(u_d))}$$

Using the inverse of Sklar's Theorem (2.3.2), bivariate Gaussian copula and bivariate Student's t copula belonging to elliptical copulas can be constructed easily.

Example 2.3.2 (Bivariate Gaussian copula)

The bivariate Gaussian copula can be constructed by using the bivariate normal distribution with zero mean vector, unit variances, and correlation ρ, and the inverse of Sklar's Theorem (2.3.2). It is given by

$$C(u_1, u_2; \rho) := \Phi_2(\Phi_1^{-1}(u_1), \Phi_1^{-1}(u_2); \rho)$$

where $\Phi(\cdot)$ is the distribution function of a standard normal $N(0, 1)$ distribution and $\Phi_2(\cdot, \cdot; \rho)$ is the bivariate normal distribution function with zero means, unit variances, and correlation ρ.

Example 2.3.3 (Bivariate Student's t copula)

The bivariate Student's t copula can be constructed by using the bivariate Student's t distribution with v degrees of freedom, zero mean, and correlation ρ, and the inverse of Sklar's Theorem (2.3.2). It is given by

$$C(u_1, u_2; v, \rho) := T_{2,v}(T_v^{-1}(u_1), T_v^{-1}(u_2); v, \rho)$$

where $T_v(\cdot)$ is the the distribution function of Student's t with degrees of freedom v and $T_{2,v}(\cdot, \cdot; v, \rho)$ is the the bivariate Student's t distribution function with degrees of freedom v and correlation ρ.

Now, we would like to introduce other well known copulas called bivariate Archimedean copulas not belonging to elliptical copulas. The copulas are not constructed by using the inverse of Sklar's Theorem (2.3.2) and the definition of bivariate Archimedean copulas is shown in the following:

Definition 2.3.5 (Bivariate Archimedean copulas)

Let Ω be the set of all continuous, strictly monotone decreasing, and convex functions $\varphi : I \to [0, \infty]$ with $\varphi(1) = 0$. Let $\varphi \in \Omega$, then

$$C(u_1, u_2) = \varphi^{[-1]}(\varphi(u_1) + \varphi(u_2))$$

is a copula. C is called a bivariate Archimedean copula with generator φ. Is $\varphi(0) = \infty$, the generator is called strict. Here $\varphi^{[-1]}$ is the pseudo-inverse of φ, which is defined as $\varphi^{[-1]} : [0, \infty] \to [0, 1]$ with

$$\varphi^{[-1]}(t) := \begin{cases} \varphi^{-1}(t) & , 0 \leq t \leq \varphi(0) \\ 0 & , \varphi(0) \leq t \leq \infty \end{cases}$$

By choosing the suitable generator φ, bivariate Archimedean copulas can be structured and some examples of parametric bivariate Archimedean copulas with a single parameter are shown in the Example 2.3.4 For the Archimedean copulas with two parameters, please refer to Czado [2019] for more details.

Example 2.3.4 (Parametric bivariate Archimedean copulas with a single parameter)

Bivariate Clayton copula
The bivariate Clayton copula is given by

$$C(u_1, u_2; \delta) := (u_1^{-\delta} + u_2^{-\delta} - 1)^{-\frac{1}{\delta}}$$

where the parameter of dependence $\delta \in (0, \infty)$ and the generator function φ is given by

$$\varphi(x) = \frac{1}{\delta}(x^{-\delta} - 1),$$

Bivariate Gumbel copula
The bivariate Gumbel copula is given by

$$C(u_1, u_2; \delta) := exp\left\{- \left[(-\log(u_1))^{\delta} + (-\log(u_2))^{\delta}\right]^{\frac{1}{\delta}}\right\}$$

where the parameter of dependence $\delta \in [1, \infty]$ and the generator function φ is given by

$$\varphi(x) = (-\log(x))^{\delta}$$

Bivariate Frank copula
The bivariate Frank copula is given by

$$C(u_1, u_2; \delta) := -\frac{1}{\delta}\log\left(1 + \frac{(exp(-\delta u_1) - 1)(exp(-\delta u_2) - 1)}{exp(-\delta) - 1}\right)$$

where the parameter of dependence $\delta \in \mathbb{R}/\{0\}$ and the generator function φ is given by

$$\varphi(x) = -\log(\frac{exp(-\delta x) - 1}{exp(-\delta) - 1}),$$

Bivariate Joe copula
The bivariate Joe copula is given by

$$C(u_1, u_2; \delta) := 1 - ((1 - u_1)^{\delta} + (1 - u_2)^{\delta} - (1 - u_1)^{\delta}(1 - u_2)^{\delta})^{\frac{1}{\delta}}$$

where the parameter of dependence $\delta \in [1, \infty]$ and the generator function φ is given by

$$\varphi(x) = -\log(1 - (1 - x)^{\delta})$$

You may find that the parameters of dependence in elliptical copulas and Archimedean copulas are different. In the Gaussian copula and Student's t copula, parameters of dependence is correlation $\rho \in [-1, 1]$. But, in the Archimedean copulas, the parameters of dependence are different and they have even different domains. In order to make the parameters of dependence consistent in different copulas for better comparison, we would like to introduce another measure for the dependency between two variables called Kendall rank correlation coefficient or Kendall's tau that we can use this measure only to represent the dependency for each of the copulas.

Definition 2.3.6 (Kendall's tau)

The Kendall's τ between the continuous random variables X_1 and X_2 is defined as

$$\tau(X_1, X_2) = P((X_{11} - X_{21})(X_{12} - X_{22}) > 0) - P((X_{11} - X_{21})(X_{12} - X_{22}) < 0),$$

where (X_{11}, X_{12}) and (X_{21}, X_{22}) are independent and identically distributed copies of (X_1, X_2).

Definition 2.3.7 (Concordant, discordant and extra pairs)

Let (X_1, X_2) be two continuous random variables and $\{x_{i1}, x_{i2}, i = 1, ..., n\}$ are the corresponding samples. Then, (x_{j1}, x_{j2}) and (x_{k1}, x_{k2}) are called concordant pairs when

$$(x_{j1} - x_{k1})(x_{j2} - x_{k2}) > 0,$$

and called discordant pairs when

$$(x_{j1} - x_{k1})(x_{j2} - x_{k2}) < 0,$$

and called extra x_1 pair or extra x_2 pair when

$$x_{j1} = x_{k1} \text{ or } x_{j2} = x_{k2}$$

Definition 2.3.8 (Estimate of Kendall's tau)

Let N_c be the number of concordant pairs, N_d be the number of discordant pairs, N_1 be the number of extra x_1 pairs, and N_2 be the number of extra x_2 pairs of random sample $x_{i1}, x_{i2}, i = 1, ..., n$ from the joint distribution of (X_1, X_2). Then an estimate of Kendall's ρ is given by

$$\hat{\tau} = \frac{N_c - N_d}{\sqrt{N_c + N_d + N_1} \times \sqrt{N_c + N_d + N_2}}$$

The properties and interpretation way for Pearson correlation ρ and Kendall's τ are quite similar and they are shown in the following: (1) The value of Kendall's τ is between -1 and 1. (2) $\tau = 1$ indicates a perfect positive monotonous relation between two variables. (3) $\tau = 0$ indicates a perfect positive monotonous relation. (4) $\tau = -1$ indicates a perfect negative monotonous relation. (5) If two variables are independent, $\tau = 0$, but the reverse does not always hold.

We have mentioned that Kendall's tau can be used solely to represent the dependency for each of the copulas. Also, Table 2.3.1 shows the relationship between Kendall's τ and copula parameters for different bivariate copula.

Example 2.3.5 (Simulation of the copula data for single parameter copula families)

In the Example 2.3.5, the scatter plots of 500 data simulated from 6 introduced bivariate copulas with single parameter for different dependency levels are shown in Figure 2.3.2. Kendall's tau $\tau = 0.2, \tau = 0.5$ and $\tau = 0.8$ represent the weak, middle and strong dependency level respectively. For the weak dependency level ($\tau = 0.2$), the scatter plots look like independence copula, whose Kendall's tau is 0. As the dependency level increases,

Table 2.3.1: Kendall's τ and copula parameters for different bivariate copula families with single parameter and Student's t copula

Family	Kendall's τ	Range of τ
Gaussian	$\tau = \frac{2}{\pi}\arcsin(\rho)$	[-1,1]
Student's t	$\tau = \frac{2}{\pi}\arcsin(\rho)$	[-1,1]
Gumbel	$\tau = 1 - \frac{1}{\delta}$	[0,1]
Clayton	$\tau = \frac{\delta}{\delta+2}$	[0,1]
Frank	$\tau = 1 - \frac{4}{\delta} + 4\frac{D_1(\delta)}{\delta}$ with $D_1(\delta) = \int_o^\delta \frac{x/\delta}{e^x-1}dx$ (Debye function)	[-1,1]
Joe	$\tau = 1 + \left(\frac{-2+2\gamma+2\ln(2)+\psi(\frac{1}{\delta})+\psi(\frac{2+\delta}{2\delta})+\delta}{-2+\psi(\delta)}\right)$ with Euler constant $\gamma = 0.57721$ and digamma function $\psi(x)$	[0,1]

the tail dependencies for each copula can be observed easier. We can see that Gaussian, Student's t and Frank are copulas with symmetric tails and Clayton, Gumbel and Joe are copulas with asymmetric tails. For stronger dependency level ($\tau = 0.5$ or 0.8), we can see the tail dependence easier and we will talk about this later after the introduction of tail dependence coefficient.

Definition 2.3.9 (Upper and lower tail dependence coefficient)

Suppose we have continuous random X_1 and X_2 with marginal distributions F_1 and F_2 respectively. The upper tail dependence coefficient of a bivariate distribution with copula C is defined as

$$\lambda^{upper} = \lim_{t\to 1^-} P(X_2 > F_2^{-1}|X_1 > F_1^{-1}) = \lim_{t\to 1^-} \frac{1-2t+C(t,t)}{1-t}$$

and the lower tail dependence coefficient is

$$\lambda^{lower} = \lim_{t\to 0^+} P(X_2 \leq F_2^{-1}|X_1 \leq F_1^{-1}) = \lim_{t\to 0^+} \frac{C(t,t)}{t}$$

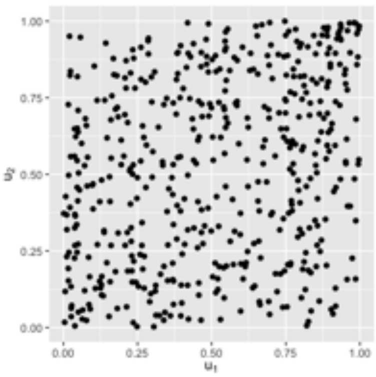
Tail dependence coefficient is a measure for the dependency level between extreme values of two random variables. We say random variables (X_1, X_2) are upper tail dependent when $\lambda^{upper} > 0$ and upper tail independent when $\lambda^{upper} = 0$. Similarly, $\lambda^{lower} > 0$ means lower tail dependent and $\lambda^{lower} = 0$ means lower tail independent. For example, for strong upper tail dependence between random variables (X_1, X_2), we can expect that when the value of X_1 is very large, very likely the value of X_2 is very large too.

The Table 2.3.2 shows you the tail dependence coefficients of different bivariate copula families in terms of their copula parameters. We can know that Student's t copula is tail dependent and Clayton copula is upper tail dependent and lower tail independent and so on. The result is quite consistent to the corresponding scatter plots in Figure 2.3.2. For example, for Clayton, there are quite a lot of points around the value 0 of u_1 and u_2 in the lower tail. Compared to the lower tail, there are just a few point around the value 1 in the upper tail. Therefore, Clayton copula is upper tail dependent and not lower tail dependent.

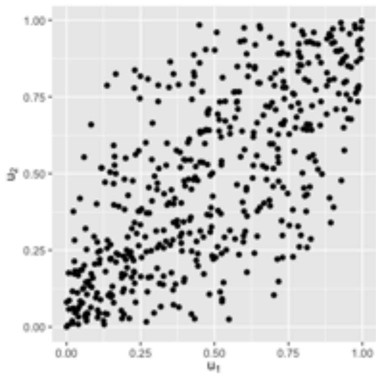
In the Table 2.3.1, you may notice that for some of the copula families, just the positive value of the Kendall's tau is well defined, for example, Gumbel, Clayton and Joe which are asymmetric copulas. Therefore, we would like to define rotated copulas which allow us to model negative correlation between two variables for those asymmetric copulas.

Definition 2.3.10 (Rotated Copulas)

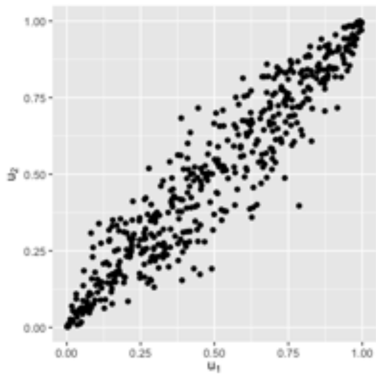
The counterclockwise rotations of the copula density $c(u_1, u_2)$ are given by



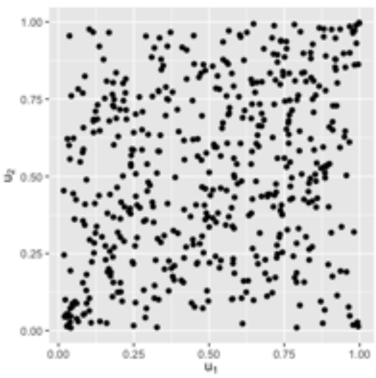
(a) Gaussian, τ=0.2



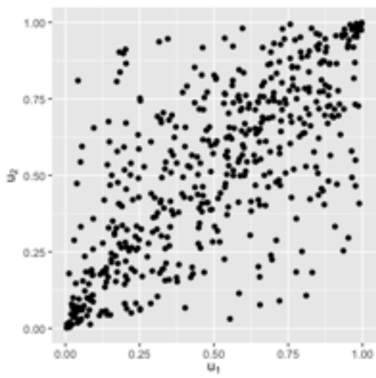
(b) Gaussian, τ=0.5



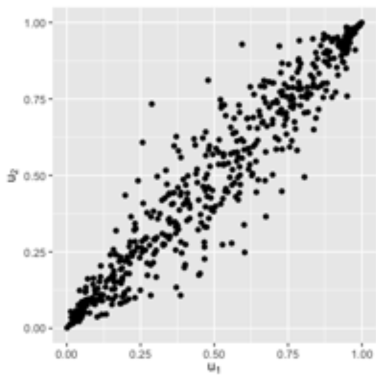
(c) Gaussian, τ=0.8



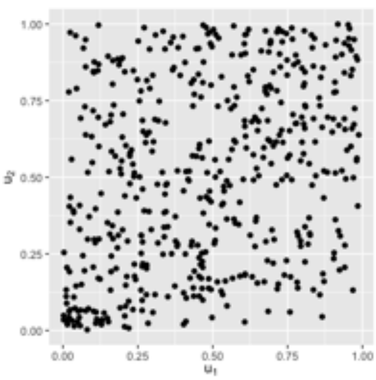
(d) Student's t, τ=0.2, v=3



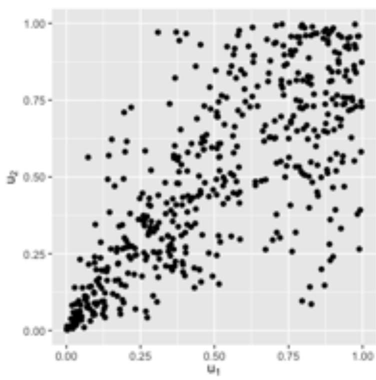
(e) Student's t, τ=0.5, v=3



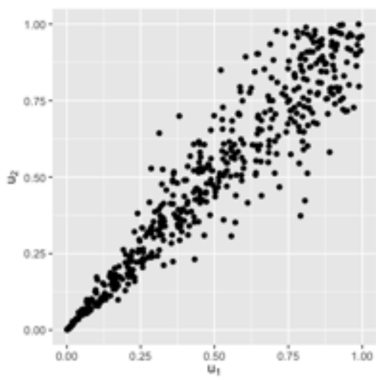
(f) Student's t, τ=0.8, v=3



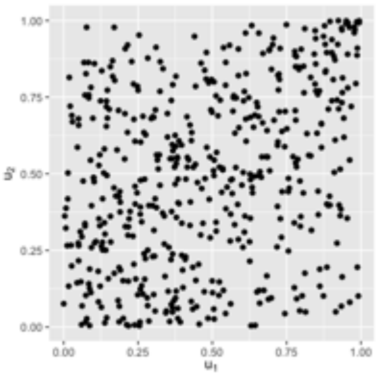
(g) Clayton, τ=0.2



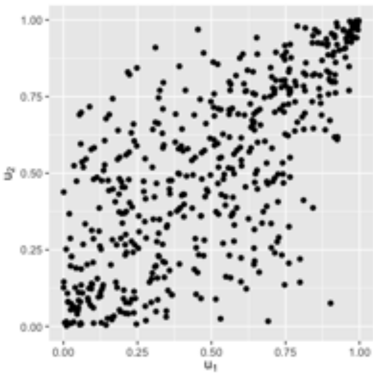
(h) Clayton, τ=0.5



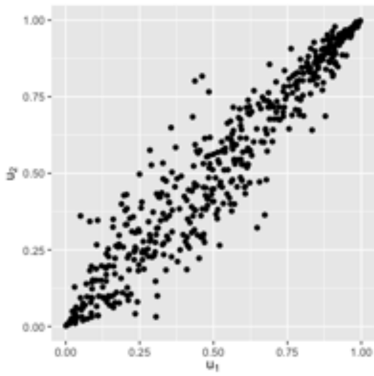
(i) Clayton, τ=0.8



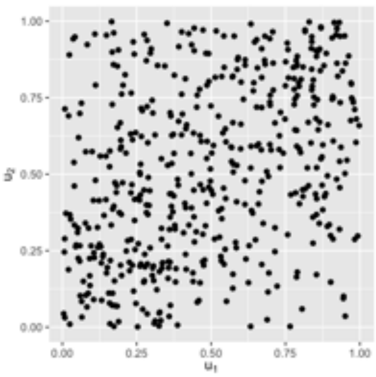
(j) Gumbel, τ=0.2



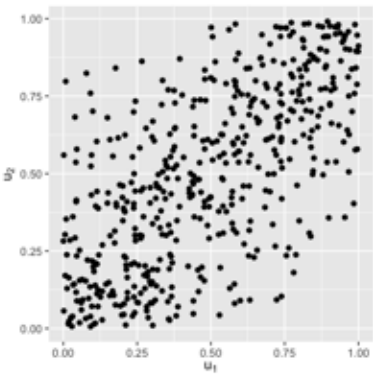
(k) Gumbel, τ=0.5



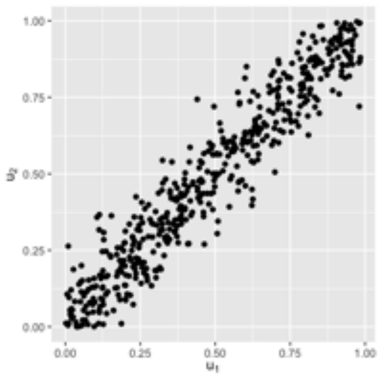
(l) Gumbel, τ=0.8



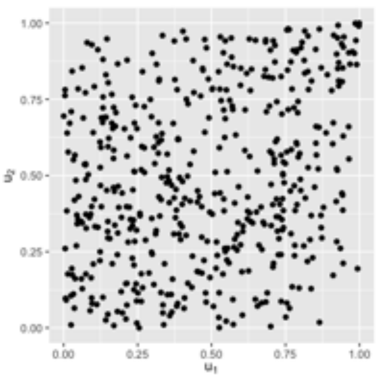
(m) Frank, τ=0.2



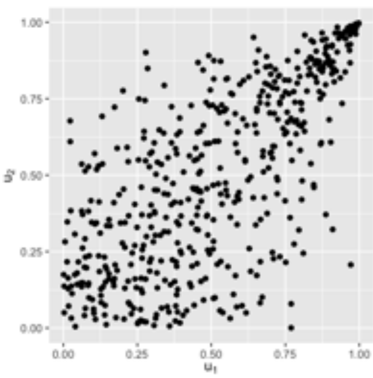
(n) Frank, τ=0.5



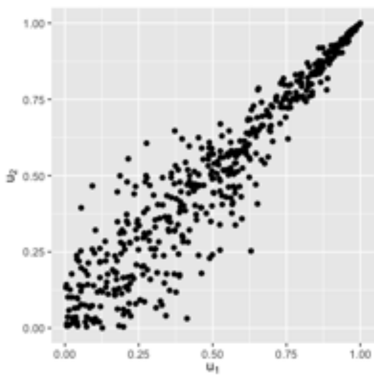
(o) Frank, τ=0.8



(p) Joe, τ=0.2



(q) Joe, τ=0.5



(r) Joe, τ=0.8

Figure 2.3.2: Scatter plots of the 500 data simulated from different copulas with single parameter for weak (τ=0.2), middle (τ=0.5) and strong (τ=0.8) dependence.

Table 2.3.2: Tail dependence coefficients of different bivariate copula families with single parameter

Family	Upper tail dependence	Lower tail dependence
Gaussian	0	0
Student's t	$2\mathrm{t}_{v+1}\left(-\sqrt{v+1}\sqrt{\frac{1-\rho}{1+\rho}}\right)$	$2\mathrm{t}_{v+1}\left(-\sqrt{v+1}\sqrt{\frac{1-\rho}{1+\rho}}\right)$
Gumbel	2-$2^{1/\delta}$	0
Clayton	0	$2^{-1/\delta}$
Frank	0	0
Joe	2-$2^{1/\delta}$	0

- **90 degrees :** $c_{90}(u_1, u_2) = c(1 - u_2, u_1)$
- **180 degrees :** $c_{180}(u_1, u_2) = c(1 - u_1, 1 - u_2)$
- **270 degrees :** $c_{270}(u_1, u_2) = c(u_2, 1 - u_1)$

After defining the rotated copulas, we can for example extend the Clayton copula to a copula with a full range of Kendall's tau values by defining

$$c_{clayton}^{extended}(u_1, u_2; \tau) := \begin{cases} c_{clayton}(u_1, u_2) \ , \ \text{if } \tau > 0 \\ c_{clayton,90}(u_1, u_2) \text{ or } c_{clayton,270}(u_1, u_2) \ , \ \text{if } \tau \leq 0 \end{cases}$$

Example 2.3.6 (Simulation of the rotated copula data for Clayton)

In the Example 2.3.6, the scatter plots of 500 data simulated from Clayton and its rotated copulas with Kendall's tau ($\tau = 0.8$) are shown in the Figure 2.3.3.

After the introduction of the characteristics of different bivariate copulas, now we would like to talk about the parameter estimation for bivariate copula models. Assume that we have the copula data $\boldsymbol{u} = \{(u_{i1}, u_{i2}), i = 1, .., n\}$ following bivariate copula C with parameter θ. Then, we can use maximum likelihood estimation (MLE) introduced in Section 2.1.1 to estimate the parameter of the copula C. The likelihood function is given by

$$l(\theta; \boldsymbol{u}) = \prod_{i=1}^{n} c(u_{i1}, u_{i2}; \theta),$$

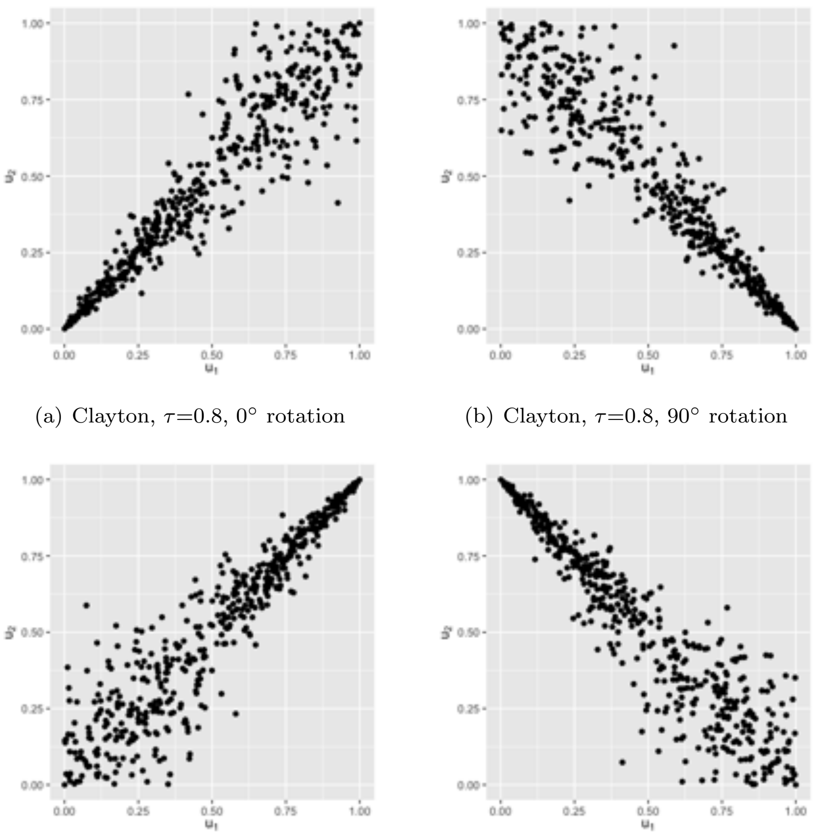
and the estimated parameter can be found by maximizing the likelihood. The maximum likelihood estimator $\hat{\theta}_{MLE}$ is given by

$$\hat{\theta}_{MLE} = \arg\max_{\boldsymbol{\theta}\in\Theta} \prod_{i=1}^{n} c(u_{i1}, u_{i2}; \theta).$$

where c is the density function of the copula C and $\boldsymbol{\Theta}$ denotes the parameter space.

From the Definition (2.3.4) (Sklar's Theorem), we know that we can construct the multivariate distribution by the marginal distributions and the copula and we know how to fit the bivariate copula data by a bivariate copula. Now, the question is whether we can fit the data with three dimension by bivariate copulas, or even higher dimension data. Luckily, a multivariate density function can be decomposed into different pair copulas and marginal distributions. Therefore, the answer is feasible. Now, we will start with some definitions used in the pair copula decomposition in three dimensions.

Definition 2.3.11 (Conditional densities and distribution functions of bivariate distribution)



(a) Clayton, τ=0.8, 0° rotation

(b) Clayton, τ=0.8, 90° rotation

(c) Clayton, τ=0.8, 180° rotation

(d) Clayton, τ=0.8, 270° rotation

Figure 2.3.3: Scatter plots of the 500 data simulated from Clayton copula and its rotated copulas for $\tau = 0.8$.

The conditional density and distribution function of X_1 given $X_2 = x_2$ can be rewritten as

$$\begin{aligned} f_{1|2}(x_1|x_2) &= c_{12}(F_1(x_1), F_2(x_2))f_1(x_1) \\ F_{1|2}(x_1|x_2) &= \frac{\partial}{\partial u_2} C_{12}(F_1(x_1), u_2)|_{u_2=F_2(x_2)} \\ &= \frac{\partial}{\partial F_2(x_2)} C_{12}(F_1(x_1), F_2(x_2)) \\ &= C_{1|2}(F_1(x_1), F_2(x_2)) = C_{1|2}(u_1|u_2) \end{aligned}$$

where $C_{1|2}(u_1|u_2) = \frac{\partial}{\partial u_2} C_{12}(u_1, u_2)$.

Definition 2.3.12 (Simplifying assumption)

When the simplifying assumption is satisfied, the copula density associated with the conditional distribution of (X_1, X_3) given $X_2 = x_2$ is given by for any $x_2 \in \mathbb{R}$

$$c_{1,3;2}(u_1, u_3; x_2) = c_{1,3;2}(u_1, u_3) \text{ for } u_1 \in [0, 1], u_3 \in [0, 1].$$

Theorem 2.3.13 (A pair copula construction in three dimensions)

Let X_1, X_2 and X_3 be random variables. The density function and marginal distribution are denoted as f_i and F_i respectively and the conditional distribution function of X_i given X_j is denoted by $F_{i|j}$. Their joint density is given by $f(x_1, x_2, x_3)$. Suppose that the simplifying assumption for copulas in the Definition (2.3.12) holds. A pair copula construction of an arbitrary three dimensional density is given as following recursions:

$$\begin{aligned} f(x_1, x_2, x_3) &= c_{1,3;2}(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)) \times c_{2,3}(F_2(x_2), F_3(x_3)) \\ &\quad \times c_{1,2}(F_1(x_1), F_2(x_2))f_3(x_3)f_2(x_2)f_1(x_1) \end{aligned}$$

where $c_{1,3;2}$ denotes the copula density associated with the conditional distribution of (X_1, X_3) given $X_2 = x_2$. From this construction we see that the joint three dimensional density can be expressed in terms of bivariate copulas and conditional distribution functions. However this construction is not unique, since

$$\begin{aligned} f(x_1, x_2, x_3) &= c_{1,2;3}(F_{1|3}(x_1|x_3), F_{2|1}(x_2|x_2)) \times c_{1,3}(F_1(x_1), F_3(x_3)) \\ &\quad \times c_{2,3}(F_2(x_2), F_3(x_3))f_3(x_3)f_2(x_2)f_1(x_1) \end{aligned}$$

and

$$\begin{aligned} f(x_1, x_2, x_3) &= c_{2,3;1}(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)) \times c_{1,3}(F_1(x_1), F_3(x_3)) \\ &\quad \times c_{1,2}(F_1(x_1), F_2(x_2))f_3(x_3)f_2(x_2)f_1(x_1) \end{aligned}$$

are two different construction.

Theorem 2.3.14 (Pair copula construction of a joint parametric density in three dimensions)

A parametric pair copula construction in three dimensions specifies a three dimensional density with copula parameter vector $\boldsymbol{\theta} = (\theta_{1,2}, \theta_{2,3}, \theta_{1,2;3})^T$ modelling the dependence and marginal densities and distributions are given by $f_j, F_j, j = 1, 2, 3$ respectively with marginal parameter vector $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \gamma_3)^T$.

$$\begin{aligned} f(x_1, x_2, x_3) &= c_{1,3;2}(F_{1|2}(x_1|x_2; \gamma_1, \gamma_2, \theta_{1,2}), F_{3|2}(x_3|x_2; \gamma_3, \gamma_2, \theta_{2,3}); \theta_{1,3;2}) \\ &\quad \times c_{2,3}(F_2(x_2; \gamma_2), F_3(x_3; \gamma_3); \theta_{2,3}) \\ &\quad \times c_{1,2}(F_1(x_1; \gamma_1), F_2(x_2; \gamma_2), ; \theta_{1,2}) \ f_3(x_3; \gamma_3) \ f_2(x_2; \gamma_2) \ f_1(x_1; \gamma_1) \end{aligned}$$

where $c_{1,3;2}(\cdot, \cdot; \theta_{1,3;2}), c_{1,2}(\cdot, \cdot; \theta_{1,2}), c_{2,3}(\cdot, \cdot; \theta_{2,3})$ are arbitrary parametric bivariate copula densities.

2.3.2 Vine Copulas

In the last section, we have introduced some basic concepts of the copula and the characteristics of different bivariate copulas. Also, we have shown that a distribution with 3 variables can be constructed by using bivariate copulas only and importantly, the construction is not unique. Therefore, we would like to introduce vine copulas which use tree structure in graph theory to represent the particular pair copula construction in higher dimensions. [Czado, 2019]

Here shows you the related definitions of the vine copula and the definitions presented follow Czado [2019] :

Definition 2.3.15 (Regular (R-) vine tree sequence)

The set of trees $\mathcal{V} = (T_1, ..., T_{d-1})$ is a regular vine tree sequence on d elements if:

(1) Each tree $T_j = (N_j, E_j)$ is connected, i.e. for all nodes $a, b \in T_j, j = 1, ..., d-1$, there exists a path $n_1, ..., n_k \subseteq N_j$ with $a = n_1, b = n_k$.

(2) T_1 is a tree with node set $N_1 = \{1, ..., d\}$ and edge set E_1.

(3) For $j \leq 2, T_j$ is a tree with node set $N_j = E_{j-1}$ and edge set E_j.

(4) For $j = 2, ..., d-1$ and $\{a, b\} \in E_j$ it must be hold that $|a \cap b| = 1$.

Remark 2.3.16 (Proximity condition)

The property (4) in the Definition (2.3.15) is called the proximity condition. It ensures that if there is an edge e connecting a and b in tree $T_j, j \geq 2$, then a and b (which are edges in T_{j-1}) must share a common node in T_{j-1}.

Definition 2.3.17 (Complete union and conditioned sets)

For any edge $e \in E_i$ define the set

$$A_e := \{j \in N_1 \mid \exists e_1 \in E_1, .., e_{i-1} \in E_{i-1} \text{ such that } j \in e_1 \in ... \in e_{i-1} \in e\}.$$

The set A_e is called the complete union of the edge e. The conditioning set D_e of an edge $e = \{a, b\}$ is defined by

$$D_e := A_a \cap A_b$$

and the conditioned sets $\mathcal{C}_{e,a}$ and $\mathcal{C}_{e,b}$ are given by

$$\mathcal{C}_{e,a} := A_a \setminus D_e,\ \mathcal{C}_{e,b} := A_b \setminus D_e \text{ and } \mathcal{C}_e := \mathcal{C}_{e,a} \cup \mathcal{C}_{e,b}.$$

We often abbreviate each edge $e = (\mathcal{C}_{e,a}, \mathcal{C}_{e,b}; D_e)$ in the vine tree sequence by

$$e = (e_a, e_b; D_e).$$

Definition 2.3.18 (Regular vine distribution)

The joint distribution F for the d dimensional random vector $\boldsymbol{X} = (X_1, ..., X_d)$ has a regular vine distribution, if we can specify a triplet $(\mathcal{F}, \mathcal{V}, \mathcal{B})$ such that:

(1) Marginal distributions: $\mathcal{F} = (F_1, ..., F_d)$ is a vector of continuous invertible marginal distribution functions, representing the marginal distribution functions of the random variable $X_i, i = 1, ..., d$.

(2) Regular vine tree sequence: $\mathcal{V}$ is an R-vine tree sequence on d elements.

(3) Bivariate copulas: The set $\mathcal{B} = \{C_e | e \in E_i; i = 1, ..., d-1\}$, where is a symmetric bivariate copula with density. Here E_i is the edge set of tree T_i in the R-vine tree sequence $\mathcal{V}$.

(4) Relationship between R-vine tree sequence $\mathcal{V}$ and the set $\mathcal{B}$ of bivariate copulas: For each $e \in E_i,\ i = 1, .., d-1,\ e = [a, b],\ C_e$ is the copula associated with the conditional distribution of $X_{\mathcal{C}_{e,a}}$ and $X_{\mathcal{C}_{e,b}}$ given $\boldsymbol{X}_{D_e} = \boldsymbol{x}_{D_e}$. Further $C_e(.,.)$ does not depend on the specific value of $\boldsymbol{x}_{D_e}$.

Definition 2.3.19 (Pair copula and copula density associated with edge e)

We will denote the copula C_e corresponding to edge e by $C_{\mathcal{C}_{e,a}\mathcal{C}_{e,b};D_e}$ and the corresponding density by $c_{\mathcal{C}_{e,a}\mathcal{C}_{e,b};D_e}$, respectively. This copula is also called a pair copula.

Definition 2.3.20 (Existence of a regular vine distribution)

Assume that $(\mathcal{F}, \mathcal{V}, \mathcal{B})$ satisfy the properties (1)–(3) of the Definition (2.3.18), then there is a unique d dimensional distribution F with density

$$f_{1,..,d}(x_1, ..., x_d) = f_1(x_1) \cdots f_d(x_d) \prod_{i=1}^{d-1} \prod_{e \in E_i} c_{\mathcal{C}_{e,a}\mathcal{C}_{e,b};D_e}(F_{\mathcal{C}_{e,a}|D_e}(x_{\mathcal{C}_{e,a}}|\boldsymbol{x}_{D_e}), F_{\mathcal{C}_{e,b}|D_e}(x_{\mathcal{C}_{e,b}}|\boldsymbol{x}_{D_e})),$$

such that for each $e \in E_i,\ i = 1, ..., d-1$, with $e = \{a, b\}$ we have for the distribution function of $X_{\mathcal{C}_{e,a}}$ and $X_{\mathcal{C}_{e,b}}$ given $\boldsymbol{X}_{D_e} = \boldsymbol{x}_{D_e}$

$$F_{\mathcal{C}_{e,a}\mathcal{C}_{e,b}|D_e}(x_{\mathcal{C}_{e,a}}, x_{\mathcal{C}_{e,b}}|\boldsymbol{x}_{D_e}) = C_e(F_{\mathcal{C}_{e,a}|D_e}(x_{\mathcal{C}_{e,a}}|\boldsymbol{x}_{D_e}), F_{\mathcal{C}_{e,b}|D_e}(x_{\mathcal{C}_{e,b}}|\boldsymbol{x}_{D_e})).$$

Further the one dimensional marginal distributions of F are given by $F_i(x_i), i = 1, ..., d$. The proof is skipped here. For more details, please refer to Bedford and Cooke [2002] and Czado [2019].

Example 2.3.7 (3-dimensional vine copula)

In the Figure 2.3.4, the vine tree sequence $\mathcal{V}$ has two tree level T_1 and T_2. The node set $N_1 = \{1, 2, 3\}$ and the edge set $E_1 = \{\{2, 1\}, \{1, 3\}\}$ are contained in the first tree T_1. In the Definition (2.3.15) (3), we know that for second tree T_2, the node set $N_2 = E_1$. Therefore, in the second tree level, the node set N_2 is $\{\{2, 1\}, \{1, 3\}\}$. According to the Definition (2.3.17), we can know the edge set of the second tree $E_2 = \{\{2, 3; 1\}\}$. The reason is in the following: Since $A_a = \{1, 2\}$ and $A_b = \{1, 3\}$, the conditioning set $D_e = \{1\}$. The conditioned sets $\mathcal{C}_{e,a}$ and $\mathcal{C}_{e,b}$ are $\{2\}$, $\{3\}$ respectively. Therefore, the edge E_2 can be represent by $\{\{2, 3; 1\}\}$.

In the first tree T_1, the number of nodes is the number of the variables of the density function. In our case, we have 3 nodes because we construct a 3 dimensional density. Also, the number inside the node represents the corresponding variable, for example, 2 means X_2 and 3 means X_3. According to the Definition (2.3.18) (4), the edge between two nodes represents the pair copula associated with the corresponding nodes. For example, the edge 2,1 in the first tree represents the copula density $c_{1,2}$ (or $c_{2,1}$). You can observe that the edge set in the vine copula is exactly the same as the subscript of the copula density $c_{2,3;1}$, $c_{1,3}$ and $c_{1,2}$. So, each pair copula construction can be represented by a particular vine tree structure.

By the Definition (2.3.20), the following pair copula construction of a 3 dimensional density

$$\begin{aligned} f(x_1, x_2, x_3) = {} & c_{2,3;1}(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)) \times c_{1,3}(F_1(x_1), F_3(x_3)) \\ & \times c_{1,2}(F_1(x_1), F_2(x_2)) f_3(x_3) f_2(x_2) f_1(x_1) \end{aligned}$$

can be represented by the vine copula tree structure in the Figure 2.3.4.

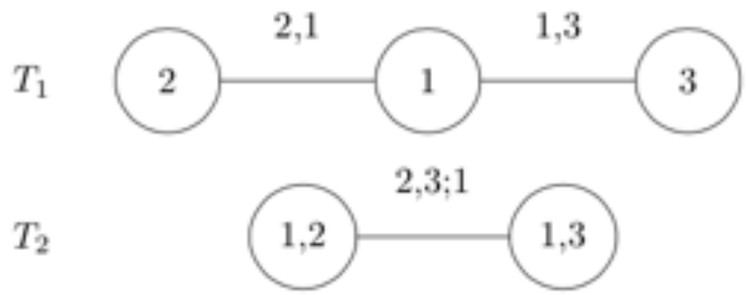


Figure 2.3.4: An example of 3-dimensional vine copula

2.4 Mixture Models

To ease the explanation, we discuss the ECM algorithm and each CM steps just updates 1 parameter θ by maximization with respect to θ, with all other parameters held fixed. In total, we have S CM steps for each iteration.

In the previous Section 2.3, we have introduced copulas for data modelling with more flexibility. But, including simple Gaussian distribution, they are uni-modal distributions that there is only one peak (single highest value) in the probability density function. But in reality, more often the data that we are trying to model follows multi-modal distribution that there appears more than one peaks in the distribution.

Example 2.4.1 (Distribution of Exam Scores)

Suppose a teacher gives a test to his class. Some of the students studied very hard, while some of them didn't study at all. After marking the exam, the teacher creates a histogram of the test scores. It follows a multi-modal distribution with one peak around low scores for the students who studied a lot and another peak around high scores for the students who didn't study in the Figure 2.4.1. We can clearly see that there are two bell shapes in the histogram. The data can probably be modelled by two normal distributions with different means and variances. We refer to such a model as a **mixture of Gaussian**. In general, it is called **mixture model** if the data can be modelled in terms of a mixture of several components, where each component has a simple parametric form.

2.4.1 Mixture Model Formulation and Parameter Estimation

Formally, a general mixture model with K components for a random vector $\boldsymbol{X} = (X_1, ..., X_p)^T$ has a density given by

$$f(\boldsymbol{x} \mid \boldsymbol{\eta}) = \sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x} \mid \boldsymbol{\psi}_k) \tag{2.4.1}$$

with the mixture weights π_k satisfying $\pi_k \geq 0$ for all k and $\sum_{k=1}^{K} \pi_k = 1$. The probability density function $f_k(\boldsymbol{x} \mid \boldsymbol{\psi}_k)$ with the parameter $\boldsymbol{\psi}_k$ is called the k^{th} mixture component of the mixture density $f(\boldsymbol{x} \mid \boldsymbol{\eta})$, and π_k is called the mixing proportion. The parameters in a general mixture model are denoted as $\boldsymbol{\eta} = \{\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_K, \pi_1, ..., \pi_K\} \in \boldsymbol{\Theta}$ where $\boldsymbol{\Theta}$ denotes the parameter space.

Example 2.4.2 (Mixture of univariate Gaussian distributions)

The Figure 2.4.2 shows an example of a mixture using univariate Gaussian distribution with 3 components and its probability density function is in the following:

$$f(x \mid \boldsymbol{\eta}) = 0.2\ f_1(x \mid -2, 0.5) + 0.3\ f_2(x \mid 1, 2) + 0.5\ f_3(x \mid 4, 1)$$

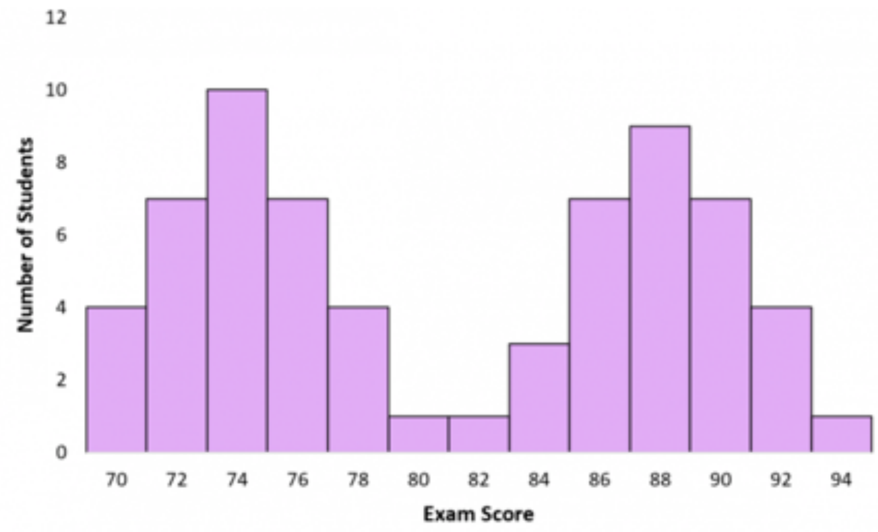


Figure 2.4.1: A histogram of test scores. It is clearly a multimodal distribution with two peaks, also called bi-modal distribution.

where $f_k(x \mid \mu_k, {\sigma_k}^2)$ denotes as a normal density with mean μ_k and variance ${\sigma_k}^2$ and is given by

$$f_k(x \mid \mu_k, {\sigma_k}^2) = \frac{1}{\sigma_k\sqrt{2\pi}} e^{-(x-\mu_k)^2/2\sigma_k^2} \quad \text{for} \quad k = 1, 2, 3.$$

Intuitively, for each sample, there is a probability with 0.2 generated from a Gaussian with mean -2 and variance 0.5, a probability with 0.3 generated from a Gaussian with mean 1 and variance 2 and a probability with 0.5 generated from a Gaussian with mean 4 and variance 1.

Assuming that $\mathcal{X} = [\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}] \in \mathbb{R}^{p\times n}$ is a data matrix with n columns i.i.d. observations generated from the mixture model given in Equation (2.4.1). We are going to estimate the parameters $\boldsymbol{\eta}$ by the observations $\boldsymbol{x}_i$ using maximum likelihood estimation. The likelihood function of the n observations is given by

$$L(\boldsymbol{\eta} \mid \boldsymbol{x}) := \prod_{i=1}^{n} f(\boldsymbol{x}_i \mid \boldsymbol{\eta}).$$

Then, the log-likelihood function can be expressed as

$$\begin{aligned} l(\boldsymbol{\eta} \mid \boldsymbol{x}) &:= \ln\left(\prod_{i=1}^{n} f(\boldsymbol{x}_i \mid \boldsymbol{\eta})\right) \\ &= \sum_{i=1}^{n} \ln\left(\sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)\right). \end{aligned}$$

The problem of parameter estimation for mixture model can be formulated as

$$\begin{aligned} \max_{\boldsymbol{\eta}\in\Theta} \quad & l(\boldsymbol{\eta} \mid \boldsymbol{x}) = \sum_{i=1}^{n} \ln\left(\sum_{k=1}^{K} \pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)\right) \\ \text{subject to} \quad & \pi_k \geq 0 \ \forall k \text{ and } \sum_{k=1}^{K} \pi_k = 1 \end{aligned} \tag{2.4.2}$$

The difficulty of solving this optimization problem is to maximize the summation inside the logarithm of objective function. Unfortunately, there is no analytical solution for this optimization problem (2.4.2) by simply taking the derivatives and setting to 0. However, if you don't want to solve it numerically, the good news is that we still may get a closed-form solution by using the EM algorithm.

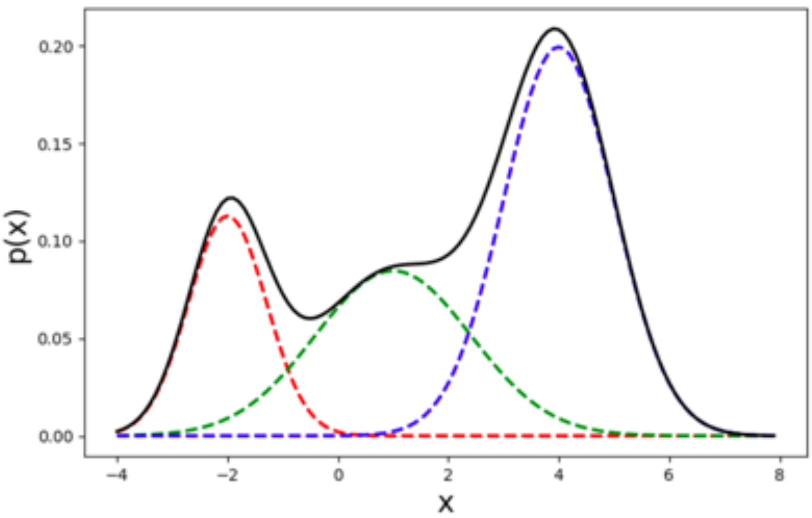


Figure 2.4.2: An example of a univariate mixture of Gaussian model with 3 components. red dashed line: density of $\mathcal{N}(-2, 0.5)$. Green dashed line: density of $\mathcal{N}(1, 2)$. Blue dashed line: density of $\mathcal{N}(4, 1)$. Black line: density of the Gaussian mixture.

The EM algorithm cannot be applied to solve the mixture model directly. On the contrary, EM algorithm handles this issue by introducing the latent variables. In mixture model, each sample just can be generated by 1 component, but we don't know which component. So, we regard the mixture components as the latent variable z, which is not observable.

Example 2.4.3 (Distribution of exam scores (continued))

In the Example 2.4.1, we know the test scores of all students. However, we cannot randomly pick a score and know whether it is from a hard-working student. So, the students who studied so hard and the students who didn't study at all are latent variables.

Suppose that the number of component K in the mixture model is known and define a random variable $\boldsymbol{Z} = (Z_1, ..., Z_k)^T \in \mathbb{R}^K$, that each $Z_k \in \{0, 1\}$ and only one of the element is equal to one and the others are zero. Intuitively, $\boldsymbol{Z}$ is somehow like a state variable to record which component is taken for that sample. For the only component we take, we set it to be 1, while others are set to be 0. The probability of choosing component k is the mixing proportion π_k. Therefore, Z follows a multinomial distribution over K category with the probability of the mixing proportion $\boldsymbol{\pi} = (\pi_1, ..., \pi_K)$ and exactly 1 trail, such that $\boldsymbol{Z} \sim Multinomial(\pi_1, ..., \pi_K)$. Recall that the probability mass function for multinomial random variable is

$$p(\boldsymbol{z}) = \frac{n!}{z_1! \cdots z_k!} {p_1}^{z_1} \cdots {p_k}^{z_k},$$

supported on $\boldsymbol{z} = (z_1,\, z_2,\, ...,\, z_k)$ where each z_i is a non-negative integer and their sum is n. In our case, n is 1 and $z_k! = 1\ \forall k$, where $z_k = \{0, 1\}$. Thus, we can write the probability $P(\boldsymbol{Z} = \boldsymbol{z}) = p(\boldsymbol{z})$ as

$$p(\boldsymbol{z}) = {\pi_1}^{z_1} \cdots {\pi_k}^{z_k} = \prod_{k=1}^{K} {\pi_k}^{z_k}.$$

And suppose that the conditional density of $\boldsymbol{X}$ at $\boldsymbol{x}$ given the latent vector $\boldsymbol{Z} = \boldsymbol{e}_k \in \mathbb{R}^k$ is $f_k(\boldsymbol{x} \mid \boldsymbol{\psi}_k)$ in Equation (2.4.1), such that

$$p(\boldsymbol{x} \mid \boldsymbol{Z} = \boldsymbol{e}_k) = f_k(\boldsymbol{x} \mid \boldsymbol{\psi}_k) \tag{2.4.3}$$

where $\boldsymbol{e}_k$ denotes the k^{th} column $K \times K$ identity matrix and Equation (2.4.3) can be also written as

$$p(\boldsymbol{x} \mid \boldsymbol{z}) = \prod_{k=1}^{K} f(\boldsymbol{x} \mid \boldsymbol{\psi}_k)^{z_k}.$$

Thus, the joint probability mass function of $\boldsymbol{X}$ and $\boldsymbol{Z}$ is

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z})p(\boldsymbol{x} \mid \boldsymbol{z}) = \prod_{k=1}^{K} \pi_k^{z_k} f(\boldsymbol{x} \mid \boldsymbol{\psi}_k)^{z_k}.$$

Example 2.4.4 (Mixture of univariate Gaussian distributions (continued))

In Example 2.4.2, the latent variables Z, the random samples X and their distributions can be expressed as in the following expression:

$$\begin{aligned}
\boldsymbol{Z} = (Z_1, Z_2, Z_3) \sim Multinomial(0.2, 0.3, 0.5) &\implies p(\boldsymbol{z}) = (0.2)^{z_1}(0.3)^{z_2}(0.5)^{z_3} \\
X \mid \boldsymbol{z} = (1,0,0) \sim \mathcal{N}(-2, 0.5) &\implies p(x \mid \boldsymbol{z} = (1,0,0)) = f_{\mathcal{N}}(x \mid -2, 0.5) \\
X \mid \boldsymbol{z} = (0,1,0) \sim \mathcal{N}(1, 2) &\implies p(x \mid \boldsymbol{z} = (0,1,0)) = f_{\mathcal{N}}(x \mid 1, 2) \\
X \mid \boldsymbol{z} = (0,0,1) \sim \mathcal{N}(4, 1) &\implies p(x \mid \boldsymbol{z} = (0,0,1)) = f_{\mathcal{N}}(x \mid 4, 1)
\end{aligned}$$

$$p(\boldsymbol{x} \mid \boldsymbol{z}) = (f_{\mathcal{N}}(x \mid -2, 0.5))^{z_1}(f_{\mathcal{N}}(x \mid -2, 0.5))^{z_2}(f_{\mathcal{N}}(x \mid -2, 0.5))^{z_3}.$$
$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z})p(\boldsymbol{x} \mid \boldsymbol{z}) = (0.2f_{\mathcal{N}}(x \mid -2, 0.5))^{z_1}(0.3f_{\mathcal{N}}(x \mid -2, 0.5))^{z_2}(0.5f_{\mathcal{N}}(x \mid -2, 0.5))^{z_3}$$

where the red, green and blue dashed lines in the Figure 2.4.2 represent component 1, 2 and 3 respectively.

After introduction of the latent variable $\boldsymbol{Z}$, we now have a new joint distribution for $\boldsymbol{X}_1, ..., \boldsymbol{X}_n$ and their latent variable vector $\boldsymbol{Z}_1, ..., \boldsymbol{Z}_n$. As the latent variable $\boldsymbol{Z}$ is unobservable, the complete data is $(\boldsymbol{z}, \boldsymbol{x})$ and the incomplete data is $\boldsymbol{x}$. In the EM algorithm, we want to maximize the log-likelihood function of the complete data. The likelihood of the complete data is given by

$$L(\boldsymbol{\eta} \mid \boldsymbol{x}, \boldsymbol{z}) := \prod_{i=1}^{n} p(\boldsymbol{x}_i, \boldsymbol{z}_i) = \prod_{i=1}^{n}\prod_{k=1}^{K} \pi_k^{z_{ik}} f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)^{z_{ik}},$$

where z_{ik} represents the k^{th} element of $\boldsymbol{z}_i = (z_1^i, ..., z_K^i)$. Again the latent variables are collected in a vector $\boldsymbol{z} = (\boldsymbol{z}_1^t, ..., \boldsymbol{z}_n^t)^t$.So that, the log-likelihood can be written as

$$l(\boldsymbol{\eta} \mid \boldsymbol{x}, \boldsymbol{z}) := \sum_{i=1}^{n} \ln(p(\boldsymbol{x}_i, \boldsymbol{z}_i)) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik} \ln(\pi_k f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)).$$

After getting the log-likelihood function, the following EM steps can be implemented:

E-step: Computes the expected value of $l(\boldsymbol{\eta} \mid \boldsymbol{x}, \boldsymbol{Z})$ given the observed data $\boldsymbol{x}$ and the current parameter estimate $\boldsymbol{\eta}_{old}$. So that,

$$\begin{aligned}
Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old}) &= E[l(\boldsymbol{\eta} \mid \boldsymbol{x}, \boldsymbol{Z}) \mid \boldsymbol{\eta}_{old}, \boldsymbol{x}] \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K} E[Z_{ik} \mid \boldsymbol{\eta}_{old}, \boldsymbol{x}] \ln(\pi_k f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)).
\end{aligned}$$

where $\boldsymbol{\eta}_{old} = \{\boldsymbol{\psi}_1, ..., \boldsymbol{\psi}_K, \pi_1, .., \pi_K\}$ is the parameter values at current iteration and $E[\ln(\pi_k f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)) \mid \boldsymbol{\eta}_{old}, \boldsymbol{x}] = \ln(\pi_k f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k))$ since $E[f(X) \mid X] = f(X)$. We denote the expectation of the latent variable given the samples as r_{ik} and we have

$$\begin{aligned} E[Z_{ik} \mid \boldsymbol{\eta}_{old}, \boldsymbol{x}] := r_{ik}^{old} &= 1\ p(Z_{ik} = 1 \mid \boldsymbol{x}_i) + 0\ p(Z_{ik} = 0 \mid \boldsymbol{x}_i) = p(Z_{ik} = 1 \mid \boldsymbol{x}_i) \\ &= p(\boldsymbol{Z}_i = \boldsymbol{e}_k \mid \boldsymbol{x}_i) \\ &= \frac{p(\boldsymbol{Z}_i = \boldsymbol{e}_k, \boldsymbol{x}_i)}{p(\boldsymbol{x}_i)} \\ &= \frac{p(\boldsymbol{Z}_i = \boldsymbol{e}_k) p(\boldsymbol{x}_i \mid \boldsymbol{Z}_i = \boldsymbol{e}_k)}{\sum_{j=1}^{K} p(\boldsymbol{Z} = \boldsymbol{e}_j) p(\boldsymbol{x}_i \mid \boldsymbol{Z}_i = \boldsymbol{e}_j)} \\ &= \frac{\pi_k^{old} f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k^{old})}{\sum_{j=1}^{K} \pi_j^{old} f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_j^{old})} \quad \text{for} \quad i = 1, ..., n \quad \text{and} \quad k = 1, ..., K \end{aligned} \tag{2.4.4}$$

where $\pi^{old}, \boldsymbol{\psi}^{old}$ are the parameters values at current iteration. Thus,

$$Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old}) = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}^{old} \ln(\pi_k f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)) \tag{2.4.5}$$

M-step: The updated parameters $\boldsymbol{\eta}_{new}$ can be obtained by maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\eta}$:

$$\begin{aligned} &\boldsymbol{\eta}_{new} := \arg\max_{\boldsymbol{\eta}} Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old}). \\ &\text{subject to} \quad \pi_k \geq 0\ \forall k \text{ and } \sum_{k=1}^{K} \pi_k = 1 \end{aligned} \tag{2.4.6}$$

In mixture model, the optimization problem is subject to some constraints with π. To get π_k, the maximization cannot be achieved simply by taking the derivatives with respect to the parameters π_k and then set to zero. Here, we use the method of Lagrange multipliers introduced in the Section 2.2 to solve this optimization problem with constrains.

For the optimization problem (2.4.6), the Lagrangian function is given by

$$\mathcal{L} = \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}^{old} \ln(\pi_k f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)) + \lambda(\sum_{k=1}^{K} \pi_k - 1)$$

Taking the derivatives with respect to π_k and then set to 0:

$$\frac{\partial \mathcal{L}}{\partial \pi_k} = \frac{\sum_{i=1}^{n} r_{ik}^{old}}{\pi_k} + \lambda = 0$$

π_k can be solved as

$$\pi_k = -\frac{\sum_{i=1}^{n} r_{ik}^{old}}{\lambda}.$$

Then, taking the derivatives with respect to λ and then set to 0:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \lambda} &= \sum_{k=1}^{K} \pi_k - 1 = 0 \\ \sum_{k=1}^{K} \pi_k &= 1 \end{aligned} \tag{2.4.7}$$

Plugging in $\pi_k = -\frac{\sum_{i=1}^{N} r_{ik}^{old}}{\lambda}$ in Equation (2.4.7):

$$\begin{aligned} \sum_{k=1}^{K} \left(-\frac{\sum_{i=1}^{N} r_{ik}^{old}}{\lambda} \right) &= 1 \\ -\sum_{k=1}^{K} \sum_{i=1}^{N} r_{ik}^{old} &= \lambda \end{aligned} \tag{2.4.8}$$

Plugging in the expression for r_{ik}^{old} in Equation (2.4.8):

$$
\begin{aligned}
&-\sum_{k=1}^{K}\sum_{i=1}^{n}\left(\frac{\pi_k^{old} f(\boldsymbol{x} \mid \boldsymbol{\theta}_k^{old})}{\sum_{j=1}^{K}\pi_j^{old} f(\boldsymbol{x} \mid \boldsymbol{\theta}_j^{old})}\right) = \lambda \\
\iff \quad & -\sum_{i=1}^{N} 1 = \lambda \\
\iff \quad & \lambda = -n
\end{aligned}
$$

Thus, we showed that updated mixing proportions are given by

$$
\pi_k = \frac{\sum_{i=1}^{n} r_{ik}^{old}}{n} \quad \text{for} \quad k = 1, ..., K \tag{2.4.9}
$$

About updating the remaining parameters, $\boldsymbol{\psi}_t^{new}$ is the root of the following equation by setting the values of the partial derivatives to zero.

$$
\sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}^{old} \, \frac{\partial \ln f(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k)}{\partial \boldsymbol{\psi}_t} = 0 \quad \text{for} \quad t = 1, ..., K \tag{2.4.10}
$$

A wonderful characteristic of the EM algorithm is that the solution of Equation (2.4.10) often exists in analytic. Gaussian mixture model is one of the example and it will be demonstrated in detail in the next Section 2.4.2.

Other than the EM or ECM algorithm, we have introduced two more extensions called MCECM and ECME in the Section 2.1.3. Because the steps for the extension are quite similar to the EM or ECM algorithm, in order to avoid content repetition, we will introduce them with the example of GMM in the next Section 2.4.2.

2.4.2 Multivariate Gaussian Mixture Models (GMM)

In this section, we are going to show you the parameter estimation for GMM by the EM algorithm and its extensions. We follow the approaches introduced in Section 2.3.1, since the GMM is just a special case of mixture models and specifying the probability density function $f(\boldsymbol{x} \mid \boldsymbol{\psi}_k)$ in Equation (2.4.1) as multivariate normal density at $\boldsymbol{x} \in \mathbb{R}^p$. As mentioned before, the analytic of GMM exists in the M-step also.

Recalling that the probability density function of multivariate normal distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^p$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{p \times p}$ is:

$$
f(\boldsymbol{x} \mid \boldsymbol{\psi}_k) = f_{\mathcal{N}}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \left(\frac{1}{(2\pi)^k |\boldsymbol{\Sigma}|}\right)^{\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right) \tag{2.4.11}
$$

Based on Equation (2.4.1), a Gaussian mixture model with K components can be written as

$$
p(\boldsymbol{x} \mid \boldsymbol{\eta}) = \sum_{k=1}^{K} \pi_k \; f_{\mathcal{N}}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)
$$

with the $\pi_k \geq 0$, $\forall k$ and $\sum_{k=1}^{K} \pi_k = 1$. The parameters in the Gaussian mixture model is collected in $\boldsymbol{\eta} = \{\pi_1, ..., \pi_K, \boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_K\} \in \boldsymbol{\Theta}$ where $\boldsymbol{\Theta}$ denotes the parameter space. Now, we can make use of the derived expressions in E and M steps in the previous section.

E-step: From Equation (2.4.5), we can get the expected value of $l(\boldsymbol{\eta} \mid \boldsymbol{x}, \boldsymbol{Z})$ given the observed data $\boldsymbol{x}$ and the current parameter estimate $\boldsymbol{\eta}_{old}$ denoted as $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$.

$$
Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old}) = \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}^{old} \ln(\pi_k f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))
$$

where the expectation of the latent variable given the samples denoted as r_{ik}^{old} is given by Equation (2.4.4) and we have

$$
r_{ik}^{old} = \frac{\pi_k^{old} f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k^{old}, \boldsymbol{\Sigma}_k^{old})}{\sum_{j=1}^{K} \pi_j^{old} f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j^{old}, \boldsymbol{\Sigma}_j^{old})} \quad \text{for} \;\; i = 1, ..., n \;\; \text{and} \;\; k = 1, ..., K \tag{2.4.12}
$$

M-step: The updated parameters $\boldsymbol{\eta}_{new}$ can be obtained by maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\eta}$, where $\boldsymbol{\eta} = \{\pi_1, ..., \pi_K, \boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_K\}$.

$$\begin{gathered}\boldsymbol{\eta}_{new} := \arg\max_{\boldsymbol{\eta}} \left(\sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}^{old} \ln(\pi_k f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)) \right). \\ \text{subject to} \quad \pi_k \geq 0 \; \forall k \text{ and } \sum_{k=1}^{K} \pi_k = 1\end{gathered} \tag{2.4.13}$$

The mixing proportion π_k is found in Equation (2.4.9) and it is

$$\pi_k = \frac{\sum_{i=1}^{n} r_{ik}^{old}}{n} \quad \text{for} \quad k = 1, ..., K \tag{2.4.14}$$

We can now update the remaining parameters $\boldsymbol{\mu}_j$, $j = 1, ..., K$ by taking the first derivatives of $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\mu}_j$ and set to zero. Then, we have

$$\begin{aligned}\sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}^{old} \frac{\partial \ln f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\mu}_j} &= 0 \iff \\ \sum_{i=1}^{n} r_{ij}^{old} \frac{1}{f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \frac{\partial f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\partial \boldsymbol{\mu}_j} &= 0 \iff \\ \sum_{i=1}^{n} r_{ij}^{old} \frac{f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \frac{\partial(-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_j))}{\partial \boldsymbol{\mu}_j} &= 0 \iff \\ \sum_{i=1}^{n} r_{ij}^{old} \frac{\partial((\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_j))}{\partial(\boldsymbol{x}_i - \boldsymbol{\mu}_j)} &= 0\end{aligned}$$

Since $\frac{\partial \boldsymbol{w}^T A \boldsymbol{w}}{\partial \boldsymbol{w}} = 2A\boldsymbol{w}$ if $\boldsymbol{w}$ doesn't depend on A and A is symmetric and $\boldsymbol{\Sigma}_k^{-1}$ is symmetric, we have

$$\begin{aligned}\sum_{i=1}^{n} r_{ij}^{old} 2\boldsymbol{\Sigma}_j^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_j) &= 0 \\ \sum_{i=1}^{n} r_{ij}^{old} (\boldsymbol{x}_i - \boldsymbol{\mu}_j) &= 0 \\ \sum_{i=1}^{n} r_{ij}^{old} \boldsymbol{x}_i - \sum_{i=1}^{n} r_{ij}^{old} \boldsymbol{\mu}_j &= 0\end{aligned}$$

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} r_{ij}^{old} \boldsymbol{x}_i}{\sum_{i=1}^{n} r_{ij}^{old}} \quad \text{for} \quad j = 1, ..., K \tag{2.4.15}$$

Similarly, for $\boldsymbol{\Sigma}_j$,

$$\begin{aligned}\sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}^{old} \frac{\partial \ln f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\Sigma}_j} &= 0 \\ \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}^{old} \frac{\partial}{\partial \boldsymbol{\Sigma}_j} \left(-\frac{1}{2} \ln |\boldsymbol{\Sigma}_k| - \frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \right) &= 0\end{aligned}$$

Since $\frac{\partial \ln |\boldsymbol{A}|}{\partial \boldsymbol{A}} = \boldsymbol{A}^{-1}$ if $\boldsymbol{A}$ is a symmetric matrix, $\boldsymbol{ABC} = tr(\boldsymbol{BCA})$ if $\boldsymbol{ABC}$ is a scalar and we know $(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)$ is a scalar, we have

$$\sum_{i=1}^{n} r_{ij}^{old} \boldsymbol{\Sigma}_j^{-1} + \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik}^{old} \frac{\partial}{\partial \boldsymbol{\Sigma}_j} \left(tr(\boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \right) = 0$$

Since $\frac{\partial tr(\boldsymbol{A}^{-1}\boldsymbol{x}\boldsymbol{x}^T)}{\partial \boldsymbol{A}} = -\boldsymbol{A}^{-1}\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{A}^{-1}$ if $\boldsymbol{A}$ is a symmetric matrix, we have

$$\sum_{i=1}^{n} r_{ij}^{old}\boldsymbol{\Sigma}_j^{-1} - \sum_{i=1}^{n} r_{ij}^{old}\left(\boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T\boldsymbol{\Sigma}_j^{-1}\right) = 0$$

$$\sum_{i=1}^{n} r_{ij}^{old} - \sum_{i=1}^{n} r_{ij}^{old}\boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T = 0$$

$$\boldsymbol{\Sigma}_j = \frac{\sum_{i=1}^{n} r_{ij}^{old}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T}{\sum_{i=1}^{n} r_{ij}^{old}} \quad \text{for} \quad j = 1, ..., K \tag{2.4.16}$$

For simplicity, the steps for the EM or ECM algorithm at a single iteration can be summarised as follows :

EM or ECM E step :

$$r_{ij} = \frac{\pi_j f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{J=1}^{K} \pi_J f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_J, \boldsymbol{\Sigma}_J)} \quad \text{for} \;\; i = 1, ..., n \;\; \text{and} \;\; j = 1, ..., K$$

EM or ECM CM step 1:

$$\pi_j = \frac{\sum_{i=1}^{n} r_{ij}}{n} \quad \text{for} \quad j = 1, ..., K$$

EM or ECM CM step 2:

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} r_{ij}\boldsymbol{x}_i}{\sum_{i=1}^{n} r_{ij}} \quad \text{for} \quad j = 1, ..., K$$

EM or ECM CM step 3:

$$\boldsymbol{\Sigma}_j = \frac{\sum_{i=1}^{n} r_{ij}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T}{\sum_{i=1}^{n} r_{ij}} \quad \text{for} \quad j = 1, ..., K$$

Now, we would like to show you the steps of the parameter estimation for GMM by the MCECM algorithm. Recalling that in MCECM, E step is performed before each CM step and a cycle is defined by one E step followed by one CM step, where each iteration involves S cycles. Because there are total 3 different parameters for each cluster in GMM, each iteration includes 3 cycles. Also, each cycle contains one E step and one M step. More precisely, the steps for the MCECM algorithm at a single iteration are defined as follows (Because of the high similarity, we can mostly make use of the expressions derived in the GMM algorithm for EM algorithm):

Cycle 1

MCECM E step 1: The E step of the MCECM algorithm is the same as the EM or ECM algorithm. From the Equation (2.4.12), the expectation of the latent variable r_{ij} is given by

$$r_{ij} = \frac{\pi_j f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{J=1}^{K} \pi_J f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_J, \boldsymbol{\Sigma}_J)} \quad \text{for} \;\; i = 1, ..., n \;\; \text{and} \;\; j = 1, ..., K$$

MCECM CM step 1: The CM step 1 of the MCECM algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to π_k for the optimization problem (2.4.13). From the Equation (2.4.14), the mixing proportion π_j is given by

$$\pi_j = \frac{\sum_{i=1}^{n} r_{ij}}{n} \quad \text{for} \quad j = 1, ..., K$$

Cycle 2

MCECM E step 2: This E step is exactly the same as MCECM E step 1. The only difference is that the mixing proportion π_j used in the calculation of the r_{ij} is updated in the CM step 1.

$$r_{ij} = \frac{\pi_j f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{J=1}^{K} \pi_J f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_J, \boldsymbol{\Sigma}_J)} \quad \text{for} \;\; i = 1, ..., n \;\; \text{and} \;\; j = 1, ..., K$$

MCECM CM step 2: Similarly, the CM step 2 of the MCECM algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to μ_j for the optimization problem (2.4.13). From the Equation (2.4.15), the mean μ_j is given by

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} r_{ij}\boldsymbol{x}_i}{\sum_{i=1}^{n} r_{ij}} \quad \text{for} \quad j = 1, ..., K$$

Cycle 3

MCECM E step 3: This E step is exactly the same as MCECM E step 2. Similarly, the only difference is that the mean $\boldsymbol{\mu}_j$ used in the calculation of the r_{ij} is updated in the CM step 2.

$$r_{ij} = \frac{\pi_j f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{J=1}^{K} \pi_J f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_J, \boldsymbol{\Sigma}_J)} \quad \text{for} \quad i = 1, ..., n \quad \text{and} \quad j = 1, ..., K$$

MCECM CM step 3: Similarly, the CM step 3 of the MCECM algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\Sigma}_j$ for the optimization problem (2.4.13). From the Equation (2.4.16), the mixing proportion $\boldsymbol{\Sigma}_j$ is given by

$$\boldsymbol{\Sigma}_j = \frac{\sum_{i=1}^{n} r_{ij}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T}{\sum_{i=1}^{n} r_{ij}} \quad \text{for} \quad j = 1, ..., K$$

Finally, the steps of the parameter estimation for GMM by the ECME algorithm will be introduced here. Recalling that the assumption for the ECME algorithm made in the Section 2.1.3, the difference between ECME and ECM algorithms is that in the last CM step in ECME, the actual log likelihood function $l(\boldsymbol{\eta} \mid \boldsymbol{x})$ from Equation (2.1.1) is maximised. More precisely, the steps for the ECME algorithm at a single iteration are defined as follows (Because of the high similarity, we can mostly make use of the expressions derived in the GMM algorithm for EM algorithm):

ECME E step : The E step of the ECME algorithm is the same as the EM or ECM algorithm. From the Equation (2.4.12), the expectation of the latent variable r_{ij} is given by

$$r_{ij} = \frac{\pi_j f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{J=1}^{K} \pi_J f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_J, \boldsymbol{\Sigma}_J)} \quad \text{for} \quad i = 1, ..., n \quad \text{and} \quad j = 1, ..., K$$

ECME CM step 1: The CM step 1 of the ECME algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to π_j for the optimization problem (2.4.13). From the Equation (2.4.14), the mixing proportion π_j is given by

$$\pi_j = \frac{\sum_{i=1}^{n} r_{ij}}{n} \quad \text{for} \quad j = 1, ..., K$$

ECME CM step 2: Similarly, the CM step 2 of the ECME algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to μ_j for the optimization problem (2.4.13). From the Equation (2.4.15), the mean μ_j is given by

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} r_{ij}\boldsymbol{x}_i}{\sum_{i=1}^{n} r_{ij}} \quad \text{for} \quad j = 1, ..., K$$

ECME CM step 3: This step is the only difference with EM or ECM algorithm, because the actual log likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{x})$ from Equation (2.1.1) is maximised with respect to $\boldsymbol{\Sigma}_j$ here, such that

$$\max_{\boldsymbol{\Sigma}_j} \quad l(\boldsymbol{\eta} \mid \boldsymbol{x}) = \sum_{i=1}^{n} \ln \left(\sum_{k=1}^{K} \pi_k f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \quad \text{for} \quad j = 1, ..., K$$

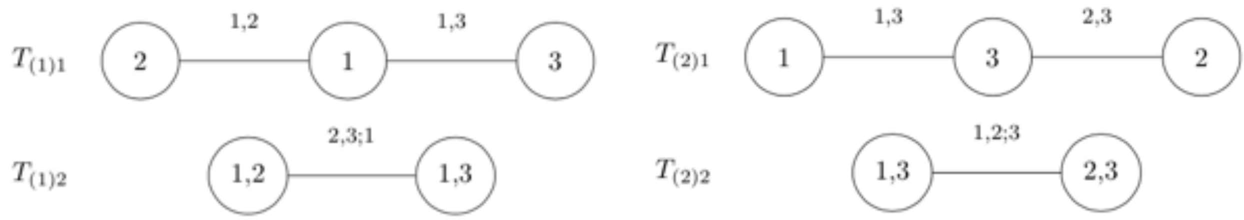
Because there is no analytic solution to maximise this actual log likelihood function, we have to get the solution numerically.

2.4.3 Vine Copula Mixture Models (VCMM)

In the Section 2.3, we have shown that copulas are more flexible distribution functions and therefore are more suitable to model complex data with clusters. In this section, we are going to extend the mixture model with the pair copula density and show you the parameter estimation for the vine copula mixture models (VCMM) by the EM algorithm and its extensions. For this section, we mainly follow Sahin and Czado [2021].

For simplicity, we would like to start with the vine copula mixture model formulation for 3 dimensions and 2 components as an example. From Equation (2.4.1), a mixture model with 2 components for a random vector $\boldsymbol{X} = (X_1, X_2, X_3)^T$ has a density given by

$$f(\boldsymbol{x} \mid \boldsymbol{\eta}) = \pi_1 f_1(\boldsymbol{x} \mid \boldsymbol{\psi}_1) + \pi_2 f_2(\boldsymbol{x} \mid \boldsymbol{\psi}_2) \tag{2.4.17}$$



(a) Component 1 (b) Component 2

Figure 2.4.3: Vine copula model of the density functions in the mixture model with two components

with the mixture weights π_1, π_2 satisfying $\pi_1, \pi_2 \geq 0$ and $\pi_1 + \pi_2 = 1$. The parameters in the mixture model are denoted as $\boldsymbol{\eta} = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \pi_1, \pi_2\}$.

An example taken from Sahin and Czado [2021], suppose that the 3 dimensional density function f_1, f_2 in the mixture model can be constructed by pair copulas and follow the vine copula model in the Figure 2.4.3.

According to the Definition 2.3.26, the pair copula construction of the density of component 1 f_1 with the corresponding copula parameter $\boldsymbol{\theta}$ and the marginal parameter $\boldsymbol{\gamma}$ is given as:

$$\begin{aligned} f_1(\boldsymbol{x} \mid \boldsymbol{\psi}_1) = {} & c_{(1)2,3;1}(F_{(1)2|1}(x_2|x_1; \gamma_{2(1)}, \gamma_{1(1)}, \boldsymbol{\theta}_{(1)1,2}), F_{(1)3|1}(x_3|x_1; \gamma_{3(1)}, \gamma_{1(1)}, \boldsymbol{\theta}_{(1)1,3}); \boldsymbol{\theta}_{(1)2,3;1}) \\ & \times c_{(1)1,2}(F_{1(1)}(x_1; \gamma_{1(1)}), F_{2(1)}(x_2; \gamma_{2(1)}); \boldsymbol{\theta}_{(1)1,2}) \\ & \times c_{(1)1,3}(F_{1(1)}(x_1; \gamma_{1(1)}), F_{2(1)}(x_3; \gamma_{3(1)}); \boldsymbol{\theta}_{(1)1,3}) \\ & \times f_{3(1)}(x_3; \gamma_{3(1)})\ f_{2(1)}(x_2; \gamma_{2(1)})\ f_{1(1)}(x_1; \gamma_{1(1)}) \end{aligned} \tag{2.4.18}$$

where the subscript (1) refers to the second component marker, $\boldsymbol{\theta}_{(1)1,2}, \boldsymbol{\theta}_{(1)1,3}, \boldsymbol{\theta}_{(1)2,3;1}$ are the parameters for the copula density $c_{(1)1,2}, c_{(1)1,3}, c_{(1)2,3;1}$ respectively, $\boldsymbol{\gamma}_{1(1)}, \boldsymbol{\gamma}_{2(1)}, \boldsymbol{\gamma}_{3(1)}$ are the parameters for the marginal density $f_{1(1)}, f_{2(1)}, f_{3(1)}$ and marginal distribution $F_{1(1)}, F_{2(1)}, F_{3(1)}$ respectively. Also, $\boldsymbol{\psi}_1 = (\boldsymbol{\gamma}_1, \boldsymbol{\theta}_1)$, where $\boldsymbol{\gamma}_1 = (\gamma_{1(1)}, \gamma_{2(1)}, \gamma_{3(1)})$ and $\boldsymbol{\theta}_1 = (\boldsymbol{\theta}_{(1)1,2}, \boldsymbol{\theta}_{(1)1,3}, \boldsymbol{\theta}_{(1)2,3;1})$.

Similarly, the pair copula construction of the density of component 2 f_2 with the corresponding copula parameter $\boldsymbol{\theta}$ and the marginal parameter $\boldsymbol{\gamma}$ is given as:

$$\begin{aligned} f_2(\boldsymbol{x} \mid \boldsymbol{\psi}_2) = {} & c_{(2)1,2;3}(F_{(2)1|3}(x_1|x_3; \gamma_{1(2)}, \gamma_{3(2)}, \boldsymbol{\theta}_{(2)1,3}), F_{(2)2|3}(x_2|x_3; \gamma_{2(2)}, \gamma_{3(2)}, \boldsymbol{\theta}_{(2)2,3}); \boldsymbol{\theta}_{(2)1,2;3}) \\ & \times c_{(2)2,3}(F_{2(2)}(x_2; \gamma_{2(2)}), F_{3(2)}(x_3; \gamma_{3(2)}); \boldsymbol{\theta}_{(2)2,3}) \\ & \times c_{(2)1,3}(F_{1(2)}(x_1; \gamma_{1(2)}), F_{3(2)}(x_3; \gamma_{3(2)}); \boldsymbol{\theta}_{(2)1,3}) \\ & \times f_{3(2)}(x_3; \gamma_{3(2)})\ f_{2(2)}(x_2; \gamma_{2(2)})\ f_{1(2)}(x_1; \gamma_{1(2)}) \end{aligned} \tag{2.4.19}$$

where the subscript (2) refers to the first component marker, $\boldsymbol{\theta}_{(2)1,3}, \boldsymbol{\theta}_{(2)2,3}, \boldsymbol{\theta}_{(2)1,2;3}$ are the parameters for the copula density $c_{(2)1,3}, c_{(2)2,3}, c_{(2)1,2;3}$ respectively, $\boldsymbol{\gamma}_{1(2)}, \boldsymbol{\gamma}_{2(2)}, \boldsymbol{\gamma}_{3(2)}$ are the parameters for the marginal density $f_{1(2)}, f_{2(2)}, f_{3(2)}$ and marginal distribution $F_{1(2)}, F_{2(2)}, F_{3(2)}$ respectively. Also, $\boldsymbol{\psi}_2 = (\boldsymbol{\gamma}_2, \boldsymbol{\theta}_2)$, where $\boldsymbol{\gamma}_2 = (\gamma_{1(2)}, \gamma_{2(2)}, \gamma_{3(2)})$ and $\boldsymbol{\theta}_2 = (\boldsymbol{\theta}_{(2)1,3}, \boldsymbol{\theta}_{(2)2,3}, \boldsymbol{\theta}_{(2)1,2;3})$. Note that although we call $\boldsymbol{\theta}$ as copula parameter, we need this to determine conditional distribution function $F_{a|b}$, so the copula parameter is one of the parameters in $F_{a|b}$.

For more details about the formulation for the general vine copula mixture model, please refer to Sahin and Czado [2021].

Now, we need to use EM algorithm and its extension to estimate the following parameters: mixture weights π, copula parameter $\boldsymbol{\theta}$ and marginal parameter $\boldsymbol{\gamma}$ for two components. We start with ECM algorithm and make use of the derived expressions in E and M steps in the previous section.

EM or ECM E-step: From Equation (2.4.5), we can get the expected value of $l(\boldsymbol{\eta} \mid \boldsymbol{x}, \boldsymbol{Z})$, denoted as $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$, given the observed data $\boldsymbol{x}$ and the current parameter estimate $\boldsymbol{\eta}_{old}$.

$$Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old}) = \sum_{i=1}^{n}\sum_{k=1}^{2} r_{ik}^{old} \ln(\pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k))$$

where the expectation of the latent variable given the samples denoted as r_{ik}^{old} is given by Equation (2.4.4) and we have

$$r_{ik}^{old} = \frac{\pi_k^{old} f_k(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k^{old})}{\sum_{j=1}^{2} \pi_j^{old} f_j(\boldsymbol{x}_i \mid \boldsymbol{\psi}_j^{old})} \quad \text{for} \quad i = 1, ..., n \quad \text{and} \quad k = 1, 2 \tag{2.4.20}$$

For the M step, the updated parameters $\boldsymbol{\eta}_{new}$ can be obtained by maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\eta}$, where $\boldsymbol{\eta} = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \pi_1, \pi_2\}$, $\boldsymbol{\psi}_1 = \{\boldsymbol{\gamma}_1, \boldsymbol{\theta}_1\}$ and $\boldsymbol{\psi}_2 = \{\boldsymbol{\gamma}_2, \boldsymbol{\theta}_2\}$

$$\begin{gathered}\boldsymbol{\eta}_{new} := \arg\max_{\boldsymbol{\eta}} \left(\sum_{i=1}^{n}\sum_{k=1}^{2} r_{ik}^{old} \ln(\pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k) \right). \\ \text{subject to } \pi_1, \pi_2 \geq 0 \text{ and } \pi_1 + \pi_2 = 1\end{gathered} \tag{2.4.21}$$

EM or ECM CM-step 1 (Mixture weights):
The mixing proportion π_k is found in Equation (2.4.9) and it is

$$\pi_k = \frac{\sum_{i=1}^{n} r_{ik}^{old}}{n} \quad \text{for} \quad k = 1, 2 \tag{2.4.22}$$

EM or ECM CM-step 2 (Pair copula parameters): We now need to update the copula parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ of the 2 components by maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ respectively.

$$\boldsymbol{\theta}_j := \arg\max_{\boldsymbol{\theta}_j} \left(\sum_{i=1}^{n}\sum_{k=1}^{2} r_{ik}^{old} \ln(\pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\gamma}_k, \boldsymbol{\theta}_k) \right) \text{ for } j = 1, 2 \tag{2.4.23}$$

However, there is no analytic solution for the optimization problem (2.4.23). For this optimization, we can use the R function `RVineSeqMLE` in the R package `VineCopula` to estimate the pair-copula parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ by using maximum likelihood estimation (MLE) with weights r_{ik}^{old} (compare to Equation (2.4.23).

EM or ECM CM-step 3 (Marginal parameters): We now want to update the marginal parameters $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2$ of the 2 components by maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2$ respectively.

$$\boldsymbol{\gamma}_j := \arg\max_{\boldsymbol{\gamma}_j} \left(\sum_{i=1}^{n}\sum_{k=1}^{2} r_{ik}^{old} \ln(\pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\gamma}_k, \boldsymbol{\theta}_k) \right) \text{ for } j = 1, 2 \tag{2.4.24}$$

Because there is no analytic solution for the optimization problem (2.4.24), we have to get the solution numerically.

Now, we would like to show you the steps of the parameter estimation for VCMM by the MCECM algorithm. Recalling that in MCECM, E step is performed before each CM step and a cycle is defined by one E step followed by one CM step.

<u>**Cycle 1**</u>

MCECM E step 1: The E step of the MCECM algorithm is the same as the EM or ECM algorithm. From the Equation (2.4.20), the expectation of the latent variable r_{ij} is given by

$$r_{ik} = \frac{\pi_k^{old} f_k(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k^{old})}{\sum_{j=1}^{2} \pi_j^{old} f_j(\boldsymbol{x}_i \mid \boldsymbol{\psi}_j^{old})} \quad \text{for} \quad i = 1, ..., n \quad \text{and} \quad k = 1, 2$$

MCECM CM step 1 (Mixture weights): The CM step 1 of the MCECM algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to π_k for the optimization problem (2.4.21). From the Equation (2.4.22), the mixing proportion π_j is given by

$$\pi_j = \frac{\sum_{i=1}^{n} r_{ij}}{n} \quad \text{for} \quad j = 1, 2$$

Cycle 2

MCECM E step 2: This E step is exactly the same as MCECM E step 1. The only difference is that the mixing proportion π_j used in the calculation of the r_{ij} is updated in the CM step 1.

$$r_{ij} = \frac{\pi_j f_j(\boldsymbol{x}_i \mid \boldsymbol{\psi}_j)}{\sum_{J=1}^{2} \pi_J f_J(\boldsymbol{x}_i \mid \boldsymbol{\psi}_J)} \quad \text{for} \ \ i = 1, ..., n \ \ \text{and} \ \ j = 1, 2$$

MCECM CM step 2 (Pair copula parameters): Similarly, the CM step 2 of the MCECM algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ for the optimization problem (2.4.21). From the Equation (2.4.23), the copula parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ are given by

$$\boldsymbol{\theta}_j := \arg\max_{\boldsymbol{\theta}_j} \left(\sum_{i=1}^{n} \sum_{k=1}^{2} r_{ik} \ln(\pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\gamma}_k, \boldsymbol{\theta}_k) \right) \ \text{for} \ j = 1, 2 \tag{2.4.25}$$

Again, there is no analytic solution for the optimization problem (2.4.25). We again use the R function `RVineSeqMLE` with specified weights in the R package `VineCopula` to estimate the pair-copula parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$.

Cycle 3

MCECM E step 3: Similarly, the E step is exactly the same as MCECM E step 2. The only difference is that the copula parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ used in the calculation of the r_{ij} is updated in the CM step 2.

$$r_{ij} = \frac{\pi_j f_j(\boldsymbol{x}_i \mid \boldsymbol{\psi}_j)}{\sum_{J=1}^{2} \pi_J f_J(\boldsymbol{x}_i \mid \boldsymbol{\psi}_J)} \quad \text{for} \ \ i = 1, ..., n \ \ \text{and} \ \ j = 1, 2$$

where $\boldsymbol{\psi}_1 = \{\boldsymbol{\gamma}_1, \boldsymbol{\theta}_1\}$ and $\boldsymbol{\psi}_2 = \{\boldsymbol{\gamma}_2, \boldsymbol{\theta}_2\}$.

MCECM CM step 3 (Marginal parameters): Similarly, the CM step 3 of the MCECM algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2$ for the optimization problem (2.4.21). From the Equation (2.4.24), the marginal parameters $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2$ are given by

$$\boldsymbol{\gamma}_j := \arg\max_{\boldsymbol{\gamma}_j} \left(\sum_{i=1}^{n} \sum_{k=1}^{2} r_{ik} \ln(\pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\gamma}_k, \boldsymbol{\theta}_k) \right) \ \text{for} \ j = 1, 2 \tag{2.4.26}$$

Because there is no analytic solution for the optimization problem (2.4.26), we have to get the solution numerically.

Finally, the steps of the parameter estimation for VCMM by the ECME algorithm will be introduced here. Recalling that the assumption for the ECME algorithm made in the Section 2.1.3, the difference between ECME and ECM algorithms is that in the last CM step in ECME, the actual log likelihood function $l(\boldsymbol{\theta} \mid \boldsymbol{x})$ from Equation (2.1.1) is maximised.

ECME E step : The E step of the ECME algorithm is the same as the EM or ECM algorithm. From the Equation (2.4.20), the expectation of the latent variable r_{ij} is given by

$$r_{ik} = \frac{\pi_k^{old} f_k(\boldsymbol{x}_i \mid \boldsymbol{\psi}_k^{old})}{\sum_{j=1}^{2} \pi_j^{old} f_j(\boldsymbol{x}_i \mid \boldsymbol{\psi}_j^{old})} \quad \text{for} \ \ i = 1, ..., n \ \ \text{and} \ \ k = 1, 2$$

ECME CM step 1 (Mixture weights): The CM step 1 of the ECME algorithm is the same as maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to π_j for the optimization problem (2.4.21). From the Equation (2.4.22), the mixing proportion π_j is given by

$$\pi_j = \frac{\sum_{i=1}^{n} r_{ij}}{n} \quad \text{for} \quad j = 1, 2$$

ECME CM step 2 (Pair copula parameters): Similarly, because there is no analytic solution for the optimization problem (2.4.25), we prefer using the R function `RVineSeqMLE` to estimate the pair-copula parameters. Theoretically, we want to update the copula parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ of the 2 components by maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$ with respect to $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ respectively.

$$\boldsymbol{\theta}_j := \arg\max_{\boldsymbol{\theta}_j} \left(\sum_{i=1}^{n} \sum_{k=1}^{2} r_{ik} \ln(\pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\gamma}_k, \boldsymbol{\theta}_k) \right) \text{ for } j = 1, 2$$

ECME CM step 3 (Marginal parameters): In the last CM step of the ECME, instead of maximizing $Q(\boldsymbol{\eta} \mid \boldsymbol{\eta}_{old})$, we here maximise the actual log likelihood function $l(\boldsymbol{\eta} \mid \boldsymbol{x})$ from Equation (2.1.1) with respect to the marginal parameters $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2$, such that

$$\boldsymbol{\gamma}_j := \max_{\boldsymbol{\gamma}_j} \quad l(\boldsymbol{\eta} \mid \boldsymbol{x}) = \sum_{i=1}^{n} \ln \left(\sum_{k=1}^{2} \pi_k f_k(\boldsymbol{x}_i \mid \boldsymbol{\gamma}_k, \boldsymbol{\theta}_k) \right) \quad \text{for} \quad j = 1, 2$$

Because there is no analytic solution to maximise this actual log likelihood function $l(\boldsymbol{\eta} \mid \boldsymbol{x})$, we have to get the solution numerically.

2.5 Clustering using Gaussian Mixture Models

In this section, we would like to investigate how the parameters of the Multivariate Gaussian Mixture Model (GMM) determine the shape of generated samples in each cluster in 3 dimensional space, especially the covariance matrix $\boldsymbol{\Sigma}$. After that, the characteristics of Gaussian mixture models used in the R package `mclust` for model-based clustering will be introduced.

2.5.1 Characteristics of clusters in model based Gaussian mixture modelling

In the Section 2.4.2, we know that the parameters of the GMM contain the mixture weight $\boldsymbol{\pi}$, mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The mixture weight $\boldsymbol{\pi}$ decides the expected number of samples in each cluster, mean $\boldsymbol{\mu}$ decides the location of centre of each cluster, where Gaussian distribution is symmetric about the mean, and covariance matrix $\boldsymbol{\Sigma}$ decides the level of dispersion or shape of the samples. Banfield and Raftery [1993] have considered a parametrization of the covariance matrix in terms of its eigenvalue decomposition based on the distribution, volume, shape and orientation of each component in the mixture model :

$$\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A}_k {\boldsymbol{D}_k}^T, \tag{2.5.1}$$

where scalar λ_k defines the volume, rotation matrix $\boldsymbol{D}_k$ is an orthogonal matrix which defines its orientation and shape matrix $\boldsymbol{A}_k$ is a diagonal matrix with determinant 1 which defines its shape. In the following, we will illustrate how the values of each parameter affect the shape of the generated samples.

Scalar λ_k

The value of λ_k decides the volume of each cluster. The larger the value is, the bigger the cluster is. The reason behind is quite simple. If we consider λ_1 is 1 and $\boldsymbol{D}_k \boldsymbol{A}_k {\boldsymbol{D}_k}^T$ is an identity matrix $\boldsymbol{I}_3$, the covariance matrix $\boldsymbol{\Sigma_1}$ results in an identity matrix $\boldsymbol{I}_3$. Therefore, the variance of the generated samples in each coordinate axis is the same with value 1. Because of the same dispersion level of the generated samples in each coordinate axis, this results in a ball shape in the Figure 2.5.1. If we choose the value of λ_2 as 3, the covariance matrix $\boldsymbol{\Sigma_2}$ results in $3\boldsymbol{I}_3$ and the variance of the generated samples in each coordinate axis is larger. The generated sample in cluster 2 has higher level of dispersion, so that the volume of cluster 2 is larger. The Figure 2.5.2 with a larger ball, compared to the Figure 2.5.1, proofs our statement.

Shape matrix $\boldsymbol{A}_k$

As mentioned before, shape matrix $\boldsymbol{A}_k$ is a diagonal matrix with determinant 1 deciding its shape. In order to illustrate the shapes with different values of $\boldsymbol{A}_k$, suppose that the scalar $\lambda_3 = 1$, rotation matrix is $\boldsymbol{D}_3 = \boldsymbol{I}_3$

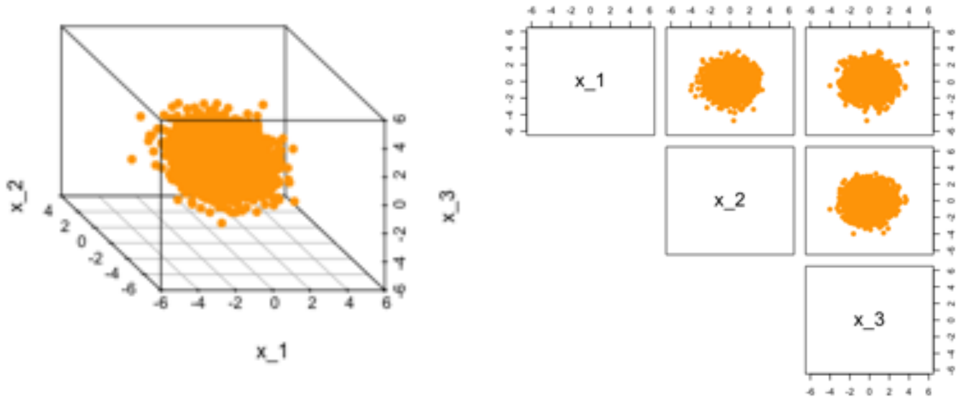


Figure 2.5.1: The scatter plot and pairs plot of generated data from multivariate Gaussian distribution with mean $\boldsymbol{\mu}_1 = [0, 0, 0]^T$, covariance matrix $\boldsymbol{\Sigma}_1 = \boldsymbol{I}_3$

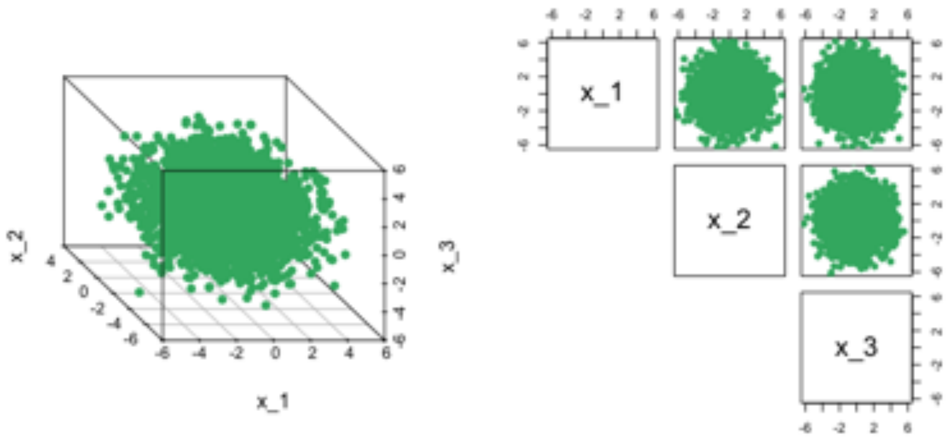


Figure 2.5.2: The scatter plot and pairs plot of generated data from multivariate Gaussian distribution with mean $\boldsymbol{\mu}_2 = [0, 0, 0]^T$, covariance matrix $\boldsymbol{\Sigma}_2 = 3\boldsymbol{I}_3$

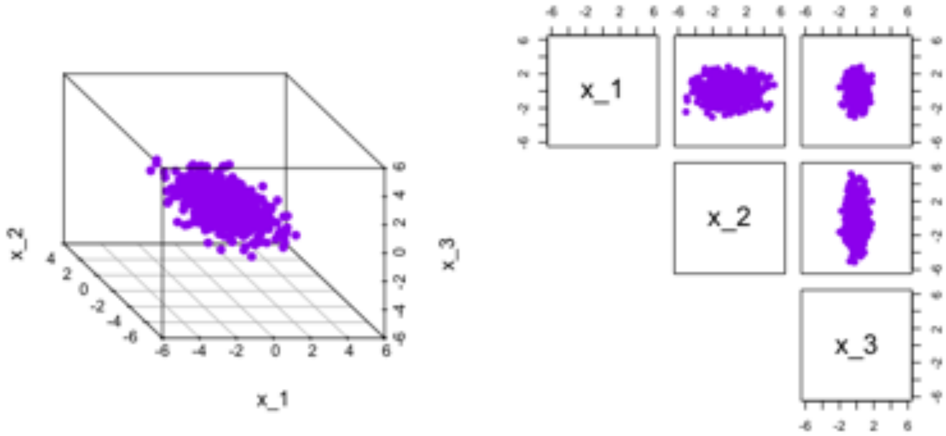


Figure 2.5.3: The scatter plot and pairs plot of generated data from multivariate Gaussian distribution with mean $\boldsymbol{\mu}_3 = [0,0,0]^T$, covariance matrix $\boldsymbol{\Sigma}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix}$

and shape matrices $\boldsymbol{A}_3$ is $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix}$ respectively. Therefore, $\boldsymbol{\Sigma}_3$ is the same as $\boldsymbol{A}_3$. From $\boldsymbol{\Sigma}_3$, we can know that the variance of the generated samples is different for different coordinate axes. The higher the variance, the more the spread of the data. Because the variance for x_1, x_2 and x_3 axis are 1, 3 and $\frac{1}{3}$ respectively, we can expect that compared to the Figure 2.5.1, the generated samples spread the same level in x_1 axis, more in the x_2 axis and less in x_3 axis. In other words, the shape of the generated data along x_2 axis is longer, along x_3 axis is shorter, so the shape becomes more elliptical, compared to the ball shape in the Figure 2.5.1. The Figure 2.5.3 proofs our statement.

Rotation matrix $\boldsymbol{D}_k$

The rotation matrix $\boldsymbol{D}_k$ determines the orientation of the corresponding shape. Different dimensional spaces have their own rotation matrix. Because this paper mainly focuses on 3 dimensional space, just the 3 dimensional rotation matrix will be introduced. Each 3D rotation matrix $\boldsymbol{D}_k$ can be decomposed further as

$$\begin{aligned} \boldsymbol{D}_k &= \boldsymbol{D}_k{}^z(\alpha)\boldsymbol{D}_k{}^y(\beta)\boldsymbol{D}_k{}^x(\gamma) \\ &= \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}. \end{aligned}$$

where α, β and γ represent yaw, pitch, and roll angles following the Figure 2.5.4 [Ellis et al., 2014]. Also, 0 degree means no orientation changes, because $\boldsymbol{D}_k{}^z(0), \boldsymbol{D}_k{}^y(0), \boldsymbol{D}_k{}^x(0)$ are just identity matrices. Positive degrees mean the object rotates along the arrow and negative degrees mean the object rotates along the opposite direction of the arrow.

Suppose that the scalar $\lambda_4 = 1$, shape matrix $\boldsymbol{A}_4$ is $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix}$ and the rotation matrix $\boldsymbol{D}_4$ is $\boldsymbol{D}_4{}^z(45°)\boldsymbol{D}_4{}^y(0)\boldsymbol{D}_4{}^x(0)$. Compared to the Figure 2.5.3, we can expect that the volume and the shape are exactly the same, because the scalar λ_4 and the shape matrix $\boldsymbol{A}_4$ are the same as λ_3 and $\boldsymbol{A}_3$ respectively. However, the orientation is different because the rotation matrix $\boldsymbol{D}_4$ is $\boldsymbol{D}_4{}^z(45°)$ and not an identity matrix. So, it is expected that the generated samples will rotate about the yaw axis by 45 degrees anticlockwise, which is shown in the Figure 2.5.5.

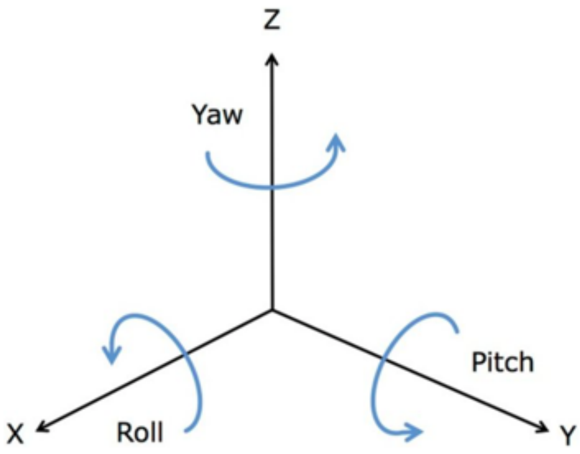


Figure 2.5.4: The position of all three axes for describing the angle of its rotations

Table 2.5.1: Table : The characteristics of the clusters in different models defined in the mclust package

Model	Expression	Distribution	Volume	Shape	Orientation
EII	$\lambda \boldsymbol{I}$	Spherical	Equal	Equal	-
VII	$\lambda_k \boldsymbol{I}$	Spherical	Variable	Equal	-
EEI	$\lambda \boldsymbol{A}$	Diagonal	Equal	Equal	Coordinate axes
VEI	$\lambda_k \boldsymbol{A}$	Diagonal	Variable	Equal	Coordinate axes
EVI	$\lambda \boldsymbol{A}_k$	Diagonal	Equal	Variable	Coordinate axes
VVI	$\lambda_k \boldsymbol{A}_k$	Diagonal	Variable	Variable	Coordinate axes
EEE	$\lambda \boldsymbol{DAD}^T$	Ellipsoidal	Equal	Equal	Equal
EVE	$\lambda \boldsymbol{D}\boldsymbol{A}_k\boldsymbol{D}^T$	Ellipsoidal	Equal	Variable	Equal
VEE	$\lambda_k \boldsymbol{DAD}^T$	Ellipsoidal	Variable	Equal	Equal
VVE	$\lambda_k \boldsymbol{D}\boldsymbol{A}_k\boldsymbol{D}^T$	Ellipsoidal	Variable	Variable	Equal
EEV	$\lambda \boldsymbol{D}_k\boldsymbol{A}\boldsymbol{D}_k{}^T$	Ellipsoidal	Equal	Equal	Variable
VEV	$\lambda_k \boldsymbol{D}_k\boldsymbol{A}\boldsymbol{D}_k{}^T$	Ellipsoidal	Variable	Equal	Variable
EVV	$\lambda \boldsymbol{D}_k\boldsymbol{A}_k\boldsymbol{D}_k{}^T$	Ellipsoidal	Equal	Variable	Variable
VVV	$\lambda_k \boldsymbol{D}_k\boldsymbol{A}_k\boldsymbol{D}_k{}^T$	Ellipsoidal	Variable	Variable	Variable

After introduction of the relationship between the shape of the generated samples and the parameters in the decomposition of the covariance matrix, we would like to introduce some models describing the difference of clusters in Gaussian mixture model. In the Table 2.5.1 [Scrucca et al., 2016], this shows you different characteristics of the clusters in different models, which is used in the R package `mclust` for model-based clustering. Because just models EEV, VEV, EVV and VVV will be used in the simulation, we just focus on these 4 models. About the name of the Model, E means "Equal" and U means "Unequal" or "Variable". Also, the first, second and the last alphabets represent Volume λ_k, Shape $\boldsymbol{A}_k$ and Orientation $\boldsymbol{D}_k$ respectively. Suppose that we have a Gaussian mixture model EEV with 2 clusters. Those 2 clusters have the same volume and shape, but different orientation. Therefore, the covariance matrix for cluster 1 and 2 can be represented by $\boldsymbol{\Sigma}_1 = \lambda \boldsymbol{D}_1\boldsymbol{A}\boldsymbol{D}_1{}^T$ and $\boldsymbol{\Sigma}_2 = \lambda \boldsymbol{D}_2\boldsymbol{A}\boldsymbol{D}_2{}^T$ respectively.

Example 2.5.1 (Visualization of simulated samples from the Gaussian Mixture Model EEV, VEV, EVV and VVV)

In the Figure 2.5.6, scatter plots of the 1000 generated data for EEV, VEV, EVV and VVV of Gaussian mixture

Table 2.5.2: The value of the parameters used in the visualization of simulated samples shown in the Figure 2.5.6.

		Gaussian mixture model type			
		(a) EEV	(b) VEV	(c) EVV	(d) VVV
$\boldsymbol{\Sigma}_1$	Covariance matrix of component 1	$\lambda \boldsymbol{D}_1 \boldsymbol{A} {\boldsymbol{D}_1}^T$	$\lambda_1 \boldsymbol{D}_1 \boldsymbol{A} {\boldsymbol{D}_1}^T$	$\lambda \boldsymbol{D}_1 \boldsymbol{A}_1 {\boldsymbol{D}_1}^T$	$\lambda_1 \boldsymbol{D}_1 \boldsymbol{A}_1 {\boldsymbol{D}_1}^T$
$\boldsymbol{\Sigma}_2$	Covariance matrix of component 2	$\lambda \boldsymbol{D}_2 \boldsymbol{A} {\boldsymbol{D}_2}^T$	$\lambda_2 \boldsymbol{D}_2 \boldsymbol{A} {\boldsymbol{D}_2}^T$	$\lambda \boldsymbol{D}_2 \boldsymbol{A}_2 {\boldsymbol{D}_2}^T$	$\lambda_2 \boldsymbol{D}_2 \boldsymbol{A}_2 {\boldsymbol{D}_2}^T$
$\boldsymbol{\Sigma}_2$	Covariance matrix of component 3	$\lambda \boldsymbol{D}_3 \boldsymbol{A} {\boldsymbol{D}_3}^T$	$\lambda_3 \boldsymbol{D}_3 \boldsymbol{A} {\boldsymbol{D}_3}^T$	$\lambda \boldsymbol{D}_3 \boldsymbol{A}_3 {\boldsymbol{D}_3}^T$	$\lambda_3 \boldsymbol{D}_3 \boldsymbol{A}_3 {\boldsymbol{D}_3}^T$
λ, λ_1	Scalar, Scalar of component 1	1	0.5	1	0.5
λ_2	Scalar of component 2	-	2	-	2
λ_3	Scalar of component 3	-	1	-	1
$\boldsymbol{D}_1$	Rotation matrix of component 1	$\alpha = 0°, \beta = 0°, \gamma = 0°$			
$\boldsymbol{D}_2$	Rotation matrix of component 2	$\alpha = 45°, \beta = 0°, \gamma = 0°$			
$\boldsymbol{D}_3$	Rotation matrix of component 3	$\alpha = -45°, \beta = 0°, \gamma = 0°$			
$\boldsymbol{A}, \boldsymbol{A}_1$	Shape matrix of component 1	$\begin{bmatrix} 4 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 4 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 4 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 4 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\boldsymbol{A}_2$	Shape matrix of component 2	-	-	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$\boldsymbol{A}_3$	Shape matrix of component 3	-	-	$\begin{bmatrix} 10 & 0 & 0 \\ 0 & \frac{1}{10} & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 10 & 0 & 0 \\ 0 & \frac{1}{10} & 0 \\ 0 & 0 & 1 \end{bmatrix}$

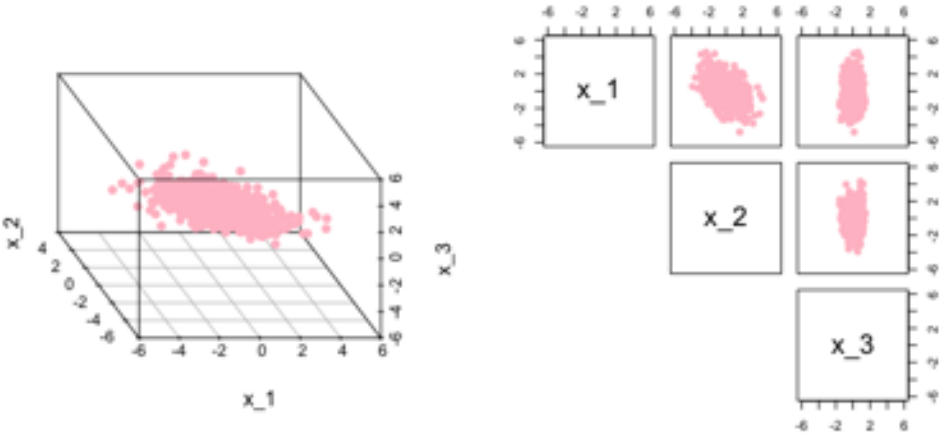
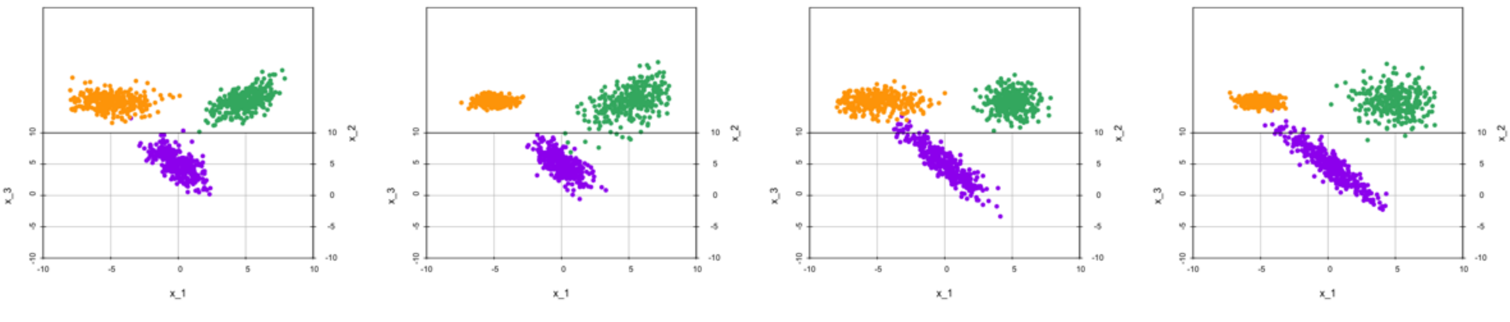


Figure 2.5.5: The scatter plot and pairs plot of generated data from multivariate Gaussian distribution with mean $\boldsymbol{\mu}_4 = [0, 0, 0]^T$, scalar $\lambda_4 = 1$, shape matrix $\boldsymbol{A}_4 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & \frac{1}{3} \end{bmatrix}$ and rotation matrix $\boldsymbol{D}_4 = \boldsymbol{D}_4{}^z(45°)$

models are shown and the corresponding parameters used are based on the Table 2.5.2. Also, orange, green and purple cluster represent the Custer 1,2 and 3 respectively. Figure 2.5.6 (a) EEV shows that the volume and shape of those 3 clusters are quite similar, just the orientations are different. In the Figure 2.5.6 (b) VEV, the shape of those clusters basically are the same as EEV, but the volumes are different. In the Figure 2.5.6 (c) EVV, the green cluster becomes a green ball and the purple cluster becomes more elliptical. Therefore, the shapes for those clusters are different. In the Figure 2.5.6 (d) VVV, the shapes are basically the same as EVV, but the volumes are different.



(a) EEV (b) VEV (c) EVV (d) VVV

Figure 2.5.6: Scatter plot of the 1000 generated data from the Clustering Gaussian mixture models (EEV, EVV, VEV and VVV) in case of three groups in three dimensions assuming the specifications given in the Table 2.5.2

2.5.2 Issues of singularity and clusters collapse in Gaussian mixture modelling for ECME

After the introduction of how the covariance matrix in the Multivariate Gaussian Mixture Model (GMM) determine the shape of generated samples, in this section, we will try to investigate singularity and cluster collapse issues that leads to error or spurious solutions (non-interesting solutions) after implementation of the

ECME algorithm involving the maximization of the actual log-likelihood function. Firstly, we will introduce the geometric meaning of eigenvalues of covariance matrix. After that, singularity and cluster collapse issues will be introduced. Finally, a solution to avoid these issues will be given.

Geometric interpretation of the eigenvalues of covariance matrix

In the last Section 2.5.1, we have mentioned that the shape of the cluster depends fully on the shape matrix $\boldsymbol{A}$ which is a diagonal matrix with determinant 1. Also, the values on the diagonal determine the variances of the generated samples in different coordinate axes. If shape matrix $\boldsymbol{A}$ is an identity matrix $\boldsymbol{I}_3$, we can expect that the shape of the generated samples is a sphere. If the shape matrix $\boldsymbol{A}$ is something else, we can expect that it is like a ellipse or even a French loaf.

Other than the shape matrix $\boldsymbol{A}$, there is another value showing the shape of the samples, which is eigenvalue. Here is the definition of the eigenvalue : Let $\boldsymbol{B}$ be a square matrix (or linear transformation). A number b is called an eigenvalue of $\boldsymbol{B}$ if there exists a non-zero vector $\boldsymbol{u}$ such that

$$\boldsymbol{B}\boldsymbol{u} = b\boldsymbol{u} \tag{2.5.2}$$

where the vector $\boldsymbol{u}$ is called an eigenvector associated with this eigenvalue b. Now, we would like to find out the eigenvalue a of a shape matrix $\boldsymbol{A}$. Equation (2.5.2) can be written as

$$(\boldsymbol{A} - b\boldsymbol{I})\boldsymbol{u} = \boldsymbol{A}\boldsymbol{u} - b\boldsymbol{u} = \boldsymbol{0}, \tag{2.5.3}$$

and we know that b is an eigenvalue b when Equation (2.5.3) has a non-trivial solution. Therefore, Equation (2.5.3) has a non-trivial solution when

$$\det(\boldsymbol{A} - b\boldsymbol{I}) = 0.$$

In our case, as shape matrix A is a three dimensional diagonal matrix, we have

$$(a_{11} - b)(a_{22} - b)(a_{33} - b) = 0.$$

where a_{11}, a_{22}, a_{33} are the entries of the diagonal of shape matrix A. Solving this equation, we have the eigenvalues b_1, b_2, b_3

$$b_1 = a_{11} \text{ or } b_2 = a_{22} \text{ or } b_3 = a_{33},$$

which means the eigenvalues of the shape matrix $\boldsymbol{A}$ are the diagonal entries of the shape matrix $\boldsymbol{A}$. Therefore, the eigenvalues are exactly the variances of the generated samples in different coordinate axes. Furthermore, from Equation (2.5.1), with $\lambda = 1$, we have

$$\boldsymbol{\Sigma} = \boldsymbol{D}\boldsymbol{A}\boldsymbol{D}^T,$$

which is a shape matrix $\boldsymbol{A}$ change after applying a rotation $\boldsymbol{D}$ to it. Not surprisingly, we expect that the eigenvalues of $\boldsymbol{D}\boldsymbol{A}\boldsymbol{D}^T$ keep the same as $\boldsymbol{A}$ because just the orientation changes and the shape remains the same. The proof is in the following: From Equation (2.5.2) and multiplying rotation matrix $\boldsymbol{D}$ on the right, we have

$$\boldsymbol{D}\boldsymbol{A}\boldsymbol{u} = b\boldsymbol{D}\boldsymbol{u}.$$

Because $\boldsymbol{D}^{-1}\boldsymbol{D} = \boldsymbol{I}$, we have

$$\boldsymbol{D}\boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{D}\boldsymbol{u} = b\boldsymbol{D}\boldsymbol{u}$$
$$\boldsymbol{D}\boldsymbol{A}\boldsymbol{D}^{-1}(\boldsymbol{D}\boldsymbol{u}) = b(\boldsymbol{D}\boldsymbol{u})$$

where the vector $\boldsymbol{D}\boldsymbol{u}$ is the eigenvector associated with the eigenvalue b of $\boldsymbol{D}\boldsymbol{A}\boldsymbol{D}^{-1}$. Therefore, the eigenvalues remains the same after applying a rotation matrix. If the volume scalar λ is not 1, the new eigenvalue of $\boldsymbol{\Sigma} = \lambda\boldsymbol{D}\boldsymbol{A}\boldsymbol{D}^T$ is λb. Because the volume scalar is applied to all the eigenvalues, the volume becomes bigger if $\lambda > 1$ and smaller if $\lambda < 1$, but the shape remains the same.

Convergence at critical points for mixture modelling

In Section 2.1.3, we have introduced that ECME algorithm are iterative methods to estimate the covariance

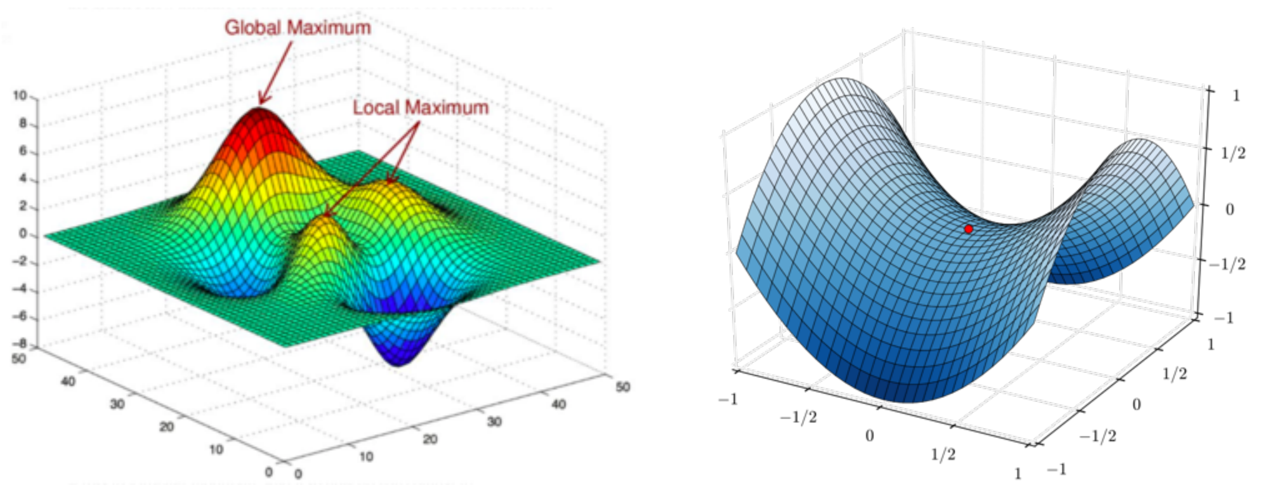


Figure 2.5.7: Illustration The global and local maximum (Left) and the saddle point (Right).

by maximizing the actual log-likelihood until convergence. Because there is no analytical solution there, it is required to do optimization numerically. Normally, the optimization algorithm will coverage to the critical point of the log-likelihood function which the derivative of the function is equal to zero. Actually, critical points refer to the maximum and minimum and saddle point. Since the log-likelihood function in general can be non-convex, it is common to have more than one maximum points, often a lot. In the Figure 2.5.7 (Left) [Heal, 2020], it shows that a function can have more than one maximums.

However, some of the local maximums that the algorithm converges to may lead to the spurious solutions which we don't want. Also, the saddle point shown in the Figure 2.5.7 (Right) is a point which is not a local maximum or minimum, but the derivative is zero. Although you may think that saddle point is not a problem for the algorithm, because it seems like it is easy to find another point with higher value of log-likelihood, most of the time, the algorithm is stuck at a saddle point and the value of log-likelihood stays the same. Based on our stopping condition in Equation (2.1.4), the algorithm stops near the saddle point before they get to the right solution.

Singularity issues in mixture modelling

One significant problem associated with the maximum likelihood and leading to error is the singularity issue. In GMM, the singularity issue makes some eigenvalues of the covariance close to 0, such that the determinant of the covariance is also close to 0. Here is the reason why this issue leads to an error: In the ECME algorithm, our purpose is to maximize the actual likelihood or log-likelihood of the density. From Equation (2.4.11), we know that the probability density function of multivariate normal distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^p$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{p \times p}$ is

$$f_{\mathcal{N}}(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \left(\frac{1}{(2\pi)^k |\boldsymbol{\Sigma}|}\right)^{\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}),\right)$$

and from the following theory: (Determinant is the product of eigenvalues) Let $\boldsymbol{B}$ be a $n \times n$ matrix and $\lambda_1, ..., \lambda_n$ be the eigenvalues of $\boldsymbol{B}$, we have

$$\det(\boldsymbol{B}) = \lambda_1 \lambda_2 \cdots \lambda_n. \tag{2.5.4}$$

Obviously, if one of the eigenvalues $\lambda_1, ..., \lambda_n$ tends to 0 for one suitable sample point x, $|\boldsymbol{\Sigma}| \to 0$ too. As we know that a singular square matrix whose determinant is 0 does not have a matrix inverse. Therefore, if $|\boldsymbol{\Sigma}| \to 0$, $\boldsymbol{\Sigma}^{-1}$ may not exist and this will lead to an error. The Figure 2.5.8 [Bishop, 2006] shows you the illustration of singularity issue in one dimension. We can see that there is a single data point fitted to a

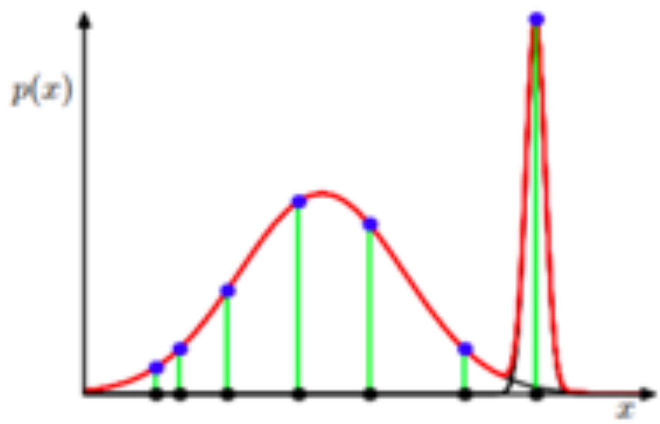


Figure 2.5.8: Illustration of singularity issue happens to Gaussian mixture model

Gaussian and the variance of the associated distribution Gaussian is very close to 0. When the variance is close to 0, the inverse of the variance may not exist and it will lead to an error.

Clusters collapse issue in mixture modelling

Another problem associated with the maximum likelihood and leading to spurious solutions is the cluster collapse issue. For two mixture components in a mixture model, clusters collapse issue is that the EM algorithm converges at a saddle point leading to a non-interesting solution : The volume of one of the cluster shrinks to almost 0, such that the determinant of covariance matrix is almost 0, because at least one of the eigenvalues of the covariance is close to 0.

The Figure 2.5.9 shows you the illustration of the estimated clusters of Gaussian mixture model with the cluster collapse issue. The dark red represents the contour plot of 95% confidence region and the black point represents the mean of the cluster. We can see that the volume of the cluster like a straight line shrinks to almost 0 with the eigenvalues of the covariance matrix 22.4, 1.43 and 8.29×10^{-16} and determinant of the covariance matrix 1.56×10^{-14}, such that no sample points are classified to belong to this cluster. The covariance matrix of the elliptical cluster is actually the covariance of all the sample points, because all the samples belong to this cluster.

Solution for avoiding mentioned issues in simulation

After a brief introduction of singularity and cluster collapse issues, now, we would like to suggest a solution to avoid these problems in simulation and explain the reason why it works.

In our simulation part, our purpose is to get the results from a certain amount of replications without singularity and cluster collapse issues. Therefore, one of the method to solve this problem is to remove replications which might have spurious solutions. [Dang et al., 2017]

The replication is removed when the following conditions are fulfilled:

$$\lambda_l(\Sigma_k^{(t)}) < 10^{-20} \quad \text{for} \quad l = 1, ..., d \ , t = 1, 2, 3, ... \quad \text{and} \quad k = 1, ..., K \tag{2.5.5}$$

or

$$\frac{\min_{l=1,...,d} \lambda_l(\Sigma_k^{(t)})}{\max_{l=1,...,d} \lambda_l(\Sigma_k^{(t)})} < 10^{-10} \quad \text{for} \quad t = 1, 2, 3, ... \quad \text{and} \quad k = 1, ..., K \tag{2.5.6}$$

where $\lambda_l(\Sigma_k^{(t)})$ is the lth eigenvalues of the estimated covariance of cluster k at tth iteration of the considered replication. [García-Escudero et al., 2015] [Dang et al., 2017] Intuitively, once one of the eigenvalues of the

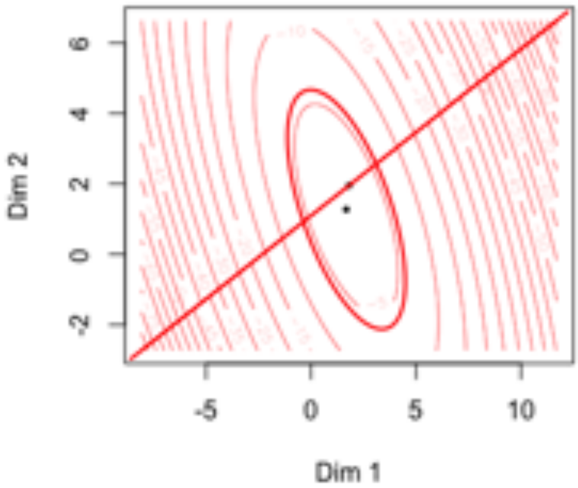


Figure 2.5.9: Illustration of cluster collapse issue happens to Gaussian mixture model.

estimated covariance is too small, we remove this replication and start a new one. Because the determinant is the product of the eigenvalues, removal of the replication result based on the above conditions can avoid the estimation of covariance with determinant close to 0. We know that the one of the obvious indicator of singularity and clusters collapse issue happening to GMM is that the determinant of estimated covariance is 0. This is reason why the proposed solution works.

2.6 Performance measures

In this section, we will introduce some measures that we will use in the simulation part for result visualization for mixture model parameter estimation. The replication R below refers to the number of replications after removing the replications with spurious solutions.

Mean number of iterations
It measures the average needed number of iterations for the replications of the experiment. The definition of the iteration for EM algorithm and its extension please refers to the Section 2.4.2. The mean number of iterations is defined as here :

$$\bar{I} := \frac{\sum_{i=1}^{R} I_i}{R}$$

where I_i is the number of iterations needed at replication i. In terms of the perspective of the mean iteration number, the lower the needed mean iteration number, the better the algorithm's performance.

However, the mean number of iterations doesn't reflect the actual computation time (seconds). We need to measure the total computation time (seconds) for time efficiency assessment.
Mean computation time (seconds)
It measures the average total needed computation time (seconds) for the replications of the experiment. The mean total computation time (seconds) is defined as here :

$$\bar{T} := \frac{\sum_{i=1}^{R} T_i}{R}$$

where T_i is the total computation time (seconds) in seconds for replication i. For time efficiency prospective, we choose the algorithm with the shortest mean total computation time (seconds).

Other than the time efficiency, we use the following measures for performance assessment about goodness of fit and clustering:

Bayesian information criterion (BIC)

The Bayesian information criterion (BIC) is a criterion for model selection or algorithms selection in our case and the one with the lowest BIC is adopted. The BIC was developed by Schwarz (1978) and it is given by

$$BIC(\boldsymbol{\eta}) = k\ln(n) - 2l(\hat{\boldsymbol{\eta}}|x),$$

where k is the the number of parameters estimated, n is the total number of observations and $l(\hat{\boldsymbol{\eta}}|x)$ is the log-likelihood introduced in Equation (3.1.1) with estimated parameter $\hat{\boldsymbol{\eta}}$ which is the output of the Algorithm 1. For GMM parameter estimation, k is determined by $K(d + \frac{(d+1)d}{2}) + (K-1)$, where K is the number of components in GMM and d is the dimension of each sample. In our experiment with 3 dimension and 2 clusters, the total number of parameters estimated is 19. The model with the lowest $BIC(\boldsymbol{\eta})$ is the preferred model.

Normalised Bayesian information criterion (BIC)

In our experiment setting, the number of samples n generated are 100 or 500. From Equation (3.1.1), we know that the log-likelihood is the summation of n terms. Therefore, it implies that the BIC for 500 samples must be higher than 100 samples. In order to have an appropriate comparison of the BIC for different number of samples n. We prefer to use the normalised BIC and it is defined as follows :

$$\frac{BIC(\boldsymbol{\eta})}{n} = \frac{k\ln(n) - 2l(\hat{\boldsymbol{\eta}}|x)}{n}$$

Classification rate

Firstly, we would like to introduce how generated samples are finally grouped into clusters after parameter estimation from EM and its extension. From Equation (2.4.4), we know that the E step for the algorithm is to compute the expectation of the latent variable given samples for GMM, such that

$$\begin{aligned} E[Z_{ik} \mid \boldsymbol{\eta}, \boldsymbol{x}] := r_{ik} &= 1 \; p(Z_{ik} = 1 \mid \boldsymbol{x}_i) + 0 \; p(Z_{ik} = 0 \mid \boldsymbol{x}_i) \\ &= p(Z_{ik} = 1 \mid \boldsymbol{x}_i) \text{ for } \; i = 1, ..., n \; \text{ and } \; k = 1, ..., K \end{aligned}$$

where $p(Z_{ik} = 1 \mid \boldsymbol{x}_i)$ is the probability of sample $\boldsymbol{x}_i$ to be in clusters k,$k = 1, ..., K$. Also, from Equation (2.4.12), the E step, such that

$$r_{ik} = \frac{\pi_k f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \quad \text{for} \quad i = 1, ..., n \quad \text{and} \quad k = 1, ..., K$$

is actually computing the probability of being each cluster for sample $\boldsymbol{x}_i$. Therefore, after the algorithm converges, the E step will be additionally implemented using the estimated parameters to do the clustering for each sample. All observations i which have $\max\limits_{k'=1,..,k} r_{ik'} = r_{ik}$ will be put in Cluster k.

The classification rate is also called the Rand index by Rand [1971] and it is defined as here :

Rand index : Given a set of n element $S = \{o_1, ..., o_n\}$ and two partitions of S to compare, $X = \{X_1, ..., X_r\}$, a partition of S into r subsets, and $Y = \{Y_1, ..., Y_s\}$, a partition of S into s subsets, the Rand index is given by

$$\text{Rand index} = \frac{a+b}{\text{Number of unordered pairs in } S} = \frac{a+b}{\binom{n}{2}}$$

where a is the number of pairs of elements in S that are in the same subset in X and in the same subset in Y, b is the number of pairs of elements in S that are in different subset in X and in different subset in Y.

Example 2.6.1 (Calculation of the Rand index)

Suppose we have a data set with 4 elements $S = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$. The correct cluster is $X = \{1, 1, 1, 2\}$, i.e. cluster 1 consists of $\{A, B, C\}$ and cluster 2 of $\{D\}$. Also, the clustering result from the EM algorithm is $Y = \{1, 1, 2, 2\}$, i.e. cluster 1 = $\{A, B\}$ and cluster 2 = $\{C, D\}$.

To calculate the Rand index, we need to list out every possible unordered pair in the dataset of four elements: $\{\mathrm{A}, \mathrm{B}\}, \{\mathrm{A}, \mathrm{C}\}, \{\mathrm{A}, \mathrm{D}\}, \{\mathrm{B}, \mathrm{C}\}, \{\mathrm{B}, \mathrm{D}\}, \{\mathrm{C}, \mathrm{D}\}$. There are 6 unordered pairs which is equal to $\binom{4}{2}$.

Now, we need to calculate a, which represents the number of unordered pairs in S that belong to the same cluster in X and in the same subset in Y: $\{\mathrm{A}, \mathrm{B}\}$. In this case, a is 1.

Then, we need to calculate b, which represents the number of unordered pairs in S that belong to different cluster in X and Y: $\{\mathrm{A}, \mathrm{D}\}, \{\mathrm{B}, \mathrm{D}\}$. In this case, b is 2.

Therefore, we can calculate the Rand index:

$$\text{Rand index} = \frac{1+2}{6} = 0.5$$

Also, the classification rate is 0.5.

3 Simulation study

In this chapter, we will use different EM algorithms presented in the Section 2.1 to estimate the parameters for the simulated data from the Gaussian mixture model (**GMM**) and vine copula mixture models (**VCMM**) in order to assess the performance, strengths and restrictions of different EM algorithms.

3.1 EM Algorithm for GMM algorithm

In the Section 2.3, the formulation of Gaussian mixture model and its parameter estimation steps for different EM algorithms are introduced. Now, our main goal is to carry out simulation studies or experiments to compare the performance of different EM algorithms for the Gaussian mixture model. The brief procedure of the experiment is in the following:

Step 1: Generate data for simulation.
Step 2: Decide on the type of EM algorithm and the order of the CM steps for testing.
Step 3: Decide on the starting values and stopping condition for the EM algorithms.
Step 4: Perform the simulation in R.
Step 5: Remove the replication results from spurious solutions.
Step 6: Create visualizations of the performance measures.
Step 7: Analyse and compare the results.

The detail for each step is given in the subsection:

3.1.1 Data simulation and the experiment setup

Step 1: Generate data for simulation.
In this step, we would like to generate some data from the Gaussian mixture model to test different EM algorithms. Because it is possible that the performance of different EM algorithms varies for a Gaussian mixture model with different number of samples, mean and covariance matrix, we would like to set up different scenarios / experiments in order to find out the corresponding characteristics, strengths and restrictions of different EM algorithms.

Table 3.1.1: The basic set up of Gaussian mixture model in each experiment

Code of Experiment	Number of samples (n)	Mean of 2 clusters	Covariance matrix of 2 clusters
EEV.emu.100	100	Equal	EEV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
VEV.emu.100	100	Equal	VEV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
EVV.emu.100	100	Equal	EVV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$
VVV.emu.100	100	Equal	VVV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$
EEV.umu.100	100	Unequal	EEV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
VEV.umu.100	100	Unequal	VEV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
EVV.umu.100	100	Unequal	EVV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$
VVV.umu.100	100	Unequal	VVV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$
EEV.emu.500	500	Equal	EEV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
VEV.emu.500	500	Equal	VEV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
EVV.emu.500	500	Equal	EVV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$
VVV.emu.500	500	Equal	VVV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$
EEV.umu.500	500	Unequal	EEV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
VEV.umu.500	500	Unequal	VEV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A} \boldsymbol{D}_k{}^T$
EVV.umu.500	500	Unequal	EVV, $\boldsymbol{\Sigma}_k = \lambda \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$
VVV.umu.500	500	Unequal	VVV, $\boldsymbol{\Sigma}_k = \lambda_k \boldsymbol{D}_k \boldsymbol{A}_k \boldsymbol{D}_k{}^T$

In our experiments, the dimension of each sample d is 3 and number of clusters K is 2. The Table 3.1.1 shows the basic set up for different experiments. For example, in experiment EEV.emu.100, 100 samples in total are generated from the Gaussian mixture model with the same mean in each cluster and different covariance matrix following the model type EEV in each cluster, which we introduced in the previous Section 2.5.1. For simplicity, just the models EEV, VEV, EVV and VVV are used in our simulation. Actually, the real data are mostly with different orientations. Therefore, we expect that the simulation result from these four models are more appropriate for the envisioned real data applications.

Other than the basic set up in the Table 3.1.1, we still need to decide the value of the mean vector $\boldsymbol{\mu}_k$, covariance matrix $\boldsymbol{\Sigma}_k$ and the rest of the needed parameters, for example, mixing proportion π_k and so on. Details are shown in Table 3.1.2 and (3.1.3. Also, the correlation matrices of the covariance matrices mentioned in Table 2.4.3) are shown in Table 3.1.4. You can see that different strength levels of correlation are included in each of the correlation matrix.

Example 3.1.1 (The process of data generation and visualization of simulated data from the Gaussian mixture model for different model types)

Let's take the experiment VVV.emu.100 as an example. Assume the number of samples is 100, the means of 2 clusters are equal and the covariance matrices of 2 clusters are following model type VVV. We generate 100 samples and the probability of generating each sample from $\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1 = \lambda \boldsymbol{D}_1 \boldsymbol{A} \boldsymbol{D}_1{}^T)$ is π_1 and from $\mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2 = \lambda \boldsymbol{D}_2 \boldsymbol{A} \boldsymbol{D}_2{}^T)$ is π_2. We use the values given in the Table 3.1.3 to show the calculation of the needed

Table 3.1.2: The parameter values used in the data simulation for Gaussian mixture model

Notation	Name	Value
n	Sample size	100 or 500
d	Dimension of each sample	3
K	Total number of components	2
π_1	Mixing proportion of component 1	0.4
π_2	Mixing proportion of component 2	0.6
Mean of 2 clusters : Equal		
$\boldsymbol{\mu}_1$	Mean vector of component 1	$(1.12 \quad 1.63 \quad 1.95)^T$
$\boldsymbol{\mu}_2$	Mean vector of component 2	$(1.12 \quad 1.63 \quad 1.95)^T$
Mean of 2 clusters : Unequal		
$\boldsymbol{\mu}_1$	Mean vector of component 1	$(1.12 \quad 1.63 \quad 1.95)^T$
$\boldsymbol{\mu}_2$	Mean vector of component 2	$(\text{-}1.80 \quad 1.50 \quad 1.13)^T$

parameters in the following:

$$\pi_1 = 0.4, \quad \pi_2 = 0.6$$

$$\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = [1.12 \ \ 1.63 \ \ 1.95]^T$$

$$\boldsymbol{\Sigma}_1 = \lambda_1 \boldsymbol{D}_1 \boldsymbol{A}_1 \boldsymbol{D}_1{}^T \approx 1 \begin{bmatrix} 0 & -0.423 & 0.906 \\ -0.5 & 0.785 & 0.366 \\ -0.866 & -0.453 & -0.211 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{6} & 0 \\ 0 & 0 & 6 \end{bmatrix} \begin{bmatrix} 0 & -0.423 & 0.906 \\ -0.5 & 0.785 & 0.366 \\ -0.866 & -0.453 & -0.211 \end{bmatrix}^T$$

$$\approx \begin{bmatrix} 4.958 & 1.935 & -1.117 \\ 1.935 & 1.156 & -0.090 \\ -1.117 & -0.090 & 1.052 \end{bmatrix}$$

$$\text{where} \ \ \boldsymbol{D}_1 = \begin{bmatrix} \cos 90^\circ & -\sin 90^\circ & 0 \\ \sin 90^\circ & \cos 90^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 120^\circ & 0 & \sin 120^\circ \\ 0 & 1 & 0 \\ -\sin 120^\circ & 0 & \cos 120^\circ \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos 65^\circ & -\sin 65^\circ \\ 0 & \sin 65^\circ & \cos 65^\circ \end{bmatrix}$$

$$\approx \begin{bmatrix} 0 & -0.423 & 0.906 \\ -0.5 & 0.785 & 0.366 \\ -0.866 & -0.453 & -0.211 \end{bmatrix}$$

$$\boldsymbol{\Sigma}_2 = \lambda_2 \boldsymbol{D}_2 \boldsymbol{A}_2 \boldsymbol{D}_2{}^T \approx 2 \begin{bmatrix} 0.25 & -0.276 & 0.928 \\ -0.433 & 0.825 & 0.362 \\ -0.866 & -0.492 & 0.087 \end{bmatrix} \begin{bmatrix} 4 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.25 & -0.276 & 0.928 \\ -0.433 & 0.825 & 0.362 \\ -0.866 & -0.492 & 0.087 \end{bmatrix}^T$$

$$\approx \begin{bmatrix} 2.261 & -0.308 & -1.503 \\ -0.308 & 2.103 & 2.860 \\ -1.503 & 2.860 & 6.136 \end{bmatrix}$$

$$\text{where} \ \ \boldsymbol{D}_2 = \begin{bmatrix} \cos 120^\circ & -\sin 120^\circ & 0 \\ \sin 120^\circ & \cos 120^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 120^\circ & 0 & \sin 120^\circ \\ 0 & 1 & 0 \\ -\sin 120^\circ & 0 & \cos 120^\circ \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos 100^\circ & -\sin 100^\circ \\ 0 & \sin 100^\circ & \cos 100^\circ \end{bmatrix}$$

$$\approx \begin{bmatrix} 0.25 & -0.276 & 0.928 \\ -0.433 & 0.825 & 0.362 \\ -0.866 & -0.492 & 0.087 \end{bmatrix}$$

For the experiment VVV.emu.100, we generate 100 samples from Cluster 1

$$\mathcal{N}\left(\begin{bmatrix} 1.12 \\ 1.63 \\ 1.95 \end{bmatrix}, \begin{bmatrix} 4.958 & 1.935 & -1.117 \\ 1.935 & 1.156 & -0.090 \\ -1.117 & -0.090 & 1.052 \end{bmatrix} \right) \text{ with probability } 0.4$$

Table 3.1.3: The value of the parameters used in the data simulation for Gaussian mixture model (Continued), where α, β and γ represent yaw, pitch, and roll angles following the Figure 2.5.4 (Katherine et al., 2014) in the Section 2.5.

		Gaussian mixture model type			
		EEV	VEV	EVV	VVV
$\boldsymbol{\Sigma}_1$	Covariance matrix of component 1	$\lambda \boldsymbol{D}_1 \boldsymbol{A} \boldsymbol{D}_1{}^T$	$\lambda_1 \boldsymbol{D}_1 \boldsymbol{A} \boldsymbol{D}_1{}^T$	$\lambda \boldsymbol{D}_1 \boldsymbol{A}_1 \boldsymbol{D}_1{}^T$	$\lambda_1 \boldsymbol{D}_1 \boldsymbol{A}_1 \boldsymbol{D}_1{}^T$
$\boldsymbol{\Sigma}_2$	Covariance matrix of component 2	$\lambda \boldsymbol{D}_2 \boldsymbol{A} \boldsymbol{D}_2{}^T$	$\lambda_2 \boldsymbol{D}_2 \boldsymbol{A} \boldsymbol{D}_2{}^T$	$\lambda \boldsymbol{D}_2 \boldsymbol{A}_2 \boldsymbol{D}_2{}^T$	$\lambda_2 \boldsymbol{D}_2 \boldsymbol{A}_2 \boldsymbol{D}_2{}^T$
λ, λ_1	Scalar, Scalar of component 1	1	1	1	1
λ_2	Scalar of component 2	-	2	-	2
$\boldsymbol{D}_1$	Rotation matrix of component 1	$\alpha = 90°, \beta = 120°, \gamma = 65°$			
$\boldsymbol{D}_2$	Rotation matrix of component 2	$\alpha = 120°, \beta = 120°, \gamma = 100°$			
$\boldsymbol{A}, \boldsymbol{A}_1$	Shape matrix of component 1	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{6} & 0 \\ 0 & 0 & 6 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{6} & 0 \\ 0 & 0 & 6 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{6} & 0 \\ 0 & 0 & 6 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{6} & 0 \\ 0 & 0 & 6 \end{bmatrix}$
$\boldsymbol{A}_2$	Shape matrix of component 2	-	$\begin{bmatrix} 4 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{bmatrix}$	-	$\begin{bmatrix} 4 & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Gaussian mixture model type	Covariance matrix of component 1, $\boldsymbol{\Sigma}_1$	Covariance matrix of component 2, $\boldsymbol{\Sigma}_2$
EEV	$\approx \begin{bmatrix} 4.958 & 1.935 & -1.117 \\ 1.935 & 1.156 & -0.090 \\ -1.117 & -0.090 & 1.052 \end{bmatrix}$	$\approx \begin{bmatrix} 5.243 & 1.870 & 0.290 \\ 1.870 & 1.088 & 0.496 \\ 0.290 & 0.496 & 0.836 \end{bmatrix}$
VEV	$\approx \begin{bmatrix} 4.958 & 1.935 & -1.117 \\ 1.935 & 1.156 & -0.090 \\ -1.117 & -0.090 & 1.052 \end{bmatrix}$	$\approx \begin{bmatrix} 10.486 & 3.741 & 0.579 \\ 3.741 & 2.176 & 0.992 \\ 0.579 & 0.992 & 1.671 \end{bmatrix}$
EVV	$\approx \begin{bmatrix} 4.958 & 1.935 & -1.117 \\ 1.935 & 1.156 & -0.090 \\ -1.117 & -0.090 & 1.052 \end{bmatrix}$	$\approx \begin{bmatrix} 1.130 & -0.154 & -0.751 \\ -0.154 & 1.051 & 1.430 \\ -0.751 & 1.430 & 3.068 \end{bmatrix}$
VVV	$\approx \begin{bmatrix} 4.958 & 1.935 & -1.117 \\ 1.935 & 1.156 & -0.090 \\ -1.117 & -0.090 & 1.052 \end{bmatrix}$	$\approx \begin{bmatrix} 2.261 & -0.308 & -1.503 \\ -0.308 & 2.103 & 2.860 \\ -1.503 & 2.860 & 6.136 \end{bmatrix}$

Table 3.1.4: The corresponding correlation matrices of the covariance matrices mentioned in Table 2.4.3), where the background colour of the text represents different strengths of the correlation, -0.082 is low, -0.489 is medium and 0.808 is high.

Gaussian mixture model type	Correlation matrix of component 1, $\boldsymbol{R}_1$	Correlation matrix of component 2, $\boldsymbol{R}_2$
EEV	$\approx \begin{bmatrix} 1 & 0.808 & -0.489 \\ 0.808 & 1 & -0.082 \\ -0.489 & -0.082 & 1 \end{bmatrix}$	$\approx \begin{bmatrix} 1 & 0.783 & 0.138 \\ 0.783 & 1 & 0.520 \\ 0.138 & 0.520 & 1 \end{bmatrix}$
VEV	$\approx \begin{bmatrix} 1 & 0.808 & -0.489 \\ 0.808 & 1 & -0.082 \\ -0.489 & -0.082 & 1 \end{bmatrix}$	$\approx \begin{bmatrix} 1 & 0.783 & 0.138 \\ 0.783 & 1 & 0.520 \\ 0.138 & 0.520 & 1 \end{bmatrix}$
EVV	$\approx \begin{bmatrix} 1 & 0.808 & -0.489 \\ 0.808 & 1 & -0.082 \\ -0.489 & -0.082 & 1 \end{bmatrix}$	$\approx \begin{bmatrix} 1 & -0.141 & -0.404 \\ -0.141 & 1 & 0.796 \\ -0.404 & 0.796 & 1 \end{bmatrix}$
VVV	$\approx \begin{bmatrix} 1 & 0.808 & -0.489 \\ 0.808 & 1 & -0.082 \\ -0.489 & -0.082 & 1 \end{bmatrix}$	$\approx \begin{bmatrix} 1 & -0.141 & -0.404 \\ -0.141 & 1 & 0.796 \\ -0.404 & 0.796 & 1 \end{bmatrix}$

and Cluster 2

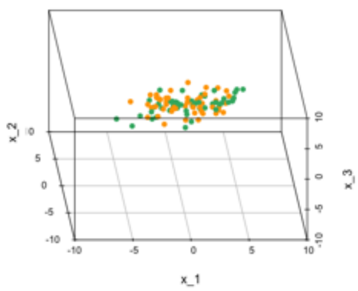
$$\mathcal{N}\left(\begin{bmatrix} 1.12 \\ 1.63 \\ 1.95 \end{bmatrix}, \begin{bmatrix} 2.261 & -0.308 & -1.503 \\ -0.308 & 2.103 & 2.860 \\ -1.503 & 2.860 & 6.136 \end{bmatrix}\right) \text{ with probability } 0.6.$$

The generated samples are plotted as a 3D scatter plot and pair plot and the orange dots are from Cluster 1 and green dots are from Cluster 2. The plots of all the data simulations are shown in the Figure 3.1.1 and Figure 3.1.2. By comparing the scatter plots, we can see that the overlapping level between 2 clusters is higher for each model type with equal mean. Also, if 2 scatters are not coloured, it is quite hard to observe there are 2 clusters in EEV and VEV. Oppositely, it is clearer to see there are 2 clusters in EVV and VVV. Therefore, we can expect that the result of the performance on classification rate is higher on 1) the unequal mean cases and data settings 2) EVV and VVV.

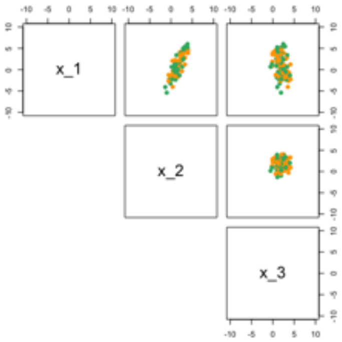
Step 2: Decide on the EM algorithms and the CM steps order for testing.
In Section 2, we have introduced different EM algorithms for parameter estimation, for example, classical EM, ECM, ECME and MCECM. As we mentioned, it is impossible to find a analytic solution to maximize a function for 2 parameters or more at the same time. Strictly speaking, we don't really use the classical EM algorithm for mixture model parameters estimation. Therefore, we prefer to use the ECM, ECME or MCECM algorithms to update one parameter in each CM step. Because we decide to update each parameter once in each iteration, there are different orders to update the parameters. Although there is no rule which orders reach better performance, for simplicity, we just consider the typical order to update the parameters, such that $\boldsymbol{\pi}$, followed by $\boldsymbol{\mu}$ and then $\boldsymbol{\Sigma}$ will be updated accordingly in each iteration.

Apart from the order, the optimization method is also needed to be chosen in the ECME algorithm because different methods affect the performance of parameter estimation when maximizing the actual log-likelihood. In the experiment, two optimization methods, Nelder-Mead and BFGS are used to perform the optimization.

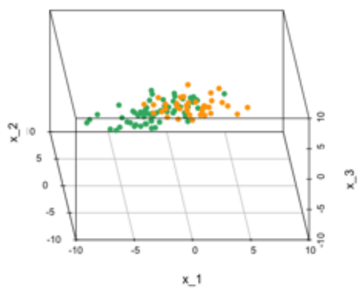
In the Table 3.1.5, the selection of the EM algorithms and CM steps order for testing is shown. Order $= \{\theta_1, \theta_2, \theta_3\}$ means that the order of updating in CM steps is θ_1, then θ_2 and followed by θ_3. For the



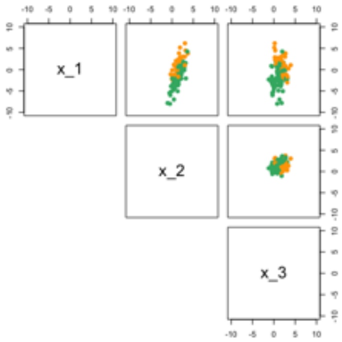
(a) 3D scatter plot : EEV.emu.100



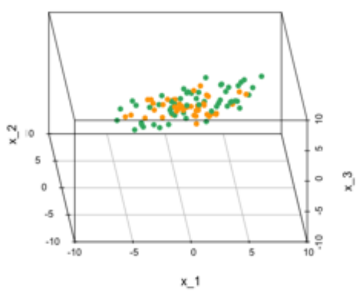
(b) Pairs plot : EEV.emu.100



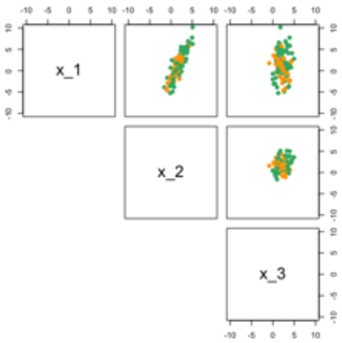
(c) 3D scatter plot : EEV.umu.100



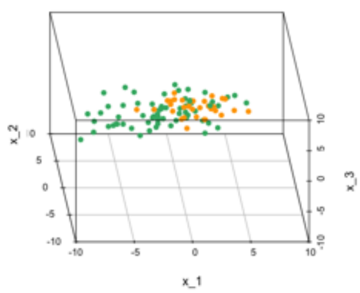
(d) Pairs plot : EEV.umu.100



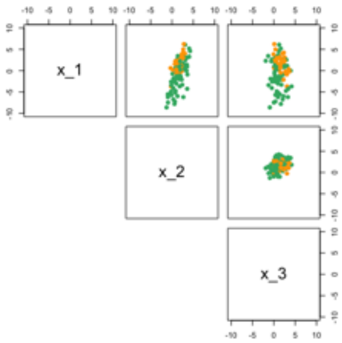
(e) 3D scatter plot : VEV.emu.100



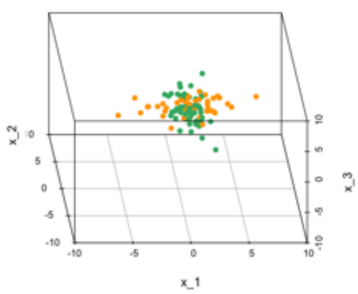
(f) Pairs plot : VEV.emu.100



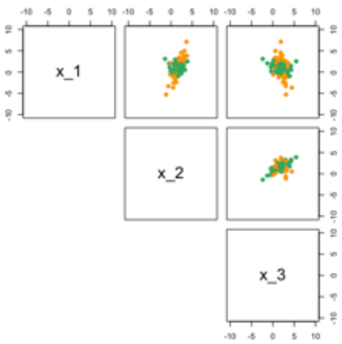
(g) 3D scatter plot : VEV.umu.100



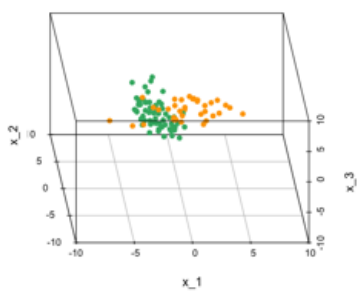
(h) Pairs plot : VEV.umu.100



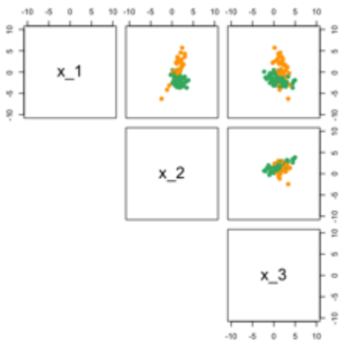
(i) 3D scatter plot : EVV.emu.100



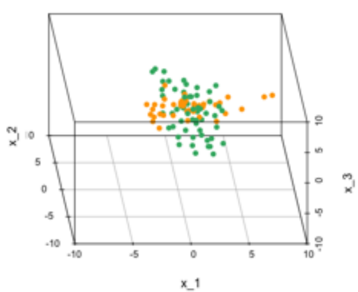
(j) Pairs plot : EVV.emu.100



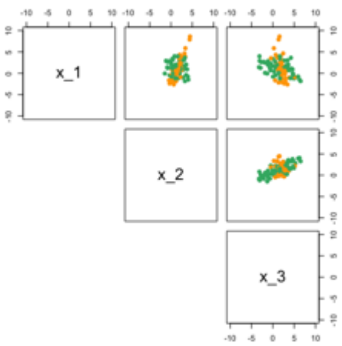
(k) 3D scatter plot : EVV.umu.100



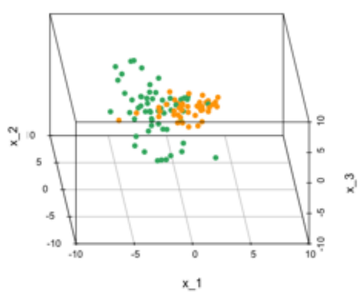
(l) Pairs plot : EVV.umu.100



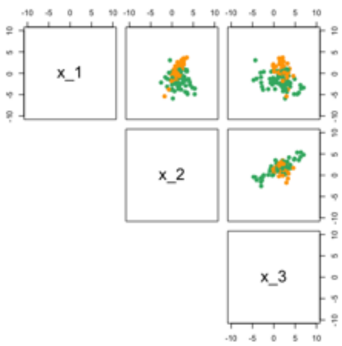
(m) 3D scatter plot : VVV.emu.100



(n) Pairs plot : VVV.emu.100

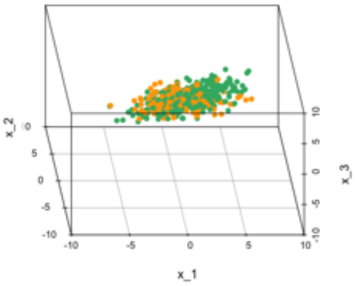


(o) 3D scatter plot : VVV.umu.100

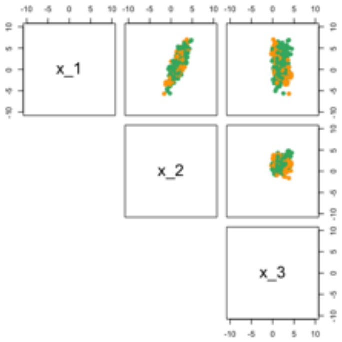


(p) Pairs plot : VVV.umu.100

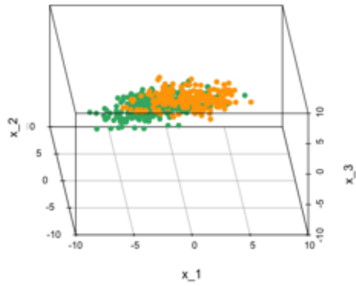
Figure 3.1.1: 3D Scatter plots and the corresponding pairs plots of 100 simulated data generated from a Gaussian Mixture Model with different model types: EEV, VEV, EVV, VVV. The orange and green colour refer to the points in Cluster 1 and 2 respectively.



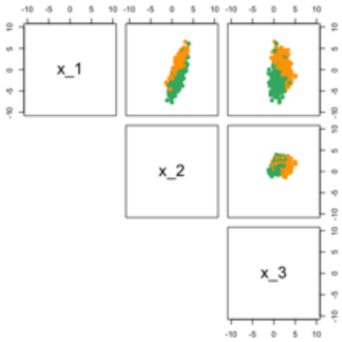
(a) 3D scatter plot : EEV.emu.500



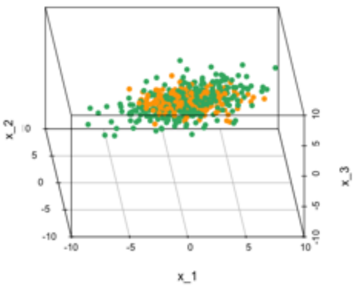
(b) Pairs plot : EEV.emu.500



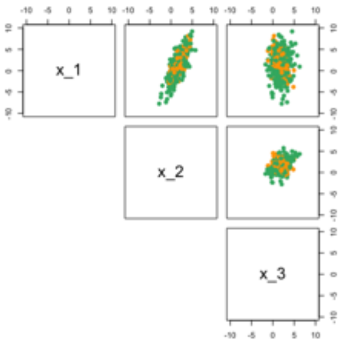
(c) 3D scatter plot : EEV.umu.500



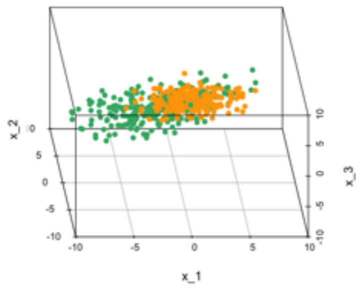
(d) Pairs plot : EEV.umu.500



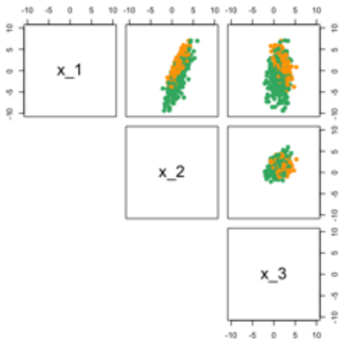
(e) 3D scatter plot : VEV.emu.500



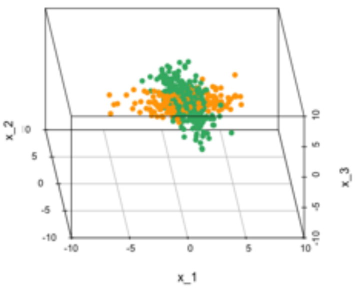
(f) Pairs plot : VEV.emu.500



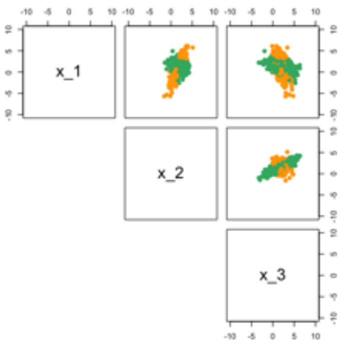
(g) 3D scatter plot : VEV.umu.500



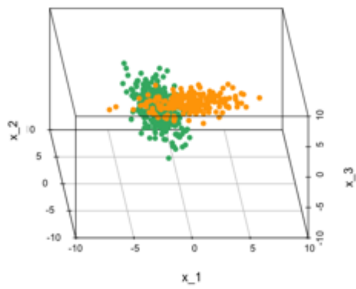
(h) Pairs plot : VEV.umu.500



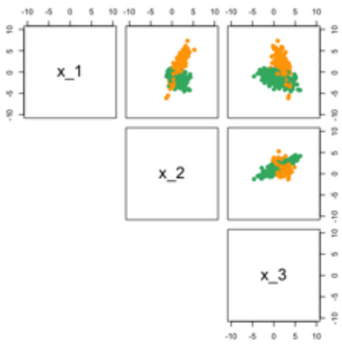
(i) 3D scatter plot : EVV.emu.500



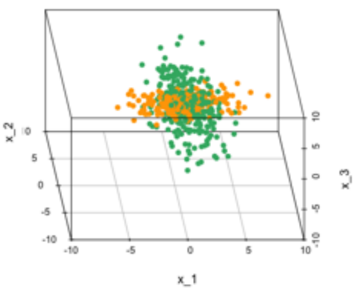
(j) Pairs plot : EVV.emu.500



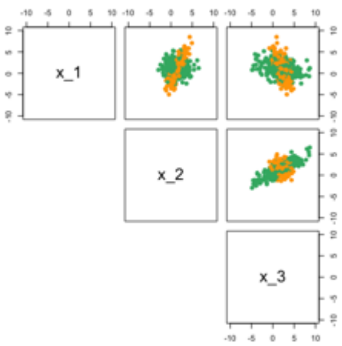
(k) 3D scatter plot : EVV.umu.500



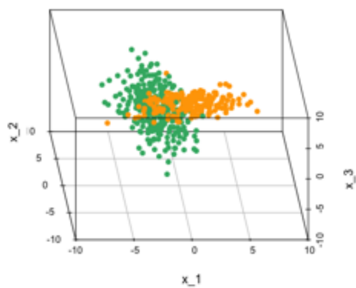
(l) Pairs plot : EVV.umu.500



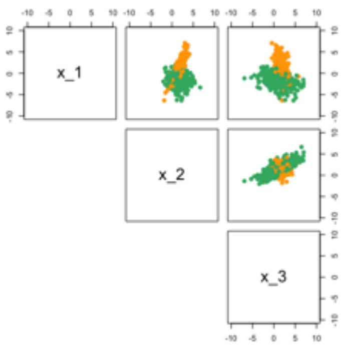
(m) 3D scatter plot : VVV.emu.500



(n) Pairs plot : VVV.emu.500



(o) 3D scatter plot : VVV.umu.500



(p) Pairs plot : VVV.umu.500

Figure 3.1.2: 3D Scatter plots and the corresponding pairs plots of 500 simulated data generated from a Gaussian Mixture Model with different model types: EEV, VEV, EVV, VVV. The orange and green colour refer to the points in Cluster 1 and 2 respectively.

Table 3.1.5: Selection of the EM algorithms and CM steps order for testing

Order $= \{\theta_1, \theta_2, \theta_3\}$	Code of order	ECM	ECME (Nelder-Mead)	ECME (BFGS)	MCECM
$\{\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$	123	✓	✓	✓	✓

Table 3.1.6: The selected starting values of the parameters for the EM algorithm used in the data simulation for Gaussian mixture model

Notation	Name	Value
π_1	Mixing proportion of component 1	0.55
π_2	Mixing proportion of component 2	0.45
$\boldsymbol{\mu}_1$	Mean vector of component 1	$(2\ \ 0\ \ 1)^T$
$\boldsymbol{\mu}_2$	Mean vector of component 2	$(0\ \ 2\ \ 1)^T$
$\boldsymbol{\Sigma}_1$	Covariance matrix of component 1	$\begin{bmatrix} 5.431 & -1.487 & -0.651 \\ -1.488 & 7.870 & 0.524 \\ -0.651 & 0.524 & 6.151 \end{bmatrix}$
$\boldsymbol{\Sigma}_2$	Covariance matrix of component 2	$\begin{bmatrix} 4.015 & 1.431 & 1.325 \\ 1.431 & 1.476 & 1.593 \\ 1.325 & 1.593 & 2.719 \end{bmatrix}$

convenience of showing the updating order in the graph, the code 1, 2 and 3 represents π, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ respectively. Therefore, the code of order 123 represents that the updating order in CM step is $\{\pi, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$.

Step 3: Decide on the starting values and stopping condition for the EM algorithms.

Since the EM algorithm and its extensions are all iterative algorithms, we need to decide the starting values of the parameters that we are going to estimate. Their initial values are denoted as $\boldsymbol{\eta}^{(0)} = \{\pi_1^{(0)}, \pi_2^{(0)}, \boldsymbol{\mu}_1^{(0)}, \boldsymbol{\mu}_2^{(0)}, \boldsymbol{\Sigma}_1^{(0)}, \boldsymbol{\Sigma}_2^{(0)}\}$. In each experiment and simulation, we use the same set of the parameters $\boldsymbol{\pi}$ and $\boldsymbol{\Sigma}$ of the Gaussian mixture model as the starting values and the chosen values should not be too close or too far from the true values. The selected starting values are shown in the Table 3.1.6.

Other than the starting values, the stopping condition is also needed in order to terminate the algorithm when it converges. Theoretically, the log-likelihood increases at each iteration of the EM algorithms. Therefore, the stopping condition 2 from Equation (2.1.4) is used and we select the prespecified ϵ as 0.001 here. Formally, the condition can be expressed as follows:

$$\frac{l(\boldsymbol{\eta}^{(t)}) - l(\boldsymbol{\eta}^{(t-1)})}{l(\boldsymbol{\eta}^{(t-1)})} < 0.001, \quad \text{for} \quad t = 1, 2, ...$$

where the the actual incomplete data log-likelihood for $\boldsymbol{\eta} = (\pi_1, \pi_2, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)$ is given by

$$l(\boldsymbol{\eta}) := l(\boldsymbol{\eta} \mid \boldsymbol{x}) = \sum_{i=1}^{n} \ln \left(\sum_{k=1}^{2} \pi_k f_{\mathcal{N}}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \tag{3.1.1}$$

and $l(\boldsymbol{\eta}^{(0)})$ represents the value of the actual log-likelihood with the starting values of the parameters and t is number of iterations ranging from 1 to the iteration when it terminates.

Step 4: Perform the simulation in R.

Each experiment shown in the Table 3.1.1 will be repeated for 100 times, which means that the replication for each experiment R is 100. However, the replication results from spurious solutions will be excluded in the needed 100 replications. More detail will be given in Step 5. The detailed steps of the algorithms for parameter estimation for a single replication is shown in the Algorithm (1).

Algorithm 1 The ECM, ECME and MCECM algorithm for GMM for a single replication

Input:

1. d-dimensional n samples $\boldsymbol{x}_i = (x_{i,1}, ..., x_{i,d})^T \in \mathbb{R}^d, i = 1, ..., n$.
2. Total number of component K.
3. Starting values of the parameters $\boldsymbol{\eta}^{(0)} = \{\pi_1^{(0)}, ..., \pi_K^{(0)}, \boldsymbol{\mu}_1^{(0)}, ..., \boldsymbol{\mu}_K^{(0)}, \boldsymbol{\Sigma}_1^{(0)}, ..., \boldsymbol{\Sigma}_{\boldsymbol{K}}^{(0)}\}$
4. The used EM algorithm, denoted as $method \in$ {ECM, ECME, MCECM}.

Output:

1. The estimated parameters $\hat{\boldsymbol{\eta}} = \{\hat{\pi}_1, ..., \hat{\pi}_K, \hat{\boldsymbol{\mu}}_1, ..., \hat{\boldsymbol{\mu}}_K, \hat{\boldsymbol{\Sigma}}_1, ..., \hat{\boldsymbol{\Sigma}}_K\}$.
2. A clustering partition of the samples.
3. Number of iterations, Total computation time, Log-likelihood, BIC, Classification rate.

iteration $t \leftarrow 0$

while $t = 0$ or relative change of the log-likelihood $\frac{l(\boldsymbol{\eta}^{(t)}) - l(\boldsymbol{\eta}^{(t-1)})}{l(\boldsymbol{\eta}^{(t-1)})} < 0.001$ **do**

iteration $t \leftarrow t + 1$

$\pi_k^{(update)} \leftarrow \pi_k^{(t-1)}, \boldsymbol{\mu}_k^{(update)} \leftarrow \boldsymbol{\mu}_k^{(t-1)}, \boldsymbol{\Sigma}_k^{(update)} \leftarrow \boldsymbol{\Sigma}_k^{(t-1)}$ for $k = 1, ..., K$

The ECM algorithm

if $method$ = ECM **then**

E step (Calculate the expected complete data log-likelihood)

$r_{ik}^{(update)} \leftarrow \frac{\pi_k^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k^{(update)}, \boldsymbol{\Sigma}_k^{(update)})}{\sum_{j=1}^K \pi_j^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j^{(update)}, \boldsymbol{\Sigma}_j^{(update)})}$ for $i = 1, ..., n$ and $k = 1, ..., K$.

CM step (Maximize the expected complete data Log-likelihood)

$\pi_k^{(update)} \leftarrow \frac{\sum_{i=1}^n r_{ik}^{(update)}}{n}$ for $k = 1, ..., K$

$\boldsymbol{\mu}_k^{(update)} \leftarrow \frac{\sum_{i=1}^n r_{ik}^{(update)} x_i}{\sum_{i=1}^n r_{ik}^{(update)}}$ for $k = 1, ..., K$

$\boldsymbol{\Sigma}_k^{(update)} = \frac{\sum_{i=1}^n r_{ik}^{(update)} (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(update)})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(update)})^T}{\sum_{i=1}^n r_{ik}^{(update)}}$ for $k = 1, ..., K$

The ECME algorithm

else if $method$ = ECME **then**

E step (Calculate the expected complete data log-likelihood)

$r_{ik}^{(update)} \leftarrow \frac{\pi_k^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k^{(update)}, \boldsymbol{\Sigma}_k^{(update)})}{\sum_{j=1}^K \pi_j^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j^{(update)}, \boldsymbol{\Sigma}_j^{(update)})}$ for $i = 1, ..., n$ and $k = 1, ..., K$.

CM step 1 (Maximize the expected complete data Log-likelihood)

$\pi_k^{(update)} \leftarrow \frac{\sum_{i=1}^n r_{ik}^{(update)}}{n}$ for $k = 1, ..., K$

$\boldsymbol{\mu}_k^{(update)} \leftarrow \frac{\sum_{i=1}^n r_{ik}^{(update)} x_i}{\sum_{i=1}^n r_{ik}^{(update)}}$ for $k = 1, ..., K$

CM step 2 (Maximize the actual log-likelihood)

$\boldsymbol{\Sigma}_k^{(update)} \leftarrow \arg\max_{\boldsymbol{\Sigma}_k} \sum_{i=1}^n \ln\left(\sum_{k=1}^K \pi_k^{(update)} f_N(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k^{(update)}, \boldsymbol{\Sigma}_k^{(update)})\right)$ for $k = 1, ..., K$

The MCECM algorithm

else if $method$ = MCECM **then**

E step (Calculate the expected complete data log-likelihood)

$r_{ik}^{(update)} \leftarrow \frac{\pi_k^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k^{(update)}, \boldsymbol{\Sigma}_k^{(update)})}{\sum_{j=1}^K \pi_j^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j^{(update)}, \boldsymbol{\Sigma}_j^{(update)})}$ for $i = 1, ..., n$ and $k = 1, ..., K$.

CM step (Maximize the expected complete data log-likelihood)

$\pi_k^{(update)} \leftarrow \frac{\sum_{i=1}^n r_{ik}^{(update)}}{n}$ for $k = 1, ..., K$

E step (Calculate the expected complete data log-likelihood)

$r_{ik}^{(update)} \leftarrow \frac{\pi_k^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k^{(update)}, \boldsymbol{\Sigma}_k^{(update)})}{\sum_{j=1}^K \pi_j^{(update)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j^{(update)}, \boldsymbol{\Sigma}_j^{(update)})}$ for $i = 1, ..., n$ and $k = 1, ..., K$.

CM step (Maximize the expected complete data log-likelihood)

$\boldsymbol{\mu}_k^{(update)} \leftarrow \frac{\sum_{i=1}^n r_{ik}^{(update)} x_i}{\sum_{i=1}^n r_{ik}^{(update)}}$ for $k = 1, ..., K$

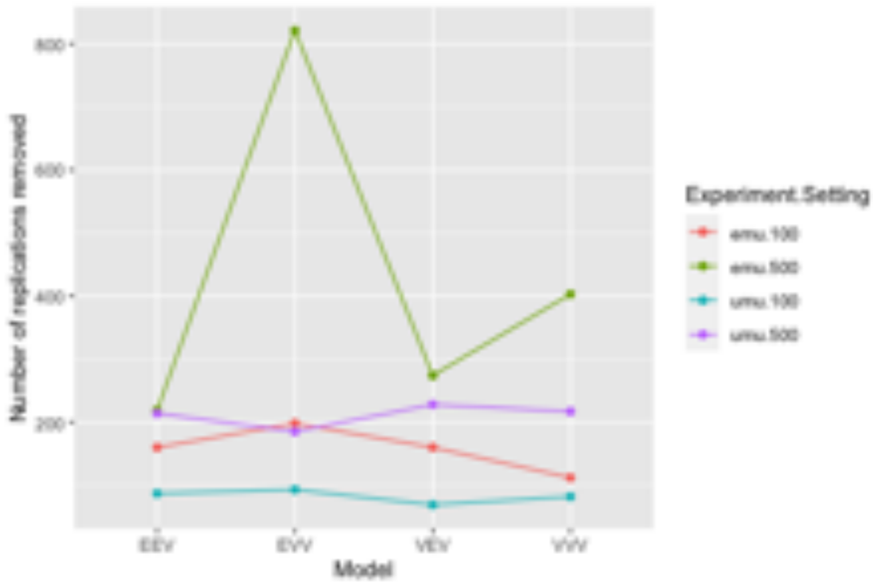


Figure 3.1.3: The number of replications removed from the GMM simulation to reach 100 replications

---

E step (Calculate the expected complete data log-likelihood)

$r_{ik}^{(update)} \leftarrow \frac{\pi_k^{(update)}\mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k^{(update)},\boldsymbol{\Sigma}_k^{(update)})}{\sum_{j=1}^{K}\pi_j^{(update)}\mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_j^{(update)},\boldsymbol{\Sigma}_j^{(update)})}$ for $i = 1, ..., n$ and $k = 1, ..., K$.

CM step (Maximize the expected complete data log-likelihood)

$\boldsymbol{\Sigma}_k^{(update)} = \frac{\sum_{i=1}^{n} r_{ik}^{(update)}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{(update)})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{(update)})^T}{\sum_{i=1}^{n} r_{ik}^{(update)}}$ for $k = 1, ..., K$

end if

$\pi_k^{(t)} \leftarrow \pi_k^{(update)}, \boldsymbol{\mu}_k^{(t)} \leftarrow \boldsymbol{\mu}_k^{(update)}, \boldsymbol{\Sigma}_k^{(t)} \leftarrow \boldsymbol{\Sigma}_k^{(update)}$ for $k = 1, ..., K$

$\boldsymbol{\eta}^{(t)} \leftarrow \{\pi_1^{(t)}, ..., \pi_K^{(t)}, \boldsymbol{\mu}_1^{(t)}, ..., \boldsymbol{\mu}_K^{(t)}, \boldsymbol{\Sigma}_1^{(t)}, ..., \boldsymbol{\Sigma}_{\boldsymbol{K}}^{(t)}\}$

end while

$\hat{\boldsymbol{\eta}} \leftarrow \{\pi_1^{(t)}, ..., \pi_K^{(t)}, \boldsymbol{\mu}_1^{(t)}, ..., \boldsymbol{\mu}_K^{(t)}, \boldsymbol{\Sigma}_1^{(t)}, ..., \boldsymbol{\Sigma}_{\boldsymbol{K}}^{(t)}\}$

Step 5: Remove the replication results from spurious solutions

In the Section 2.5.2, we have proposed a solution to tackle the issues of singularity and cluster collapse. When the condition from Equation (2.5.5) or (2.5.6) is fulfilled, the replication is stopped and removed immediately and start a new one, until 100 valid replications are completed.

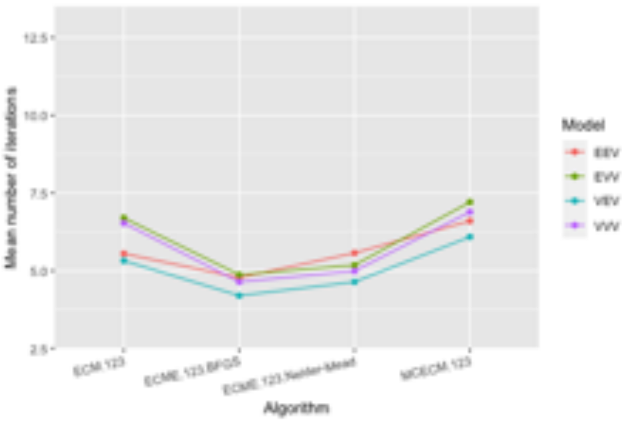
Figure 3.1.3 is a line plot showing the total number of replications removed from the GMM simulation to reach 100 replications. We can see that, comparing the same number of samples, equal mean experiment setting leads to spurious solutions more than different mean experiment setting. For the equal mean experiment setting, it is more difficult to separate the simulated data into 2 clusters. Therefore, the clusters collapse issue introduced in the Section 2.5.2 happens easily, so that all simulated data tend to be classified into 1 cluster only. Also, comparing the same mean (emu) or different mean (umu), a sample size of $n = 500$ leads more often spurious results compared to $n = 100$. Last but not least, for the experiment setting emu.500, there are about 400 spurious solutions that the sample generated from VVV and 800 spurious solutions that the sample generated from EVV.

3.1.2 Result visualisation and the performance

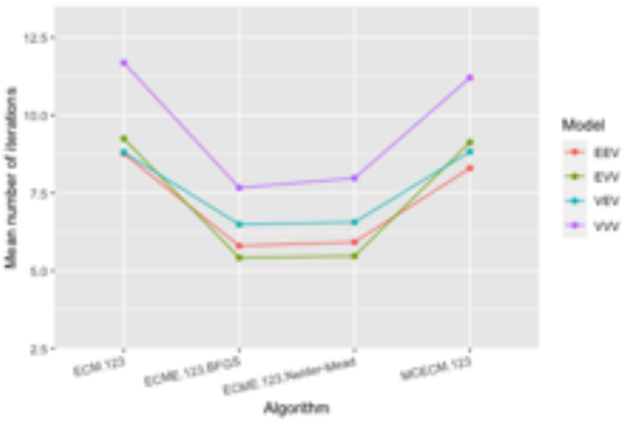
Step 6: Create visualization of the performance measures
The graphs for result visualization are shown in Step 7.

Step 7: Analyse and compare the results.
The output of the algorithm shown above is not only the estimated parameters, but also other performance measures, for example mean number of iterations, mean computation time (seconds), mean log-likelihood, mean BIC, mean classification rate. In order to analyse the result, it is more convenient for us to show the results for each algorithm in a graph, for example, line-plot, box-plot to compare the result with regard to the mean or median. The discussion of the result is in the following:

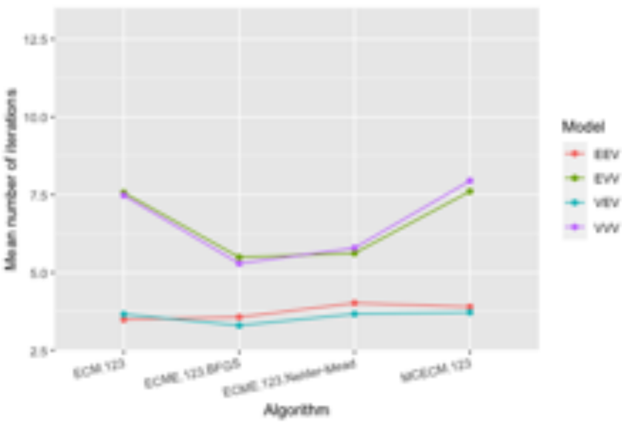
1. Mean number of iterations
We evaluate the clustering performance about iteration of the selected algorithms by visualizing the mean number of iterations and mean total computation time per simulation replication in line plots. For 100 observations, Figure 3.1.4 (a) and 3.1.4 (b) show that among the selected algorithms, ECME requires iterations for each type of Gaussian clustering models. compared to ECME, ECM and MCECM require more iterations. The experiment result proves the following statement that ECME can have a substantially faster convergence rate than either EM or ECM, measured using the number of iterations [Liu and Rubin, 1994]. Secondly, we expect that the mean number of iteration for MCECM should be lower than ECM, because Meng and Rubin [1993] mentioned that performing an E-step before each CM-step may result in larger increases in likelihood function L per iteration since Q is being updated more often. However, according to our experiment result, MCECM requires similar number of iterations like ECM. Last but not least, in the Figure 3.1.4, we can also see that for each type of the Gaussian clustering models, scenarios with different means require more numbers of iterations than scenarios with same cluster means. The reason behind is that it is quite difficult to separate the clusters in the scenario of same means. The log-likelihood cannot be improved so much by updating the parameters in each step. Therefore, the mean number of iteration in the case of sme means is less than the case of different means. For 500 observations, Figure 3.1.4 (c) and 3.1.4 (d) show that the result is quite similar to 100 observations. However, the needed mean number of iteration for EEV and VEV is almost the same around 3 to 4 times for all algorithms.



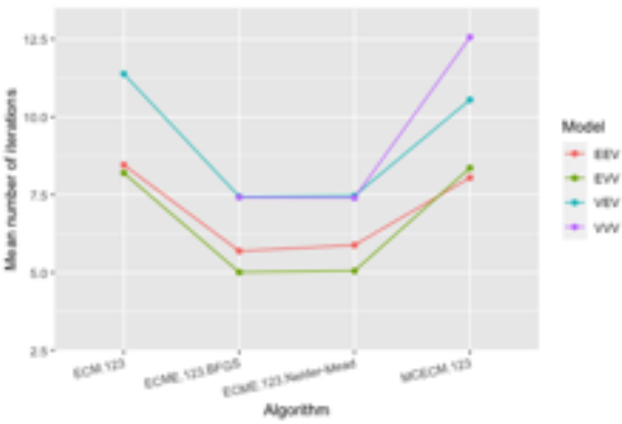
(a) $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2, n = 100$



(b) $\boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2, n = 100$

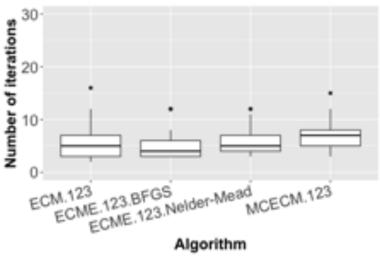


(c) $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2, n = 500$

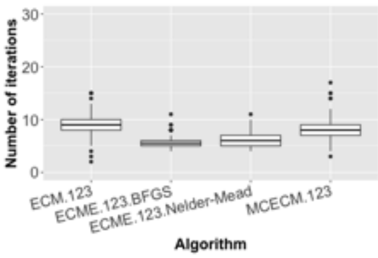


(d) $\boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2, n = 500$

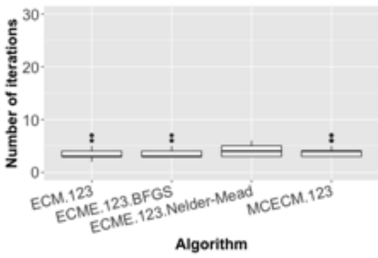
Figure 3.1.4: The mean number of iterations for the EM algorithms over 100 replications with $n = 100$ in the Figure (a) and (b) and $n = 500$ in the Figure (c) and (d) generated from different types of the Gaussian clustering models.



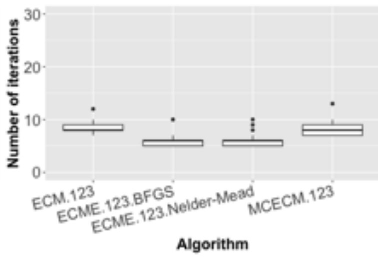
(a) EEV.emu.100



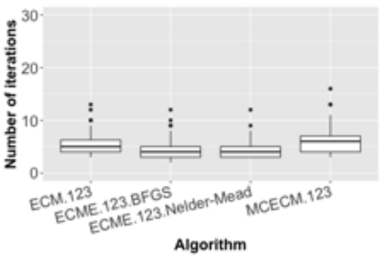
(b) EEV.umu.100



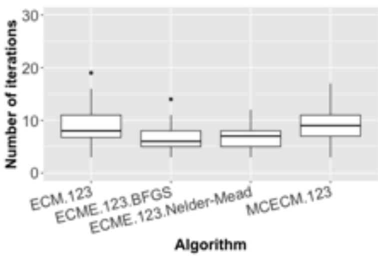
(c) EEV.emu.500



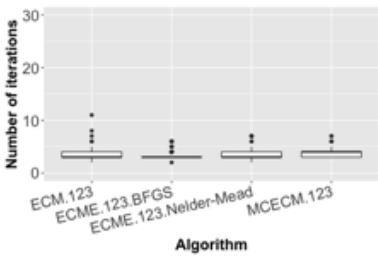
(d) EEV.umu.500



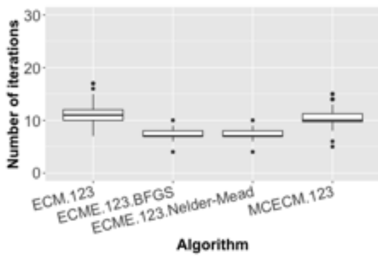
(e) VEV.emu.100



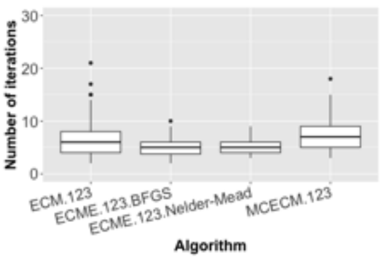
(f) VEV.umu.100



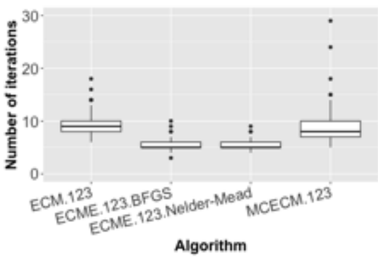
(g) VEV.emu.500



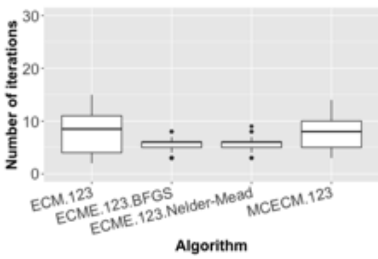
(h) VEV.umu.500



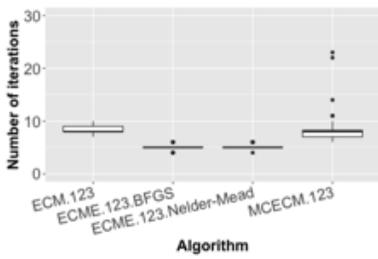
(i) EVV.emu.100



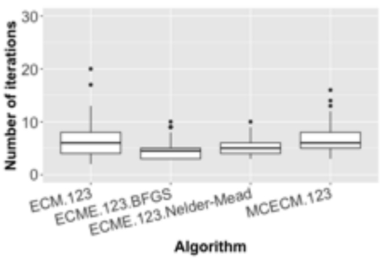
(j) EVV.umu.100



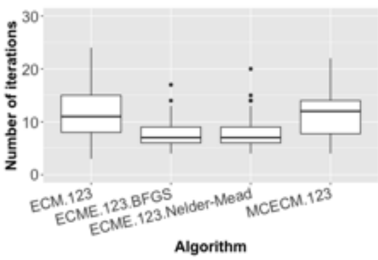
(k) EVV.emu.500



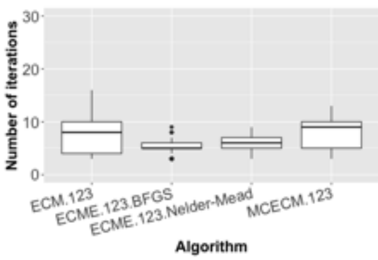
(l) EVV.umu.500



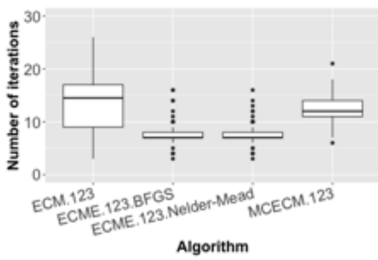
(m) VVV.emu.100



(n) VVV.umu.100



(o) VVV.emu.500

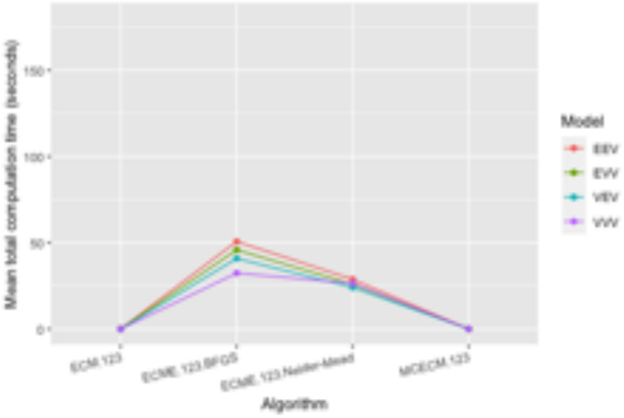


(p) VVV.umu.500

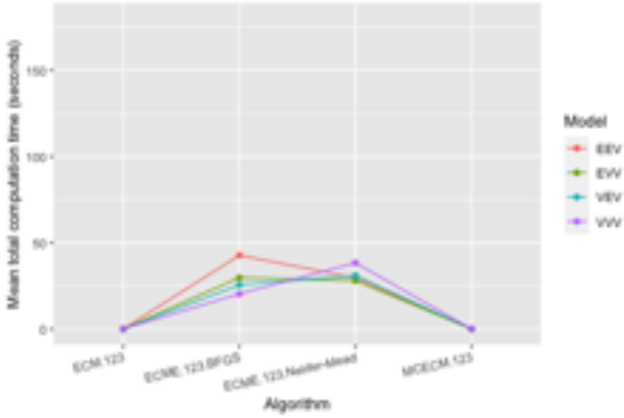
Figure 3.1.5: Box plots of the number of iterations for the EM algorithms over 100 replications with sample size $n = 100$ and 500 generated from different types of the Gaussian clustering models.

2. Mean total computation time

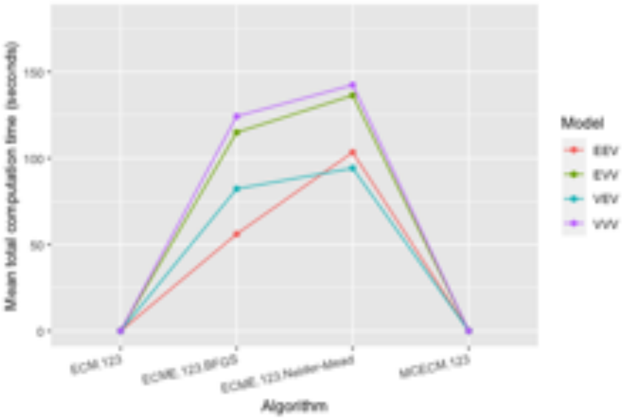
About the mean total computation time, Figure 3.1.6 shows that among the selected algorithms, ECM and MCECM require the shortest time, just less than 0.04 seconds and ECME requires much longer time, at least about 25 seconds. It makes sense because ECME needs to implement the optimization numerically and ECM and MCECM have the analytic to maximize the Q in the E step. Also, compared 100 to 500 observations, it is obvious to see that, for ECME, the required mean total computation time for 500 observations is much higher than 100 observations. For ECM and MCECM, the required time remains similar. The reason is basically the same as mentioned before.



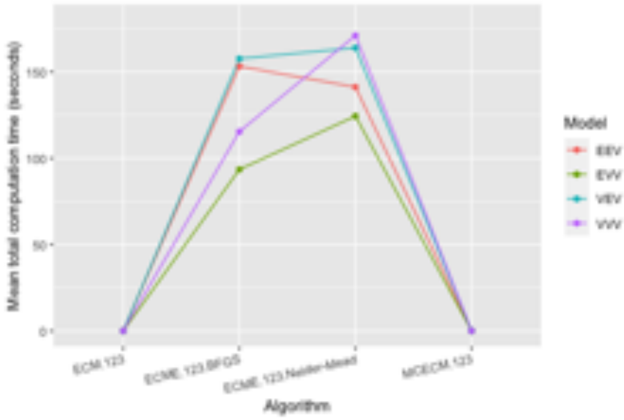
(a) $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2, n = 100$



(b) $\boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2, n = 100$

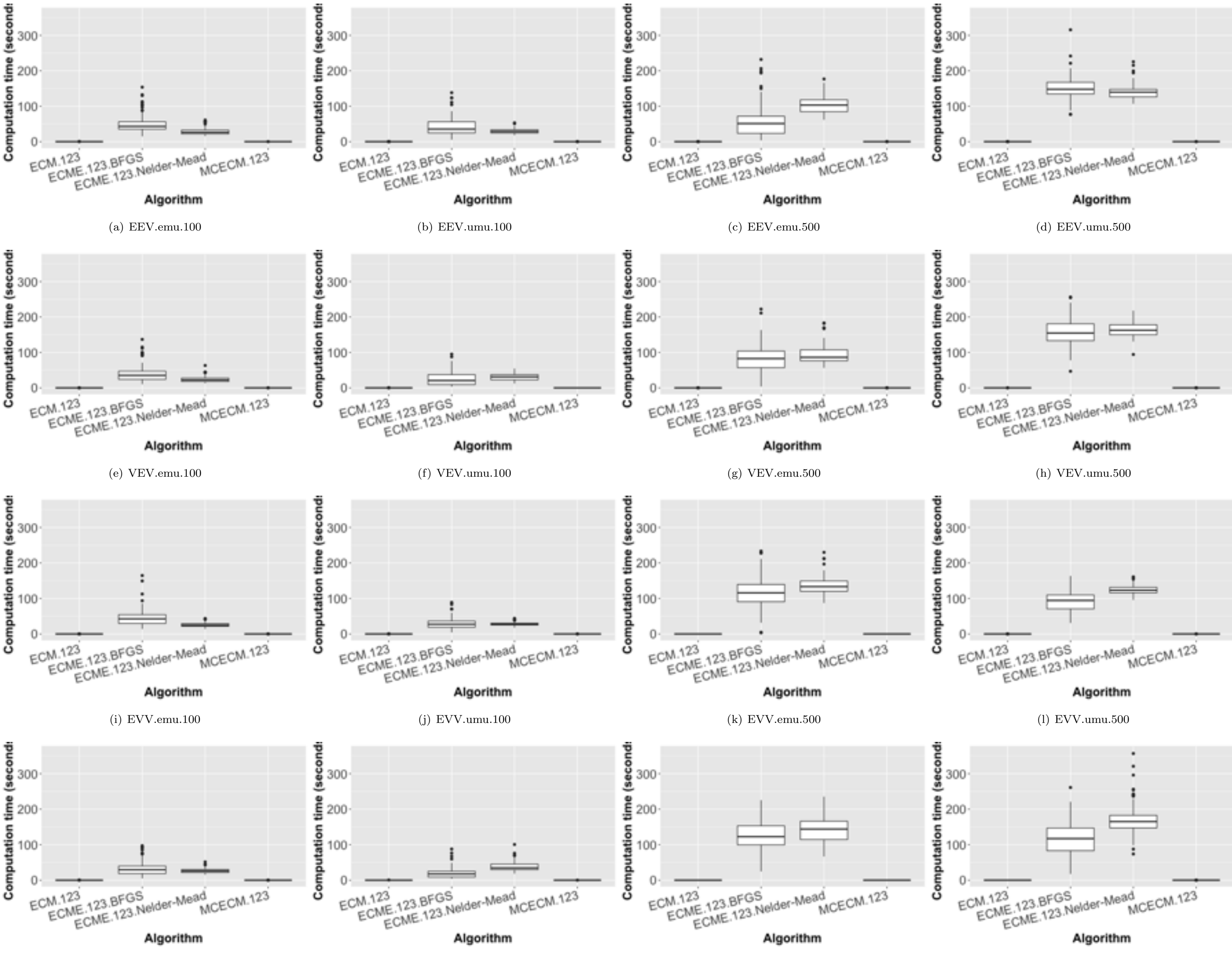


(c) $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2, n = 500$



(d) $\boldsymbol{\mu}_1 \neq \boldsymbol{\mu}_2, n = 500$

Figure 3.1.6: The mean total computation time (seconds) of EM algorithms over 100 replications with $n = 100$ in the Figure (a) and (b) and $n = 500$ in the Figure (c) and (d) generated from different types of the Gaussian clustering models.



(a) EEV.emu.100 (b) EEV.umu.100 (c) EEV.emu.500 (d) EEV.umu.500

(e) VEV.emu.100 (f) VEV.umu.100 (g) VEV.emu.500 (h) VEV.umu.500

(i) EVV.emu.100 (j) EVV.umu.100 (k) EVV.emu.500 (l) EVV.umu.500

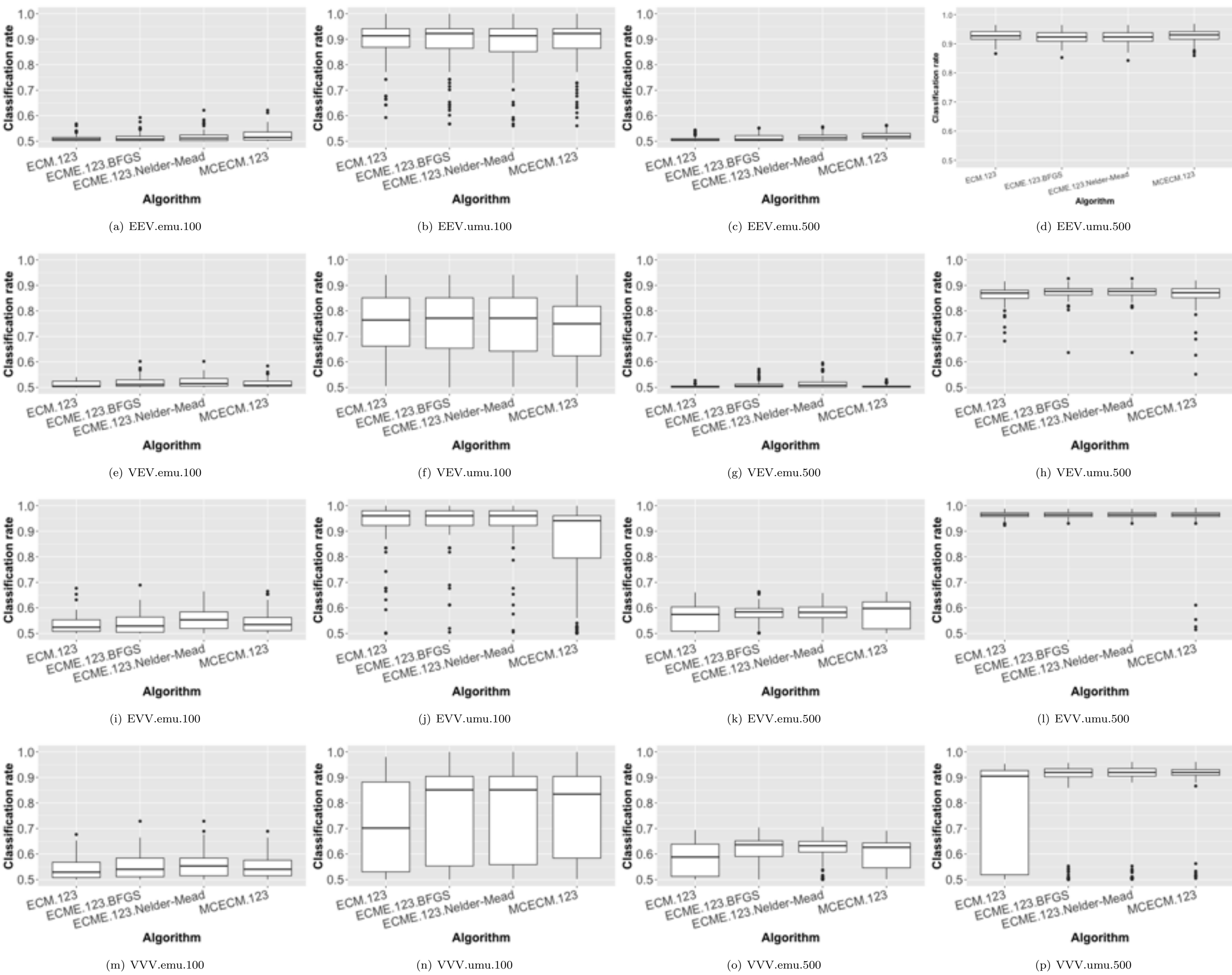
(m) VVV.emu.100 (n) VVV.umu.100 (o) VVV.emu.500 (p) VVV.umu.500

Figure 3.1.7: Box plots of the **computation time** (seconds) of the EM algorithms over 100 replications with sample size $n = 100$ and 500 generated from different types of the Gaussian clustering models.

3. Classification rate

The results of the classification rate per simulation are shown by using a box plot. For 100 and 500 observations, Figure 3.1.8 show that the median of classification rate in the scenario of different mean is obviously higher than the scenario of same mean. It makes a lot of sense because in the Figure 3.1.1 and 3.1.2, the level of overlapping between two clusters for the same mean is higher than different mean. Also, we expected that the classification rate is higher in EVV and VVV, but the result here doesn't show that.

For 100 observations, for the scenario of same mean, ECME.Nelder-Mead has the highest median of classification rate in VEV, EVV and VVV, and the second highest median of classification rate in EEV. Oppositely, ECM has the lowest median of classification rate in all 4 Gaussain clustering models. For the scenario of different mean, it seems that ECME.BFGS has the best performance because it leads to the highest classification rate among all 4 algorithms. However, there is not a clear pattern which algorithm is the worst for classification rate, because for each scenario, the worst algorithm is different. Last but not least, except EEV.emu and VEV.emu, the classification rate for 500 observations is higher than for 100 observations. According to the Figure 3.1.1 and 3.1.2, we can see that other than EEV.emu and VEV.emu, there are more samples for 500 observations in non-overlapping area than for 100 observations. Therefore, this is the reason why the classification rate improves in 500 observations.



(a) EEV.emu.100 (b) EEV.umu.100 (c) EEV.emu.500 (d) EEV.umu.500

(e) VEV.emu.100 (f) VEV.umu.100 (g) VEV.emu.500 (h) VEV.umu.500

(i) EVV.emu.100 (j) EVV.umu.100 (k) EVV.emu.500 (l) EVV.umu.500

(m) VVV.emu.100 (n) VVV.umu.100 (o) VVV.emu.500 (p) VVV.umu.500

Figure 3.1.8: Box plots of the **classification rate** for the EM algorithms over 100 replications with sample size $n = 100$ and 500 generated from different types of the Gaussian clustering models.

4. Normalised BIC

About the value of the normalised BIC and for 100 observations, the Figure 3.1.9 shows that for the scenario of same mean, the median value of the BIC for ECME.Nelder-Mead is the lowest and for ECM is the highest in all 4 Gaussian clustering models. Theoretically, the model with the lowest BIC or higher classification rate is preferred. Therefore, the BIC value and the classification rate shown in the Figure 3.1.8 and 3.1.9 match this model selection rule basically, that the algorithms with higher classification rate have lower BIC and vice versa. For the scenario of different mean, ECME.Nelder-Mead seems to have the lowest BIC and MCECM has the highest BIC in all 4 Gaussian clustering models.

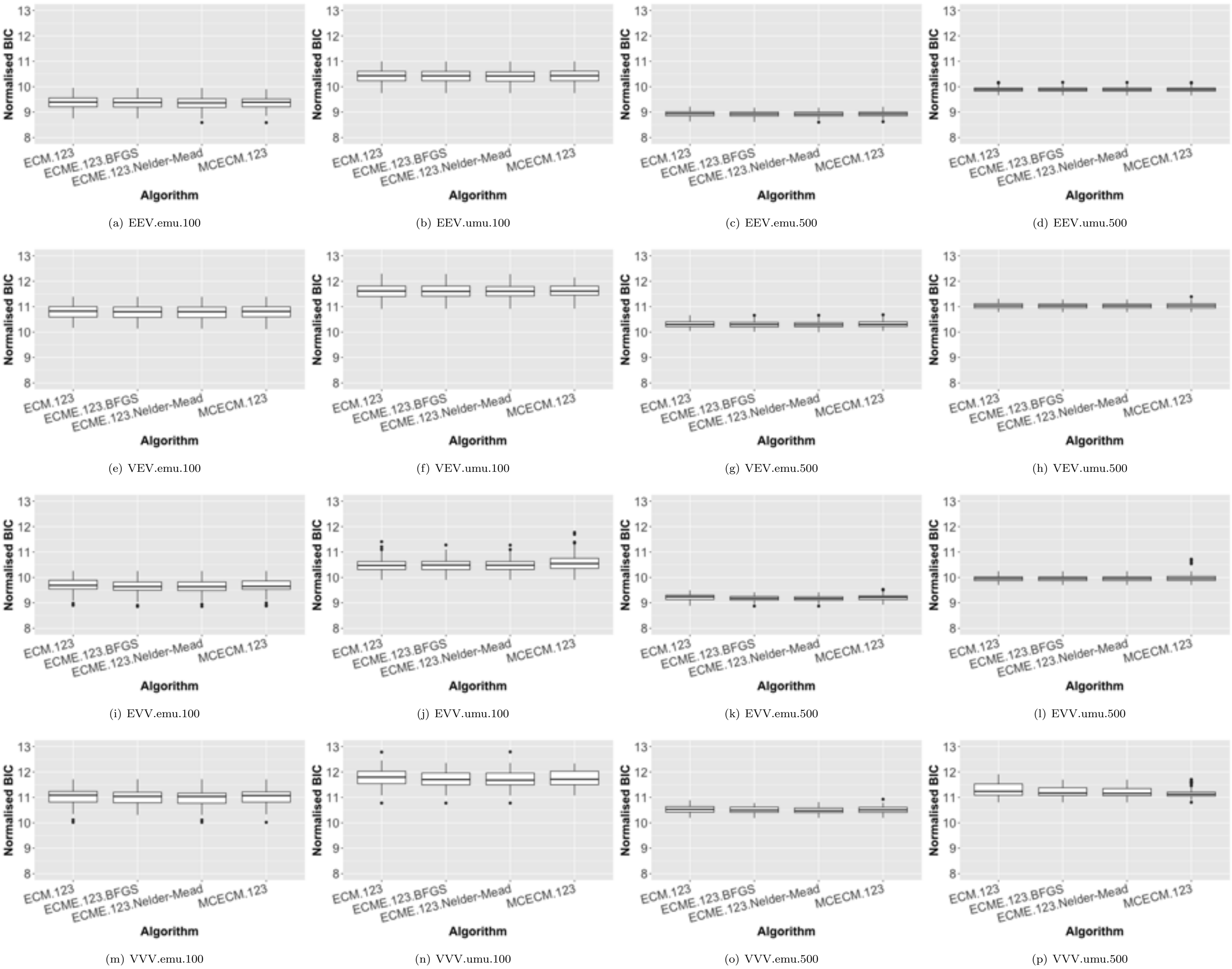


Figure 3.1.9: Box plots of the **normalised BIC** of the EM algorithms over 100 replications with sample size $n = 100$ and 500 generated from different types of the Gaussian clustering models.

Ranking of the performance for each EM algorithm

The Figure 3.1.7 shows the ranking of the EM algorithms regarding the performance for GMM algorithm. In this paper, we focus on clustering more, so we regard the higher mean classification rate as better performance. If the mean classification rate is the same, the lower mean normalised BIC and shorter mean computation time are preferred. For each scenario, we can see that the mean classification rate and mean normalised BIC are mostly very similar among different EM algorithms, but the mean computation time varies as we mentioned before. In particular, in EVV.umu.100, VVV.umu.100 and VVV.umu.500, MCECM, ECM and ECM perform the worst respectively and their classification rates are at least 5% less than the third ranking methods. Because we don't know which GMM data being clustered in the real situation, for better or more stable clustering performance, we would suggest using ECME.Nelder.Mead and ECME.BFGS for clustering with GMM algorithm.

Table 3.1.7: The ranking of the EM algorithms regarding the performance for GMM algorithm, where the algorithm with 1) higher mean classification rate, 2) lower mean normalised BIC and 3) shorter mean computation time for 100 replications is considered to be the better performance and the priority is given by (1) > (2) > (3).

Code of experiment	Ranking	Algorithm	Classification rate	Normalised BIC	Computation time (Seconds)	Code of experiment	Ranking	Algorithm	Classification rate	Normalised BIC	Computation time (Seconds)
EEV.emu.100	1	MCECM	0.52	9.4	0.023	EEV.emu.500	1	MCECM	0.52	8.9	0.0226
	2	ECM	0.51	9.4	0.0191		2	ECM	0.51	8.9	0.0121
	3	ECME.Nelder.Mead	0.51	9.4	28.8		3	ECME.BFGS	0.51	8.9	56.1
	4	ECME.BFGS	0.51	9.4	50.7		4	ECME.Nelder.Mead	0.51	8.9	103
EEV.umu.100	1	ECM	0.89	10.4	0.0211	EEV.umu.500	1	ECM	0.93	9.9	0.0276
	2	MCECM	0.88	10.4	0.0288		2	MCECM	0.93	9.9	0.0449
	3	ECME.Nelder.Mead	0.88	10.4	29.6		3	ECME.Nelder.Mead	0.92	9.9	141
	4	ECME.BFGS	0.88	10.4	42.7		4	ECME.BFGS	0.92	9.9	153
VEV.emu.100	1	ECM	0.51	10.8	0.013	VEV.emu.500	1	ECME.BFGS	0.51	10.3	82.3
	2	MCECM	0.51	10.8	0.0222		2	ECME.Nelder.Mead	0.51	10.3	94.2
	3	ECME.Nelder.Mead	0.51	10.8	24.1		3	ECM	0.50	10.3	0.012
	4	ECME.BFGS	0.51	10.8	40.9		4	MCECM	0.50	10.3	0.0205
VEV.umu.100	1	ECM	0.75	11.6	0.027	VEV.umu.500	1	ECME.BFGS	0.87	11.0	158
	2	ECME.BFGS	0.75	11.6	25.6		2	ECME.Nelder.Mead	0.87	11.0	164
	3	ECME.Nelder.Mead	0.74	11.6	31.1		3	ECM	0.86	11.0	0.0374
	4	MCECM	0.72	11.6	0.0315		4	MCECM	0.86	11.0	0.0597
EVV.emu.100	1	ECME.Nelder.Mead	0.55	9.6	25.9	EVV.emu.500	1	ECME.BFGS	0.58	9.2	115
	2	MCECM	0.53	9.7	0.0236		2	ECME.Nelder.Mead	0.58	9.2	136
	3	ECME.BFGS	0.53	9.7	45.9		3	MCECM	0.57	9.2	0.0461
	4	ECM	0.52	9.7	0.0155		4	ECM	0.55	9.2	0.0276
EVV.umu.100	1	ECM	0.93	10.5	0.0314	EVV.umu.500	1	ECM	0.96	10.0	0.0272
	2	ECME.Nelder.Mead	0.93	10.5	28		2	ECME.BFGS	0.96	10.0	93.5
	3	ECME.BFGS	0.93	10.5	29.7		3	ECME.Nelder.Mead	0.96	10.0	124
	4	MCECM	0.84	10.6	0.032		4	MCECM	0.95	10.0	0.0478
VVV.emu.100	1	MCECM	0.55	11.0	0.0247	VVV.emu.500	1	ECME.BFGS	0.61	10.5	124
	2	ECME.Nelder.Mead	0.55	11.0	26.5		2	ECME.Nelder.Mead	0.61	10.5	142
	3	ECME.BFGS	0.54	11.0	32.4		3	MCECM	0.60	10.5	0.0454
	4	ECM	0.53	11.0	0.0218		4	ECM	0.58	10.5	0.0247
VVV.umu.100	1	MCECM	0.74	11.7	0.0389	VVV.umu.500	1	MCECM	0.89	11.1	0.0752
	2	ECME.BFGS	0.74	11.7	20.4		2	ECME.Nelder.Mead	0.83	11.2	171
	3	ECME.Nelder.Mead	0.74	11.7	38.3		3	ECME.BFGS	0.82	11.2	115
	4	ECM	0.69	11.8	0.0339		4	ECM	0.75	11.3	0.0448

3.2 EM Algorithm and initialization strategies for GMM algorithm

In the previous Section 3.1, the simulation is done by a particular and fixed starting value. As log-likelihood function for mixture model is a non-convex optimization problem, the point to which the EM algorithm converges depends on the initial value [Jin et al., 2016]. Therefore, a poor set of starting values for the EM algorithm can significantly impact the the quality of the resulting solution [Shireman et al., 2017]. In this section, we will try to use different initialization strategies to perform our simulation for different GMM models.

The basic set up of the experiment is almost the same as the last simulation, but we just implement EEV.umu.500, VEV.umu.500, EVV.umu.500 and VVV.umu.500 this time, which is the experiment setting - Unequal mean of 2 clusters and $n = 500$ generated samples. The parameters of the GMM for generating samples is the same as before shown in the Table 3 in the last section.

About the starting value of the EM algorithm, we will try "True" and 5 different initialization strategies. "True" means using the true parameters for generating samples as the starting values, so that we can know what is the best performance we can have through different EM algorithms. For initialization strategies, we use 5 strategies, called "Random", "Kmeans", "Hierarchical", "Kmeans (scale)" and "Hierarchical (scale)", to decide the starting values. Basically, all the initialization strategies separate the generated samples into two groups first. Then, the starting value of mean and covariance of each cluster are the sample mean and the sample covariance of each cluster respectively. "Kmeans" is a well-known K-Means Clustering [Hartigan and Wong, 1979]. "Hierarchical" is an agglomerative hierarchical clustering based on the parameterization of VVV model [Fraley, 1998]. Also, the methods with "(scale)" means that the data is applied to the function `scale` in R for standardization before application of the initialization strategies. The Table 3.2 summarises the used initialization strategies.

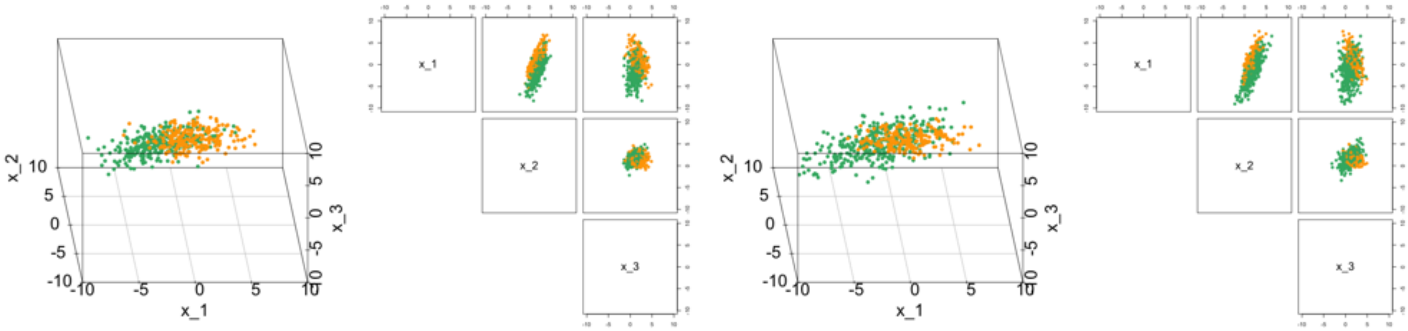
Table 3.2.1: The summary of the initialization strategies used.

Initialization strategies	R package	R function	Brief description
Random	-	-	Separate the samples into 2 groups with the same number randomly.
Kmeans	`stat`	`kmeans`	Minimize a Euclidean distance between the cluster center and the samples of the cluster.
Hierarchical	`mclust`	`hcVVV`	Agglomerative is a "bottom-up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy, based on the maximum likelihood for a VVV model.
Kmeans (scale)	`stat` -	`kmeans` `scale`	Data is applied to the function `scale` for standardization before application of the function `kmeans`.
Hierarchical (scale)	`mclust` -	`hcVVV` `scale`	Data is applied to the function `scale` for standardization before application of the function `hcVVV`.

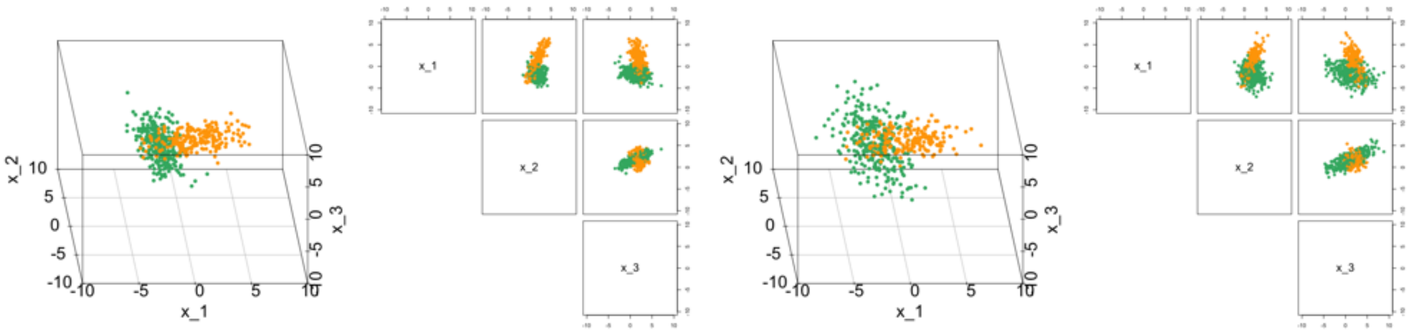
Example 3.2.1 (Visualization of 500 simulated data set generated from the Gaussian mixture model, the initialization clustering by different initialization strategies)

The Figure 3.2.1 shows the true clustering of the scatter plots for each data setting. As mentioned before, if the scatters are not coloured, it is quite hard to observe there are 2 clusters in EEV and VEV. Oppositely, it is clearer to see there are 2 clusters in EVV and VVV. Therefore, we can divide them into two groups, such

True cluster



(a) **EEV.umu.500** (b) **EEV.umu.500** (c) **VEV.umu.500** (d) **VEV.umu.500**



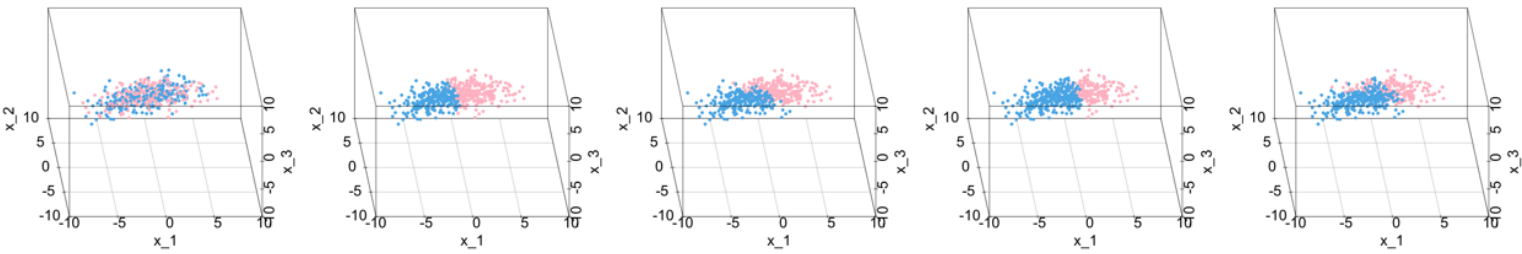
(e) **EVV.umu.500** (f) **EVV.umu.500** (g) **VVV.umu.500** (h) **VVV.umu.500**

Figure 3.2.1: The scatter plots and the corresponding pairs plots of 500 simulated data generated from the Gaussian Mixture Model for model types: EEV, VEV, EVV and VVV. The orange and green colour refer to the points in Cluster 1 and 2 respectively.

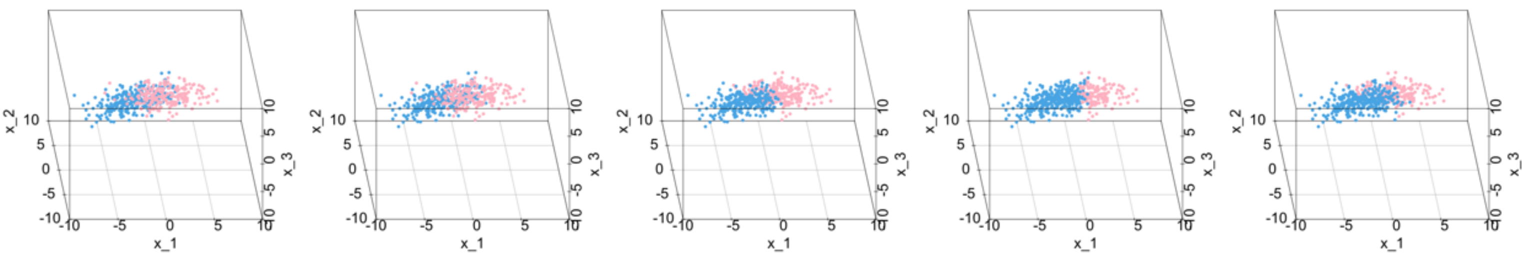
that 1. Better separated (EVV and VVV) and 2. Worse separated (EEV and VEV).

Because we use different initialisation strategies in the GMM algorithm, the visualisations of the 500 generated data after initialization clustering by different strategies are shown also in the Figure 3.2.2 and Figure 3.2.3. We can see that all initialisation strategies give us different result of initialisation clustering. Also, from the scatter plots of initializing clustering in the Figure 3.2.2 and Figure 3.2.3, we know that the true cluster is with some level of overlapping, but apart from the "random" case, the scatter plots for all other initialisation strategies do not show any level of the overlapping. It implies that "Kmeans", "Hierarchical" and their "(scale)" strategies cannot classify the overlapping scatters.

EVV.umu.500

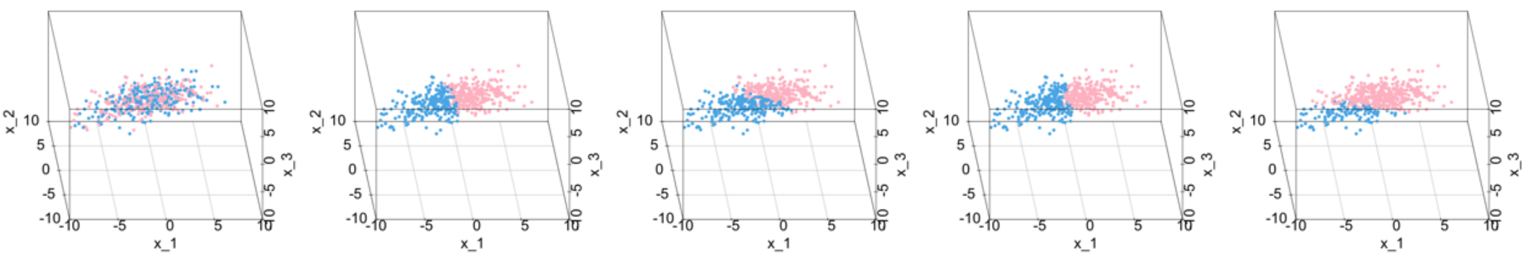


(a) **Random (50.0%)** (b) **Kmeans (60.5%)** (c) **Kmeans scale (53.4%)** (d) **Hierarchical (57.9%)** (e) **Hierarchical scale (51.5%)**

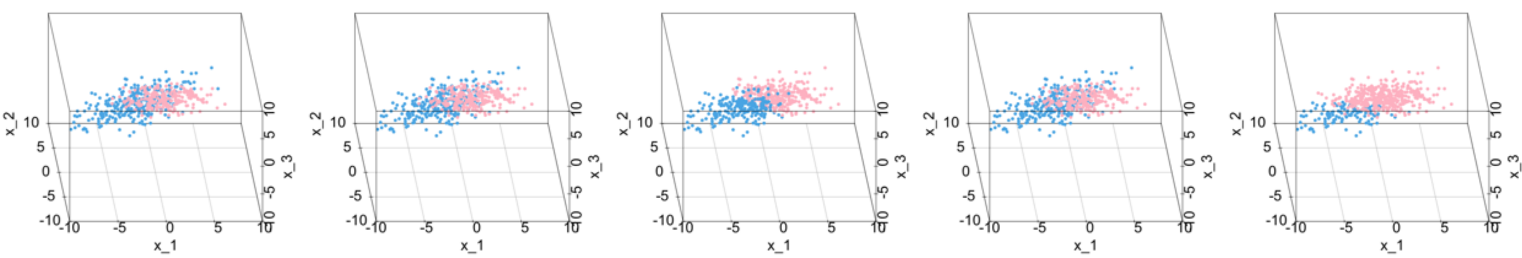


(f) **Random (91.9%)** (g) **Kmeans (92.3%)** (h) **Kmeans scale (52.9%)** (i) **Hierarchical (58.4%)** (j) **Hierarchical scale (51.5%)**

VEV.umu.500



(k) **Random (49.9%)** (l) **Kmeans (55.3%)** (m) **Kmeans scale (50.5%)** (n) **Hierarchical (55.8%)** (o) **Hierarchical scale (52.2%)**



(p) **Random (82.3%)** (q) **Kmeans (78.2%)** (r) **Kmeans scale (50.6%)** (s) **Hierarchical (77.9%)** (t) **Hierarchical scale (52.2%)**

Figure 3.2.2: The scatter plots of initializing clustering and final clustering by using different initialization strategies for model type **EEV**, Figure 3.2.1 (a) and **VEV**, Figure 3.2.1 (c). The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. First row : EEV.umu.500 (Initializing clustering), second row : EEV.umu.500 (Final clustering), third row : VEV.umu.500 (Initializing clustering) and fourth row : VEV.umu.500 (Final clustering).

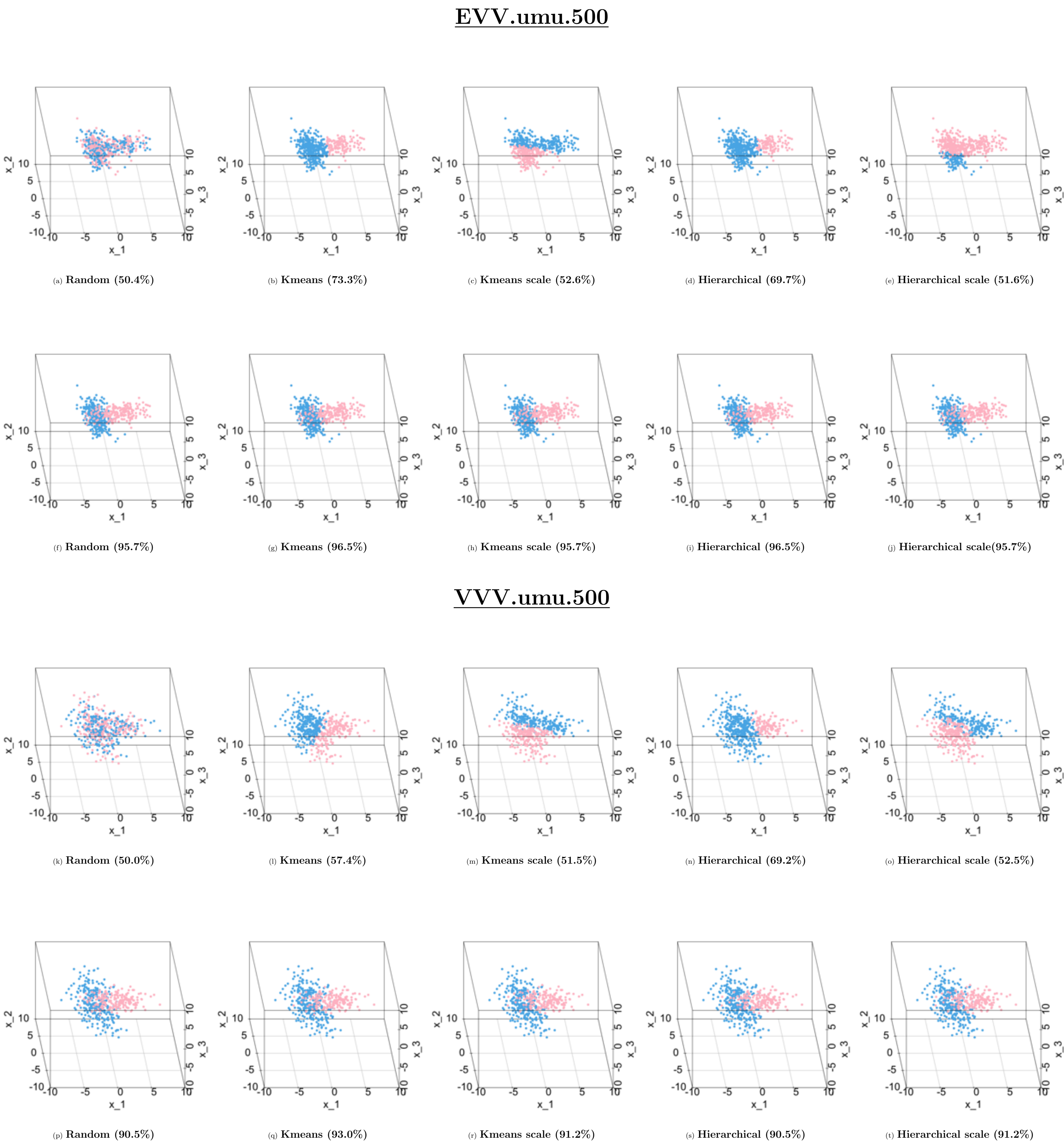


Figure 3.2.3: The scatter plots of initializing clustering and final clustering by using different initialization strategies for model type **EVV**, Figure 3.2.1 (e) and **VVV**, Figure 3.2.1 (g). The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. First row : EVV.umu.500 (Initializing clustering), second row : EVV.umu.500 (Final clustering), third row : VVV.umu.500 (Initializing clustering) and fourth row : VVV.umu.500 (Final clustering).

Discussion of the result

1. Scatter plots after initializing and final clustering

The Figure 3.2.2 and Figure 3.2.3 also show the scatter plots with the classification rate after final clustering. You can see that for a particular initialization strategy, the classification rate increases after final clustering, which means the GMM algorithm improves the clustering performance after using initialization strategy.

2. Number of replications removed for ECME.BFGS

The Table 3.2.2 shows the total number of replications removed of ECME.BFGS algorithm with different initialization strategies. The strategies with "(scale)" lead to obviously lower replications removed than the strategies without "(scale)". Also, for each strategies, EVV leads to the most conditions from Equation (2.5.5) or (2.5.6) obviously and there are over 300 removed replications for EVV and the corresponding strategy "Kmeans" and "Hierarchical". For the algorithm with over 300 removed replications, the simulation cannot be finished, so there is no performance measures information for EVV and "Kmeans" and "Hierarchical". Note that there are no any replications removed for other EM algorithm.

The result of the performance measures, including mean computation time, mean normalised BIC and mean classification rate, for each EM algorithm and its extension with different initialization strategies is shown in the Table 3.2.3. A simple analysis is given in the following:

3. Mean computation time

The result is quite regular. For each initialization strategy, the mean total computation time needed in descending order is in the following : ECME.Nelder.Mead > ECME.BFGS > MCECM > ECM. The reason is in the following : 1) There are analytical solutions in ECM and MCECM and ECME involves solving the optimization problem numerically. So, ECME takes more time. 2) MCECM requires E step before each M step and ECM just require a step in each iteration. So, MCECM takes more time than ECM. 3) ECME.BFGS is a gradient based numerical method and the gradient function is given to the R code. ECME.Nelder.Mead takes longer time than ECME.BFGS.

4. Mean normalised BIC

We can see that the difference of mean normalised BIC among each algorithm is quite small. I think this results from the normalisation of the BIC. However, you can also observe that a small change in normalised BIC lead to a bigger difference in classification rate. Also, recalling that the algorithms with the lowest normalised BIC is adopted. For each initialization strategies and model type, ECME.Nelder algorithm always leads to the lowest normalised BIC.

5. Mean classification rate

"Kmeans" outperforms all other strategies with the highest mean classification rate for EEV, VEV and EVV. Also, in VVV, it is quite unexpected that "Random" outperforms all other strategies for ECME quite a lot. Furthermore, for each initialization strategy, the ECME.Nelder algorithm outperforms the other EM algorithms for each data model type. Last but not least, the strategies Kmeans outperforms Kmeans (scale) and Hierarchical outperforms Hierarchical (scale) for all the data sets.

Table 3.2.2: The number of replications removed to reach 50 convergent replications by using ECME.BFGS algorithm with different initialization strategies, where the green and red colour represent the best and the worst initialization strategies for each EM algorithm respectively .

	Starting values	Initialization strategies				
	True	Random	Kmeans	Kmeans (scale)	Hierarchical	Hierarchical (scale)
EEV	47	22	120	44	89	48
VEV	34	26	200	68	132	49
EVV	44	182	>300	116	>300	218
VVV	30	101	152	52	214	117

Ranking of the performance for each EM algorithm and initialization strategy

The Table 3.2.4 shows the ranking of the performance of EM algorithms and the initialization strategies by GMM algorithm. The same as before, the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3).

Firstly, apart from data setting EVV, we can see that the mean classification rates vary with different EM algorithms and initialization strategies. Secondly, EM algorithms and the initialization strategies in the top 7 performance can perform better in EVV and VVV than in EEV and VEV. It varies our previous statement, that the data with better separation can be clustered more accurately with a suitable method. Thirdly, some methods always perform better for each data settings which are already highlighted in the table with different colours, for example, ECME.Nelder.Mead (Kmeans), ECME.Nelder.Mead (Hierarchical) and so on. This implies that, there are some clustering methods likely performing better, no matter how the characteristics of the data generated from different Gaussian mixture models.

According to the result from the four data settings, for the best fit, ECME.Nelder.Mead with Kmeans or ECME.Nelder.Mead with Hierarchical are recommended for clustering with GMM algorithm. For shorter computation time and good fit, MCECM with Kmeans is recommended.

Table 3.2.3: Mean value of the performance measures for each EM algorithm and its extension with different starting values and initialization strategies for 50 replications for GMM algorithm, where the green and red colour represent the best and the worst initialization strategies for each EM algorithm respectively.

Data Setting	Performance measures	Algorithm	Starting value	Initialization strategies				
			True	Random	Kmeans	Kmeans (scale)	Hierarchical	Hierarchical (scale)
EEV	Computation time	ECM	0.018	0.011	0.033	0.037	0.039	0.041
		ECME.BFGS	29	47	87	35	58	47
		ECME.Nelder.Mead	62	219	208	167	213	201
		MCECM	0.011	0.020	0.040	0.042	0.035	0.046
VEV	Computation time	ECM	0.007	0.007	0.036	0.039	0.032	0.015
		ECME.BFGS	41	36	165	74	134	55
		ECME.Nelder.Mead	63	117	261	245	261	179
		MCECM	0.011	0.013	0.056	0.045	0.049	0.027
EVV	Computation time	ECM	0.018	0.031	0.022	0.032	0.026	0.031
		ECME.BFGS	30	136	-	40	-	37
		ECME.Nelder.Mead	58	238	155	225	146	233
		MCECM	0.011	0.049	0.030	0.059	0.029	0.046
VVV	Computation time	ECM	0.006	0.018	0.035	0.022	0.022	0.021
		ECME.BFGS	36	132	83	41	56	31
		ECME.Nelder.Mead	59	268	197	203	198	222
		MCECM	0.011	0.038	0.044	0.030	0.041	0.040
EEV	Normalised BIC	ECM	9.9	10.3	10.1	10.3	10.2	10.2
		ECME.BFGS	9.9	10.3	10.2	10.3	10.3	10.3
		ECME.Nelder.Mead	9.9	10.2	10.0	10.2	10.1	10.1
		MCECM	9.9	10.3	10.1	10.2	10.1	10.2
VEV	Normalised BIC	ECM	11.0	11.3	11.0	11.2	11.1	11.2
		ECME.BFGS	11.0	11.3	11.0	11.2	11.1	11.3
		ECME.Nelder.Mead	11.0	11.2	11.0	11.1	11.0	11.1
		MCECM	11.0	11.3	11.0	11.1	11.0	11.2
EVV	Normalised BIC	ECM	9.9	10.2	9.9	10.2	9.9	10.1
		ECME.BFGS	9.9	10.0	-	10.5	-	10.5
		ECME.Nelder.Mead	9.9	10.0	9.9	10.1	9.9	10.0
		MCECM	9.9	10.1	9.9	10.1	9.9	10.1
VVV	Normalised BIC	ECM	11.1	11.6	11.4	11.5	11.3	11.5
		ECME.BFGS	11.1	11.2	11.3	11.6	11.5	11.6
		ECME.Nelder.Mead	11.1	11.2	11.2	11.4	11.2	11.4
		MCECM	11.1	11.4	11.3	11.5	11.2	11.4
EEV	Classication rate	ECM	0.94	0.59	0.74	0.58	0.70	0.67
		ECME.BFGS	0.94	0.60	0.68	0.58	0.60	0.55
		ECME.Nelder.Mead	0.94	0.72	0.82	0.64	0.78	0.69
		MCECM	0.94	0.63	0.76	0.60	0.72	0.68
VEV	Classication rate	ECM	0.88	0.56	0.76	0.64	0.73	0.56
		ECME.BFGS	0.88	0.55	0.75	0.61	0.68	0.56
		ECME.Nelder.Mead	0.88	0.62	0.80	0.72	0.79	0.64
		MCECM	0.88	0.55	0.79	0.69	0.78	0.58
EVV	Classication rate	ECM	0.96	0.84	0.96	0.78	0.96	0.84
		ECME.BFGS	0.96	0.96	-	0.59	-	0.58
		ECME.Nelder.Mead	0.97	0.95	0.96	0.88	0.96	0.91
		MCECM	0.97	0.88	0.96	0.82	0.96	0.88
VVV	Classication rate	ECM	0.93	0.69	0.74	0.64	0.82	0.66
		ECME.BFGS	0.93	0.87	0.74	0.56	0.59	0.54
		ECME.Nelder.Mead	0.93	0.92	0.84	0.72	0.89	0.74
		MCECM	0.93	0.76	0.82	0.64	0.86	0.69

Table 3.2.4: The top 7 performance of EM algorithms and the initialization strategies for GMM algorithm, where the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3). *The separation here refers to the level of separation whether the two clusters can be observed easily if the dots are not coloured or marked.

Data setting	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
EEV (Worse separation)	1	ECME.Nelder.Mead	Kmeans	0.82	10.0	208
	2	ECME.Nelder.Mead	Hierarchical	0.78	10.1	213
	3	MCECM	Kmeans	0.76	10.1	0.040
	4	ECM	Kmeans	0.74	10.1	0.33
	5	MCECM	Hierarchical	0.72	10.1	0.035
	6	ECME.Nelder.Mead	Random	0.72	10.2	219
	7	ECM	Hierarchical	0.70	10.2	0.039
VEV (Worse separation)	1	ECME.Nelder.Mead	Kmeans	0.80	11.0	261
	2	MCECM	Kmeans	0.79	11.0	0.056
	3	ECME.Nelder.Mead	Hierarchical	0.79	11.0	261
	4	MCECM	Hierarchical	0.78	11.0	0.049
	5	ECM	Kmeans	0.76	11.0	0.036
	6	ECM	Hierarchical	0.73	11.1	0.032
	7	ECME.Nelder.Mead	Kmeans scale	0.72	11.1	245
EVV (Better separation)	1	ECM	Kmeans	0.96	9.9	0.022
	2	ECM	Hierarchical	0.96	9.9	0.026
	3	MCECM	Hierarchical	0.96	9.9	0.029
	4	MCECM	Kmeans	0.96	9.9	0.030
	5	ECME.Nelder.Mead	Hierarchical	0.96	9.9	146
	6	ECME.Nelder.Mead	Kmeans	0.96	9.9	155
	7	ECME.BFGS	Random	0.96	10.0	136
VVV (Better separation)	1	ECME.Nelder.Mead	Random	0.92	11.2	268
	2	ECME.Nelder.Mead	Hierarchical	0.89	11.2	198
	3	ECME.BFGS	Random	0.87	11.2	132
	4	MCECM	Hierarchical	0.86	11.2	0.041
	5	ECME.Nelder.Mead	Kmeans	0.84	11.2	197
	6	ECM	Hierarchical	0.82	11.3	0.022
	7	MCECM	Kmeans	0.82	11.3	0.044

3.3 EM Algorithm and initialization strategies for VCMM algorithm

In Section 2.4.3, the formulation of the vine copula mixture model (VCMM) and the corresponding steps of EM algorithm and its extension are introduced. After the simulation studies for GMM, in this section, our main goal is to compare the performance of different EM algorithms and initialisation strategies for the algorithm of vine copula mixture model clustering (VCMM) proposed by Sahin and Czado [2021]. In that research paper, ECM algorithm is used for the parameter estimation only. In our simulation studies, basically, we will generate different data sets and then use the algorithm to implement classification. On top of that, the algorithm will be amended a bit for the assessment of different EM algorithms, numerical optimization methods and initialisation strategies, in order to assess the performance under different setting. More details will be given in the following steps:

Step 1: Generate data for simulation.
Step 2: Decide on the EM algorithms and the CM steps order in the VCMM algorithm.
Step 3: Decide on the initialization strategies for the VCMM algorithm.
Step 4: Decide on the marginal distribution and copula families for modelling in the VCMM algorithm.
Step 5: Perform the simulation in R.
Step 6: Create visualizations of the performance measures.
Step 7: Analyse and compare the results.

The detail for each step is given in the following subsection:

3.3.1 Data simulation and the experiment setup

Step 1: Generate data for simulation.

In this step, we would like to generate some data from the vine copula mixture model to test different EM algorithms. Compared to the Gaussian mixture model, the vine copula mixture model allows modelling the data much more flexible, as it allows different univariate marginal distribution and asymmetric dependency. Therefore, we would like to not only generate Gaussian data, but also generate the non-Gaussian data to assess the performance of different EM algorithms and initialization strategies under VCMM.

In our experiments, we generate data from four settings following the ways from Sahin and Czado [2021] in their section of simulation studies. In each of the setting, the dimension of each sample d is 3 and number of clusters K is 2. The Table 3.3.1 shows the basic set up for the data generation. Also, about the characteristics of the generated data set, data sets for setting 1 are generated from the mixture model of vine copulas with Non-Gaussian pair copulas and Non-Gaussian margins, data sets for setting 2, 8 and 9 are from the mixture model of vine copulas with Non-Gaussian pair copulas and Gaussian or Non-Gaussian margins, data sets for setting 3 are from the mixture model of vine copulas with Gaussian copulas and Gaussian margins and the data sets for setting 4, 5, 6 and 7 are from the mixture model of multivariate skew t distributions with different degrees of freedom. The overview of the nine settings are shown in the Table 3.3.2.

The generated data sets for settings 1, 2, 3, 8 and 9 actually follow the vine copula mixture model introduced in the Section 2.4.3. Recalling from Equation (2.4.17), a mixture model with 2 components and 3 dimensions has a density given by

$$f(\boldsymbol{x} \mid \boldsymbol{\eta}) = 0.4\ f_1(\boldsymbol{x} \mid \boldsymbol{\psi}_1) + 0.6\ f_2(\boldsymbol{x} \mid \boldsymbol{\psi}_2)$$

where the parameters $\boldsymbol{\eta}$ in the mixture model are denoted as $\boldsymbol{\eta} = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \pi_1, \pi_2\}$ and recalling from Equation

Table 3.3.1: The parameter values used in the nine settings of data simulation

Notation	Name	Value
n	Sample size	500 (For all settings) and 1000 (For setting 1, 3, 5, 6, 7)
d	Dimension of each sample	3
K	Total number of components	2
π_1	Mixing proportion of component 1	0.4
π_2	Mixing proportion of component 2	0.6

Table 3.3.2: The overview of the characteristics of the nine settings of data simulation

	Type of mixture model	Marginal distribution	Dependency relationship (Copula family)
Setting 1	VCMM	Non-Gaussian	Non-Gaussian
Setting 2	VCMM	Gaussian and Non-Gaussian	Non-Gaussian
Setting 3	VCMM (= Multivariate Gaussian)	Gaussian	Gaussian
Setting 4	Mixture of multivariate skew t	Skew t	-
Setting 5	Mixture of multivariate skew t	Skew t	-
Setting 6	Mixture of multivariate skew t	Skew t	-
Setting 7	Mixture of multivariate skew t	Skew t	-
Setting 8	VCMM	Gaussian and Non-Gaussian	Non-Gaussian
Setting 9	VCMM	Gaussian and Non-Gaussian	Non-Gaussian

(2.4.18), (2.4.19), the $f_1(\boldsymbol{x} \mid \boldsymbol{\psi}_1)$ and $f_2(\boldsymbol{x} \mid \boldsymbol{\psi}_2)$ for each data set are given in the following:

$$
\begin{aligned}
f_1(\boldsymbol{x} \mid \boldsymbol{\psi}_2) = {} & c_{(2)1,3;2}(F_{(2)1|2}(x_1|x_2;\boldsymbol{\gamma}_{1(2)},\boldsymbol{\gamma}_{2(2)},\boldsymbol{\theta}_{(2)1,2}), F_{(2)3|2}(x_3|x_2;\boldsymbol{\gamma}_{3(2)},\boldsymbol{\gamma}_{2(2)},\boldsymbol{\theta}_{(2)2,3});\boldsymbol{\theta}_{(2)1,3;2}) \\
& \times c_{(2)2,3}(F_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)}), F_{3(2)}(x_3;\boldsymbol{\gamma}_{3(2)});\boldsymbol{\theta}_{(2)2,3}) \\
& \times c_{(2)1,2}(F_{1(2)}(x_1;\boldsymbol{\gamma}_{1(2)}), F_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)});\boldsymbol{\theta}_{(2)1,2}) \\
& \times f_{3(2)}(x_3;\boldsymbol{\gamma}_{3(2)})\ f_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)})\ f_{1(2)}(x_1;\boldsymbol{\gamma}_{1(2)})
\end{aligned}
$$

<u>Setting 1, 8 and 9</u>

$$
\begin{aligned}
f_2(\boldsymbol{x} \mid \boldsymbol{\psi}_2) = {} & c_{(2)1,2;3}(F_{(2)1|3}(x_1|x_3;\boldsymbol{\gamma}_{1(2)},\boldsymbol{\gamma}_{3(2)},\boldsymbol{\theta}_{(2)1,3}), F_{(2)2|3}(x_2|x_3;\boldsymbol{\gamma}_{2(2)},\boldsymbol{\gamma}_{3(2)},\boldsymbol{\theta}_{(2)2,3});\boldsymbol{\theta}_{(2)1,2;3}) \\
& \times c_{(2)2,3}(F_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)}), F_{3(2)}(x_3;\boldsymbol{\gamma}_{3(2)});\boldsymbol{\theta}_{(2)2,3}) \\
& \times c_{(2)1,3}(F_{1(2)}(x_1;\boldsymbol{\gamma}_{1(2)}), F_{3(2)}(x_3;\boldsymbol{\gamma}_{3(2)});\boldsymbol{\theta}_{(2)1,3}) \\
& \times f_{3(2)}(x_3;\boldsymbol{\gamma}_{3(2)})\ f_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)})\ f_{1(2)}(x_1;\boldsymbol{\gamma}_{1(2)})
\end{aligned}
$$

<u>Setting 2 and 3</u>

$$
\begin{aligned}
f_2(\boldsymbol{x} \mid \boldsymbol{\psi}_2) = {} & c_{(2)1,3;2}(F_{(2)1|2}(x_1|x_2;\boldsymbol{\gamma}_{1(2)},\boldsymbol{\gamma}_{2(2)},\boldsymbol{\theta}_{(2)1,2}), F_{(2)3|2}(x_3|x_2;\boldsymbol{\gamma}_{3(2)},\boldsymbol{\gamma}_{2(2)},\boldsymbol{\theta}_{(2)2,3});\boldsymbol{\theta}_{(2)1,3;2}) \\
& \times c_{(2)2,3}(F_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)}), F_{3(2)}(x_3;\boldsymbol{\gamma}_{3(2)});\boldsymbol{\theta}_{(2)2,3}) \\
& \times c_{(2)1,2}(F_{1(2)}(x_1;\boldsymbol{\gamma}_{1(2)}), F_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)});\boldsymbol{\theta}_{(2)1,2}) \\
& \times f_{3(2)}(x_3;\boldsymbol{\gamma}_{3(2)})\ f_{2(2)}(x_2;\boldsymbol{\gamma}_{2(2)})\ f_{1(2)}(x_1;\boldsymbol{\gamma}_{1(2)})
\end{aligned}
$$

In particular, for setting 1, 2, 3, 8 and 9 of the marginal distribution F and the associated marginal parameters $\boldsymbol{\gamma}$ are shown in the Table 3.3.4. Also, the bivariate copula families c used and the associated copula parameters $\boldsymbol{\theta}$ are shown in the Figure 3.3.1.

The setting 4, 5, 6 and 7 are generated by the mixture of multivariate skew t distributions with two clusters and the density is given by

Setting 4, 5, 6 and 7

$$0.4\ ST(\boldsymbol{x} \mid \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \boldsymbol{\lambda}_1, v_1) + 0.6\ ST(\boldsymbol{x} \mid \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2, \boldsymbol{\lambda}_2, v_2).$$

where $ST(. \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, v)$ is the density of multivariate skew t distributions, $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are location vectors, $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ are scale matrices, $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$ are skewness vectors and v_1 and v_2 are degrees of freedom for cluster 1 and 2 respectively. Also, the parameters used for data simulation in mixture of multivariate skew t distributions are shown in the Table 3.3.3.

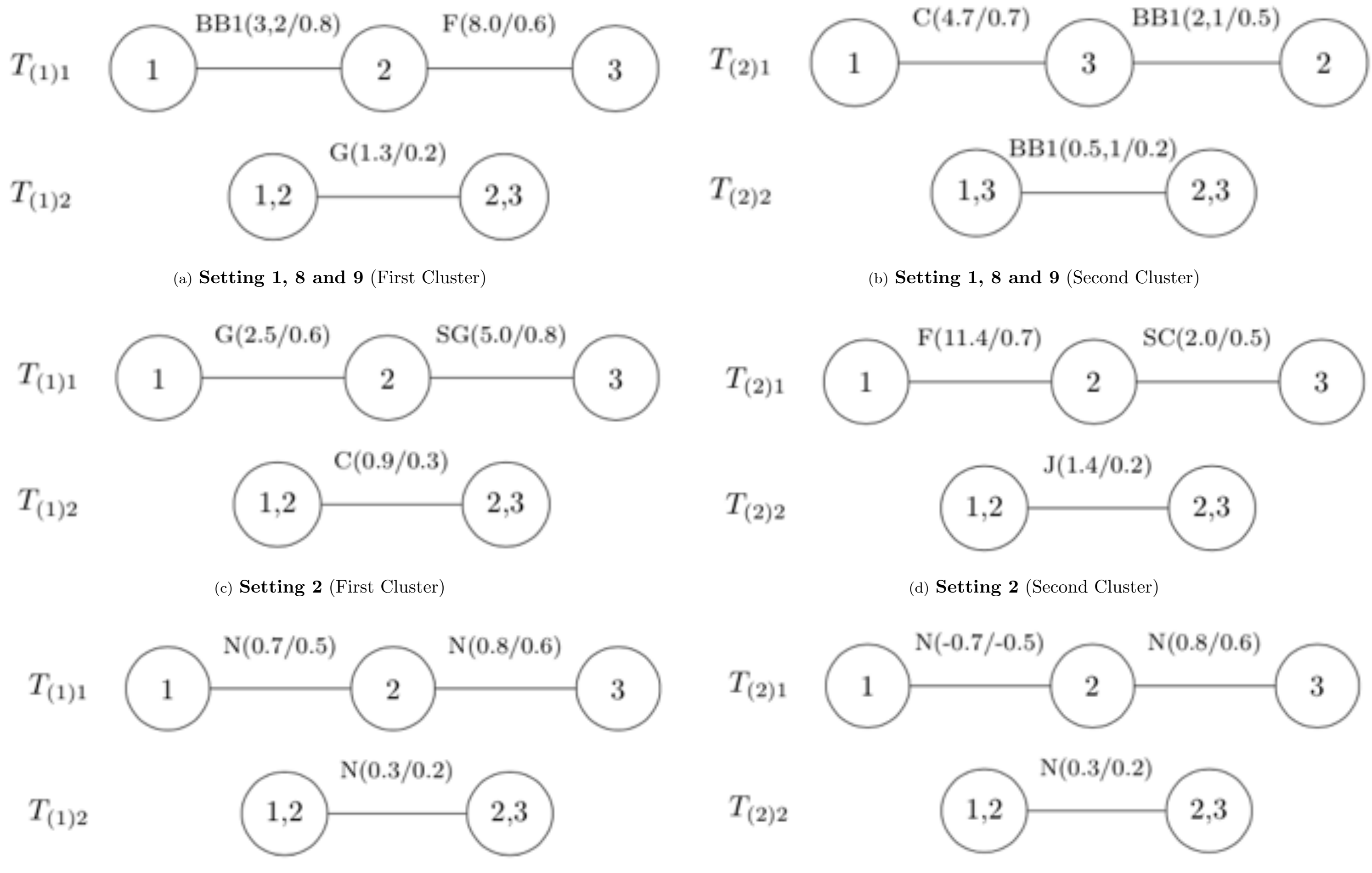
	Cluster 1				Cluster 2			
	$\boldsymbol{\mu}_1$	$\boldsymbol{\Sigma}_1$	$\boldsymbol{\lambda}_1$	v_1	$\boldsymbol{\mu}_2$	$\boldsymbol{\Sigma}_2$	$\boldsymbol{\lambda}_2$	v_2
Setting 4	$(1,1,0)^T$	$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$	$(4,-4,4)^T$	8	$(-2,-2,-2)^T$	$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$	$(-4,4,4)^T$	10
Setting 5	$(1,1,0)^T$	$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$	$(4,-4,4)^T$	3	$(-2,-2,-2)^T$	$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$	$(-4,4,4)^T$	3
Setting 6	$(1,1,0)^T$	$\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$	$(4,-4,4)^T$	3	$(-4.5,1,0)^T$	$\begin{bmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{bmatrix}$	$(-4,4,4)^T$	3
Setting 7	$(1,1,0)^T$	$\begin{bmatrix} 0.5 & 0.1 & 0.1 \\ 0.1 & 0.5 & 0.1 \\ 0.1 & 0.1 & 0.5 \end{bmatrix}$	$(4,-4,4)^T$	3	$(-5,-5,-5)^T$	$\begin{bmatrix} 50 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{bmatrix}$	$(-4,4,4)^T$	3

Table 3.3.3: The parameters used for data simulation in the mixture of multivariate skew t distributions for each cluster.

	Cluster 1			Cluster 2		
	$F_{1(1)}(\gamma_{1(1)})$	$F_{2(1)}(\gamma_{2(1)})$	$F_{3(1)}(\gamma_{3(1)})$	$F_{1(2)}(\gamma_{1(2)})$	$F_{2(2)}(\gamma_{2(2)})$	$F_{3(2)}(\gamma_{3(2)})$
Setting 1	$llogis(1.5, 1.25)$ (6.41,doesn't exist)	$exp(0.1)$ (10,10)	$lnorm(0.1, 1.3)$ (2.57,5.41)	$lnorm(2.5, 0.5)$ (13.80,7.36)	$logis(5, 3)$ (5,9.64)	$exp(0.05)$ (20,20)
Setting 2	$\mathcal{N}(1, 2)$ (1,2)	$exp(0.2)$ (5,5)	$lnorm(0.8, 0.8)$ (3.06,2.90)	$lnorm(1.5, 0.4)$ (4.85,2.02)	$\mathcal{N}(18, 5)$ (18,5)	$exp(0.2)$ (5,5)
Setting 3	$\mathcal{N}(0, 2)$ (0,2)	$\mathcal{N}(1, 2)$ (1,2)	$\mathcal{N}(1, 2)$ (1,2)	$\mathcal{N}(0, 2)$ (0,2)	$\mathcal{N}(1, 2)$ (1,2)	$\mathcal{N}(-2, 2)$ (-2,2)
Setting 8	$\mathcal{N}(20, 25)$ (20,25)	$\mathcal{N}(15, 10)$ (15,10)	$lnorm(2.3, 0.8)$ (13.74,13.051)	$lnorm(3, 0.4)$ (21.76,9.06)	$logis(8, 10)$ (8,32.15)	$\Gamma(0.5, 1)$ (0.5,0.71)
Setting 9	$\mathcal{N}(20, 25)$ (20,25)	$\mathcal{N}(15, 10)$ (15,10)	$lnorm(0, 0.3)$ (1.05,0.32)	$lnorm(3, 0.4)$ (21.76,9.06)	$logis(8, 10)$ (8,32.15)	$\Gamma(0.5, 1)$ (0.5,0.71)

Table 3.3.4: The parameters of the univariate marginal distributions for each dimension and cluster. The abbreviation for the marginal distributions : $\mathcal{N}(\mu, \sigma)$: normal distribution with mean μ and standard deviation σ. $exp(\lambda)$: exponential distribution with rate parameter λ. $lnorm(\mu, \sigma)$: log-normal distribution with mean μ and standard deviation σ on the logarithmic scale. $logis(l, s)$: logistic distribution with location parameter l and scale parameter s. $llogis(\alpha, \beta)$: log-logistic distribution with shape parameter α and scale parameter β. $\Gamma(\alpha, \beta)$: gamma distribution with shape parameter α and rate parameter β. The mean and the standard deviation of the univariate marginal distributions are given inside the parenthesis (mean, standard deviation) below the abbreviation.

Example 3.3.1 (Visualization of sample data sets from the vine copula mixture model)



(a) **Setting 1, 8 and 9** (First Cluster)

(b) **Setting 1, 8 and 9** (Second Cluster)

(c) **Setting 2** (First Cluster)

(d) **Setting 2** (Second Cluster)

(e) **Setting 3** (First Cluster)

(f) **Setting 3** (Second Cluster)

Figure 3.3.1: Vine tree structure of simulated data from the vine copula mixture model for **Setting 1, 8 and 9** : (a), (b), **Setting 2** : (c), (d) and **Setting 3** : (e), (f). A capital letter at an edge refers to the bivariate copula family: BB1: BB1 copula, C: Clayton, SC: Survival Clayton, F: Frank, G: Gumbel, SG: Survival Gumbel, J: Joe copula and N : Gaussian copula. The true parameter value and corresponding Kendall's τ of the pair copula are given inside the parenthesis (parameter(s)/Kendall's τ).

Table 3.3.5: The characteristics of the clusters in nine settings of data simulation. *The separation here refers to the level of separation whether the two clusters can be observed easily if the dots of each cluster are not coloured or marked. **The overlap here refers whether the two clusters can be separated by a plane almost completely.

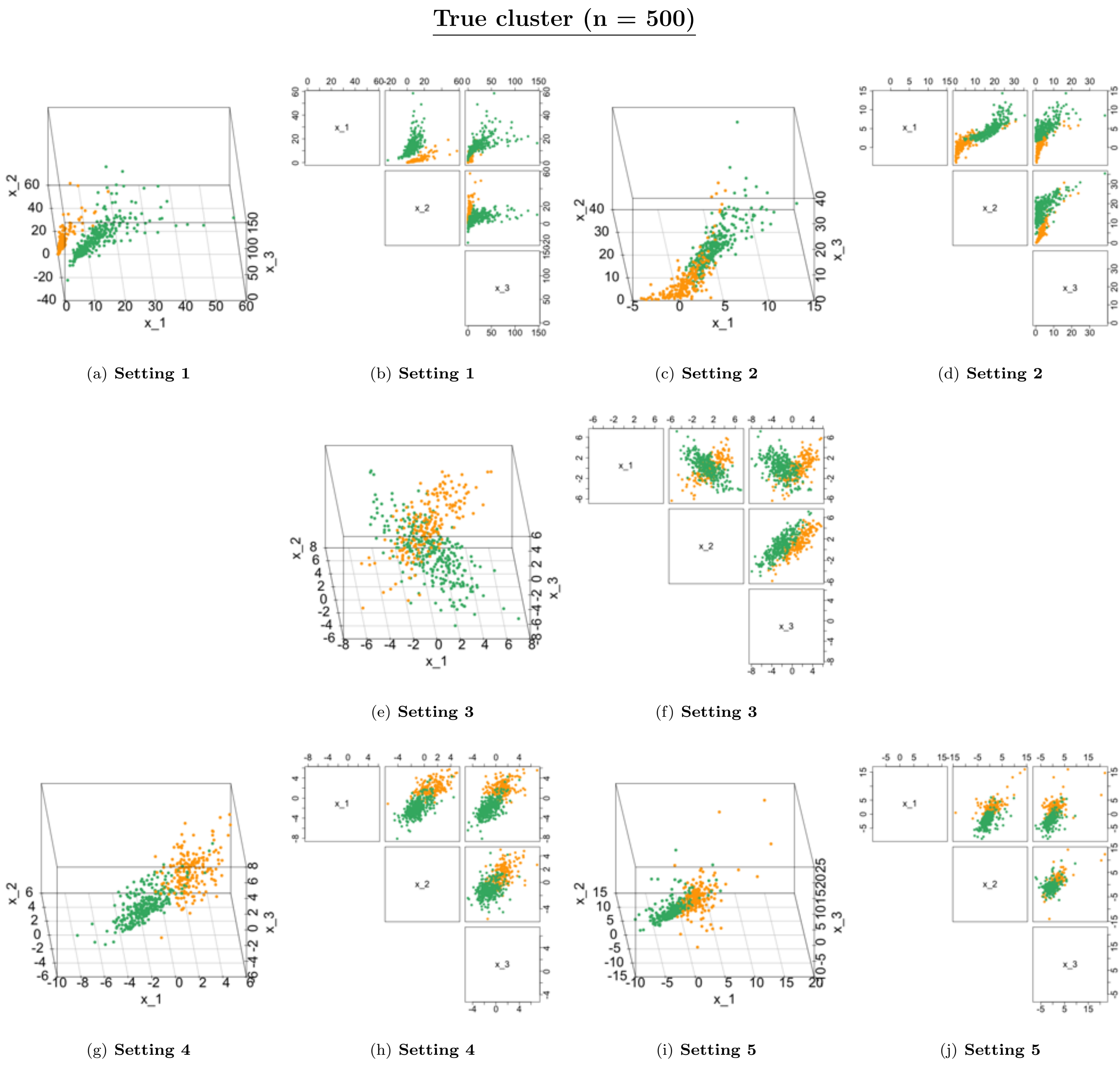
Setting	Characteristics of clusters			
	Shape (Elliptical / Skew elliptical / Non elliptical)	Separation* (Well / Not well)	Overlap** (Overlapping / Non-overlapping)	Volume (Similar / Different / Very different)
1	Non elliptical	Well separated	Non-overlapping	Different
2	Non elliptical	Not well separated	Overlapping	Similar
3	Elliptical	Well separated	Overlapping (X shape)	Similar
4	Elliptical / Skew elliptical	Not well separated	Overlapping	Similar
5	Skew elliptical	Not well separated	Overlapping	Different
6	Skew elliptical	Well separated	Non-overlapping	Similar
7	Skew elliptical	Well separated	Non-overlapping	Very different
8	Non elliptical	Well separated	Non-overlapping	Different
9	Non elliptical	Well separated	Overlapping (X shape)	Similar

This example will show you the visualisations of each data settings and the scatter plots for 500 and 1000 samples are shown in the Figure 3.3.2, 3.3.3 and 3.3.4. According to the scatter plots for the sample data, we can conclude four different characteristics of the clusters for each data setting, including shapes, separation, overlap and volumes and the conclusion is shown in the Table 3.3.5. Note that the separation here is not exactly the same as overlap. Separation here refers to the level of separation whether the two clusters can be observed easily if the dots are not coloured or marked. Also, the overlap here refers whether the two clusters can be separated by a plane almost completely. Therefore, for data setting 3 and 9, even the clusters are heavily overlapping like a "X", but they are still well separated, because we can observe that the data consists of two clusters clearly. The last characteristics volume classifies the size difference between two clusters. In data setting 7, the size difference of the clusters is much bigger than other data settings, so its characteristics volume is classified as "Very different".

Table 3.3.6: Selection of the EM algorithms, optimization methods for marginal parameter $\boldsymbol{\gamma}$ and their CM steps order for testing.

Algorithm (Optimization method for $\boldsymbol{\gamma}$)	Parameter update		
	CM step 1	CM step 2	CM step 3
ECM (Nelder-Mead)	π	$\boldsymbol{\theta}$	$\boldsymbol{\gamma}$
ECM (BFGS)	π	$\boldsymbol{\theta}$	$\boldsymbol{\gamma}$
MCECM (Nelder-Mead)	π	$\boldsymbol{\theta}$	$\boldsymbol{\gamma}$
MCECM (BFGS)	π	$\boldsymbol{\theta}$	$\boldsymbol{\gamma}$
ECME (Nelder-Mead)	π	$\boldsymbol{\theta}$	$\boldsymbol{\gamma}$
ECME (BFGS)	π	$\boldsymbol{\theta}$	$\boldsymbol{\gamma}$

Step 2: Decide on the EM algorithms and the CM steps order in the VCMM algorithm
In our experiment, we will use 3 EM algorithms including ECM, MCECM and ECME algorithm to estimate the parameters in the vine copula mixture model. As mentioned in the Section 2.4.3, the parameters that we need to estimate are mixing proportion π, copula parameter $\boldsymbol{\theta}$ and marginal parameter $\boldsymbol{\gamma}$. For mixing proportion π, we can estimate it by using the Equation (2.4.20). Also, we have no analytical solution for the copula parameter $\boldsymbol{\theta}$, so we can use the R function `RVineSeqMLE` in the R package `VineCopula` to estimate the pair-copula parameters by maximization likelihood estimation (MLE) which uses L-BFGS-B to estimate the copula parameters. For marginal parameter $\boldsymbol{\gamma}$, there is no analytical solution for estimation and two numerical optimization methods of which Nelder-Mead is a heuristic search optimization without derivatives and BFGS is gradient based optimization method are used for marginal parameter estimation. In the experiment, each EM



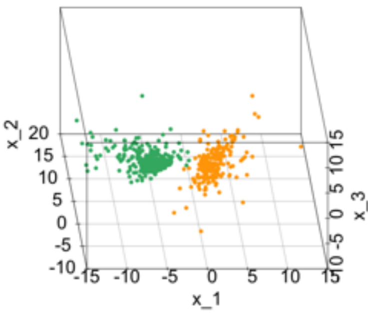
(a) **Setting 1** (b) **Setting 1** (c) **Setting 2** (d) **Setting 2**

(e) **Setting 3** (f) **Setting 3**

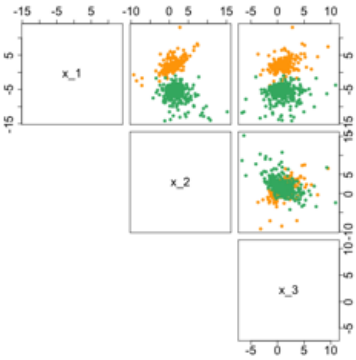
(g) **Setting 4** (h) **Setting 4** (i) **Setting 5** (j) **Setting 5**

Figure 3.3.2: The scatter plots and the corresponding pairs plots of 500 simulated data generated from setting 1 to 5. The orange and green colour refer to the points in Cluster 1 and 2 respectively.

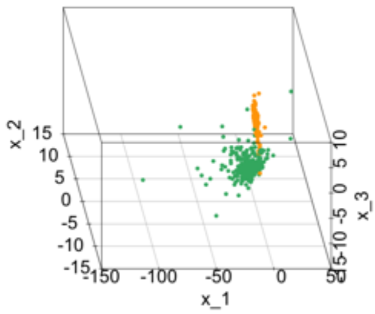
True cluster (n = 500)



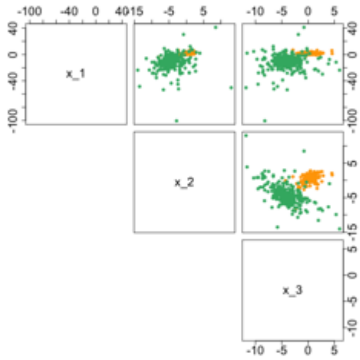
(a) **Setting 6**



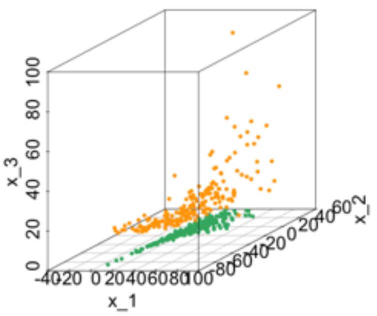
(b) **Setting 6**



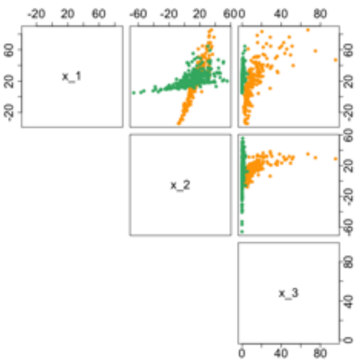
(c) **Setting 7**



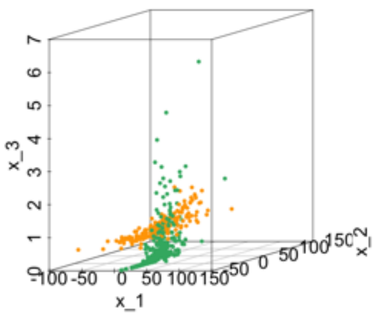
(d) **Setting 7**



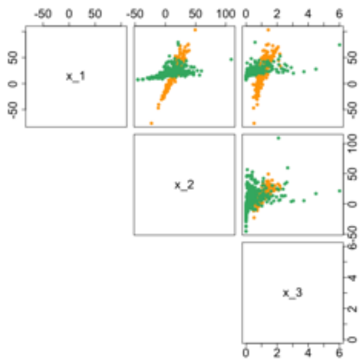
(e) **Setting 8**



(f) **Setting 8**

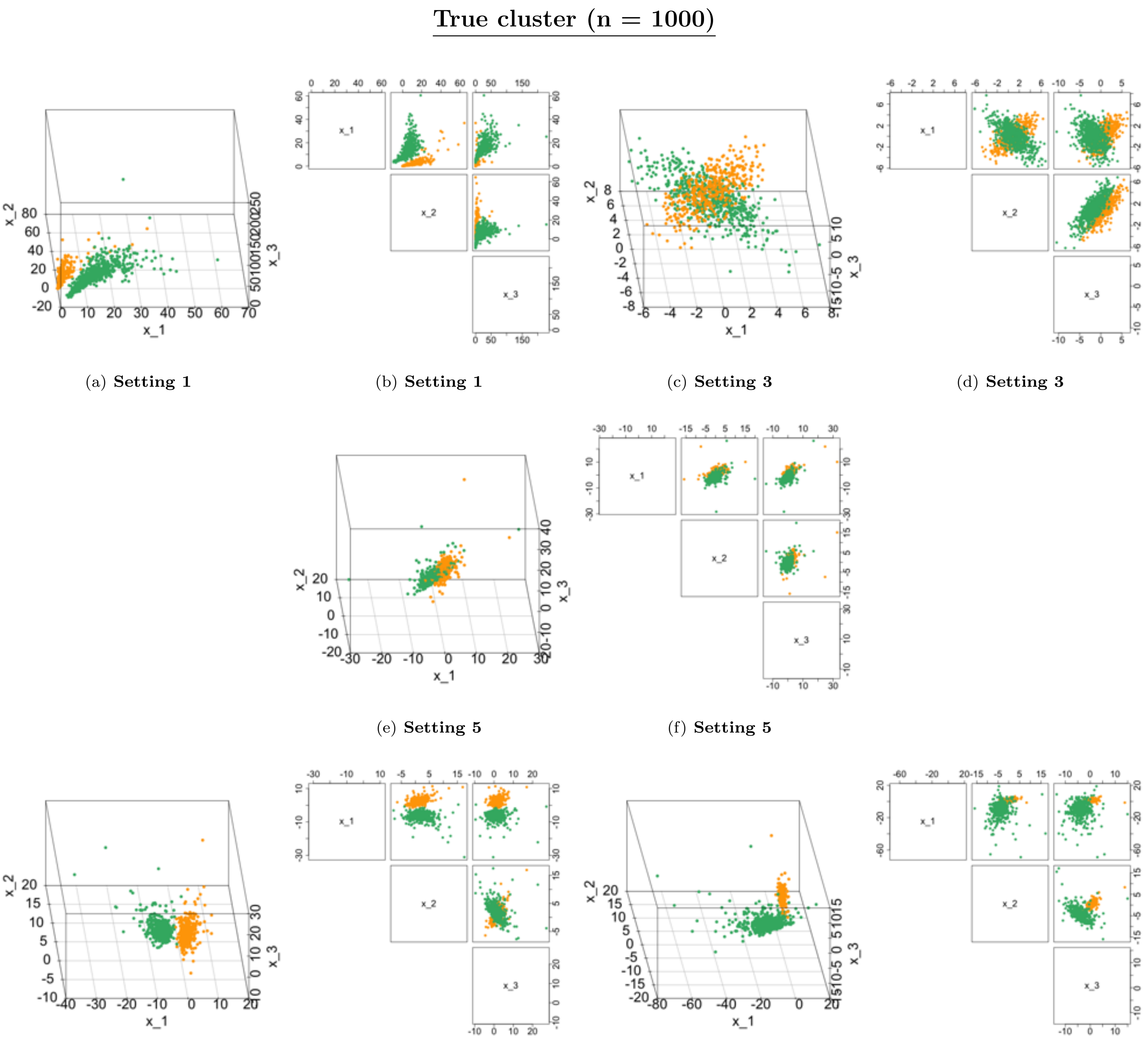


(g) **Setting 9**



(h) **Setting 9**

Figure 3.3.3: The scatter plots and the corresponding pairs plots of 500 simulated data generated from setting 6 to 9. The orange and green colour refer to the points in Cluster 1 and 2 respectively.



(a) **Setting 1** (b) **Setting 1** (c) **Setting 3** (d) **Setting 3**

(e) **Setting 5** (f) **Setting 5**

(g) **Setting 6** (h) **Setting 6** (i) **Setting 7** (j) **Setting 7**

Figure 3.3.4: The scatter plots and the corresponding pairs plots of 1000 simulated data generated from setting 1, 3, 5, 6 and 7. The orange and green colour refer to the points in Cluster 1 and 2 respectively.

algorithm is used with two different optimization methods to estimate the marginal parameter $\boldsymbol{\gamma}$, so we study six algorithms in total. About the order of the CM steps, we just follow the steps shown in the Section 2.4.3. Basically, the parameters update in each CM step follows the order : Mixing proportion π > Copula parameter $\boldsymbol{\theta}$ > Marginal parameter $\boldsymbol{\gamma}$. The overview of the selection of the EM algorithms, optimization methods for marginal parameter and their CM steps in our experiment are shown in the Table 3.3.6.

Step 3: Decide on the initialization strategies for the VCMM algorithm

In Sahin and Czado [2021], "Kmeans (scale)" is used for the initialization. Here, we use further initialization strategies which were already introduced in the last Section 3.2. They are "Random", "Kmeans", "Hierarchical", "Kmeans (scale)" and "Hierarchical (scale)".

Step 4: Decide on the marginal distribution and copula families for modelling in the VCMM algorithm

VCMM algorithm is a very flexible modelling method, because it can model many combinations of marginal distribution, copula family and pair copula construction. Here is a brief introduction for the flexibility of the VCMM algorithm:

1. Pair copula construction (Vine tree structure)

In the Theorem 2.3.18 (A pair copula construction in three dimensions), we have mentioned that there are 3 different ways for copula construction to model different relationship between the variables:

$$\begin{aligned}
f(x_1, x_2, x_3) &= c_{1,3;2}(F_{1|2}(x_1|x_2), F_{3|2}(x_3|x_2)) \times c_{2,3}(F_2(x_2), F_3(x_3)) \\
&\quad \times c_{1,2}(F_1(x_1), F_2(x_2)) f_3(x_3) f_2(x_2) f_1(x_1) \\
f(x_1, x_2, x_3) &= c_{1,2;3}(F_{1|3}(x_1|x_3), F_{2|1}(x_2|x_2)) \times c_{1,3}(F_1(x_1), F_3(x_3)) \\
&\quad \times c_{2,3}(F_2(x_2), F_3(x_3)) f_3(x_3) f_2(x_2) f_1(x_1) \\
f(x_1, x_2, x_3) &= c_{2,3;1}(F_{2|1}(x_2|x_1), F_{3|1}(x_3|x_1)) \times c_{1,3}(F_1(x_1), F_3(x_3)) \\
&\quad \times c_{1,2}(F_1(x_1), F_2(x_2)) f_3(x_3) f_2(x_2) f_1(x_1)
\end{aligned}$$

2. Marginal distribution F

In the VCMM algorithm, we will select 7 different candidates for the marginal distribution: 1. Gaussian distribution, 2. Log-normal distribution, 3. Exponential distribution, 4. Log-logistic distribution, 5. Logistic distribution, 6. Gamma distribution and 7. Student's t distribution with degrees of freedom 3.

3. Copula family c

In the VCMM algorithm, we will use the following copula families :
Gaussian copula, Student's t copula, Gumbel copula, Clayton copula, Frank copula, Joe copula, BB1 copula, BB6 copula, BB8 copula and their rotated copulas with 90°, 180°, 270° degrees.

In our simulation studies, we will use 2 different modelling methods which allow the use of different marginal distributions and copula families. The detail is shown in the Table 3.3.7.

Table 3.3.7: The overview of the marginal distribution and copula families used for the modelling in VCMM algorithm. *All refers to the marginal distributions and copula families mentioned above in step 4.

Modelling method	Marginal distribution	Copula family	Pair copula construction
Gaussian VCMM	Gaussian	Gaussian	Determined by the VCMM algorithm
General VCMM	All*	All*	Determined by the VCMM algorithm

Step 5: Perform the simulation in R

Each experiment shown in the Table 2.1.9 will be repeated for 50 times, which means that the replication for each experiment R is 50. The detail of the VCMM algorithm is not shown here. For more information, please refer to Sahin and Czado [2021] for the VCMM algorithm and the Section 2.4.2 in this thesis for the E and CM steps of the EM algorithms for parameter estimation.

Table 3.3.8: Abbreviations for the studied experiments

Abbreviations	Setting	Initialization strategies	Modelling method	Sample size (n)
1.R.Gau.n	1	Random	Gaussian VCMM	500 and 1000
1.K.Gau.n	1	Kemans	Gaussian VCMM	500 and 1000
1.H.Gau.n	1	Hierarchical	Gaussian VCMM	500 and 1000
1.KS.Gau.n	1	Kemans (scale)	Gaussian VCMM	500 and 1000
1.HS.Gau.n	1	Hierarchical (scale)	Gaussian VCMM	500 and 1000
2.R.Gau.n	2	Random	Gaussian VCMM	500
2.K.Gau.n	2	Kemans	Gaussian VCMM	500
2.H.Gau.n	2	Hierarchical	Gaussian VCMM	500
2.KS.Gau.n	2	Kemans (scale)	Gaussian VCMM	500
2.HS.Gau.n	2	Hierarchical (scale)	Gaussian VCMM	500
3.R.Gau.n	3	Random	Gaussian VCMM	500 and 1000
3.K.Gau.n	3	Kemans	Gaussian VCMM	500 and 1000
3.H.Gau.n	3	Hierarchical	Gaussian VCMM	500 and 1000
3.KS.Gau.n	3	Kemans (scale)	Gaussian VCMM	500 and 1000
3.HS.Gau.n	3	Hierarchical (scale)	Gaussian VCMM	500 and 1000
4.R.Gau.n	4	Random	Gaussian VCMM	500
4.K.Gau.n	4	Kemans	Gaussian VCMM	500
4.H.Gau.n	4	Hierarchical	Gaussian VCMM	500
4.KS.Gau.n	4	Kemans (scale)	Gaussian VCMM	500
4.HS.Gau.n	4	Hierarchical (scale)	Gaussian VCMM	500
5.R.Gau.n	5	Random	Gaussian VCMM	500 and 1000
5.K.Gau.n	5	Kemans	Gaussian VCMM	500 and 1000
5.H.Gau.n	5	Hierarchical	Gaussian VCMM	500 and 1000
5.KS.Gau.n	5	Kemans (scale)	Gaussian VCMM	500 and 1000
5.HS.Gau.n	5	Hierarchical (scale)	Gaussian VCMM	500 and 1000
6.R.Gau.n	6	Random	Gaussian VCMM	500 and 1000
6.K.Gau.n	6	Kemans	Gaussian VCMM	500 and 1000
6.H.Gau.n	6	Hierarchical	Gaussian VCMM	500 and 1000
6.KS.Gau.n	6	Kemans (scale)	Gaussian VCMM	500 and 1000
6.HS.Gau.n	6	Hierarchical (scale)	Gaussian VCMM	500 and 1000
7.R.Gau.n	7	Random	Gaussian VCMM	500 and 1000
7.K.Gau.n	7	Kemans	Gaussian VCMM	500 and 1000
7.H.Gau.n	7	Hierarchical	Gaussian VCMM	500 and 1000
7.KS.Gau.n	7	Kemans (scale)	Gaussian VCMM	500 and 1000
7.HS.Gau.n	7	Hierarchical (scale)	Gaussian VCMM	500 and 1000
8.R.Gau.n	8	Random	Gaussian VCMM	500
8.K.Gau.n	8	Kemans	Gaussian VCMM	500
8.H.Gau.n	8	Hierarchical	Gaussian VCMM	500
8.KS.Gau.n	8	Kemans (scale)	Gaussian VCMM	500
8.HS.Gau.n	8	Hierarchical (scale)	Gaussian VCMM	500
9.R.Gau.n	9	Random	Gaussian VCMM	500
9.K.Gau.n	9	Kemans	Gaussian VCMM	500
9.H.Gau.n	9	Hierarchical	Gaussian VCMM	500
9.KS.Gau.n	9	Kemans (scale)	Gaussian VCMM	500
9.HS.Gau.n	9	Hierarchical (scale)	Gaussian VCMM	500

Abbreviations	Setting	Initialization strategies	Modelling method	Sample size (n)
1.R.Gen.n	1	Random	General VCMM	500 and 1000
1.K.Gen.n	1	Kemans	General VCMM	500 and 1000
1.H.Gen.n	1	Hierarchical	General VCMM	500 and 1000
1.KS.Gen.n	1	Kemans (scale)	General VCMM	500 and 1000
1.HS.Gen.n	1	Hierarchical (scale)	General VCMM	500 and 1000
2.R.Gen.n	2	Random	General VCMM	500
2.K.Gen.n	2	Kemans	General VCMM	500
2.H.Gen.n	2	Hierarchical	General VCMM	500
2.KS.Gen.n	2	Kemans (scale)	General VCMM	500
2.HS.Gen.n	2	Hierarchical (scale)	General VCMM	500
3.R.Gen.n	3	Random	General VCMM	500 and 1000
3.K.Gen.n	3	Kemans	General VCMM	500 and 1000
3.H.Gen.n	3	Hierarchical	General VCMM	500 and 1000
3.KS.Gen.n	3	Kemans (scale)	General VCMM	500 and 1000
3.HS.Gen.n	3	Hierarchical (scale)	General VCMM	500 and 1000
4.R.Gen.n	4	Random	General VCMM	500
4.K.Gen.n	4	Kemans	General VCMM	500
4.H.Gen.n	4	Hierarchical	General VCMM	500
4.KS.Gen.n	4	Kemans (scale)	General VCMM	500
4.HS.Gen.n	4	Hierarchical (scale)	General VCMM	500
5.R.Gen.n	5	Random	General VCMM	500 and 1000
5.K.Gen.n	5	Kemans	General VCMM	500 and 1000
5.H.Gen.n	5	Hierarchical	General VCMM	500 and 1000
5.KS.Gen.n	5	Kemans (scale)	General VCMM	500 and 1000
5.HS.Gen.n	5	Hierarchical (scale)	General VCMM	500 and 1000
6.R.Gen.n	6	Random	General VCMM	500 and 1000
6.K.Gen.n	6	Kemans	General VCMM	500 and 1000
6.H.Gen.n	6	Hierarchical	General VCMM	500 and 1000
6.KS.Gen.n	6	Kemans (scale)	General VCMM	500 and 1000
6.HS.Gen.n	6	Hierarchical (scale)	General VCMM	500 and 1000
7.R.Gen.n	7	Random	General VCMM	500 and 1000
7.K.Gen.n	7	Kemans	General VCMM	500 and 1000
7.H.Gen.n	7	Hierarchical	General VCMM	500 and 1000
7.KS.Gen.n	7	Kemans (scale)	General VCMM	500 and 1000
7.HS.Gen.n	7	Hierarchical (scale)	General VCMM	500 and 1000
8.R.Gen.n	8	Random	General VCMM	500
8.K.Gen.n	8	Kemans	General VCMM	500
8.H.Gen.n	8	Hierarchical	General VCMM	500
8.KS.Gen.n	8	Kemans (scale)	General VCMM	500
8.HS.Gen.n	8	Hierarchical (scale)	General VCMM	500
9.R.Gen.n	9	Random	General VCMM	500
9.K.Gen.n	9	Kemans	General VCMM	500
9.H.Gen.n	9	Hierarchical	General VCMM	500
9.KS.Gen.n	9	Kemans (scale)	General VCMM	500
9.HS.Gen.n	9	Hierarchical (scale)	General VCMM	500

3.3.2 Result visualisation and the performance

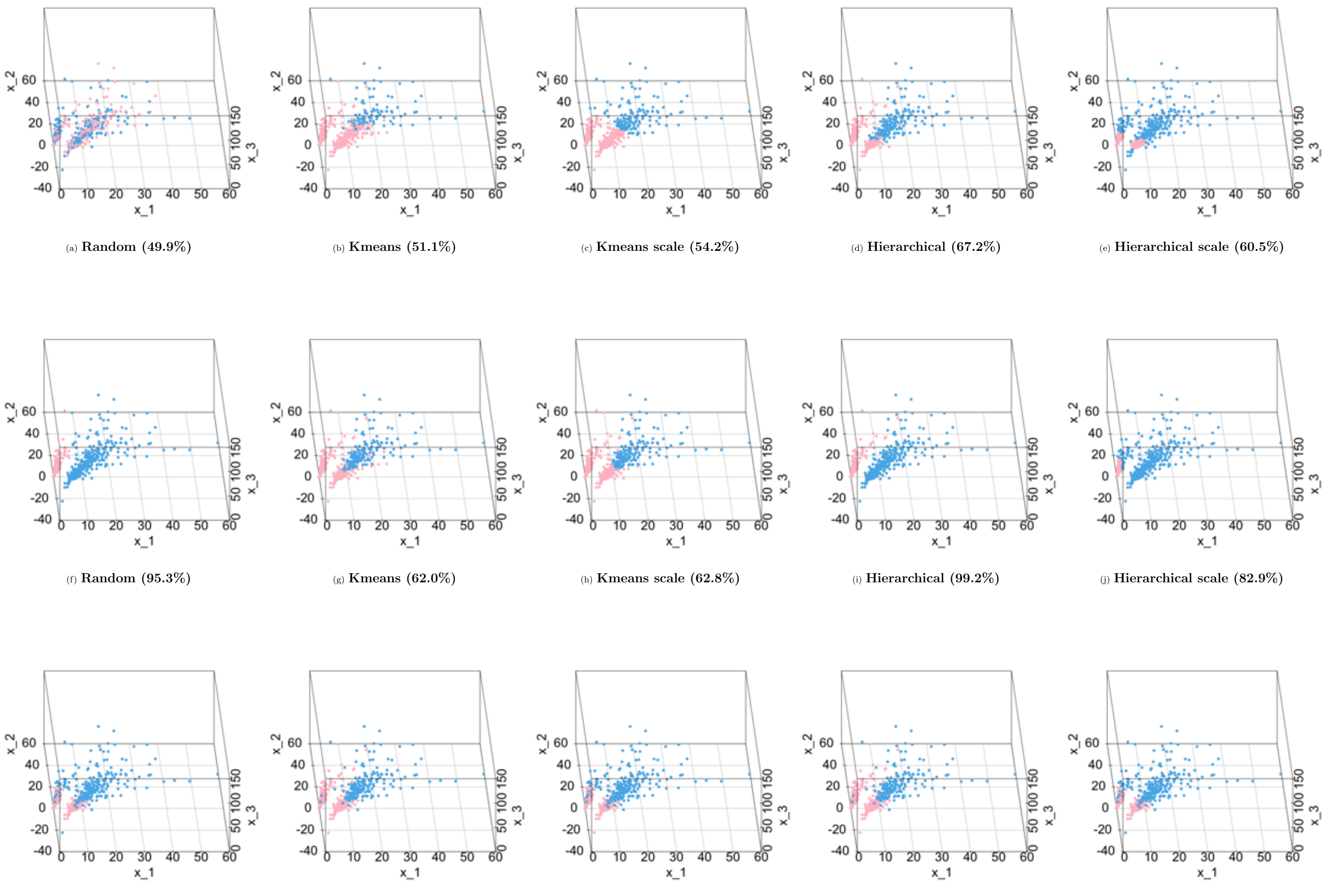
Step 6: Create visualization of the performance measures

The graphs for result visualization are shown in Step 7.

Step 7: Analyse and compare the results

1. Scatter plots after initializing and final clustering

Setting 1, n = 500



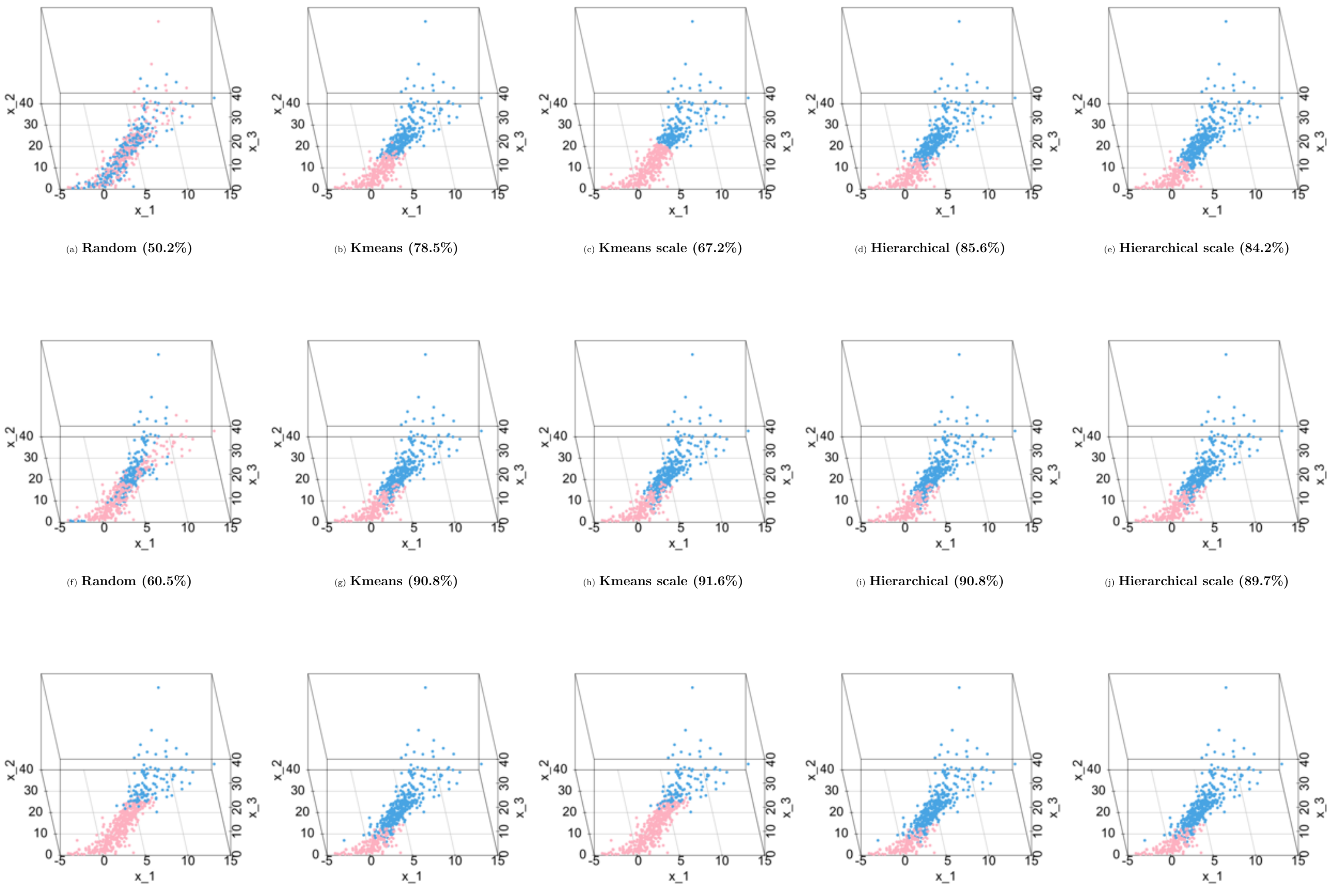
(a) Random (49.9%) (b) Kmeans (51.1%) (c) Kmeans scale (54.2%) (d) Hierarchical (67.2%) (e) Hierarchical scale (60.5%)

(f) Random (95.3%) (g) Kmeans (62.0%) (h) Kmeans scale (62.8%) (i) Hierarchical (99.2%) (j) Hierarchical scale (82.9%)

(k) Random (60.9%) (l) Kmeans (62.4%) (m) Kmeans scale (60.9%) (n) Hierarchical (62.4%) (o) Hierarchical scale (60.9%)

Figure 3.3.5: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 1 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 2, n = 500



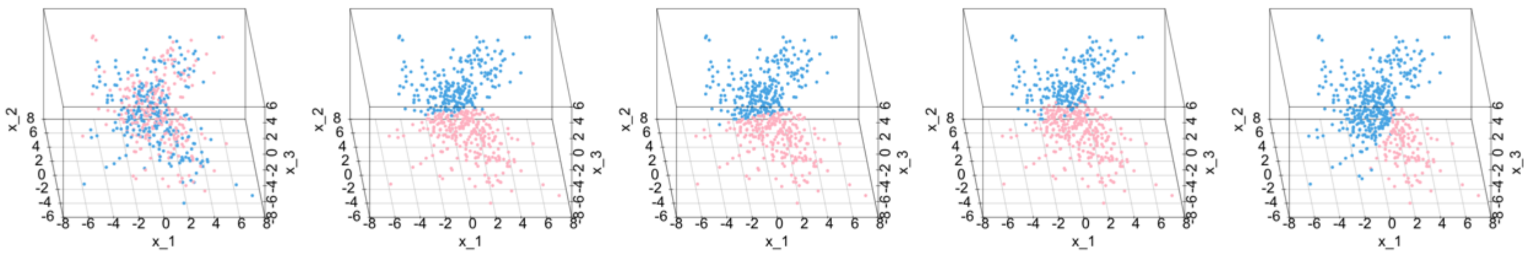
(a) Random (50.2%) (b) Kmeans (78.5%) (c) Kmeans scale (67.2%) (d) Hierarchical (85.6%) (e) Hierarchical scale (84.2%)

(f) Random (60.5%) (g) Kmeans (90.8%) (h) Kmeans scale (91.6%) (i) Hierarchical (90.8%) (j) Hierarchical scale (89.7%)

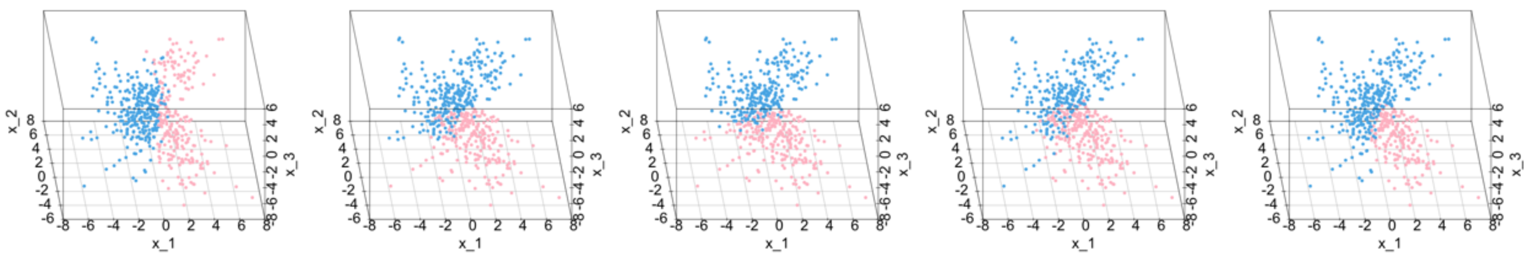
(k) Random (51.2%) (l) Kmeans (81.0%) (m) Kmeans scale (51.2%) (n) Hierarchical (81.0%) (o) Hierarchical scale (81.0%)

Figure 3.3.6: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 2 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

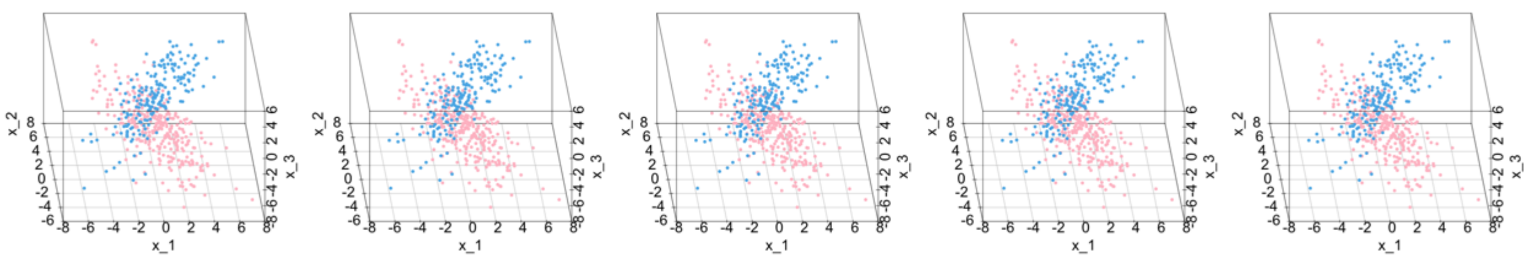
Setting 3, n = 500



(a) Random (50.1%) (b) Kmeans (59.8%) (c) Kmeans scale (56.1%) (d) Hierarchical (60.7%) (e) Hierarchical scale (54.8%)



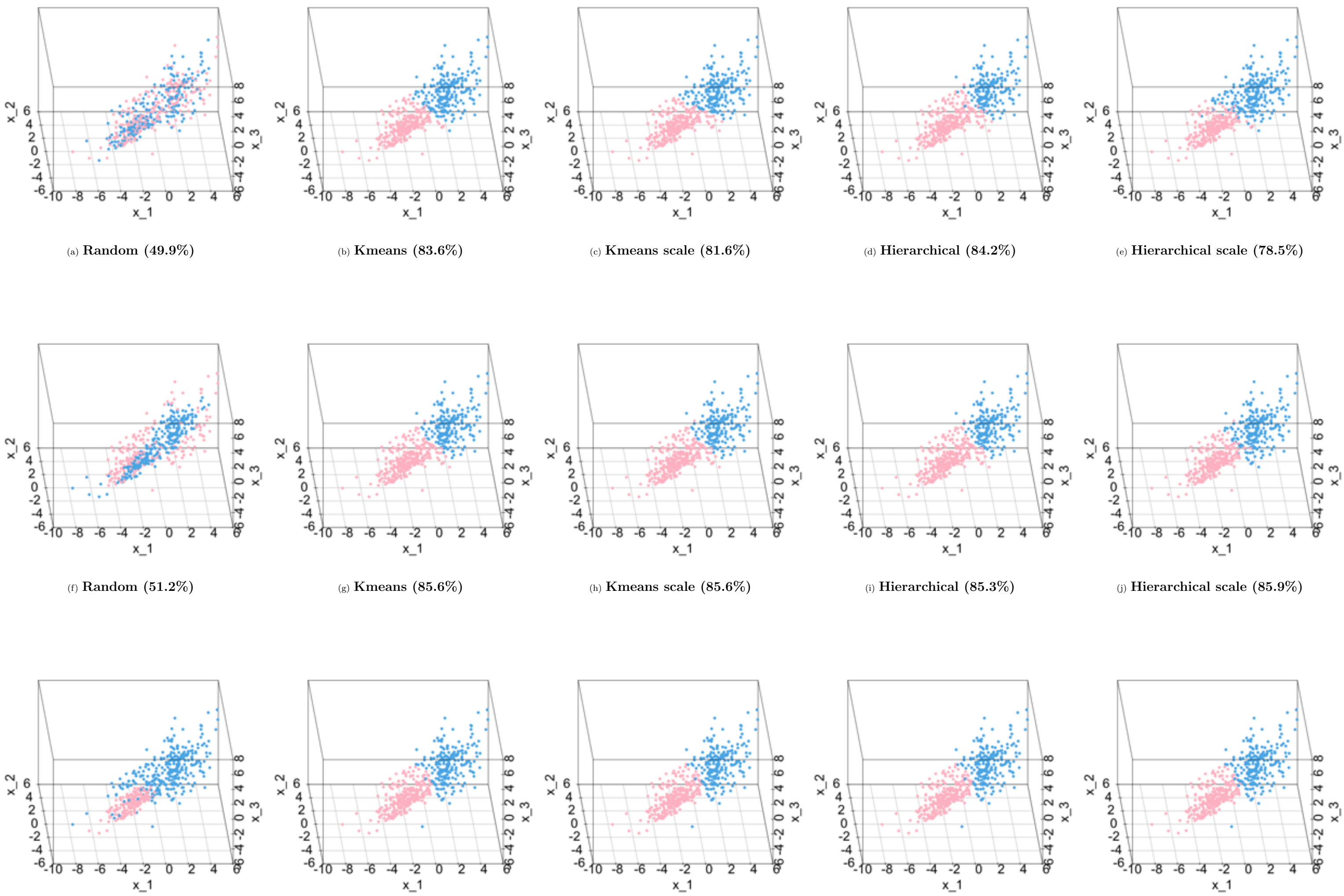
(f) Random (50.3%) (g) Kmeans (64.7%) (h) Kmeans scale (58.6%) (i) Hierarchical (69.7%) (j) Hierarchical scale (61.4%)



(k) Random (89.4%) (l) Kmeans (88.0%) (m) Kmeans scale (88.0%) (n) Hierarchical (88.0%) (o) Hierarchical scale (89.1%)

Figure 3.3.7: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 3 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 4, n = 500



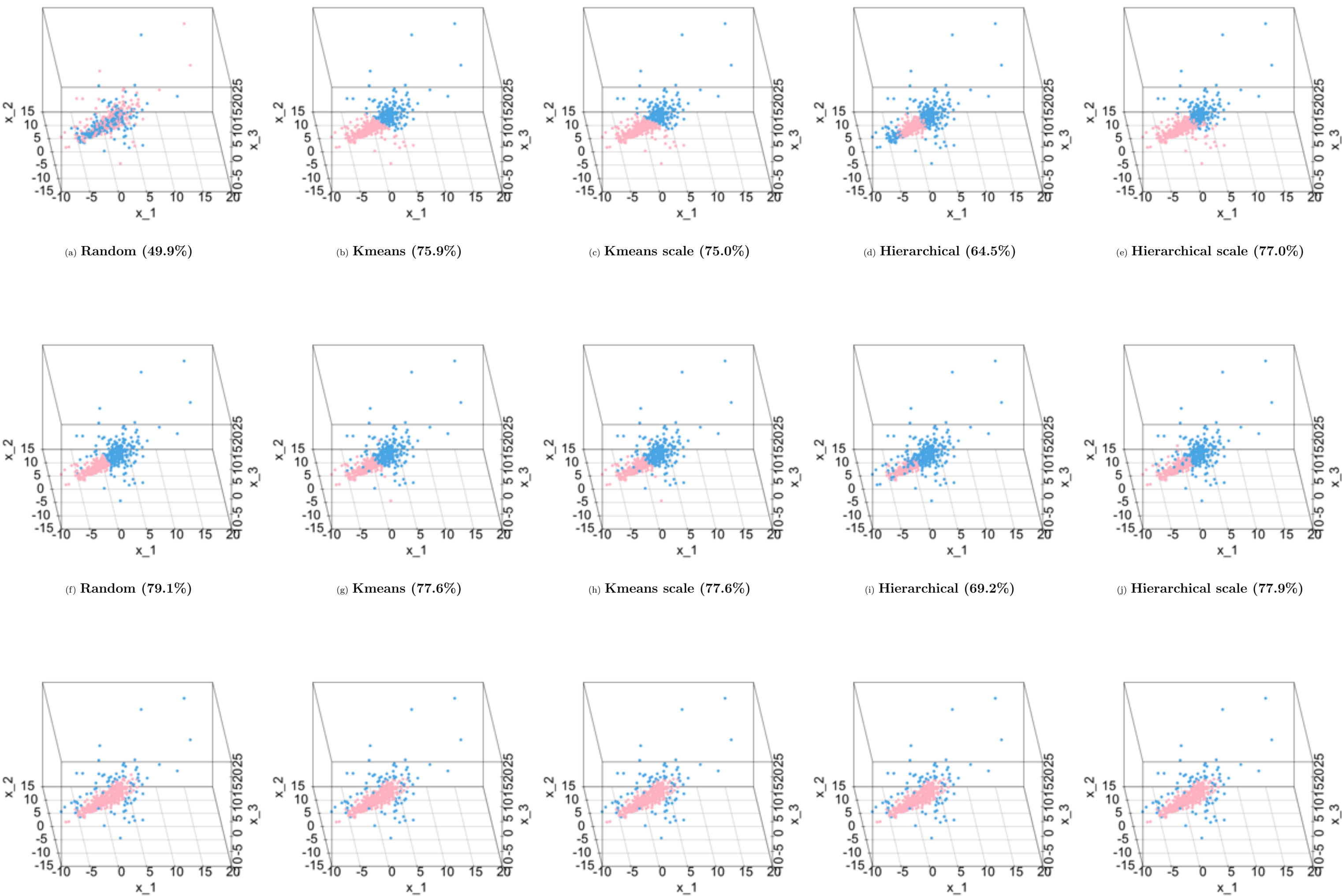
(a) Random (49.9%) (b) Kmeans (83.6%) (c) Kmeans scale (81.6%) (d) Hierarchical (84.2%) (e) Hierarchical scale (78.5%)

(f) Random (51.2%) (g) Kmeans (85.6%) (h) Kmeans scale (85.6%) (i) Hierarchical (85.3%) (j) Hierarchical scale (85.9%)

(k) Random (65.2%) (l) Kmeans (87.3%) (m) Kmeans scale (87.6%) (n) Hierarchical (88.3%) (o) Hierarchical scale (87.3%)

Figure 3.3.8: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 3 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 5, n = 500



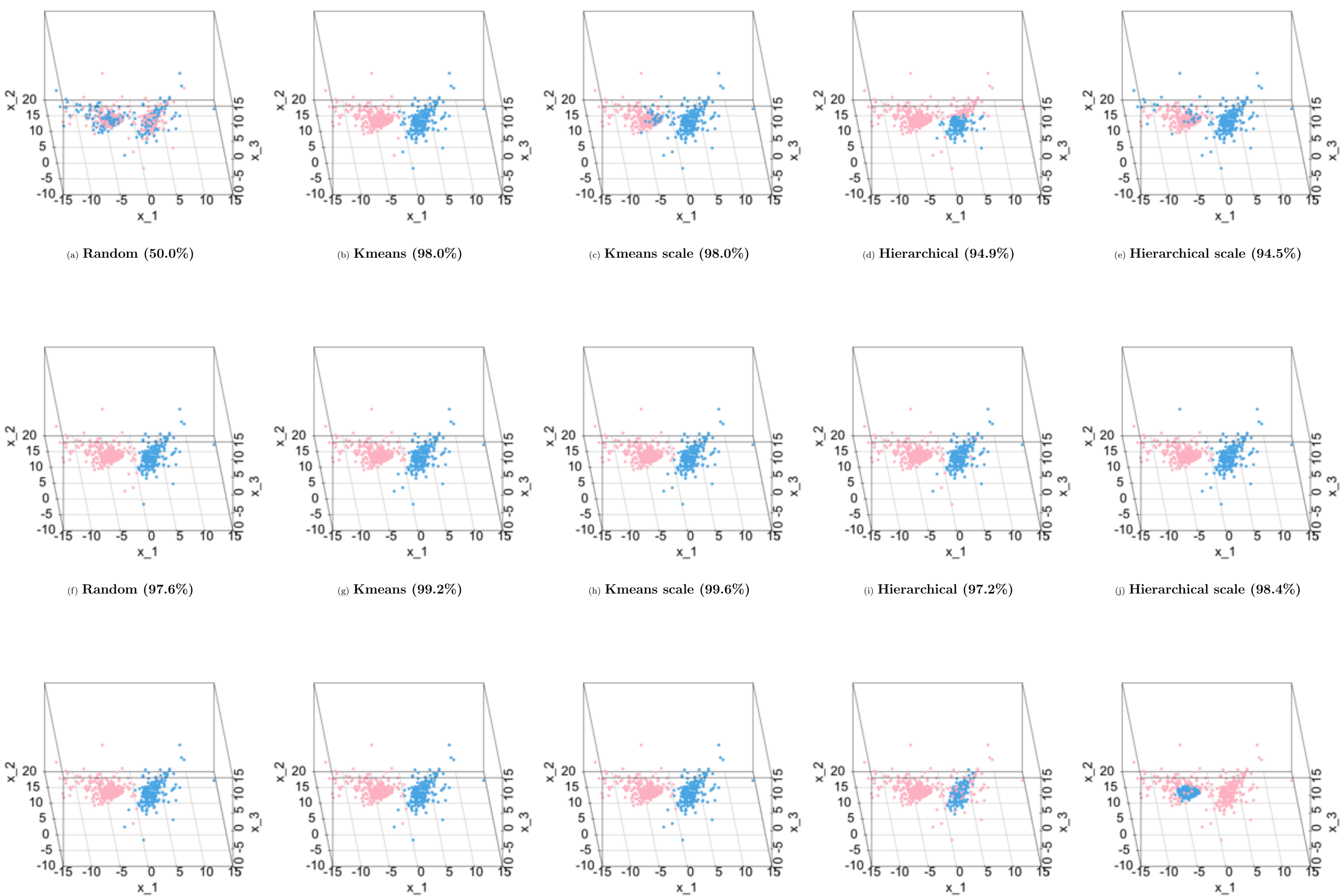
(a) **Random (49.9%)** (b) **Kmeans (75.9%)** (c) **Kmeans scale (75.0%)** (d) **Hierarchical (64.5%)** (e) **Hierarchical scale (77.0%)**

(f) **Random (79.1%)** (g) **Kmeans (77.6%)** (h) **Kmeans scale (77.6%)** (i) **Hierarchical (69.2%)** (j) **Hierarchical scale (77.9%)**

(k) **Random (53.0%)** (l) **Kmeans (53.0%)** (m) **Kmeans scale (53.0%)** (n) **Hierarchical (53.0%)** (o) **Hierarchical scale (53.0%)**

Figure 3.3.9: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 5 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 6, n = 500



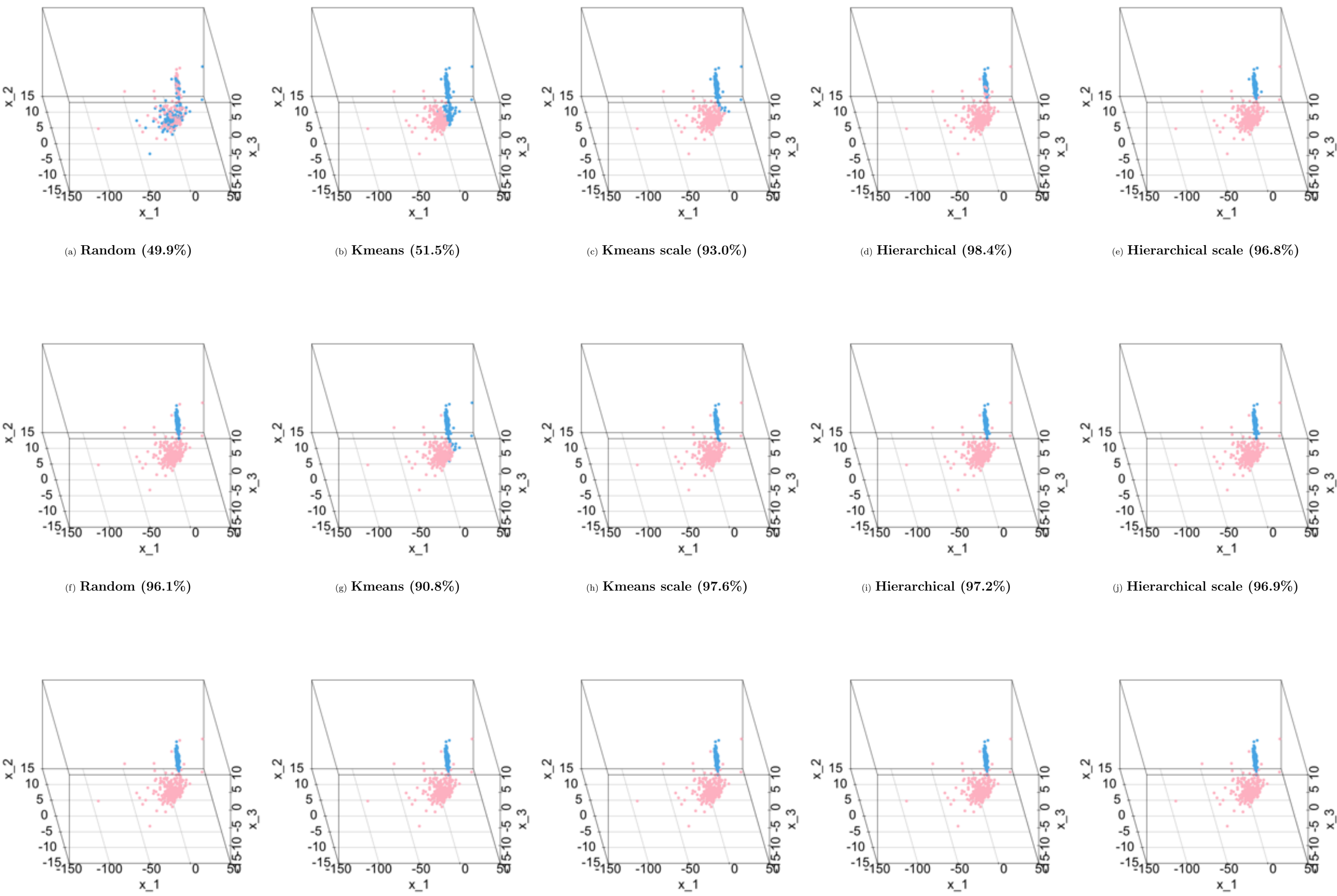
(a) Random (50.0%) (b) Kmeans (98.0%) (c) Kmeans scale (98.0%) (d) Hierarchical (94.9%) (e) Hierarchical scale (94.5%)

(f) Random (97.6%) (g) Kmeans (99.2%) (h) Kmeans scale (99.6%) (i) Hierarchical (97.2%) (j) Hierarchical scale (98.4%)

(k) Random (98.0%) (l) Kmeans (98.4%) (m) Kmeans scale (98.8%) (n) Hierarchical (80.7%) (o) Hierarchical scale (64.3%)

Figure 3.3.10: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 6 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 7, n = 500



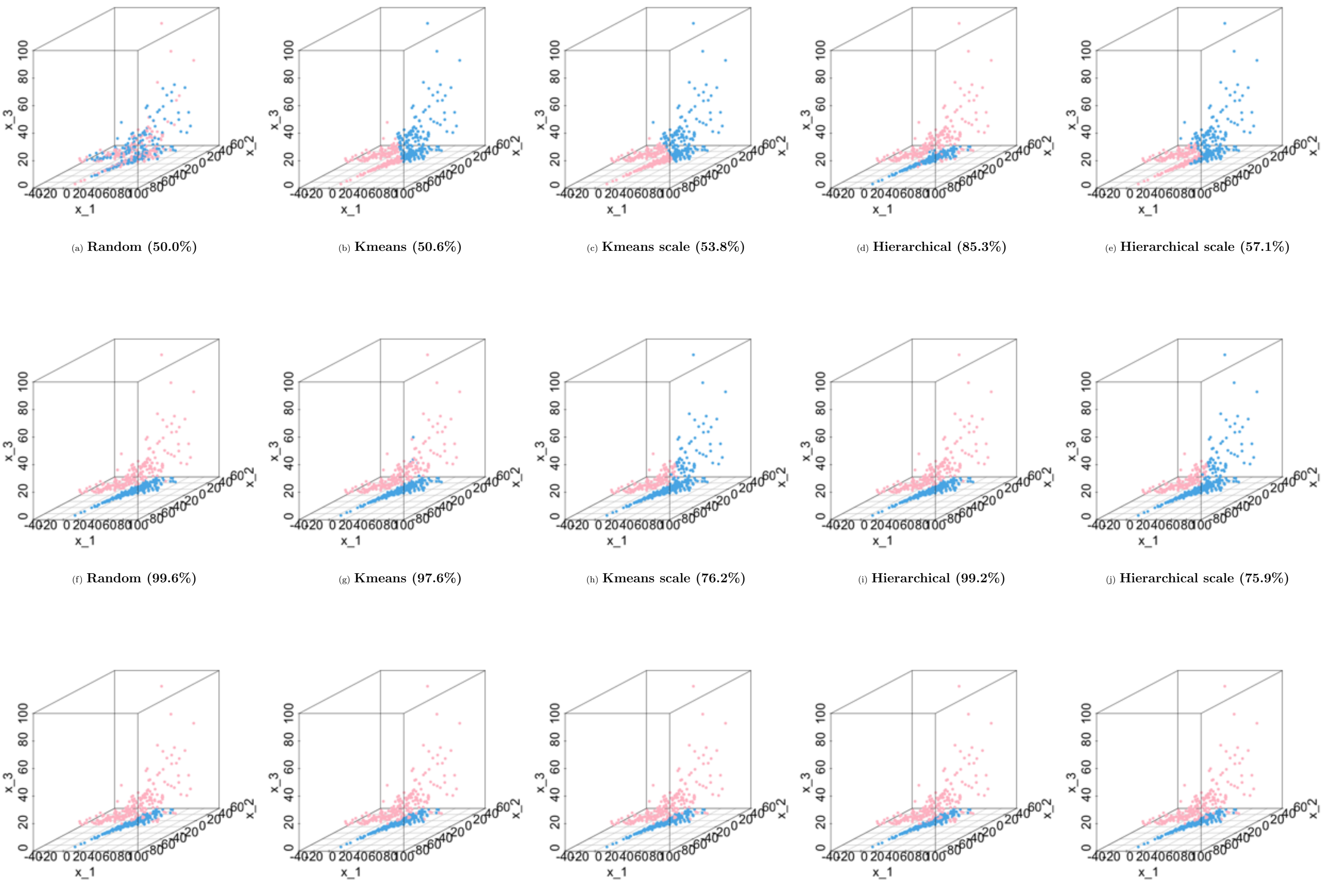
(a) **Random (49.9%)** (b) **Kmeans (51.5%)** (c) **Kmeans scale (93.0%)** (d) **Hierarchical (98.4%)** (e) **Hierarchical scale (96.8%)**

(f) **Random (96.1%)** (g) **Kmeans (90.8%)** (h) **Kmeans scale (97.6%)** (i) **Hierarchical (97.2%)** (j) **Hierarchical scale (96.9%)**

(k) **Random (94.9%)** (l) **Kmeans (96.5%)** (m) **Kmeans scale (96.5%)** (n) **Hierarchical (96.5%)** (o) **Hierarchical scale (96.5%)**

Figure 3.3.11: The scatter plots of initializing clustering and final clustering by using different initialization strategies for Setting 7 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 8, n = 500



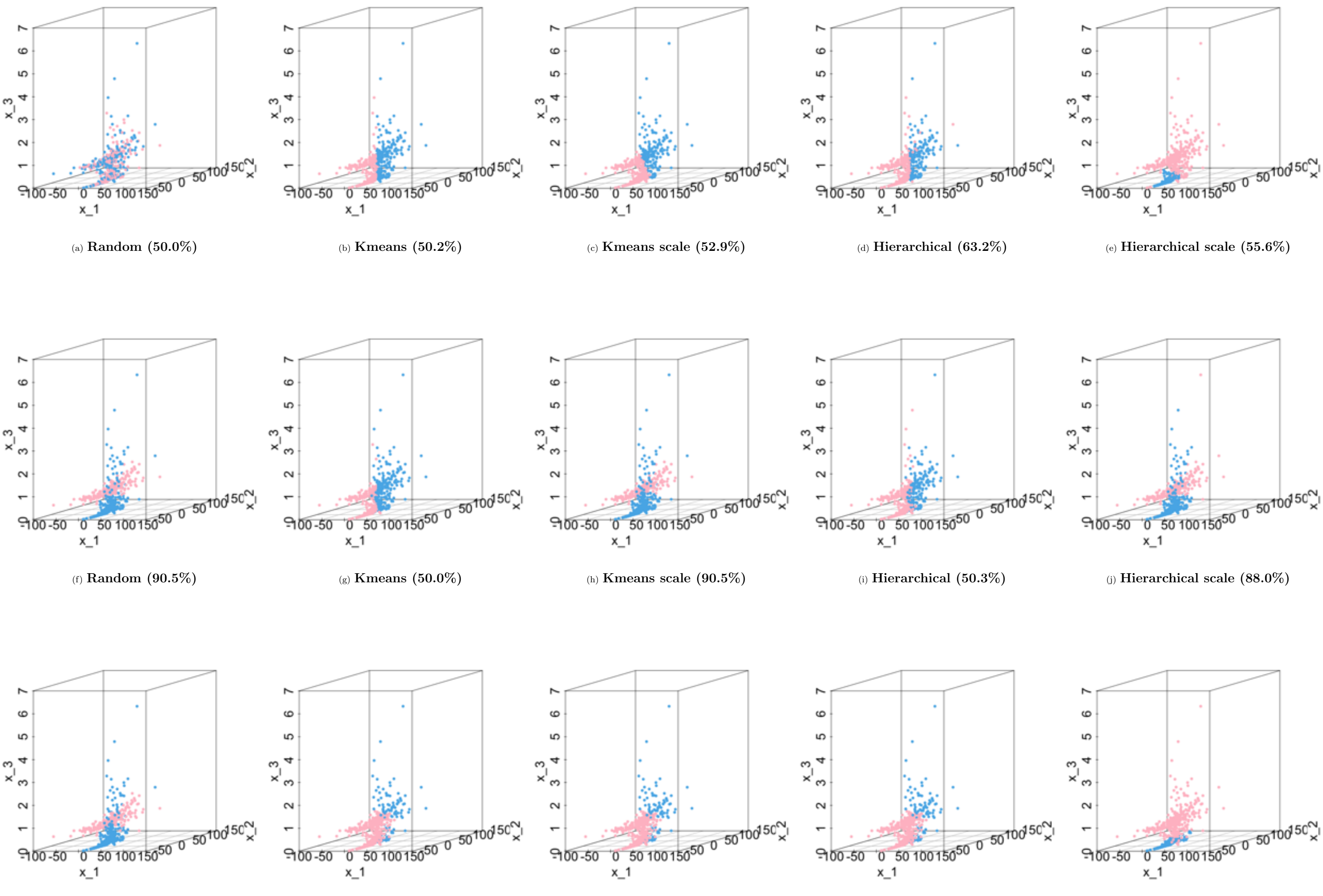
(a) Random (50.0%) (b) Kmeans (50.6%) (c) Kmeans scale (53.8%) (d) Hierarchical (85.3%) (e) Hierarchical scale (57.1%)

(f) Random (99.6%) (g) Kmeans (97.6%) (h) Kmeans scale (76.2%) (i) Hierarchical (99.2%) (j) Hierarchical scale (75.9%)

(k) Random (93.0%) (l) Kmeans (94.2%) (m) Kmeans scale (94.2%) (n) Hierarchical (91.2%) (o) Hierarchical scale (94.2%)

Figure 3.3.12: The scatter plots of initializing clustering and final clustering by using different initialization strategies for Setting 8 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 9, n = 500



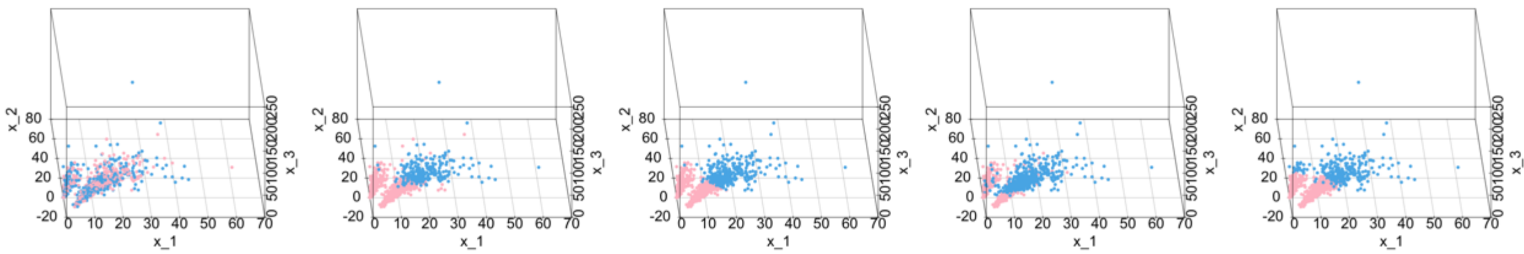
(a) **Random (50.0%)** (b) **Kmeans (50.2%)** (c) **Kmeans scale (52.9%)** (d) **Hierarchical (63.2%)** (e) **Hierarchical scale (55.6%)**

(f) **Random (90.5%)** (g) **Kmeans (50.0%)** (h) **Kmeans scale (90.5%)** (i) **Hierarchical (50.3%)** (j) **Hierarchical scale (88.0%)**

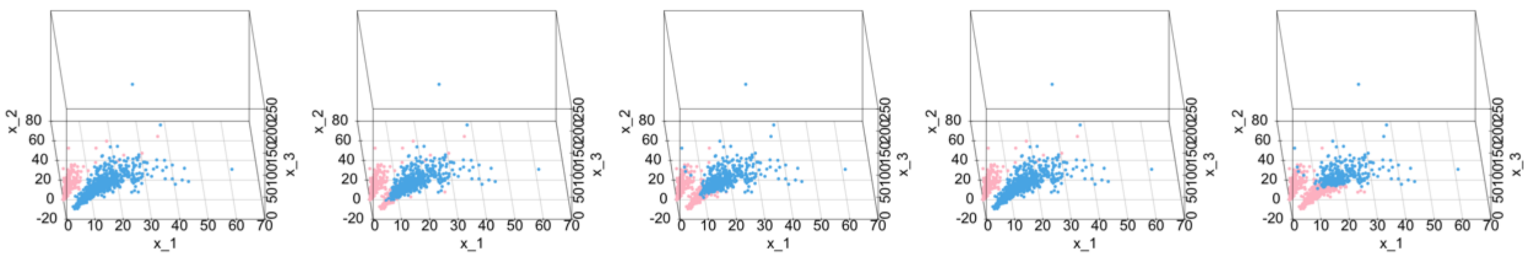
(k) **Random (89.4%)** (l) **Kmeans (50.6%)** (m) **Kmeans scale (50.6%)** (n) **Hierarchical (50.6%)** (o) **Hierarchical scale (63.0%)**

Figure 3.3.13: The scatter plots of initializing clustering and final clustering by using different initialization strategies for Setting 9 with sample size 500. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

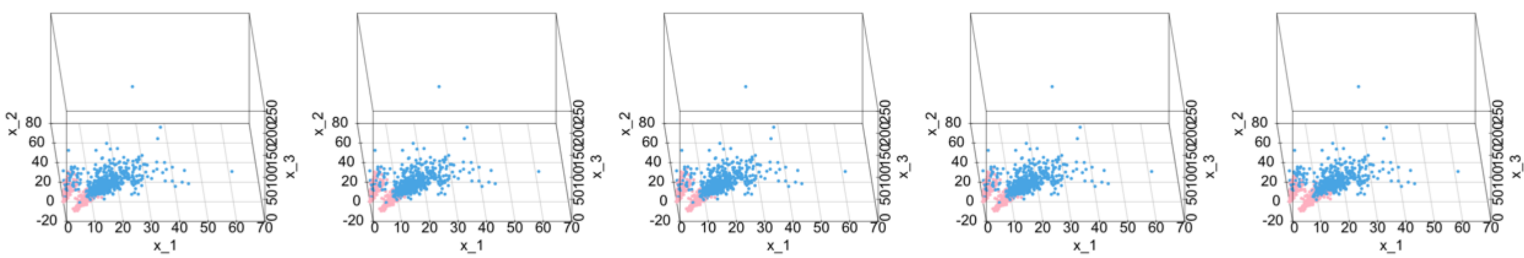
Setting 1, n = 1000



(a) **Random (50.0%)** (b) **Kmeans (52.7%)** (c) **Kmeans scale (55.3%)** (d) **Hierarchical (71.0%)** (e) **Hierarchical scale (51.2%)**



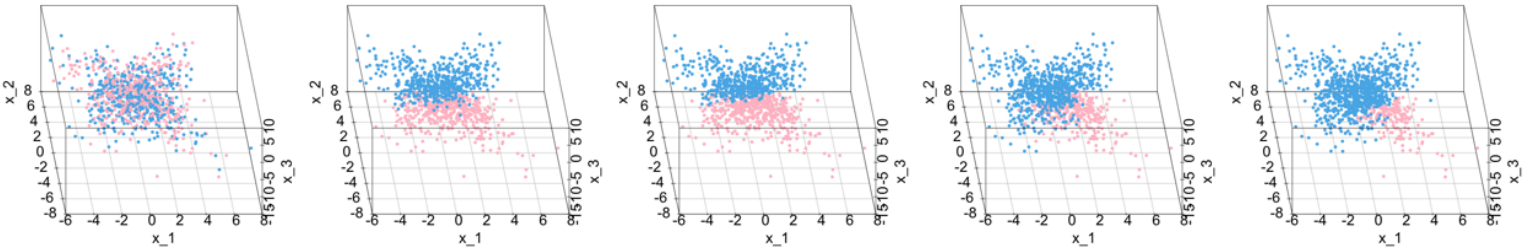
(f) **Random (98.8%)** (g) **Kmeans (80.9%)** (h) **Kmeans scale (65.0%)** (i) **Hierarchical (99.0%)** (j) **Hierarchical scale (54.5%)**



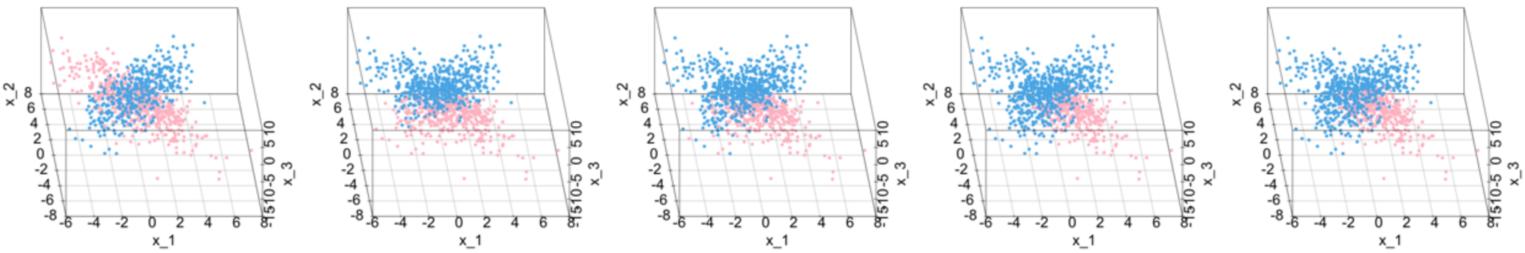
(k) **Random (63.8%)** (l) **Kmeans (64.9%)** (m) **Kmeans scale (64.8%)** (n) **Hierarchical (64.8%)** (o) **Hierarchical scale (58.8%)**

Figure 3.3.14: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 1 with sample size 1000. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

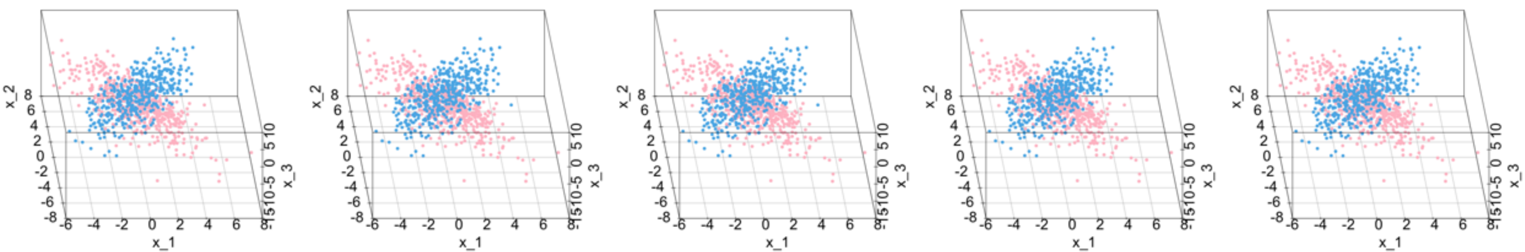
<u>Setting 3, n = 1000</u>



(a) **Random (50.0%)** (b) **Kmeans (55.9%)** (c) **Kmeans scale (53.9%)** (d) **Hierarchical (56.5%)** (e) **Hierarchical scale (50.8%)**



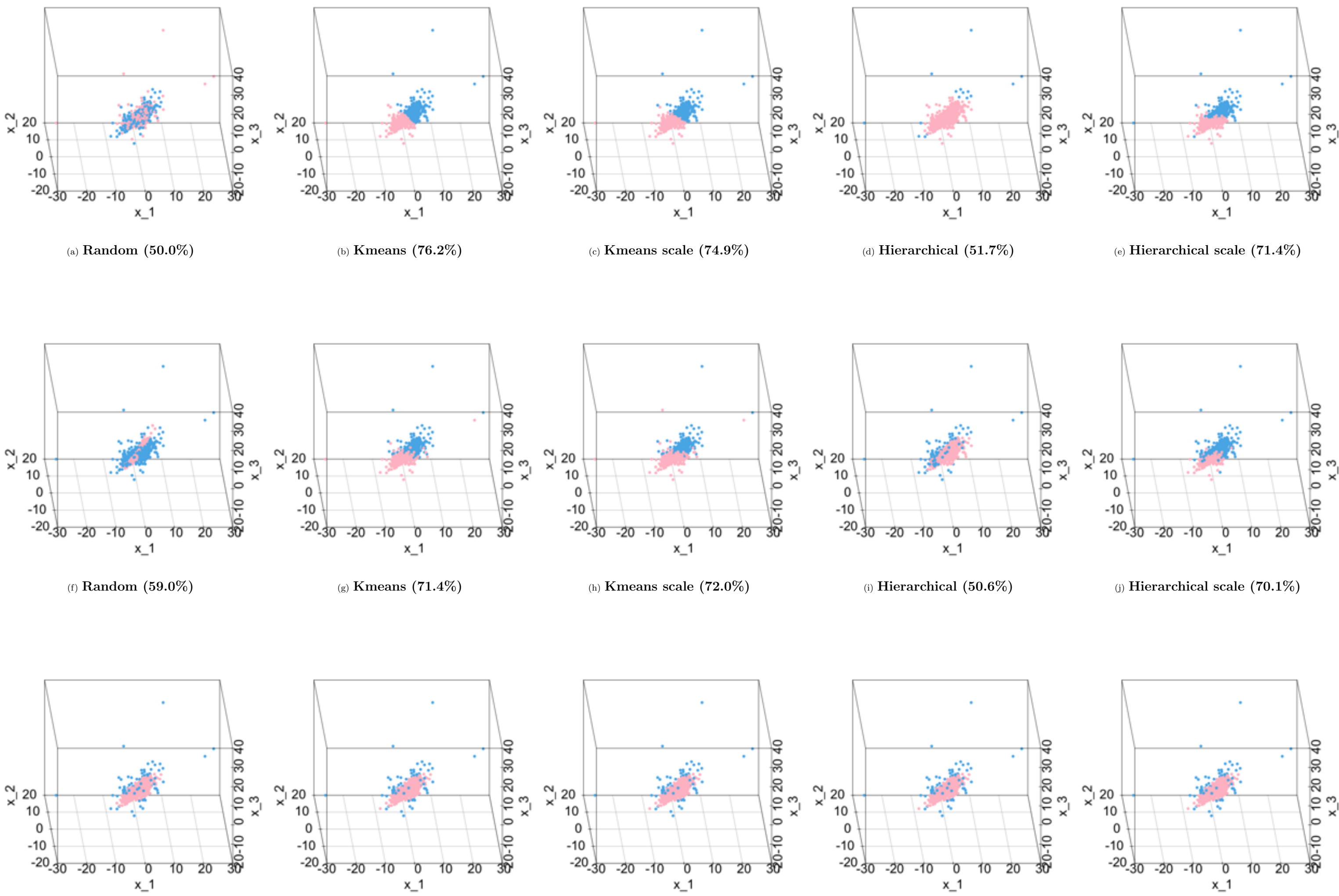
(f) **Random (86.5%)** (g) **Kmeans (57.3%)** (h) **Kmeans scale (60.3%)** (i) **Hierarchical (59.9%)** (j) **Hierarchical scale (62.5%)**



(k) **Random (86.6%)** (l) **Kmeans (85.6%)** (m) **Kmeans scale (85.6%)** (n) **Hierarchical (86.5%)** (o) **Hierarchical scale (86.1%)**

Figure 3.3.15: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 3 with sample size 1000. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 5, n = 1000



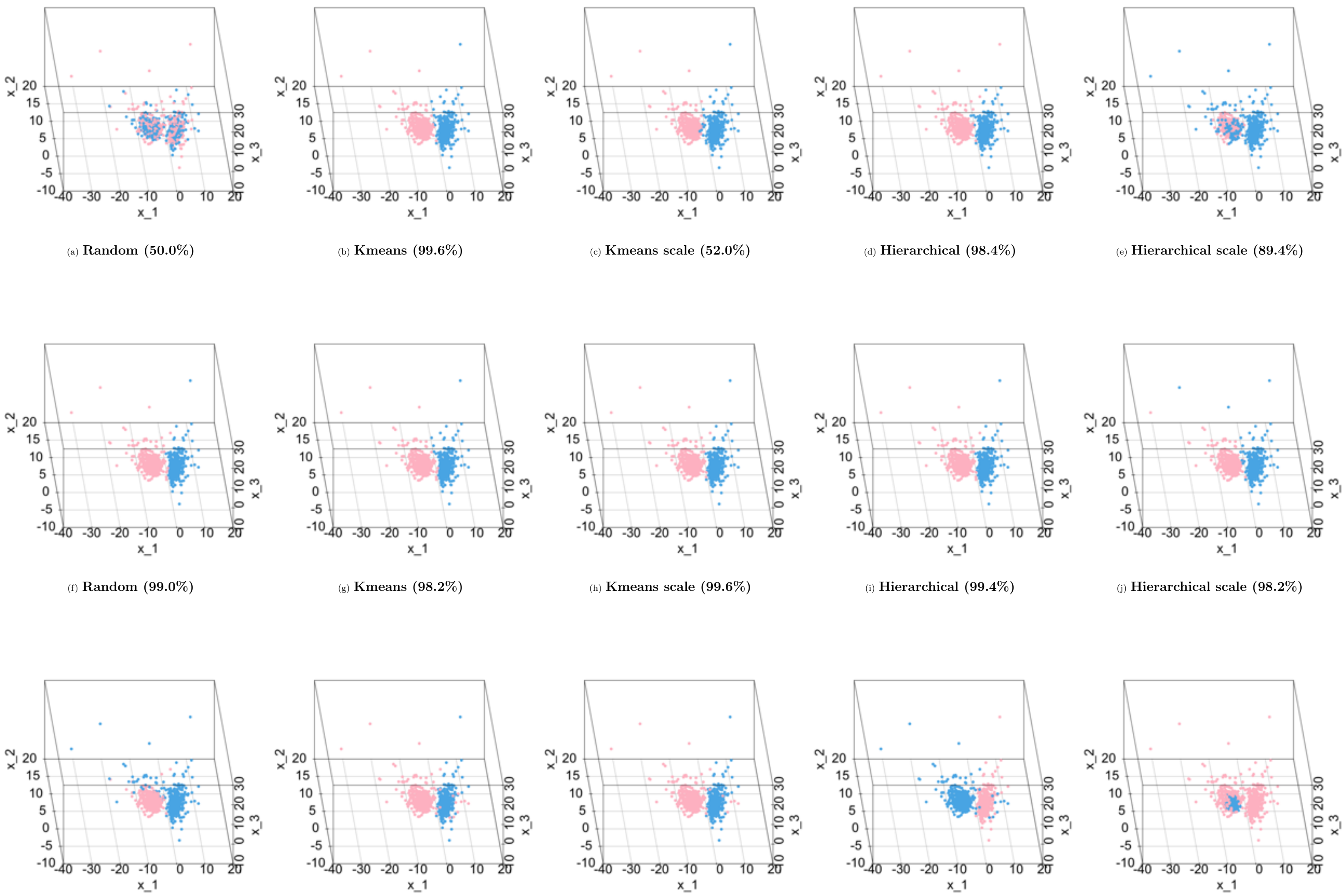
(a) Random (50.0%) (b) Kmeans (76.2%) (c) Kmeans scale (74.9%) (d) Hierarchical (51.7%) (e) Hierarchical scale (71.4%)

(f) Random (59.0%) (g) Kmeans (71.4%) (h) Kmeans scale (72.0%) (i) Hierarchical (50.6%) (j) Hierarchical scale (70.1%)

(k) Random (51.6%) (l) Kmeans (51.5%) (m) Kmeans scale (51.5%) (n) Hierarchical (51.5%) (o) Hierarchical scale (51.5%)

Figure 3.3.16: The scatter plots of initializing clustering and final clustering by using different initialization strategies for setting 5 with sample size 1000. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

Setting 6, n = 1000



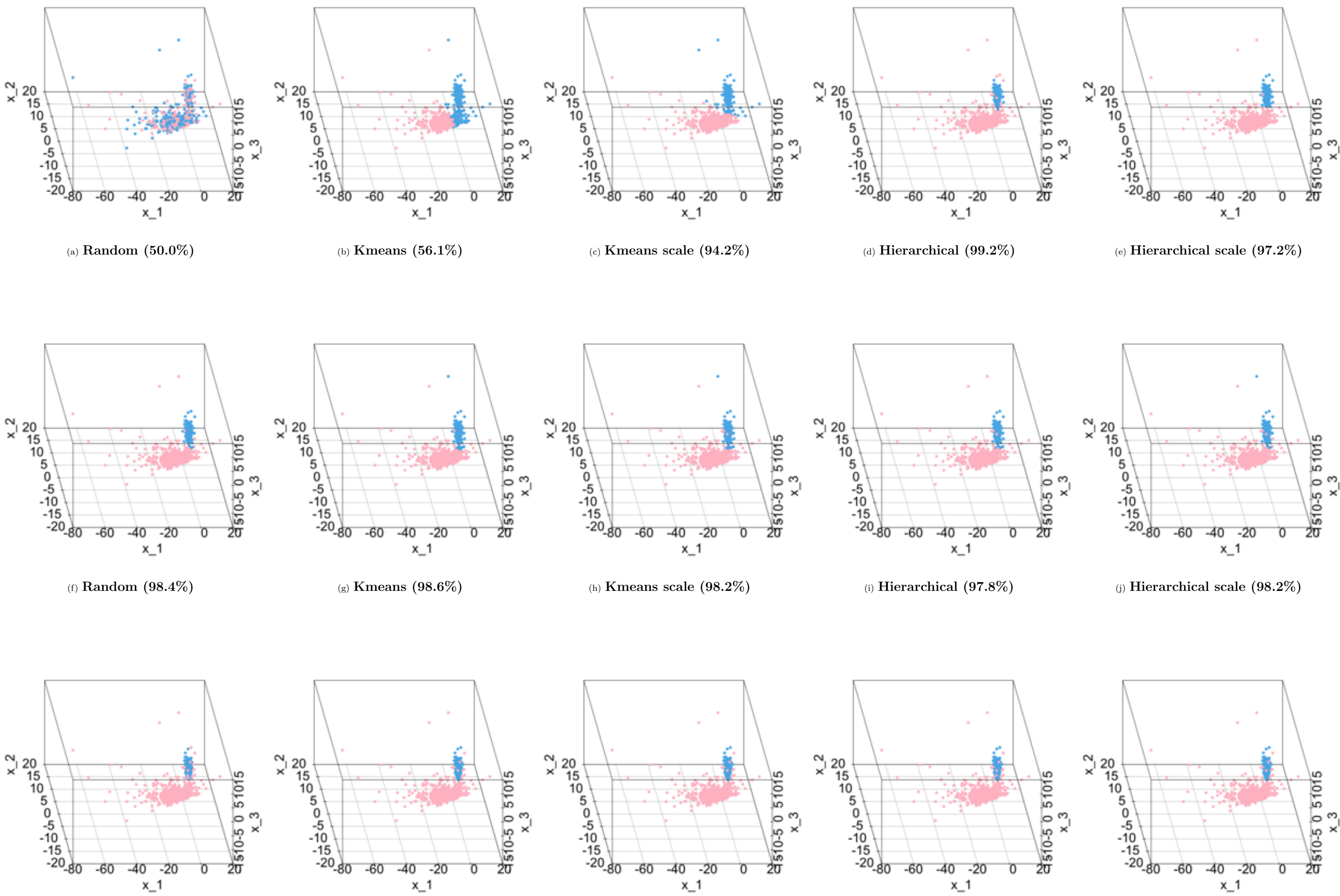
(a) Random (50.0%) (b) Kmeans (99.6%) (c) Kmeans scale (52.0%) (d) Hierarchical (98.4%) (e) Hierarchical scale (89.4%)

(f) Random (99.0%) (g) Kmeans (98.2%) (h) Kmeans scale (99.6%) (i) Hierarchical (99.4%) (j) Hierarchical scale (98.2%)

(k) Random (92.7%) (l) Kmeans (97.2%) (m) Kmeans scale (98.2%) (n) Hierarchical (97.2%) (o) Hierarchical scale (69.4%)

Figure 3.3.17: The scatter plots of initializing clustering and final clustering by using different initialization strategies for Setting 6 with sample size 1000. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

<u>Setting 7, n = 1000</u>



(a) **Random (50.0%)** (b) **Kmeans (56.1%)** (c) **Kmeans scale (94.2%)** (d) **Hierarchical (99.2%)** (e) **Hierarchical scale (97.2%)**

(f) **Random (98.4%)** (g) **Kmeans (98.6%)** (h) **Kmeans scale (98.2%)** (i) **Hierarchical (97.8%)** (j) **Hierarchical scale (98.2%)**

(k) **Random (95.9%)** (l) **Kmeans (96.8%)** (m) **Kmeans scale (96.8%)** (n) **Hierarchical (96.7%)** (o) **Hierarchical scale (96.8%)**

Figure 3.3.18: The scatter plots of initializing clustering and final clustering by using different initialization strategies for Setting 7 with sample size 1000. The blue and pink colour refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster. For final clustering, the EM algorithm reaching the highest classification rate is used here. First row : Initializing clustering, second row : Final clustering by General VCMM and third row : Final clustering by Gaussian VCMM.

The Figure 3.3.5 - 3.3.18 show you the scatter plots of initializing clustering and final clustering by using different initialization strategies for each data setting. In the scatter plots 3.3.2, 3.3.3 and 3.3.4, we know that there are totally no overlapping data in data setting 1 and 8. However, after initializing clustering with Kmeans, the data cannot be clustered accurately and Hierarchical can do better than Kmeans. We think that the reason is that Kmeans cannot cluster the non-Gaussian data well, even if the data are not overlapping. Also, for data setting 1, 5, 7 and 8, the Table 3.3.5 shows the volumes of two clusters are not similar. The initializing clustering with Kmeans (scale) for those data settings can cluster better than Kmeans. In particular, for data 7, the volume of the two clusters are very different and it turns out that the classification rate with Kmeans (scale) is much better than Kmeans.

For the result with the classification rate after final clustering with General VCMM, you can observe that for a particular initialization strategy, the classification rate mostly increases after final clustering, which means the General VCMM algorithm improves the clustering performance after using initialization strategy. It proves that the high flexibility of General VCMM can do clustering better than well-known clustering methods.

2. Comparison of the result for different data settings

The result of the performance measures, including mean computation time, mean normalised BIC and mean classification rate, for each EM algorithm with different initialization strategies are shown in the Table 3.3.9, 3.3.10 and 3.3.11 respectively. We can see that for a particular initialization strategy, the normalised BIC and classification rate perform usually similar within different EM algorithms, but there is a great difference in computation time. In the table 3.3.9, the EM algorithms with heuristic based optimization method (Nelder-Maad) is taking more time than with gradient based optimization method (BFGS) in many situations. This could be due to the fact that VCMM might have a nonlinear objective function with many local optima, making heuristic approaches convergence time longer. Also, for General VCMM, for a particular initialization strategy, ECME.BFGS requires more computation time than ECM.BFGS and MCECM.BFGS.

For a particular EM algorithm, the performance can vary a lot by using different initialization strategies. Therefore, regarding to the accuracy of clustering, the way of initialization for EM algorithm is the key factor. Also, we can see that, in many cases, random definitely requires longer computation times than other methods. After initializing clustering with random, we just can get the 50% accuracy of clustering, which is definitely lower than others. Although it cannot improve after initializing clustering, it still tries to find the best partition while taking too much time.

Also, we would like to analyse and discuss the classification rate of different data settings based on the four characteristics mentioned in the Table 3.3.5, for example, shape, separation, overlap and volume. Because our main goal is to assess the performance for data clustering, we will focus on the classification rate here for assessment.

Separation

For data setting 1, 3, 6, 7, 8 and 9, the two clusters are well separated, where the two clusters can be observed easily even the dots are not marked. You can see that the classification rate with General VCMM and Random can reach over 84%. Although classification rate after initializing cluster is just 50%, the performance after General and Gaussian VCMM turns out to be surprisingly good. Also, in particular, for data setting 1, 8, where the data generated from VCMM model and non-overlapping, although Random performs very well, Hierarchical performs more often better.

Overlap

For data setting 3 and 9, the two clusters are overlapping like a "X". In the Figure 3.3.2 (e) and 3.3.3 (g), you can see that the two clusters are heavily overlapping, but well separated. According to the result of classification rate of General VCMM, random outperforms all other initialization strategies for the "X" cluster. For the data settings with overlapping clusters and not good separation, such as 2, 4 and 5, Kmeans performs the best.

Volume

Recalling in the Example 3.3.1, the volumes of two clusters are not similar for data setting 1, 5, 7 and 8 and the initializing clustering with Kmeans (scale) for those data settings can cluster better than Kmeans. The result in the Table 3.3.11 shows that after final clustering with General VCMM, for data setting 1, 5, 7, the classification rate of Kmeans (scale) is not better than Kmeans. However, for the data setting 7 with very different volumes of clusters, the performances of Kmeans (scale) is better than Kmeans and Hierarchical (scale) is at least as good as Hierarchical.

Shape

The shape of the data settings can be separated into 3 groups, which are elliptical, skew elliptical and non elliptical. For data setting 4, its characteristics is in between elliptical and skew elliptical, because the degrees of freedoms for each cluster are quite high which are 8 and 10 respectively. For a particular EM algorithm and initialization strategy, we can see that the Gaussian VCMM can perform better than General VCMM for data setting 3 and 4 with elliptical cluster. For data setting 1, 2, 8 and 9 with non elliptical cluster, the General VCMM can perform better than Gaussian VCMM. For data setting 5, 6 and 7 with skew elliptical cluster, the General VCMM can perform mostly better than Gaussian VCMM, except data setting 9 (1000 samples) clustered by ECM or MCECM with Kmeans. Also, for data setting 4, 5, 6 generated from mixture of skew t distribution without very different volumes of clusters, Kmeans outperform other initialization strategies.

Summary and recommendation

Here is the summary and recommendation, according to the analysis and discussion above:

- For clustering with VCMM algorithm, the process of parameter estimation by different EM algorithms doesn't change the accuracy much.

- The EM algorithms with Neldar-Mead is taking more time than with BFGS in many situations. For general VCMM, ECM.BFGS or MCECM.BFGS are recommended due to shorter required computation time.

- For non elliptical and skew elliptical data, General VCMM for clustering is performing better than Gaussian VCMM as expected.

- In general, random requires longer computation time.

- For the clusters with good separation, random as an initialization strategy is a good choice for initial clustering. More specifically, for the shape of the clusters like a "X", Random is the best choice. Also, for the data without any overlap, Hierarchical is the best choice.

- For the clusters with overlap and not good separation, Kmeans is suggested.

- For the clusters with very different volumes, standardization first before applying General VCMM is suggested.

- For data generated from skew elliptical with similar / small different volumes of clusters, General VCMM with Kmeans is preferred.

Table 3.3.9: Mean value of the computation time for each EM algorithm and its extension with different initialization strategies for 50 replications, where green and red colour represent the best and the worst initialization strategies for each EM algorithm respectively.

Setting (Sample size)	Performance measures	Algorithm	General VCMM Random	General VCMM Kmeans	General VCMM Hierarchical	General VCMM Kmeans (scale)	General VCMM Hierarchical (scale)	Gaussian VCMM Random	Gaussian VCMM Kmeans	Gaussian VCMM Hierarchical	Gaussian VCMM Kmeans (scale)	Gaussian VCMM Hierarchical (scale)
VCMM (Non elliptical)												
1 (500)	Computation time	ECM.Nelder.Mead	217	124	92	95	97	144	110	77	102	80
		ECM.BFGS	186	97	84	73	77	134	92	67	89	72
		ECME.Nelder.Mead	269	139	154	110	145	87	85	67	69	66
		ECME.BFGS	181	85	117	70	96	73	67	54	59	55
		MCECM.Nelder.Mead	210	124	95	93	103	147	106	77	99	83
		MCECM.BFGS	167	94	84	71	76	137	91	65	91	73
1 (1000)	Computation time	ECM.Nelder.Mead	324	453	130	201	201	235	186	131	175	159
		ECM.BFGS	285	347	113	164	170	211	164	112	155	143
		ECME.Nelder.Mead	370	416	183	209	218	120	124	96	110	106
		ECME.BFGS	307	298	163	165	170	107	105	88	98	95
		MCECM.Nelder.Mead	296	456	128	173	190	236	177	126	163	153
		MCECM.BFGS	285	341	112	160	158	217	159	115	153	138
2 (500)	Computation time	ECM.Nelder.Mead	157	70	71	110	90	143	103	77	94	64
		ECM.BFGS	88	61	74	89	79	108	82	56	74	49
		ECME.Nelder.Mead	180	102	129	177	137	148	99	71	103	64
		ECME.BFGS	98	88	109	144	114	129	84	62	89	57
		MCECM.Nelder.Mead	130	73	72	107	83	146	97	68	95	60
		MCECM.BFGS	85	61	72	88	79	113	83	57	78	50
8 (500)	Computation time	ECM.Nelder.Mead	104	214	59	115	145	64	74	39	71	60
		ECM.BFGS	86	168	47	89	121	53	61	33	59	50
		ECME.Nelder.Mead	117	279	87	150	196	43	61	36	61	55
		ECME.BFGS	105	229	73	118	147	38	52	33	53	48
		MCECM.Nelder.Mead	103	224	55	117	143	62	72	37	69	58
		MCECM.BFGS	441	1105	48	101	121	55	62	33	59	50
9 (500)	Computation time	ECM.Nelder.Mead	123	156	111	167	140	88	88	72	96	80
		ECM.BFGS	98	116	84	118	97	82	75	68	90	74
		ECME.Nelder.Mead	141	206	173	178	170	52	85	57	84	75
		ECME.BFGS	119	150	128	137	124	48	74	53	75	66
		MCECM.Nelder.Mead	117	161	111	169	139	85	84	71	96	76
		MCECM.BFGS	98	115	85	116	100	76	76	61	83	67
VCMM (Multivariate Gaussian)												
3 (500)	Computation time	ECM.Nelder.Mead	114	93	78	67	66	68	79	63	98	82
		ECM.BFGS	121	94	82	67	62	59	73	58	85	70
		ECME.Nelder.Mead	118	79	89	70	66	80	83	70	108	93
		ECME.BFGS	121	77	82	73	62	78	83	72	96	83
		MCECM.Nelder.Mead	104	88	75	64	67	55	73	57	90	75
		MCECM.BFGS	111	93	80	68	64	53	70	55	81	67
3 (1000)	Computation time	ECM.Nelder.Mead	191	153	154	111	153	87	144	99	152	132
		ECM.BFGS	194	161	155	117	151	87	145	99	154	132
		ECME.Nelder.Mead	185	131	147	119	150	77	137	101	132	123
		ECME.BFGS	192	142	148	122	157	76	135	100	137	127
		MCECM.Nelder.Mead	171	155	146	111	152	77	137	95	148	123
		MCECM.BFGS	175	163	148	116	150	76	140	95	150	125
Mixture of multivariate skew t												
4 (500)	Computation time	ECM.Nelder.Mead	161	50	50	50	56	127	47	64	51	65
		ECM.BFGS	158	45	42	42	49	116	43	57	47	59
		ECME.Nelder.Mead	177	55	58	58	66	142	60	76	51	74
		ECME.BFGS	166	46	47	45	54	146	48	64	51	74
		MCECM.Nelder.Mead	166	44	42	44	54	124	42	59	41	65
		MCECM.BFGS	155	42	42	42	50	115	41	60	41	56
5 (500)	Computation time	ECM.Nelder.Mead	360	103	100	92	112	43	55	54	54	60
		ECM.BFGS	297	86	86	77	105	42	54	53	52	59
		ECME.Nelder.Mead	436	113	108	111	127	56	59	60	58	62
		ECME.BFGS	360	100	92	97	120	66	68	70	62	72
		MCECM.Nelder.Mead	324	99	98	87	113	38	52	50	51	56
		MCECM.BFGS	272	88	85	79	103	38	56	54	51	56
5 (1000)	Computation time	ECM.Nelder.Mead	748	159	221	138	217	73	86	86	87	100
		ECM.BFGS	615	138	187	123	206	72	87	85	86	99
		ECME.Nelder.Mead	751	166	215	162	220	85	84	87	84	102
		ECME.BFGS	636	143	188	144	198	99	95	99	89	111
		MCECM.Nelder.Mead	689	155	206	139	217	69	83	80	84	94
		MCECM.BFGS	523	133	178	118	197	64	79	77	79	89
6 (500)	Computation time	ECM.Nelder.Mead	233	49	63	50	101	57	22	28	28	28
		ECM.BFGS	204	40	54	42	84	55	21	26	25	26
		ECME.Nelder.Mead	215	73	93	71	119	77	31	34	37	35
		ECME.BFGS	187	59	72	59	99	73	28	32	38	34
		MCECM.Nelder.Mead	228	50	65	51	101	51	23	27	28	28
		MCECM.BFGS	200	41	53	43	87	50	21	25	23	27
6 (1000)	Computation time	ECM.Nelder.Mead	495	92	135	112	163	79	32	47	34	43
		ECM.BFGS	364	65	112	85	140	75	30	44	35	43
		ECME.Nelder.Mead	345	137	182	161	194	87	40	46	46	46
		ECME.BFGS	294	119	175	137	177	96	43	49	46	49
		MCECM.Nelder.Mead	445	94	141	108	168	70	30	42	33	45
		MCECM.BFGS	361	79	143	101	152	73	38	44	33	42
7 (500)	Computation time	ECM.Nelder.Mead	265	125	47	85	45	145	56	18	37	17
		ECM.BFGS	214	87	38	69	36	128	52	17	25	14
		ECME.Nelder.Mead	294	185	72	119	105	75	48	25	33	22
		ECME.BFGS	249	146	57	94	52	72	45	25	36	22
		MCECM.Nelder.Mead	269	123	47	82	45	141	57	18	29	16
		MCECM.BFGS	224	90	38	71	36	128	51	17	25	14
7 (1000)	Computation time	ECM.Nelder.Mead	465	736	344	545	349	248	1004	87	44	25
		ECM.BFGS	315	206	72	133	61	239	92	31	42	24
		ECME.Nelder.Mead	1166	1536	565	176	106	97	66	35	47	28
		ECME.BFGS	412	552	448	639	329	259	645	372	45	27
		MCECM.Nelder.Mead	517	679	362	540	278	248	970	247	44	24
		MCECM.BFGS	343	258	73	139	71	1014	147	149	155	88

Table 3.3.10: Mean value of the normalised BIC for each EM algorithm and its extension with different initialization strategies for 50 replications, where green and red colour represent the best and the worst initialization strategies for each EM algorithm respectively.

Setting (Sample size)	Performance measures	Algorithm	General VCMM Random	General VCMM Kmeans	General VCMM Hierarchical	General VCMM Kmeans (scale)	General VCMM Hierarchical (scale)	Gaussian VCMM Random	Gaussian VCMM Kmeans	Gaussian VCMM Hierarchical	Gaussian VCMM Kmeans (scale)	Gaussian VCMM Hierarchical (scale)
VCMM (Non elliptical)												
1 (500)	Normalised BIC	ECM.Nelder.Mead	16.76	17.98	16.59	17.99	17.74	20.03	19.95	19.74	19.90	19.82
		ECM.BFGS	16.70	17.99	16.57	17.99	17.70	20.00	19.95	19.74	19.91	19.82
		ECME.Nelder.Mead	16.74	17.97	16.57	18.01	17.77	19.92	19.93	19.70	19.90	19.79
		ECME.BFGS	16.74	18.02	16.54	18.02	17.69	19.97	19.94	19.70	19.90	19.81
		MCECM.Nelder.Mead	16.79	17.98	16.59	17.98	17.76	20.01	19.94	19.74	19.90	19.82
		MCECM.BFGS	16.74	17.99	16.57	17.98	17.70	20.01	19.94	19.74	19.90	19.82
1 (1000)	Normalised BIC	ECM.Nelder.Mead	16.50	17.22	16.50	17.80	17.71	19.93	19.88	19.66	19.86	19.83
		ECM.BFGS	16.50	17.28	16.48	17.72	17.70	19.93	19.88	19.68	19.90	19.86
		ECME.Nelder.Mead	16.49	17.53	16.51	17.82	17.78	19.87	19.88	19.66	19.86	19.83
		ECME.BFGS	16.49	17.51	16.48	17.76	17.74	19.92	19.92	19.66	19.89	19.86
		MCECM.Nelder.Mead	16.53	17.22	16.49	17.82	17.71	19.94	19.88	19.66	19.86	19.84
		MCECM.BFGS	16.48	17.28	16.48	17.72	17.70	19.96	19.90	19.68	19.88	19.88
2 (500)	Normalised BIC	ECM.Nelder.Mead	13.96	13.15	13.17	13.24	13.18	14.67	14.24	14.31	14.52	14.28
		ECM.BFGS	13.97	13.16	13.16	13.24	13.17	14.67	14.24	14.31	14.52	14.28
		ECME.Nelder.Mead	13.78	13.13	13.14	13.18	13.19	14.69	14.23	14.31	14.50	14.28
		ECME.BFGS	13.88	13.12	13.16	13.24	13.17	14.69	14.23	14.31	14.50	14.28
		MCECM.Nelder.Mead	13.98	13.15	13.16	13.24	13.18	14.65	14.24	14.31	14.52	14.28
		MCECM.BFGS	13.98	13.16	13.16	13.24	13.17	14.66	14.24	14.31	14.52	14.28
8 (500)	Normalised BIC	ECM.Nelder.Mead	17.67	18.69	17.26	18.92	18.40	19.92	19.92	19.90	19.96	19.95
		ECM.BFGS	17.67	18.68	17.26	18.95	18.36	19.92	19.92	19.90	19.96	19.95
		ECME.Nelder.Mead	17.60	18.72	17.26	18.96	18.36	19.91	19.91	19.90	19.96	19.95
		ECME.BFGS	17.56	18.65	17.26	18.92	18.41	19.91	19.91	19.90	19.96	19.96
		MCECM.Nelder.Mead	17.73	18.68	17.26	18.91	18.42	19.92	19.92	19.90	19.96	19.95
		MCECM.BFGS	17.73	18.67	17.26	18.89	18.38	19.92	19.92	19.90	19.96	19.95
9 (500)	Normalised BIC	ECM.Nelder.Mead	14.65	16.00	15.32	15.46	15.59	16.68	17.35	16.83	16.99	17.05
		ECM.BFGS	14.60	15.99	15.32	15.46	15.62	16.68	17.37	16.83	16.99	17.05
		ECME.Nelder.Mead	14.57	15.95	15.20	15.67	15.55	16.56	17.28	16.80	16.98	17.00
		ECME.BFGS	14.58	15.95	15.24	15.49	15.50	16.61	17.28	16.80	16.98	17.00
		MCECM.Nelder.Mead	14.65	15.95	15.32	15.46	15.59	16.73	17.35	16.83	17.02	17.08
		MCECM.BFGS	14.58	15.99	15.32	15.50	15.61	16.73	17.35	16.86	17.02	17.08
VCMM (Multivariate Gaussian)												
3 (500)	Normalised BIC	ECM.Nelder.Mead	12.10	12.39	12.30	12.41	12.38	12.01	12.14	12.06	12.16	12.14
		ECM.BFGS	12.10	12.40	12.30	12.41	12.38	12.01	12.14	12.06	12.16	12.14
		ECME.Nelder.Mead	12.13	12.38	12.29	12.40	12.38	12.06	12.15	12.06	12.15	12.14
		ECME.BFGS	12.12	12.38	12.29	12.40	12.38	12.06	12.15	12.06	12.15	12.14
		MCECM.Nelder.Mead	12.08	12.39	12.30	12.41	12.37	12.01	12.14	12.06	12.16	12.14
		MCECM.BFGS	12.08	12.39	12.30	12.41	12.38	12.01	12.14	12.06	12.16	12.14
3 (1000)	Normalised BIC	ECM.Nelder.Mead	12.01	12.32	12.18	12.32	12.28	11.95	12.06	12.00	12.07	12.02
		ECM.BFGS	12.01	12.32	12.18	12.31	12.27	11.95	12.06	12.00	12.07	12.02
		ECME.Nelder.Mead	12.04	12.31	12.17	12.30	12.26	11.99	12.08	12.00	12.08	12.02
		ECME.BFGS	12.04	12.31	12.17	12.30	12.26	11.99	12.08	12.00	12.08	12.02
		MCECM.Nelder.Mead	12.01	12.32	12.18	12.31	12.27	11.95	12.06	12.00	12.07	12.02
		MCECM.BFGS	12.01	12.32	12.18	12.32	12.27	11.95	12.06	12.00	12.07	12.02
Mixture of multivariate skew t												
4 (500)	Normalised BIC	ECM.Nelder.Mead	10.97	10.79	10.81	10.78	10.80	10.84	10.78	10.78	10.77	10.78
		ECM.BFGS	10.97	10.79	10.81	10.79	10.80	10.84	10.77	10.78	10.77	10.78
		ECME.Nelder.Mead	10.97	10.79	10.81	10.79	10.80	10.84	10.77	10.78	10.77	10.78
		ECME.BFGS	10.97	10.79	10.80	10.79	10.80	10.84	10.77	10.78	10.77	10.78
		MCECM.Nelder.Mead	10.97	10.79	10.81	10.78	10.80	10.84	10.78	10.78	10.77	10.78
		MCECM.BFGS	10.97	10.79	10.81	10.78	10.80	10.84	10.77	10.78	10.77	10.78
5 (500)	Normalised BIC	ECM.Nelder.Mead	12.16	12.08	12.10	12.08	12.08	12.12	12.12	12.10	12.12	12.12
		ECM.BFGS	12.17	12.08	12.10	12.08	12.08	12.13	12.12	12.10	12.12	12.12
		ECME.Nelder.Mead	12.17	12.08	12.10	12.08	12.09	12.12	12.12	12.11	12.12	12.13
		ECME.BFGS	12.17	12.08	12.10	12.08	12.08	12.12	12.12	12.11	12.12	12.13
		MCECM.Nelder.Mead	12.17	12.08	12.10	12.08	12.08	12.13	12.12	12.10	12.12	12.12
		MCECM.BFGS	12.17	12.08	12.10	12.08	12.09	12.13	12.12	12.10	12.12	12.12
5 (1000)	Normalised BIC	ECM.Nelder.Mead	12.09	12.02	12.05	12.02	12.01	12.12	12.12	12.12	12.12	12.12
		ECM.BFGS	12.09	12.02	12.05	12.02	12.01	12.12	12.12	12.12	12.12	12.12
		ECME.Nelder.Mead	12.08	12.01	12.05	12.02	12.01	12.12	12.12	12.12	12.12	12.12
		ECME.BFGS	12.08	12.01	12.04	12.01	12.01	12.12	12.12	12.12	12.12	12.12
		MCECM.Nelder.Mead	12.10	12.02	12.05	12.02	12.01	12.12	12.12	12.12	12.12	12.12
		MCECM.BFGS	12.10	12.01	12.05	12.02	12.01	12.12	12.12	12.12	12.12	12.12
6 (500)	Normalised BIC	ECM.Nelder.Mead	13.49	13.32	13.41	13.38	13.38	14.36	14.35	14.36	14.37	14.42
		ECM.BFGS	13.47	13.32	13.41	13.38	13.38	14.36	14.35	14.36	14.37	14.42
		ECME.Nelder.Mead	13.47	13.32	13.41	13.38	13.38	14.38	14.35	14.36	14.36	14.42
		ECME.BFGS	13.53	13.32	13.41	13.38	13.38	14.38	14.34	14.36	14.35	14.42
		MCECM.Nelder.Mead	13.48	13.32	13.41	13.38	13.38	14.37	14.35	14.36	14.36	14.42
		MCECM.BFGS	13.47	13.32	13.41	13.38	13.38	14.37	14.35	14.36	14.37	14.42
6 (1000)	Normalised BIC	ECM.Nelder.Mead	13.35	13.26	13.29	13.30	13.30	14.41	14.44	14.47	14.46	14.50
		ECM.BFGS	13.33	13.26	13.29	13.31	13.30	14.41	14.46	14.47	14.44	14.50
		ECME.Nelder.Mead	13.34	13.26	13.29	13.31	13.30	14.41	14.44	14.47	14.46	14.50
		ECME.BFGS	13.36	13.26	13.29	13.30	13.30	14.41	14.45	14.47	14.44	14.50
		MCECM.Nelder.Mead	13.36	13.26	13.29	13.31	13.30	14.41	14.44	14.47	14.46	14.50
		MCECM.BFGS	13.35	13.26	13.29	13.31	13.30	14.41	14.46	14.47	14.44	14.50
7 (500)	Normalised BIC	ECM.Nelder.Mead	14.11	14.40	13.92	13.92	13.93	14.76	14.80	14.70	14.78	14.70
		ECM.BFGS	14.07	14.41	13.92	13.92	13.93	14.75	14.80	14.70	14.78	14.70
		ECME.Nelder.Mead	14.10	14.06	13.92	13.92	13.93	14.74	14.76	14.70	14.74	14.70
		ECME.BFGS	14.11	14.06	13.92	13.92	13.93	14.74	14.74	14.70	14.74	14.70
		MCECM.Nelder.Mead	14.09	14.38	13.92	13.92	13.93	14.76	14.80	14.70	14.78	14.70
		MCECM.BFGS	14.04	14.41	13.92	13.92	13.92	14.76	14.80	14.70	14.78	14.70
7 (1000)	Normalised BIC	ECM.Nelder.Mead	13.84	14.06	13.79	13.82	13.79	14.73	14.72	14.78	14.86	14.70
		ECM.BFGS	13.83	14.06	13.80	13.82	13.79	14.73	14.72	14.78	14.82	14.70
		ECME.Nelder.Mead	13.84	13.83	13.79	13.82	13.79	14.70	14.72	14.78	14.76	14.70
		ECME.BFGS	13.90	13.85	13.80	13.82	13.79	14.67	14.72	14.78	14.76	14.70
		MCECM.Nelder.Mead	13.85	14.06	13.79	13.82	13.79	14.73	14.72	14.78	14.85	14.70
		MCECM.BFGS	13.82	14.06	13.80	13.82	13.79	14.73	14.72	14.78	14.82	14.70

Table 3.3.11: Mean value of the classification rate for each EM algorithm and its extension with different initialization strategies for 50 replications, where green and red colour represent the best and the worst initialization strategies for each EM algorithm respectively. Remark : the classification rate close to the best value (within 3%) is highlighted as orange .

Setting (Sample size)	Performance measures	Algorithm	General VCMM Random	General VCMM Kmeans	General VCMM Hierarchical	General VCMM Kmeans (scale)	General VCMM Hierarchical (scale)	Gaussian VCMM Random	Gaussian VCMM Kmeans	Gaussian VCMM Hierarchical	Gaussian VCMM Kmeans (scale)	Gaussian VCMM Hierarchical (scale)
Data generation : VCMM (Non elliptical)												
1 (500)	Classification rate	ECM.Nelder.Mead	0.872	0.597	0.934	0.610	0.637	0.592	0.599	0.724	0.589	0.611
		ECM.BFGS	0.885	0.596	0.942	0.610	0.647	0.594	0.598	0.724	0.589	0.612
		ECME.Nelder.Mead	0.879	0.602	0.940	0.601	0.628	0.604	0.606	0.730	0.593	0.617
		ECME.BFGS	0.875	0.584	0.944	0.600	0.647	0.620	0.603	0.725	0.593	0.615
		MCECM.Nelder.Mead	0.868	0.597	0.937	0.611	0.634	0.587	0.598	0.723	0.590	0.613
		MCECM.BFGS	0.880	0.596	0.943	0.610	0.649	0.588	0.597	0.723	0.590	0.613
1 (1000)	Classification rate	ECM.Nelder.Mead	0.931	0.778	0.924	0.628	0.630	0.582	0.601	0.735	0.589	0.586
		ECM.BFGS	0.930	0.765	0.927	0.644	0.634	0.582	0.601	0.736	0.586	0.584
		ECME.Nelder.Mead	0.930	0.709	0.923	0.628	0.619	0.596	0.605	0.735	0.595	0.590
		ECME.BFGS	0.932	0.709	0.927	0.640	0.623	0.602	0.601	0.735	0.592	0.588
		MCECM.Nelder.Mead	0.923	0.778	0.924	0.626	0.631	0.586	0.601	0.735	0.588	0.586
		MCECM.BFGS	0.935	0.767	0.927	0.644	0.634	0.584	0.601	0.736	0.587	0.583
2 (500)	Classification rate	ECM.Nelder.Mead	0.627	0.894	0.865	0.861	0.863	0.532	0.811	0.765	0.621	0.777
		ECM.BFGS	0.625	0.889	0.871	0.863	0.870	0.532	0.811	0.765	0.621	0.777
		ECME.Nelder.Mead	0.659	0.889	0.870	0.870	0.856	0.519	0.808	0.764	0.621	0.775
		ECME.BFGS	0.645	0.889	0.871	0.863	0.869	0.519	0.807	0.765	0.621	0.775
		MCECM.Nelder.Mead	0.621	0.894	0.866	0.862	0.863	0.546	0.811	0.764	0.616	0.776
		MCECM.BFGS	0.619	0.889	0.871	0.863	0.870	0.544	0.811	0.764	0.617	0.776
8 (500)	Classification rate	ECM.Nelder.Mead	0.862	0.650	0.991	0.608	0.732	0.907	0.905	0.899	0.892	0.888
		ECM.BFGS	0.861	0.654	0.990	0.602	0.745	0.907	0.905	0.899	0.892	0.888
		ECME.Nelder.Mead	0.900	0.648	0.990	0.603	0.752	0.870	0.881	0.893	0.874	0.867
		ECME.BFGS	0.912	0.658	0.990	0.611	0.743	0.866	0.882	0.892	0.874	0.862
		MCECM.Nelder.Mead	0.843	0.654	0.990	0.607	0.726	0.906	0.905	0.900	0.892	0.887
		MCECM.BFGS	0.843	0.656	0.990	0.614	0.738	0.906	0.905	0.900	0.892	0.887
9 (500)	Classification rate	ECM.Nelder.Mead	0.847	0.564	0.699	0.756	0.685	0.722	0.576	0.645	0.634	0.606
		ECM.BFGS	0.860	0.562	0.700	0.756	0.678	0.722	0.575	0.645	0.634	0.606
		ECME.Nelder.Mead	0.877	0.569	0.728	0.731	0.698	0.664	0.580	0.639	0.622	0.608
		ECME.BFGS	0.878	0.573	0.720	0.757	0.708	0.690	0.580	0.639	0.620	0.608
		MCECM.Nelder.Mead	0.847	0.569	0.699	0.757	0.685	0.726	0.576	0.645	0.631	0.603
		MCECM.BFGS	0.867	0.561	0.700	0.750	0.679	0.726	0.576	0.637	0.631	0.603
Data generation : VCMM (Multivariate Gaussian)												
3 (500)	Classification rate	ECM.Nelder.Mead	0.821	0.575	0.678	0.537	0.560	0.857	0.756	0.824	0.743	0.759
		ECM.BFGS	0.821	0.570	0.677	0.537	0.558	0.857	0.756	0.824	0.743	0.759
		ECME.Nelder.Mead	0.792	0.573	0.678	0.535	0.557	0.820	0.746	0.824	0.747	0.756
		ECME.BFGS	0.799	0.572	0.680	0.536	0.557	0.821	0.746	0.824	0.747	0.756
		MCECM.Nelder.Mead	0.835	0.575	0.677	0.537	0.563	0.865	0.756	0.824	0.740	0.760
		MCECM.BFGS	0.835	0.571	0.677	0.537	0.560	0.865	0.756	0.824	0.741	0.760
3 (1000)	Classification rate	ECM.Nelder.Mead	0.850	0.558	0.713	0.528	0.587	0.877	0.787	0.844	0.772	0.825
		ECM.BFGS	0.850	0.559	0.713	0.528	0.587	0.876	0.787	0.844	0.772	0.825
		ECME.Nelder.Mead	0.816	0.550	0.714	0.530	0.593	0.839	0.770	0.844	0.768	0.826
		ECME.BFGS	0.816	0.550	0.714	0.529	0.593	0.839	0.770	0.844	0.768	0.825
		MCECM.Nelder.Mead	0.851	0.559	0.713	0.528	0.588	0.877	0.787	0.844	0.776	0.825
		MCECM.BFGS	0.851	0.559	0.713	0.528	0.587	0.877	0.787	0.844	0.776	0.825
Data generation : Mixture of multivariate skew t												
4 (500)	Classification rate	ECM.Nelder.Mead	0.535	0.826	0.796	0.805	0.774	0.578	0.833	0.818	0.829	0.807
		ECM.BFGS	0.535	0.826	0.797	0.804	0.774	0.578	0.827	0.814	0.825	0.803
		ECME.Nelder.Mead	0.538	0.824	0.796	0.803	0.772	0.590	0.826	0.811	0.824	0.794
		ECME.BFGS	0.541	0.825	0.796	0.804	0.774	0.590	0.826	0.812	0.825	0.793
		MCECM.Nelder.Mead	0.535	0.826	0.796	0.805	0.774	0.568	0.831	0.818	0.829	0.803
		MCECM.BFGS	0.535	0.826	0.796	0.805	0.774	0.568	0.826	0.811	0.825	0.803
5 (500)	Classification rate	ECM.Nelder.Mead	0.556	0.755	0.677	0.746	0.705	0.520	0.521	0.522	0.522	0.521
		ECM.BFGS	0.557	0.756	0.678	0.746	0.707	0.520	0.522	0.522	0.522	0.521
		ECME.Nelder.Mead	0.560	0.753	0.679	0.746	0.705	0.521	0.522	0.522	0.522	0.523
		ECME.BFGS	0.563	0.753	0.679	0.746	0.708	0.521	0.522	0.522	0.522	0.523
		MCECM.Nelder.Mead	0.545	0.756	0.677	0.747	0.708	0.520	0.521	0.522	0.522	0.521
		MCECM.BFGS	0.542	0.755	0.677	0.746	0.707	0.520	0.521	0.522	0.522	0.521
5 (1000)	Classification rate	ECM.Nelder.Mead	0.601	0.756	0.664	0.744	0.701	0.520	0.520	0.520	0.520	0.520
		ECM.BFGS	0.602	0.756	0.662	0.744	0.700	0.520	0.520	0.520	0.520	0.520
		ECME.Nelder.Mead	0.624	0.754	0.664	0.742	0.702	0.520	0.521	0.520	0.521	0.520
		ECME.BFGS	0.626	0.754	0.663	0.742	0.703	0.520	0.521	0.520	0.521	0.520
		MCECM.Nelder.Mead	0.574	0.757	0.662	0.744	0.701	0.520	0.521	0.520	0.521	0.520
		MCECM.BFGS	0.573	0.756	0.662	0.744	0.700	0.520	0.521	0.520	0.521	0.520
6 (500)	Classification rate	ECM.Nelder.Mead	0.905	0.987	0.939	0.958	0.960	0.599	0.970	0.856	0.927	0.835
		ECM.BFGS	0.918	0.987	0.939	0.958	0.960	0.599	0.970	0.856	0.927	0.835
		ECME.Nelder.Mead	0.918	0.987	0.939	0.958	0.961	0.556	0.969	0.855	0.926	0.832
		ECME.BFGS	0.897	0.987	0.939	0.958	0.960	0.546	0.969	0.855	0.925	0.832
		MCECM.Nelder.Mead	0.912	0.987	0.939	0.958	0.960	0.615	0.969	0.855	0.927	0.833
		MCECM.BFGS	0.914	0.987	0.939	0.958	0.960	0.615	0.969	0.856	0.927	0.833
6 (1000)	Classification rate	ECM.Nelder.Mead	0.948	0.988	0.973	0.974	0.974	0.554	0.971	0.876	0.929	0.840
		ECM.BFGS	0.959	0.988	0.973	0.968	0.974	0.554	0.970	0.876	0.927	0.840
		ECME.Nelder.Mead	0.955	0.988	0.973	0.973	0.974	0.528	0.971	0.881	0.931	0.837
		ECME.BFGS	0.949	0.988	0.973	0.974	0.975	0.528	0.970	0.880	0.929	0.837
		MCECM.Nelder.Mead	0.945	0.988	0.973	0.968	0.974	0.555	0.971	0.882	0.929	0.839
		MCECM.BFGS	0.950	0.988	0.973	0.968	0.974	0.554	0.970	0.881	0.927	0.839
7 (500)	Classification rate	ECM.Nelder.Mead	0.937	0.834	0.974	0.980	0.971	0.909	0.947	0.954	0.953	0.954
		ECM.BFGS	0.949	0.830	0.974	0.980	0.972	0.909	0.947	0.954	0.953	0.954
		ECME.Nelder.Mead	0.946	0.939	0.975	0.980	0.972	0.918	0.939	0.954	0.951	0.954
		ECME.BFGS	0.946	0.935	0.975	0.980	0.972	0.918	0.940	0.954	0.951	0.954
		MCECM.Nelder.Mead	0.946	0.840	0.974	0.980	0.972	0.909	0.946	0.954	0.953	0.954
		MCECM.BFGS	0.958	0.830	0.975	0.980	0.972	0.909	0.946	0.954	0.953	0.954
7 (1000)	Classification rate	ECM.Nelder.Mead	0.973	0.898	0.980	0.974	0.980	0.938	0.958	0.950	0.952	0.957
		ECM.BFGS	0.974	0.899	0.978	0.974	0.980	0.938	0.958	0.950	0.951	0.957
		ECME.Nelder.Mead	0.974	0.969	0.980	0.974	0.980	0.947	0.958	0.950	0.950	0.956
		ECME.BFGS	0.960	0.960	0.978	0.974	0.980	0.947	0.958	0.951	0.950	0.956
		MCECM.Nelder.Mead	0.973	0.898	0.979	0.974	0.980	0.938	0.958	0.950	0.951	0.956
		MCECM.BFGS	0.976	0.899	0.978	0.974	0.980	0.938	0.957	0.950	0.950	0.956

3. Paired t test for the comparison between General VCMM and Gaussian VCMM

In the last summary part, we have recommended using General VCMM over Gaussian VCMM for clustering non-elliptical and skew elliptical data. Now, we would like to use paired t test for normalised BIC and classification rate to prove our statement with some statistical evidences and the hypotheses are formally defined below:

$H_0 : \Delta \leq 0$ versus $H_1 : \Delta > 0$
For classification rate :
Δ = General VCMM - Gaussian VCMM
For normalised BIC :
Δ = Gaussian VCMM - General VCMM

The Table 3.3.12 shows the p value of the paired t test based on independent replications for the difference in the performance measures normalised BIC, classification rate between General VCMM and Gaussian VCMM. P value close to 0 means General VCMM better than Gaussian VCMM. The value smaller than 0.05 is highlighted with green colour. Other than data setting 3 and 4 with elliptical clusters, the p values for normalised BIC or classification rate are mostly smaller than 0.05, which means General VCMM is more preferred. This is what we recommended before. However, the p values for classification rate in data setting 8 are a bit weird. Other than Hierarchical, the paired t test with p values close to 1 suggests that there is no difference between General VCMM and Gaussian VCMM. Although we focus more on classification rate for clustering, in general, General VCMM still outperforms Gaussian VCMM in many cases and we cannot know the characteristics of each cluster easily. So, General VCMM is still recommended.

Table 3.3.12: The p-value of the one sided paired t-test comparing the mean of performance measures by using General VCMM and Gaussian VCMM. green indicates the p-value smaller than 0.05 that we can conclude the performance using General VCMM is better than Gaussian VCMM with significance level of 0.05.

Setting (Sample size)	Algorithm	Normalised BIC: Random	Normalised BIC: Kmeans	Normalised BIC: Hierarchical	Normalised BIC: Kmeans (scale)	Normalised BIC: Hierarchical (scale)	Classification rate: Random	Classification rate: Kmeans	Classification rate: Hierarchical	Classification rate: Kmeans (scale)	Classification rate: Hierarchical (scale)
Data generation : VCMM (Non elliptical)											
1 (500)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.560	0.000	0.030	0.074
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.551	0.000	0.026	0.030
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.589	0.000	0.223	0.216
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.909	0.000	0.255	0.047
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.515	0.000	0.028	0.105
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.512	0.000	0.028	0.027
1 (1000)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.056	0.052
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.003	0.022
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.086	0.106
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.008	0.058
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.054	0.051
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.005	0.022
2 (500)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
8 (500)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.937	1.000	0.000	1.000	1.000
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.940	1.000	0.000	1.000	1.000
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.128	1.000	0.000	1.000	1.000
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.034	1.000	0.000	1.000	1.000
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.980	1.000	0.000	1.000	1.000
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.980	1.000	0.000	1.000	1.000
9 (500)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.732	0.005	0.000	0.002
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.752	0.004	0.000	0.003
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.714	0.000	0.000	0.000
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.630	0.000	0.000	0.000
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.633	0.005	0.000	0.001
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.781	0.001	0.000	0.002
Data generation : VCMM (Multivariate Gaussian)											
3 (500)	ECM.Nelder.Mead	1.000	1.000	1.000	1.000	1.000	0.977	1.000	1.000	1.000	1.000
	ECM.BFGS	1.000	1.000	1.000	1.000	1.000	0.977	1.000	1.000	1.000	1.000
	ECME.Nelder.Mead	0.999	1.000	1.000	1.000	1.000	0.922	1.000	1.000	1.000	1.000
	ECME.BFGS	0.989	1.000	1.000	1.000	1.000	0.833	1.000	1.000	1.000	1.000
	MCECM.Nelder.Mead	1.000	1.000	1.000	1.000	1.000	0.990	1.000	1.000	1.000	1.000
	MCECM.BFGS	1.000	1.000	1.000	1.000	1.000	0.991	1.000	1.000	1.000	1.000
3 (1000)	ECM.Nelder.Mead	1.000	1.000	1.000	1.000	1.000	0.977	1.000	1.000	1.000	1.000
	ECM.BFGS	1.000	1.000	1.000	1.000	1.000	0.978	1.000	1.000	1.000	1.000
	ECME.Nelder.Mead	0.966	1.000	1.000	1.000	1.000	0.841	1.000	1.000	1.000	1.000
	ECME.BFGS	0.966	1.000	1.000	1.000	1.000	0.843	1.000	1.000	1.000	1.000
	MCECM.Nelder.Mead	1.000	1.000	1.000	1.000	1.000	0.978	1.000	1.000	1.000	1.000
	MCECM.BFGS	1.000	1.000	1.000	1.000	1.000	0.978	1.000	1.000	1.000	1.000
Data generation : Mixture of multivariate skew t											
4 (500)	ECM.Nelder.Mead	1.000	0.985	0.997	0.874	0.981	1.000	0.849	0.994	1.000	1.000
	ECM.BFGS	1.000	0.986	0.998	0.910	0.991	1.000	0.555	0.944	0.984	0.994
	ECME.Nelder.Mead	1.000	0.967	0.998	0.908	0.992	1.000	0.564	0.911	0.986	0.964
	ECME.BFGS	1.000	0.967	0.996	0.922	0.987	1.000	0.559	0.914	0.982	0.950
	MCECM.Nelder.Mead	1.000	0.988	0.998	0.845	0.989	1.000	0.731	0.992	1.000	0.993
	MCECM.BFGS	1.000	0.990	0.998	0.853	0.982	1.000	0.506	0.896	0.983	0.993
5 (500)	ECM.Nelder.Mead	0.997	0.000	0.385	0.001	0.004	0.002	0.000	0.000	0.000	0.000
	ECM.BFGS	0.991	0.000	0.338	0.000	0.002	0.003	0.000	0.000	0.000	0.000
	ECME.Nelder.Mead	1.000	0.000	0.302	0.000	0.004	0.001	0.000	0.000	0.000	0.000
	ECME.BFGS	1.000	0.000	0.309	0.000	0.002	0.000	0.000	0.000	0.000	0.000
	MCECM.Nelder.Mead	0.999	0.000	0.362	0.001	0.002	0.015	0.000	0.000	0.000	0.000
	MCECM.BFGS	1.000	0.000	0.373	0.001	0.004	0.021	0.000	0.000	0.000	0.000
5 (1000)	ECM.Nelder.Mead	0.007	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	ECM.BFGS	0.019	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	ECME.Nelder.Mead	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	ECME.BFGS	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	MCECM.Nelder.Mead	0.107	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	MCECM.BFGS	0.135	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
6 (500)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010	0.000
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010	0.000
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.008	0.000
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.008	0.000
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010	0.000
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.009	0.000
6 (1000)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.003	0.000
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.000
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.000
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.007	0.000
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.000
7 (500)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.119	1.000	0.000	0.003	0.000
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.033	1.000	0.000	0.003	0.000
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.065	0.533	0.000	0.002	0.000
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.060	0.688	0.000	0.002	0.000
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.041	1.000	0.000	0.003	0.000
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.006	1.000	0.000	0.003	0.000
7 (1000)	ECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.998	0.000	0.000	0.000
	ECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.998	0.000	0.000	0.000
	ECME.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.009	0.000	0.000	0.000
	ECME.BFGS	0.000	0.000	0.000	0.000	0.000	0.055	0.373	0.000	0.000	0.000
	MCECM.Nelder.Mead	0.000	0.000	0.000	0.000	0.000	0.000	0.998	0.000	0.000	0.000
	MCECM.BFGS	0.000	0.000	0.000	0.000	0.000	0.000	0.998	0.000	0.000	0.000

4. Ranking of the performance for each EM algorithm and initialization strategy

The full tables for the ranking of the performance of EM algorithms and the initialization strategies for the nine data set by General VCMM and Gaussian VCMM algorithm are shown in the Appendix B.6. For better and more convenient comparison, four subsets of the ranking table of the data setting with similar characteristics are shown in this part, where the Table 3.3.13 shows the ranking of data setting 4, 5, 6 (Multivariate skew t distribution (Two clusters without big volume difference)), the Table 3.3.14 shows the ranking of data setting 3 and 9 (Well separated with the "X" shape), the Table 3.3.15 shows the ranking of data setting 1 and 8 (Well separated without overlap (Except Gaussian and skew t distribution)) and the last Table 3.3.16 shows the ranking of data setting 7 (Two clusters with big volume difference).

Data setting 4, 5, 6 (Multivariate skew t distribution (Two clusters without big volume difference)

The Table 3.3.13 shows Kmeans outperforms other initialization strategies with data following skew t distribution. For General VCMM, the performances of different EM algorithms with Kmeans are similarly very good and MCECM.Nelder.Mead with Kmeans reaches the highest classification rate in that 5 simulations. Also, the characteristics of data setting 4 is in between elliptical and skew elliptical. The Gaussian VCMM with ECM.Nelder.Mead and Kemans performs the best with 83.3% classification rate, which is higher than the classification rate 82.6% for General VCMM with the best method.

Data setting 3 and 9 (Well separated with the "X" shape)

The Table 3.3.14 shows Random outperforms other initialization strategies for those clusters like a "X" shape. Recalling that data setting 3 is Gaussian data. The classification rate for Gaussian VCMM with the best method can be higher than General VCMM. We can see that for both General VCMM and Gaussian VCMM, MCECM.Nelder.Mead and MCECM.BFGS perform the best. For data setting 9 with non-Gaussian data, ECME.BFGS with Random for General VCMM perform the best.

Data setting 1 and 8 (Well separated without overlap (Except Gaussian and skew t distribution))

The Table 3.3.15 shows the EM algorithms with Random or Hierarchical are the top 12 performance for data setting 1 and 8. In particular, EM algorithms with Hierarchical perform the best data setting 1 with 500 sample sizes and data setting 8. And EM algorithms with Random perform the best data setting 1 with 1000 sample size. Basically, we don't have a best method to cluster all data in this group. However, you can see that the difference of classification rate between different EM algorithms with Random and MCECM.BFGS with Hierarchical in data setting 1 with 1000 sample sizes are smaller than 1% only. So, we will suggest using Hierarchical to cluster data in this group. Comparing the classification rate with different EM algorithms for data setting 1, 8, ECME.BFGS / MCECM.BFGS with Hierarchical should be a good choice.

Data setting 7 (Two clusters with big volume difference)

The Table 3.3.16 shows the Kmeans scale and Hierarchical scale perform the best in data setting 7 with sample size 500 and 1000 respectively, which verify our previous statement that standardization first before applying General VCMM is suggested for the clusters with very different volumes. However, the performances between Kmeans scale and Hierarchical scale in data setting 7 with sample size 500 and 1000 are quite different respectively. We have no suggestion which one to use after standardization. Also, Hierarchical without standardization is the second best initialization strategy for both setting. According to the simulation result here, Hierarchical without standardization may be also a good choice.

Table 3.3.13: The top 10 performance of EM algorithms and the initialization strategies for data setting 4, 5, 6, where the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3).

Setting (Sample size)	General VCMM Ranking	General VCMM Algorithm	General VCMM Initialization strategy	General VCMM Classification rate	General VCMM Normalised BIC	General VCMM Computation time (Seconds)	Gaussian VCMM Ranking	Gaussian VCMM Algorithm	Gaussian VCMM Initialization strategy	Gaussian VCMM Classification rate	Gaussian VCMM Normalised BIC	Gaussian VCMM Computation time (Seconds)
Characteristics : Multivariate skew t distribution (Two clusters without big volume difference)												
4 (500)	1	MCECM.BFGS	Kmeans	0.826	10.79	42	1	ECM.Nelder.Mead	Kmeans	0.833	10.78	47
	2	MCECM.Nelder.Mead	Kmeans	0.826	10.79	44	2	MCECM.Nelder.Mead	Kmeans	0.831	10.78	42
	3	ECM.BFGS	Kmeans	0.826	10.79	45	3	MCECM.Nelder.Mead	Kmeans scale	0.829	10.77	40
	4	ECM.Nelder.Mead	Kmeans	0.826	10.79	50	4	ECM.Nelder.Mead	Kmeans scale	0.829	10.77	51
	5	ECME.BFGS	Kmeans	0.825	10.79	46	5	ECM.BFGS	Kmeans	0.827	10.77	43
	6	ECME.Nelder.Mead	Kmeans	0.824	10.79	55	6	MCECM.BFGS	Kmeans	0.826	10.77	41
	7	MCECM.BFGS	Kmeans scale	0.805	10.78	42	7	ECME.BFGS	Kmeans	0.826	10.77	48
	8	MCECM.Nelder.Mead	Kmeans scale	0.805	10.78	44	8	ECME.Nelder.Mead	Kmeans	0.826	10.77	60
	9	ECM.Nelder.Mead	Kmeans scale	0.805	10.78	50	9	MCECM.BFGS	Kmeans scale	0.825	10.77	41
	10	ECM.BFGS	Kmeans scale	0.804	10.79	42	10	ECM.BFGS	Kmeans scale	0.825	10.77	47
5 (500)	1	ECM.BFGS	Kmeans	0.756	12.08	86	1	ECME.Nelder.Mead	Hierarchical scale	0.523	12.13	62
	2	MCECM.Nelder.Mead	Kmeans	0.756	12.08	99	2	ECME.BFGS	Hierarchical scale	0.523	12.13	72
	3	MCECM.BFGS	Kmeans	0.755	12.08	88	3	MCECM.Nelder.Mead	Hierarchical	0.522	12.10	50
	4	ECM.Nelder.Mead	Kmeans	0.755	12.08	103	4	ECM.BFGS	Hierarchical	0.522	12.10	53
	5	ECME.BFGS	Kmeans	0.753	12.08	100	5	ECM.Nelder.Mead	Hierarchical	0.522	12.10	54
	6	ECME.Nelder.Mead	Kmeans	0.753	12.08	113	6	MCECM.BFGS	Hierarchical	0.522	12.10	54
	7	MCECM.Nelder.Mead	Kmeans scale	0.747	12.08	87	7	ECME.Nelder.Mead	Hierarchical	0.522	12.11	60
	8	ECM.BFGS	Kmeans scale	0.746	12.08	77	8	ECME.BFGS	Hierarchical	0.522	12.11	70
	9	MCECM.BFGS	Kmeans scale	0.746	12.08	79	9	MCECM.Nelder.Mead	Kmeans scale	0.522	12.12	51
	10	ECM.Nelder.Mead	Kmeans scale	0.746	12.08	92	10	MCECM.BFGS	Kmeans scale	0.522	12.12	51
5 (1000)	1	MCECM.Nelder.Mead	Kmeans	0.757	12.02	155	1	MCECM.BFGS	Kmeans	0.521	12.12	79
	2	MCECM.BFGS	Kmeans	0.756	12.01	133	2	MCECM.BFGS	Kmeans scale	0.521	12.12	79
	3	ECM.BFGS	Kmeans	0.756	12.02	138	3	MCECM.Nelder.Mead	Kmeans	0.521	12.12	83
	4	ECM.Nelder.Mead	Kmeans	0.756	12.02	159	4	ECME.Nelder.Mead	Kmeans	0.521	12.12	84
	5	ECME.BFGS	Kmeans	0.754	12.01	143	5	ECME.Nelder.Mead	Kmeans scale	0.521	12.12	84
	6	ECME.Nelder.Mead	Kmeans	0.754	12.01	166	6	MCECM.Nelder.Mead	Kmeans scale	0.521	12.12	84
	7	MCECM.BFGS	Kmeans scale	0.744	12.02	118	7	ECME.BFGS	Kmeans scale	0.521	12.12	89
	8	ECM.BFGS	Kmeans scale	0.744	12.02	123	8	ECME.BFGS	Kmeans	0.521	12.12	95
	9	ECM.Nelder.Mead	Kmeans scale	0.744	12.02	138	9	MCECM.BFGS	Random	0.520	12.12	64
	10	MCECM.Nelder.Mead	Kmeans scale	0.744	12.02	139	10	MCECM.Nelder.Mead	Random	0.520	12.12	69
6 (500)	1	ECM.BFGS	Kmeans	0.987	13.32	40	1	ECM.BFGS	Kmeans	0.970	14.35	21
	2	MCECM.BFGS	Kmeans	0.987	13.32	41	2	ECM.Nelder.Mead	Kmeans	0.970	14.35	22
	3	ECM.Nelder.Mead	Kmeans	0.987	13.32	49	3	ECME.BFGS	Kmeans	0.969	14.34	28
	4	MCECM.Nelder.Mead	Kmeans	0.987	13.32	50	4	MCECM.BFGS	Kmeans	0.969	14.35	21
	5	ECME.BFGS	Kmeans	0.987	13.32	59	5	MCECM.Nelder.Mead	Kmeans	0.969	14.35	23
	6	ECME.Nelder.Mead	Kmeans	0.987	13.32	73	6	ECME.Nelder.Mead	Kmeans	0.969	14.35	31
	7	ECME.Nelder.Mead	Hierarchical scale	0.961	13.38	119	7	MCECM.Nelder.Mead	Kmeans scale	0.927	14.36	28
	8	ECM.BFGS	Hierarchical scale	0.960	13.38	84	8	MCECM.BFGS	Kmeans scale	0.927	14.37	23
	9	MCECM.BFGS	Hierarchical scale	0.960	13.38	87	9	ECM.BFGS	Kmeans scale	0.927	14.37	25
	10	ECME.BFGS	Hierarchical scale	0.960	13.38	99	10	ECM.Nelder.Mead	Kmeans scale	0.927	14.37	28
6 (1000)	1	ECM.BFGS	Kmeans	0.988	13.26	65	1	MCECM.Nelder.Mead	Kmeans	0.971	14.44	30
	2	MCECM.BFGS	Kmeans	0.988	13.26	79	2	ECM.Nelder.Mead	Kmeans	0.971	14.44	32
	3	ECM.Nelder.Mead	Kmeans	0.988	13.26	92	3	ECME.Nelder.Mead	Kmeans	0.971	14.44	40
	4	MCECM.Nelder.Mead	Kmeans	0.988	13.26	94	4	ECME.BFGS	Kmeans	0.970	14.45	43
	5	ECME.BFGS	Kmeans	0.988	13.26	119	5	ECM.BFGS	Kmeans	0.970	14.46	30
	6	ECME.Nelder.Mead	Kmeans	0.988	13.26	137	6	MCECM.BFGS	Kmeans	0.970	14.46	38
	7	ECME.BFGS	Hierarchical scale	0.975	13.30	177	7	ECME.Nelder.Mead	Kmeans scale	0.931	14.46	46
	8	ECM.Nelder.Mead	Kmeans scale	0.974	13.30	112	8	ECME.BFGS	Kmeans scale	0.929	14.44	46
	9	ECME.BFGS	Kmeans scale	0.974	13.30	137	9	MCECM.Nelder.Mead	Kmeans scale	0.929	14.46	33
	10	ECM.BFGS	Hierarchical scale	0.974	13.30	140	10	ECM.Nelder.Mead	Kmeans scale	0.929	14.46	34

Table 3.3.14: The top 10 performance of EM algorithms and the initialization strategies for data setting 3 and 9, where the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3).

Setting (Sample size)	Ranking	General VCMM Algorithm	General VCMM Initialization strategy	General VCMM Classification rate	General VCMM Normalised BIC	General VCMM Computation time (Seconds)	Ranking	Gaussian VCMM Algorithm	Gaussian VCMM Initialization strategy	Gaussian VCMM Classification rate	Gaussian VCMM Normalised BIC	Gaussian VCMM Computation time (Seconds)
Characteristics : Well separated with the "X" shape												
3 (500)	1	MCECM.Nelder.Mead	Random	0.835	12.08	104	1	MCECM.BFGS	Random	0.865	12.01	53
	2	MCECM.BFGS	Random	0.835	12.08	111	2	MCECM.Nelder.Mead	Random	0.865	12.01	55
	3	ECM.Nelder.Mead	Random	0.821	12.10	114	3	ECM.BFGS	Random	0.857	12.01	59
	4	ECM.BFGS	Random	0.821	12.10	121	4	ECM.Nelder.Mead	Random	0.857	12.01	68
	5	ECME.BFGS	Random	0.799	12.12	121	5	MCECM.BFGS	Hierarchical	0.824	12.06	55
	6	ECME.Nelder.Mead	Random	0.792	12.13	118	6	MCECM.Nelder.Mead	Hierarchical	0.824	12.06	57
	7	ECME.BFGS	Hierarchical	0.680	12.29	82	7	ECM.BFGS	Hierarchical	0.824	12.06	58
	8	ECME.Nelder.Mead	Hierarchical	0.678	12.29	89	8	ECM.Nelder.Mead	Hierarchical	0.824	12.06	63
	9	ECM.Nelder.Mead	Hierarchical	0.678	12.30	78	9	ECME.Nelder.Mead	Hierarchical	0.824	12.06	70
	10	MCECM.Nelder.Mead	Hierarchical	0.677	12.30	75	10	ECME.BFGS	Hierarchical	0.824	12.06	72
3 (1000)	1	MCECM.Nelder.Mead	Random	0.851	12.01	171	1	MCECM.BFGS	Random	0.877	11.95	76
	2	MCECM.BFGS	Random	0.851	12.01	175	2	MCECM.Nelder.Mead	Random	0.877	11.95	77
	3	ECM.Nelder.Mead	Random	0.850	12.01	191	3	ECM.Nelder.Mead	Random	0.877	11.95	87
	4	ECM.BFGS	Random	0.850	12.01	194	4	ECM.BFGS	Random	0.876	11.95	87
	5	ECME.Nelder.Mead	Random	0.816	12.04	185	5	MCECM.Nelder.Mead	Hierarchical	0.844	12.00	95
	6	ECME.BFGS	Random	0.816	12.04	192	6	MCECM.BFGS	Hierarchical	0.844	12.00	95
	7	ECME.Nelder.Mead	Hierarchical	0.714	12.17	147	7	ECM.Nelder.Mead	Hierarchical	0.844	12.00	99
	8	ECME.BFGS	Hierarchical	0.714	12.17	148	8	ECM.BFGS	Hierarchical	0.844	12.00	99
	9	MCECM.Nelder.Mead	Hierarchical	0.713	12.18	146	9	ECME.BFGS	Hierarchical	0.844	12.00	100
	10	MCECM.BFGS	Hierarchical	0.713	12.18	148	10	ECME.Nelder.Mead	Hierarchical	0.844	12.00	101
9 (500)	1	ECME.BFGS	Random	0.878	14.58	119	1	MCECM.BFGS	Random	0.726	16.73	76
	2	ECM.Nelder.Mead	Random	0.877	14.57	141	2	MCECM.Nelder.Mead	Random	0.726	16.73	85
	3	MCECM.BFGS	Random	0.867	14.58	98	3	ECM.BFGS	Random	0.722	16.68	82
	4	ECM.BFGS	Random	0.860	14.60	98	4	ECM.Nelder.Mead	Random	0.722	16.68	88
	5	MCECM.Nelder.Mead	Random	0.847	14.65	117	5	ECME.BFGS	Random	0.690	16.61	48
	6	ECM.Nelder.Mead	Random	0.847	14.65	123	6	ECME.Nelder.Mead	Random	0.664	16.56	52
	7	MCECM.Nelder.Mead	Kmeans scale	0.757	15.46	169	7	ECM.BFGS	Hierarchical	0.645	16.83	68
	8	ECME.BFGS	Kmeans scale	0.757	15.49	137	8	MCECM.Nelder.Mead	Hierarchical	0.645	16.83	71
	9	ECM.BFGS	Kmeans scale	0.756	15.46	118	9	ECM.Nelder.Mead	Hierarchical	0.645	16.83	72
	10	ECM.Nelder.Mead	Kmeans scale	0.756	15.46	167	10	ECME.BFGS	Hierarchical	0.639	16.80	53

Table 3.3.15: The top 12 performance of EM algorithms and the initialization strategies for data setting 1 and 8, where the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3).

Setting (Sample size)	Ranking	General VCMM Algorithm	General VCMM Initialization strategy	General VCMM Classification rate	General VCMM Normalised BIC	General VCMM Computation time (Seconds)	Ranking	Gaussian VCMM Algorithm	Gaussian VCMM Initialization strategy	Gaussian VCMM Classification rate	Gaussian VCMM Normalised BIC	Gaussian VCMM Computation time (Seconds)
Characteristics : Well separated without overlap (Except Gaussian and skew t distribution)												
1 (500)	1	ECME.BFGS	Hierarchical	0.944	16.54	117	1	ECME.Nelder.Mead	Hierarchical	0.730	19.70	67
	2	MCECM.BFGS	Hierarchical	0.943	16.57	83	2	ECME.BFGS	Hierarchical	0.725	19.70	54
	3	ECM.BFGS	Hierarchical	0.942	16.57	84	3	ECM.BFGS	Hierarchical	0.724	19.74	67
	4	ECME.Nelder.Mead	Hierarchical	0.940	16.57	154	4	ECM.Nelder.Mead	Hierarchical	0.724	19.74	77
	5	MCECM.Nelder.Mead	Hierarchical	0.937	16.59	95	5	MCECM.BFGS	Hierarchical	0.723	19.74	65
	6	ECM.Nelder.Mead	Hierarchical	0.934	16.59	92	6	MCECM.Nelder.Mead	Hierarchical	0.723	19.74	77
	7	ECM.BFGS	Random	0.885	16.70	186	7	ECME.BFGS	Random	0.620	19.97	73
	8	MCECM.BFGS	Random	0.880	16.74	167	8	ECME.Nelder.Mead	Hierarchical scale	0.617	19.79	66
	9	ECME.Nelder.Mead	Random	0.879	16.74	269	9	ECME.BFGS	Hierarchical scale	0.615	19.81	55
	10	ECME.BFGS	Random	0.875	16.74	181	10	MCECM.BFGS	Hierarchical scale	0.613	19.82	73
	11	ECM.Nelder.Mead	Random	0.872	16.76	217	11	MCECM.Nelder.Mead	Hierarchical scale	0.613	19.82	83
	12	MCECM.Nelder.Mead	Random	0.868	16.79	210	12	ECM.BFGS	Hierarchical scale	0.612	19.82	72
1 (1000)	1	MCECM.BFGS	Random	0.935	16.48	285	1	ECM.BFGS	Hierarchical	0.736	19.68	112
	2	ECME.BFGS	Random	0.932	16.49	307	2	MCECM.BFGS	Hierarchical	0.736	19.68	115
	3	ECM.Nelder.Mead	Random	0.931	16.50	324	3	ECME.BFGS	Hierarchical	0.735	19.66	88
	4	ECME.Nelder.Mead	Random	0.930	16.49	370	4	ECME.Nelder.Mead	Hierarchical	0.735	19.66	96
	5	ECM.BFGS	Random	0.930	16.50	285	5	MCECM.Nelder.Mead	Hierarchical	0.735	19.66	126
	6	MCECM.BFGS	Hierarchical	0.927	16.48	112	6	ECM.Nelder.Mead	Hierarchical	0.735	19.66	131
	7	ECM.BFGS	Hierarchical	0.927	16.48	113	7	ECME.Nelder.Mead	Kmeans	0.605	19.88	124
	8	ECME.BFGS	Hierarchical	0.927	16.48	163	8	ECME.BFGS	Random	0.602	19.92	107
	9	MCECM.Nelder.Mead	Hierarchical	0.924	16.49	128	9	ECM.BFGS	Kmeans	0.601	19.88	164
	10	ECM.Nelder.Mead	Hierarchical	0.924	16.50	130	10	MCECM.Nelder.Mead	Kmeans	0.601	19.88	177
	11	ECME.Nelder.Mead	Hierarchical	0.923	16.51	183	11	ECM.Nelder.Mead	Kmeans	0.601	19.88	186
	12	MCECM.Nelder.Mead	Random	0.923	16.53	296	12	MCECM.BFGS	Kmeans	0.601	19.90	159
8 (500)	1	ECM.Nelder.Mead	Hierarchical	0.991	17.26	59	1	ECM.BFGS	Random	0.907	19.92	53
	2	ECM.BFGS	Hierarchical	0.990	17.26	47	2	ECM.Nelder.Mead	Random	0.907	19.92	64
	3	MCECM.BFGS	Hierarchical	0.990	17.26	48	3	MCECM.BFGS	Random	0.906	19.92	55
	4	MCECM.Nelder.Mead	Hierarchical	0.990	17.26	55	4	MCECM.Nelder.Mead	Random	0.906	19.92	62
	5	ECME.BFGS	Hierarchical	0.990	17.26	73	5	ECM.BFGS	Kmeans	0.905	19.92	61
	6	ECME.Nelder.Mead	Hierarchical	0.990	17.26	87	6	MCECM.BFGS	Kmeans	0.905	19.92	62
	7	ECME.BFGS	Random	0.912	17.56	105	7	MCECM.Nelder.Mead	Kmeans	0.905	19.92	72
	8	ECME.Nelder.Mead	Random	0.900	17.60	117	8	ECM.Nelder.Mead	Kmeans	0.905	19.92	74
	9	ECM.Nelder.Mead	Random	0.862	17.67	104	9	MCECM.BFGS	Hierarchical	0.900	19.90	33
	10	ECM.BFGS	Random	0.861	17.67	86	10	MCECM.Nelder.Mead	Hierarchical	0.900	19.90	37
	11	MCECM.Nelder.Mead	Random	0.843	17.73	103	11	ECM.BFGS	Hierarchical	0.899	19.90	33
	12	MCECM.BFGS	Random	0.843	17.73	441	12	ECM.Nelder.Mead	Hierarchical	0.899	19.90	39

Table 3.3.16: The top 20 performance of EM algorithms and the initialization strategies for data setting 7, where the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3).

	General VCMM						Gaussian VCMM					
Setting (Sample size)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
Characteristics : Two clusters with big volume difference												
7 (500)	1	ECM.BFGS	Kmeans scale	0.980	13.92	69	1	ECM.BFGS	Hierarchical scale	0.954	14.70	14
	2	MCECM.BFGS	Kmeans scale	0.980	13.92	71	2	MCECM.BFGS	Hierarchical scale	0.954	14.70	14
	3	MCECM.Nelder.Mead	Kmeans scale	0.980	13.92	82	3	MCECM.Nelder.Mead	Hierarchical scale	0.954	14.70	16
	4	ECM.Nelder.Mead	Kmeans scale	0.980	13.92	85	4	ECM.BFGS	Hierarchical	0.954	14.70	17
	5	ECME.BFGS	Kmeans scale	0.980	13.92	94	5	MCECM.BFGS	Hierarchical	0.954	14.70	17
	6	ECME.Nelder.Mead	Kmeans scale	0.980	13.92	119	6	ECM.Nelder.Mead	Hierarchical scale	0.954	14.70	17
	7	MCECM.BFGS	Hierarchical	0.975	13.92	38	7	ECM.Nelder.Mead	Hierarchical	0.954	14.70	18
	8	ECME.BFGS	Hierarchical	0.975	13.92	57	8	MCECM.Nelder.Mead	Hierarchical	0.954	14.70	18
	9	ECME.Nelder.Mead	Hierarchical	0.975	13.92	72	9	ECME.Nelder.Mead	Hierarchical scale	0.954	14.70	22
	10	ECM.BFGS	Hierarchical	0.974	13.92	38	10	ECME.BFGS	Hierarchical scale	0.954	14.70	22
	11	ECM.Nelder.Mead	Hierarchical	0.974	13.92	47	11	ECME.Nelder.Mead	Hierarchical	0.954	14.70	25
	12	MCECM.Nelder.Mead	Hierarchical	0.974	13.92	47	12	ECME.BFGS	Hierarchical	0.954	14.70	25
	13	MCECM.BFGS	Hierarchical scale	0.972	13.92	36	13	ECM.BFGS	Kmeans scale	0.953	14.78	25
	14	ECM.BFGS	Hierarchical scale	0.972	13.93	36	14	MCECM.BFGS	Kmeans scale	0.953	14.78	25
	15	MCECM.Nelder.Mead	Hierarchical scale	0.972	13.93	45	15	MCECM.Nelder.Mead	Kmeans scale	0.953	14.78	29
	16	ECME.BFGS	Hierarchical scale	0.972	13.93	52	16	ECM.Nelder.Mead	Kmeans scale	0.953	14.78	37
	17	ECME.Nelder.Mead	Hierarchical scale	0.972	13.93	105	17	ECME.Nelder.Mead	Kmeans scale	0.951	14.74	33
	18	ECM.Nelder.Mead	Hierarchical scale	0.971	13.93	45	18	ECME.BFGS	Kmeans scale	0.951	14.74	36
	19	MCECM.BFGS	Random	0.958	14.04	224	19	ECM.BFGS	Kmeans	0.947	14.80	52
	20	ECM.BFGS	Random	0.949	14.07	214	20	ECM.Nelder.Mead	Kmeans	0.947	14.80	56
7 (1000)	1	ECM.BFGS	Hierarchical scale	0.980	13.79	61	1	ECME.Nelder.Mead	Kmeans	0.958	14.72	66
	2	MCECM.BFGS	Hierarchical scale	0.980	13.79	71	2	ECM.BFGS	Kmeans	0.958	14.72	92
	3	ECME.Nelder.Mead	Hierarchical scale	0.980	13.79	106	3	ECME.BFGS	Kmeans	0.958	14.72	645
	4	MCECM.Nelder.Mead	Hierarchical scale	0.980	13.79	278	4	MCECM.Nelder.Mead	Kmeans	0.958	14.72	970
	5	ECME.BFGS	Hierarchical scale	0.980	13.79	329	5	ECM.Nelder.Mead	Kmeans	0.958	14.72	1004
	6	ECM.Nelder.Mead	Hierarchical	0.980	13.79	344	6	ECM.BFGS	Hierarchical scale	0.957	14.70	24
	7	ECM.Nelder.Mead	Hierarchical scale	0.980	13.79	349	7	ECM.Nelder.Mead	Hierarchical scale	0.957	14.70	25
	8	ECME.Nelder.Mead	Hierarchical	0.980	13.79	565	8	MCECM.BFGS	Kmeans	0.957	14.72	147
	9	MCECM.Nelder.Mead	Hierarchical	0.979	13.79	362	9	MCECM.Nelder.Mead	Hierarchical scale	0.956	14.70	24
	10	ECM.BFGS	Hierarchical	0.978	13.80	72	10	ECME.BFGS	Hierarchical scale	0.956	14.70	27
	11	MCECM.BFGS	Hierarchical	0.978	13.80	73	11	ECME.Nelder.Mead	Hierarchical scale	0.956	14.70	28
	12	ECME.BFGS	Hierarchical	0.978	13.80	448	12	MCECM.BFGS	Hierarchical scale	0.956	14.70	88
	13	MCECM.BFGS	Random	0.976	13.82	343	13	ECM.Nelder.Mead	Kmeans scale	0.952	14.86	44
	14	ECM.BFGS	Kmeans scale	0.974	13.82	133	14	ECME.BFGS	Hierarchical	0.951	14.78	372
	15	MCECM.BFGS	Kmeans scale	0.974	13.82	139	15	ECM.BFGS	Kmeans scale	0.951	14.82	42
	16	ECME.Nelder.Mead	Kmeans scale	0.974	13.82	176	16	MCECM.Nelder.Mead	Kmeans scale	0.951	14.85	44
	17	MCECM.Nelder.Mead	Kmeans scale	0.974	13.82	540	17	ECME.BFGS	Kmeans scale	0.950	14.76	45
	18	ECM.Nelder.Mead	Kmeans scale	0.974	13.82	545	18	ECME.Nelder.Mead	Kmeans scale	0.950	14.76	47
	19	ECME.BFGS	Kmeans scale	0.974	13.82	639	19	ECM.BFGS	Hierarchical	0.950	14.78	31
	20	ECM.BFGS	Random	0.974	13.83	315	20	ECME.Nelder.Mead	Hierarchical	0.950	14.78	35

5. The summary of the best initialization strategy and EM algorithm for clustering data setting 1 - 9.

According to the results we have from the VCMM simulation, we know that some initialization strategies perform better for the data with some particular characteristics. The best initialization strategy and initialization strategy is summarised as a flow chart in the Figure 3.3.19. The red rectangular box with curved edge is the start of data clustering. Then, you will follow the arrow with a suitable condition to go to other boxes. The orange rectangular boxes refer to the characteristics of the data and the green diamond boxes refer to best initialization strategies and the EM algorithm with the best performance which are the end of the flow chart.

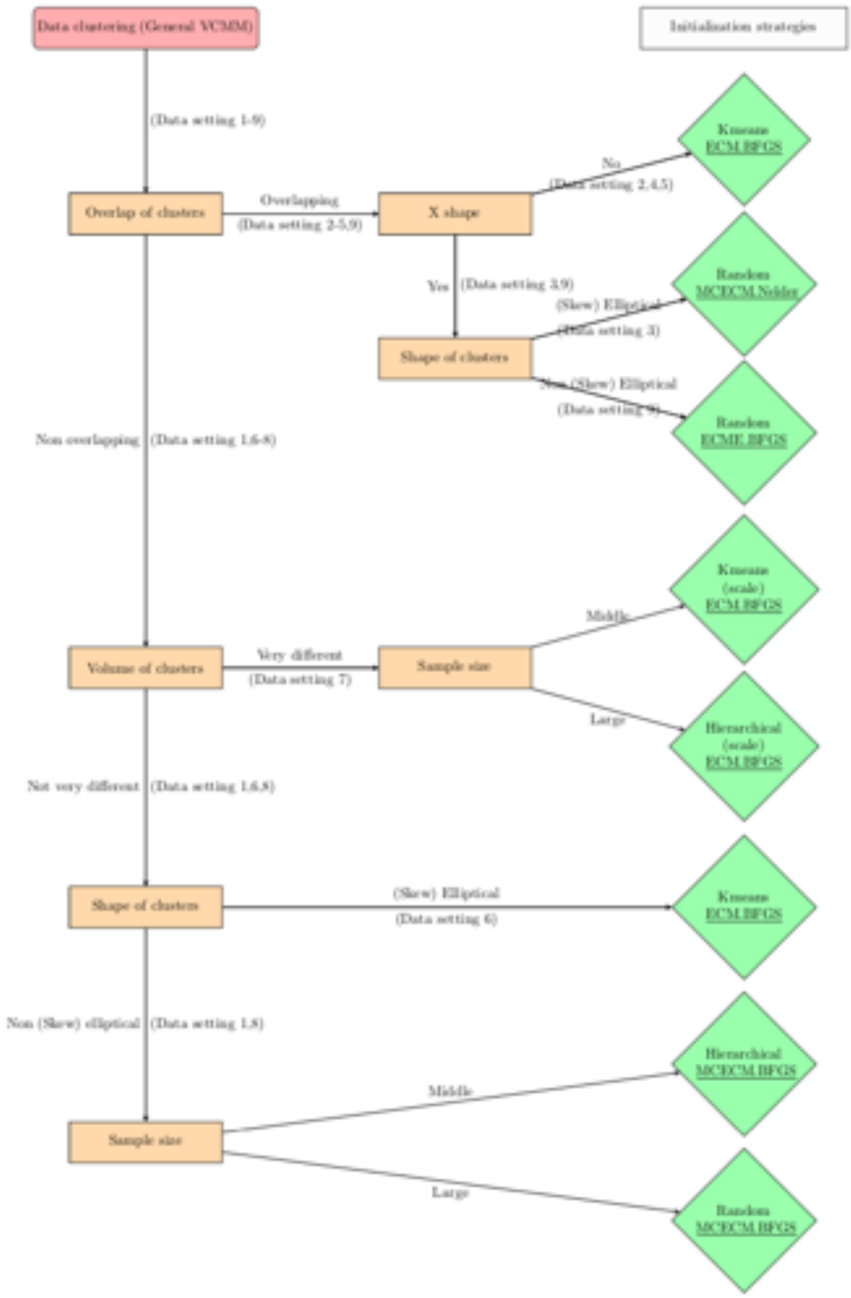


Figure 3.3.19: The summary of the best initialization strategy and EM algorithm for clustering data setting 1 - 9.

6. Selection of the EM algorithm and initialization strategy for clustering data without characteristics information

We have mentioned that some initialization strategies perform better for the data with some particular characteristics. However, we cannot know the shape, overlap and volume of each cluster clearly if the clusters are not marked. Therefore, in general, we cannot choose the best initialization strategy for clustering real data set with unknown data characteristics.

We can calculate the BIC value for each initialization strategy and choose the one with lowest BIC and that one probably gives us the best classification rate, but it can probably leads to a very high computation time if the the data is high-dimensional. In order to save some computation time, we would suggest two better initialization strategies according to our simulation result and choose the one with lower BIC. Also, we know that random outperforms other initialization strategies for those clusters like a "X" shape and "X" shape still can be observed easily for three dimensional data.

Based on these two selection's principle, we have come up with two flow charts in the Figure 3.3.20 and 3.3.21 to select the suitable initialization strategy for general VCMM and Gaussian VCMM algorithm respectively. The flow chart is basically similar to the one above. The only difference is that the green rectangular boxes refer to the final decision of the initialization strategy which are the end of the flow chart.

In the Figure 3.3.20 and 3.3.21, we know that if the data has the "X" shape, we recommend random with ECM.Nelder.Mead and MCECM.BFGS for general VCMM and Gaussian VCMM respectively. The result is from the Table 3.3.14 and the EM algorithms topping or almost topping in the list are selected. If the data doesn't look like "X" shape, for general VCMM, we have suggested Hierarchical or Kmeans (scale) with ECM.BFGS / MCECM.BFGS. The selection of the EM algorithms is because of shorter required computation time that we mentioned before. In the Table 3.3.11, except data setting 3,9 with "X" shape, we know that Hierarchical performs quite well in data setting 1,2,4,6-8 and Kmeans (scale) perform quite well in data setting 4-7. More importantly, we know that the clusters in data setting 7 have larger volume or scale difference and real data set usually have this characteristics. So, Hierarchical and Kmeans (scale) is selected. For Gaussian VCMM, EM algorithms with BFGS are selected as we mentioned that it take less time than Nelder.Mead. Also, except data setting 3,9 with "X" shape, Kmeans performs well in data setting 1,2,4-8 and Hierarchical performs quite well in data setting 1,4,5,7 and they don't have worst performance. So, Kmeans and Hierarchical are selected.

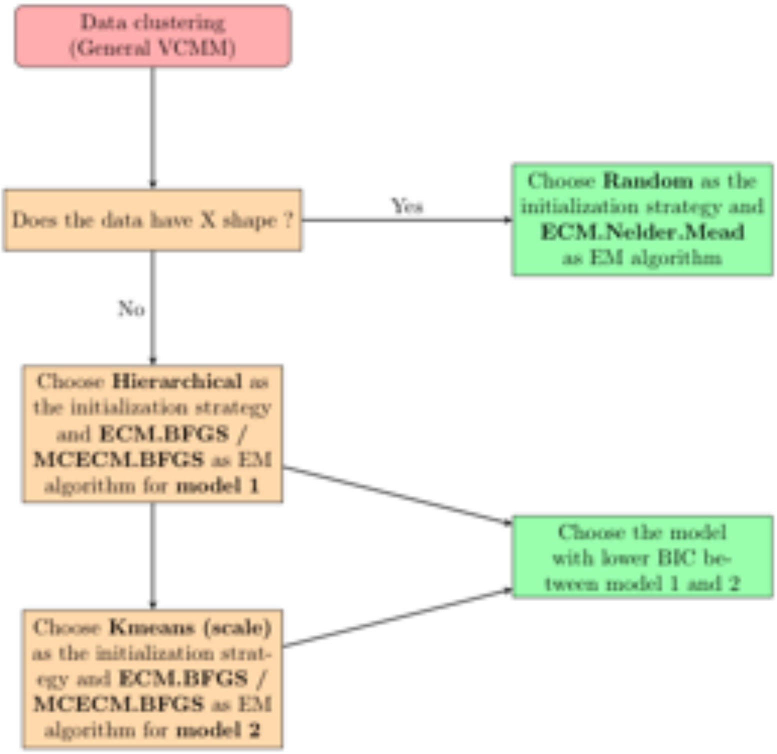


Figure 3.3.20: The selection of the initialization strategy for general VCMM algorithm.

Figure 3.3.21: The selection of the initialization strategy for Gaussian VCMM algorithm.

Table 4.0.1: The overview of the real data sets we used for assessment.

Notation	Name	AIS	BCW
n	Sample size	202	569
d	Dimension of each sample (Variables used)	5 (LBM, Wt, BMI, EBC and PBF)	4 (PSE, ES, EC and ECP)
K	Total number of clusters	2	2
π_1	Mixing proportion of cluster 1 (Name of cluster 1)	0.495 (Female)	0.594 (Benign)
π_2	Mixing proportion of cluster 2 (Name of cluster 2)	0.505 (Male)	0.406 (Malignant)

4 Real data sets

In Section 3 simulation study, we have generated different data sets to assess the performance of Gaussian mixture model (GMM) and vine copula mixture model (VCMM) algorithm with different EM algorithms and initialization strategies. However, the data for simulation is just 3 dimensions and generated by GMM, VCMM and mixture of multivariate skew t. The real data is probably more complicated, for example, higher dimension, more clusters, following special distributions, more outliers and so on. Therefore, in this section, we just follow the real data sets and the selected variables used in Sahin and Czado [2021] to test the performance of VCMM algorithm : 1. Australian Institute of Sport data set, 2. Breast Cancer Wisconsin Diagnostic data set. The Table 4.0.1 shows you the general overview of two real data sets and more details will be given in the Section 4.1.

4.1 Introduction to the real data set

Australian Institute of Sport (AIS)

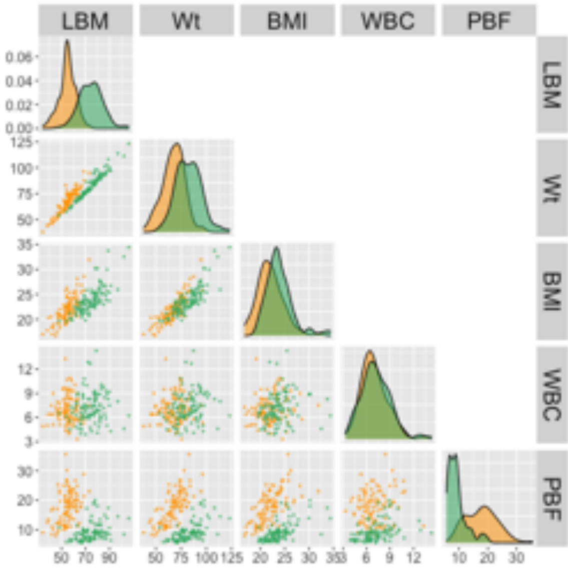
Australian Institute of Sport (AIS) data set contains 13 information of 202 athletes, including the sex of the athletes, the sport of the athletes and 11 different body and blood measurements. There are in total 102 male and 100 female athletes in the data set. We would like to use VCMM algorithm to identify the gender of the athletes, based on 5 selected information of the athletes, which are lean body mass (LBM), weight (Wt), body mass index (BMI), white blood cell count (WBC), and percentage of body fat (PBF). The data is available in the R package `DAAG` and the data is called `ais`.

The pairwise scatter plots of the subset of AIS data are shown in the Figure 4.1.1. The orange and green dots represent the observation for female and male athletes respectively. From the panels of the Figure 4.1.1 (a), you can see that the two clusters are separated the most between PBF and LBM, which means there are a significant different between male and female for those measurements. In the diagonal of the Figure 4.1.1 (a), the marginal distributions for each cluster are shown. For all variables, the values near the center (median) appear more frequently than values at the tails. Also, the marginal distributions between male and female are quite similar for WBC. For PBF with female, it shows a positive skew that tail is on the right. In the Figure 4.1.1 (b), (c) and (d), we can see the dependency between each variable. From the normalized contour plots, we can know that the dependency between the same pair of variables is usually quite similar in females and males and the associated Kendall's tau too, except there are more differences between WBC and PBF. Also, we can see the non-Gaussian dependence patterns in many pair of variables, and asymmetric tail dependence between LBM and Wt.

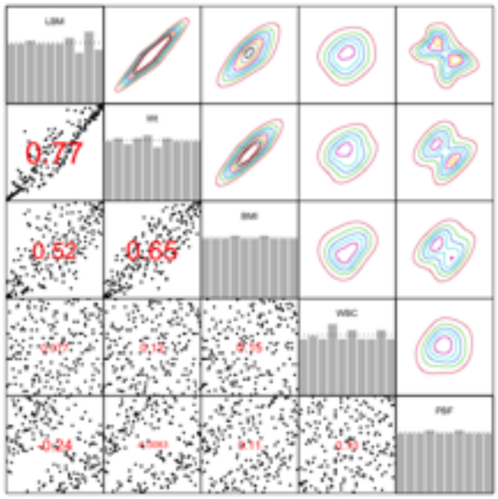
Breast Cancer Wisconsin Diagnostic (BCW)

Breast Cancer Wisconsin Diagnostic (BCW) is a data set from UCI Machine Learning Repository containing 30 features of cell nuclei from breast mass and the classification of breast cancer: malignant (cancer) or benign (not cancer). Ten features, such as size, shape and regularity, are computed for each cell nucleus from a breast mass. The mean, standard error, and extreme value (mean of the three largest values) of each of 10 nuclear parameters is reported for a total of 30 features. There are in total 569 patients, including 212 malignant and 357 benign. With the selection of 4 features, perimeter standard error (PSE), extreme smoothness (ES), extreme concavity (EC), and extreme concave points (ECP), we would like to use VCMM algorithm to classify the status of the breast mass. The data is available in the R package `dslabs` and the data is called `brca`.

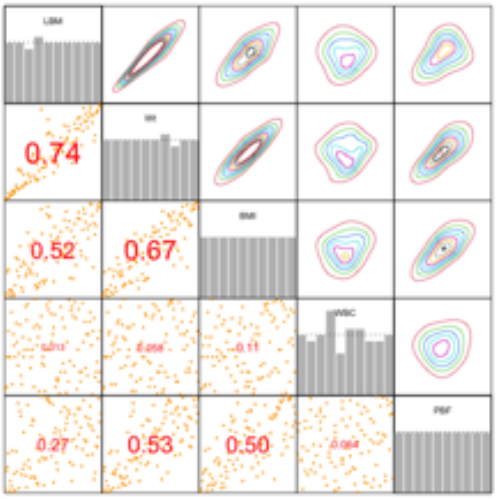
The pairwise scatter plots of the subset of the BCW data are shown in the Figure 4.1.2. The orange and green dots represent the observation for benign and malignant respectively. Same as the AIS data, for all variables, the values near the center (median) appear more frequently than values at the tails. Apart from ES, the distributions are skewed at a particular level. In particular, EC with malignant and PSE show a positive skew that the tail is on the right. About the dependency, it is quite similar between the same pair of variables in benign and malignant, but some of them the strength are quite different, for example, ES and EC. Also, some of the pairs show the asymmetric tail dependency, for example, the pair of EC and ECP shows the strong asymmetric tail dependency.



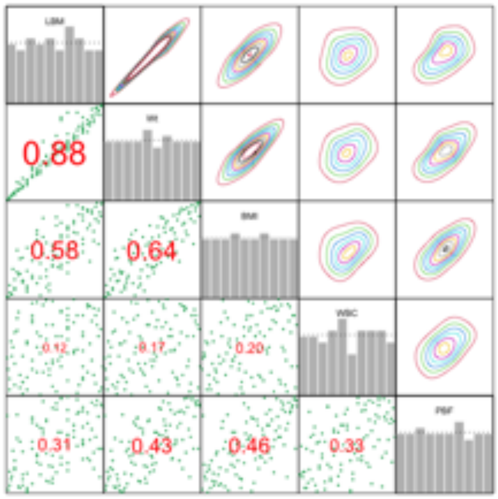
(a) **True cluster**



(b) **Pairs plot (Copula data)**

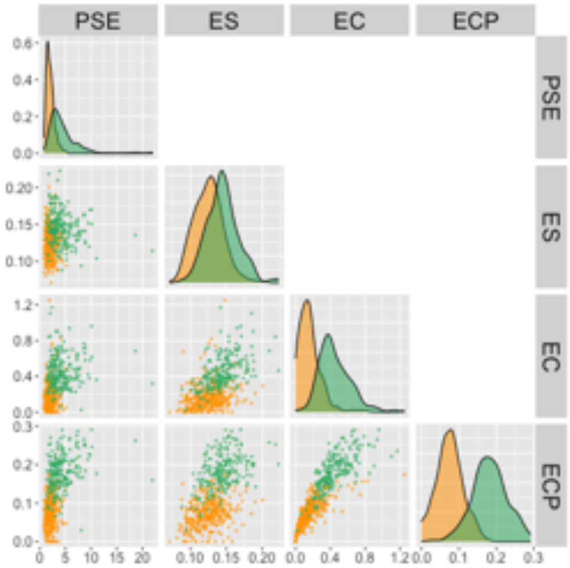


(c) **Pairs plot for females (Copula data)**

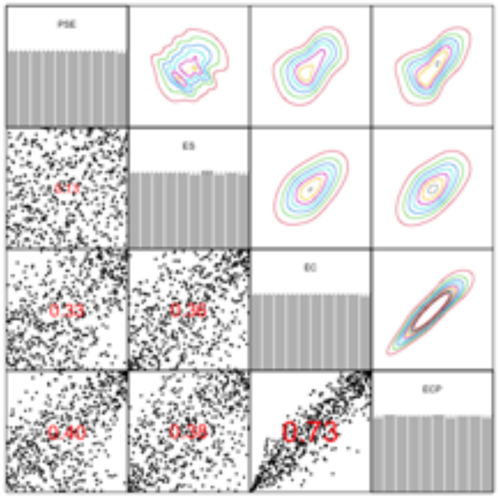


(d) **Pairs plot for males (Copula data)**

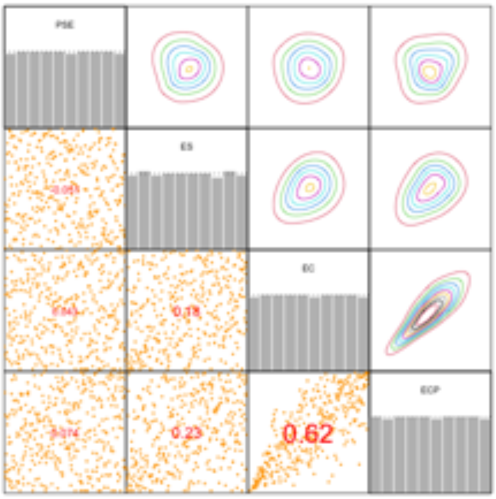
Figure 4.1.1: The pairwise scatter plot of the subset of AIS data (top left) with orange dots for observations of females and green dots for males. Pairs plots of all data (top right), females (bottom left) and males (bottom right), where lower: pairs plots of copula data with the values of Kendall's tau, diagonal: histogram of copula margins, upper: normalized contour plots.



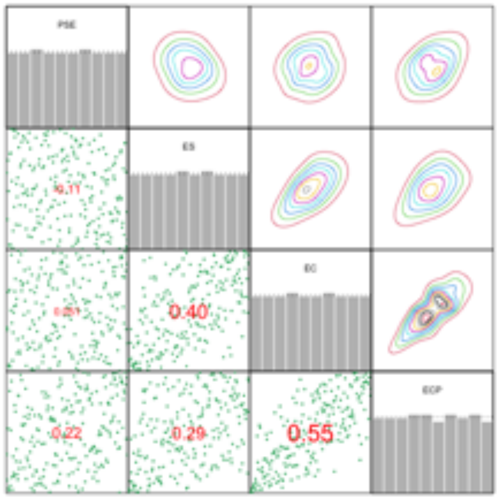
(a) **True cluster**



(b) **Pairs plot (Copula data)**



(c) **Pairs plot for benign (Copula data)**



(d) **Pairs plot for malignant (Copula data)**

Figure 4.1.2: The pairwise scatter plot of the subset of BCW data (top left) with orange dots for observations of benign and green dots for malignant. Pairs plots of all data (top right), benign (bottom left) and malignant (bottom right), where lower: pairs plots of copula data with the values of Kendall's tau, diagonal: histogram of copula margins, upper: normalized contour plots.

4.2 Introduction to the modelling methods for clustering

1. Modelling methods used for clustering

In this section, four modelling methods will be used for clustering and the overview is shown in the Table 4.2.1. GMM is the Algorithm (1) used in the Section 3.1 and 3.2. General VCMM and Gaussian VCMM are the algorithms used in the Section 3.3. The new modelling method called General VCMM (Copula with single parameter) is used here and the only difference is that just the copula families with one parameter are used for modelling, including Gaussian, Student t, Frank, Clayton, Gumbel and Joe copula and their rotated copulas. And all other factors keep same as General VCMM, for example, marginal distributions used for modelling, CM steps order for EM algorithms.

Table 4.2.1: The overview of the modelling method, the associated marginal distribution and copula families used for clustering. *For marginal distribution, "All" refers to 1. Gaussian distribution, 2. Log-normal distribution, 3. Exponential distribution, 4. Log-logistic distribution, 5. Logistic distribution, 6. Gamma distribution and 7. Student's t distribution with degrees of freedom 3.3. For copula family, "All" refers to Gaussian copula, Student's t copula, Gumbel copula, Clayton copula, Frank copula, Joe copula, BB1 copula, BB6 copula, BB8 copula and their rotated copulas with 90°, 180°, 270° degrees.

Modelling method	Marginal distribution	Copula family	Pair copula construction
GMM	Gaussian	-	-
General VCMM	All*	All*	Determined by the VCMM algorithm
General VCMM (Copula with single parameter)	All*	Copula with single parameter and Student's t copula	Determined by the VCMM algorithm
Gaussian VCMM	Gaussian	Gaussian	Determined by the VCMM algorithm

2. The combination of the modelling methods, EM algorithms and initialisation strategies used for clustering

For each data set, we have in total 110 ways for clustering, including 4 VCMM algorithms, 1 GMM algorithm, 5 initialisation stragties and 6 EM algorithms for VCMM and 4 EM algorithms for GMM. The details of each item are shown in the Table 4.2.2.

Table 4.2.2: The combination of the modelling method, EM algorithm and initialisation strategy used for each real data set clustering.

Modelling method	EM algorithm	Initialisation strategy	Number of combinations
1. General VCMM 2. General VCMM (Copula with single parameter) 3. Gaussian VCMM	1. ECM.Nelder.Mead 2. ECM.BFGS 3. ECME.Nelder.Mead 4. ECME.BFGS 5. MCECM.Nelder.Mead 6. MCECM.BFGS	1. Random 2. Kmeans 3. Kmeans scale 4. Hierarchical 5. Hierarchical scale	90
1. GMM	1. ECM 2. ECME.Nelder.Mead 3. ECME.ECME 4. MCECM	1. Random 2. Kmeans 3. Kmeans scale 4. Hierarchical 5. Hierarchical scale	20
		Total :	110

4.3 Result and performance of clustering

1. Comparison of the result with different methods for clustering

In the Table 4.3.1, it shows the performance of clustering AIS and BCW data, that the mean for 50 replications of the performance measure with different modelling methods, EM algorithms and initialisation strategies. Also, the pair plots and associated classification rates after initialising and final clustering are shown in Appendix C.3 and C.4. From the clustering results for the **AIS data set**, we observe the following:

- For a particular initialization strategy, the normalised BIC and classification rate perform usually similar within different EM algorithms, but there is a great difference in computation time.
- Mostly, the EM algorithms with Neldar-Mead is taking more time than with BFGS.
- For General VCMM and General VCMM (Copula with single parameter), the performance varies quite much with different initialization strategy.
- For each of the initialization strategy, General VCMM (Copula with single parameter) performs mostly better than General VCMM.
- Gaussian VCMM performs better than GMM.
- For each modelling method other than GMM, Hierarchical (scale) usually leads to the best classification rate.
- Apart from Gaussian VCMM, the classification rate after final clustering is always higher or very similar to the classification rate after initialising clustering for a particular initialization strategy.

Also, from the clustering results of **BCW data set**, we can observe the following:

- For a particular initialization strategy, the normalised BIC and classification rate perform usually similar within different EM algorithms, but there is a great difference in computation time.
- Mostly, the EM algorithms with Neldar-Mead is taking more time than with BFGS.
- Random requires longer computation time.
- For General VCMM and General VCMM (Copula with single parameter), the performance varies quite much with different initialization strategy.
- In terms of normalised BIC, General VCMM and General VCMM (Copula with single parameter) performs always better than Gaussian VCMM.
- For General VCMM and General VCMM (Copula with single parameter), Hierarchical (scale) always leads to the best classification rate.
- Apart from Gaussian VCMM, the classification rate after final clustering is always higher or very similar to the classification rate after initialising clustering for a particular initialization strategy.

Combining the observations for AIS and BCW data set above, we can conclude the following:

- Although different EM algorithms mostly don't affect the performance for clustering significantly, there is a significant effect on the running time needed.
- The EM algorithms with Neldar-Mead is taking more time than with BFGS.
- The starting value / initialization strategy for the EM algorithm is the key factor of the performance of the clustering.
- The general VCMM improves the clustering performance after initialization strategy.

- The more complex of the model, it doesn't have to lead to a better result / better fit for clustering.
- For clustering of non-Gaussian data, more flexible method like General VCMM with a suitable initialization strategy is more recommended than Gaussian mixture model.
- For real data set, Hierarchical (scale) is suggested as an initialization strategy.

Table 4.3.1: The mean for 50 replications of the performance measure for each EM algorithm and its extension with different initialization strategies for data set of AIS and BCW, where the green and red colour represent the best and the worst initialization strategies for each EM algorithm respectively and "-" represents not available as the conditions (2.5.5) or (2.5.6) is fulfilled for GMM algorithm.

AIS

		General VCMM						Gaussian VCMM				
Performance measures	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)
Computation time	ECM.Nelder.Mead	402	83	219	393	640	ECM.Nelder.Mead	352	550	227	557	256
	ECM.BFGS	386	53	149	219	303	ECM.BFGS	268	392	155	362	164
	ECME.Nelder.Mead	478	218	464	683	914	ECME.Nelder.Mead	787	882	378	950	452
	ECME.BFGS	502	120	203	359	527	ECME.BFGS	524	632	265	660	356
	MCECM.Nelder.Mead	345	81	232	413	488	MCECM.Nelder.Mead	371	549	210	582	256
	MCECM.BFGS	413	42	149	236	277	MCECM.BFGS	247	348	159	367	151
Normalised BIC	ECM.Nelder.Mead	25.60	25.87	24.21	24.70	23.92	ECM.Nelder.Mead	23.06	23.57	23.57	23.57	23.57
	ECM.BFGS	25.81	25.87	24.21	24.70	23.92	ECM.BFGS	23.06	23.57	23.57	23.57	23.57
	ECME.Nelder.Mead	25.76	26.00	24.21	24.01	23.92	ECME.Nelder.Mead	23.06	23.57	23.57	23.57	23.57
	ECME.BFGS	25.86	26.00	24.21	23.52	23.92	ECME.BFGS	23.06	23.57	23.57	23.57	23.57
	MCECM.Nelder.Mead	25.82	25.87	24.21	24.70	23.92	MCECM.Nelder.Mead	23.06	23.57	23.57	23.57	23.57
	MCECM.BFGS	25.81	25.87	24.21	24.70	23.92	MCECM.BFGS	23.06	23.57	23.57	23.57	23.57
Classification rate	ECM.Nelder.Mead	0.580	0.753	0.914	0.790	0.924	ECM.Nelder.Mead	0.658	0.914	0.905	0.905	0.914
	ECM.BFGS	0.576	0.753	0.914	0.790	0.924	ECM.BFGS	0.658	0.914	0.905	0.905	0.914
	ECME.Nelder.Mead	0.572	0.775	0.914	0.905	0.924	ECME.Nelder.Mead	0.658	0.914	0.905	0.905	0.914
	ECME.BFGS	0.602	0.775	0.914	0.914	0.924	ECME.BFGS	0.658	0.914	0.905	0.905	0.914
	MCECM.Nelder.Mead	0.602	0.753	0.914	0.790	0.924	MCECM.Nelder.Mead	0.658	0.914	0.905	0.905	0.914
	MCECM.BFGS	0.576	0.753	0.914	0.790	0.924	MCECM.BFGS	0.658	0.914	0.905	0.905	0.914

		General VCMM (Copula with single parameter)						GMM				
Performance measures	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)
Computation time	ECM.Nelder.Mead	372	104	222	383	538	ECM	0.331	0.034	0.100	0.076	0.263
	ECM.BFGS	599	56	169	213	295	MCECM	0.134	0.047	0.464	0.091	0.268
	ECME.Nelder.Mead	383	254	452	648	1003	ECME.Nelder.Mead	-	103	263	109	77
	ECME.BFGS	699	106	301	321	496	ECME.BFGS	-	-	-	-	-
	MCECM.Nelder.Mead	346	78	244	395	593						
	MCECM.BFGS	486	51	154	229	272						
Normalised BIC	ECM.Nelder.Mead	25.75	25.85	23.83	25.03	23.85	ECM	23.56	24.28	23.56	24.28	23.81
	ECM.BFGS	26.44	25.85	23.86	25.03	23.85	MCECM	23.56	24.28	23.56	24.28	23.81
	ECME.Nelder.Mead	25.70	25.79	23.86	24.13	23.85	ECME.Nelder.Mead	-	24.33	24.14	24.34	24.24
	ECME.BFGS	26.46	25.79	23.86	23.65	23.85	ECME.BFGS	-	-	-	-	-
	MCECM.Nelder.Mead	25.75	25.85	23.86	25.03	23.85						
	MCECM.BFGS	26.46	25.85	23.83	25.03	23.85						
Classification rate	ECM.Nelder.Mead	0.569	0.753	0.924	0.821	0.924	ECM	0.652	0.767	0.652	0.760	0.712
	ECM.BFGS	0.616	0.753	0.924	0.821	0.924	MCECM	0.652	0.767	0.652	0.767	0.712
	ECME.Nelder.Mead	0.589	0.767	0.924	0.897	0.924	ECME.Nelder.Mead	-	0.753	0.870	0.699	0.862
	ECME.BFGS	0.606	0.767	0.924	0.914	0.924	ECME.BFGS	-	-	-	-	-
	MCECM.Nelder.Mead	0.569	0.753	0.924	0.821	0.924						
	MCECM.BFGS	0.606	0.753	0.924	0.821	0.924						

BCW

		General VCMM						Gaussian VCMM				
Performance measures	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)
Computation time	ECM.Nelder.Mead	255	111	384	55	112	ECM.Nelder.Mead	327	124	267	102	101
	ECM.BFGS	420	66	232	61	74	ECM.BFGS	286	109	237	85	86
	ECME.Nelder.Mead	532	304	260	106	165	ECME.Nelder.Mead	355	143	307	153	148
	ECME.BFGS	302	125	155	73	108	ECME.BFGS	326	134	294	122	133
	MCECM.Nelder.Mead	635	153	367	55	111	MCECM.Nelder.Mead	336	118	266	97	108
	MCECM.BFGS	349	66	249	52	91	MCECM.BFGS	296	104	246	84	92
Normalised BIC	ECM.Nelder.Mead	-6.98	-7.07	-7.16	-7.21	-7.24	ECM.Nelder.Mead	-6.61	-6.26	-6.61	-6.60	-6.61
	ECM.BFGS	-6.90	-7.07	-7.11	-7.21	-7.24	ECM.BFGS	-6.61	-6.26	-6.61	-6.60	-6.61
	ECME.Nelder.Mead	-7.02	-7.10	-7.15	-7.21	-7.24	ECME.Nelder.Mead	-6.61	-6.26	-6.61	-6.60	-6.61
	ECME.BFGS	-6.98	-7.09	-7.15	-7.21	-7.24	ECME.BFGS	-6.61	-6.26	-6.61	-6.60	-6.61
	MCECM.Nelder.Mead	-6.87	-7.07	-7.16	-7.21	-7.24	MCECM.Nelder.Mead	-6.61	-6.26	-6.61	-6.60	-6.61
	MCECM.BFGS	-6.87	-7.07	-7.15	-7.21	-7.24	MCECM.BFGS	-6.61	-6.26	-6.61	-6.60	-6.61
Classification rate	ECM.Nelder.Mead	0.499	0.621	0.814	0.851	0.869	ECM.Nelder.Mead	0.784	0.685	0.776	0.797	0.784
	ECM.BFGS	0.499	0.621	0.814	0.851	0.869	ECM.BFGS	0.784	0.685	0.776	0.797	0.784
	ECME.Nelder.Mead	0.504	0.636	0.822	0.851	0.869	ECME.Nelder.Mead	0.784	0.685	0.776	0.797	0.784
	ECME.BFGS	0.499	0.636	0.822	0.851	0.869	ECME.BFGS	0.784	0.685	0.779	0.797	0.784
	MCECM.Nelder.Mead	0.500	0.621	0.814	0.851	0.869	MCECM.Nelder.Mead	0.784	0.685	0.771	0.797	0.784
	MCECM.BFGS	0.499	0.621	0.822	0.851	0.869	MCECM.BFGS	0.784	0.685	0.776	0.797	0.784

		General VCMM (Copula with single parameter)						GMM				
Performance measures	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)	Algorithm	Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)
Computation time	ECM.Nelder.Mead	604	131	241	218	162	ECM	0.080	0.088	0.073	0.095	0.039
	ECM.BFGS	428	67	138	147	135	MCECM	0.104	0.098	0.099	0.125	0.060
	ECME.Nelder.Mead	267	267	183	281	214	ECME.Nelder.Mead	-	-	-	-	-
	ECME.BFGS	545	117	98	199	143	ECME.BFGS	-	-	-	-	-
	MCECM.Nelder.Mead	517	141	300	154	147						
	MCECM.BFGS	389	73	159	138	128						
Normalised BIC	ECM.Nelder.Mead	-7.04	-7.07	-7.09	-7.10	-7.08	ECM	-6.10	-6.10	-6.44	-6.44	-6.44
	ECM.BFGS	-6.98	-7.08	-7.09	-7.12	-7.14	MCECM	-6.10	-6.10	-6.44	-6.44	-6.44
	ECME.Nelder.Mead	-7.00	-7.10	-7.10	-7.11	-7.08	ECME.Nelder.Mead	-	-	-	-	-
	ECME.BFGS	-7.06	-7.08	-7.10	-7.12	-7.12	ECME.BFGS	-	-	-	-	-
	MCECM.Nelder.Mead	-7.04	-7.07	-7.09	-7.14	-7.08						
	MCECM.BFGS	-7.00	-7.08	-7.09	-7.12	-7.11						
Classification rate	ECM.Nelder.Mead	0.510	0.618	0.701	0.837	0.878	ECM	0.681	0.681	0.781	0.784	0.784
	ECM.BFGS	0.511	0.618	0.699	0.828	0.869	MCECM	0.681	0.681	0.781	0.784	0.784
	ECME.Nelder.Mead	0.507	0.630	0.748	0.828	0.878	ECME.Nelder.Mead	-	-	-	-	-
	ECME.BFGS	0.508	0.628	0.748	0.828	0.866	ECME.BFGS	-	-	-	-	-
	MCECM.Nelder.Mead	0.509	0.618	0.703	0.839	0.875						
	MCECM.BFGS	0.511	0.618	0.701	0.828	0.872						

Table 4.3.2: The best estimated vine copula model for the data set AIS and BCW with the highest classification rate and smallest normalised BIC. The variable encoding is given as follows: (AIS) 1: LBM, 2: Wt, 3: BMI, 4: WBC, 5: PBF and (BCW) 1: PSE, 2: ES, 3: EC and 4: ECP. For marginal distributions (Left column), the estimated marginal distributions and parameters for each cluster are shown. For vine tree structure (Right column), the first and second tree level of the estimated vine copula models are shown here. The number 1,5 represents the edge of the tree level, letter N is the abbreviation of the copula and the true parameter value and corresponding Kendall's τ of the pair copula are given inside the parenthesis (parameter(s)/Kendall's τ) near the letter. The meaning of the abbreviation for marginal distribution and copula families is shown in the appendix B.1 and B.2.

	Marginal distributions		Vine tree structure							
Variable	Cluster 1	Cluster 2	Cluster 1				Cluster 2			
			Tree 1		Tree 2		Tree 1		Tree 2	
AIS										
General VCMM (Copula with single parameter) (Hierarchical)										
Classification rate : 92.4% Normalised BIC : 23.83										
1	$llogis(12.42, 55.34)$	$lnorm(4.3, 0.13)$	2,5	C(1.53/0.43)	1,5;2	R90G(-3.38/-0.7)	2,1	SG(9.87/0.9)	3,1;2	R90G(-1.32/-0.24)
2	$\mathcal{N}(68.59, 12.29)$	$lnorm(4.4, 0.14)$	2,1	T(0.94,6.26/0.78)	3,1;2	N(-0.25/-0.16)	3,2	N(0.83/0.62)	5,2;3	C(0.2/0.09)
3	$lnorm(3.09, 0.12)$	$llogis(18.22, 23.46)$	3,2	N(0.87/0.67)	4,2;3	R270C(0/0)	5,3	T(0.64,11.61/0.45)	4,3;5	R270C(-0.17/-0.08)
4	$lnorm(1.93, 0.24)$	$lnorm(1.93, 0.26)$	4,3	F(1.56/0.17)			5,4	F(2.53/0.26)		
5	$\Gamma(11.04, 0.62)$	$snorm(8.81, 2.28, 4.3)$								
BCW										
General VCMM (Copula with single parameter) (Hierarchical scale)										
Classification rate : 87.8% Normalised BIC : -7.08										
1	$snorm(2.01, 0.74, 2.4)$	$sstd(4.45, 2.5, 4.36, 2.79)$	4,1	F(0.59/0.07)	2,1;4	R270C(-0.19/-0.08)	2,1	N(-0.3/-0.19)	3,1;2	SC(0.13/0.06)
2	$\Gamma(39.12, 313.49)$	$llogis(12.5, 0.14)$	4,2	SG(1.27/0.22)	3,2;4	R270G(-1.05/-0.04)	3,2	SG(1.44/0.31)	4,2;3	SJ(1.11/0.06)
3	$snorm(0.16, 0.12, 190.82)$	$sstd(0.48, 0.19, 7.18, 1.94)$	4,3	T(0.77,30/0.56)			4,3	SG(1.82/0.45)		
4	$\mathcal{N}(0.07, 0.03)$	$sstd(0.19, 0.04, 9.8, 1.41)$								

2. The best estimated vine copula model for VCMM algorithm

The best estimated vine copula model with the highest classification rate and smallest normalised BIC for AIS and BCW is shown in the Table 4.3.2. For AIS data set, General VCMM (Copula with single parameter) with Hierarchical performs the best with 92.4% classification rate. As the same as expected before, the variable WBC for both clusters is fitted by the same distribution, which is log-normal distribution with the same mean and very similar standard deviation. For PBE female data with a positive skew, it is fitted by skew normal distribution. About the estimated vine tree structure pairs with large Kendall's tau, the pair between LBM and Wt for male is fitted by survival Gumbel with asymmetric tail. However, the female data is finally fitted by symmetric t copula, where the asymmetry level of tail for the female data is not as high as male. For the pair between Wt and BMI, the female and male data are both fitted by Gaussian copula.

Also, for BCW data set, General VCMM (Copula with single parameter) with Hierarchical scale performs the best with 87.8% classification rate. Due to the skewness of variables PSE and EC, the estimated marginal densities for PSE and EC are finally fitted by skew normal distribution and skew Student's t distribution respectively. About the strong dependency with asymmetric tail between EC and ECP, female data is fitted by symmetric t copula and male data is fitted by survival Gumbel with asymmetric tail.

For other VCMM models and initialisation strategies, the estimated vine copula models are shown in the Appendix C.2.

3. Vuong test for the comparisons between VCMM model and Gaussian model

In the last part, we have concluded that for clustering non-Gaussian data, it is more recommended to use General VCMM. We still would like to determine whether there is enough statistical evidence to support our suggestion. Therefore, Voung test with BIC correction [Vuong, 1989] [Desmarais and Harden, 2013] is used

Table 4.3.3: The z statistics of the the Vuong test with BIC correction, where the colour yellow , orange , green , blue represent $z \leq -1.65, -1.65 < z \leq 0, 0 < z \leq 1.65$ and $1.65 < z$ respectively. (-1.65 / 1.65 is the z statistics at the significance level of 0.05.) Also, "-" represents not available as the conditions (2.5.5) or (2.5.6) is fulfilled for GMM algorithm.

Algorithm	Initialization strategies: Random	Kmeans	Hierarchical	Kmeans (scale)	Hierarchical (scale)	Initialization strategies: Random	Kmeans	Hierarchical (scale)	Kmeans (scale)	Hierarchical
	AIS					**BCW**				
	General VCMM and Gaussian VCMM									
ECM.Nelder.Mead	12.012	10.720	2.931	5.212	1.806	-3.681	-6.212	-5.423	-5.664	-6.083
ECM.BFGS	12.812	10.720	2.932	5.212	1.806	-2.904	-6.212	-4.738	-5.665	-6.077
ECME.Nelder.Mead	12.606	10.288	2.931	1.899	1.806	-4.365	-6.452	-5.393	-5.663	-6.124
ECME.BFGS	13.911	10.288	2.932	-0.253	1.806	-3.678	-6.625	-5.357	-5.664	-6.082
MCECM.Nelder.Mead	14.442	10.720	2.931	5.212	1.806	-2.680	-6.212	-5.453	-5.665	-6.082
MCECM.BFGS	12.812	10.720	2.932	5.212	1.806	-2.684	-6.212	-5.440	-5.664	-6.082
	General VCMM (Copula with single parameter) and Gaussian VCMM									
ECM.Nelder.Mead	12.649	10.602	1.251	6.249	1.425	-4.538	-6.595	-5.072	-5.467	-5.123
ECM.BFGS	15.929	10.602	1.422	6.249	1.425	-3.782	-6.582	-5.069	-5.672	-5.708
ECME.Nelder.Mead	12.152	9.464	1.422	2.578	1.425	-4.179	-6.674	-4.992	-5.548	-5.123
ECME.BFGS	16.296	9.464	1.422	0.343	1.425	-4.643	-6.745	-4.952	-5.669	-5.548
MCECM.Nelder.Mead	12.650	10.601	1.422	6.249	1.425	-4.421	-6.595	-5.150	-5.775	-5.127
MCECM.BFGS	16.308	10.602	1.251	6.248	1.425	-4.023	-6.582	-5.070	-5.671	-5.669
	General VCMM and GMM									
ECM	9.681	8.989	2.828	2.279	0.438	-6.953	-7.563	-6.880	-6.950	-7.439
MCECM	10.528	8.972	2.829	2.283	0.438	-6.686	-7.558	-6.871	-6.948	-7.426
ECME.Nelder.Mead	-	8.014	0.330	-1.567	-1.642	-	-	-	-	-
ECME.BFGS	-	-	-	-	-	-	-	-	-	-

here to determine whether the model fits the data statistically significantly better than the another one.

We follow the formula in the paper "Testing for zero inflation in count models: Bias correction for the Vuong test" [Desmarais and Harden, 2013] to calculate the z statistics of the the Vuong test with BIC correction. The Table 3.2.2 shows the z statistics of the the Vuong test between three pairs of models: 1. General VCMM and Gaussian VCMM, 2. General VCMM (Copula with single parameter) and Gaussian VCMM and 3. General VCMM and GMM. The interpretation of the z statistics is in the following:

- $z \leq -1.65$: The first model is preferred with significance level of 0.05.
- $-1.65 < z \leq 0$: The first model is preferred, but not significantly.
- $0 < z \leq 1.65$: The second model is preferred, but not significantly.
- $1.65 < z$: The second model is preferred with significance level of 0.05.

For data BCW, all the z statistics are smaller than -1.65, so we can conclude that General VCMM / General VCMM (Copula with single parameter) is preferred significantly over Gaussian VCMM or GMM with significance level of 0.05. This is what we can expect because the high flexibility of VCMM algorithm should fit the BCW data with non-Gaussian data better. However, For data AIS, most algorithms are with z statistics larger 1.65, which means the Gaussian VCMM or GMM are preferred. Note that the BIC value tends to select a model that reasonably approximates the density. However, the model does not always give a good clustering partition. Such model examples can be found in Punzo and McNicholas [2016] and Scrucca et al. [2016]. Although the result in Vuong test with BIC correction is out of our expectation, referring to the Table 4.3.1, the classification rate of general VCMM and General VCMM (Copula with single parameter) with Hierarchical (scale) is still higher than Gaussian VCMM. For clustering non-Gaussian data, general VCMM is still more recommended.

4. Ranking of the performance for each modelling method

The Table 4.3.4 shows the ranking of the performance of EM algorithms and the initialization strategies for AIS and BCW data set by VCMM and GMM algorithm. As we mentioned before, based on the result from real data sets, we recommend using Hierarchical scale as the initialization strategy. With regard to classification rate, the methods with top ranking here are using Hierarchical scale. Also, among the EM algorithms with Hierarchical scale, you can see that needed computation time of MCECM.BFGS is mostly the least. Therefore, we would recommend using general VCMM with MCECM.BFGS and Hierarchical scale for clustering non Gaussian data.

Table 4.3.4: The top 8 and bottom 4 performance of EM algorithms and the initialization strategies for the data set of AIS and BCW by using VCMM and GMM algorithm, where the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3).

AIS

General VCMM

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	MCECM.BFGS	Hierarchical scale	0.924	23.92	277
2	ECM.BFGS	Hierarchical scale	0.924	23.92	303
3	MCECM.Nelder.Mead	Hierarchical scale	0.924	23.92	488
4	ECME.BFGS	Hierarchical scale	0.924	23.92	527
5	ECM.Nelder.Mead	Hierarchical scale	0.924	23.92	640
6	ECME.Nelder.Mead	Hierarchical scale	0.924	23.92	914
7	ECME.BFGS	Kmeans scale	0.914	23.52	359
8	ECM.BFGS	Hierarchical	0.914	24.21	149
27	ECM.Nelder.Mead	Random	0.580	25.60	402
28	ECM.BFGS	Random	0.576	25.81	386
29	MCECM.BFGS	Random	0.576	25.81	414
30	ECME.Nelder.Mead	Random	0.572	25.76	478

Gaussian VCMM

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	MCECM.BFGS	Hierarchical scale	0.914	23.57	151
2	ECM.BFGS	Hierarchical scale	0.914	23.57	164
3	ECM.Nelder.Mead	Hierarchical scale	0.914	23.57	256
4	MCECM.Nelder.Mead	Hierarchical scale	0.914	23.57	256
5	MCECM.BFGS	Kmeans	0.914	23.57	348
6	ECME.BFGS	Hierarchical scale	0.914	23.57	356
7	ECM.BFGS	Kmeans	0.914	23.57	392
8	ECME.Nelder.Mead	Hierarchical scale	0.914	23.57	452
27	MCECM.Nelder.Mead	Random	0.658	23.06	352
28	ECM.Nelder.Mead	Random	0.658	23.06	371
29	ECME.BFGS	Random	0.658	23.06	524
30	ECME.Nelder.Mead	Random	0.658	23.06	787

General VCMM (Copula with single parameter)

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	MCECM.BFGS	Hierarchical	0.924	23.83	154
2	ECM.Nelder.Mead	Hierarchical	0.924	23.83	222
3	MCECM.BFGS	Hierarchical scale	0.924	23.85	272
4	ECM.BFGS	Hierarchical scale	0.924	23.85	295
5	ECME.BFGS	Hierarchical scale	0.924	23.85	496
6	ECM.Nelder.Mead	Hierarchical scale	0.924	23.85	538
7	MCECM.Nelder.Mead	Hierarchical scale	0.924	23.85	593
8	ECME.Nelder.Mead	Hierarchical scale	0.924	23.85	1003
27	ECME.BFGS	Random	0.606	26.46	699
28	ECME.Nelder.Mead	Random	0.589	25.70	383
29	MCECM.Nelder.Mead	Random	0.569	25.75	346
30	ECM.Nelder.Mead	Random	0.569	25.75	372

GMM

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	ECME.Nelder.Mead	Hierarchical	0.870	24.14	263
2	ECME.Nelder.Mead	Hierarchical scale	0.862	24.24	77
3	ECM	Kmeans	0.767	24.28	0.034
4	MCECM	Kmeans	0.767	24.28	0.047
5	MCECM	Kmeans scale	0.767	24.28	0.091
6	ECM	Kmeans scale	0.760	24.28	0.076

BCW

General VCMM

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	ECM.BFGS	Hierarchical scale	0.869	-7.24	74
2	MCECM.BFGS	Hierarchical scale	0.869	-7.24	91
3	ECME.BFGS	Hierarchical scale	0.869	-7.24	108
4	MCECM.Nelder.Mead	Hierarchical scale	0.869	-7.24	111
5	ECM.Nelder.Mead	Hierarchical scale	0.869	-7.24	112
6	ECME.Nelder.Mead	Hierarchical scale	0.869	-7.24	165
7	MCECM.BFGS	Kmeans scale	0.851	-7.21	52
8	ECM.Nelder.Mead	Kmeans scale	0.851	-7.21	55
27	ECM.Nelder.Mead	Random	0.499	-6.98	255
28	ECME.BFGS	Random	0.499	-6.98	392
29	ECM.BFGS	Random	0.499	-6.90	420
30	MCECM.BFGS	Random	0.499	-6.87	349

Gaussian VCMM

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	MCECM.BFGS	Kmeans scale	0.797	-6.60	84
2	ECM.BFGS	Kmeans scale	0.797	-6.60	85
3	MCECM.Nelder.Mead	Kmeans scale	0.797	-6.60	97
4	ECM.Nelder.Mead	Kmeans scale	0.797	-6.60	102
5	ECME.BFGS	Kmeans scale	0.797	-6.60	122
6	ECME.Nelder.Mead	Kmeans scale	0.797	-6.60	153
7	ECM.BFGS	Hierarchical scale	0.784	-6.61	86
8	MCECM.BFGS	Hierarchical scale	0.784	-6.61	92
27	MCECM.Nelder.Mead	Kmeans	0.685	-6.26	118
28	ECM.Nelder.Mead	Kmeans	0.685	-6.26	124
29	ECME.BFGS	Kmeans	0.685	-6.26	134
30	ECME.Nelder.Mead	Kmeans	0.685	-6.26	143

General VCMM (Copula with single parameter)

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	ECM.Nelder.Mead	Hierarchical scale	0.878	-7.08	162
2	ECME.Nelder.Mead	Hierarchical scale	0.878	-7.08	214
3	MCECM.Nelder.Mead	Hierarchical scale	0.875	-7.08	147
4	MCECM.BFGS	Hierarchical scale	0.872	-7.11	128
5	ECM.BFGS	Hierarchical scale	0.869	-7.14	135
6	ECME.BFGS	Hierarchical scale	0.866	-7.12	143
7	MCECM.Nelder.Mead	Kmeans scale	0.839	-7.14	154
8	ECM.Nelder.Mead	Kmeans scale	0.837	-7.10	218
27	ECM.Nelder.Mead	Random	0.510	-7.04	604
28	MCECM.Nelder.Mead	Random	0.509	-7.04	517
29	ECME.BFGS	Random	0.508	-7.06	545
30	ECME.Nelder.Mead	Random	0.507	-7.00	267

GMM

Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1	ECM	Hierarchical scale	0.784	-6.44	0.039
2	MCECM	Hierarchical scale	0.784	-6.44	0.060
3	ECM	Kmeans scale	0.784	-6.44	0.095
4	MCECM	Kmeans scale	0.784	-6.44	0.125
5	ECM	Hierarchical	0.781	-6.44	0.073
6	MCECM	Hierarchical	0.781	-6.44	0.125

5. The performance of the selected models in the flow chart 3.3.20

In the flow chart 3.3.20, we have came up with a way to select the suitable initialisation strategies for data clustering without characteristics information for general VCMM algorithm. We try to apply the method there to see how the performances for the real data set AIS and BCW are.

The real data set is not three dimensional and we haven't defined what "X" shape in higher dimensional data is, so we select the path that the data has no "X" shape. Then, for general VCMM algorithm, we should compare the BIC value of Hierarchical and Kmeans (scale) with ECM.BFGS or MCECM.BFGS and select the lower BIC. In order to save time, we don't try both and just take ECM.BFGS instead. Actually, from our simulation reuslt, the times needed for ECM.BFGS and MCECM.BFGS are quite similar.

According to the Table 4.3.1, the initialisation strategies with lower BIC for ECM.BFGS is Hierarchical and Kmeans (scale) for AIS and BCW respectively. The performances of the selected model and the best ranking model are shown in the Table 4.3.5. Although the classification rate of the selected model is not as good as the best ranking model, the difference of the classification rate is just less than 2% for both real data set AIS and BCW. Therefore, we think that by the flow chart proposed in the Figure 3.3.20, we still can choose a suitable initialisation strategy and EM algorithm for data clustering with general VCMM algorithm.

Table 4.3.5: The performances of the selected model and the best ranking model for general VCMM. *The model is selected by the lower BIC value, so the total computation time is equal to the summation of the computation time of the two models.

	AIS	
	Selected model	Best ranking model
Initialisation strategy	Hierarchical	Hierarchical (scale)
EM algorithm	ECM.BFGS	MCECM.BFGS
Total computation time (seconds)	149 (+219) = 368*	277
Normalised BIC	24.21	23.92
Classification rate	0.914	0.924
	BCW	
	Selected model	Best ranking model
Initialisation strategy	Kmeans (scale)	Hierarchical (scale)
EM algorithm	ECM.BFGS	ECM.BFGS
Total computation time (seconds)	61 (+232) = 293*	74
Normalised BIC	-7.21	-7.24
Classification rate	0.851	0.869

5 Conclusion

In this master thesis, we aim at assessing the performance of the iterative algorithm - Gaussian mixture model (GMM) algorithm and vine copula mixture model (VCMM) algorithm for clustering simulated data with different characteristics and real data. In particular, we focus on the influence on the performance by using different versions of EM algorithms for parameter estimation of the GMM and VCMM algorithms with various initialisation strategies and optimization.

The simulation studies show that a particular initialisation strategy, using different expectation and maximization (EM) algorithms for parameter estimation of VCMM doesn't impact the classification rate significantly, but this is not the case for GMM algorithm. For different sets of Gaussian simulated data, GMM algorithm with maximisation either (ECME) algorithm and the optimization method Nelder-Mead perform mostly the best with regard to classification rate. Using the best and the worst EM algorithm in GMM algorithm can lead to almost 30% difference in classification rate. However, the difference is much smaller in VCMM algorithm and it is just from 0% to 5%.

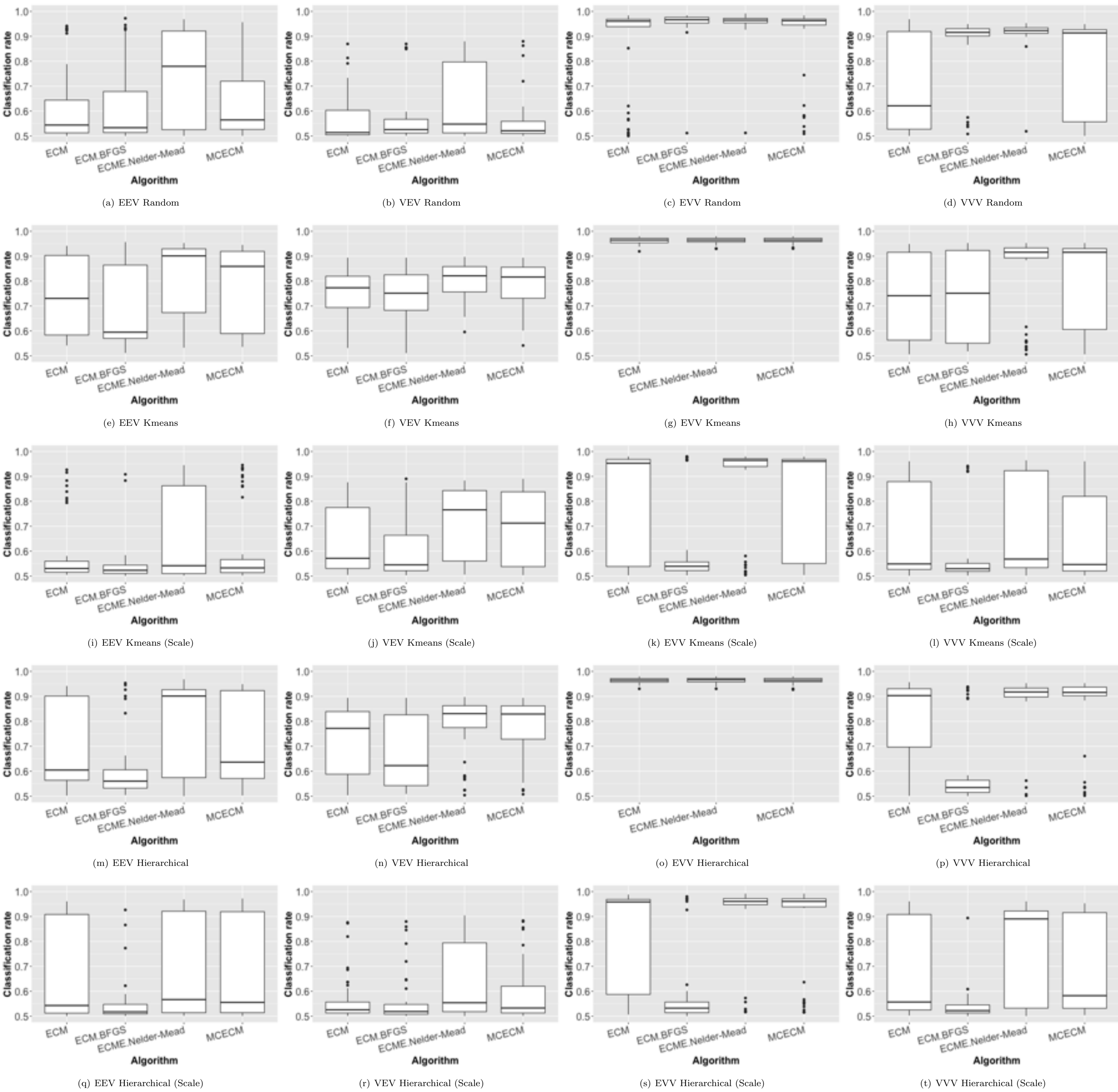
Although there is no significant effect on the performance regarding to classification rate for VCMM with different EM algorithms and optimization, it does affect the computation time. We found that generally, heuristic based optimization method (Nelder-Maad) takes more time than with gradient based optimization method (BFGS). Also, different initialisation strategies do have the strong effect on clustering accuracy. We have found that there is usually a strategy outperforming others on the classification rate for simulated data with different characteristics and we can summarise it as a flow chart, for example, the best strategy for clustering data with the clusters like a "X" is "Random", for data with large volume difference of clusters is standardization first and so on. However, the flow chart cannot be used directly for data clustering without characteristics information. So, we proposed a way to select a the initialization strategy by comparing the BIC value of two suggested strategies and the one with lower BIC is selected.

Lastly, we have tried the selected model for clustering two sets of real data with general VCMM algorithm and the clustering accuracy is almost as good as the best ranking model where the difference of the classification rate is just less than 2%.

A Appendices

A Appendix for Section 3.2 GMM Algorithm

A.1 Box plots of the classification rate



(a) EEV Random (b) VEV Random (c) EVV Random (d) VVV Random

(e) EEV Kmeans (f) VEV Kmeans (g) EVV Kmeans (h) VVV Kmeans

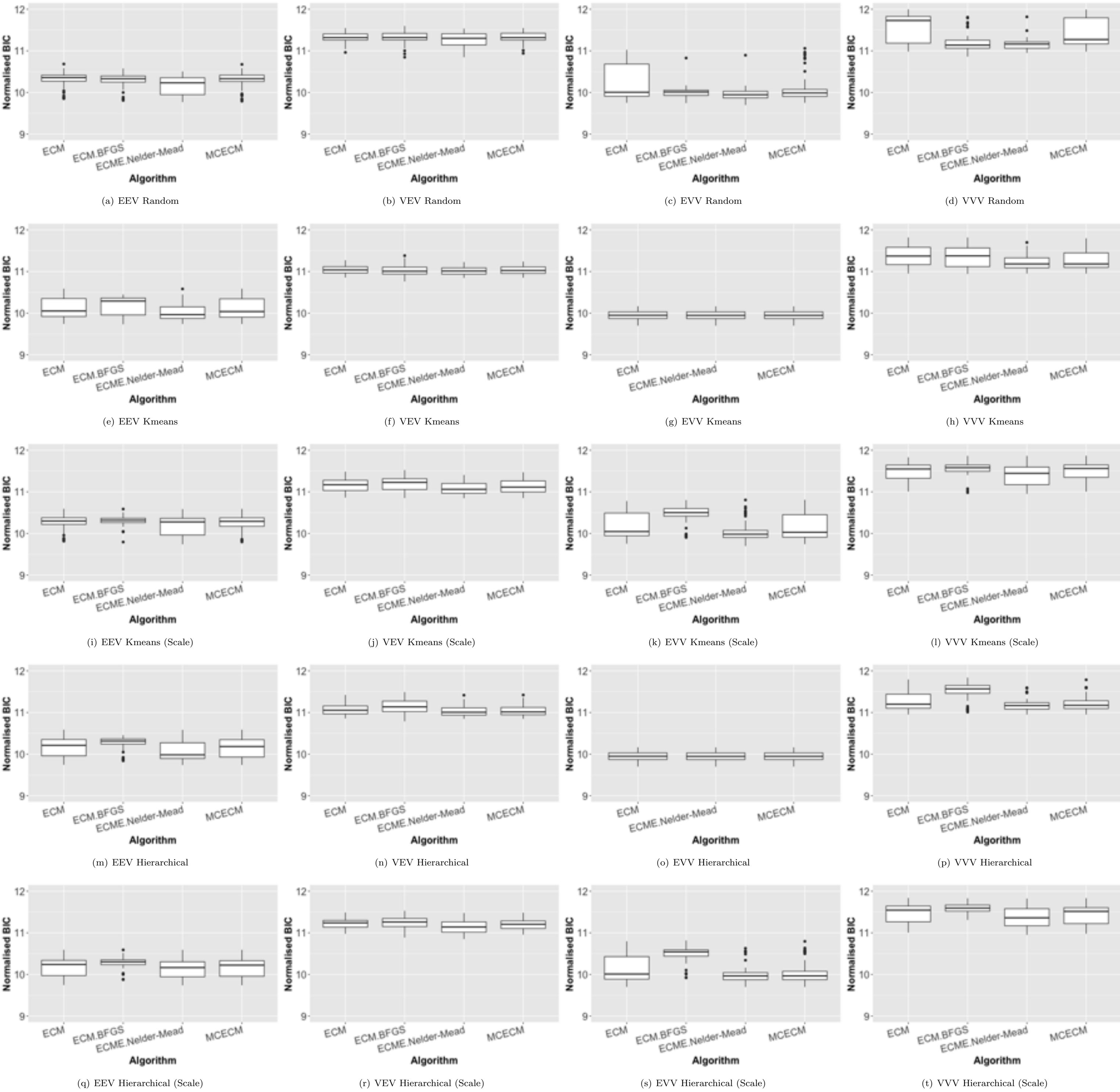
(i) EEV Kmeans (Scale) (j) VEV Kmeans (Scale) (k) EVV Kmeans (Scale) (l) VVV Kmeans (Scale)

(m) EEV Hierarchical (n) VEV Hierarchical (o) EVV Hierarchical (p) VVV Hierarchical

(q) EEV Hierarchical (Scale) (r) VEV Hierarchical (Scale) (s) EVV Hierarchical (Scale) (t) VVV Hierarchical (Scale)

Figure A.1: Box plots of the **classification rate** for the EM algorithms for different initialization strategies over 50 replications with n = 500 generated from different types of the Gaussian clustering models

A.2 Box plots of the normalised BIC



(a) EEV Random (b) VEV Random (c) EVV Random (d) VVV Random

(e) EEV Kmeans (f) VEV Kmeans (g) EVV Kmeans (h) VVV Kmeans

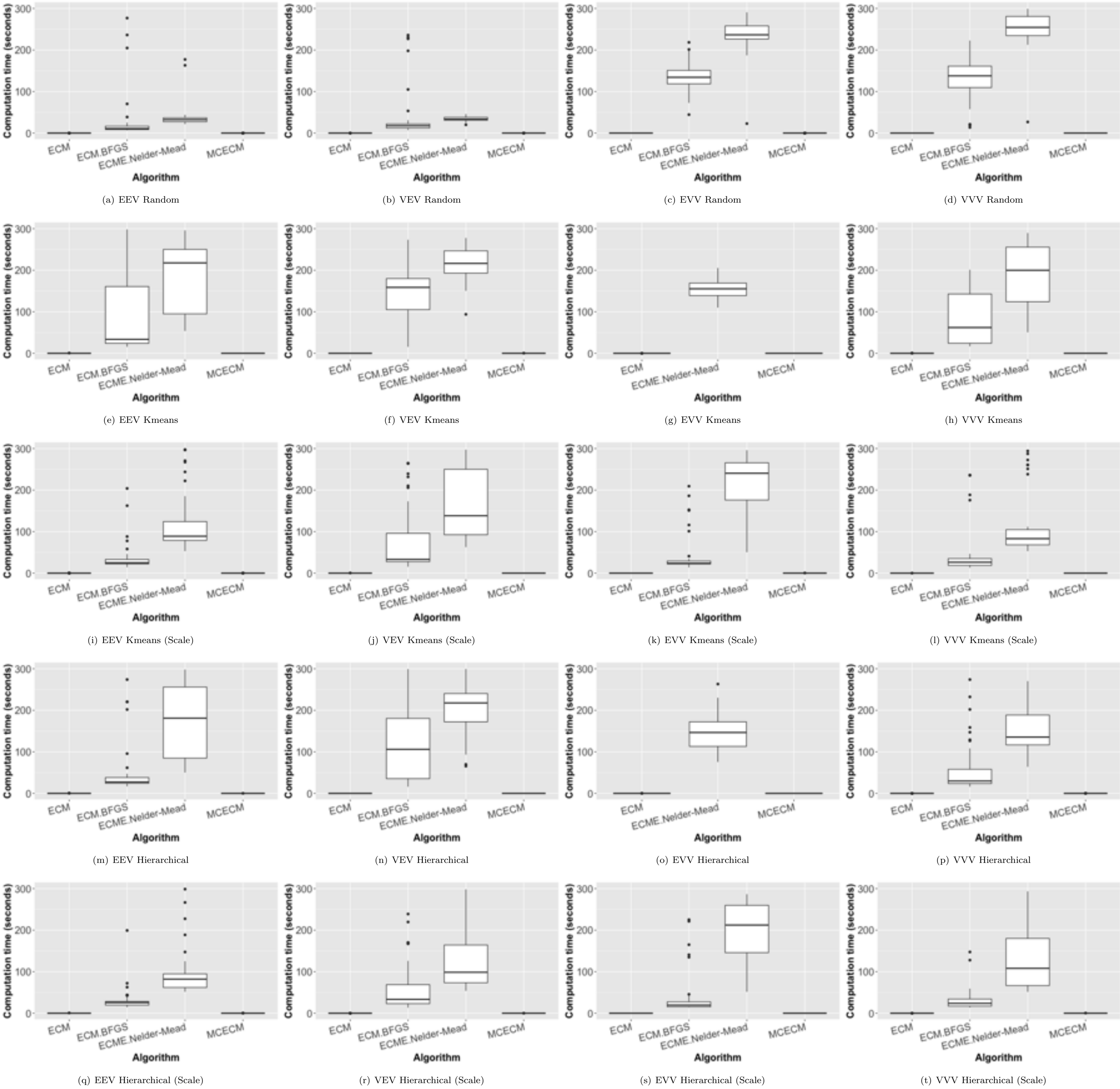
(i) EEV Kmeans (Scale) (j) VEV Kmeans (Scale) (k) EVV Kmeans (Scale) (l) VVV Kmeans (Scale)

(m) EEV Hierarchical (n) VEV Hierarchical (o) EVV Hierarchical (p) VVV Hierarchical

(q) EEV Hierarchical (Scale) (r) VEV Hierarchical (Scale) (s) EVV Hierarchical (Scale) (t) VVV Hierarchical (Scale)

Figure A.2: Box plots of the **normalised BIC** of the EM algorithms for different initialization strategies over 50 replications with n = 500 generated from different types of the Gaussian clustering models

A.3 Box plots of the computation time



(a) EEV Random (b) VEV Random (c) EVV Random (d) VVV Random

(e) EEV Kmeans (f) VEV Kmeans (g) EVV Kmeans (h) VVV Kmeans

(i) EEV Kmeans (Scale) (j) VEV Kmeans (Scale) (k) EVV Kmeans (Scale) (l) VVV Kmeans (Scale)

(m) EEV Hierarchical (n) VEV Hierarchical (o) EVV Hierarchical (p) VVV Hierarchical

(q) EEV Hierarchical (Scale) (r) VEV Hierarchical (Scale) (s) EVV Hierarchical (Scale) (t) VVV Hierarchical (Scale)

Figure A.3: Box plots of the total omputation time of the EM algorithms for different initialization strategies over 50 replications with n = 500 generated from different types of the Gaussian clustering models

B Appendix for Section 3.3 VCMM Algorithm

B.1 Abbreviation for marginal distributions

$\mathcal{N}(\mu, \sigma)$: normal distribution with mean μ and standard deviation σ.
$exp(\lambda)$: exponential distribution with rate parameter λ.
$lnorm(\mu, \sigma)$: log-normal distribution with mean μ and standard deviation σ on the logarithmic scale.
$logis(l, s)$: logistic distribution with location parameter l and scale parameter s.
$llogis(\alpha, \beta)$: log-logistic distribution with shape parameter α and scale parameter β.
$\Gamma(\alpha, \beta)$: gamma distribution with shape parameter α and rate parameter β.
$t_3(\mu, \sigma)$: Student's t distribution with mean parameter μ, standard deviation parameter σ, and degrees of freedom 3.
$snorm(\mu, \sigma, \lambda)$: Skewed normal distribution with with location parameter μ, scale parameter σ, skewness parameter λ.
$sstd(\mu, \sigma, \alpha, \lambda)$: Skewed student's t distribution with location parameter μ, scale parameter σ, shape parameter α (Degrees of freedom), skewness parameter λ.

B.2 Abbreviation for copula families

Table B.1: The abbreviation for copula families with different degrees of rotation.

Copula	Degrees of rotation (Anticlockwise)			
	0°	90°	180°	270°
Copulas with symmetric tail				
Gaussian	N	-	-	-
Student t	T	-	-	-
Frank	F	-	-	-
Copulas with asymmetric tail				
Clayton	C	R90C	R180C	R270C
Gumbel	G	R90G	R180G	R270G
Joe	J	R90J	R180J	R270J
BB1	BB1	R90BB1	R180BB1	R270BB1
BB6	BB6	R90BB6	R180BB6	R270BB6
BB8	BB8	R90BB8	R180BB8	R270BB8

B.3 Box plots of the classification rate

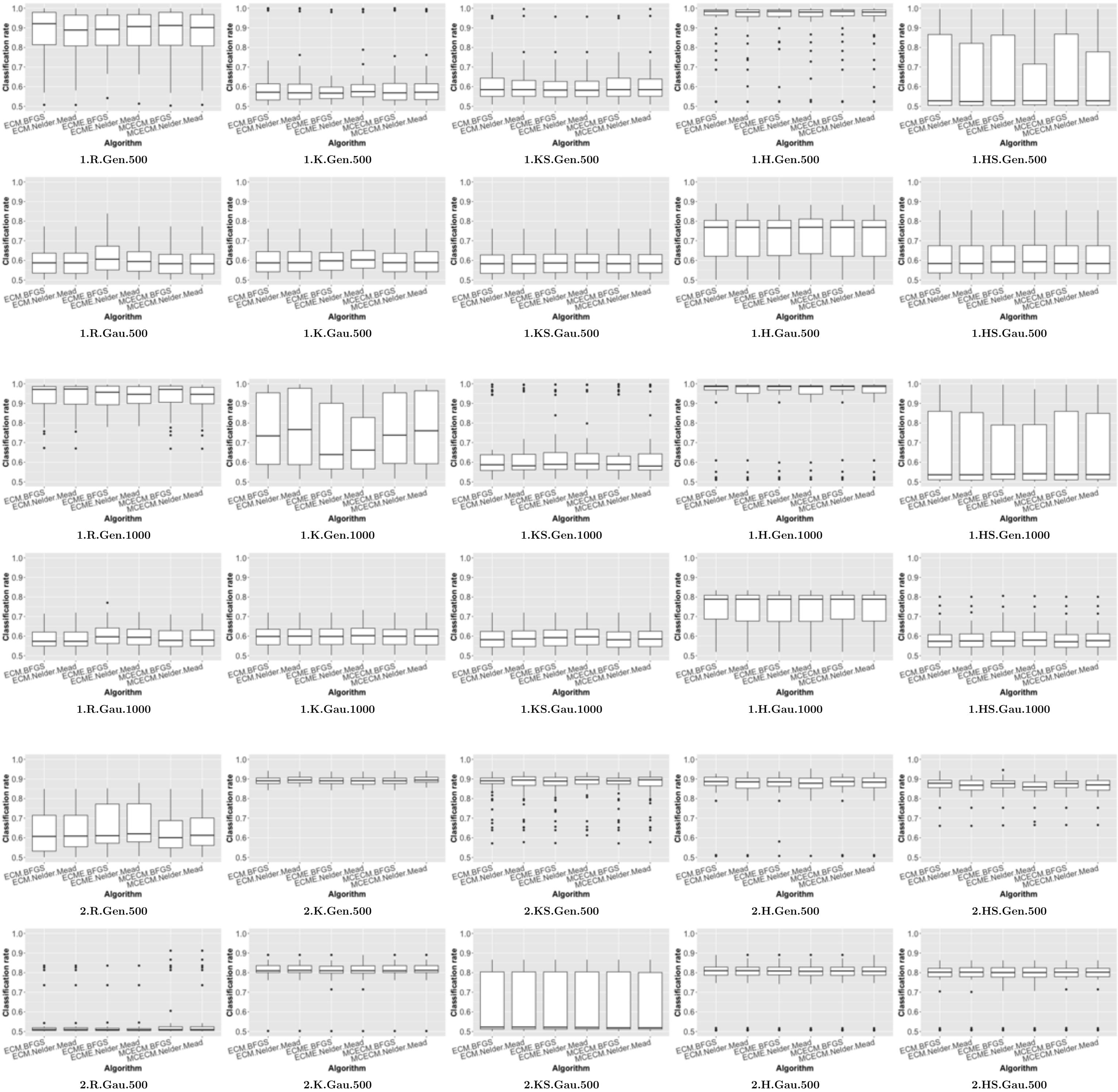


Figure B.1: Box plots of the **classification rate** for the EM algorithms using different initialization strategies for 50 replications from settings 1 and 2.

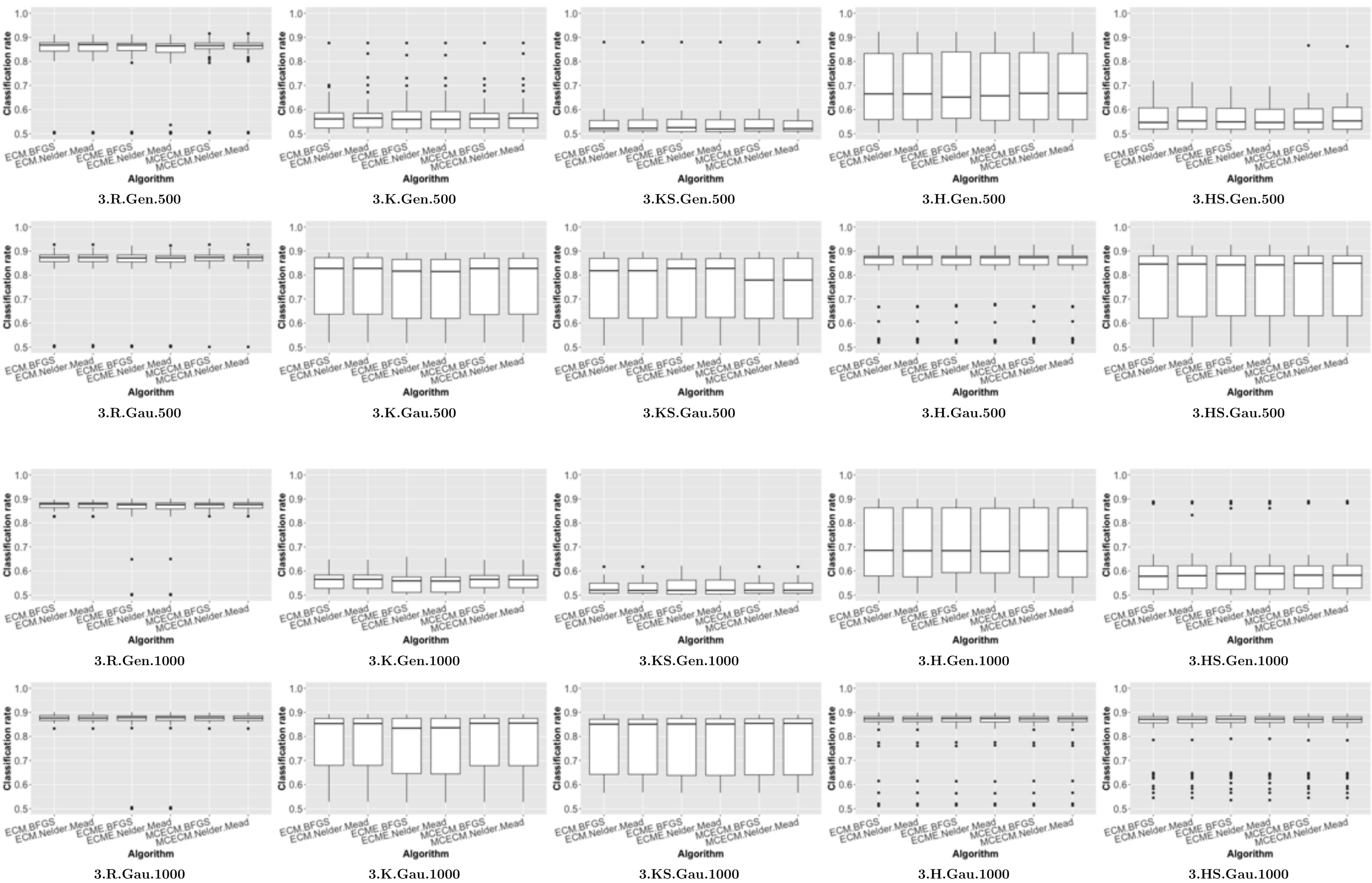


Figure B.2: Box plots of the **classification rate** for the EM algorithms using different initialization strategies for 50 replications from settings 3.

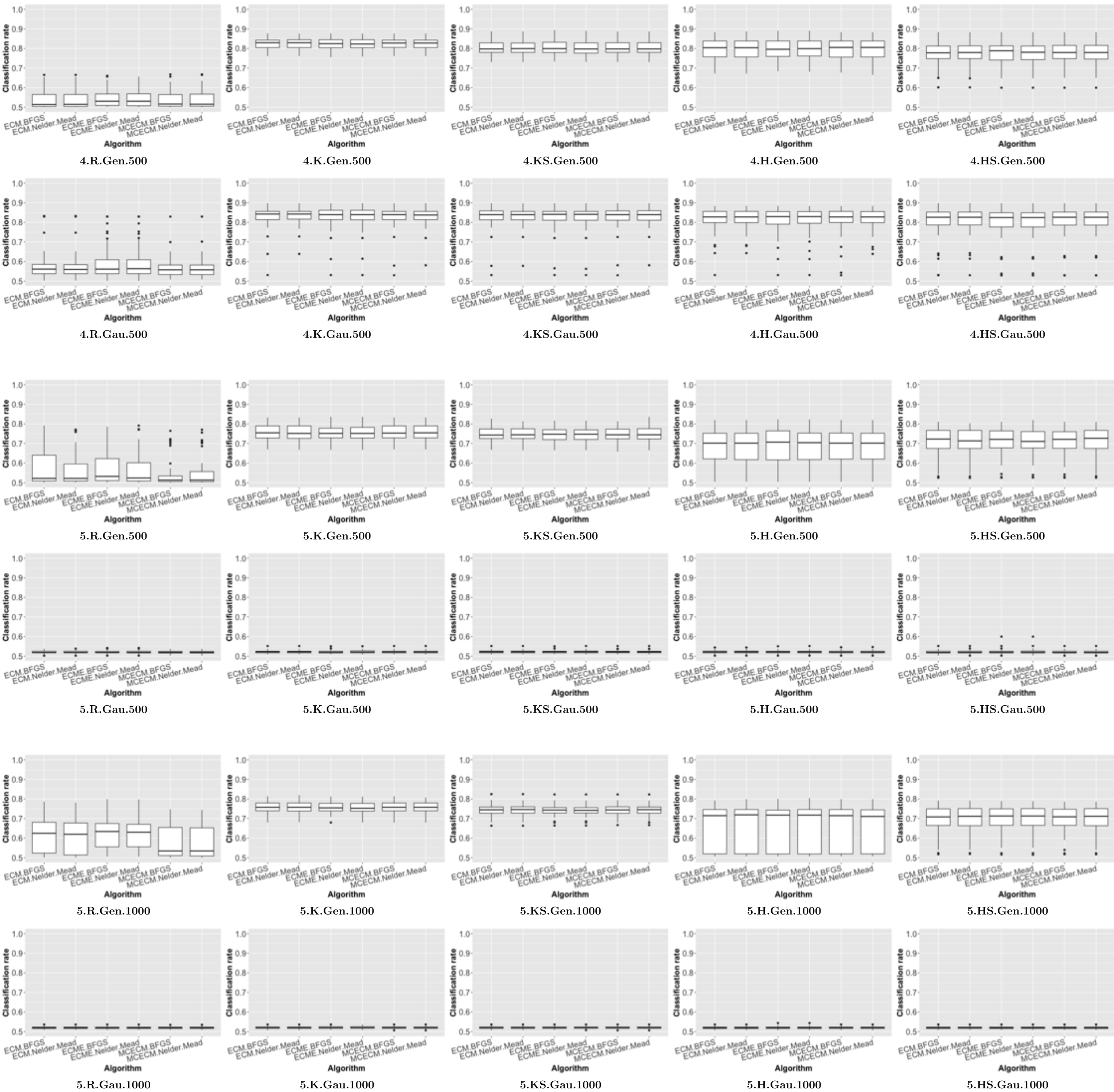


Figure B.3: Box plots of the **classification rate** for the EM algorithms using different initialization strategies for 50 replications from settings 4 and 5.

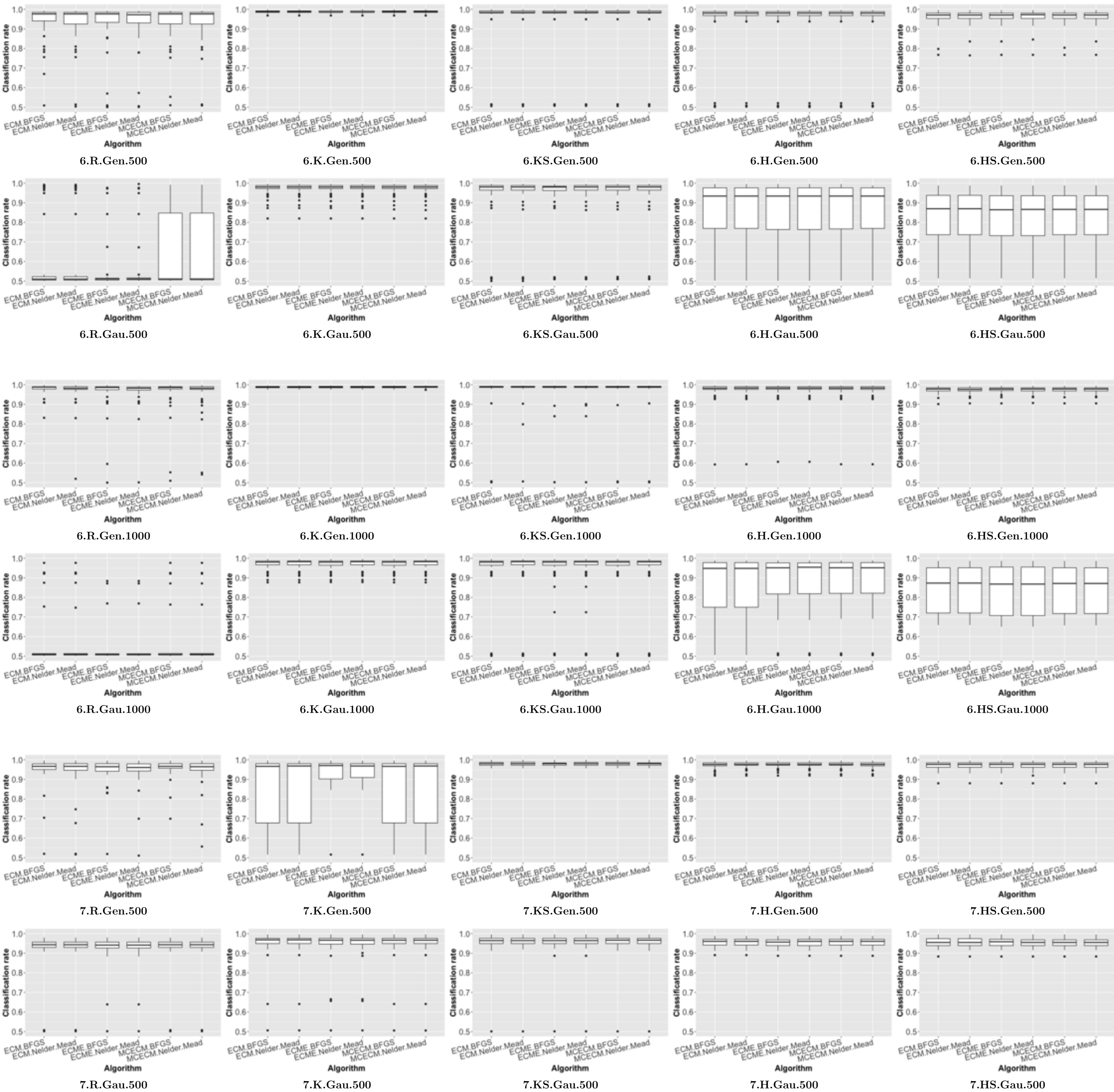


Figure B.4: Box plots of the **classification rate** for the EM algorithms using different initialization strategies for 50 replications from settings 6 and 7.

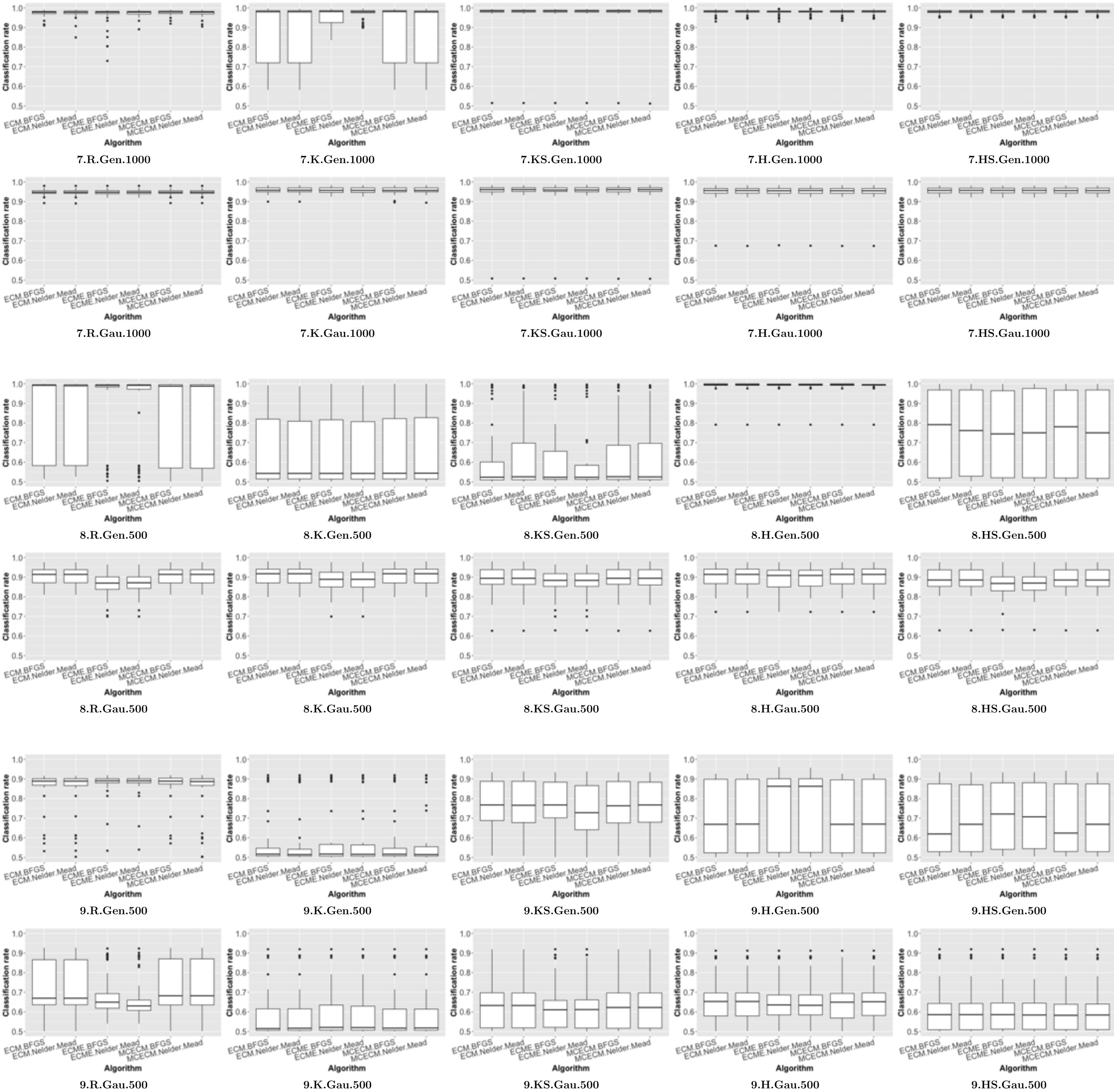


Figure B.5: Box plots of the **classification rate** for the EM algorithms using different initialization strategies for 50 replications from settings 7, 8 and 9.

B.4 Box plots of the normalised BIC

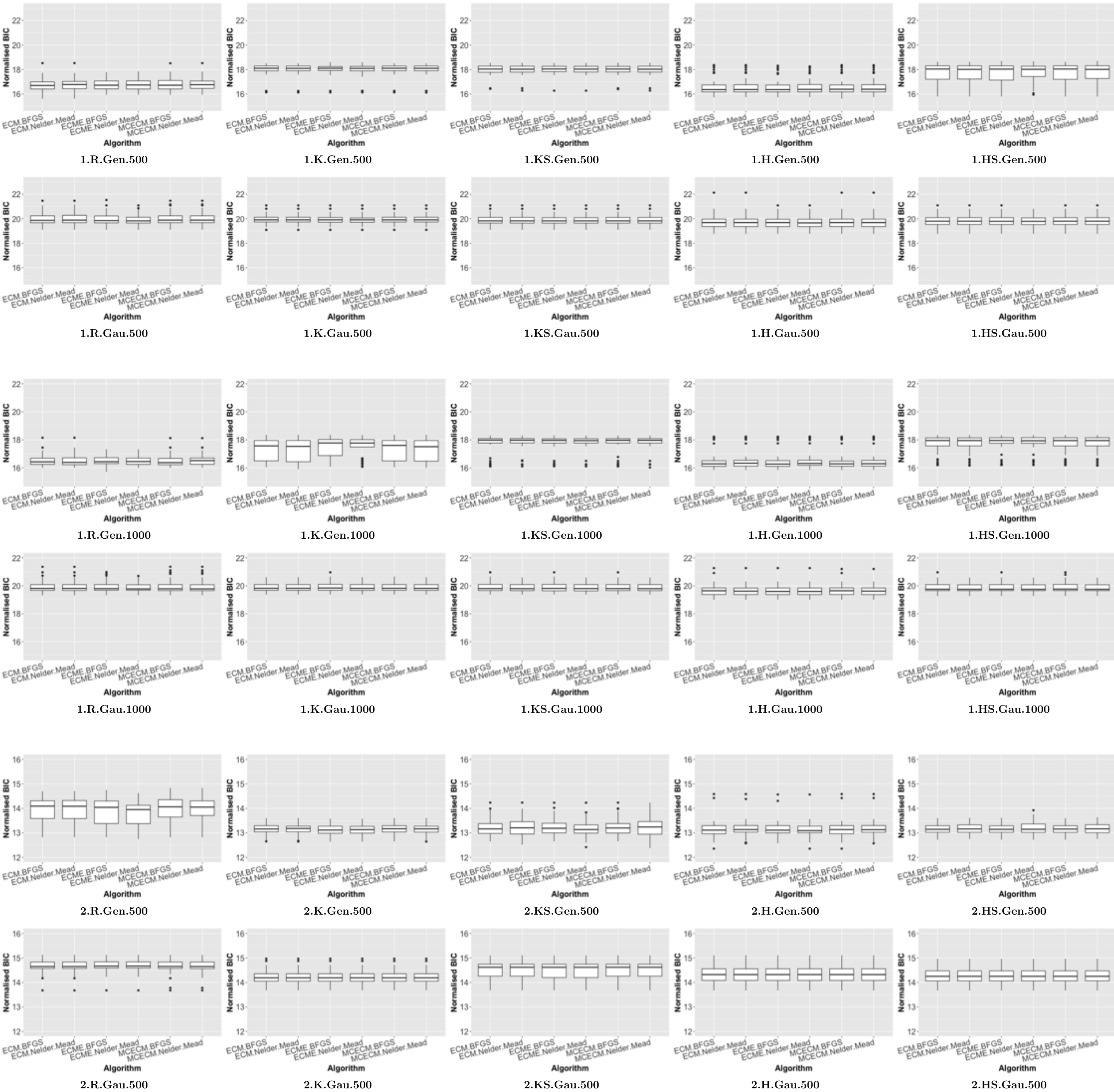


Figure B.6: Box plots of the **normalised BIC** for the EM algorithms using different initialization strategies for 50 replications from settings 1 and 2.

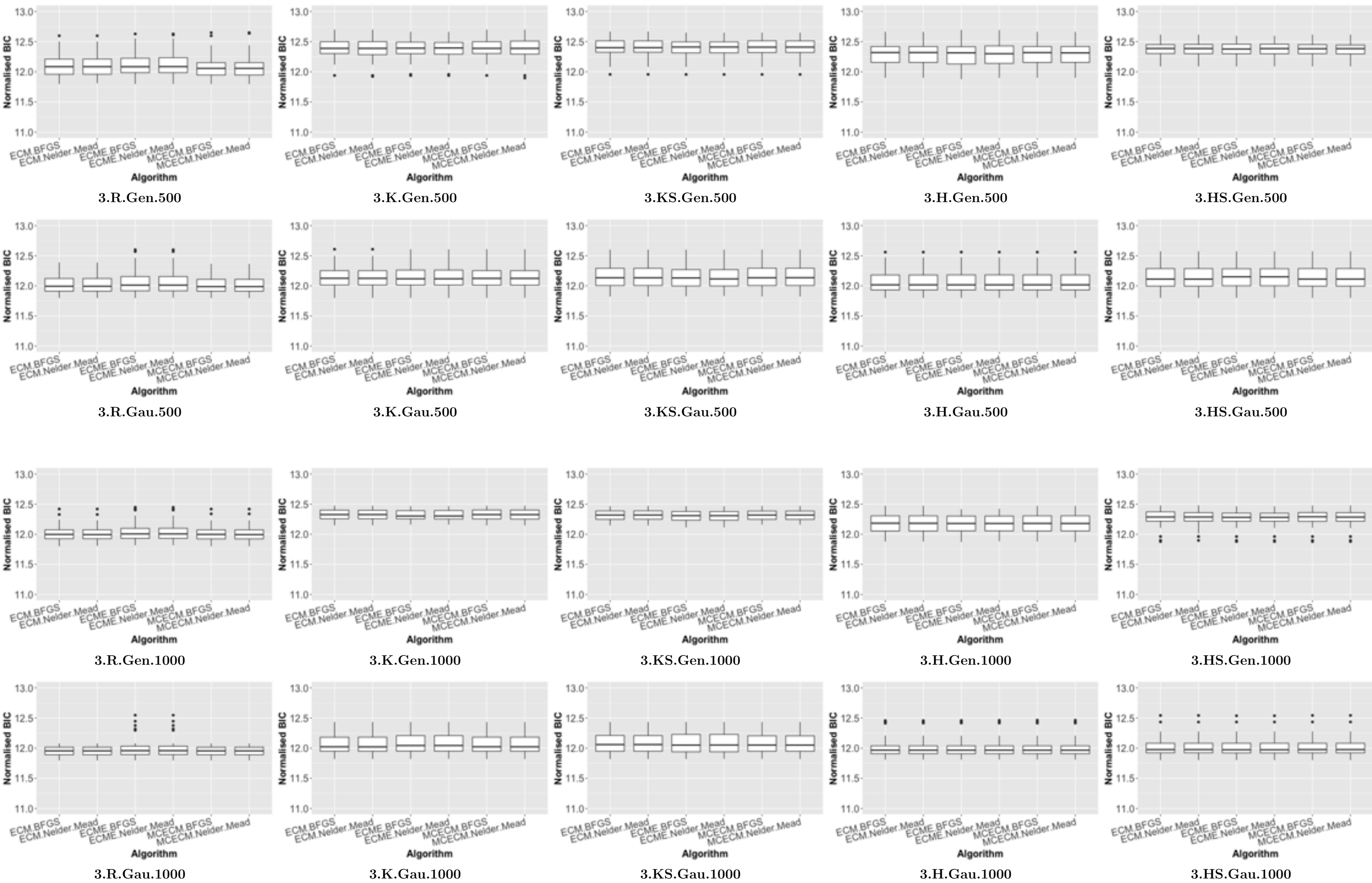


Figure B.7: Box plots of the **normalised BIC** for the EM algorithms using different initialization strategies for 50 replications from settings 3.

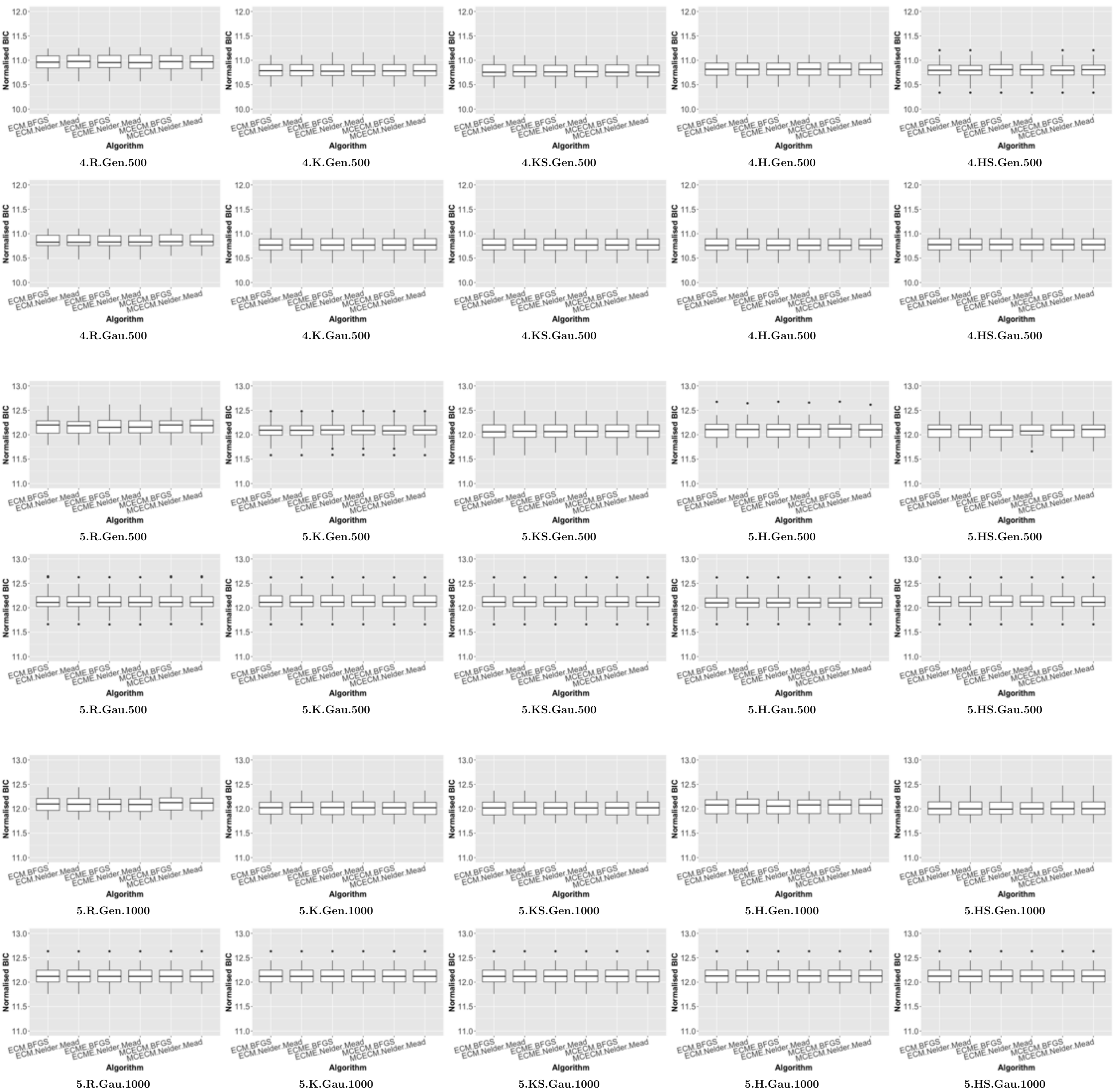


Figure B.8: Box plots of the **normalised BIC** for the EM algorithms using different initialization strategies for 50 replications from settings 4 and 5.

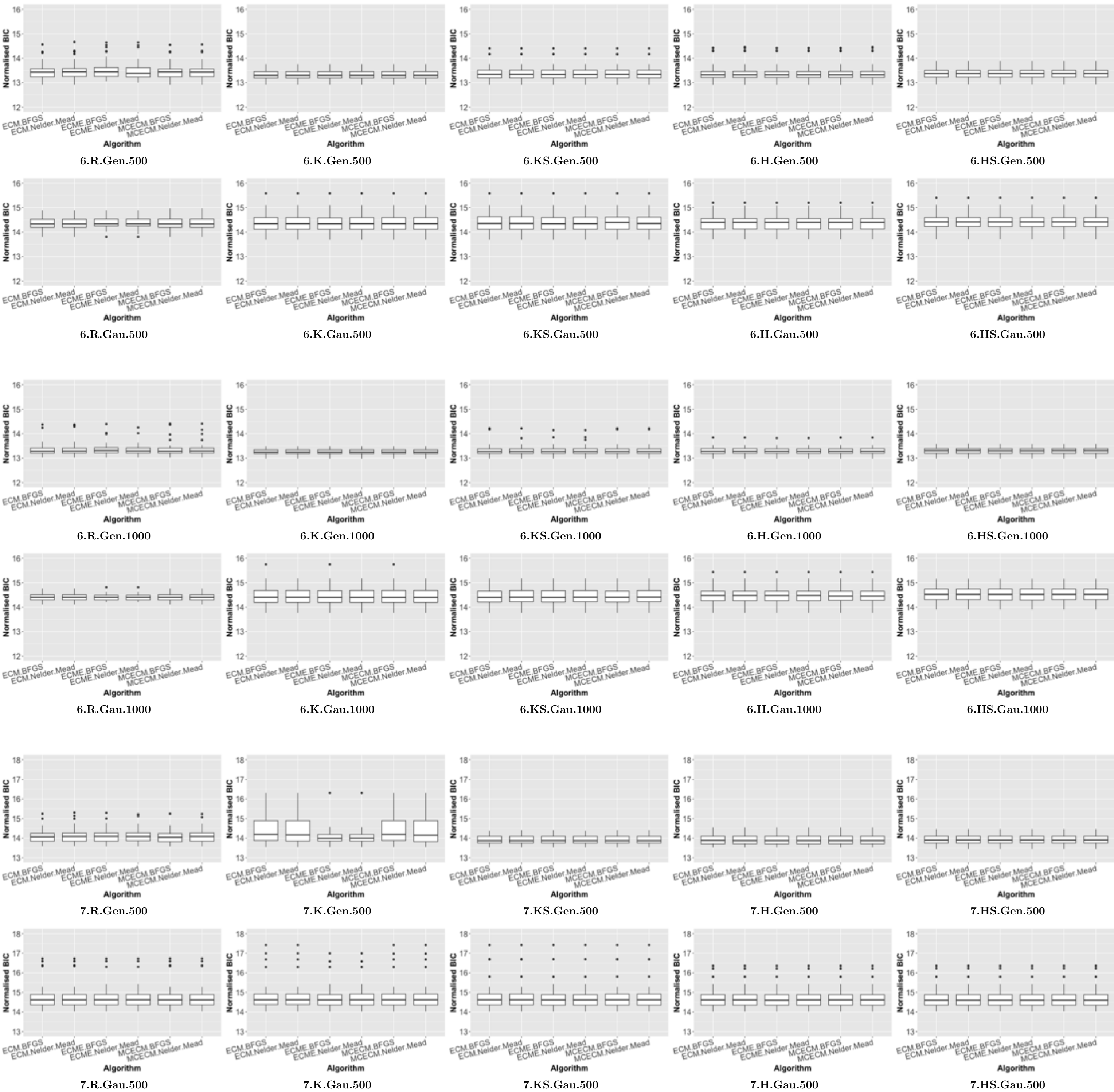
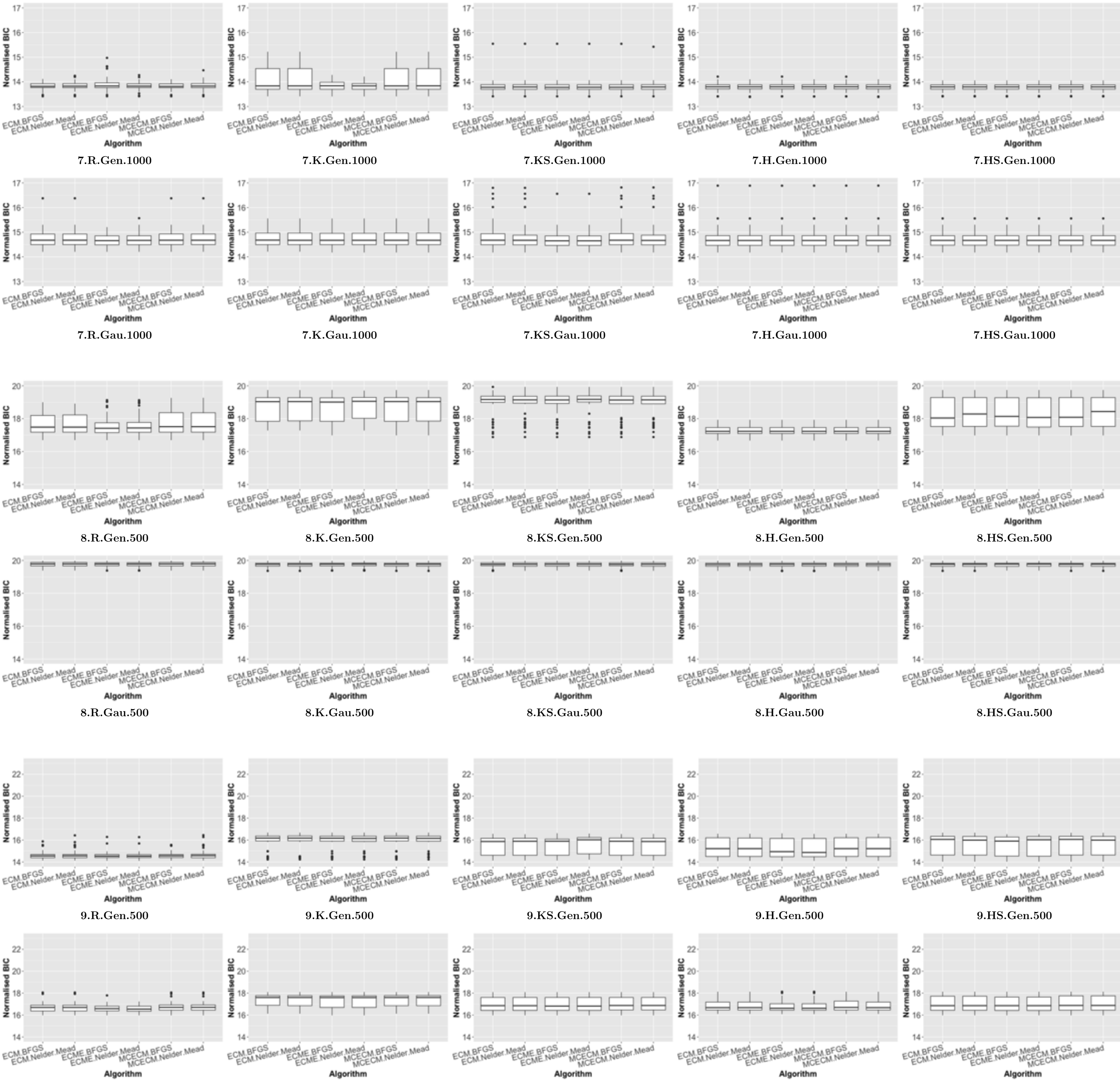


Figure B.9: Box plots of the **normalised BIC** for the EM algorithms using different initialization strategies for 50 replications from settings 6 and 7.



7.R.Gen.1000 | 7.K.Gen.1000 | 7.KS.Gen.1000 | 7.H.Gen.1000 | 7.HS.Gen.1000

7.R.Gau.1000 | 7.K.Gau.1000 | 7.KS.Gau.1000 | 7.H.Gau.1000 | 7.HS.Gau.1000

8.R.Gen.500 | 8.K.Gen.500 | 8.KS.Gen.500 | 8.H.Gen.500 | 8.HS.Gen.500

8.R.Gau.500 | 8.K.Gau.500 | 8.KS.Gau.500 | 8.H.Gau.500 | 8.HS.Gau.500

9.R.Gen.500 | 9.K.Gen.500 | 9.KS.Gen.500 | 9.H.Gen.500 | 9.HS.Gen.500

9.R.Gau.500 | 9.K.Gau.500 | 9.KS.Gau.500 | 9.H.Gau.500 | 9.HS.Gau.500

Figure B.10: Box plots of the **normalised BIC** for the EM algorithms using different initialization strategies for 50 replications from settings 7, 8 and 9.

B.5 Box plots of the computation time

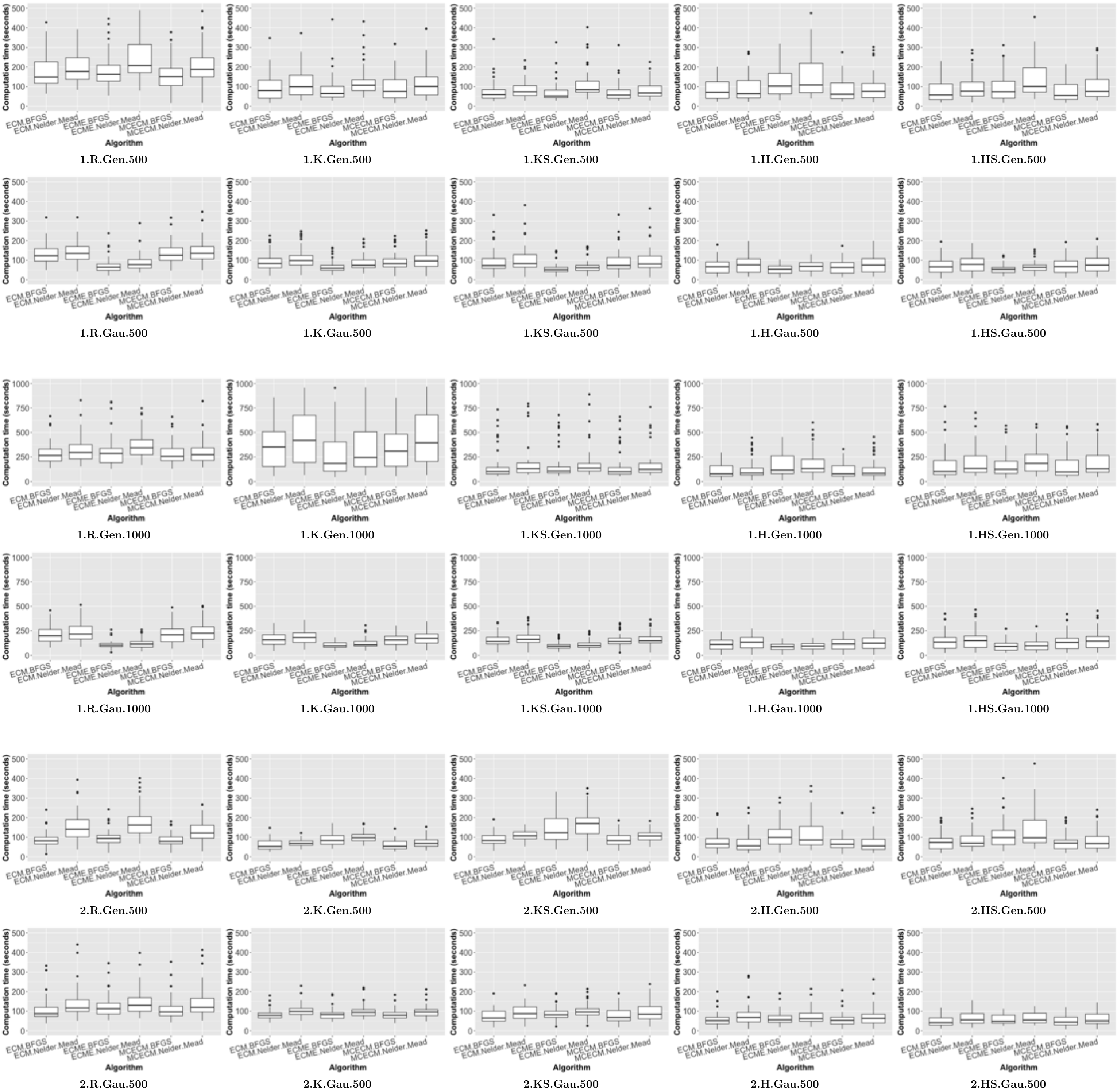


Figure B.11: Box plots of the **computation time** for the EM algorithms using different initialization strategies for 50 replications from settings 1 and 2.

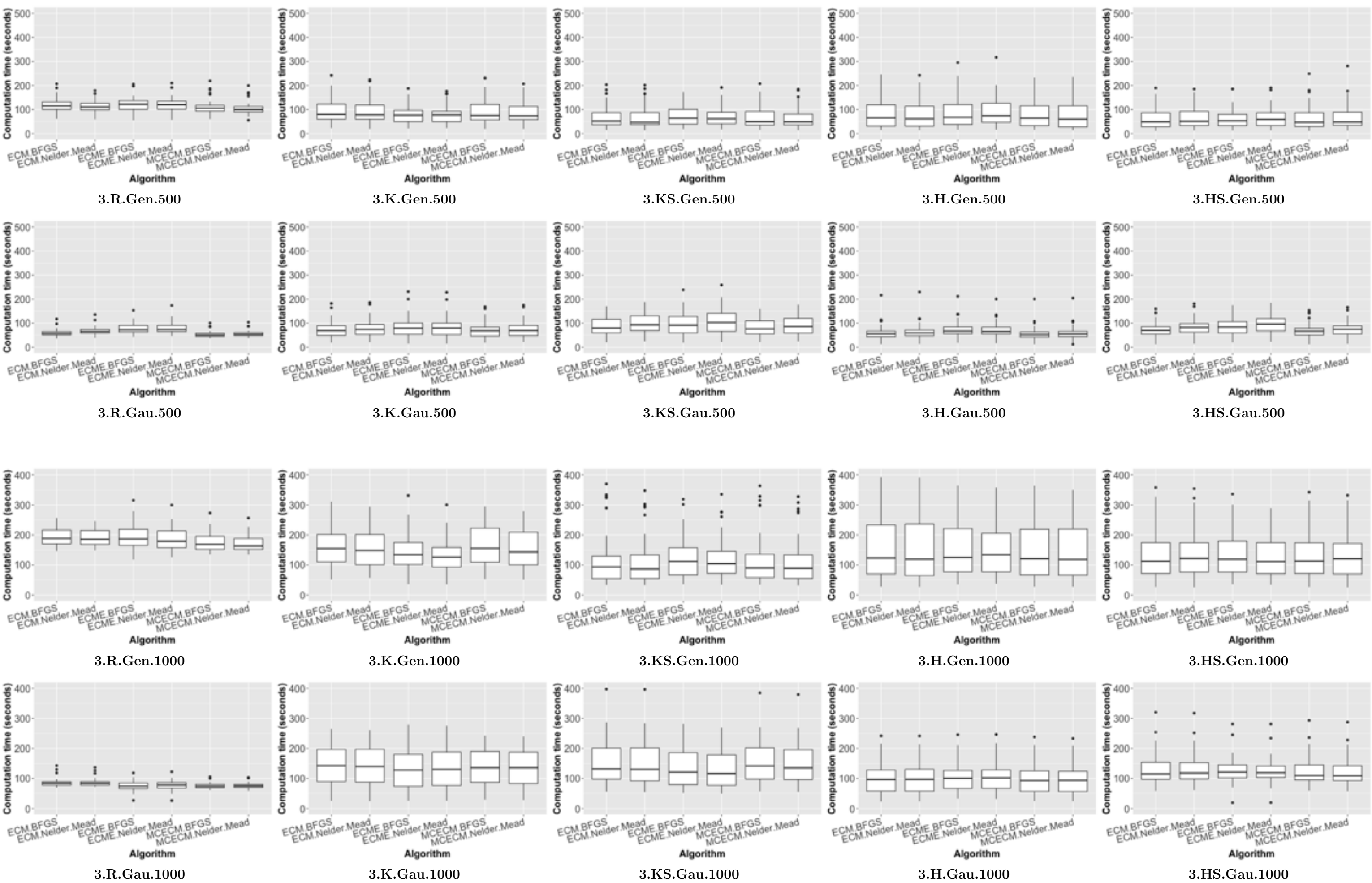


Figure B.12: Box plots of the **computation time** for the EM algorithms using different initialization strategies for 50 replications from settings 3.

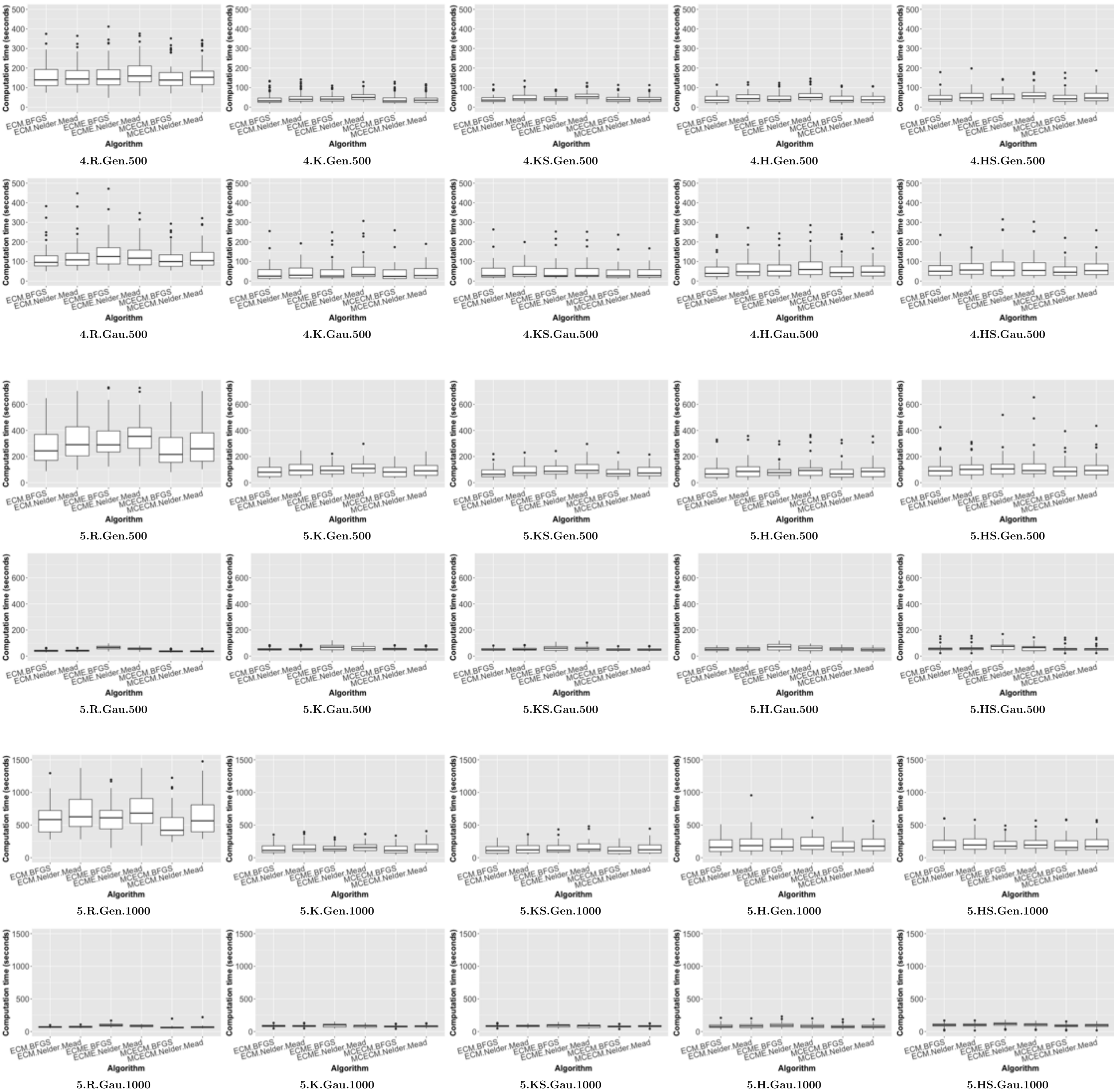


Figure B.13: Box plots of the **computation time** for the EM algorithms using different initialization strategies for 50 replications from settings 4 and 5.

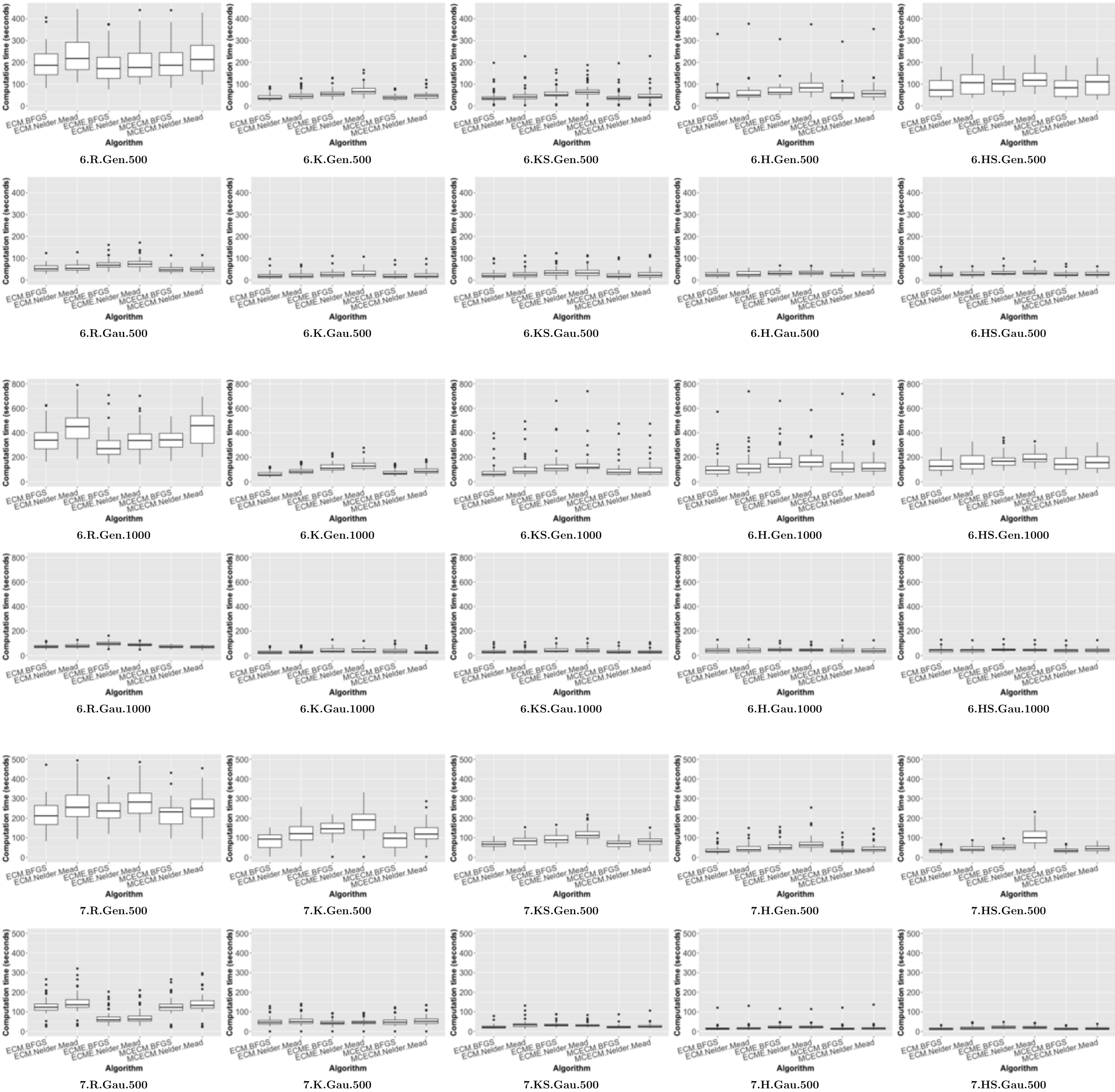


Figure B.14: Box plots of the **computation time** for the EM algorithms using different initialization strategies for 50 replications from settings 6 and 7.

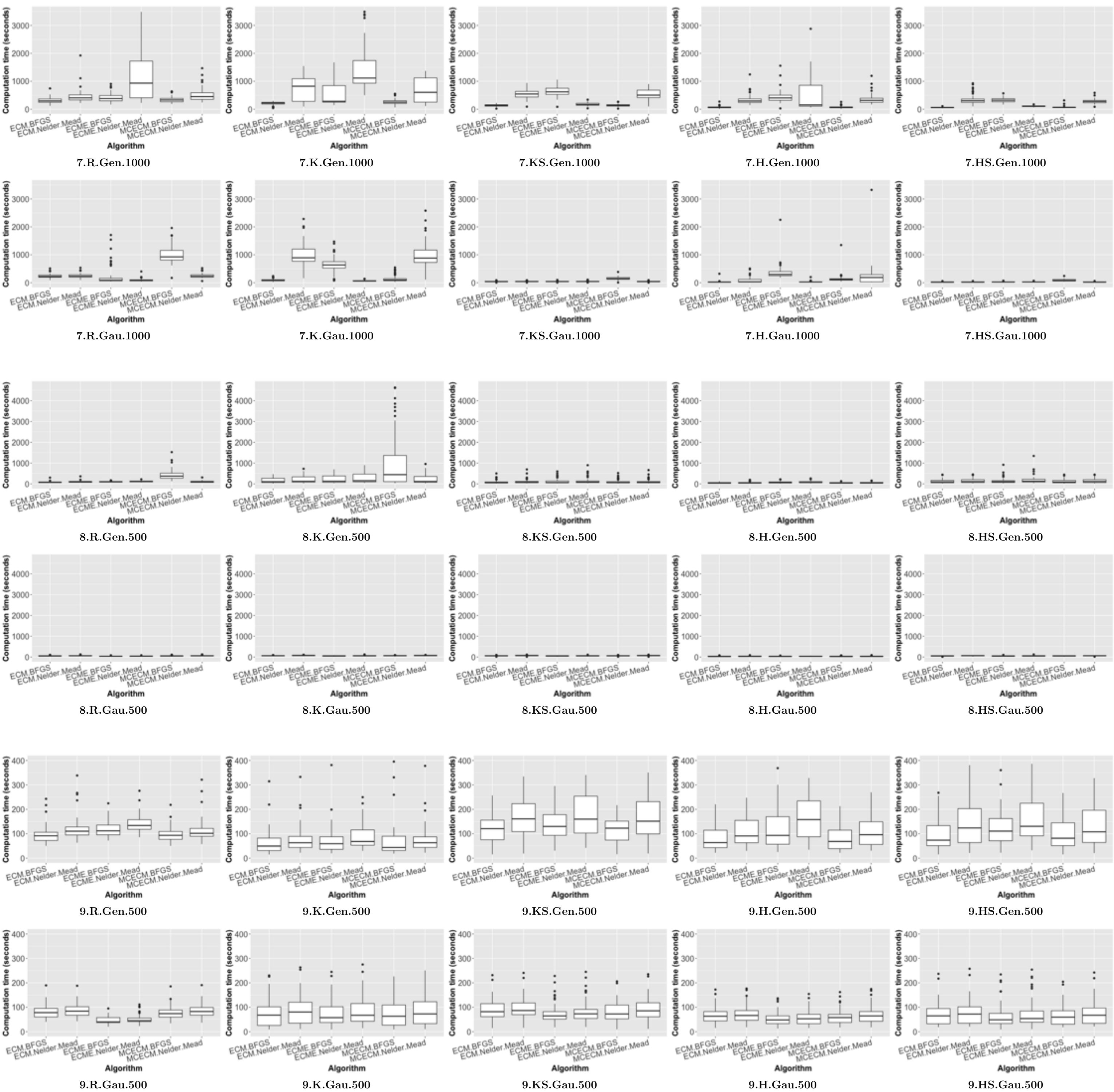


Figure B.15: Box plots of the **computation time** for the EM algorithms using different initialization strategies for 50 replications from settings 7, 8 and 9.

B.6 Ranking of the performance of EM algorithms and initialization strategies

Table B.2: The ranking of the performance of EM algorithms and initialization strategies for VCMM algorithm, where the algorithm with 1) higher classification rate, 2) lower normalised BIC and 3) shorter computation time is considered to be the better performance and the priority is given by (1) > (2) > (3).

	General VCMM						Gaussian VCMM					
Setting (Sample size)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
1 (500)	1	ECME.BFGS	Hierarchical	0.944	16.54	117	1	ECME.Nelder.Mead	Hierarchical	0.730	19.70	67
	2	MCECM.BFGS	Hierarchical	0.943	16.57	83	2	ECME.BFGS	Hierarchical	0.725	19.70	54
	3	ECM.BFGS	Hierarchical	0.942	16.57	84	3	ECM.BFGS	Hierarchical	0.724	19.74	67
	4	ECME.Nelder.Mead	Hierarchical	0.940	16.57	154	4	ECM.Nelder.Mead	Hierarchical	0.724	19.74	77
	5	MCECM.Nelder.Mead	Hierarchical	0.937	16.59	95	5	MCECM.BFGS	Hierarchical	0.723	19.74	65
	6	ECM.Nelder.Mead	Hierarchical	0.934	16.59	92	6	MCECM.Nelder.Mead	Hierarchical	0.723	19.74	77
	7	ECM.BFGS	Random	0.885	16.70	186	7	ECME.BFGS	Random	0.620	19.97	73
	8	MCECM.BFGS	Random	0.880	16.74	167	8	ECME.Nelder.Mead	Hierarchical scale	0.617	19.79	66
	9	ECME.Nelder.Mead	Random	0.879	16.74	269	9	ECME.BFGS	Hierarchical scale	0.615	19.81	55
	10	ECME.BFGS	Random	0.875	16.74	181	10	MCECM.BFGS	Hierarchical scale	0.613	19.82	73
	11	ECM.Nelder.Mead	Random	0.872	16.76	217	11	MCECM.Nelder.Mead	Hierarchical scale	0.613	19.82	83
	12	MCECM.Nelder.Mead	Random	0.868	16.79	210	12	ECM.BFGS	Hierarchical scale	0.612	19.82	72
	13	MCECM.BFGS	Hierarchical scale	0.649	17.70	76	13	ECM.Nelder.Mead	Hierarchical scale	0.611	19.82	80
	14	ECME.BFGS	Hierarchical scale	0.647	17.69	96	14	ECME.Nelder.Mead	Kmeans	0.606	19.93	85
	15	ECM.BFGS	Hierarchical scale	0.647	17.70	77	15	ECME.Nelder.Mead	Random	0.604	19.92	87
	16	ECM.Nelder.Mead	Hierarchical scale	0.637	17.74	97	16	ECME.BFGS	Kmeans	0.603	19.94	67
	17	MCECM.Nelder.Mead	Hierarchical scale	0.634	17.76	103	17	ECM.Nelder.Mead	Kmeans	0.599	19.95	110
	18	ECME.Nelder.Mead	Hierarchical scale	0.628	17.77	145	18	MCECM.Nelder.Mead	Kmeans	0.598	19.94	106
	19	MCECM.Nelder.Mead	Kmeans scale	0.611	17.98	93	19	ECM.BFGS	Kmeans	0.598	19.95	92
	20	MCECM.BFGS	Kmeans scale	0.610	17.98	71	20	MCECM.BFGS	Kmeans	0.597	19.94	91
	21	ECM.BFGS	Kmeans scale	0.610	17.99	73	21	ECM.BFGS	Random	0.594	20.00	134
	22	ECM.Nelder.Mead	Kmeans scale	0.610	17.99	95	22	ECME.BFGS	Kmeans scale	0.593	19.90	59
	23	ECME.Nelder.Mead	Kmeans	0.602	17.97	139	23	ECME.Nelder.Mead	Kmeans scale	0.593	19.90	69
	24	ECME.Nelder.Mead	Kmeans scale	0.601	18.01	110	24	ECM.Nelder.Mead	Random	0.592	20.03	144
	25	ECME.BFGS	Kmeans scale	0.600	18.02	69	25	MCECM.BFGS	Kmeans scale	0.590	19.90	91
	26	ECM.Nelder.Mead	Kmeans	0.597	17.98	124	26	MCECM.Nelder.Mead	Kmeans scale	0.590	19.90	99
	27	MCECM.Nelder.Mead	Kmeans	0.597	17.98	124	27	ECM.Nelder.Mead	Kmeans scale	0.589	19.90	102
	28	MCECM.BFGS	Kmeans	0.596	17.99	94	28	ECM.BFGS	Kmeans scale	0.589	19.91	89
	29	ECM.BFGS	Kmeans	0.596	17.99	97	29	MCECM.BFGS	Random	0.588	20.01	137
	30	ECME.BFGS	Kmeans	0.584	18.02	85	30	MCECM.Nelder.Mead	Random	0.587	20.01	147
1 (1000)	1	MCECM.BFGS	Random	0.935	16.48	285	1	ECM.BFGS	Hierarchical	0.736	19.68	112
	2	ECME.BFGS	Random	0.932	16.49	307	2	MCECM.BFGS	Hierarchical	0.736	19.68	115
	3	ECM.Nelder.Mead	Random	0.931	16.50	324	3	ECME.BFGS	Hierarchical	0.735	19.66	88
	4	ECME.Nelder.Mead	Random	0.930	16.49	370	4	ECME.Nelder.Mead	Hierarchical	0.735	19.66	96
	5	ECM.BFGS	Random	0.930	16.50	285	5	MCECM.Nelder.Mead	Hierarchical	0.735	19.66	126
	6	MCECM.BFGS	Hierarchical	0.927	16.48	112	6	ECM.Nelder.Mead	Hierarchical	0.735	19.66	131
	7	ECM.BFGS	Hierarchical	0.927	16.48	113	7	ECME.Nelder.Mead	Kmeans	0.605	19.88	124
	8	ECME.BFGS	Hierarchical	0.927	16.48	163	8	ECME.BFGS	Random	0.602	19.92	107
	9	MCECM.Nelder.Mead	Hierarchical	0.924	16.49	128	9	ECM.BFGS	Kmeans	0.601	19.88	164
	10	ECM.Nelder.Mead	Hierarchical	0.924	16.50	130	10	MCECM.Nelder.Mead	Kmeans	0.601	19.88	177
	11	ECME.Nelder.Mead	Hierarchical	0.923	16.51	183	11	ECM.Nelder.Mead	Kmeans	0.601	19.88	186
	12	MCECM.Nelder.Mead	Random	0.923	16.53	296	12	MCECM.BFGS	Kmeans	0.601	19.90	159
	13	ECM.Nelder.Mead	Kmeans	0.778	17.22	453	13	ECME.BFGS	Kmeans	0.601	19.92	105
	14	MCECM.Nelder.Mead	Kmeans	0.778	17.22	456	14	ECME.Nelder.Mead	Random	0.596	19.87	120
	15	MCECM.BFGS	Kmeans	0.767	17.28	341	15	ECME.Nelder.Mead	Kmeans scale	0.595	19.86	110
	16	ECM.BFGS	Kmeans	0.765	17.28	347	16	ECME.BFGS	Kmeans scale	0.592	19.89	98
	17	ECME.BFGS	Kmeans	0.709	17.51	298	17	ECME.Nelder.Mead	Hierarchical scale	0.590	19.83	106
	18	ECME.Nelder.Mead	Kmeans	0.709	17.53	416	18	ECM.Nelder.Mead	Kmeans scale	0.589	19.86	175
	19	MCECM.BFGS	Kmeans scale	0.644	17.72	160	19	ECME.BFGS	Hierarchical scale	0.588	19.86	95
	20	ECM.BFGS	Kmeans scale	0.644	17.72	164	20	MCECM.Nelder.Mead	Kmeans scale	0.588	19.86	163
	21	ECME.BFGS	Kmeans scale	0.640	17.76	165	21	MCECM.BFGS	Kmeans scale	0.587	19.88	153
	22	MCECM.BFGS	Hierarchical scale	0.634	17.70	158	22	ECM.Nelder.Mead	Hierarchical scale	0.586	19.83	159
	23	ECM.BFGS	Hierarchical scale	0.634	17.70	170	23	MCECM.Nelder.Mead	Hierarchical scale	0.586	19.84	153
	24	MCECM.Nelder.Mead	Hierarchical scale	0.631	17.71	190	24	ECM.BFGS	Kmeans scale	0.586	19.90	155
	25	ECM.Nelder.Mead	Hierarchical scale	0.630	17.71	201	25	MCECM.Nelder.Mead	Random	0.586	19.94	236
	26	ECM.Nelder.Mead	Kmeans scale	0.628	17.80	201	26	ECM.BFGS	Hierarchical scale	0.584	19.86	143
	27	ECME.Nelder.Mead	Kmeans scale	0.628	17.82	209	27	MCECM.BFGS	Random	0.584	19.96	217
	28	MCECM.Nelder.Mead	Kmeans scale	0.626	17.82	173	28	MCECM.BFGS	Hierarchical scale	0.583	19.88	138
	29	ECME.BFGS	Hierarchical scale	0.623	17.74	170	29	ECM.BFGS	Random	0.582	19.93	211
	30	ECME.Nelder.Mead	Hierarchical scale	0.619	17.78	218	30	ECM.Nelder.Mead	Random	0.582	19.93	235

		General VCMM						Gaussian VCMM				
Setting (Sample size)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
2 (500)	1	ECM.Nelder.Mead	Kmeans	0.894	13.15	70	1	ECM.BFGS	Kmeans	0.811	14.24	82
	2	MCECM.Nelder.Mead	Kmeans	0.894	13.15	73	2	MCECM.BFGS	Kmeans	0.811	14.24	83
	3	ECME.BFGS	Kmeans	0.889	13.12	88	3	MCECM.Nelder.Mead	Kmeans	0.811	14.24	97
	4	ECME.Nelder.Mead	Kmeans	0.889	13.13	102	4	ECM.Nelder.Mead	Kmeans	0.811	14.24	103
	5	ECM.BFGS	Kmeans	0.889	13.16	61	5	ECME.Nelder.Mead	Kmeans	0.808	14.23	99
	6	MCECM.BFGS	Kmeans	0.889	13.16	61	6	ECME.BFGS	Kmeans	0.807	14.23	84
	7	MCECM.BFGS	Hierarchical	0.871	13.16	72	7	ECM.BFGS	Hierarchical scale	0.777	14.28	49
	8	ECM.BFGS	Hierarchical	0.871	13.16	74	8	ECM.Nelder.Mead	Hierarchical scale	0.777	14.28	64
	9	ECME.BFGS	Hierarchical	0.871	13.16	109	9	MCECM.BFGS	Hierarchical scale	0.776	14.28	50
	10	ECME.Nelder.Mead	Hierarchical	0.870	13.14	129	10	MCECM.Nelder.Mead	Hierarchical scale	0.776	14.28	60
	11	ECM.BFGS	Hierarchical scale	0.870	13.17	79	11	ECME.BFGS	Hierarchical scale	0.775	14.28	57
	12	MCECM.BFGS	Hierarchical scale	0.870	13.17	79	12	ECME.Nelder.Mead	Hierarchical scale	0.775	14.28	64
	13	ECME.Nelder.Mead	Kmeans scale	0.870	13.18	177	13	ECM.BFGS	Hierarchical	0.765	14.31	56
	14	ECME.BFGS	Hierarchical scale	0.869	13.17	114	14	ECME.BFGS	Hierarchical	0.765	14.31	62
	15	MCECM.Nelder.Mead	Hierarchical	0.866	13.16	72	15	ECM.Nelder.Mead	Hierarchical	0.765	14.31	77
	16	ECM.Nelder.Mead	Hierarchical	0.865	13.17	71	16	MCECM.BFGS	Hierarchical	0.764	14.31	57
	17	MCECM.Nelder.Mead	Hierarchical scale	0.863	13.18	83	17	MCECM.Nelder.Mead	Hierarchical	0.764	14.31	68
	18	ECM.Nelder.Mead	Hierarchical scale	0.863	13.18	90	18	ECME.Nelder.Mead	Hierarchical	0.764	14.31	71
	19	MCECM.BFGS	Kmeans scale	0.863	13.24	88	19	ECME.BFGS	Kmeans scale	0.621	14.50	89
	20	ECM.BFGS	Kmeans scale	0.863	13.24	89	20	ECME.Nelder.Mead	Kmeans scale	0.621	14.50	103
	21	ECME.BFGS	Kmeans scale	0.863	13.24	144	21	ECM.BFGS	Kmeans scale	0.621	14.52	74
	22	MCECM.Nelder.Mead	Kmeans scale	0.862	13.24	107	22	ECM.Nelder.Mead	Kmeans scale	0.621	14.52	94
	23	ECM.Nelder.Mead	Kmeans scale	0.861	13.24	110	23	MCECM.BFGS	Kmeans scale	0.617	14.52	78
	24	ECME.Nelder.Mead	Hierarchical scale	0.856	13.19	137	24	MCECM.Nelder.Mead	Kmeans scale	0.616	14.52	95
	25	ECME.Nelder.Mead	Random	0.659	13.78	180	25	MCECM.Nelder.Mead	Random	0.546	14.65	146
	26	ECME.BFGS	Random	0.645	13.88	98	26	MCECM.BFGS	Random	0.544	14.66	113
	27	ECM.Nelder.Mead	Random	0.627	13.96	157	27	ECM.BFGS	Random	0.532	14.67	108
	28	ECM.BFGS	Random	0.625	13.97	88	28	ECM.Nelder.Mead	Random	0.532	14.67	143
	29	MCECM.Nelder.Mead	Random	0.621	13.98	130	29	ECME.BFGS	Random	0.519	14.69	129
	30	MCECM.BFGS	Random	0.619	13.98	85	30	ECME.Nelder.Mead	Random	0.519	14.69	148
3 (500)	1	MCECM.Nelder.Mead	Random	0.835	12.08	104	1	MCECM.BFGS	Random	0.865	12.01	53
	2	MCECM.BFGS	Random	0.835	12.08	111	2	MCECM.Nelder.Mead	Random	0.865	12.01	55
	3	ECM.Nelder.Mead	Random	0.821	12.10	114	3	ECM.BFGS	Random	0.857	12.01	59
	4	ECM.BFGS	Random	0.821	12.10	121	4	ECM.Nelder.Mead	Random	0.857	12.01	68
	5	ECME.BFGS	Random	0.799	12.12	121	5	MCECM.BFGS	Hierarchical	0.824	12.06	55
	6	ECME.Nelder.Mead	Random	0.792	12.13	118	6	MCECM.Nelder.Mead	Hierarchical	0.824	12.06	57
	7	ECME.BFGS	Hierarchical	0.680	12.29	82	7	ECM.BFGS	Hierarchical	0.824	12.06	58
	8	ECME.Nelder.Mead	Hierarchical	0.678	12.29	89	8	ECM.Nelder.Mead	Hierarchical	0.824	12.06	63
	9	ECM.Nelder.Mead	Hierarchical	0.678	12.30	78	9	ECME.Nelder.Mead	Hierarchical	0.824	12.06	70
	10	MCECM.Nelder.Mead	Hierarchical	0.677	12.30	75	10	ECME.BFGS	Hierarchical	0.824	12.06	72
	11	MCECM.BFGS	Hierarchical	0.677	12.30	80	11	ECME.BFGS	Random	0.821	12.06	78
	12	ECM.BFGS	Hierarchical	0.677	12.30	82	12	ECME.Nelder.Mead	Random	0.820	12.06	80
	13	MCECM.Nelder.Mead	Kmeans	0.575	12.39	88	13	MCECM.BFGS	Hierarchical scale	0.760	12.14	67
	14	ECM.Nelder.Mead	Kmeans	0.575	12.39	93	14	MCECM.Nelder.Mead	Hierarchical scale	0.760	12.14	75
	15	ECME.Nelder.Mead	Kmeans	0.573	12.38	78	15	ECM.BFGS	Hierarchical scale	0.759	12.14	70
	16	ECME.BFGS	Kmeans	0.572	12.38	77	16	ECM.Nelder.Mead	Hierarchical scale	0.759	12.14	82
	17	MCECM.BFGS	Kmeans	0.571	12.39	93	17	MCECM.BFGS	Kmeans	0.756	12.14	70
	18	ECM.BFGS	Kmeans	0.570	12.40	94	18	ECM.BFGS	Kmeans	0.756	12.14	73
	19	MCECM.Nelder.Mead	Hierarchical scale	0.563	12.37	67	19	MCECM.Nelder.Mead	Kmeans	0.756	12.14	73
	20	MCECM.BFGS	Hierarchical scale	0.560	12.38	64	20	ECM.Nelder.Mead	Kmeans	0.756	12.14	79
	21	ECM.Nelder.Mead	Hierarchical scale	0.560	12.38	66	21	ECME.BFGS	Hierarchical scale	0.756	12.14	83
	22	ECM.BFGS	Hierarchical scale	0.558	12.38	62	22	ECME.Nelder.Mead	Hierarchical scale	0.756	12.14	93
	23	ECME.BFGS	Hierarchical scale	0.557	12.38	62	23	ECME.BFGS	Kmeans scale	0.747	12.15	96
	24	ECME.Nelder.Mead	Hierarchical scale	0.557	12.38	66	24	ECME.Nelder.Mead	Kmeans scale	0.747	12.15	108
	25	MCECM.Nelder.Mead	Kmeans scale	0.537	12.41	64	25	ECME.Nelder.Mead	Kmeans	0.746	12.15	83
	26	ECM.Nelder.Mead	Kmeans scale	0.537	12.41	67	26	ECME.BFGS	Kmeans	0.746	12.15	83
	27	ECM.BFGS	Kmeans scale	0.537	12.41	67	27	ECM.BFGS	Kmeans scale	0.743	12.16	85
	28	MCECM.BFGS	Kmeans scale	0.537	12.41	68	28	ECM.Nelder.Mead	Kmeans scale	0.743	12.16	98
	29	ECME.BFGS	Kmeans scale	0.536	12.40	73	29	MCECM.BFGS	Kmeans scale	0.741	12.16	81
	30	ECME.Nelder.Mead	Kmeans scale	0.535	12.40	70	30	MCECM.Nelder.Mead	Kmeans scale	0.740	12.16	90

Setting (Sample size)	General VCMM						Gaussian VCMM					
	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
3 (1000)	1	MCECM.Nelder.Mead	Random	0.851	12.01	171	1	MCECM.BFGS	Random	0.877	11.95	76
	2	MCECM.BFGS	Random	0.851	12.01	175	2	MCECM.Nelder.Mead	Random	0.877	11.95	77
	3	ECM.Nelder.Mead	Random	0.850	12.01	191	3	ECM.Nelder.Mead	Random	0.877	11.95	87
	4	ECM.BFGS	Random	0.850	12.01	194	4	ECM.BFGS	Random	0.876	11.95	87
	5	ECME.Nelder.Mead	Random	0.816	12.04	185	5	MCECM.Nelder.Mead	Hierarchical	0.844	12.00	95
	6	ECME.BFGS	Random	0.816	12.04	192	6	MCECM.BFGS	Hierarchical	0.844	12.00	95
	7	ECME.Nelder.Mead	Hierarchical	0.714	12.17	147	7	ECM.Nelder.Mead	Hierarchical	0.844	12.00	99
	8	ECME.BFGS	Hierarchical	0.714	12.17	148	8	ECM.BFGS	Hierarchical	0.844	12.00	99
	9	MCECM.Nelder.Mead	Hierarchical	0.713	12.18	146	9	ECME.BFGS	Hierarchical	0.844	12.00	100
	10	MCECM.BFGS	Hierarchical	0.713	12.18	148	10	ECME.Nelder.Mead	Hierarchical	0.844	12.00	101
	11	ECM.Nelder.Mead	Hierarchical	0.713	12.18	154	11	ECME.BFGS	Random	0.839	11.99	76
	12	ECM.BFGS	Hierarchical	0.713	12.18	155	12	ECME.Nelder.Mead	Random	0.839	11.99	77
	13	ECME.Nelder.Mead	Hierarchical scale	0.593	12.26	150	13	ECME.Nelder.Mead	Hierarchical scale	0.826	12.02	123
	14	ECME.BFGS	Hierarchical scale	0.593	12.26	157	14	MCECM.Nelder.Mead	Hierarchical scale	0.825	12.02	123
	15	MCECM.Nelder.Mead	Hierarchical scale	0.588	12.27	152	15	MCECM.BFGS	Hierarchical scale	0.825	12.02	125
	16	MCECM.BFGS	Hierarchical scale	0.587	12.27	150	16	ECME.BFGS	Hierarchical scale	0.825	12.02	127
	17	ECM.BFGS	Hierarchical scale	0.587	12.27	151	17	ECM.Nelder.Mead	Hierarchical scale	0.825	12.02	132
	18	ECM.Nelder.Mead	Hierarchical scale	0.587	12.28	153	18	ECM.BFGS	Hierarchical scale	0.825	12.02	132
	19	MCECM.Nelder.Mead	Kmeans	0.559	12.32	155	19	MCECM.Nelder.Mead	Kmeans	0.787	12.06	137
	20	ECM.BFGS	Kmeans	0.559	12.32	161	20	MCECM.BFGS	Kmeans	0.787	12.06	140
	21	MCECM.BFGS	Kmeans	0.559	12.32	163	21	ECM.Nelder.Mead	Kmeans	0.787	12.06	144
	22	ECM.Nelder.Mead	Kmeans	0.558	12.32	153	22	ECM.BFGS	Kmeans	0.787	12.06	145
	23	ECME.Nelder.Mead	Kmeans	0.550	12.31	131	23	MCECM.Nelder.Mead	Kmeans scale	0.776	12.07	148
	24	ECME.BFGS	Kmeans	0.550	12.31	142	24	MCECM.BFGS	Kmeans scale	0.776	12.07	150
	25	ECME.Nelder.Mead	Kmeans scale	0.530	12.30	119	25	ECM.Nelder.Mead	Kmeans scale	0.772	12.07	152
	26	ECME.BFGS	Kmeans scale	0.529	12.30	122	26	ECM.BFGS	Kmeans scale	0.772	12.07	154
	27	MCECM.Nelder.Mead	Kmeans scale	0.528	12.31	111	27	ECME.BFGS	Kmeans	0.770	12.08	135
	28	ECM.BFGS	Kmeans scale	0.528	12.31	117	28	ECME.Nelder.Mead	Kmeans	0.770	12.08	137
	29	ECM.Nelder.Mead	Kmeans scale	0.528	12.32	111	29	ECME.Nelder.Mead	Kmeans scale	0.768	12.08	132
	30	MCECM.BFGS	Kmeans scale	0.528	12.32	116	30	ECME.BFGS	Kmeans scale	0.768	12.08	137
4 (500)	1	MCECM.BFGS	Kmeans	0.826	10.79	42	1	ECM.Nelder.Mead	Kmeans	0.833	10.78	47
	2	MCECM.Nelder.Mead	Kmeans	0.826	10.79	44	2	MCECM.Nelder.Mead	Kmeans	0.831	10.78	42
	3	ECM.BFGS	Kmeans	0.826	10.79	45	3	MCECM.Nelder.Mead	Kmeans scale	0.829	10.77	40
	4	ECM.Nelder.Mead	Kmeans	0.826	10.79	50	4	ECM.Nelder.Mead	Kmeans scale	0.829	10.77	51
	5	ECME.BFGS	Kmeans	0.825	10.79	46	5	ECM.BFGS	Kmeans	0.827	10.77	43
	6	ECME.Nelder.Mead	Kmeans	0.824	10.79	55	6	MCECM.BFGS	Kmeans	0.826	10.77	41
	7	MCECM.BFGS	Kmeans scale	0.805	10.78	42	7	ECME.BFGS	Kmeans	0.826	10.77	48
	8	MCECM.Nelder.Mead	Kmeans scale	0.805	10.78	44	8	ECME.Nelder.Mead	Kmeans	0.826	10.77	60
	9	ECM.Nelder.Mead	Kmeans scale	0.805	10.78	50	9	MCECM.BFGS	Kmeans scale	0.825	10.77	41
	10	ECM.BFGS	Kmeans scale	0.804	10.79	42	10	ECM.BFGS	Kmeans scale	0.825	10.77	47
	11	ECME.BFGS	Kmeans scale	0.804	10.79	45	11	ECME.BFGS	Kmeans scale	0.825	10.77	51
	12	ECME.Nelder.Mead	Kmeans scale	0.803	10.79	58	12	ECME.Nelder.Mead	Kmeans scale	0.824	10.77	51
	13	ECM.BFGS	Hierarchical	0.797	10.81	42	13	MCECM.Nelder.Mead	Hierarchical	0.818	10.78	59
	14	ECME.BFGS	Hierarchical	0.796	10.80	47	14	ECM.Nelder.Mead	Hierarchical	0.818	10.78	64
	15	MCECM.Nelder.Mead	Hierarchical	0.796	10.81	42	15	ECM.BFGS	Hierarchical	0.814	10.78	57
	16	MCECM.BFGS	Hierarchical	0.796	10.81	42	16	ECME.BFGS	Hierarchical	0.812	10.78	64
	17	ECM.Nelder.Mead	Hierarchical	0.796	10.81	50	17	MCECM.BFGS	Hierarchical	0.811	10.78	59
	18	ECME.Nelder.Mead	Hierarchical	0.796	10.81	58	18	ECME.Nelder.Mead	Hierarchical	0.811	10.78	76
	19	ECM.BFGS	Hierarchical scale	0.774	10.80	49	19	ECM.Nelder.Mead	Hierarchical scale	0.807	10.78	65
	20	MCECM.BFGS	Hierarchical scale	0.774	10.80	50	20	MCECM.BFGS	Hierarchical scale	0.803	10.78	56
	21	ECME.BFGS	Hierarchical scale	0.774	10.80	54	21	ECM.BFGS	Hierarchical scale	0.803	10.78	59
	22	MCECM.Nelder.Mead	Hierarchical scale	0.774	10.80	54	22	MCECM.Nelder.Mead	Hierarchical scale	0.803	10.78	65
	23	ECM.Nelder.Mead	Hierarchical scale	0.774	10.80	56	23	ECME.Nelder.Mead	Hierarchical scale	0.794	10.78	74
	24	ECME.Nelder.Mead	Hierarchical scale	0.772	10.80	66	24	ECME.BFGS	Hierarchical scale	0.793	10.78	74
	25	ECME.BFGS	Random	0.541	10.97	166	25	ECME.Nelder.Mead	Random	0.590	10.84	142
	26	ECME.Nelder.Mead	Random	0.538	10.97	177	26	ECME.BFGS	Random	0.590	10.84	146
	27	MCECM.BFGS	Random	0.535	10.97	155	27	ECM.BFGS	Random	0.578	10.84	116
	28	ECM.BFGS	Random	0.535	10.97	158	28	ECM.Nelder.Mead	Random	0.578	10.84	127
	29	ECM.Nelder.Mead	Random	0.535	10.97	161	29	MCECM.BFGS	Random	0.568	10.84	115
	30	MCECM.Nelder.Mead	Random	0.535	10.97	166	30	MCECM.Nelder.Mead	Random	0.568	10.84	124

	General VCMM						Gaussian VCMM					
Setting (Sample size)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
	1	ECM.BFGS	Kmeans	0.756	12.08	86	1	ECME.Nelder.Mead	Hierarchical scale	0.523	12.13	62
	2	MCECM.Nelder.Mead	Kmeans	0.756	12.08	99	2	ECME.BFGS	Hierarchical scale	0.523	12.13	72
	3	MCECM.BFGS	Kmeans	0.755	12.08	88	3	MCECM.Nelder.Mead	Hierarchical	0.522	12.10	50
	4	ECM.Nelder.Mead	Kmeans	0.755	12.08	103	4	ECM.BFGS	Hierarchical	0.522	12.10	53
	5	ECME.BFGS	Kmeans	0.753	12.08	100	5	ECM.Nelder.Mead	Hierarchical	0.522	12.10	54
	6	ECME.Nelder.Mead	Kmeans	0.753	12.08	113	6	MCECM.BFGS	Hierarchical	0.522	12.10	54
	7	MCECM.Nelder.Mead	Kmeans scale	0.747	12.08	87	7	ECME.Nelder.Mead	Hierarchical	0.522	12.11	60
	8	ECM.BFGS	Kmeans scale	0.746	12.08	77	8	ECME.BFGS	Hierarchical	0.522	12.11	70
	9	MCECM.BFGS	Kmeans scale	0.746	12.08	79	9	MCECM.Nelder.Mead	Kmeans scale	0.522	12.12	51
	10	ECM.Nelder.Mead	Kmeans scale	0.746	12.08	92	10	MCECM.BFGS	Kmeans scale	0.522	12.12	51
	11	ECME.BFGS	Kmeans scale	0.746	12.08	97	11	ECM.BFGS	Kmeans scale	0.522	12.12	52
	12	ECME.Nelder.Mead	Kmeans scale	0.746	12.08	111	12	ECM.BFGS	Kmeans	0.522	12.12	54
	13	MCECM.Nelder.Mead	Hierarchical scale	0.708	12.08	113	13	ECM.Nelder.Mead	Kmeans scale	0.522	12.12	54
	14	ECME.BFGS	Hierarchical scale	0.708	12.08	120	14	ECME.Nelder.Mead	Kmeans scale	0.522	12.12	58
5 (500)	15	ECM.BFGS	Hierarchical scale	0.707	12.08	105	15	ECME.Nelder.Mead	Kmeans	0.522	12.12	59
	16	MCECM.BFGS	Hierarchical scale	0.707	12.09	103	16	ECME.BFGS	Kmeans scale	0.522	12.12	62
	17	ECM.Nelder.Mead	Hierarchical scale	0.705	12.08	112	17	ECME.BFGS	Kmeans	0.522	12.12	68
	18	ECME.Nelder.Mead	Hierarchical scale	0.705	12.09	127	18	MCECM.Nelder.Mead	Kmeans	0.521	12.12	52
	19	ECME.BFGS	Hierarchical	0.679	12.10	92	19	ECM.Nelder.Mead	Kmeans	0.521	12.12	55
	20	ECME.Nelder.Mead	Hierarchical	0.679	12.10	108	20	ECME.Nelder.Mead	Random	0.521	12.12	56
	21	ECM.BFGS	Hierarchical	0.678	12.10	86	21	MCECM.BFGS	Kmeans	0.521	12.12	56
	22	MCECM.BFGS	Hierarchical	0.677	12.10	85	22	MCECM.Nelder.Mead	Hierarchical scale	0.521	12.12	56
	23	MCECM.Nelder.Mead	Hierarchical	0.677	12.10	98	23	MCECM.BFGS	Hierarchical scale	0.521	12.12	56
	24	ECM.Nelder.Mead	Hierarchical	0.677	12.10	100	24	ECM.BFGS	Hierarchical scale	0.521	12.12	59
	25	ECME.BFGS	Random	0.563	12.17	360	25	ECM.Nelder.Mead	Hierarchical scale	0.521	12.12	60
	26	ECME.Nelder.Mead	Random	0.560	12.17	436	26	ECME.BFGS	Random	0.521	12.12	66
	27	ECM.BFGS	Random	0.557	12.17	297	27	ECM.Nelder.Mead	Random	0.520	12.12	43
	28	ECM.Nelder.Mead	Random	0.556	12.16	360	28	MCECM.Nelder.Mead	Random	0.520	12.13	38
	29	MCECM.Nelder.Mead	Random	0.545	12.17	324	29	MCECM.BFGS	Random	0.520	12.13	38
	30	MCECM.BFGS	Random	0.542	12.17	272	30	ECM.BFGS	Random	0.520	12.13	42
	1	MCECM.Nelder.Mead	Kmeans	0.757	12.02	155	1	MCECM.BFGS	Kmeans	0.521	12.12	79
	2	MCECM.BFGS	Kmeans	0.756	12.01	133	2	MCECM.BFGS	Kmeans scale	0.521	12.12	79
	3	ECM.BFGS	Kmeans	0.756	12.02	138	3	MCECM.Nelder.Mead	Kmeans	0.521	12.12	83
	4	ECM.Nelder.Mead	Kmeans	0.756	12.02	159	4	ECME.Nelder.Mead	Kmeans	0.521	12.12	84
	5	ECME.BFGS	Kmeans	0.754	12.01	143	5	ECME.Nelder.Mead	Kmeans scale	0.521	12.12	84
	6	ECME.Nelder.Mead	Kmeans	0.754	12.01	166	6	MCECM.Nelder.Mead	Kmeans scale	0.521	12.12	84
	7	MCECM.BFGS	Kmeans scale	0.744	12.02	118	7	ECME.BFGS	Kmeans scale	0.521	12.12	89
	8	ECM.BFGS	Kmeans scale	0.744	12.02	123	8	ECME.BFGS	Kmeans	0.521	12.12	95
	9	ECM.Nelder.Mead	Kmeans scale	0.744	12.02	138	9	MCECM.BFGS	Random	0.520	12.12	64
	10	MCECM.Nelder.Mead	Kmeans scale	0.744	12.02	139	10	MCECM.Nelder.Mead	Random	0.520	12.12	69
	11	ECME.BFGS	Kmeans scale	0.742	12.01	144	11	ECM.BFGS	Random	0.520	12.12	72
	12	ECME.Nelder.Mead	Kmeans scale	0.742	12.02	162	12	ECM.Nelder.Mead	Random	0.520	12.12	73
	13	ECME.BFGS	Hierarchical scale	0.703	12.01	198	13	MCECM.BFGS	Hierarchical	0.520	12.12	77
	14	ECME.Nelder.Mead	Hierarchical scale	0.702	12.01	220	14	MCECM.Nelder.Mead	Hierarchical	0.520	12.12	80
5 (1000)	15	ECM.Nelder.Mead	Hierarchical scale	0.701	12.01	217	15	ECME.Nelder.Mead	Random	0.520	12.12	85
	16	MCECM.Nelder.Mead	Hierarchical scale	0.701	12.01	217	16	ECM.BFGS	Hierarchical	0.520	12.12	85
	17	MCECM.BFGS	Hierarchical scale	0.700	12.01	197	17	ECM.Nelder.Mead	Kmeans	0.520	12.12	86
	18	ECM.BFGS	Hierarchical scale	0.700	12.01	206	18	ECM.Nelder.Mead	Hierarchical	0.520	12.12	86
	19	ECME.Nelder.Mead	Hierarchical	0.664	12.05	215	19	ECM.BFGS	Kmeans scale	0.520	12.12	86
	20	ECM.Nelder.Mead	Hierarchical	0.664	12.05	221	20	ECM.BFGS	Kmeans	0.520	12.12	87
	21	ECME.BFGS	Hierarchical	0.663	12.04	188	21	ECME.Nelder.Mead	Hierarchical	0.520	12.12	87
	22	MCECM.BFGS	Hierarchical	0.662	12.05	178	22	ECM.Nelder.Mead	Kmeans scale	0.520	12.12	87
	23	ECM.BFGS	Hierarchical	0.662	12.05	187	23	MCECM.BFGS	Hierarchical scale	0.520	12.12	89
	24	MCECM.Nelder.Mead	Hierarchical	0.662	12.05	206	24	MCECM.Nelder.Mead	Hierarchical scale	0.520	12.12	94
	25	ECME.BFGS	Random	0.626	12.08	636	25	ECME.BFGS	Random	0.520	12.12	99
	26	ECME.Nelder.Mead	Random	0.624	12.08	751	26	ECME.BFGS	Hierarchical	0.520	12.12	99
	27	ECM.BFGS	Random	0.602	12.09	615	27	ECM.BFGS	Hierarchical scale	0.520	12.12	99
	28	ECM.Nelder.Mead	Random	0.601	12.09	748	28	ECM.Nelder.Mead	Hierarchical scale	0.520	12.12	100
	29	MCECM.Nelder.Mead	Random	0.574	12.10	689	29	ECME.Nelder.Mead	Hierarchical scale	0.520	12.12	102
	30	MCECM.BFGS	Random	0.573	12.10	523	30	ECME.BFGS	Hierarchical scale	0.520	12.12	111

	General VCMM						Gaussian VCMM					
Setting (Sample size)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
	1	ECM.BFGS	Kmeans	0.987	13.32	40	1	ECM.BFGS	Kmeans	0.970	14.35	21
	2	MCECM.BFGS	Kmeans	0.987	13.32	41	2	ECM.Nelder.Mead	Kmeans	0.970	14.35	22
	3	ECM.Nelder.Mead	Kmeans	0.987	13.32	49	3	ECME.BFGS	Kmeans	0.969	14.34	28
	4	MCECM.Nelder.Mead	Kmeans	0.987	13.32	50	4	MCECM.BFGS	Kmeans	0.969	14.35	21
	5	ECME.BFGS	Kmeans	0.987	13.32	59	5	MCECM.Nelder.Mead	Kmeans	0.969	14.35	23
	6	ECME.Nelder.Mead	Kmeans	0.987	13.32	73	6	ECME.Nelder.Mead	Kmeans	0.969	14.35	31
	7	ECME.Nelder.Mead	Hierarchical scale	0.961	13.38	119	7	MCECM.Nelder.Mead	Kmeans scale	0.927	14.36	28
	8	ECM.BFGS	Hierarchical scale	0.960	13.38	84	8	MCECM.BFGS	Kmeans scale	0.927	14.37	23
	9	MCECM.BFGS	Hierarchical scale	0.960	13.38	87	9	ECM.BFGS	Kmeans scale	0.927	14.37	25
	10	ECME.BFGS	Hierarchical scale	0.960	13.38	99	10	ECM.Nelder.Mead	Kmeans scale	0.927	14.37	28
	11	ECM.Nelder.Mead	Hierarchical scale	0.960	13.38	101	11	ECME.Nelder.Mead	Kmeans scale	0.926	14.36	37
	12	MCECM.Nelder.Mead	Hierarchical scale	0.960	13.38	101	12	ECME.BFGS	Kmeans scale	0.925	14.35	38
	13	ECM.BFGS	Kmeans scale	0.958	13.38	42	13	MCECM.BFGS	Hierarchical	0.856	14.36	25
	14	MCECM.BFGS	Kmeans scale	0.958	13.38	43	14	ECM.BFGS	Hierarchical	0.856	14.36	26
6 (500)	15	ECM.Nelder.Mead	Kmeans scale	0.958	13.38	50	15	ECM.Nelder.Mead	Hierarchical	0.856	14.36	28
	16	MCECM.Nelder.Mead	Kmeans scale	0.958	13.38	51	16	MCECM.Nelder.Mead	Hierarchical	0.855	14.36	27
	17	ECME.BFGS	Kmeans scale	0.958	13.38	59	17	ECME.BFGS	Hierarchical	0.855	14.36	32
	18	ECME.Nelder.Mead	Kmeans scale	0.958	13.38	71	18	ECME.Nelder.Mead	Hierarchical	0.855	14.36	34
	19	MCECM.BFGS	Hierarchical	0.939	13.41	53	19	ECM.BFGS	Hierarchical scale	0.835	14.42	26
	20	ECM.BFGS	Hierarchical	0.939	13.41	54	20	ECM.Nelder.Mead	Hierarchical scale	0.835	14.42	28
	21	ECM.Nelder.Mead	Hierarchical	0.939	13.41	63	21	MCECM.BFGS	Hierarchical scale	0.833	14.42	27
	22	MCECM.Nelder.Mead	Hierarchical	0.939	13.41	65	22	MCECM.Nelder.Mead	Hierarchical scale	0.833	14.42	28
	23	ECME.BFGS	Hierarchical	0.939	13.41	72	23	ECME.BFGS	Hierarchical scale	0.832	14.42	34
	24	ECME.Nelder.Mead	Hierarchical	0.939	13.41	93	24	ECME.Nelder.Mead	Hierarchical scale	0.832	14.42	35
	25	ECM.BFGS	Random	0.918	13.47	204	25	MCECM.BFGS	Random	0.615	14.37	50
	26	ECME.Nelder.Mead	Random	0.918	13.47	215	26	MCECM.Nelder.Mead	Random	0.615	14.37	51
	27	MCECM.BFGS	Random	0.914	13.47	200	27	ECM.BFGS	Random	0.599	14.36	55
	28	MCECM.Nelder.Mead	Random	0.912	13.48	228	28	ECM.Nelder.Mead	Random	0.599	14.36	57
	29	ECM.Nelder.Mead	Random	0.905	13.49	233	29	ECME.Nelder.Mead	Random	0.556	14.38	77
	30	ECME.BFGS	Random	0.897	13.53	187	30	ECME.BFGS	Random	0.546	14.38	73
	1	ECM.BFGS	Kmeans	0.988	13.26	65	1	MCECM.Nelder.Mead	Kmeans	0.971	14.44	30
	2	MCECM.BFGS	Kmeans	0.988	13.26	79	2	ECM.Nelder.Mead	Kmeans	0.971	14.44	32
	3	ECM.Nelder.Mead	Kmeans	0.988	13.26	92	3	ECME.Nelder.Mead	Kmeans	0.971	14.44	40
	4	MCECM.Nelder.Mead	Kmeans	0.988	13.26	94	4	ECME.BFGS	Kmeans	0.970	14.45	43
	5	ECME.BFGS	Kmeans	0.988	13.26	119	5	ECM.BFGS	Kmeans	0.970	14.46	30
	6	ECME.Nelder.Mead	Kmeans	0.988	13.26	137	6	MCECM.BFGS	Kmeans	0.970	14.46	38
	7	ECME.BFGS	Hierarchical scale	0.975	13.30	177	7	ECME.Nelder.Mead	Kmeans scale	0.931	14.46	46
	8	ECM.Nelder.Mead	Kmeans scale	0.974	13.30	112	8	ECME.BFGS	Kmeans scale	0.929	14.44	46
	9	ECME.BFGS	Kmeans scale	0.974	13.30	137	9	MCECM.Nelder.Mead	Kmeans scale	0.929	14.46	33
	10	ECM.BFGS	Hierarchical scale	0.974	13.30	140	10	ECM.Nelder.Mead	Kmeans scale	0.929	14.46	34
	11	MCECM.BFGS	Hierarchical scale	0.974	13.30	152	11	MCECM.BFGS	Kmeans scale	0.927	14.44	33
	12	ECM.Nelder.Mead	Hierarchical scale	0.974	13.30	163	12	ECM.BFGS	Kmeans scale	0.927	14.44	35
	13	MCECM.Nelder.Mead	Hierarchical scale	0.974	13.30	168	13	MCECM.Nelder.Mead	Hierarchical	0.882	14.47	42
	14	ECME.Nelder.Mead	Hierarchical scale	0.974	13.30	194	14	MCECM.BFGS	Hierarchical	0.881	14.47	44
6 (1000)	15	ECM.BFGS	Hierarchical	0.973	13.29	112	15	ECME.Nelder.Mead	Hierarchical	0.881	14.47	46
	16	ECM.Nelder.Mead	Hierarchical	0.973	13.29	135	16	ECME.BFGS	Hierarchical	0.880	14.47	49
	17	MCECM.Nelder.Mead	Hierarchical	0.973	13.29	141	17	ECM.BFGS	Hierarchical	0.876	14.47	44
	18	MCECM.BFGS	Hierarchical	0.973	13.29	143	18	ECM.Nelder.Mead	Hierarchical	0.876	14.47	47
	19	ECME.BFGS	Hierarchical	0.973	13.29	175	19	ECM.Nelder.Mead	Hierarchical scale	0.840	14.50	43
	20	ECME.Nelder.Mead	Hierarchical	0.973	13.29	182	20	ECM.BFGS	Hierarchical scale	0.840	14.50	43
	21	ECME.Nelder.Mead	Kmeans scale	0.973	13.31	161	21	MCECM.BFGS	Hierarchical scale	0.839	14.50	42
	22	ECM.BFGS	Kmeans scale	0.968	13.31	85	22	MCECM.Nelder.Mead	Hierarchical scale	0.839	14.50	45
	23	MCECM.BFGS	Kmeans scale	0.968	13.31	101	23	ECME.Nelder.Mead	Hierarchical scale	0.837	14.50	46
	24	MCECM.Nelder.Mead	Kmeans scale	0.968	13.31	108	24	ECME.BFGS	Hierarchical scale	0.837	14.50	49
	25	ECM.BFGS	Random	0.959	13.33	364	25	MCECM.Nelder.Mead	Random	0.555	14.41	70
	26	ECME.Nelder.Mead	Random	0.955	13.34	345	26	MCECM.BFGS	Random	0.554	14.41	73
	27	MCECM.BFGS	Random	0.950	13.35	361	27	ECM.BFGS	Random	0.554	14.41	75
	28	ECME.BFGS	Random	0.949	13.36	294	28	ECM.Nelder.Mead	Random	0.554	14.41	79
	29	ECM.Nelder.Mead	Random	0.948	13.35	495	29	ECME.Nelder.Mead	Random	0.528	14.41	87
	30	MCECM.Nelder.Mead	Random	0.945	13.36	445	30	ECME.BFGS	Random	0.528	14.41	96

Setting	General VCMM						Gaussian VCMM					
Setting (Sample size)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
7 (500)	1	ECM.BFGS	Kmeans scale	0.980	13.92	69	1	ECM.BFGS	Hierarchical scale	0.954	14.70	14
	2	MCECM.BFGS	Kmeans scale	0.980	13.92	71	2	MCECM.BFGS	Hierarchical scale	0.954	14.70	14
	3	MCECM.Nelder.Mead	Kmeans scale	0.980	13.92	82	3	MCECM.Nelder.Mead	Hierarchical scale	0.954	14.70	16
	4	ECM.Nelder.Mead	Kmeans scale	0.980	13.92	85	4	ECM.BFGS	Hierarchical	0.954	14.70	17
	5	ECME.BFGS	Kmeans scale	0.980	13.92	94	5	MCECM.BFGS	Hierarchical	0.954	14.70	17
	6	ECME.Nelder.Mead	Kmeans scale	0.980	13.92	119	6	ECM.Nelder.Mead	Hierarchical scale	0.954	14.70	17
	7	MCECM.BFGS	Hierarchical	0.975	13.92	38	7	ECM.Nelder.Mead	Hierarchical	0.954	14.70	18
	8	ECME.BFGS	Hierarchical	0.975	13.92	57	8	MCECM.Nelder.Mead	Hierarchical	0.954	14.70	18
	9	ECME.Nelder.Mead	Hierarchical	0.975	13.92	72	9	ECME.Nelder.Mead	Hierarchical scale	0.954	14.70	22
	10	ECM.BFGS	Hierarchical	0.974	13.92	38	10	ECME.BFGS	Hierarchical scale	0.954	14.70	22
	11	ECM.Nelder.Mead	Hierarchical	0.974	13.92	47	11	ECME.Nelder.Mead	Hierarchical	0.954	14.70	25
	12	MCECM.Nelder.Mead	Hierarchical	0.974	13.92	47	12	ECME.BFGS	Hierarchical	0.954	14.70	25
	13	MCECM.BFGS	Hierarchical scale	0.972	13.92	36	13	ECM.BFGS	Kmeans scale	0.953	14.78	25
	14	ECM.BFGS	Hierarchical scale	0.972	13.93	36	14	MCECM.BFGS	Kmeans scale	0.953	14.78	25
	15	MCECM.Nelder.Mead	Hierarchical scale	0.972	13.93	45	15	MCECM.Nelder.Mead	Kmeans scale	0.953	14.78	29
	16	ECME.BFGS	Hierarchical scale	0.972	13.93	52	16	ECM.Nelder.Mead	Kmeans scale	0.953	14.78	37
	17	ECME.Nelder.Mead	Hierarchical scale	0.972	13.93	105	17	ECME.Nelder.Mead	Kmeans scale	0.951	14.74	33
	18	ECM.Nelder.Mead	Hierarchical scale	0.971	13.93	45	18	ECME.BFGS	Kmeans scale	0.951	14.74	36
	19	MCECM.BFGS	Random	0.958	14.04	224	19	ECM.BFGS	Kmeans	0.947	14.80	52
	20	ECM.BFGS	Random	0.949	14.07	214	20	ECM.Nelder.Mead	Kmeans	0.947	14.80	56
	21	MCECM.Nelder.Mead	Random	0.946	14.09	269	21	MCECM.BFGS	Kmeans	0.946	14.80	51
	22	ECME.Nelder.Mead	Random	0.946	14.10	294	22	MCECM.Nelder.Mead	Kmeans	0.946	14.80	57
	23	ECME.BFGS	Random	0.946	14.11	249	23	ECME.BFGS	Kmeans	0.940	14.74	45
	24	ECME.Nelder.Mead	Kmeans	0.939	14.06	185	24	ECME.Nelder.Mead	Kmeans	0.939	14.76	48
	25	ECM.Nelder.Mead	Random	0.937	14.11	265	25	ECME.BFGS	Random	0.918	14.74	72
	26	ECME.BFGS	Kmeans	0.935	14.06	146	26	ECME.Nelder.Mead	Random	0.918	14.74	75
	27	MCECM.Nelder.Mead	Kmeans	0.840	14.38	123	27	ECM.BFGS	Random	0.909	14.75	128
	28	ECM.Nelder.Mead	Kmeans	0.834	14.40	125	28	MCECM.BFGS	Random	0.909	14.76	128
	29	ECM.BFGS	Kmeans	0.830	14.41	87	29	MCECM.Nelder.Mead	Random	0.909	14.76	141
	30	MCECM.BFGS	Kmeans	0.830	14.41	90	30	ECM.Nelder.Mead	Random	0.909	14.76	145
7 (1000)	1	ECM.BFGS	Hierarchical scale	0.980	13.79	61	1	ECME.Nelder.Mead	Kmeans	0.958	14.72	66
	2	MCECM.BFGS	Hierarchical scale	0.980	13.79	71	2	ECM.BFGS	Kmeans	0.958	14.72	92
	3	ECME.Nelder.Mead	Hierarchical scale	0.980	13.79	106	3	ECME.BFGS	Kmeans	0.958	14.72	645
	4	MCECM.Nelder.Mead	Hierarchical scale	0.980	13.79	278	4	MCECM.Nelder.Mead	Kmeans	0.958	14.72	970
	5	ECME.BFGS	Hierarchical scale	0.980	13.79	329	5	ECM.Nelder.Mead	Kmeans	0.958	14.72	1004
	6	ECM.Nelder.Mead	Hierarchical	0.980	13.79	344	6	ECM.BFGS	Hierarchical scale	0.957	14.70	24
	7	ECM.Nelder.Mead	Hierarchical scale	0.980	13.79	349	7	ECM.Nelder.Mead	Hierarchical scale	0.957	14.70	25
	8	ECME.Nelder.Mead	Hierarchical	0.980	13.79	565	8	MCECM.BFGS	Kmeans	0.957	14.72	147
	9	MCECM.Nelder.Mead	Hierarchical	0.979	13.79	362	9	MCECM.Nelder.Mead	Hierarchical scale	0.956	14.70	24
	10	ECM.BFGS	Hierarchical	0.978	13.80	72	10	ECME.BFGS	Hierarchical scale	0.956	14.70	27
	11	MCECM.BFGS	Hierarchical	0.978	13.80	73	11	ECME.Nelder.Mead	Hierarchical scale	0.956	14.70	28
	12	ECME.BFGS	Hierarchical	0.978	13.80	448	12	MCECM.BFGS	Hierarchical scale	0.956	14.70	88
	13	MCECM.BFGS	Random	0.976	13.82	343	13	ECM.Nelder.Mead	Kmeans scale	0.952	14.86	44
	14	ECM.BFGS	Kmeans scale	0.974	13.82	133	14	ECME.BFGS	Hierarchical	0.951	14.78	372
	15	MCECM.BFGS	Kmeans scale	0.974	13.82	139	15	ECM.BFGS	Kmeans scale	0.951	14.82	42
	16	ECME.Nelder.Mead	Kmeans scale	0.974	13.82	176	16	MCECM.Nelder.Mead	Kmeans scale	0.951	14.85	44
	17	MCECM.Nelder.Mead	Kmeans scale	0.974	13.82	540	17	ECME.BFGS	Kmeans scale	0.950	14.76	45
	18	ECM.Nelder.Mead	Kmeans scale	0.974	13.82	545	18	ECME.Nelder.Mead	Kmeans scale	0.950	14.76	47
	19	ECME.BFGS	Kmeans scale	0.974	13.82	639	19	ECM.BFGS	Hierarchical	0.950	14.78	31
	20	ECM.BFGS	Random	0.974	13.83	315	20	ECME.Nelder.Mead	Hierarchical	0.950	14.78	35
	21	ECME.Nelder.Mead	Random	0.974	13.84	1166	21	ECM.Nelder.Mead	Hierarchical	0.950	14.78	87
	22	ECM.Nelder.Mead	Random	0.973	13.84	465	22	MCECM.BFGS	Hierarchical	0.950	14.78	149
	23	MCECM.Nelder.Mead	Random	0.973	13.85	517	23	MCECM.Nelder.Mead	Hierarchical	0.950	14.78	247
	24	ECME.Nelder.Mead	Kmeans	0.969	13.83	1536	24	MCECM.BFGS	Kmeans scale	0.950	14.82	155
	25	ECME.BFGS	Kmeans	0.960	13.85	552	25	ECME.BFGS	Random	0.947	14.67	259
	26	ECME.BFGS	Random	0.960	13.90	412	26	ECME.Nelder.Mead	Random	0.947	14.70	97
	27	ECM.BFGS	Kmeans	0.899	14.06	206	27	ECM.BFGS	Random	0.938	14.73	239
	28	MCECM.BFGS	Kmeans	0.899	14.06	258	28	ECM.Nelder.Mead	Random	0.938	14.73	248
	29	MCECM.Nelder.Mead	Kmeans	0.898	14.06	679	29	MCECM.Nelder.Mead	Random	0.938	14.73	248
	30	ECM.Nelder.Mead	Kmeans	0.898	14.06	736	30	MCECM.BFGS	Random	0.938	14.73	1014

	General VCMM						Gaussian VCMM					
Setting (Sample size)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)	Ranking	Algorithm	Initialization strategy	Classification rate	Normalised BIC	Computation time (Seconds)
8 (500)	1	ECM.Nelder.Mead	Hierarchical	0.991	17.26	59	1	ECM.BFGS	Random	0.907	19.92	53
	2	ECM.BFGS	Hierarchical	0.990	17.26	47	2	ECM.Nelder.Mead	Random	0.907	19.92	64
	3	MCECM.BFGS	Hierarchical	0.990	17.26	48	3	MCECM.BFGS	Random	0.906	19.92	55
	4	MCECM.Nelder.Mead	Hierarchical	0.990	17.26	55	4	MCECM.Nelder.Mead	Random	0.906	19.92	62
	5	ECME.BFGS	Hierarchical	0.990	17.26	73	5	ECM.BFGS	Kmeans	0.905	19.92	61
	6	ECME.Nelder.Mead	Hierarchical	0.990	17.26	87	6	MCECM.BFGS	Kmeans	0.905	19.92	62
	7	ECME.BFGS	Random	0.912	17.56	105	7	MCECM.Nelder.Mead	Kmeans	0.905	19.92	72
	8	ECME.Nelder.Mead	Random	0.900	17.60	117	8	ECM.Nelder.Mead	Kmeans	0.905	19.92	74
	9	ECM.Nelder.Mead	Random	0.862	17.67	104	9	MCECM.BFGS	Hierarchical	0.900	19.90	33
	10	ECM.BFGS	Random	0.861	17.67	86	10	MCECM.Nelder.Mead	Hierarchical	0.900	19.90	37
	11	MCECM.Nelder.Mead	Random	0.843	17.73	103	11	ECM.BFGS	Hierarchical	0.899	19.90	33
	12	MCECM.BFGS	Random	0.843	17.73	441	12	ECM.Nelder.Mead	Hierarchical	0.899	19.90	39
	13	ECME.Nelder.Mead	Hierarchical scale	0.752	18.36	196	13	ECME.Nelder.Mead	Hierarchical	0.893	19.90	36
	14	ECM.BFGS	Hierarchical scale	0.745	18.36	121	14	ECME.BFGS	Hierarchical	0.892	19.90	33
	15	ECME.BFGS	Hierarchical scale	0.743	18.41	147	15	ECM.BFGS	Kmeans scale	0.892	19.96	59
	16	MCECM.BFGS	Hierarchical scale	0.738	18.38	121	16	MCECM.BFGS	Kmeans scale	0.892	19.96	59
	17	ECM.Nelder.Mead	Hierarchical scale	0.732	18.40	145	17	MCECM.Nelder.Mead	Kmeans scale	0.892	19.96	69
	18	MCECM.Nelder.Mead	Hierarchical scale	0.726	18.42	143	18	ECM.Nelder.Mead	Kmeans scale	0.892	19.96	71
	19	ECME.BFGS	Kmeans	0.658	18.65	229	19	ECM.BFGS	Hierarchical scale	0.888	19.95	50
	20	MCECM.BFGS	Kmeans	0.656	18.67	1105	20	ECM.Nelder.Mead	Hierarchical scale	0.888	19.95	60
	21	ECM.BFGS	Kmeans	0.654	18.68	168	21	MCECM.BFGS	Hierarchical scale	0.887	19.95	50
	22	MCECM.Nelder.Mead	Kmeans	0.654	18.68	224	22	MCECM.Nelder.Mead	Hierarchical scale	0.887	19.95	58
	23	ECM.Nelder.Mead	Kmeans	0.650	18.69	214	23	ECME.BFGS	Kmeans	0.882	19.91	52
	24	ECME.Nelder.Mead	Kmeans	0.648	18.72	279	24	ECME.Nelder.Mead	Kmeans	0.881	19.91	61
	25	MCECM.BFGS	Kmeans scale	0.614	18.89	101	25	ECME.BFGS	Kmeans scale	0.874	19.96	53
	26	ECME.BFGS	Kmeans scale	0.611	18.92	118	26	ECME.Nelder.Mead	Kmeans scale	0.874	19.96	61
	27	ECM.Nelder.Mead	Kmeans scale	0.608	18.92	115	27	ECME.Nelder.Mead	Random	0.870	19.91	43
	28	MCECM.Nelder.Mead	Kmeans scale	0.607	18.91	117	28	ECME.Nelder.Mead	Hierarchical scale	0.867	19.95	55
	29	ECME.Nelder.Mead	Kmeans scale	0.603	18.96	150	29	ECME.BFGS	Random	0.866	19.91	38
	30	ECM.BFGS	Kmeans scale	0.602	18.95	89	30	ECME.BFGS	Hierarchical scale	0.862	19.96	48
9 (500)	1	ECME.BFGS	Random	0.878	14.58	119	1	MCECM.BFGS	Random	0.726	16.73	76
	2	ECME.Nelder.Mead	Random	0.877	14.57	141	2	MCECM.Nelder.Mead	Random	0.726	16.73	85
	3	MCECM.BFGS	Random	0.867	14.58	98	3	ECM.BFGS	Random	0.722	16.68	82
	4	ECM.BFGS	Random	0.860	14.60	98	4	ECM.Nelder.Mead	Random	0.722	16.68	88
	5	MCECM.Nelder.Mead	Random	0.847	14.65	117	5	ECME.BFGS	Random	0.690	16.61	48
	6	ECM.Nelder.Mead	Random	0.847	14.65	123	6	ECME.Nelder.Mead	Random	0.664	16.56	52
	7	MCECM.Nelder.Mead	Kmeans scale	0.757	15.46	169	7	ECM.BFGS	Hierarchical	0.645	16.83	68
	8	ECME.BFGS	Kmeans scale	0.757	15.49	137	8	MCECM.Nelder.Mead	Hierarchical	0.645	16.83	71
	9	ECM.BFGS	Kmeans scale	0.756	15.46	118	9	ECM.Nelder.Mead	Hierarchical	0.645	16.83	72
	10	ECM.Nelder.Mead	Kmeans scale	0.756	15.46	167	10	ECME.BFGS	Hierarchical	0.639	16.80	53
	11	MCECM.BFGS	Kmeans scale	0.750	15.50	116	11	ECME.Nelder.Mead	Hierarchical	0.639	16.80	57
	12	ECME.Nelder.Mead	Kmeans scale	0.731	15.67	178	12	MCECM.BFGS	Hierarchical	0.637	16.86	61
	13	ECME.Nelder.Mead	Hierarchical	0.728	15.20	173	13	ECM.BFGS	Kmeans scale	0.634	16.99	90
	14	ECME.BFGS	Hierarchical	0.720	15.24	128	14	ECM.Nelder.Mead	Kmeans scale	0.634	16.99	96
	15	ECME.BFGS	Hierarchical scale	0.708	15.50	124	15	MCECM.BFGS	Kmeans scale	0.631	17.02	83
	16	ECM.BFGS	Hierarchical	0.700	15.32	84	16	MCECM.Nelder.Mead	Kmeans scale	0.631	17.02	96
	17	MCECM.BFGS	Hierarchical	0.700	15.32	85	17	ECME.Nelder.Mead	Kmeans scale	0.622	16.98	84
	18	ECM.Nelder.Mead	Hierarchical	0.699	15.32	111	18	ECME.BFGS	Kmeans scale	0.620	16.98	75
	19	MCECM.Nelder.Mead	Hierarchical	0.699	15.32	111	19	ECME.BFGS	Hierarchical scale	0.608	17.00	66
	20	ECME.Nelder.Mead	Hierarchical scale	0.698	15.55	170	20	ECME.Nelder.Mead	Hierarchical scale	0.608	17.00	75
	21	MCECM.Nelder.Mead	Hierarchical scale	0.685	15.59	139	21	ECM.BFGS	Hierarchical scale	0.606	17.05	74
	22	ECM.Nelder.Mead	Hierarchical scale	0.685	15.59	140	22	ECM.Nelder.Mead	Hierarchical scale	0.606	17.05	80
	23	MCECM.BFGS	Hierarchical scale	0.679	15.61	100	23	MCECM.BFGS	Hierarchical scale	0.603	17.08	67
	24	ECM.BFGS	Hierarchical scale	0.678	15.62	97	24	MCECM.Nelder.Mead	Hierarchical scale	0.603	17.08	76
	25	ECME.BFGS	Kmeans	0.573	15.95	150	25	ECME.BFGS	Kmeans	0.580	17.28	74
	26	MCECM.Nelder.Mead	Kmeans	0.569	15.95	161	26	ECME.Nelder.Mead	Kmeans	0.580	17.28	85
	27	ECME.Nelder.Mead	Kmeans	0.569	15.95	206	27	MCECM.BFGS	Kmeans	0.576	17.35	76
	28	ECM.Nelder.Mead	Kmeans	0.564	16.00	156	28	MCECM.Nelder.Mead	Kmeans	0.576	17.35	84
	29	ECM.BFGS	Kmeans	0.562	15.99	116	29	ECM.Nelder.Mead	Kmeans	0.576	17.35	88
	30	MCECM.BFGS	Kmeans	0.561	15.99	115	30	ECM.BFGS	Kmeans	0.575	17.37	75

C Appendix for Section 4 Real data sets

C.1 Estimated vine copula model for AIS data

Table C.1: The result of the fitted vine copula model for the AIS data by using different VCMM algorithms and initialization strategies. The variable encoding is given as follows: (AIS) 1: LBM, 2: Wt, 3: BMI, 4: WBC, 5: PBF. For marginal distributions (Left column), the estimated marginal distributions and parameters for each cluster are shown. For vine tree structure (Right column), the first and second tree level of the estimated vine copula models are shown here. The number 1,5 represents the edge of the tree level, letter N is the abbreviation of the copula and the true parameter value and corresponding Kendall's τ of the pair copula are given inside the parenthesis (parameter(s)/Kendall's τ) near the letter. The meaning of the abbreviation for marginal distribution and copula families is shown in the appendix B.1 and B.2.

	Marginal distributions		Vine tree structure							
			Cluster 1				Cluster 2			
Variable	Cluster 1	Cluster 2	Tree 1		Tree 2		Tree 1		Tree 2	
AIS										
General VCMM (Random)	Classification rate : 60.2% Normalised BIC : 25.82									
1	$snorm(70.79, 11.85, 0.61)$	$llogis(9.28, 59.85)$	1,5	R90J(-3.35/-0.56)	2,5;1	SBB8(4.43,0.82/0.51)	2,1	BB1(1.5,2.16/0.74)	3,1;2	N(-0.18/-0.12)
2	$\mathcal{N}(79.54, 10.06)$	$llogis(8.58, 70.85)$	2,1	N(0.99/0.91)	3,1;2	R90G(-1.14/-0.12)	3,2	N(0.89/0.69)	4,2;3	SC(0.04/0.02)
3	$\Gamma(183.87, 7.97)$	$snorm(22.96, 3.32, 1.67)$	3,2	N(0.79/0.58)	4,2;3	R90C(-0.07/-0.03)	3,4	F(1.49/0.16)	5,4;3	SG(1.09/0.08)
4	$llogis(7.34, 6.83)$	$\Gamma(16.14, 2.26)$	4,3	J(1.26/0.13)			5,3	F(2.45/0.26)		
5	$snorm(11.98, 4.4, 142.37)$	$snorm(14.54, 6.73, 268.28)$								
General VCMM (Copula with single parameter) (Random)	Classification rate : 61.6% Normalised BIC : 26.44									
1	$snorm(70.17, 11.75, 0.65)$	$llogis(9.1, 59.53)$	1,5	R90J(-3.52/-0.57)	2,5;1	F(-0.08/-0.01)	2,1	SG(4.02/0.75)	3,1;2	N(-0.27/-0.17)
2	$snorm(77.51, 11.35, 2.97)$	$\Gamma(21.87, 0.3)$	2,1	F(23.02/0.84)	3,1;2	SC(0.27/0.12)	3,2	N(0.9/0.71)	5,2;3	F(-0.79/-0.09)
3	$\mathcal{N}(22.93, 1.78)$	$llogis(12.35, 22.59)$	3,2	N(0.76/0.54)	4,2;3	SC(0/0)	3,4	F(1.36/0.15)	5,4;3	SG(1.08/0.07)
4	$llogis(7.32, 6.8)$	$\Gamma(16.21, 2.27)$	4,3	G(1.2/0.17)			5,3	F(2.74/0.28)		
5	$sstd(11.23, 4.93, 4.27, 146.43)$	$snorm(14.88, 6.99, 121.58)$								
Gaussian VCMM (Random)	Classification rate : 65.8% Normalised BIC : 23.06									
1	$\mathcal{N}(67.69, 13.37)$	$\mathcal{N}(61.37, 11.71)$	2,3	N(0.88/0.68)	1,3;2	N(-0.16/-0.1)	1,5	N(-0.34/-0.22)	2,5;1	N(0.99/0.91)
2	$\mathcal{N}(74.14, 14.32)$	$\mathcal{N}(76.08, 13.26)$	1,2	N(1/0.94)	5,2;1	N(0.97/0.83)	2,1	N(0.95/0.81)	3,1;2	N(-0.19/-0.12)
3	$\mathcal{N}(22.38, 2.46)$	$\mathcal{N}(23.68, 3.13)$	5,1	N(-0.33/-0.22)	4,1;5	N(0.14/0.09)	3,2	N(0.85/0.65)	4,2;3	N(0.05/0.03)
4	$\mathcal{N}(7.03, 1.84)$	$\mathcal{N}(7.2, 1.73)$	5,4	N(0.23/0.15)			4,3	N(0.27/0.18)		
5	$\mathcal{N}(8.8, 1.97)$	$\mathcal{N}(19.36, 4.35)$								
General VCMM (Kmeans)	Classification rate : 77.5% Normalised BIC : 26.00									
1	$snorm(56.44, 7.22, 0.6)$	$snorm(78.08, 7.62, 83.9)$	3,4	SC(0.24/0.11)	2,4;3	R90C(-0.02/-0.01)	2,1	G(3.36/0.7)	3,1;2	N(-0.31/-0.2)
2	$sstd(66.86, 9.67, 8.29, 0.75)$	$snorm(87.38, 10.26, 160.5)$	2,1	SJ(3.77/0.6)	3,1;2	N(-0.2/-0.13)	3,2	BB8(4.39,0.85/0.53)	5,2;3	N(0.04/0.03)
3	$lnorm(3.08, 0.1)$	$sstd(24.82, 3.22, 3.03, 1.59)$	3,2	N(0.83/0.62)	5,2;3	SC(0.25/0.11)	5,3	G(1.71/0.42)	4,3;5	R270C(-0.26/-0.12)
4	$lnorm(1.9, 0.24)$	$lnorm(1.98, 0.26)$	5,3	SC(0.97/0.33)			5,4	SBB8(2.58,0.79/0.28)		
5	$snorm(15.11, 7.16, 164.59)$	$sstd(10.39, 4.85, 3.2, 176.82)$								
General VCMM (Copula with single parameter) (Kmeans)	Classification rate : 76.7% Normalised BIC : 25.79									
1	$snorm(56.35, 7.17, 0.58)$	$snorm(78.41, 7.87, 92.25)$	3,4	SC(0.27/0.12)	2,4;3	F(-0.47/-0.05)	2,1	G(3.23/0.69)	3,1;2	F(-2.45/-0.26)
2	$snorm(66.64, 9.15, 0.67)$	$snorm(87.47, 10.33, 133.42)$	2,1	SJ(3.79/0.6)	3,1;2	R90C(-0.25/-0.11)	3,2	G(1.99/0.5)	5,2;3	R90C(-0.01/-0.01)
3	$lnorm(3.07, 0.1)$	$sstd(24.84, 3.18, 3.09, 1.58)$	3,2	N(0.83/0.62)	5,2;3	SC(0.27/0.12)	5,3	F(4.87/0.45)	4,3;5	R270G(-1.06/-0.06)
4	$lnorm(1.9, 0.24)$	$lnorm(1.97, 0.26)$	5,3	SC(0.92/0.31)			5,4	C(0.35/0.15)		
5	$snorm(15.03, 7.1, 231.11)$	$sstd(10.84, 7.86, 2.42, 118.25)$								
Gaussian VCMM (Kmeans)	Classification rate : 91.4% Normalised BIC : 23.57									
1	$\mathcal{N}(56.33, 8.89)$	$\mathcal{N}(74.68, 9.72)$	2,5	N(0.61/0.42)	1,5;2	N(-0.98/-0.87)	2,1	N(0.99/0.9)	3,1;2	N(-0.28/-0.18)
2	$\mathcal{N}(69.02, 12.71)$	$\mathcal{N}(81.89, 11.86)$	2,1	N(0.94/0.78)	3,1;2	N(-0.14/-0.09)	3,2	N(0.83/0.62)	5,2;3	N(0.23/0.15)
3	$\mathcal{N}(22.32, 3.02)$	$\mathcal{N}(23.69, 2.46)$	3,2	N(0.87/0.67)	4,2;3	N(-0.01/0)	5,3	N(0.65/0.45)	4,3;5	N(-0.15/-0.1)
4	$\mathcal{N}(7.1, 1.69)$	$\mathcal{N}(7.12, 1.91)$	4,3	N(0.23/0.15)			5,4	N(0.36/0.23)		
5	$\mathcal{N}(17.79, 5.28)$	$\mathcal{N}(8.59, 2.16)$								

	Marginal distributions		Vine tree structure							
Variable	Cluster 1	Cluster 2	Cluster 1				Cluster 2			
			Tree 1		Tree 2		Tree 1		Tree 2	
AIS										
General VCMM (Hierarchical)										
Classification rate : 91.4% Normalised BIC : 24.21										
1	$t_3(55.42, 6.12)$	$lnorm(4.3, 0.13)$	2,1	N(0.94/0.77)	5,1;2	N(-0.86/-0.66)	2,1	N(0.99/0.9)	3,1;2	F(-2.13/-0.23)
2	$llogis(9.14, 68.41)$	$lnorm(4.39, 0.14)$	3,4	F(1.57/0.17)	2,4;3	F(-0.41/-0.05)	3,2	N(0.81/0.6)	5,2;3	C(0.2/0.09)
3	$llogis(13.2, 22.04)$	$llogis(19.93, 23.36)$	2,3	N(0.88/0.68)	5,3;2	N(0.18/0.11)	5,3	N(0.58/0.39)	4,3;5	R270C(-0.17/-0.08)
4	$lnorm(1.93, 0.24)$	$lnorm(1.93, 0.26)$	5,2	SBB8(3.78,0.88/0.5)			5,4	F(2.67/0.28)		
5	$\Gamma(11.31, 0.63)$	$snorm(8.63, 2.14, 4.08)$								
General VCMM (Copula with single parameter) (Hierarchical)										
Classification rate : 92.4% Normalised BIC : 23.83										
1	$llogis(12.42, 55.34)$	$lnorm(4.3, 0.13)$	2,5	C(1.53/0.43)	1,5;2	R90G(-3.38/-0.7)	2,1	SG(9.87/0.9)	3,1;2	R90G(-1.32/-0.24)
2	$\mathcal{N}(68.59, 12.29)$	$lnorm(4.4, 0.14)$	2,1	T(0.94,6.26/0.78)	3,1;2	N(-0.25/-0.16)	3,2	N(0.83/0.62)	5,2;3	C(0.2/0.09)
3	$lnorm(3.09, 0.12)$	$llogis(18.22, 23.46)$	3,2	N(0.87/0.67)	4,2;3	R270C(0/0)	5,3	T(0.64,11.61/0.45)	4,3;5	R270C(-0.17/-0.08)
4	$lnorm(1.93, 0.24)$	$lnorm(1.93, 0.26)$	4,3	F(1.56/0.17)			5,4	F(2.53/0.26)		
5	$\Gamma(11.04, 0.62)$	$snorm(8.81, 2.28, 4.3)$								
Gaussian VCMM (Hierarchical)										
Classification rate : 90.5% Normalised BIC : 23.57										
1	$\mathcal{N}(56.27, 8.9)$	$\mathcal{N}(74.56, 9.74)$	2,5	N(0.62/0.42)	1,5;2	N(-0.98/-0.86)	2,1	N(0.99/0.9)	3,1;2	N(-0.28/-0.18)
2	$\mathcal{N}(69, 12.77)$	$\mathcal{N}(81.77, 11.85)$	2,1	N(0.94/0.79)	3,1;2	N(-0.14/-0.09)	3,2	N(0.83/0.63)	5,2;3	N(0.2/0.13)
3	$\mathcal{N}(22.31, 3.04)$	$\mathcal{N}(23.68, 2.45)$	3,2	N(0.87/0.67)	4,2;3	N(-0.01/-0.01)	5,3	N(0.63/0.44)	4,3;5	N(-0.16/-0.1)
4	$\mathcal{N}(7.08, 1.69)$	$\mathcal{N}(7.14, 1.91)$	4,3	N(0.23/0.15)			5,4	N(0.37/0.24)		
5	$\mathcal{N}(17.85, 5.27)$	$\mathcal{N}(8.62, 2.17)$								
General VCMM (Kmeans scale)										
Classification rate : 91.4% Normalised BIC : 23.52										
1	$logis(55.22, 3.77)$	$lnorm(4.32, 0.12)$	2,5	F(6.06/0.52)	1,5;2	N(-0.99/-0.9)	2,1	SG(6.35/0.84)	3,1;2	R90C(-0.42/-0.17)
2	$\mathcal{N}(66.92, 10.53)$	$snorm(83.75, 11.59, 1.72)$	2,1	SG(3.56/0.72)	3,1;2	N(-0.25/-0.16)	3,2	BB8(4.87,0.87/0.58)	5,2;3	N(0.23/0.15)
3	$lnorm(3.08, 0.11)$	$sstd(24.17, 2.81, 4.67, 1.95)$	3,2	N(0.85/0.65)	4,2;3	R270C(0/0)	5,3	BB8(4.07,0.82/0.48)	4,3;5	R90J(-1.16/-0.08)
4	$lnorm(1.92, 0.24)$	$lnorm(1.95, 0.26)$	4,3	F(1.16/0.13)			5,4	SBB8(6,0.42/0.3)		
5	$\mathcal{N}(17.18, 5.82)$	$sstd(9.46, 3.66, 3.8, 5.55)$								
General VCMM (Copula with single parameter) (Kmeans scale)										
Classification rate : 91.4% Normalised BIC : 23.65										
1	$logis(55.22, 3.77)$	$lnorm(4.32, 0.12)$	2,5	F(6.06/0.52)	1,5;2	N(-0.99/-0.9)	2,1	SG(6.35/0.84)	3,1;2	R90C(-0.48/-0.19)
2	$\mathcal{N}(66.92, 10.53)$	$snorm(83.75, 11.59, 1.72)$	2,1	SG(3.56/0.72)	3,1;2	N(-0.25/-0.16)	3,2	G(2.24/0.55)	5,2;3	C(0.25/0.11)
3	$lnorm(3.08, 0.11)$	$sstd(24.17, 2.81, 4.67, 1.95)$	3,2	N(0.85/0.65)	4,2;3	R270C(0/0)	5,3	G(1.79/0.44)	4,3;5	N(-0.07/-0.05)
4	$lnorm(1.92, 0.24)$	$lnorm(1.95, 0.26)$	4,3	F(1.16/0.13)			5,4	T(0.41,30/0.27)		
5	$\mathcal{N}(17.18, 5.82)$	$sstd(9.46, 3.66, 3.8, 5.55)$								
Gaussian VCMM (Kmeans scale)										
Classification rate : 90.5% Normalised BIC : 23.57										
1	$\mathcal{N}(56.27, 8.9)$	$\mathcal{N}(74.56, 9.74)$	2,5	N(0.62/0.42)	1,5;2	N(-0.98/-0.86)	2,1	N(0.99/0.9)	3,1;2	N(-0.28/-0.18)
2	$\mathcal{N}(69, 12.77)$	$\mathcal{N}(81.77, 11.85)$	2,1	N(0.94/0.79)	3,1;2	N(-0.14/-0.09)	3,2	N(0.83/0.63)	5,2;3	N(0.2/0.13)
3	$\mathcal{N}(22.31, 3.04)$	$\mathcal{N}(23.68, 2.45)$	3,2	N(0.87/0.67)	4,2;3	N(-0.01/-0.01)	5,3	N(0.63/0.44)	4,3;5	N(-0.16/-0.1)
4	$\mathcal{N}(7.08, 1.69)$	$\mathcal{N}(7.14, 1.91)$	4,3	N(0.23/0.15)			5,4	N(0.37/0.24)		
5	$\mathcal{N}(17.85, 5.27)$	$\mathcal{N}(8.62, 2.17)$								
General VCMM (Hierarchical scale)										
Classification rate : 92.4% Normalised BIC : 23.92										
1	$llogis(12.36, 55.59)$	$lnorm(4.31, 0.13)$	2,5	SBB8(4.07,0.82/0.48)	1,5;2	T(-0.92,3.56/-0.74)	2,1	SG(10.7/0.91)	3,1;2	R270G(-1.27/-0.21)
2	$\mathcal{N}(68.68, 12.13)$	$lnorm(4.4, 0.14)$	2,1	BB1(1.05,2.81/0.77)	3,1;2	N(-0.24/-0.15)	3,2	N(0.83/0.63)	5,2;3	N(0.26/0.17)
3	$llogis(14.4, 21.97)$	$llogis(17.89, 23.49)$	3,2	N(0.87/0.67)	4,2;3	F(-0.27/-0.03)	5,3	G(1.78/0.44)	4,3;5	R270C(-0.16/-0.08)
4	$\Gamma(18, 2.54)$	$lnorm(1.93, 0.26)$	4,3	F(1.55/0.17)			5,4	F(2.58/0.27)		
5	$\Gamma(10.82, 0.61)$	$snorm(8.74, 2.23, 4.24)$								
General VCMM (Copula with single parameter) (Hierarchical scale)										
Classification rate : 92.4% Normalised BIC : 23.85										
1	$llogis(12.36, 55.59)$	$lnorm(4.31, 0.13)$	2,1	T(0.94,5.09/0.77)	5,1;2	R270G(-3.87/-0.74)	2,1	SG(10.7/0.91)	3,1;2	R270G(-1.27/-0.21)
2	$\mathcal{N}(68.68, 12.13)$	$lnorm(4.4, 0.14)$	3,4	F(1.55/0.17)	2,4;3	F(-0.27/-0.03)	3,2	N(0.83/0.63)	5,2;3	N(0.26/0.17)
3	$llogis(14.4, 21.97)$	$llogis(17.89, 23.49)$	2,3	N(0.87/0.67)	5,3;2	SC(0.39/0.16)	5,3	G(1.78/0.44)	4,3;5	R270C(-0.16/-0.08)
4	$\Gamma(18, 2.54)$	$lnorm(1.93, 0.26)$	5,2	C(1.4/0.41)			5,4	F(2.58/0.27)		
5	$\Gamma(10.82, 0.61)$	$snorm(8.74, 2.23, 4.24)$								
Gaussian VCMM (Hierarchical scale)										
Classification rate : 91.4% Normalised BIC : 23.57										
1	$\mathcal{N}(56.33, 8.89)$	$\mathcal{N}(74.68, 9.72)$	2,5	N(0.61/0.42)	1,5;2	N(-0.98/-0.87)	2,1	N(0.99/0.9)	3,1;2	N(-0.28/-0.18)
2	$\mathcal{N}(69.02, 12.71)$	$\mathcal{N}(81.89, 11.86)$	2,1	N(0.94/0.78)	3,1;2	N(-0.14/-0.09)	3,2	N(0.83/0.62)	5,2;3	N(0.23/0.15)
3	$\mathcal{N}(22.32, 3.02)$	$\mathcal{N}(23.69, 2.46)$	3,2	N(0.87/0.67)	4,2;3	N(-0.01/0)	5,3	N(0.65/0.45)	4,3;5	N(-0.15/-0.1)
4	$\mathcal{N}(7.1, 1.69)$	$\mathcal{N}(7.12, 1.91)$	4,3	N(0.23/0.15)			5,4	N(0.36/0.23)		
5	$\mathcal{N}(17.79, 5.28)$	$\mathcal{N}(8.59, 2.16)$								

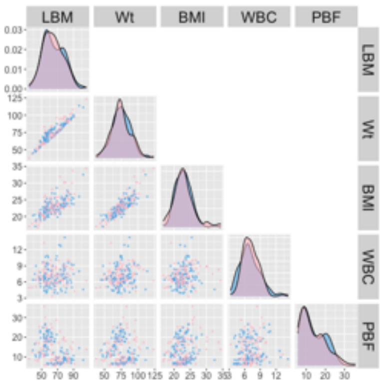
C.2 Estimated vine copula model for BCW data

Table C.2: The result of the fitted vine copula model for BCW data by using different VCMM algorithms and initialization strategies. The variable encoding is given as follows: (BCW) 1: PSE, 2: ES, 3: EC and 4: ECP. For marginal distributions (Left column), the estimated marginal distributions and parameters for each cluster are shown. For vine tree structure (Right column), the first and second tree level of the estimated vine copula models are shown here. The number 1,5 represents the edge of the tree level, letter N is the abbreviation of the copula and the true parameter value and corresponding Kendall's τ of the pair copula are given inside the parenthesis (parameter(s)/Kendall's τ) near the letter. The meaning of the abbreviation for marginal distribution and copula families is shown in the appendix B.1 and B.2.

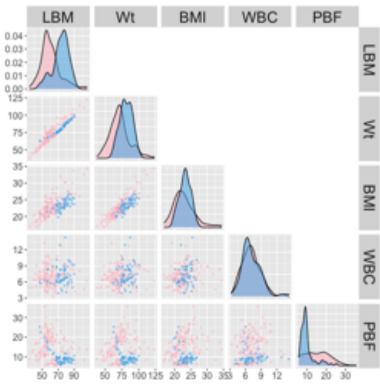
Variable	Marginal distributions: Cluster 1	Marginal distributions: Cluster 2	Vine tree structure: Cluster 1, Tree 1		Cluster 1, Tree 2		Cluster 2, Tree 1		Cluster 2, Tree 2	
BCW										
General VCMM (Random)										
Classification rate : 50.4% Normalised BIC : -7.02										
1	$lnorm(1.04, 0.59)$	$sstd(2.4, 1.47, 3.42, 4.25)$	4,2	T(0.41,8.73/0.27)	1,2;4	F(0.69/0.08)	3,2	F(4.21/0.4)	4,2;3	SJ(1.21/0.11)
2	$\Gamma(42.86, 358.15)$	$lnorm(-1.95, 0.14)$	4,1	BB8(3.45,0.96/0.53)	3,1;4	R270G(-1.15/-0.13)	4,1	BB8(3.61,0.93/0.52)	3,1;4	F(-0.67/-0.07)
3	$snorm(0.23, 0.17, 296.73)$	$snorm(0.32, 0.22, 315.76)$	4,3	SBB8(6,0.88/0.65)			4,3	F(13.63/0.74)		
4	$snorm(0.09, 0.06, 1.9)$	$snorm(0.13, 0.06, 1.85)$								
General VCMM (Copula with single parameter) (Random)										
Classification rate : 51.1% Normalised BIC : -7.00										
1	$sstd(3.44, 2.85, 2.98, 5.46)$	$snorm(2.14, 0.85, 2.33)$	3,2	T(0.42,22.14/0.28)	4,2;3	SJ(1.09/0.05)	4,2	T(0.67,9.46/0.47)	1,2;4	R90C(-0.22/-0.1)
2	$\Gamma(55.5, 416.91)$	$snorm(0.13, 0.03, 1.69)$	4,1	SC(1.29/0.39)	3,1;4	SC(0.08/0.04)	4,1	G(1.46/0.31)	3,1;4	F(0.69/0.08)
3	$snorm(0.25, 0.19, 192.62)$	$snorm(0.31, 0.23, 8.48)$	4,3	F(22.07/0.83)			4,3	F(17.45/0.79)		
4	$snorm(0.12, 0.07, 1.41)$	$snorm(0.1, 0.06, 2.19)$								
Gaussian VCMM (Random)										
Classification rate : 78.4% Normalised BIC : -6.61										
1	$\mathcal{N}(2, 0.69)$	$\mathcal{N}(4.69, 2.61)$	4,2	N(0.45/0.3)	1,2;4	N(-0.07/-0.05)	2,1	N(-0.24/-0.16)	3,1;2	N(-0.01/-0.01)
2	$\mathcal{N}(0.13, 0.02)$	$\mathcal{N}(0.14, 0.02)$	4,1	N(0.15/0.1)	3,1;4	N(-0.01/-0.01)	3,2	N(0.44/0.29)	4,2;3	N(0.27/0.18)
3	$\mathcal{N}(0.17, 0.11)$	$\mathcal{N}(0.49, 0.2)$	4,3	N(0.84/0.63)			4,3	N(0.55/0.37)		
4	$\mathcal{N}(0.08, 0.04)$	$\mathcal{N}(0.19, 0.05)$								
General VCMM (Kmeans)										
Classification rate : 63.6% Normalised BIC : -7.10										
1	$sstd(6.18, 2.76, 3.28, 114.46)$	$snorm(2.15, 0.8, 2.16)$	3,2	SG(1.69/0.41)	4,2;3	C(0.27/0.12)	4,2	T(0.58,14.54/0.39)	1,2;4	R270J(-1.09/-0.05)
2	$\mathcal{N}(0.14, 0.02)$	$lnorm(-2.04, 0.18)$	4,1	F(4.03/0.39)	3,1;4	N(-0.01/-0.01)	4,1	SC(0.78/0.28)	3,1;4	R270C(-0.06/-0.03)
3	$logis(0.4, 0.1)$	$sstd(0.24, 0.19, 15.92, 244.39)$	4,3	SBB8(5.36,0.92/0.65)			4,3	SBB8(6,0.91/0.67)		
4	$\mathcal{N}(0.18, 0.06)$	$snorm(0.1, 0.06, 1.62)$								
General VCMM (Copula with single parameter) (Kmeans)										
Classification rate : 63.0% Normalised BIC : -7.10										
1	$sstd(6.56, 2.76, 3.29, 124.58)$	$snorm(2.22, 0.87, 2.32)$	3,2	SG(1.74/0.42)	4,2;3	SJ(1.19/0.1)	4,2	T(0.58,17.02/0.39)	1,2;4	R270J(-1.09/-0.05)
2	$\mathcal{N}(0.13, 0.02)$	$\Gamma(32.13, 243.51)$	4,1	F(3.83/0.37)	3,1;4	C(0.14/0.06)	4,1	SC(0.79/0.28)	3,1;4	R270C(-0.07/-0.04)
3	$snorm(0.42, 0.18, 1.3)$	$sstd(0.25, 0.2, 10.23, 317.63)$	4,3	SG(2.63/0.62)			4,3	F(12.2/0.72)		
4	$logis(0.19, 0.03)$	$snorm(0.1, 0.06, 1.63)$								
Gaussian VCMM (Kmeans)										
Classification rate : 68.5% Normalised BIC : -6.26										
1	$\mathcal{N}(5.8, 2.72)$	$\mathcal{N}(2.14, 0.77)$	3,1	N(0.2/0.13)	2,1;3	N(-0.26/-0.17)	4,2	N(0.59/0.4)	1,2;4	N(-0.07/-0.04)
2	$\mathcal{N}(0.14, 0.02)$	$\mathcal{N}(0.13, 0.02)$	3,2	N(0.58/0.39)	4,2;3	N(0.28/0.18)	4,1	N(0.39/0.26)	3,1;4	N(0.05/0.03)
3	$\mathcal{N}(0.43, 0.17)$	$\mathcal{N}(0.23, 0.2)$	4,3	N(0.81/0.6)			4,3	N(0.86/0.66)		
4	$\mathcal{N}(0.19, 0.05)$	$\mathcal{N}(0.1, 0.05)$								

	Marginal distributions		Vine tree structure							
			Cluster 1				Cluster 2			
Variable	Cluster 1	Cluster 2	Tree 1		Tree 2		Tree 1		Tree 2	
BCW										
General VCMM (Hierarchical)										
Classification rate : 82.2% Normalised BIC : -7.15										
1	$snorm(2.04, 0.74, 2.25)$	$lnorm(1.47, 0.45)$	4,1	BB8(1.59,0.87/0.15)	2,1;4	R90J(-1.08/-0.04)	2,1	R90C(-0.38/-0.16)	3,1;2	SC(0.19/0.09)
2	$\Gamma(37.47, 293.68)$	$\Gamma(38.33, 266.39)$	4,2	T(0.46,9.34/0.3)	3,2;4	F(-0.38/-0.04)	3,2	C(1.53/0.43)	4,2;3	SJ(1.1/0.05)
3	$sstd(0.19, 0.16, 7.49, 84.9)$	$sstd(0.46, 0.19, 9.25, 1.26)$	4,3	SBB8(5.71,0.9/0.65)			4,3	BB1(1.87,1.22/0.58)		
4	$snorm(0.08, 0.04, 1.22)$	$logis(0.19, 0.03)$								
General VCMM (Copula with single parameter) (Hierarchical)										
Classification rate : 74.8% Normalised BIC : -7.10										
1	$snorm(2.04, 0.72, 2)$	$sstd(5.49, 2.92, 3.31, 125.19)$	4,1	F(1.62/0.18)	2,1;4	R90J(-1.06/-0.03)	3,2	C(1.35/0.4)	4,2;3	SJ(1.21/0.11)
2	$lnorm(-2.06, 0.17)$	$\Gamma(41.71, 296.14)$	4,2	T(0.54,9.03/0.36)	3,2;4	T(-0.05,21.79/-0.03)	4,1	J(1.29/0.14)	3,1;4	F(-0.8/-0.09)
3	$sstd(0.21, 0.18, 7.98, 308.6)$	$snorm(0.45, 0.18, 1.21)$	4,3	F(11.19/0.69)			4,3	SG(2.42/0.59)		
4	$snorm(0.09, 0.05, 1.46)$	$logis(0.19, 0.03)$								
Gaussian VCMM (Hierarchical)										
Classification rate : 77.9% Normalised BIC : -6.61										
1	$\mathcal{N}(2.03, 0.71)$	$\mathcal{N}(4.77, 2.64)$	4,1	N(0.19/0.12)	2,1;4	N(-0.08/-0.05)	3,1	N(-0.15/-0.09)	4,1;3	N(0.31/0.2)
2	$\mathcal{N}(0.13, 0.02)$	$\mathcal{N}(0.14, 0.02)$	4,2	N(0.47/0.31)	3,2;4	N(0.01/0)	3,2	N(0.44/0.29)	4,2;3	N(0.27/0.17)
3	$\mathcal{N}(0.17, 0.11)$	$\mathcal{N}(0.5, 0.19)$	4,3	N(0.84/0.64)			4,3	N(0.52/0.35)		
4	$\mathcal{N}(0.08, 0.04)$	$\mathcal{N}(0.19, 0.05)$								
General VCMM (Kmeans scale)										
Classification rate : 85.1% Normalised BIC : -7.21										
1	$sstd(4.24, 2.52, 4.32, 3.07)$	$sstd(2.05, 0.87, 6.97, 2.55)$	3,2	F(1.85/0.2)	4,2;3	SJ(1.11/0.06)	4,1	SC(0.2/0.09)	2,1;4	R270C(-0.19/-0.09)
2	$llogis(13.05, 0.15)$	$\Gamma(42.24, 342.35)$	4,1	N(0.39/0.26)	3,1;4	F(-1.37/-0.15)	4,2	C(0.44/0.18)	3,2;4	T(-0.06,19.8/-0.04)
3	$lnorm(-0.8, 0.35)$	$snorm(0.14, 0.11, 351.6)$	4,3	F(4.27/0.41)			4,3	SBB8(4.22,0.95/0.59)		
4	$snorm(0.19, 0.04, 1.87)$	$snorm(0.07, 0.03, 0.85)$								
General VCMM (Copula with single parameter) (Kmeans scale)										
Classification rate : 83.9% Normalised BIC : -7.14										
1	$lnorm(1.29, 0.52)$	$sstd(2.04, 0.86, 6.36, 2.48)$	3,2	F(2.2/0.23)	4,2;3	R90J(-1.05/-0.03)	4,2	C(0.43/0.18)	1,2;4	R90C(-0.19/-0.09)
2	$llogis(12.78, 0.15)$	$\Gamma(43.58, 354.61)$	4,1	N(0.43/0.28)	3,1;4	F(-1.17/-0.13)	4,1	G(1.09/0.08)	3,1;4	N(-0.1/-0.07)
3	$lnorm(-0.84, 0.39)$	$snorm(0.15, 0.11, 178.44)$	4,3	F(5.19/0.47)			4,3	T(0.77,30/0.56)		
4	$snorm(0.18, 0.04, 1.55)$	$snorm(0.07, 0.03, 0.81)$								
Gaussian VCMM (Kmeans scale)										
Classification rate : 79.7% Normalised BIC : -6.00										
1	$\mathcal{N}(4.64, 2.66)$	$\mathcal{N}(2.05, 0.76)$	2,1	N(-0.31/-0.2)	3,1;2	N(-0.02/-0.01)	4,1	N(0.16/0.1)	2,1;4	N(-0.12/-0.08)
2	$\mathcal{N}(0.15, 0.02)$	$\mathcal{N}(0.13, 0.02)$	3,2	N(0.32/0.21)	4,2;3	N(0.17/0.11)	4,2	N(0.4/0.26)	3,2;4	N(-0.02/-0.01)
3	$\mathcal{N}(0.5, 0.19)$	$\mathcal{N}(0.17, 0.11)$	4,3	N(0.47/0.31)			4,3	N(0.83/0.62)		
4	$\mathcal{N}(0.19, 0.04)$	$\mathcal{N}(0.08, 0.04)$								
General VCMM (Hierarchical scale)										
Classification rate : 86.9% Normalised BIC : -7.24										
1	$snorm(1.99, 0.73, 2.37)$	$lnorm(1.36, 0.48)$	4,1	F(0.51/0.06)	2,1;4	R270C(-0.15/-0.07)	2,1	N(-0.28/-0.18)	3,1;2	SC(0.1/0.05)
2	$\Gamma(40.23, 322.64)$	$llogis(12.03, 0.14)$	4,2	SG(1.27/0.22)	3,2;4	F(-0.39/-0.04)	3,2	SG(1.42/0.29)	4,2;3	SJ(1.16/0.08)
3	$snorm(0.15, 0.12, 346.35)$	$sstd(0.48, 0.19, 7.12, 1.9)$	4,3	SBB8(4.22,0.96/0.6)			4,3	SG(1.77/0.44)		
4	$\mathcal{N}(0.07, 0.03)$	$sstd(0.19, 0.04, 9.19, 1.46)$								
General VCMM (Copula with single parameter) (Hierarchical scale)										
Classification rate : 87.8% Normalised BIC : -7.08										
1	$snorm(2.01, 0.74, 2.4)$	$sstd(4.45, 2.5, 4.36, 2.79)$	4,1	F(0.59/0.07)	2,1;4	R270C(-0.19/-0.08)	2,1	N(-0.3/-0.19)	3,1;2	SC(0.13/0.06)
2	$\Gamma(39.12, 313.49)$	$llogis(12.5, 0.14)$	4,2	SG(1.27/0.22)	3,2;4	R270G(-1.05/-0.04)	3,2	SG(1.44/0.31)	4,2;3	SJ(1.11/0.06)
3	$snorm(0.16, 0.12, 190.82)$	$sstd(0.48, 0.19, 7.18, 1.94)$	4,3	T(0.77,30/0.56)			4,3	SG(1.82/0.45)		
4	$\mathcal{N}(0.07, 0.03)$	$sstd(0.19, 0.04, 9.8, 1.41)$								
Gaussian VCMM (Hierarchical scale)										
Classification rate : 78.4% Normalised BIC : -6.61										
1	$\mathcal{N}(2.03, 0.73)$	$\mathcal{N}(4.73, 2.64)$	4,2	N(0.44/0.29)	1,2;4	N(-0.11/-0.07)	2,1	N(-0.27/-0.17)	3,1;2	N(-0.04/-0.02)
2	$\mathcal{N}(0.13, 0.02)$	$\mathcal{N}(0.15, 0.02)$	4,1	N(0.17/0.11)	3,1;4	N(-0.03/-0.02)	3,2	N(0.41/0.27)	4,2;3	N(0.25/0.16)
3	$\mathcal{N}(0.17, 0.11)$	$\mathcal{N}(0.5, 0.19)$	4,3	N(0.83/0.63)			4,3	N(0.51/0.34)		
4	$\mathcal{N}(0.08, 0.04)$	$\mathcal{N}(0.19, 0.05)$								

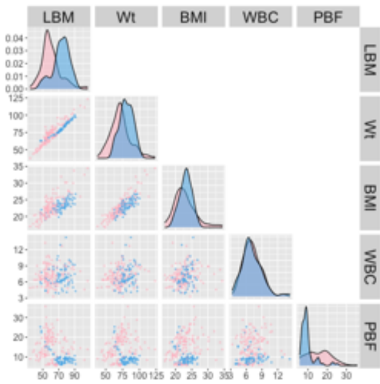
C.3 Pair plots of AIS data with initialization and final clustering



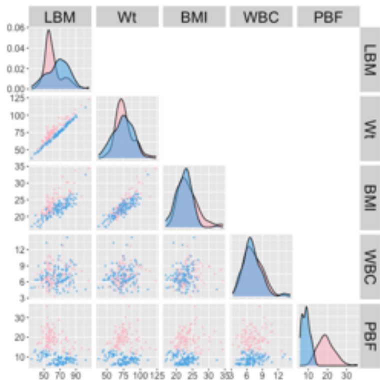
Initializing clustering
Random (50.2%)



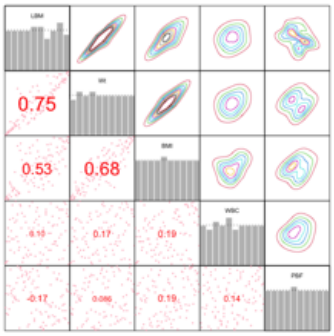
Final clustering
(General VCMM)
Random (60.2%)



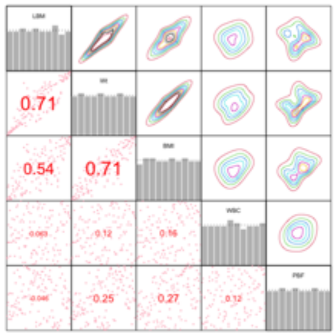
Final clustering
(General VCMM (Copula with single parameter))
Random (61.6%)



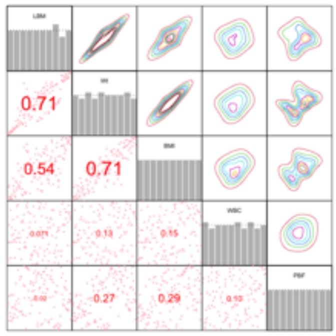
Final clustering
(Gaussian VCMM)
Random (65.8%)



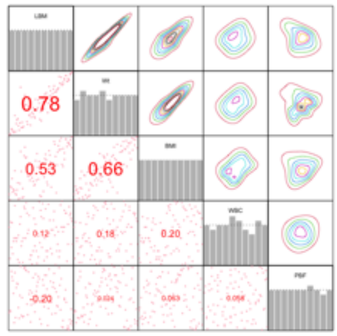
Copula data for cluster 1 after initializing clustering



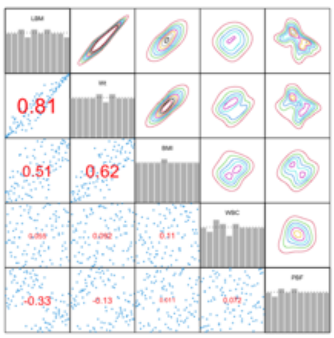
Copula data for cluster 1 after final clustering
(General VCMM)



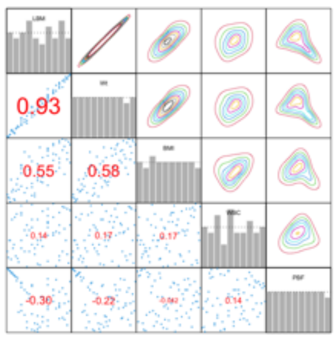
Copula data for cluster 1 after final clustering
(General VCMM (Copula with single parameter))



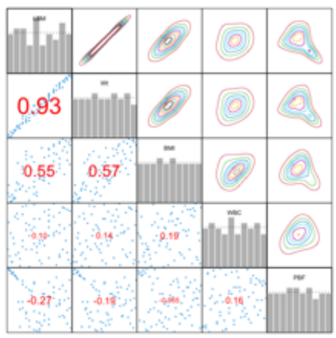
Copula data for cluster 1 after final clustering
(Gaussian VCMM)



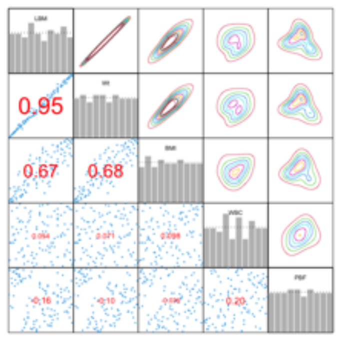
Copula data for cluster 2 after initializing clustering



Copula data for cluster 2 after final clustering
(General VCMM)

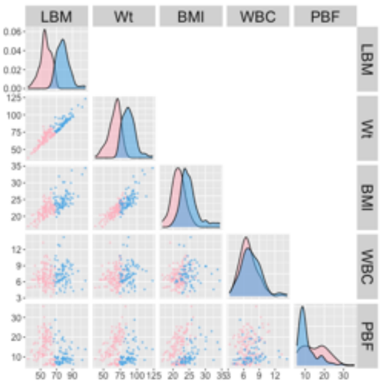


Copula data for cluster 2 after final clustering
(General VCMM (Copula with single parameter))

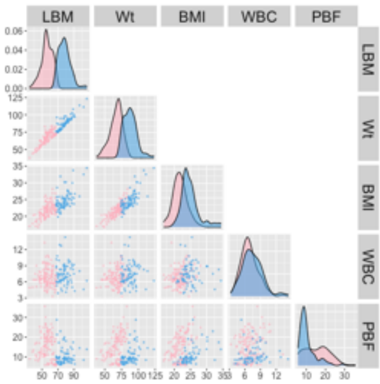


Copula data for cluster 2 after final clustering
(Gaussian VCMM)

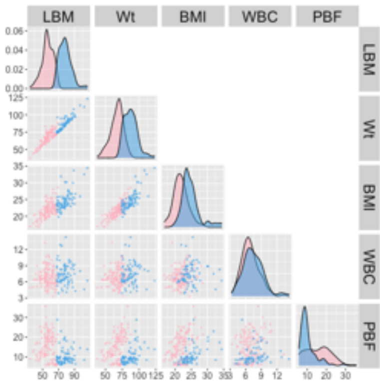
Figure C.1: The pairwise scatter plot of the subset of AIS data and its u data for each cluster, after initializing clustering with Random (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Random and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.



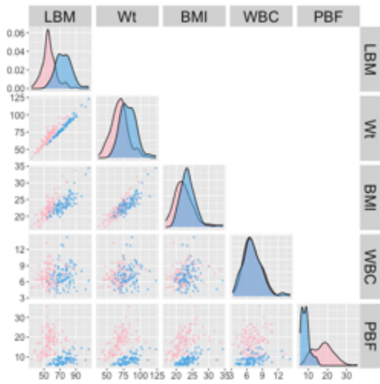
Initializing clustering Kmeans (71.9%)



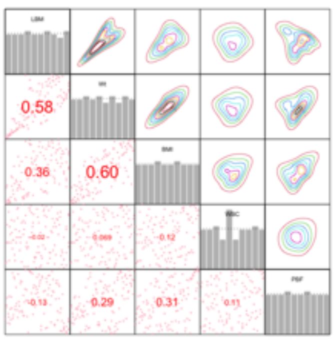
Final clustering (General VCMM) Kmeans (77.5%)



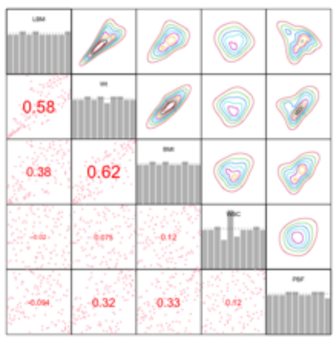
Final clustering (General VCMM (Copula with single parameter)) Kmeans (76.7%)



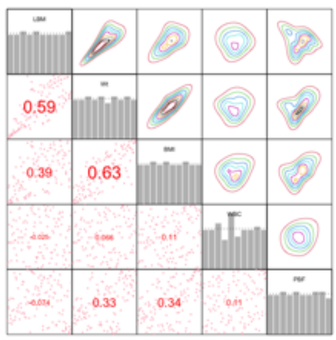
Final clustering (Gaussian VCMM) Kmeans (91.4%)



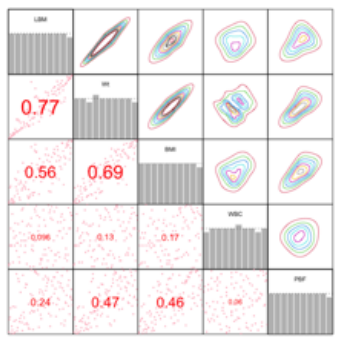
Copula data for cluster 1 after initializing clustering



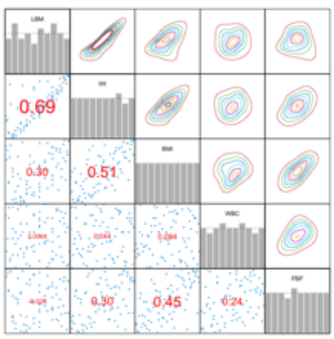
Copula data for cluster 1 after final clustering (General VCMM)



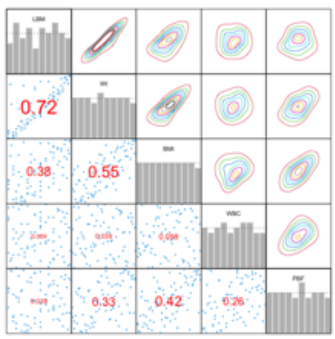
Copula data for cluster 1 after final clustering (General VCMM (Copula with single parameter))



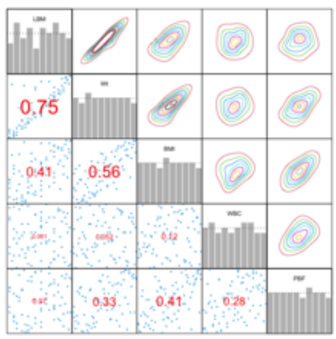
Copula data for cluster 1 after final clustering (Gaussian VCMM)



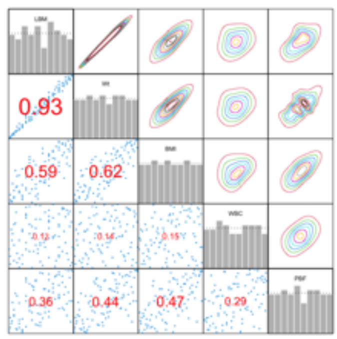
Copula data for cluster 2 after initializing clustering



Copula data for cluster 2 after final clustering (General VCMM)



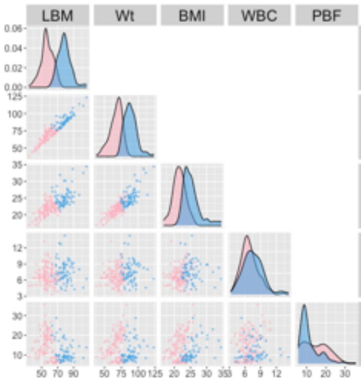
Copula data for cluster 2 after final clustering (General VCMM (Copula with single parameter))



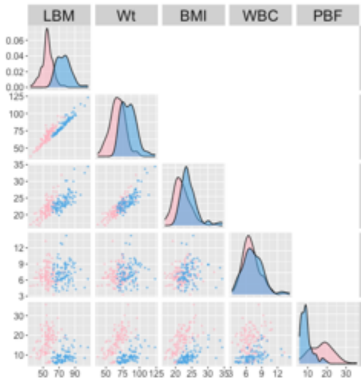
Copula data for cluster 2 after final clustering (Gaussian VCMM)

Figure C.2: The pairwise scatter plot of the subset of AIS data and its u data for each cluster, after initializing clustering with Kmeans (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Kmeans and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.

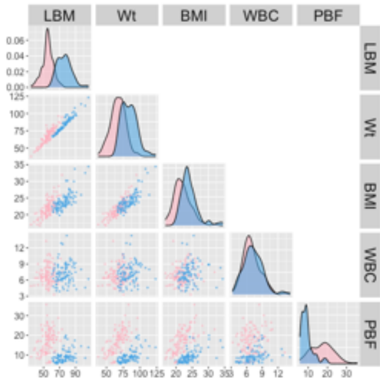
[!htbp]



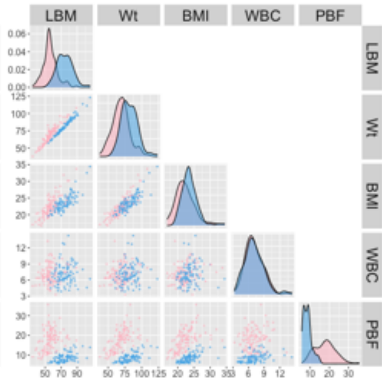
Initializing clustering
Kmeans scale (68.1%)



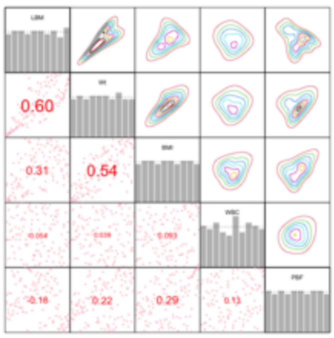
Final clustering
(General VCMM)
Kmeans scale (91.4%)



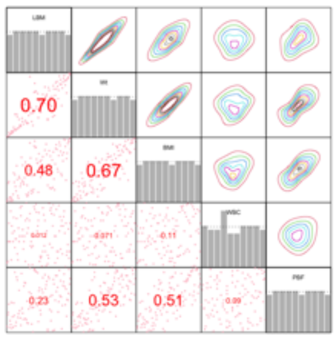
Final clustering
(General VCMM (Copula with single parameter))
Kmeans scale (91.4%)



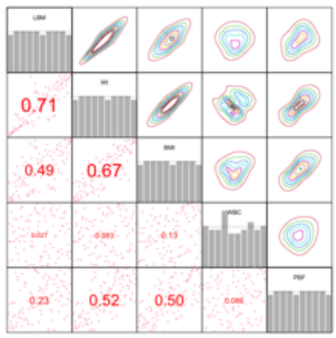
Final clustering
(Gaussian VCMM)
Kmeans scale (90.5%)



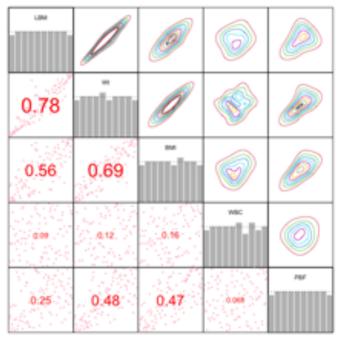
Copula data for cluster 1 after
initializing clustering



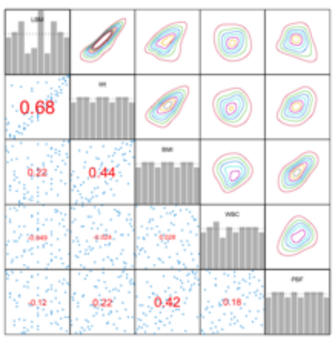
Copula data for cluster 1 after
final clustering
(General VCMM)



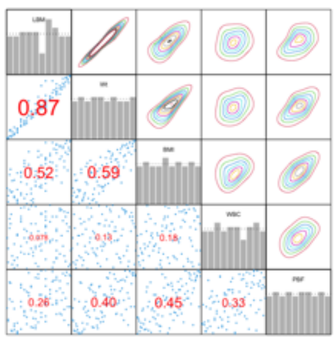
Copula data for cluster 1 after
final clustering
(General VCMM (Copula with single parameter))



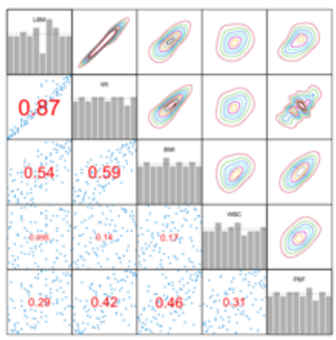
Copula data for cluster 1 after
final clustering
(Gaussian VCMM)



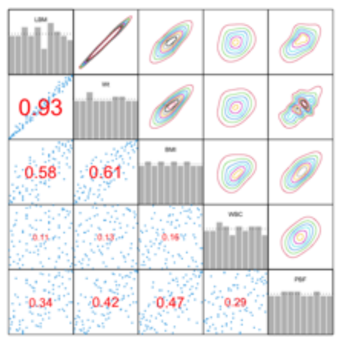
Copula data for cluster 2 after
initializing clustering



Copula data for cluster 2 after
final clustering
(General VCMM)

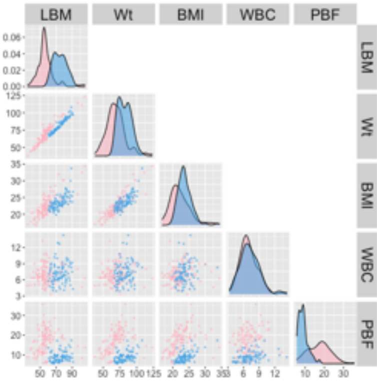


Copula data for cluster 2 after
final clustering
(General VCMM (Copula with single parameter))

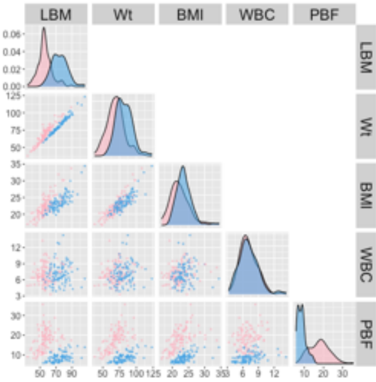


Copula data for cluster 2 after
final clustering
(Gaussian VCMM)

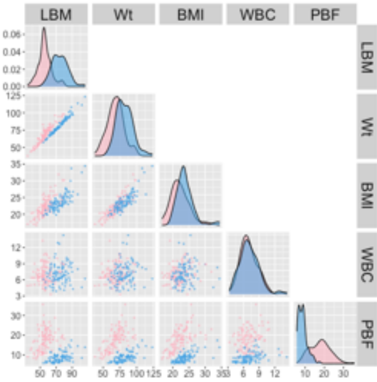
Figure C.3: The pairwise scatter plot of the subset of AIS data and its u data for each cluster, after initializing clustering with Kmeans scale (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Kmeans scale and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.



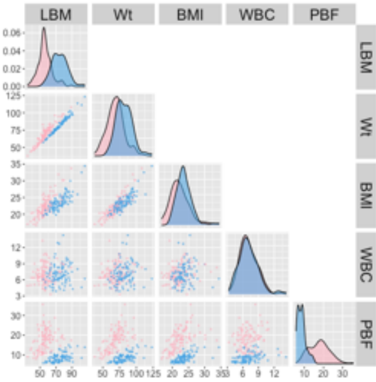
**Initializing clustering
Hierarchical (85.3%)**



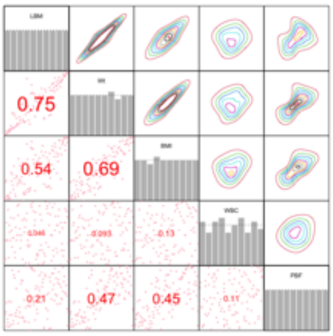
**Final clustering
(General VCMM)
Hierarchical (91.4%)**



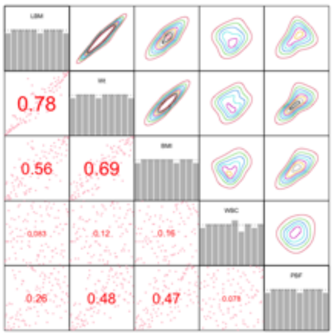
**Final clustering
(General VCMM (Copula with single parameter))
Hierarchical (92.4%)**



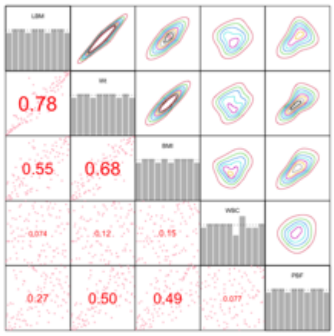
**Final clustering
(Gaussian VCMM)
Hierarchical (90.5%)**



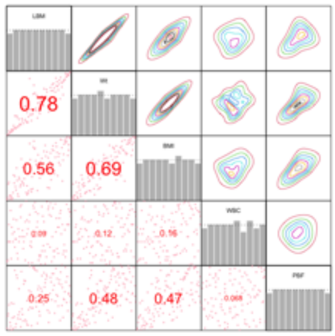
**Copula data for cluster 1 after
initializing clustering**



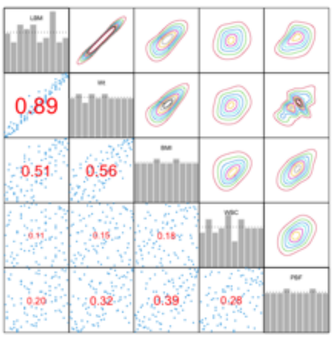
**Copula data for cluster 1 after
final clustering
(General VCMM)**



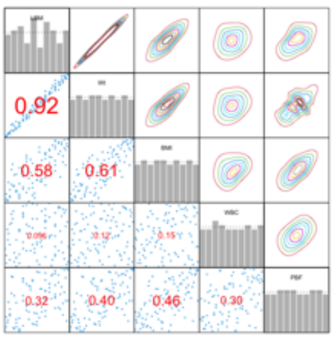
**Copula data for cluster 1 after
final clustering
(General VCMM (Copula with single parameter))**



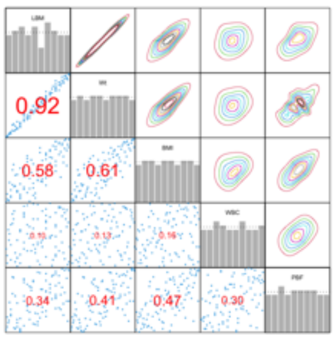
**Copula data for cluster 1 after
final clustering
(Gaussian VCMM)**



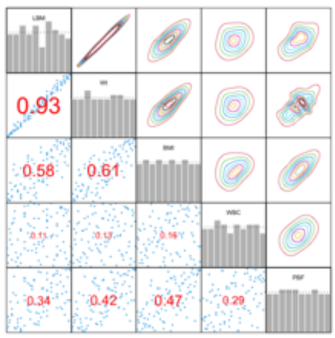
**Copula data for cluster 2 after
initializing clustering**



**Copula data for cluster 2 after
final clustering
(General VCMM)**

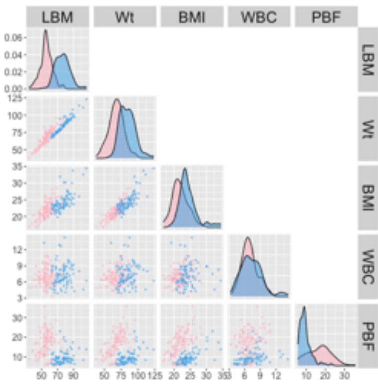


**Copula data for cluster 2 after
final clustering
(General VCMM (Copula with single parameter))**

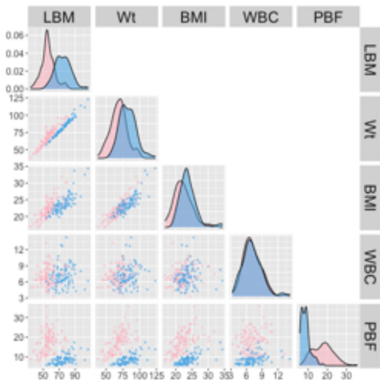


**Copula data for cluster 2 after
final clustering
(Gaussian VCMM)**

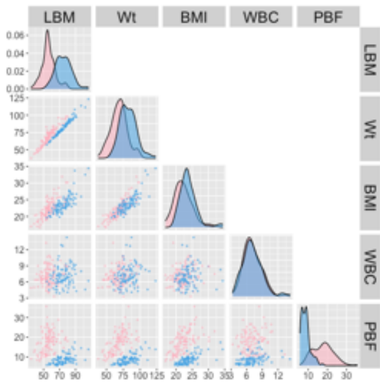
Figure C.4: The pairwise scatter plot of the subset of AIS data and its u data for each cluster, after initializing clustering with Hierarchical (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Hierarchical and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.



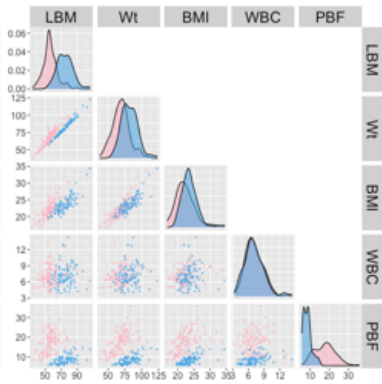
Initializing clustering
Hierarchical scale (81.3%)



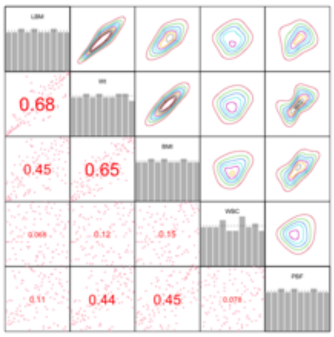
Final clustering
(General VCMM)
Hierarchical scale (92.4%)



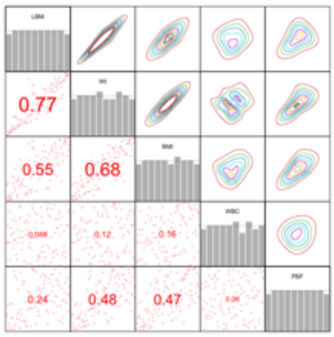
Final clustering
(General VCMM (Copula with single parameter))
Hierarchical scale (92.4%)



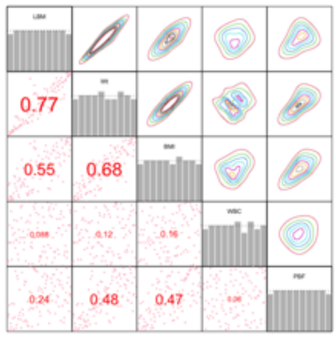
Final clustering
(Gaussian VCMM)
Hierarchical scale (91.4%)



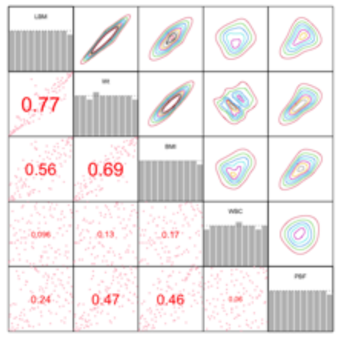
Copula data for cluster 1 after initializing clustering



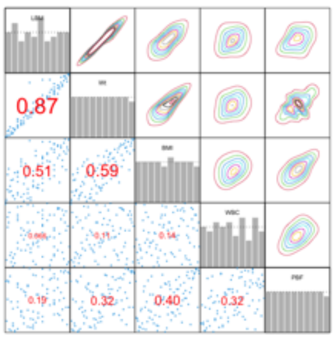
Copula data for cluster 1 after final clustering
(General VCMM)



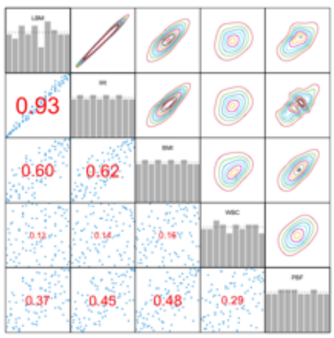
Copula data for cluster 1 after final clustering
(General VCMM (Copula with single parameter))



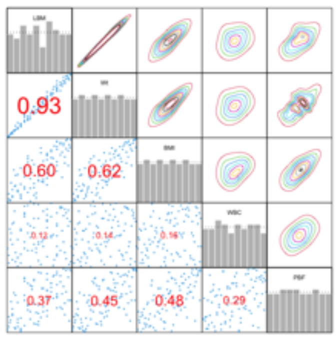
Copula data for cluster 1 after final clustering
(Gaussian VCMM)



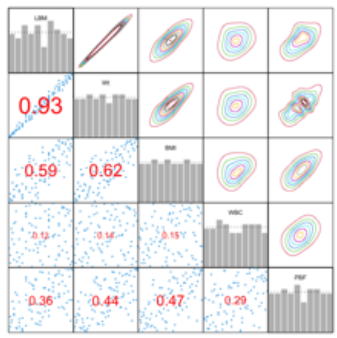
Copula data for cluster 2 after initializing clustering



Copula data for cluster 2 after final clustering
(General VCMM)



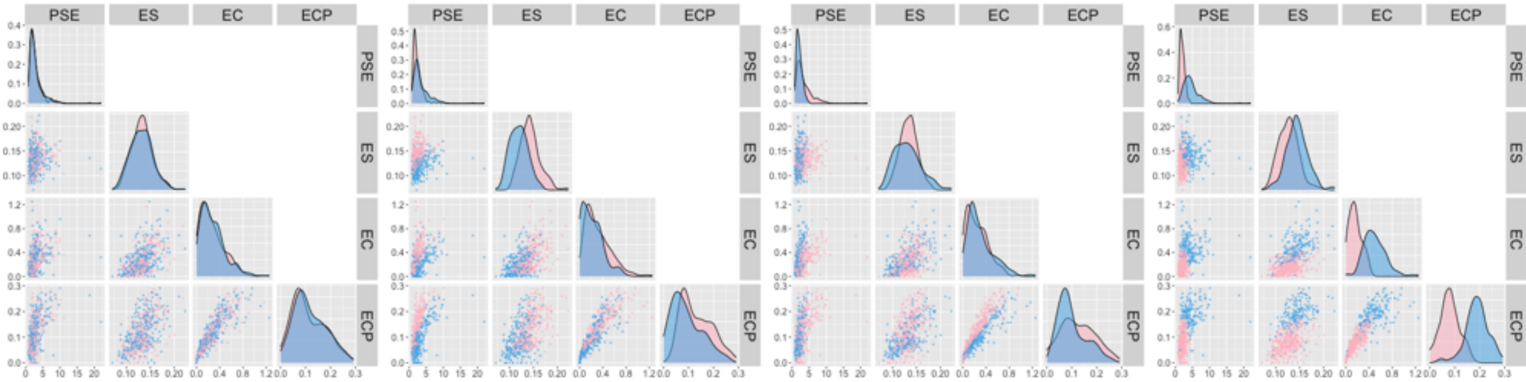
Copula data for cluster 2 after final clustering
(General VCMM (Copula with single parameter))



Copula data for cluster 2 after final clustering
(Gaussian VCMM)

Figure C.5: The pairwise scatter plot of the subset of AIS data and its u data for each cluster, after initializing clustering with Hierarchical scale (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Hierarchical scale and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.

C.4 Pair plots of BCW data with initialization and final clustering

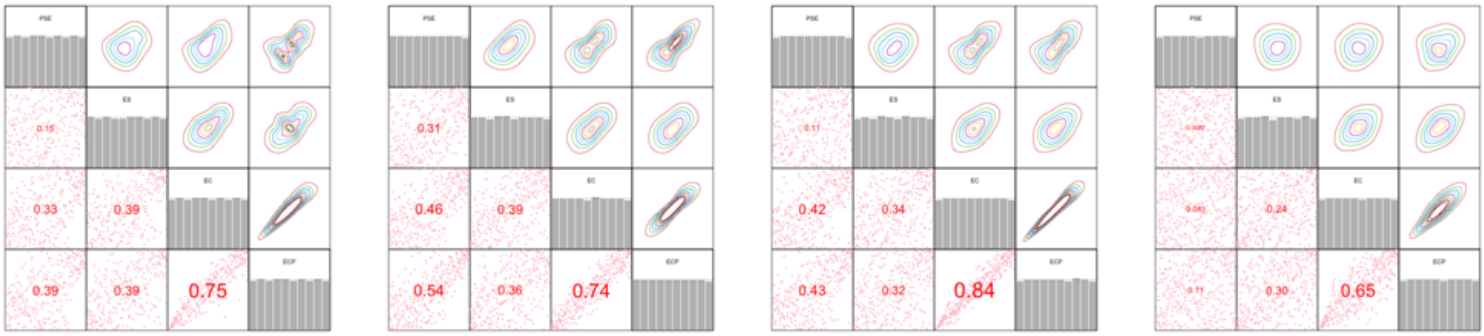


Initializing clustering Random (51.1%)

Final clustering (General VCMM) Random (50.4%)

Final clustering (General VCMM (Copula with single parameter)) Random (51.1%)

Final clustering (Gaussian VCMM) Random (78.4%)

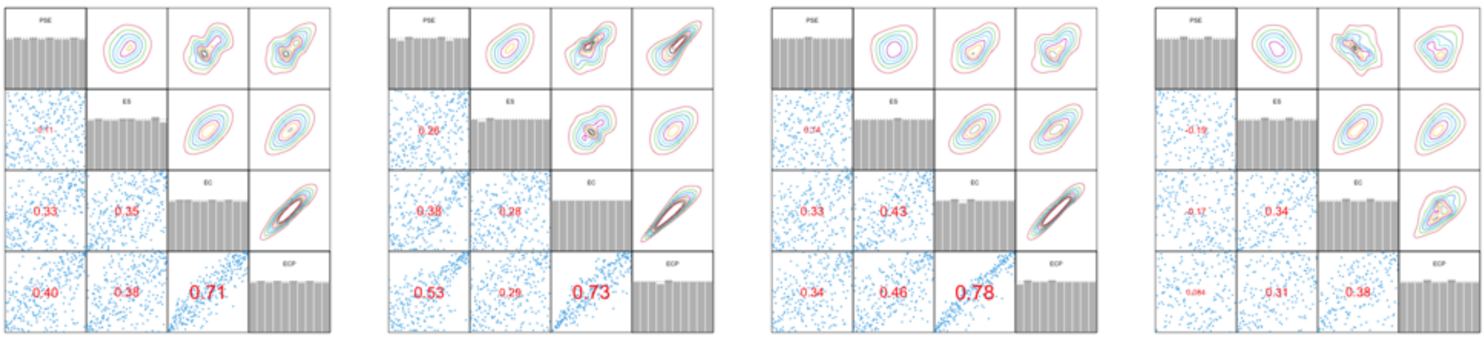


Copula data for cluster 1 after initializing clustering

Copula data for cluster 1 after final clustering (General VCMM)

Copula data for cluster 1 after final clustering (General VCMM (Copula with single parameter))

Copula data for cluster 1 after final clustering (Gaussian VCMM)



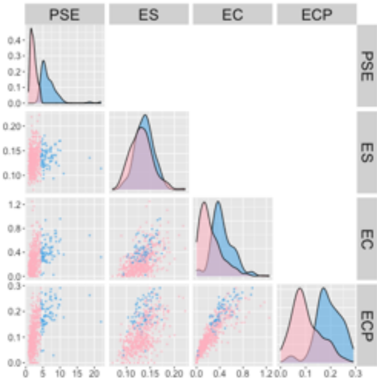
Copula data for cluster 2 after initializing clustering

Copula data for cluster 2 after final clustering (General VCMM)

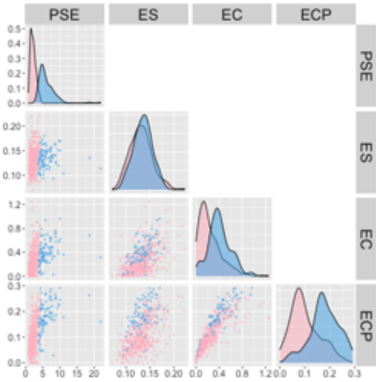
Copula data for cluster 2 after final clustering (General VCMM (Copula with single parameter))

Copula data for cluster 2 after final clustering (Gaussian VCMM)

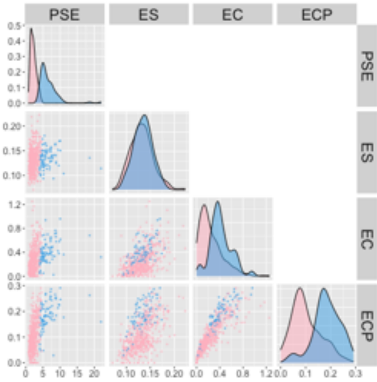
Figure C.6: The pairwise scatter plot of the subset of Breast Cancer Wisconsin Diagnostic data and its u data for each cluster, after initializing clustering with Random (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Random and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.



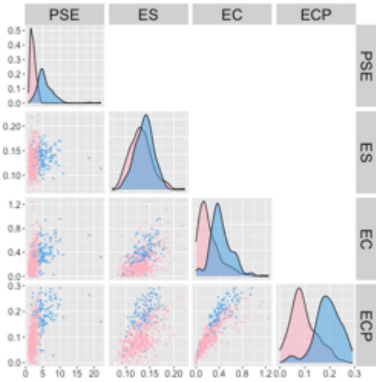
Initializing clustering
Kmeans (62.8%)



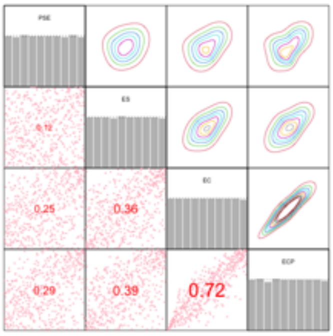
Final clustering
(General VCMM)
Kmeans (63.6%)



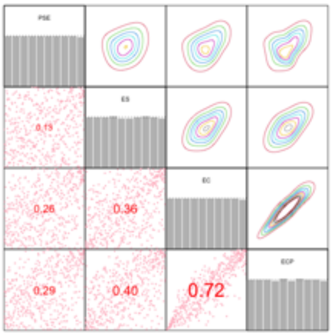
Final clustering
(General VCMM (Copula with single parameter))
Kmeans (63.0%)



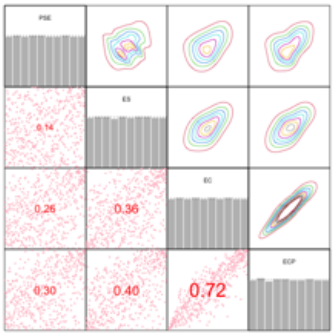
Final clustering
(Gaussian VCMM)
Kmeans (68.5%)



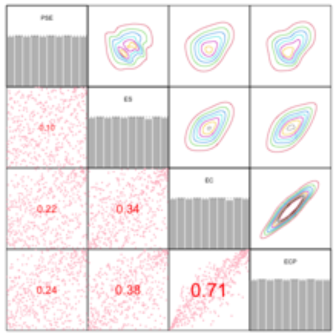
Copula data for cluster 1 after initializing clustering



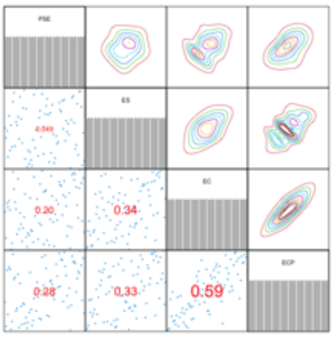
Copula data for cluster 1 after final clustering
(General VCMM)



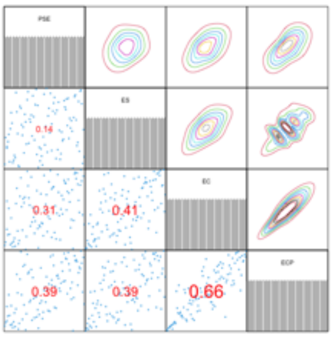
Copula data for cluster 1 after final clustering
(General VCMM (Copula with single parameter))



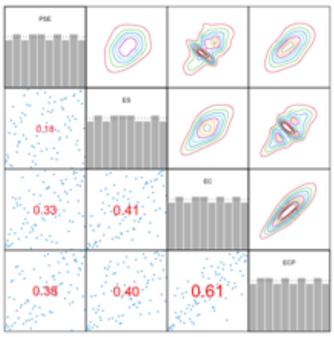
Copula data for cluster 1 after final clustering
(Gaussian VCMM)



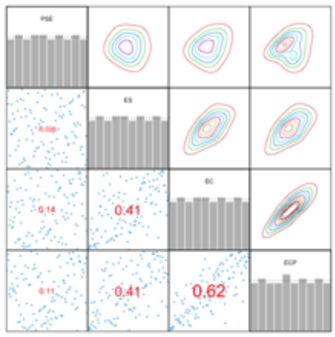
Copula data for cluster 2 after initializing clustering



Copula data for cluster 2 after final clustering
(General VCMM)

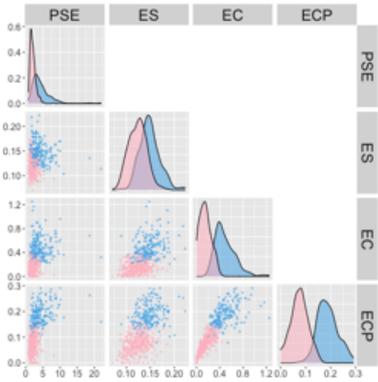


Copula data for cluster 2 after final clustering
(General VCMM (Copula with single parameter))

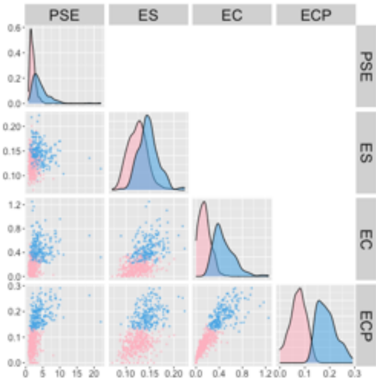


Copula data for cluster 2 after final clustering
(Gaussian VCMM)

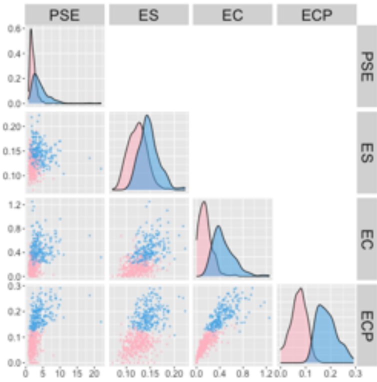
Figure C.7: The pairwise scatter plot of the subset of Breast Cancer Wisconsin Diagnostic data and its u data for each cluster, after initializing clustering with Kmeans (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Kmeans and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.



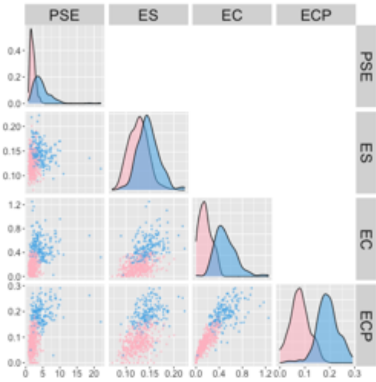
Initializing clustering
Kmeans scale (82.2%)



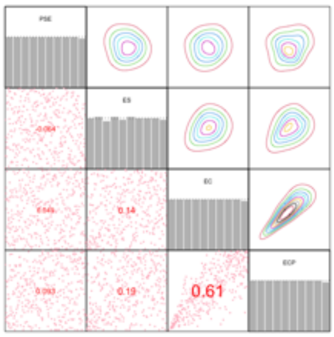
Final clustering
(General VCMM)
Kmeans scale (85.1%)



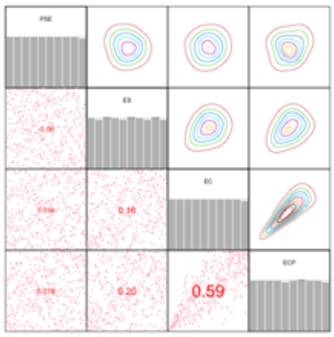
Final clustering
(General VCMM (Copula with single parameter))
Kmeans scale (83.9%)



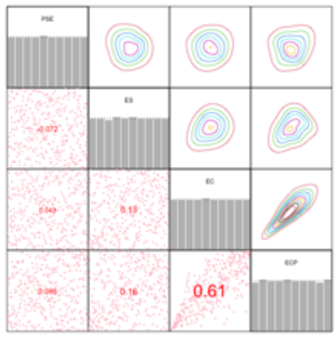
Final clustering
(Gaussian VCMM)
Kmeans scale (79.7%)



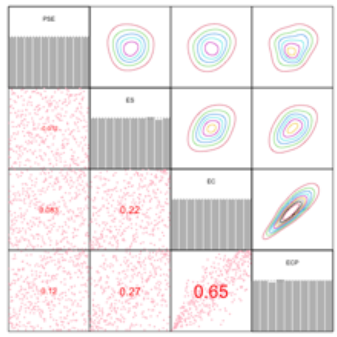
Copula data for cluster 1 after
initializing clustering



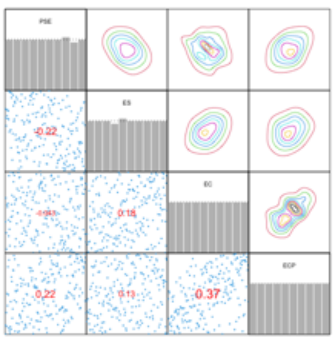
Copula data for cluster 1 after
final clustering
(General VCMM)



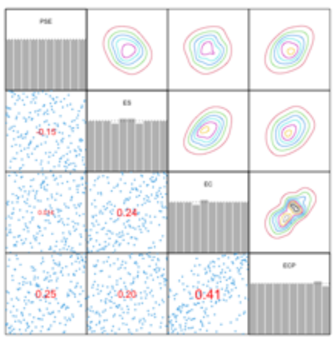
Copula data for cluster 1 after
final clustering
(General VCMM (Copula with single parameter))



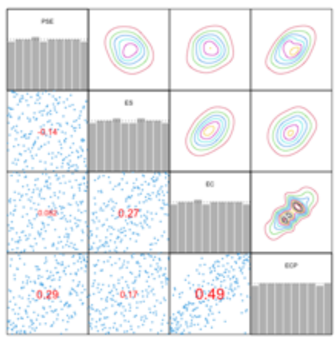
Copula data for cluster 1 after
final clustering
(Gaussian VCMM)



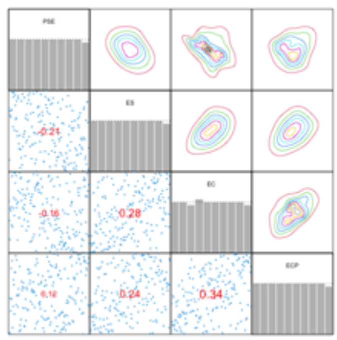
Copula data for cluster 2 after
initializing clustering



Copula data for cluster 2 after
final clustering
(General VCMM)

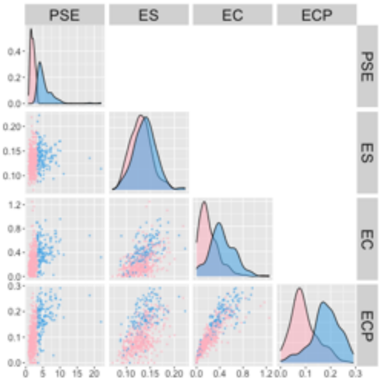


Copula data for cluster 2 after
final clustering
(General VCMM (Copula with single parameter))

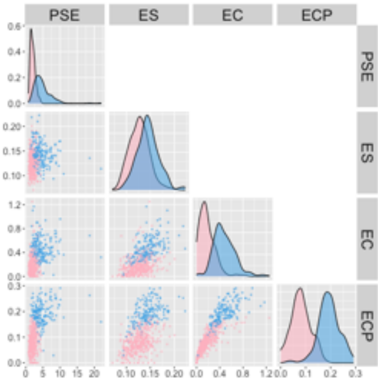


Copula data for cluster 2 after
final clustering
(Gaussian VCMM)

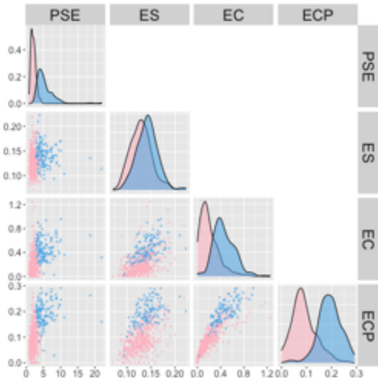
Figure C.8: The pairwise scatter plot of the subset of Breast Cancer Wisconsin Diagnostic data and its u data for each cluster, after initializing clustering with Kmeans scale (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Kmeans scale and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.



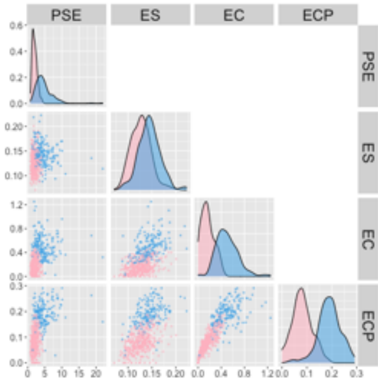
Initializing clustering
Hierarchical (68.5%)



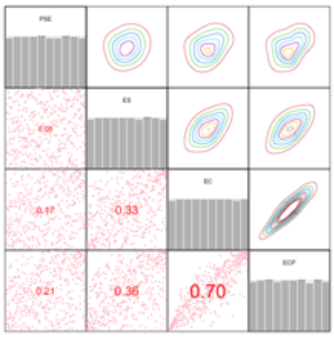
Final clustering
(General VCMM)
Hierarchical (82.2%)



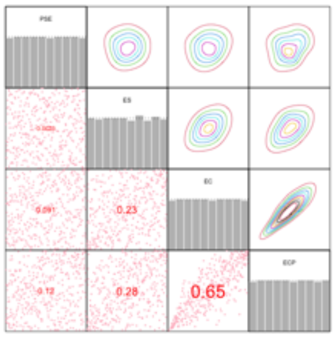
Final clustering
(General VCMM (Copula with single parameter))
Hierarchical (74.8%)



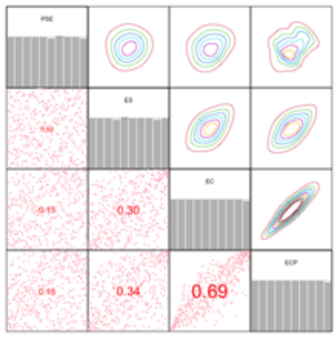
Final clustering
(Gaussian VCMM)
Hierarchical (77.9%)



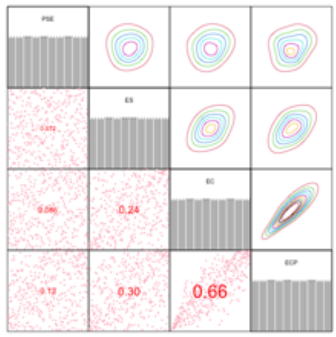
Copula data for cluster 1 after initializing clustering



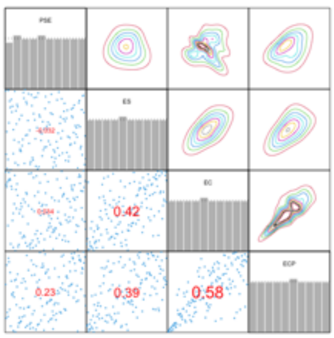
Copula data for cluster 1 after final clustering
(General VCMM)



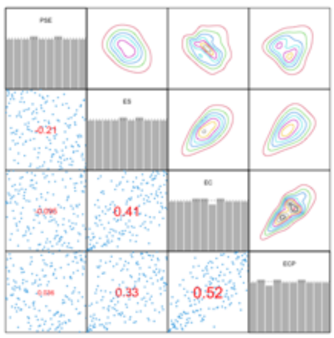
Copula data for cluster 1 after final clustering
(General VCMM (Copula with single parameter))



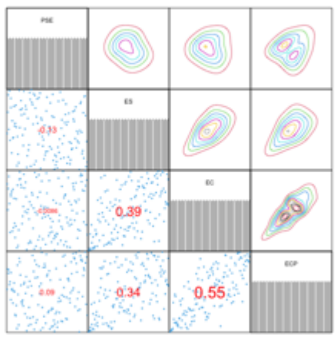
Copula data for cluster 1 after final clustering
(Gaussian VCMM)



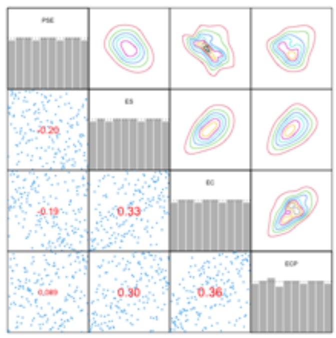
Copula data for cluster 2 after initializing clustering



Copula data for cluster 2 after final clustering
(General VCMM)

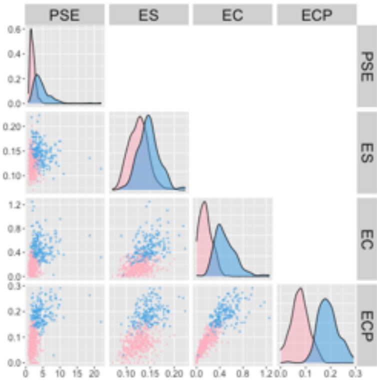


Copula data for cluster 2 after final clustering
(General VCMM (Copula with single parameter))

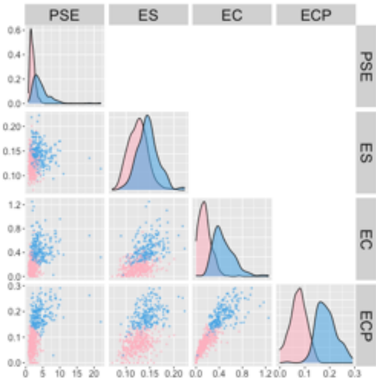


Copula data for cluster 2 after final clustering
(Gaussian VCMM)

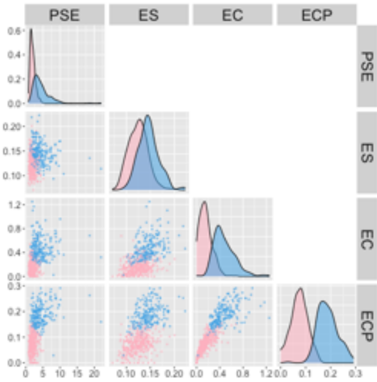
Figure C.9: The pairwise scatter plot of the subset of Breast Cancer Wisconsin Diagnostic data and its u data for each cluster, after initializing clustering with Hierarchical (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Hierarchical and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.



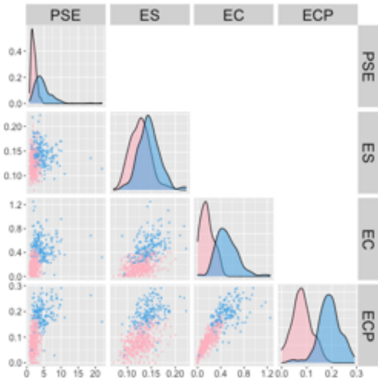
Initializing clustering
Hierarchical scale (84.5%)



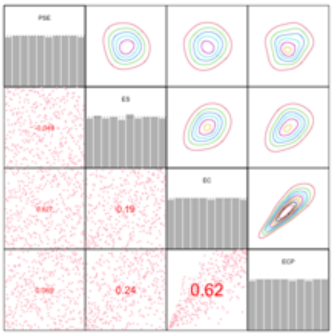
Final clustering
(General VCMM)
Hierarchical scale (86.9%)



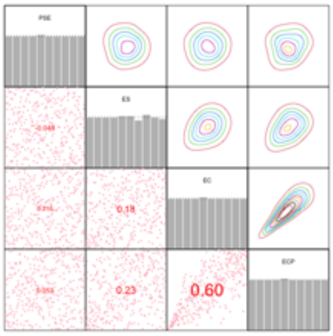
Final clustering
(General VCMM (Copula with single parameter))
Hierarchical scale (87.8%)



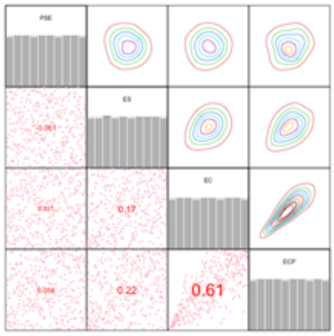
Final clustering
(Gaussian VCMM)
Hierarchical scale (78.4%)



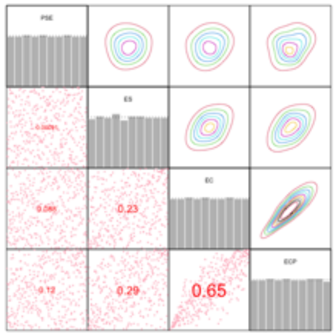
Copula data for cluster 1 after initializing clustering



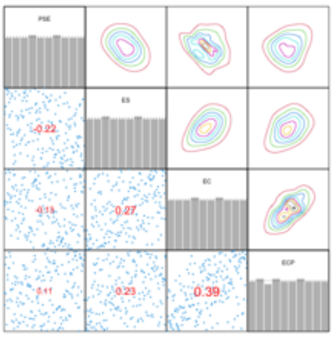
Copula data for cluster 1 after final clustering
(General VCMM)



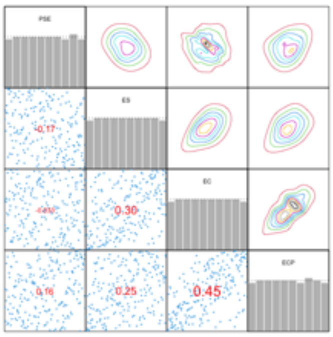
Copula data for cluster 1 after final clustering
(General VCMM (Copula with single parameter))



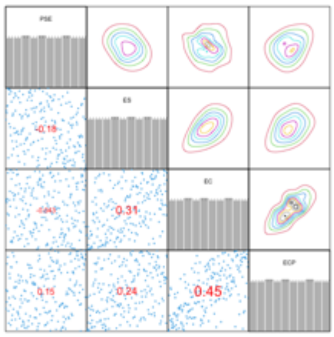
Copula data for cluster 1 after final clustering
(Gaussian VCMM)



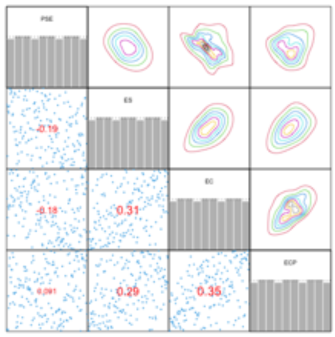
Copula data for cluster 2 after initializing clustering



Copula data for cluster 2 after final clustering
(General VCMM)



Copula data for cluster 2 after final clustering
(General VCMM (Copula with single parameter))



Copula data for cluster 2 after final clustering
(Gaussian VCMM)

Figure C.10: The pairwise scatter plot of the subset of Breast Cancer Wisconsin Diagnostic data and its u data for each cluster, after initializing clustering with Hierarchical scale (First column), final clustering by general VCMM (Second column), General VCMM (Copula with single parameter) (Third column) and Gaussian VCMM (Fourth column) with Hierarchical scale and the EM algorithm reaching the highest classification rate. Different colours refer to the points from different clusters. The number inside the round bracket is the classification rate, compared to the true cluster.

References

J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49(3):803–821, 1993. doi: https://doi.org/10.2307/2532201.

T. Bedford and R. M. Cooke. Vines : a new graphical model for dependent random variables. *Annals of Statistics*, 30(4):1031–1068, 2002. doi: https://doi.org/10.1214/aos/1031689016.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006. URL `https://www.springer.com/gp/book/9780387310732#aboutAuthors`.

C. Czado. *Analyzing Dependent Data with Vine Copulas - A Practical Guide With R*. Springer International Publishing, 2019. doi: https://doi.org/10.1007/978-3-030-13785-4.

U. J. Dang, A. Punzo, S. I. Paul D. McNicholas, and R. P. Browne. Multivariate response and parsimony for gaussian cluster-weighted models. *Journal of Classification volume*, 34:4–34, 2017. URL `https://link.springer.com/article/10.1007/s00357-017-9221-2`.

B. A. Desmarais and J. J. Harden. Testing for zero inflation in count models: Bias correction for the vuong test. *Stata Journal*, 13(4):810–835, 2013. doi: https://doi.org/10.1177/1536867X1301300408.

K. Ellis, S. Godbole, S. Marshall, G. Lanckriet, J. Staudenmayer, and J. Kerr. Identifying active travel behaviors in challenging environments using gps, accelerometers, and machine learning algorithms. *Frontiers in Public Health*, 2(36), 2014. doi: https://doi.org/10.3389/fpubh.2014.00036.

C. Fraley. Algorithms for model-based gaussian hierarchical clustering. *SIAM Journal on Scientific Computing*, 57(1):270–281, 1998. doi: https://doi.org/10.1137/S1064827596311451.

L. A. García-Escudero, A. Gordaliza, C. Matrán, and A. Mayo-Iscar. Avoiding spurious local maximizers in mixture modeling. *Statistics and Computing*, 25:619–633, 2015. doi: https://doi.org/10.1007/s11222-014-9455-3.

J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979. doi: https://doi.org/10.2307/2346830.

N. Heal. Agile, hackathons, and the danger of local maxima, 2020. URL `https://medium.com/swlh/agile-hackathons-and-the-danger-of-local-maxima-4c2255e8f63b`.

C. Jin, Y. Zhang, S. Balakrishnan, M. J. Wainwright, and M. Jordan. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. *Neural Information Processing Systems (NIPS)*, 2016. URL `https://arxiv.org/abs/1609.00978`.

C. Liu and D. B. Rubin. The ecme algorithm: A simple extension of em and ecm with faster monotone convergence. *Biometrika*, 81(4):633–648, 1994. doi: https://doi.org/10.2307/2337067.

G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions, Second Edition*. Wiley, 2007. doi: https://doi.org/10.1002/9780470191613.

X.-L. Meng and D. B. Rubin. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2):267–278, 1993. doi: https://doi.org/10.2307/2337198.

K. P. Murphy. *Machine Learning : A Probabilistic Perspective*. The MIT Press, 2012.

A. Punzo and P. D. McNicholas. Parsimonious mixtures of multivariate contaminated normal distributions. *Biometrical Journal*, 58(6):1506–1537, 2016. doi: https://doi.org/10.1002/bimj.201500144.

W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. doi: https://doi.org/10.2307/2284239.

Ö. Sahin and C. Czado. Vine copula mixture models and clustering for non-gaussian data. *Econometrics and Statistics*, 2021. ISSN 2452-3062. doi: https://doi.org/10.1016/j.ecosta.2021.08.011. URL `https://www.sciencedirect.com/science/article/pii/S2452306221001052`.

L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery. mclust 5: Clustering, classification and density estimation using gaussian finite mixture models. *The R Journal*, 8(1):205–233, 2016. doi: https://doi.org/10.32614/RJ-2016-021.

E. Shireman, D. Steinley, and M. J. Brusco. Examining the effect of initialization strategies on the performance of gaussian mixture modeling. *Behavior Research Methods volume*, 49(1):282–293, 2017. doi: https://doi.org/10.3758/s13428-015-0697-6.

A. Sklar. Fonctions de répartition à n dimensions et leurs marges. *Publications de l'Institut Statistique de l'Universit e de Paris*, 8:229–231, 1959.

D. A. van Dyk and X.-L. Meng. On the orderings and groupings of conditional maximizations within ecm-type algorithms. *Journal of Computational and Graphical Statistics*, 6(2):202–223, 1997. doi: https://doi.org/10.2307/1390931.

Q. H. Vuong. Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica*, 57(2):307–333, 1989. doi: https://doi.org/10.2307/1912557.