ROBUST MIXTURE REGRESSION MODELING WITH PEARSON TYPE VII DISTRIBUTION

by

JINGYI ZHANG

M.S., Fudan University, China, 2011
B.S., Wuhan University, China, 2008

A REPORT

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Statistics
College of Art and Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2013

Approved by:

Major Professor
Weixing Song

Copyright

Abstract

A robust estimation procedure for parametric regression models is proposed in the paper by assuming the error terms follow a Pearson type VII distribution. The estimation procedure is implemented by an EM algorithm based on the fact that the Pearson type VII distributions are a scale mixture of a normal distribution and a Gamma distribution. A trimmed version of proposed procedure is also discussed in this paper, which can successfully trim the high leverage points away from the data. Finite sample performance of the proposed algorithm is evaluated by some extensive simulation studies, together with the comparisons made with other existing procedures in the literature.

Table of Contents

Table of Contents iv

List of Tables v

Acknowledgements vi

1 Introduction 1

2 Mixture Regression Model 2
- 2.1 EM Algorithm for Mixture Model 2
- 2.2 Mixture Linear Regression Model 4

3 Mixture of Pearson type VII distribution 6

4 Robust Mixture Regression Model with Pearson type VII distribution 11

5 Simulation Study 16

6 Conclusion 23

Bibliography 25

A R code 26

List of Tables

5.1 MSE(Bias) of Point Estimates for $n = 100$ 19
5.2 MSE(Bias) of Point Estimates for $n = 200$ 20
5.3 MSE(Bias) of Point Estimates for $n = 400$ 21
5.4 The mean(median) of estimated degree of freedom by MixregP and MixregP-MCD. 22

Acknowledgments

I would first like to thank my advisor, Dr. Weixing Song, without whose guidance and help I could not be able to complete this report. I would also like to thank Dr. Weixin Yao, since his research gives me the original inspiration and I also benefit a lot from the course of mixture model which is offered by him. I also appreciate Dr. Nelson's suggestions not only for my master's report, but also for my future career.

Special thanks to Dr. Weixin Yao for generously sharing his R code with me.

Chapter 1

Introduction

Mixture models have been applied in various fields including biology, medicine, economics, agriculture, animal sciences and so on. One of the first major work involving mixture model was undertaken by Karl Pearson in his research about crab morphometry over one hundred years ago. In his research, Pearson (1894) fitted the data collected from two different species of crab by mixing two normal distributions with different parameters and proportions. Although Pearson's approach based on method of moments successfully demonstrated the flexibility of mixture models, it was a significant computational challenge to get the solution of a 9th degree polynomial at that time. The advent of modern computer and the method of maximum likelihood has considerable increased the application of mixture models. In a classic paper by Dempster, Laird and Rubin (1977), EM algorithm began to be applied in fitting mixture model based on maximum likelihood estimation.

Since the traditional normal mixture model is sensitive to outliers, Wei (2012) proposed a robust estimation method for mixture regression model based on t-distribution by extending the research from Peel and McLachlan (2000). In this paper, we will further extending Wei (2012)'s work by substituting Pearson type VII distribution for t-distribution, based on the results from Sun and Garibaldi (2010), in which the robust mixture modeling with Pearson type VII distribution is proposed.

Chapter 2

Mixture Regression Model

We let $y_1, y_2, ..., y_n$ denote a random sample of size n from a g-component mixture model with some unknown proportions $\pi_1, \pi_2, ..., \pi_g$. The probability density function of random vector Y has a g-component mixture form:

$$f(y; \boldsymbol{\theta}) = \sum_{i=1}^{g} \pi_i f_i(y; \lambda_i), \tag{2.1}$$

where $f_i(y; \lambda_i)$ is the density function for the i^{th} component with parameter λ_i, π_i is the probability that y belongs to the i^{th} component with restrictions $0 \leq \pi_i \leq 1$ and $\sum_{i=1}^{g} \pi_i = 1$. Since π_i and λ_i are unknown, the parameters vector is $\boldsymbol{\theta} = (\pi_1, ..., \pi_g, \lambda_1, ..., \lambda_g)$.

In this formulation of mixture model, we assume that the number of components g is fixed. Of course in some applications, g is unknown and has to be estimated along with proportions and parameters. But in this paper, we only consider the situation that g is known and propose to estimate unknown parameter $\boldsymbol{\theta}$.

2.1 EM Algorithm for Mixture Model

Let $\mathbf{y} = (y_1, y_2, ..., y_n)^T$. The log-likelihood function for $\boldsymbol{\theta}$ is given by

$$\log L(\boldsymbol{\theta}; \mathbf{y}) = \sum_{j=1}^{n} \log f(y_j; \boldsymbol{\theta}) = \sum_{j=1}^{n} \log \sum_{i=1}^{g} \pi_i f_i(y_j; \lambda_i). \tag{2.2}$$

The maximum likelihood estimation (MLE) of $\boldsymbol{\theta}$ is

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log L(\boldsymbol{\theta}; \mathbf{y}) \tag{2.3}$$

Taking derivatives with respect to parameters and equating the results to zero clearly does not yield explicit solutions.

In order to use EM algorithm, we introduce as the unobservable or missing data the component label vector $\mathbf{z} = (z_1^T, ..., z_n^T)^T$, where z_j is a g-dimensional vector with elements:

$$z_{ij} = \begin{cases} 1, & \text{if the } j^{th} \text{ observation is from the } i^{th} \text{ component;} \\ 0, & \text{otherwise.} \end{cases} \tag{2.4}$$

Then the complete log-likelihood function for $(\mathbf{y}, \mathbf{z})$ is

$$\log L_c(\boldsymbol{\theta}; \mathbf{y}, \mathbf{z}) = \log \Big\{ \prod_{j=1}^{n} \prod_{i=1}^{g} [\pi_i f_i(y_j; \lambda_i)]^{z_{ij}} \Big\} = \sum_{j=1}^{n} \sum_{i=1}^{g} z_{ij} \log[\pi_i f_i(y_j; \lambda_i)]. \tag{2.5}$$

In the E-step at $(k+1)^{th}$ iteration, we calculate

$$Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(k)}) = E(\log L_c(\boldsymbol{\theta}; \mathbf{y}, \mathbf{z}) | \mathbf{y}, \boldsymbol{\theta}^{(k)}). \tag{2.6}$$

Since (2.6) is linear to z_{ij}, the E-step simply requires the calculation of the conditional expectation $E(z_{ij} | \mathbf{y}, \boldsymbol{\theta}^{(k)})$. This can be explained as follows:

$$\begin{aligned} E(\log L_c(\boldsymbol{\theta}; \mathbf{y}, \mathbf{z}) | \mathbf{y}, \boldsymbol{\theta}^{(k)}) &= E\Bigg\{ \sum_{j=1}^{n} \sum_{i=1}^{g} z_{ij} \log[\pi_i f_i(y_j; \lambda_i)] \Bigg\} \\ &= \sum_{j=1}^{n} \sum_{i=1}^{g} E(z_{ij} | \mathbf{y}, \boldsymbol{\theta}^{(k)}) \big\{ \log[\pi_i f_i(y_j; \lambda_i)] \big\}. \end{aligned}$$

Based on Bayes theorem, we have

$$E(z_{ij} | \mathbf{y}, \boldsymbol{\theta}^{(k)}) = \frac{\pi_i^{(k)} f_i(y_j; \lambda_i^{(k)})}{\sum_{i=1}^{g} \pi_i^{(k)} f_i(y_j; \lambda_i^{(k)})} = \tau_{ij}^{(k+1)}, \tag{2.7}$$

In the M-step for $(k+1)^{th}$ iteration, we find the maximizer of the expected complete log-likelihood function:

$$\boldsymbol{\theta}^{(k+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(k)}) \tag{2.8}$$

Then we repeat the E-step and M-step until convergence.

In particular, we introduce the EM algorithm for normal mixture model. Suppose that the observations distributed according to a mixture of normal distributions with different means and variances. The component density function with mean μ_i and variance σ_i^2 can be written as:

$$f_i(y;\lambda_i) = \phi_i(y;\mu_i,\sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i}\exp\big\{-\frac{(y-\mu_i)^2}{2\sigma_i^2}\big\}. \tag{2.9}$$

Based on the above argument, the EM algorithm for normal mixture model can be expressed in the following steps:

(1) Choose some initial value $\pi_i^{(0)}$, $\mu_i^{(0)}$, $\sigma_i^{2(0)}$.

(2) E-step: On the $(k+1)^{th}$ iteration, according to Bayes theorem, we can compute conditional expectations as follows:

$$E(z_{ij}|\mathbf{y},\boldsymbol{\theta}^{(k)}) = \frac{\pi_i^{(k)}\phi_i(y_j;\mu_i^{(k)},\sigma_i^{2(k)})}{\sum_{i=1}^g \pi_i^{(k)}\phi_i(y_j;\mu_i^{(k)},\sigma_i^{2(k)})} = \tau_{ij}^{(k+1)},$$

(3) M-step: On the $(k+1)^{th}$ iteration, compute the estimator of parameters which maximize the expected complete log-likelihood. The estimators can be written as:

$$\pi_i^{(k+1)} = \sum_{j=1}^n \tau_{ij}^{(k+1)}/n, \tag{2.10}$$

$$\mu_i^{(k+1)} = \frac{\sum_{j=1}^n \tau_{ij}^{(k+1)} y_j}{\sum_{j=1}^n \tau_{ij}^{(k+1)}}, \tag{2.11}$$

$$\sigma_i^{2(k)} = \frac{\sum_{j=1}^n \tau_{ij}^{(k+1)}(y_j-\mu_i^{(k+1)})^2}{\sum_{j=1}^n \tau_{ij}^{(k+1)}} \tag{2.12}$$

(4) Repeat the E-step and M-step until the convergence is obtained.

2.2 Mixture Linear Regression Model

In this section, we propose to apply the previous results to mixture linear regression models in order to estimate the regression coefficients. Suppose that the response y has the following linear relationship with predictor $\mathbf{x}$:

$$y = \mathbf{x}^T\boldsymbol{\beta}_i + \varepsilon_i,\ i = 1,2,...,g, \tag{2.13}$$

where $\boldsymbol{\beta}_i = (\beta_{i1}, ..., \beta_{id})^T$, d is the dimension of $\mathbf{x}$ and ε_i is the random error with density function f_i. Similar to previous argument in mixture model, the conditional density function of Y given $\mathbf{x}$ has the form:

$$f(y|\mathbf{x}, \boldsymbol{\theta}) = \sum_{i=1}^{g} \pi_i f_i(y; \mathbf{x}^T\boldsymbol{\beta}_i, \sigma_i^2), \tag{2.14}$$

where the parameter vector $\boldsymbol{\theta} = (\pi_1, ..., \pi_g, \boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_g, \sigma_1^2, ..., \sigma_g^2)^T$.

We assume that the error terms have density of normal distribution. The log-likelihood function of $\mathbf{y}$ is

$$\log L(\boldsymbol{\theta}; \mathbf{y}) = \sum_{j=1}^{n} \log \sum_{i=1}^{g} \pi_i \phi_i(y_j; \mathbf{x}^T\boldsymbol{\beta}_i, \sigma_i^2). \tag{2.15}$$

The maximizer of (2.15) does not has a explicit solution, so we use the following EM algorithm to estimate the parameters:

(1) Choose some initial value $\pi_i^{(0)}$, $\boldsymbol{\beta}_i^{(0)}$, $\sigma_i^{2(0)}$.

(2) E-step: On the $(k+1)^{th}$ iteration, according to Bayes theorem, we can compute conditional expectations as follows:

$$E(z_{ij}|\mathbf{y}, \boldsymbol{\theta}^{(k)}) = \frac{\pi_i^{(k)}\phi_i(y_j; \mathbf{x}_j^T\boldsymbol{\beta}_i^{(k)}, \sigma_i^{2(k)})}{\sum_{i=1}^{g} \pi_i^{(k)}\phi_i(y_j; \mathbf{x}_j^T\boldsymbol{\beta}_i^{(k)}, \sigma_i^{2(k)})} = \tau_{ij}^{(k+1)},$$

(3) M-step: On the $(k+1)^{th}$ iteration, compute the estimator of parameters which maximize the expected complete log-likelihood:

$$\pi_i^{(k+1)} = \sum_{j=1}^{n} \tau_{ij}^{(k+1)}/n, \tag{2.16}$$

$$\boldsymbol{\beta}_i^{(k+1)} = (\sum_{j=1}^{n} \tau_{ij}^{(k+1)}\mathbf{x}_j\mathbf{x}_j^T)^{-1}(\sum_{j=1}^{n} \tau_{ij}^{(k+1)}\mathbf{x}_j y_j), \tag{2.17}$$

$$\sigma_i^{2(k)} = \frac{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}(y_j - \mathbf{x}_j^T\boldsymbol{\beta}_i^{(k+1)})^2}{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}}. \tag{2.18}$$

(4) Repeat the E-step and M-step until convergence.

Chapter 3

Mixture of Pearson type VII distribution

It is widely known that Gaussian mixture modeling is sensitive to outliers. In order to robustly fit the data, Wei (2012) proposed a mixture regression model by assuming that the error terms follow the Student t-distribution. Simulation results showed that the mixture regression model with t-distribution performed better than Gaussian mixture regression as well as some other procedures. A trimming method has also been proposed by Wei (2012) , which was demonstrated to be useful in trimming the high leverage points.

On the other hand, Sun and Garibaldi (2010) proposed a robust mixture modeling with Pearson Type VII distribution. The experimental results showed that the Pearson VII mixture was more stable than Student t mixture in classification and outlier detection. Sun and Garibaldi (2010) also pointed out that, in the Pearson VII mixture, the equation which we need to solve for estimating degrees of freedom had a less complicated form. Thus the estimation of degrees of freedom in the Pearson VII mixture could be easier than which in Student t mixture.

Inspired by the above-mentioned references, we propose to extend the Pearson type VII mixture into the mixture regression model and anticipate to see that mixture regression mixture with Pearson VII distribution may perform comparably or better than the mixture regression model with t-distribution proposed in Wei (2012). In this chapter, we first introduce the mixture of Pearson type VII distribution which is proposed by Sun and Garibaldi (2010) and then extend it into a mixture regression model in the next chapter.

The Pearson type VII distribution is defined as follows:

$$f(\mathbf{y};\mu,\Sigma,m) = \frac{\Gamma(m)}{\pi^{\frac{d}{2}}\Gamma(m-\frac{d}{2})}|\Sigma|^{-\frac{1}{2}}(1+\Delta^2)^{-m} \tag{3.1}$$

where $\Delta^2 = (\mathbf{y}-\mu)^T\Sigma^{-1}(\mathbf{y}-\mu)$ is the Mahalanobis distance and $2m > d$, m is the degree of freedom that controls the degree of robustness, and d is the dimensionality of $\mathbf{y}$. It can be shown that the normal distribution arises as a special case of Pearson type VII distribution if m approaches infinity. The Student t-distribution $\mathcal{S}(\mathbf{y};\mu,\Sigma,\nu)$ with degree of freedom ν

$$\mathcal{S}(\mathbf{y};\mu,\Sigma,\nu) = \frac{\Gamma(\frac{\nu+d}{2})|\Sigma|^{-\frac{1}{2}}}{(\pi\nu)^{\frac{d}{2}}\Gamma(\frac{\nu}{2})(1+\frac{\Delta^2}{\nu})^{\frac{\nu+d}{2}}}$$

is also a special case of the Pearson type VII distribution if we set

$$m = \frac{\nu+d}{2}, \Sigma = \Sigma\nu.$$

The mixture of Pearson type VII distribution with g-components has the density

$$f(\mathbf{y};\boldsymbol{\theta}) = \sum_{i=1}^{g}\pi_i f(\mathbf{y};\mu_i,\Sigma_i,m_i), \tag{3.2}$$

where $\boldsymbol{\theta} = (\boldsymbol{\pi}^T, \boldsymbol{\xi}^T, \boldsymbol{m}^T)$, $\boldsymbol{\xi} = (\xi_1^T, ..., \xi_g^T)$ consists of the elements of the component means, $\mu_1, ..., \mu_g$, and the distinct elements of the component covariance, $\Sigma_1, ..., \Sigma_g$, $\mathbf{m} = (m_1, ..., m_g)$, and $\boldsymbol{\pi} = (\pi_1, ...\pi_{g-1})^T$ is the vector containing the $g-1$ mixing proportions $\pi_1, ...\pi_{g-1}$ with $\sum_{i=1}^{g}\pi_i = 1$.

We let $\mathbf{Y} = (\mathbf{y}_1^T, ..., \mathbf{y}_n^T)^T$ denote the random sample obtained from the mixture density with size of n, where $\mathbf{y}_j$ is the j^{th} random vector, $j = 1, ..., n$. The log-likelihood function for $\boldsymbol{\theta}$ is given by

$$\log L(\boldsymbol{\theta};\mathbf{Y}) = \sum_{j=1}^{n}\log f(\mathbf{y}_j;\boldsymbol{\theta}) = \sum_{j=1}^{n}\log\sum_{i=1}^{g}\pi_i f(\mathbf{y}_j;\mu_i,\Sigma_i,m_i). \tag{3.3}$$

The maximum likelihood estimation (MLE) of $\boldsymbol{\theta}$ is calculated by maximizing the log-likelihood function (3.3), which does not yield an explicit solution. It is well-known that the

Pearson type VII distribution belongs to the class of scale mixture of normal distribution. In order to simplify the calculation of MLE, we introduce another latent variable u, such that

$$\mathbf{y}|u \sim N(\mu, \Sigma/u),$$
$$u \sim \text{gamma}(m - \frac{1}{2}, \frac{1}{2}),$$

where $N(\mu, \Sigma/u)$ is a normal distribution has a density with mean μ and variance Σ/u:

$$\phi(\mathbf{y}; \mu, \Sigma/u) = \frac{1}{(2\pi)^{d/2}|\Sigma/u|^{1/2}} \exp\{-\frac{1}{2}(\mathbf{y} - \mu)^T(\Sigma/u)^{-1}(\mathbf{y} - \mu)\}, \tag{3.4}$$

and gamma$(m - \frac{d}{2}, \frac{1}{2})$ has a density

$$G(u; m - \frac{d}{2}, \frac{1}{2}) = \frac{1}{\Gamma(m - \frac{d}{2})}(\frac{1}{2})^{m-\frac{d}{2}} u^{m-\frac{d}{2}-1} \exp\{-\frac{u}{2}\} \tag{3.5}$$

Marginally $\mathbf{y}$ has the Pearson type VII distribution in the following form:

$$f(\mathbf{y}; \mu, \Sigma, m) = \int_0^\infty \phi(\mathbf{y}; \mu, \Sigma/u) G(u; m - \frac{d}{2}, \frac{1}{2}) du \tag{3.6}$$

Let $\mathbf{u} = (u_1, ..., u_n)$. To apply the EM algorithm for the MLE of model parameters, we introduce indicator variables z_{ij} and write the complete log-likelihood as follows:

$$\begin{aligned} \log L_c(\boldsymbol{\theta}; \mathbf{Y}, \mathbf{z}, \mathbf{u}) &= \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log\{\pi_i \phi(\mathbf{y}_j; \mu_i, \Sigma_i/u_j) G(u_j; m_i - \frac{d}{2}, \frac{1}{2})\} \\ &= \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log \pi_i + \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log \phi(\mathbf{y}_j; \mu_i, \Sigma_i/u_j) \\ &\quad + \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log G(u_j; m_i - \frac{d}{2}, \frac{1}{2}), \end{aligned} \tag{3.7}$$

where

$$\begin{aligned} &\sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log \phi(\mathbf{y}_j; \mu_i, \Sigma_i/u_j) \\ = &\sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \Big[-\frac{d}{2}\log(2\pi_i) - \frac{1}{2}\log|\Sigma_i/u_j| - \frac{1}{2}(\mathbf{y}_j - \mu_i)^T(\Sigma_i/u_j)^{-1}(\mathbf{y}_j - \mu_i)\Big], \end{aligned}$$

and where

$$\begin{aligned}
&\sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij}\log G(u_j; m_i-\frac{d}{2},\frac{1}{2}) \\
=\ &\sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij}\Big[-\log\Gamma(m_i-\frac{d}{2})+(m_i-\frac{d}{2})\log(\frac{1}{2})+(m_i-\frac{d}{2}-1)\log u_j-\frac{u_j}{2}\Big],
\end{aligned}$$

In E-step on the $(k+1)^{th}$ iteration, we need to calculate the conditional expectation of the log-likelihood function of complete data, which is $E(\log L_c(\boldsymbol{\theta};\mathbf{z},\mathbf{u})|\mathbf{Y},\boldsymbol{\theta}^{(k)})$. Based on the above argument, the E-step requires the calculations of $E(z_{ij}|\mathbf{Y},\boldsymbol{\theta}^{(k)})$, $E(u_j|\mathbf{Y}, z_{ij}=1,\boldsymbol{\theta}^{(k)})$ and $E(\log u_j|\mathbf{Y}, z_{ij}=1,\boldsymbol{\theta}^{(k)})$. Thus, the EM algorithm can be written as:

(1) Choose some initial value $\pi_i^{(0)}$, $\mu_i^{(0)}$, $\Sigma_i^{(0)}$ and $m_i^{(0)}$. We can use some clustering method (such as K-means clustering)to cluster the data into g groups. Then in each cluster, estimate the component parameter and use these estimation values as the initial values.

(2) E-step: On the $(k+1)^{th}$ iteration, according to Bayes theorem, we can compute conditional expectations as follows:

$$E(z_{ij}|\mathbf{Y},\boldsymbol{\theta}^{(k)}) = \frac{\pi_i^{(k)} f(\mathbf{y}_j;\mu_i^{(k)},\Sigma_i^{(k)},m_i^{(k)})}{f(\mathbf{y}_j;\boldsymbol{\theta}^{(k)})} = \tau_{ij}^{(k+1)},$$

where

$$f(\mathbf{y}_j;\mu_i^{(k)},\Sigma_i^{(k)},m_i^{(k)}) = \frac{\Gamma(m_i^{(k)})}{(\pi_i^{(k)})^{\frac{d}{2}}\Gamma(m_i^{(k)}-\frac{d}{2})}|\Sigma_i^{(k)}|^{-\frac{1}{2}}[1+\Delta^2(\mathbf{y}_j;\mu_i^{(k)},\Sigma_i^{(k)},m_i^{(k)})]^{-m_i^{(k)}},$$

$$f(\mathbf{y}_j;\boldsymbol{\theta}^{(k)}) = \sum_{i=1}^{g}\pi_i^{(k)} f(\mathbf{y}_j;\mu_i^{(k)},\Sigma_i^{(k)},m_i^{(k)})$$

and

$$\Delta^2(\mathbf{y}_j;\mu_i^{(k)},\Sigma_i^{(k)},m_i^{(k)}) = (\mathbf{y}_j-\mu_i^{(k)})^T(\Sigma_i^{(k)})^{-1}(\mathbf{y}_j-\mu_i^{(k)}).$$

$$E(u_j|\mathbf{Y}, z_{ij}=1,\boldsymbol{\theta}^{(k)}) = \frac{2m_i^{(k)}}{1+\Delta^2(\mathbf{y}_j;\mu_i^{(k)},\Sigma_i^{(k)},m_i^{(k)})} = u_{ij}^{(k+1)}. \tag{3.8}$$

$$E(\log u_j|\mathbf{Y}, z_{ij}=1,\boldsymbol{\theta}^{(k)}) = \psi(m_i^{(k)}) - \log\frac{1+\Delta^2(\mathbf{y}_j;\mu_i^{(k)},\Sigma_i^{(k)},m_i^{(k)})}{2}, \tag{3.9}$$

where ψ is the digamma function and $\psi(x)=\frac{\Gamma'(x)}{\Gamma(x)}$.

(3) M-step: On the $(k+1)^{th}$ iteration, compute the estimator of parameters which maximize the expected complete log-likelihood. The estimators can be written as:

$$\pi_i^{(k+1)} = \sum_{j=1}^{n} \tau_{ij}^{(k+1)}/n, \tag{3.10}$$

$$\mu_i^{(k+1)} = \frac{\sum_{j=1}^{n} \tau_{ij}^{(k+1)} u_{ij}^{(k+1)} \mathbf{y}_j}{\sum_{j=1}^{n} \tau_{ij}^{(k+1)} u_{ij}^{(k+1)}}, \tag{3.11}$$

$$\Sigma_i^{(k)} = \frac{\sum_{j=1}^{n} \tau_{ij}^{(k+1)} u_{ij}^{(k+1)} (\mathbf{y}_j - \mu_i^{(k+1)})(\mathbf{y}_j - \mu_i^{(k+1)})^T}{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}} \tag{3.12}$$

To get $m_i^{(k+1)}$, we need to solve the following equation:

$$\sum_{j=1}^{n} \tau_{ij}^{(k+1)} \Big[E(\log u_j | \mathbf{Y}, z_{ij} = 1, \boldsymbol{\theta}^{(k)}) - \log(2) - \psi(m_i^{(k+1)} - \frac{d}{2}) \Big] = 0 \tag{3.13}$$

(4) Repeat the E-step and M-step until the convergence is obtained. One stopping rule we can choose is to stop the iteration when the change of the likelihood value is smaller than 10^{-6} or runs are more than 500.

Chapter 4

Robust Mixture Regression Model with Pearson type VII distribution

In order to robustly estimate the mixture regression parameters, we assume that the error density $f_i(\varepsilon)$ is a Pearson type VII distribution with degree of freedom m_i and scale parameter σ_i. Hence, given $\mathbf{x}_j$, the density function of y_j can be write as follows

$$f(y_j; \mathbf{x}_j, \boldsymbol{\theta}) = \sum_{i=1}^{g} \pi_i f(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i, \sigma_i^2, m_i), \tag{4.1}$$

where

$$f(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i, \sigma_i^2, m_i) = \frac{\Gamma(m_i)\sigma_i^{-1}}{\pi_i^{\frac{1}{2}} \Gamma(m_i - \frac{1}{2}) \{1 + \Delta^2(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i, \sigma_i^2)\}^{m_i}}$$

and

$$\Delta^2(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i, \sigma_i^2) = (y_j - \mathbf{x}_j^T \boldsymbol{\beta}_i)^2 / \sigma_i^2.$$

We propose to use the EM algorithm for the maximum likelihood estimation(MLE) of the model parameters. First of all, we assume that all m_i's are known. The unknown parameter $\boldsymbol{\theta}$ can be estimated by maximizing the log-likelihood function. Note that the complete log-likelihood function for $\boldsymbol{\theta}$ is

$$\log L_c(\boldsymbol{\theta}; \mathbf{X}, \mathbf{y}, \mathbf{z}) = \sum_{j=1}^{n} \sum_{i=1}^{g} z_{ij} \log\{\pi_i f(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i, \sigma_i^2, m_i)\}, \tag{4.2}$$

where $\mathbf{X} = (\mathbf{x_1}, ..., \mathbf{x_1})^T$, $\mathbf{y} = (y_1, ..., y_n)^T$, $\mathbf{z} = (z_{ji})_{n \times g}$.

Since the Pearson type VII distribution can be considered as a scale mixture of normal distribution, we introduce a latent variable u such that

$$y|u \sim N(\mathbf{x}^T\boldsymbol{\beta}, \sigma^2/u),$$

$$u \sim \text{gamma}(m - \frac{1}{2}, \frac{1}{2}).$$

Then the complete log-likelihood function can be write as

$$\begin{aligned}
L_c(\boldsymbol{\theta}; \mathbf{X}, \mathbf{y}, \mathbf{z}) &= \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log\{\pi_i \phi(y_j; \mathbf{x}_j^T\boldsymbol{\beta}_i, \sigma_i^2/u_j) G(u_j; m_i - \frac{1}{2}, \frac{1}{2})\} \\
&= \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log \pi_i + \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log\{-\frac{1}{2}\log(2\pi\sigma_i^2) - \frac{u_j}{2\sigma_i^2}(y_j - \mathbf{x}_j^T\boldsymbol{\beta}_i)^2\} + C
\end{aligned}$$

where

$$C = \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij} \log G(u_j; m_i - \frac{1}{2}, \frac{1}{2}) + \sum_{j=1}^{n}\sum_{i=1}^{g} z_{ij}(\frac{1}{2}\log u_j)$$

does not depend on $\boldsymbol{\theta}$.

Therefore, in the E-step on the $(k{+}1)^{th}$ iteration,the calculation of $E(\log L_c(\boldsymbol{\theta}; \mathbf{X}, \mathbf{y}, \mathbf{z})|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)})$ simplifies to the calculation of $E(z_{ij}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)})$ and $E(u_j|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}, z_{ij} = 1)$. According to Bayes Theorem, we can evaluate the posteriors as follows:

$$\begin{aligned}
p(z_{ij} = 1|\mathbf{X}, \mathbf{y}) &= \frac{p(z_{ij} = 1|\mathbf{x}_j) f(y_j|\mathbf{x}_j, z_{ij} = 1)}{f(y_j|\mathbf{x}_j)} \\
&= \frac{\pi_i f(y_j|\mathbf{x}_j, z_{ij} = 1)}{f(y_j|\mathbf{x}_j)}
\end{aligned}$$

$$\begin{aligned}
f(u_j|\mathbf{X}, \mathbf{y}, z_{ij} = 1) &= \frac{f(y_j|u_j, \mathbf{x}_j, z_{ij} = 1) f(u_j|\mathbf{x}_j, z_{ij} = 1)}{f(y_j|\mathbf{x}_j, z_{ij} = 1))} \\
&= gamma(u_j; m_i, (1 + \Delta^2(y_j; \mathbf{x}_j^T\boldsymbol{\beta}_i, \sigma_i^2))/2)
\end{aligned}$$

So the corresponding conditional expectations on the $(k+1)^{th}$ iteration are

$$E(z_{ij}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}) = \frac{\pi_i^{(k)} f(y_j; \mathbf{x}_j^T\boldsymbol{\beta}_i^{(k)}, \sigma_i^{2(k)}, m_i^{(k)})}{\sum_{i=1}^{g} \pi_i^{(k)} f(y_j; \mathbf{x}_j^T\boldsymbol{\beta}_i^{(k)}, \sigma_i^{2(k)}, m_i^{(k)})} = \tau_{ij}^{(k+1)}, \tag{4.3}$$

$$E(u_j|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}, z_{ij} = 1) = \frac{2m_i^{(k)}}{1 + \Delta^2(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i^{(k)}, \sigma_i^{2(k)}, m_i^{(k)})} = u_{ij}^{(k+1)}. \tag{4.4}$$

In the M-step on the $(k+1)^{th}$ iteration, we find the maximizer of the following expected complete log-likelihood with respect to $\pi_i, \boldsymbol{\beta}_i$ and σ_i^2 separately:

$$E(\log L_c(\boldsymbol{\theta}; \mathbf{X}, \mathbf{y}, \mathbf{z})|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}) \propto$$
$$\sum_{j=1}^{n} \sum_{i=1}^{g} E(z_{ij}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}) \Big[\log \pi_i - \frac{1}{2} \log(2\pi\sigma_i^2) - \frac{E(u_j|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}, z_{ij} = 1)}{2\sigma_i^2} (y_j - \mathbf{x}_j^T \boldsymbol{\beta}_i)^2 \Big], \tag{4.5}$$

then we obtain the estimators of parameters $\pi_i, \boldsymbol{\beta}_i, \sigma_i^2$

$$\pi_i^{(k+1)} = \sum_{j=1}^{n} \tau_{ij}^{(k+1)}/n, \tag{4.6}$$

$$\boldsymbol{\beta}_i^{(k+1)} = \Big(\sum_{j=1}^{n} \mathbf{x}_j \mathbf{x}_j^T w_{ij}^{(k+1)} \Big)^{-1} \sum_{j=1}^{n} \mathbf{x}_j y_j w_{ij}^{(k+1)}, \tag{4.7}$$

$$\sigma_i^{2(k+1)} = \frac{\sum_{j=1}^{n} w_{ij}^{(k+1)} (y_j - \mathbf{x}_j^T \boldsymbol{\beta}_i^{(k+1)})^2}{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}} \tag{4.8}$$

where $w_{ij}^{(k+1)} = \tau_{ij}^{(k+1)} u_{ij}^{(k+1)}$. Repeat the E-step and M-step until the convergence is obtained.

If we further assume that all σ_i^2's are equal, then in the M-step, we can update σ^2 by

$$\sigma^{2(k+1)} = \frac{\sum_{i=1}^{g} \sum_{j=1}^{n} w_{ij}^{(k+1)} (y_j - \mathbf{x}_j^T \boldsymbol{\beta}_i^{(k+1)})^2}{n} \tag{4.9}$$

In M-step of the above EM procedure, the formulae (4.7), which is used to update the regression parameters β_i in $(k+1)^{th}$ iteration, can be considered as a weighted least squares estimate with the weights depending on $u_{ij}^{(k+1)}$. Note from (4.4) that $u_{ij}^{(k+1)}$ is reversely related to the term $\Delta^2(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i^{(k)}, \sigma_i^{2(k)}, m_i^{(k)})$. In other words, larger residuals result in smaller $u_{ij}^{(k+1)}$ which means smaller weights. Thus the effects of the outliers decrease and robust estimate for regression parameters can be obtained. In addition, it is obvious from (4.8) that larger residuals also have smaller effects on $\sigma_i^{2(k+1)}$ due to the same reason.

The above EM procedure based on Pearson type VII distribution is robust when the outliers are in y-direction, but not in x-direction. Hence, similar to the traditional M-estimate for linear regression, our method is not robust when outliers are high leverage points. To deal with it, we need to find out those high leverage points and exclude them from further analysis. One classical method for identifying high leverage points is to use the diagonal elements h_{jj} of hat matrix $H = X(X^TX)^{-1}X^T$ as the indicator. The j^{th} observation would be considered as a high leverage point when $h_{jj} > 2p/n$, where $\sum_{j=1}^n h_{jj} = p$ is the dimension of $\mathbf{X}$. Note that

$$h_{jj} = n^{-1} + (n-1)^{-1} MD_j \tag{4.10}$$

where $MD_j = (\mathbf{x}_j - \bar{\mathbf{x}})^T S^{-1} (\mathbf{x}_j - \bar{\mathbf{x}})$, $\bar{\mathbf{x}}$ and S denote the sample mean and sample covariance of $\mathbf{x}_j$'s, respectively. Rousseeuw and Van Zomeren (1990) point out that the h_{jj} are not entirely safe for detecting leverage points reliably. Some outliers do not necessarily have large MD_j values due to the masking effect caused by $\bar{\mathbf{x}}$ and S. In order to avoid masking effect, we would use robust estimates of location and scatter matrix of $\mathbf{X}$ (after removing the the first column 1s) to substitute $\bar{\mathbf{x}}$ and S in the calculation of MD_j. Wei (2012) propose to use the minimum covariance determinant (MCD) estimators, which are highly robust estimators for location and scatter, and implement it by Fast-MCD algorithm proposed in Rousseeuw and Van Driessen (1999). Pison et al.(2002) suggests a threshold of $\chi^2_{p,0.975}$. In this paper, the proposed EM algorithm based on Pearson type VII distribution will be implemented after removing the observations with $MD_j > \chi^2_{p,0.975}$, where p is the dimension of $\mathbf{X}$.

In the previous discussion, we assume that all the degrees of freedom m_i's are known. Now let us consider the estimation of m_i. In the M-step of EM procedure for mixture regression model with Pearson type VII distribution, take derivatives of the complete log-likelihood function with respect to $m_i^{(k)}$ and obtain the equation

$$\sum_{j=1}^n \tau_{ij}^{(k+1)} \Big[E(\log u_j | \mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}, z_{ij} = 1) - \log(2) - \psi(m_i^{(k+1)} - \frac{d}{2}) \Big] = 0, \tag{4.11}$$

or specifically:

$$\psi\Big(m_i^{(k+1)} - \frac{d}{2}\Big) = \frac{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}[E(\log u_j|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}, z_{ij} = 1) - \log(2)]}{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}}, \tag{4.12}$$

where

$$E(\log u_j|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^{(k)}, z_{ij} = 1) = \psi(m_i^{(k)}) - \log \frac{1 + \Delta^2(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i^{(k)}, \sigma_i^{2(k)}, m_i^{(k)})}{2}. \tag{4.13}$$

Comparing to the equation for estimating the degree of freedom $\nu_i^{(k+1)}$ in mixture regression model with t-distribution

$$\psi\Big(\frac{\nu_i^{(k+1)}}{2}\Big) - \log\Big(\frac{\nu_i^{(k+1)}}{2}\Big) = 1 + \frac{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}(\log u_{ij}^{(k+1)} - u_{ij}^{(k+1)})}{\sum_{j=1}^{n} \tau_{ij}^{(k+1)}} + \psi\Big(\frac{\nu_i^{(k)}+1}{2}\Big) - \log\Big(\frac{\nu_i^{(k)}+1}{2}\Big), \tag{4.14}$$

it is obvious that the estimation of m_i could be easier. In fact, Peel and McLachlan (2000) claim that there is no expicit solution for ν_i. On the other hand, M. Pike and I.Hill(1966) propose an algorithm that can be efficient for solving the inverse digamma function, and thus can solve the equation for m_i.

Since the degree of freedom ν_i for mixture regression model with t-distribution does not have an explicit solution, Wei (2012) propose a method to estimate it, which can also be used for our model based on Pearson type VII distribution. To simplify the computation, we assume that all m_i's equal to a common m. The proposed profile likelihood for m is:

$$L(m) = \max_{\boldsymbol{\theta}} \sum_{j=1}^{n} \log \Big\{ \sum_{i=1}^{g} \pi_i f(y_j; \mathbf{x}_j^T \boldsymbol{\beta}_i, \sigma_i^2, m) \Big\} \tag{4.15}$$

Based on the proposed EM procedure in this section, we can obtain the estimation of m by

$$\hat{m} = \arg\max_{m} L(m) \tag{4.16}$$

We can substitute m by a set of positive values within $[0.6, m_{\max}]$ in the function of $L(m)$. Usually, the value of $m_{\max}$ does not need to too large and 15 to 20 would be enough.

Chapter 5

Simulation Study

In this section, we will evaluate the performance of the proposed robust mixture regression model based on Pearson type VII distribution through a simulation study. The comparison will also be made among different estimation methods.

The model used in the simulation is

$$Y = \begin{cases} 0 + X_1 + X_2 + \varepsilon_1, & \text{if Z=1;} \\ 0 - X_1 - X_2 + \varepsilon_2, & \text{if Z=2.} \end{cases}$$

where $X_1 \sim N(0,1)$, $X_2 \sim N(0,1)$, ε_1 and ε_2 have the same distribution as ε, and Z is a component indicator of Y with $P(Z=1) = 0.25$.

The following error distribution will be considered:

Case I: $\varepsilon \sim$ N(0,1)

Case II: $\varepsilon \sim t_1$, t-distribution with degree of freedom 1

Case III: $\varepsilon \sim t_3$, t-distribution with degree of freedom 3

Case IV: $\varepsilon \sim 0.95N(0,1) + 0.05N(0,5^2)$

Case V: $\varepsilon \sim N(0,1)$ with 5% of high leverage outliers being X_1=X_2=20 and $Y = 100$.

In Case I, the error is exactly normally distributed and there are no outliers. It is often used to evaluate the efficiency of different estimation methods compared to the traditional MLE. Both Case II and III are heavy tailed distributions. In case IV, the 5% data from

$N(0, 5^2)$ are likely to be low leverage outliers, and in Case V, 5% observations are replicated high leverage outliers, which will be used to check the robustness of estimation procedures against the outliers in the x-direction.

The following algorithm will be compared:

- MLE based on normality assumption
- Trimmed likelihood estimator (TLE) proposed by Neykov et al. (2007)
- The robust modified EM algorithm based on Bisquare (MEM-bisquare) proposed by Bai et al. (2012)
- The robust EM mixture regression based on t-distribution (Mixregt)
- The trimmed EM mixture regression based on t-distribution (Mixregt-MCD)
- The proposed robust EM mixture regression based on Pearson type VII distribution (MixregP)
- The proposed trimmed EM mixture regression based on Pearson type VII distribution (MixregP-MCD)

From the simulation results shown in the following tables, we can see that the MLE works the best when the true error distribution is normal, and this superiority becomes more obviously when the sample size is 400. But for Case II and V, in which the distribution of error has heavy tails or high leverage outliers, the MLE fails to provide good estimations. The performance of TLE is satisfying in Case II and V, which means TLE works better than MLE when ε has t-distribution with degree of freedom 1, but worse when ε is normal distributed. The overall performance of Bisquare is satisfying except Case II, under which Mixregt and MixregP make better estimations.

The superiority of Mixregt-MCD is obvious in Case V when the data set includes high leverage points. Mixregt-MCD works much better than Mixregt and also better than MLE, TLE and Bisquare in dealing with high leverage points.

We anticipate that the proposed method in this paper is superior to or comparable to other methods, and it has been confirmed in the simulation results. MixregP works well in Case I, II, III and IV, and MixregP-MCD successfully removes the high leverage outliers in Case V. Since the t-distribution is subsumed to the Pearson type VII distribution, it is not surprising that the performance of Mixregt and Mixregt-MCD are very close to our proposed methods in some cases, but the superiority of the proposed methods are obvious in Case II, when the error distribution is Cauchy distribution.

By using the profile likelihood for the selection of degrees of freedom for Pearson type VII distribution, we report the mean and median of estimated degrees of freedom for MixregP and MixregP-MCD separately in table 5.4. The degrees of freedom are chosen from the 30 equally spaced points within [0.6,15]. From the simulation results, we can see that the optimal degree of freedom is 15 when error terms follow normal distribution. It is a reasonable estimation since the Pearson type VII distribution will be close to the normal distribution when the degree of freedom becomes large. We know that the degree of free for Pearson type VII, m, and the degree of free for t-distribution, ν, have a relation that $m = \frac{\nu+1}{2}$. Based on this relationship, the results in Case II and III show that the profile likelihood method is adaptive in estimating the degree of freedom for Pearson type VII distribution. In Case IV, both MixregP and MixregP-MCD and able to use a heavy-tailed Pearson type VII dsitribution to approximate the normal mixture. In Case V, MixregP-MCD successfully trimmed the high leverage outliers since the optimal distribution is not heavy-tailed and close to normal distribution.

	MLE	TLE	Bisquare	Mixregt	Mixregt-MCD	MixregP	MixregP-MCD
				Case I: $\varepsilon \sim N(0,1)$			
β_{10}	0.155(0.009)	0.387(0.202)	0.164(0.053)	0.169(0.047)	0.142(0.026)	0.154(0.051)	0.143(0.020)
β_{11}	0.184(-0.051)	1.601(-1.038)	0.261(-0.089)	0.154(-0.061)	0.164(-0.111)	0.151(-0.049)	0.124(-0.087)
β_{12}	0.165(-0.052)	0.856(-0.636)	0.247(-0.097)	0.141(-0.016)	0.173(-0.001)	0.143(-0.028)	0.167(0.005)
β_{20}	0.017(-0.007)	0.091(-0.046)	0.024(-0.011)	0.022(-0.014)	0.020(0.009)	0.022(-0.014)	0.019(0.011)
β_{21}	0.023(-0.020)	0.037(-0.008)	0.027(-0.023)	0.021(-0.004)	0.022(0.001)	0.021(-0.006)	0.021(-0.002)
β_{22}	0.020(0.009)	0.032(0.019)	0.021(0.011)	0.021(-0.010)	0.021(-0.019)	0.021(-0.010)	0.021(-0.017)
π_1	0.006(0.008)	0.014(0.052)	0.007(0.013)	0.005(0.013)	0.005(0.010)	0.005(0.013)	0.005(0.008)
				Case II: $\varepsilon \sim t_1$			
β_{10}	243.003(-0.114)	3.077(-0.060)	1.735(-0.020)	1.654(0.020)	0.633(0.020)	1.737(-0.013)	3.241(-0.130)
β_{11}	174.651(-1.566)	1.568(-0.490)	1.597(-0.421)	2.336(-0.524)	1.161(-0.620)	2.457(-0.422)	1.575(-0.323)
β_{12}	148.115(-1.773)	1.413(-0.130)	1.475(-0.301)	2.640(-0.620)	1.083(-0.562)	2.725(-0.573)	1.300(-0.342)
β_{20}	242.169(0.079)	1.749(0.004)	0.934(0.022)	0.133(-0.024)	0.180(0.021)	0.120(-0.022)	0.087(0.026)
β_{21}	175.300(-1.187)	0.974(-0.183)	0.548(-0.116)	0.104(-0.049)	0.093(-0.016)	0.101(-0.039)	0.067(-0.007)
β_{22}	142.066(-0.420)	0.944(-0.149)	0.654(-0.114)	0.141(-0.049)	0.107(-0.032)	0.149(-0.045)	0.088(-0.032)
π_1	0.087(0.220)	0.049(0.087)	0.053(0.115)	0.035(0.127)	0.027(0.110)	0.032(0.112)	0.017(0.076)
				Case III: $\varepsilon \sim t_3$			
β_{10}	1.611(-0.092)	0.605(0.202)	0.585(0.078)	0.453(0.049)	0.437(0.133)	0.550(0.020)	0.459(0.137)
β_{11}	1.028(-0.280)	1.173(-0.777)	0.556(-0.166)	0.450(-0.061)	0.584(-0.116)	0.443(-0.059)	0.457(-0.013)
β_{12}	1.233(-0.069)	0.590(-0.418)	0.497(-0.161)	0.368(-0.045)	0.525(-0.106)	0.369(-0.046)	0.514(-0.084)
β_{20}	0.760(-0.024)	0.096(-0.041)	0.077(-0.012)	0.044(0.000)	0.117(-0.009)	0.043(0.000)	0.112(-0.001)
β_{21}	0.235(0.017)	0.063(0.016)	0.050(-0.013)	0.050(-0.018)	0.325(-0.012)	0.050(-0.017)	0.323(0.016)
β_{22}	0.123(0.040)	0.044(0.007)	0.047(-0.034)	0.034(-0.018)	0.339(-0.010)	0.034(-0.017)	0.334(-0.009)
π_1	0.031(0.036)	0.014(0.060)	0.012(0.042)	0.008(0.022)	0.014(0.032)	0.009(0.022)	0.014(0.029)
				Case IV: $\varepsilon \sim 0.95N(0,1) + 0.05N(0,25)$			
β_{10}	2.185(-0.022)	0.363(0.136)	0.227(0.062)	0.166(0.011)	0.248(0.086)	0.183(-0.006)	0.214(0.068)
β_{11}	1.500(-0.088)	1.186(-0.806)	0.530(-0.239)	0.215(-0.064)	0.253(-0.122)	0.242(-0.067)	0.190(-0.057)
β_{12}	2.256(-0.141)	0.696(-0.515)	0.468(-0.225)	0.184(-0.070)	0.371(-0.004)	0.179(-0.063)	0.380(0.069)
β_{20}	1.592(0.142)	0.078(0.004)	0.094(0.032)	0.028(0.020)	0.030(-0.013)	0.028(0.020)	0.025(-0.007)
β_{21}	1.142(-0.101)	0.043(-0.012)	0.040(-0.027)	0.023(-0.009)	0.027(0.003)	0.022(-0.008)	0.026(0.004)
β_{22}	0.459(-0.017)	0.039(-0.013)	0.041(-0.010)	0.120(-0.040)	0.029(-0.016)	0.120(-0.039)	0.028(-0.016)
π_1	0.045(0.027)	0.017(0.064)	0.020(0.053)	0.007(0.020)	0.007(0.012)	0.007(0.018)	0.007(0.007)
				Case V: $\varepsilon \sim N(0,1)$ with 5% high leverage outliers			
β_{10}	18.157(-2.891)	0.339(0.100)	0.176(0.031)	2.472(0.186)	0.185(0.002)	2.481(0.195)	0.158(-0.017)
β_{11}	5.855(1.452)	0.942(-0.646)	0.446(-0.241)	3.437(1.434)	0.264(-0.083)	3.432(1.430)	0.212(-0.032)
β_{12}	6.320(1.611)	0.575(-0.421)	0.445(-0.218)	3.814(1.555)	0.220(-0.028)	3.831(1.559)	0.179(0.010)
β_{20}	11.761(2.464)	0.054(-0.018)	0.035(-0.011)	0.023(0.002)	0.036(-0.001)	0.023(0.003)	0.026(0.010)
β_{21}	12.279(3.302)	0.038(-0.014)	0.028(-0.017)	0.053(0.141)	0.024(0.019)	0.053(0.141)	0.021(0.016)
β_{22}	12.658(3.360)	0.027(0.010)	0.029(0.012)	0.053(0.136)	0.020(-0.016)	0.053(0.137)	0.020(-0.012)
π_1	0.113(0.174)	0.010(0.035)	0.016(0.022)	0.007(-0.074)	0.006(0.010)	0.007(-0.074)	0.005(0.002)

Table 5.1: MSE(Bias) of Point Estimates for $n = 100$

	MLE	TLE	Bisquare	Mixregt	Mixregt-MCD	MixregP	MixregP-MCD
	Case I: $\varepsilon \sim N(0,1)$						
β_{10}	0.043(-0.010)	0.234(0.150)	0.044(-0.009)	0.047(-0.022)	0.041(-0.024)	0.048(-0.022)	0.041(-0.024)
β_{11}	0.041(-0.007)	0.952(-0.754)	0.044(-0.007)	0.035(0.008)	0.057(-0.001)	0.035(0.007)	0.057(-0.001)
β_{12}	0.040(-0.020)	0.474(-0.455)	0.044(-0.018)	0.044(-0.007)	0.039(-0.027)	0.044(-0.007)	0.040(-0.026)
β_{20}	0.009(-0.006)	0.036(-0.026)	0.009(-0.007)	0.008(-0.010)	0.008(0.017)	0.008(-0.010)	0.008(0.017)
β_{21}	0.008(-0.006)	0.013(0.018)	0.008(-0.007)	0.009(-0.001)	0.009(-0.002)	0.009(0.000)	0.009(-0.002)
β_{22}	0.010(-0.014)	0.018(-0.018)	0.012(-0.015)	0.008(-0.006)	0.010(-0.009)	0.008(-0.005)	0.010(-0.008)
π_1	0.002(0.010)	0.006(0.020)	0.003(0.012)	0.002(0.007)	0.002(0.001)	0.002(0.007)	0.002(0.001)
	Case II: $\varepsilon \sim t_1$						
β_{10}	286.807(1.712)	1.535(-0.105)	1.260(-0.052)	0.369(0.025)	0.368(0.044)	0.250(0.051)	0.284(0.012)
β_{11}	36.051(-0.902)	0.907(-0.183)	1.012(-0.228)	0.716(-0.436)	0.840(-0.498)	0.698(-0.396)	0.658(-0.349)
β_{12}	85.818(-0.726)	0.991(-0.018)	1.095(-0.255)	0.957(-0.483)	0.869(-0.528)	0.851(-0.460)	0.713(-0.414)
β_{20}	283.651(1.486)	0.757(0.098)	0.585(-0.017)	0.048(0.001)	0.072(-0.067)	0.041(-0.011)	0.047(-0.035)
β_{21}	30.042(1.057)	0.388(-0.030)	0.276(0.018)	0.043(0.008)	0.053(-0.002)	0.041(0.011)	0.053(0.000)
β_{22}	49.440(0.368)	0.241(0.020)	0.293(0.033)	0.043(-0.012)	0.060(-0.024)	0.044(-0.005)	0.054(-0.003)
π_1	0.069(0.243)	0.028(0.045)	0.034(0.097)	0.016(0.067)	0.024(0.086)	0.016(0.065)	0.020(0.068)
	Case III: $\varepsilon \sim t_3$						
β_{10}	0.600(-0.070)	0.199(0.096)	0.147(-0.017)	0.108(-0.024)	0.120(0.027)	0.111(-0.025)	0.117(0.024)
β_{11}	0.498(-0.173)	0.504(-0.465)	0.125(0.001)	0.113(0.009)	0.175(-0.037)	0.112(0.004)	0.148(-0.025)
β_{12}	0.790(-0.056)	0.227(-0.207)	0.103(-0.022)	0.132(-0.056)	0.153(-0.012)	0.136(-0.059)	0.148(-0.010)
β_{20}	3.143(-0.166)	0.023(-0.017)	0.018(0.005)	0.015(-0.005)	0.096(0.017)	0.015(-0.004)	0.096(0.017)
β_{21}	0.466(-0.021)	0.032(-0.001)	0.027(-0.027)	0.013(-0.006)	0.106(0.013)	0.013(-0.006)	0.104(0.010)
β_{22}	0.426(0.014)	0.026(-0.020)	0.028(-0.035)	0.018(-0.001)	0.386(-0.061)	0.018(-0.001)	0.386(-0.060)
π_1	0.033(0.024)	0.006(0.042)	0.005(0.033)	0.004(0.009)	0.006(0.012)	0.004(0.009)	0.006(0.012)
	Case IV: $\varepsilon \sim 0.95N(0,1) + 0.05N(0,25)$						
β_{10}	1.150(0.018)	0.217(0.131)	0.051(0.006)	0.076(-0.022)	0.078(0.021)	0.075(-0.023)	0.061(0.010)
β_{11}	0.859(0.017)	0.538(-0.499)	0.046(0.014)	0.084(-0.036)	0.067(-0.008)	0.084(-0.036)	0.062(0.004)
β_{12}	0.679(-0.114)	0.274(-0.303)	0.055(0.006)	0.073(-0.002)	0.078(0.002)	0.074(-0.002)	0.063(0.008)
β_{20}	0.348(0.061)	0.024(-0.011)	0.010(0.012)	0.016(-0.012)	0.013(-0.016)	0.016(-0.012)	0.012(-0.015)
β_{21}	0.043(0.074)	0.012(0.006)	0.009(0.001)	0.012(0.003)	0.013(0.009)	0.012(0.003)	0.012(0.009)
β_{22}	0.037(0.067)	0.013(-0.002)	0.010(-0.005)	0.012(0.005)	0.013(-0.015)	0.012(0.005)	0.012(-0.014)
π_1	0.016(-0.023)	0.005(0.024)	0.003(0.014)	0.003(0.004)	0.003(0.000)	0.003(0.004)	0.003(-0.001)
	Case V: $\varepsilon \sim N(0,1)$ with 5% high leverage outliers						
β_{10}	13.277(-2.232)	0.108(0.034)	0.048(-0.016)	1.745(0.011)	0.050(-0.012)	1.809(0.048)	0.050(-0.010)
β_{11}	4.653(1.470)	0.292(-0.315)	0.151(-0.081)	3.111(1.523)	0.055(-0.027)	3.030(1.501)	0.054(-0.024)
β_{12}	4.898(1.544)	0.139(-0.174)	0.167(-0.091)	2.967(1.475)	0.055(0.013)	3.008(1.495)	0.041(0.022)
β_{20}	15.087(2.656)	0.013(-0.014)	0.011(-0.004)	0.010(-0.009)	0.009(-0.003)	0.010(-0.009)	0.009(-0.003)
β_{21}	12.422(3.232)	0.010(0.005)	0.009(0.009)	0.031(0.134)	0.009(0.005)	0.031(0.134)	0.009(0.006)
β_{22}	13.629(3.408)	0.015(-0.011)	0.012(-0.001)	0.027(0.133)	0.011(-0.016)	0.027(0.134)	0.011(-0.015)
π_1	0.144(0.214)	0.003(0.010)	0.007(0.011)	0.008(-0.085)	0.002(0.006)	0.008(-0.085)	0.002(0.005)

Table 5.2: MSE(Bias) of Point Estimates for $n = 200$

	MLE	TLE	Bisquare	Mixregt	Mixregt-MCD	MixregP	MixregP-MCD
	Case I: $\varepsilon \sim N(0,1)$						
β_{10}	0.018(-0.006)	0.130(0.147)	0.020(-0.005)	0.019(0.004)	0.023(0.009)	0.019(0.004)	0.023(0.009)
β_{11}	0.020(0.002)	0.626(-0.602)	0.021(-0.001)	0.018(-0.014)	0.022(-0.010)	0.018(-0.014)	0.023(-0.011)
β_{12}	0.018(-0.006)	0.304(-0.381)	0.020(0.000)	0.016(0.008)	0.020(0.004)	0.016(0.008)	0.020(0.003)
β_{20}	0.004(0.003)	0.013(-0.011)	0.004(0.002)	0.005(-0.006)	0.005(0.012)	0.005(-0.006)	0.005(0.012)
β_{21}	0.004(0.004)	0.006(0.019)	0.004(0.002)	0.004(-0.009)	0.005(0.001)	0.004(-0.009)	0.005(0.001)
β_{22}	0.004(-0.005)	0.006(0.000)	0.004(-0.006)	0.005(-0.004)	0.004(-0.002)	0.005(-0.004)	0.004(-0.002)
π_1	0.001(0.000)	0.004(-0.010)	0.001(0.002)	0.001(0.001)	0.001(0.004)	0.001(0.001)	0.001(0.004)
	Case II: $\varepsilon \sim t_1$						
β_{10}	313.762(-0.917)	0.850(-0.032)	0.630(-0.088)	0.170(0.026)	0.142(-0.002)	0.126(0.029)	0.112(0.001)
β_{11}	278.218(-3.135)	0.579(-0.044)	0.686(-0.236)	0.538(-0.240)	0.541(-0.290)	0.475(-0.182)	0.393(-0.188)
β_{12}	455.182(-1.369)	0.565(0.047)	0.781(-0.193)	0.428(-0.219)	0.552(-0.269)	0.419(-0.202)	0.415(-0.171)
β_{20}	313.752(-0.917)	0.021(-0.003)	0.514(-0.049)	0.024(-0.010)	0.027(-0.002)	0.020(-0.015)	0.020(0.003)
β_{21}	269.681(-1.135)	0.046(0.003)	0.051(0.048)	0.016(0.009)	0.019(0.000)	0.016(0.015)	0.016(0.004)
β_{22}	453.685(0.630)	0.101(-0.002)	0.090(0.028)	0.017(0.012)	0.017(-0.011)	0.015(0.022)	0.013(-0.002)
π_1	0.062(0.249)	0.009(0.010)	0.017(0.067)	0.010(0.036)	0.011(0.046)	0.011(0.034)	0.008(0.030)
	Case III: $\varepsilon \sim t_3$						
β_{10}	0.311(0.029)	0.094(0.018)	0.038(-0.010)	0.039(-0.014)	0.050(-0.022)	0.039(-0.014)	0.049(-0.023)
β_{11}	0.308(-0.069)	0.168(-0.216)	0.044(0.049)	0.034(-0.013)	0.046(-0.008)	0.034(-0.013)	0.045(-0.007)
β_{12}	0.344(-0.031)	0.111(-0.146)	0.034(0.021)	0.046(0.000)	0.046(-0.021)	0.045(-0.001)	0.047(-0.021)
β_{20}	0.066(0.017)	0.010(-0.021)	0.007(-0.007)	0.006(-0.007)	0.006(0.010)	0.006(-0.007)	0.006(0.010)
β_{21}	0.070(0.056)	0.008(-0.001)	0.006(-0.025)	0.007(-0.005)	0.006(-0.009)	0.007(-0.004)	0.006(-0.009)
β_{22}	0.069(0.056)	0.009(-0.002)	0.008(-0.025)	0.008(0.009)	0.007(-0.003)	0.008(0.009)	0.007(-0.003)
π_1	0.011(-0.016)	0.002(0.024)	0.002(0.023)	0.002(0.004)	0.002(0.008)	0.002(0.004)	0.002(0.008)
	Case IV: $\varepsilon \sim 0.95N(0,1) + 0.05N(0,25)$						
β_{10}	0.098(0.000)	0.113(0.053)	0.024(0.004)	0.029(-0.007)	0.034(0.011)	0.029(-0.007)	0.034(0.011)
β_{11}	0.407(0.020)	0.374(-0.383)	0.021(0.027)	0.022(0.011)	0.033(0.001)	0.022(0.010)	0.033(0.001)
β_{12}	0.097(-0.056)	0.162(-0.240)	0.022(0.014)	0.026(0.001)	0.033(0.003)	0.027(0.001)	0.034(0.004)
β_{20}	0.041(0.015)	0.010(-0.004)	0.005(0.002)	0.006(-0.002)	0.005(0.006)	0.006(-0.001)	0.005(0.006)
β_{21}	0.089(0.047)	0.007(0.005)	0.005(-0.008)	0.006(0.006)	0.006(0.005)	0.006(0.005)	0.006(0.005)
β_{22}	0.135(0.042)	0.006(0.010)	0.004(0.000)	0.005(0.002)	0.006(0.004)	0.005(0.002)	0.006(0.004)
π_1	0.007(-0.033)	0.002(0.001)	0.001(0.006)	0.001(0.000)	0.001(-0.003)	0.001(0.000)	0.001(-0.003)
	Case V: $\varepsilon \sim N(0,1)$ with 5% high leverage outliers						
β_{10}	10.626(-1.807)	0.059(0.021)	0.020(-0.010)	1.369(0.227)	0.019(-0.017)	1.397(0.241)	0.019(-0.018)
β_{11}	5.047(1.626)	0.104(-0.198)	0.078(-0.037)	2.478(1.473)	0.020(0.002)	2.501(1.468)	0.020(0.001)
β_{12}	4.316(1.355)	0.059(-0.127)	0.073(-0.038)	2.606(1.514)	0.026(0.004)	2.645(1.518)	0.027(0.005)
β_{20}	13.231(2.446)	0.005(0.001)	0.005(0.005)	0.005(0.005)	0.004(0.004)	0.005(0.005)	0.004(0.004)
β_{21}	11.933(3.343)	0.006(0.007)	0.005(0.013)	0.021(0.125)	0.005(0.006)	0.022(0.126)	0.005(0.006)
β_{22}	12.296(3.396)	0.006(-0.003)	0.005(0.003)	0.020(0.122)	0.005(-0.002)	0.020(0.123)	0.005(-0.002)
π_1	0.146(0.202)	0.002(0.001)	0.005(-0.004)	0.008(-0.089)	0.001(0.004)	0.008(-0.089)	0.001(0.004)

Table 5.3: MSE(Bias) of Point Estimates for $n = 400$

Case	n	MixregP	MixregP-MCD
I:$\varepsilon \sim N(0, 1)$	100	12.157(15)	11.949(15)
	200	12.897(15)	13.160(15)
	400	14.019(15)	13.813(15)
II:$\varepsilon \sim t_1$	100	1.146(1.097)	1.156(1.097)
	200	1.106(1.097)	1.099(1.097)
	400	1.099(1.097)	1.097(1.097)
III:$\varepsilon \sim t_3$	100	2.862(2.090)	3.309(2.090)
	200	2.261(2.090)	2.286(2.090)
	400	2.035(2.090)	2.132(2.090)
IV:$\varepsilon \sim 0.95N(0, 1) + 0.05N(0, 25)$	100	3.582(2.090)	3.986(2.090)
	200	2.450(2.090)	2.571(2.090)
	400	2.301(2.090)	2.291(2.090)
V:$\varepsilon \sim N(0, 1)$ with 5% high leverage outliers	100	5.228(2.586)	12.264(15)
	200	3.232(2.586)	13.230(15)
	400	2.621(2.586)	14.114(15)

Table 5.4: The mean(median) of estimated degree of freedom by MixregP and MixregP-MCD.

Chapter 6

Conclusion

In this paper, we propose a robust estimation procedure for parametric regression models by assuming the error terms follow a Pearson type VII distribution. The estimation procedure is implemented by an EM algorithm based on the fact that the Pearson type VII distributions are a scale mixture of a normal distribution and a Gamma distribution. A trimmed version of proposed procedure, which is using MCD estimator for mean and covariance matrix, can efficiently trim the high leverage points away from the data. The simulation study shows that the proposed method is superior to or at least comparable to those estimation procedures we mentioned in the simulation study under all situations.

Bibliography

[1] Bai, X., Yao, W., and Boyer, J. E. (2012). Robust fitting of mixture regression models. *Computaional Statistics and Data Analysis*, 56, 2347–2359.

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

[3] McLachlan, G.J. (2000). Fininte mixture models. *New York: Wiley*.

[4] Neykov, N., Filzmoser, P., Dimova, R., and Neytchev, P. (2007). Robust fitting of mixtures using the trimmed likelihood estimator. *Computaional Statistics and Data Analysis*, 52, 299–308.

[5] Pearson, K. (1984). Contributions to the theory of mathematical evolution. *Philosophical Transactions of the Royal Society of London*, 185, 71–110.

[6] Peel, D. and McLachlan, G.J. (2000). Robust mixture modeling using the t distribution. *Statistics and Computation*, 10, 339–348.

[7] Pike, M.C. and Hill, I.D. (1966). Algorithm 291. *Communications of ACM*, 9(9), 684–687.

[8] Pison, G., Van Aelst, S. and Willems, G. (2002). Small sample corrections for LTS and MCD. *Metrika*, 55, 111–123.

[9] Rosseeuw, P.J. and Leroy, A.M. (1987). Robust regression and outlier detection. *Wiley-Interscience, New York*.

[10] Rosseeuw, P.J. and Van Zomeren, B.C. (1990). Unmasking multivariate outliers and leverate points. *Journal of the American Statistical Association*, 85, 633–639.

[11] Rosseeuw, P.J. and Van Driessen, K. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41, 212–223.

[12] Sun, J.Y. and J. M. Garibaldi (2010). Robust mixture modeling with Pearson type VII distribution. *Pattern Recognition Letters*, 31(16), 2447–2454.

[13] Wei, Y. (2012). Robust mixture regression models using t-distribution. *Master's report, Kansas State University.*

Appendix A

R code

```
library(MASS)
library(gregmisc)
library(robustbase)

## labswc.r
# solving label switching

slab=function(theta)
{
theta0=c(0.25,0,1,1,0.75,0,-1,-1)
ind1=c(1,2,3,4,5,6,7,8)
ind2=c(5,6,7,8,1,2,3,4)
res1=sum((theta[ind1]-theta0)^2)
res2=sum((theta[ind2]-theta0)^2)
theta=theta[ind1]*(res1<=res2)+theta[ind2]*(res1>res2)
return(theta)
}


## Kmixlin.r
#  the EM algorithm to fit the mixture of linear regression
#  mixlin estimates the mixture regression parameters by MLE
#  based on ONE initial value

mixlin=function(x,y,bet,sig,pr,m=2)
{
run=0;
n=length(y);
X=cbind(rep(1,n),x);
if(length(sig)>1 )
{ #the case when the variance is unequal
r=matrix(rep(0,m*n),nrow=n);
```

```
pk=r;
lh=0;
for(j in seq(m))
{r[,j]=y-X%*%bet[j,];
lh=lh+pr[j]*dnorm(r[,j],0,sig[j]);
}
lh=sum(log(lh));

#E-steps
repeat
{prest=c(bet,sig,pr);
run=run+1; plh=lh;
for(j in seq(m))
{pk[,j]=pr[j]*pmax(10^(-6),dnorm(r[,j],0,sig[j]))
}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);

#M-step
np=apply(pk,2,sum); pr=np/n; lh=0;
for(j in seq(m))
{
w=diag(pk[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]= y-X%*%bet[j,]; sig[j]=sqrt(t(pk[,j])%*%(r[,j]^2)/np[j]);
lh=lh+pr[j]*dnorm(r[,j],0,sig[j]);}
lh=sum(log(lh));
dif=lh-plh;
if(dif<1e-6|run>500){break}
}
}
else
{ #the case when the variance is equal
r=matrix(rep(0,m*n),nrow=n);pk=r; lh=0
for(j in seq(m))
{r[,j]=y-X%*%bet[j,];lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));

#E-steps
repeat
{prest=c(bet,sig,pr);run=run+1;plh=lh;
for(j in seq(m))
{pk[,j]=pr[j]* pmax(10^(-6),dnorm(r[,j],0,sig))}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);

#M-step
```

```
np=apply(pk,2,sum);pr=np/n;
for(j in seq(m))
{w=diag(pk[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]= y-X%*%bet[j,]; }
sig=sqrt(sum(pk*(r^2))/n);lh=0;
for(j in seq(m))
{
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));
dif=lh-plh;
if((dif<1e-6)|(run>500)){break}}
sig=sig*rep(1,m)}
est=list(theta= matrix(c(bet,sig,pr),nrow=m),likelihood=lh,
run=run,diflh=dif)
est}

#MLE Based on Normality Assumption
# EM Algorithm

mix.MLE.normal=function(x,y,beta,sigma,prob,group,tol=1e-6)
{n=length(y);X=cbind(rep(1,n),x);run=1;
if(length(sigma)>1)
{gsumi=matrix(0,n,group);
for(i in seq(group))
{gsumi[,i]=prob[i]*dnorm(y,X%*%beta[i,],sigma[i]);}
prest=sum(log(apply(gsumi,1,sum)));
repeat
{ tji=matrix(0,n,group)
for(i in seq(group))
{tji[,i]=prob[i]*pmax(dnorm(y,X%*%beta[i,],sigma[i]),10^(-6));}
tji=solve(diag(apply(tji,1,sum)))%*%tji;
prob=apply(tji,2,mean);
sse=rep(0,group)
for(i in seq(group))
{beta[i,]=lm(y~x,weight=tji[,i])$coefficient;
sse[i]=sum(tji[,i]*(y-X%*%beta[i,])^2);
sigma[i]=sqrt(sse[i]/sum(tji[,i]))}
sigma2=sigma^2;
for(i in seq(group))
{gsumi[,i]=prob[i]*dnorm(y,X%*%beta[i,],sigma[i]);}
latest=sum(log(apply(gsumi,1,sum)));
dif=abs(latest-prest);
prest=latest;
run=run+1;
```

```
if(dif<=tol|run>500){break}}}
else
{gsumi=matrix(0,n,group);
for(i in seq(group))
{gsumi[,i]=prob[i]*dnorm(y,X%*%beta[i,],sigma);}
prest=sum(log(apply(gsumi,1,sum)));

repeat
{tji=matrix(0,n,group)
for(i in seq(group))
{tji[,i]=prob[i]*pmax(dnorm(y,X%*%beta[i,],sigma),10^(-6));}
tji=solve(diag(apply(tji,1,sum)))%*%tji;
prob=apply(tji,2,mean);
sse=rep(0,group);
for(i in seq(group))
{beta[i,]=lm(y~x,weight=tji[,i])$coefficient;
sse[i]=sum(tji[,i]*(y-X%*%beta[i,])^2);}
sigma2=sum(sse)/n;
sigma=sqrt(sigma2);
for(i in seq(group))
{gsumi[,i]=prob[i]*dnorm(y,X%*%beta[i,],sigma);}
latest=sum(log(apply(gsumi,1,sum)));
dif=abs(latest-prest);
prest=latest; run=run+1;
if(dif<=tol|run>500){break}}}

btemp=c(prob[1],beta[1,],prob[2],beta[2,]);
btemp=slab(btemp);
prob=c(btemp[1],btemp[5]);
beta=rbind(btemp[2:4],btemp[6:8]);

result=cbind(beta,sigma2,prob)
result}

##Ctle.r
# Trimmed MLE

trimmix=function(x,y,k=2,alpha=0.9,bet,sig,pr,acc=1e-6)
{n=length(y);
n1=round(n*alpha);
x=matrix(x,nrow=n);
a=dim(x);p=a[2]+1;X=cbind(rep(1,n),x);
if(dim(bet)[2]==k){bet=t(bet)};lh=0
for (i in seq(k))
{lh=lh+pr[i]*dnorm(y-X%*%bet[i,],0,sig[1])}
```

```
ind=order(-lh);run=0;
obj=sum(log(lh[ind[1:n1]]));
repeat
{
pobj=obj;run=run+1; x1=x[ind[1:n1],];
y1=y[ind[1:n1]];fit=mixlin(x1,y1,bet,sig,pr,k);
fit=fit$theta;bet=matrix(fit[1:(p*k)],nrow=k);
sig=fit[p*k+1];
pr=fit[(p*k+k+1):(p*k+2*k)];lh=0;
for(i in seq(k))
{lh=lh+pr[i]*dnorm(y-X%*%bet[i,],0,sig[1]);}
ind=order(-lh);
obj=sum(log(lh[ind[1:n1]]));dif=obj-pobj;
if(dif<acc|run>50){break}}
if(length(sig)<2){sig=rep(sig,k)};

btemp=c(pr[1],bet[1,],pr[2],bet[2,]);
btemp=slab(btemp);
pr=c(btemp[1],btemp[5]);
bet=rbind(btemp[2:4],btemp[6:8]);

theta=matrix(c(bet,sig,pr),nrow=k);
est=list(theta=theta,likelihood=obj,diflikelihood=dif,run=run)
est}

## Dbisquare.r
# Robust Mixture Regression Robust Based on Bisquare Function

bisquare=function(t,k=4.685)
{out=t*pmax(0,(1-(t/k)^2))^2;out}

biscalew=function(t)
{t[which(t==0)]=min(t[which(t!=0)])/10;
out=pmin(1-(1-t^2/1.56^2)^3,1)/t^2;out}

mixbi2<-function(x,y,bet,sig,pr,m=2)
{run=0;acc=10^(-4)*max(abs(c(bet,sig,pr)));
n=length(y);X=cbind(rep(1,n),x);
p=dim(X)[2];
if(length(sig)>1)
{r=matrix(rep(0,m*n),nrow=n);
pk=r;for(j in seq(m))
{r[,j]=(y-X%*%bet[j,])/sig[j];}

#E-steps
```

```
repeat
{prest=c(sig,bet,pr);run=run+1;
for(j in seq(m))
{pk[,j]=pr[j]*pmax(10^(-6),dnorm(r[,j],0,1))/sig[j]}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);

#M-step
np=apply(pk,2,sum);pr=np/n;
r[which(r==0)]=min(r[which(r!=0)])/10;
for(j in seq(m))
{w=diag(pk[,j]*bisquare(r[,j])/r[,j]);
bet[j,]= solve(t(X)%*%w%*%X+10^(-3)*diag(rep(1,p)))%*%t(X)%*%w%*%y;
r[,j]= (y-X%*%bet[j,])/sig[j];
sig[j]=sqrt(sum(r[,j]^2*sig[j]^2*pk[,j]*biscalew(r[,j]))/np[j]/0.5);}
dif=max(abs(c(sig,bet,pr)-prest))
if(dif<acc|run>500){break}}
else
{r=matrix(rep(0,m*n),nrow=n);pk=r;
for(j in seq(m))
{r[,j]=(y-X%*%bet[j,])/sig;}

#E-steps
repeat
{prest=c(sig,bet,pr);run=run+1;
for(j in seq(m))
{pk[,j]=pr[j]*pmax(10^(-6),dnorm(r[,j],0,1))/sig}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);

#M-step
np=apply(pk,2,sum);
pr=np/n;
r[which(r==0)]=min(r[which(r!=0)])/10;
for(j in seq(m))
{w=diag(pk[,j]*bisquare(r[,j])/r[,j]);
bet[j,]=solve(t(X)%*%w%*%X+10^(-3)*diag(rep(1,p)))%*%t(X)%*%w%*%y;
r[,j]=(y-X%*%bet[j,])/sig;}
sig=sqrt(sum(pk*(r^2*sig[1]^2)*biscalew(r))/n/0.5)
dif=max(abs(c(sig,bet,pr)-prest))
if(dif<acc|run>500){break}}
sig=rep(sig,m);}

btemp=c(pr[1],bet[1,],pr[2],bet[2,]);
btemp=slab(btemp);
pr=c(btemp[1],btemp[5]);
bet=rbind(btemp[2:4],btemp[6:8]);
```

```
theta=matrix(c(bet,sig,pr),nrow=m);
est=list(theta=theta,difpar=dif,run=run)
est}

## MCD&SD
lev.p=function(X,method="classic")
{if(method=="sd")
{
# Stahel-Donoho Estimator
# Package needed: rrcov
Temp=CovSde(X)
mx=Temp@center
cx=Temp@cov}
else if(method=="mcd")
{# Fast MCD Estimate
# Package Needed: robustbase
Temp=covMcd(X)
mx=Temp$center
cx=Temp$cov }
else
{# Sample mean and variance
mx=apply(X,2,mean)
cx=var(X)}
list(mean=mx,covar=cx)}

## MLE, Bisquare and TLE

BV=array(0,dim=c(42,6,3));
total=200
kk=1;
for(n in c(100,200,400))
{jj=1;
for(dist in seq(6))
{set.seed(88888)
b101=b111=b121=b201=b211=b221=pi11=rep(0,total)
b102=b112=b122=b202=b212=b222=pi12=rep(0,total)
b103=b113=b123=b203=b213=b223=pi13=rep(0,total)
for(k in seq(total))
{g=2;              # Number of Groups
u=runif(n,0,1);   # A random number for assigning groups
p1=(u<=0.25);     # probability of group 1
p2=1-p1;          # probability of group 2

x1=rnorm(n,0,1);  # 1: normal,
x2=rnorm(n,0,1);
```

```
e1n=rnorm(n,0,1);
e2n=rnorm(n,0,1);

e1L=rexp(n,sqrt(2))-rexp(n,sqrt(2)); # 2: laplace,
e2L=rexp(n,sqrt(2))-rexp(n,sqrt(2));

e1t1=rt(n,1);   # 3: t(1),
e2t1=rt(n,1);

e1t3=rt(n,3);   # 4: t(3),
e2t3=rt(n,3);

uu=runif(n,0,1)
e1mn=(uu<0.95)*rnorm(n,0,1)+(uu>=0.95)*rnorm(n,0,5)  # 5: normal_mixture,
e2mn=(uu<0.95)*rnorm(n,0,1)+(uu>=0.95)*rnorm(n,0,5)

e1=e1n*((dist==1)|(dist==6))+e1L*(dist==2)+e1t1*(dist==3)
+e1t3*(dist==4)+e1mn*(dist==5)
e2=e2n*((dist==1)|(dist==6))+e2L*(dist==2)+e2t1*(dist==3)
 +e2t3*(dist==4)+e2mn*(dist==5)

y1=0+x1+x2+e1;
y2=0-x1-x2+e2;

x=cbind(x1,x2);
y=y1*p1+y2*p2;

if(dist==6)   # 6: normal+0.05outlier
{
nout=round(n*0.95)+1;
x[nout:n,]=20;
y[nout:n]=100;
}

# Initial Values

prob=c(0.5,0.5);
b1=c(-0.5,-0.9,-0.9);
b2=c(0.5,-0.5,0.5);
beta=rbind(b1,b2);

clust=kmeans(y,2)
sig01=sd(y[clust$cluster==1])
sig02=sd(y[clust$cluster==2])
```

```
sig0=ifelse((is.na(sig01)|(sig01==0)),sig02,sig01)
group=2;

# MLE
result=mix.MLE.normal(x,y,beta,sig0,prob,group,1e-6);
b101[k]=result[1,1]
b111[k]=result[1,2]
b121[k]=result[1,3]
b201[k]=result[2,1]
b211[k]=result[2,2]
b221[k]=result[2,3]
pi11[k]=result[1,5]

# TLE
result=trimmix(x,y,2,0.9,beta,sig0,prob,1e-6)
b102[k]=result$theta[1,1]
b112[k]=result$theta[1,2]
b122[k]=result$theta[1,3]
b202[k]=result$theta[2,1]
b212[k]=result$theta[2,2]
b222[k]=result$theta[2,3]
pi12[k]=result$theta[1,5]

# Bisquare
result=mixbi2(x,y,beta,sig0,prob,group);
b103[k]=result$theta[1,1]
b113[k]=result$theta[1,2]
b123[k]=result$theta[1,3]
b203[k]=result$theta[2,1]
b213[k]=result$theta[2,2]
b223[k]=result$theta[2,3]
pi13[k]=result$theta[1,5]
}

tb10=0; tb11=1;  tb12=1;
tb20=0; tb21=-1; tb22=-1; tpi1=0.25;

# MLE

Rmle=cbind(b101-tb10,b111-tb11,b121-tb12,b201-tb20,b211-tb21,
b221-tb22,pi11-tpi1);
BV[seq((jj-1)*7+1,jj*7),1:2,kk]
=cbind(apply(Rmle,2,function(x){mean(x^2)}),apply(Rmle,2,mean))

# TML
```

```
Rtml=cbind(b102-tb10,b112-tb11,b122-tb12,b202-tb20,b212-tb21,
b222-tb22,pi12-tpi1);
BV[seq((jj-1)*7+1,jj*7),3:4,kk]
=cbind(apply(Rtml,2,function(x){mean(x^2)}),apply(Rtml,2,mean))

# Bisquare

Rbsq=cbind(b103-tb10,b113-tb11,b123-tb12,b203-tb20,b213-tb21,
b223-tb22,pi13-tpi1);
BV[seq((jj-1)*7+1,jj*7),5:6,kk]
=cbind(apply(Rbsq,2,function(x){mean(x^2)}),apply(Rbsq,2,mean))
jj=jj+1; }
kk=kk+1;}
ending=Sys.time()
ending-begining

## Robust Mixture Regression Based on t-Distribution
# Definition of t density
dent=function(y,mu,sig,v)
{est=gamma((v+1)/2)*sig^(-1)/((pi*v)^(1/2)*gamma(v/2)*
(1+(y-mu)^2/(sig^2*v))^(0.5*(v+1)));
est}

BV=matrix(0,nrow=42,ncol=2)
n=200    # Sample Size
total=200
for(dist in seq(6))
{
set.seed(98765)
b107=b117=b127=b207=b217=b227=pi17=rep(0,total)
for(k in seq(total))
{repeat{
u=runif(n,0,1);   # A random number for assigning groups
p1=(u<=0.25);     # probability of group 1
p2=1-p1;          # probability of group 2

x1=rnorm(n,0,1);
x2=rnorm(n,0,1);

e1n=rnorm(n,0,1);  # 1: normal,
e2n=rnorm(n,0,1);

e1L=rexp(n,sqrt(2))-rexp(n,sqrt(2)); # 2: laplace,
e2L=rexp(n,sqrt(2))-rexp(n,sqrt(2));
```

```
e1t1=rt(n,1);   # 3: t(1),
e2t1=rt(n,1);

e1t3=rt(n,3);   # 4: t(3),
e2t3=rt(n,3);

u=runif(n,0,1)
e1mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)  # 5: normal_mixture,
e2mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)

e1=e1n*((dist==1)|(dist==6))+e1L*(dist==2)+e1t1*(dist==3)
+e1t3*(dist==4)+e1mn*(dist==5)
e2=e2n*((dist==1)|(dist==6))+e2L*(dist==2)+e2t1*(dist==3)
+e2t3*(dist==4)+e2mn*(dist==5)

y1=0+x1+x2+e1;
y2=0-x1-x2+e2;

x=cbind(x1,x2);
y=y1*p1+y2*p2;

if(dist==6)   # 6: normal+0.05outlier
{nout=round(n*0.95)+1;
x[nout:n,]=20;y[nout:n]=100;}

# Initial Values
prob=c(0.5,0.5);
b1=c(-0.5,-0.9,-0.9);
b2=c(0.5,-0.5,0.5);
beta=rbind(b1,b2);
clust=kmeans(y,2)
sig01=sd(y[clust$cluster==1])
sig02=sd(y[clust$cluster==2])
if(is.na(sig01)){sig01=0};
if(is.na(sig02)){sig02=0};
if((sig01==0)|(sig02=0)){sig0=max(sig01,sig02)} else
{sig0=(sig01+sig02)/2};
group=2;

X=cbind(rep(1,n),x);
m=2;maxv=15;
likhod=flag=rep(0,15);
theta=array(0,dim=c(2,5,15));
for(v in seq(maxv))
```

```
{bet=beta; sig=sig0; pr=prob; run=0;
n=length(y);
X=cbind(rep(1,n),x);
lh=-10^10;a=dim(x);p=a[2]+1;
pk=u=r=matrix(rep(0,m*n),nrow=n);
for(j in seq(m))
{r[,j]=(y-X%*%bet[j,])/sig;}
repeat
{run=run+1;prelh=lh;
for(j in seq(m))
{pk[,j]=pr[j]*dent(y, X%*%bet[j,],sig,v);
u[,j]=(v+1)/(v+r[,j]^2)}
lh=sum(log(apply(pk,1,sum)));
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
dif=lh-prelh;
if(!is.finite(lh)){flag[v]=1; break}
if(dif<1e-6|run>500){flag[v]=0; break}
np=apply(pk,2,sum);pr=np/n;
for(j in seq(m))
{w=diag(pk[,j]*u[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]=y-X%*%bet[j,];}
sig=sqrt(sum(pk*r^2*u)/sum(pk));
r=r/sig;}
sig=sig*rep(1,m);
theta[,,v]= matrix(c(bet,pr,sig),nrow=m);
likhod[v]=lh;}
if(all(flag==0)){break}}

fv=which(likhod==max(likhod))
if(fv<15)
{result=theta[,,fv]} else
{run=0;n=length(y);X=cbind(rep(1,n),x);
r=matrix(rep(0,m*n),nrow=n);pk=r;lh=0
for(j in seq(m))
{r[,j]=y-X%*%bet[j,];
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));

repeat
{prest=c(bet,sig,pr);run=run+1;plh=lh;
for(j in seq(m))
{pk[,j]=pr[j]* pmax(10^(-6),dnorm(r[,j],0,sig))}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
np=apply(pk,2,sum);pr=np/n;
```

```
for(j in seq(m))
{w=diag(pk[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]= y-X%*%bet[j,];}
sig=sqrt(sum(pk*(r^2))/n);lh=0;
for(j in seq(m))
{lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));
dif=lh-plh;
if((dif<1e-6)|(run>500)){break}}
sig=sig*rep(1,m)
result=matrix(c(bet,pr,sig),nrow=m}

btemp=c(result[1,4], result[1,1:3], result[2,4], result[2,1:3]);
btemp=slab(btemp);
prob=c(btemp[1],btemp[5]);
beta=rbind(btemp[2:4],btemp[6:8]);
rest7=cbind(beta,prob)

b107[k]=rest7[1,1]
b117[k]=rest7[1,2]
b127[k]=rest7[1,3]
b207[k]=rest7[2,1]
b217[k]=rest7[2,2]
b227[k]=rest7[2,3]
pi17[k]=rest7[1,4]}

tb10=0;  tb11=1;   tb12=1;
tb20=0;  tb21=-1;   tb22=-1;tpi1=0.25;

# Mixregt

RL=cbind(b107-tb10,b117-tb11,b127-tb12,b207-tb20,b217-tb21,
b227-tb22,pi17-tpi1);
BV[seq((dist-1)*7+1,dist*7),1:2]=cbind(apply(RL,2,function(x){mean(x^2)}),
apply(RL,2,mean))
print(BV[seq((dist-1)*7+1,dist*7),1:2])}
round(BV,3)

## Robust Mixture Regression Based on t-Distribution with MCD
# Definition of t density
dent=function(y,mu,sig,v)
{est=gamma((v+1)/2)*sig^(-1)/((pi*v)^(1/2)*gamma(v/2)*
(1+(y-mu)^2/(sig^2*v))^(0.5*(v+1)));
est}
```

```
BV=matrix(0,nrow=42,ncol=2)
n=200;    # Sample Size
total=200;
for(dist in seq(6))
{set.seed(98765)
b107=b117=b127=b207=b217=b227=pi17=rep(0,total)
for(k in seq(total))
{repeat
{n=200;u=runif(n,0,1); # A random number for assigning groups
p1=(u<=0.25);     # probability of group 1
p2=1-p1;   # probability of group 2

x1=rnorm(n,0,1); x2=rnorm(n,0,1);

e1n=rnorm(n,0,1);  # 1: normal,
e2n=rnorm(n,0,1);

e1L=rexp(n,sqrt(2))-rexp(n,sqrt(2)); # 2: laplace,
e2L=rexp(n,sqrt(2))-rexp(n,sqrt(2));

e1t1=rt(n,1);   # 3: t(1),
e2t1=rt(n,1);

e1t3=rt(n,3);   # 4: t(3),
e2t3=rt(n,3);

u=runif(n,0,1)
e1mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)  # 5: normal_mixture,
e2mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)

e1=e1n*((dist==1)|(dist==6))+e1L*(dist==2)+e1t1*(dist==3)
e1t3*(dist==4)+e1mn*(dist==5)
e2=e2n*((dist==1)|(dist==6))+e2L*(dist==2)+e2t1*(dist==3)
+e2t3*(dist==4)+e2mn*(dist==5)

y1=0+x1+x2+e1; y2=0-x1-x2+e2;
x=cbind(x1,x2); y=y1*p1+y2*p2;

if(dist==6)   # 6: normal+0.05outlier
{
nout=round(n*0.95)+1;
x[nout:n,]=20;
y[nout:n]=100;
}
```

```
# Initial Values
prob=c(0.5,0.5);
b1=c(-0.5,-0.9,-0.9);
b2=c(0.5,-0.5,0.5);
beta=rbind(b1,b2);
clust=kmeans(y,2)
sig01=sd(y[clust$cluster==1])
sig02=sd(y[clust$cluster==2])
if(is.na(sig01)){sig01=0};
if(is.na(sig02)){sig02=0};
if((sig01==0)|(sig02=0)){sig0=max(sig01,sig02)} else
{sig0=(sig01+sig02)/2};group=2;

mv=lev.p(x,method="mcd");
TX=cbind(x[,1]-mv$mean[1],x[,2]-mv$mean[2]);
ind=which(diag(TX%*%solve(mv$covar)%*%t(TX))>=qchisq(0.975,2));
if(length(ind)>0)
{x=x[-ind,];y=y[-ind];}
x1=x[,1];x2=x[,2];n=length(y);
X=cbind(rep(1,n),x);m=2;maxv=15;

likhod=flag=rep(0,15);
theta=array(0,dim=c(2,5,15));
for(v in seq(maxv))
{bet=beta
sig=sig0;
pr=prob;
run=0;
n=length(y);
X=cbind(rep(1,n),x);
lh=-10^10;
a=dim(x);
p=a[2]+1;
pk=u=r=matrix(rep(0,m*n),nrow=n);
for(j in seq(m))
{r[,j]=(y-X%*%bet[j,])/sig;}
repeat{
run=run+1;
prelh=lh;
for(j in seq(m)){
pk[,j]=pr[j]*dent(y, X%*%bet[j,],sig,v);
u[,j]=(v+1)/(v+r[,j]^2)}
lh=sum(log(apply(pk,1,sum)));
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
dif=lh-prelh;
```

```
if(!is.finite(lh)){flag[v]=1; break}
if(dif<1e-6|run>500){flag[v]=0; break}

np=apply(pk,2,sum);pr=np/n;
for(j in seq(m)){
w=diag(pk[,j]*u[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]=y-X%*%bet[j,];}
sig=sqrt(sum(pk*r^2*u)/sum(pk));
r=r/sig;}
sig=sig*rep(1,m);
theta[,,v]= matrix(c(bet,pr,sig),nrow=m);likhod[v]=lh;}
if(all(flag==0)){break}}
fv=which(likhod==max(likhod))
if(fv<15)
{result=theta[,,fv]} else
{run=0;n=length(y);X=cbind(rep(1,n),x);
r=matrix(rep(0,m*n),nrow=n);pk=r;lh=0
for(j in seq(m))
{r[,j]=y-X%*%bet[j,];
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));
repeat
{prest=c(bet,sig,pr);run=run+1;plh=lh;
for(j in seq(m)){
pk[,j]=pr[j]* pmax(10^(-6),dnorm(r[,j],0,sig))}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
np=apply(pk,2,sum);pr=np/n;
for(j in seq(m)){
w=diag(pk[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]= y-X%*%bet[j,];}
sig=sqrt(sum(pk*(r^2))/n);lh=0;
for(j in seq(m)){
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));dif=lh-plh;
if((dif<1e-6)|(run>500)){break}}
sig=sig*rep(1,m)
result=matrix(c(bet,pr,sig),nrow=m)}
btemp=c(result[1,4], result[1,1:3], result[2,4], result[2,1:3]);
btemp=slab(btemp);
prob=c(btemp[1],btemp[5]);
beta=rbind(btemp[2:4],btemp[6:8]);
rest7=cbind(beta,prob)
```

```
b107[k]=rest7[1,1]
b117[k]=rest7[1,2]
b127[k]=rest7[1,3]
b207[k]=rest7[2,1]
b217[k]=rest7[2,2]
b227[k]=rest7[2,3]
pi17[k]=rest7[1,4]}
tb10=0;  tb11=1;   tb12=1;
tb20=0;  tb21=-1;   tb22=-1;tpi1=0.25;
# Mixregt_MCD
RL=cbind(b107-tb10,b117-tb11,b127-tb12,b207-tb20,b217-tb21,
b227-tb22,pi17-tpi1);
BV[seq((dist-1)*7+1,dist*7),1:2]=cbind(apply(RL,2,function(x){mean(x^2)}),
apply(RL,2,mean))
print(BV[seq((dist-1)*7+1,dist*7),1:2])}
round(BV,3)

## Robust Mixture Regression By Type VII Distribution
# Definition of type vii density
# Definition of type vii density

den=function(y,mu,sig,v){
est=gamma(v)*sig^(-1)/((pi)^(1/2)*gamma(v-1/2)*
(1+(y-mu)^2/sig^2)^v);
est}
BV=matrix(0,nrow=42,ncol=2)
n=400    # Sample Size
total=200
for(dist in seq(6)){
set.seed(98765)
b107=b117=b127=b207=b217=b227=pi17=df7=rep(0,total)
for(k in seq(total)){
repeat{
u=runif(n,0,1);   # A random number for assigning groups
p1=(u<=0.25);     # probability of group 1
p2=1-p1;          # probability of group 2

x1=rnorm(n,0,1);
x2=rnorm(n,0,1);

e1n=rnorm(n,0,1);  # 1: normal,
e2n=rnorm(n,0,1);

e1L=rexp(n,sqrt(2))-rexp(n,sqrt(2)); # 2: laplace,
e2L=rexp(n,sqrt(2))-rexp(n,sqrt(2));
```

```
e1t1=rt(n,1);   # 3: t(1),
e2t1=rt(n,1);

e1t3=rt(n,3);   # 4: t(3),
e2t3=rt(n,3);

u=runif(n,0,1)
e1mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)  # 5: normal_mixture,
e2mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)

e1=e1n*((dist==1)|(dist==6))+e1L*(dist==2)+e1t1*(dist==3)
+e1t3*(dist==4)+e1mn*(dist==5)
e2=e2n*((dist==1)|(dist==6))+e2L*(dist==2)+e2t1*(dist==3)
+e2t3*(dist==4)+e2mn*(dist==5)

y1=0+x1+x2+e1;y2=0-x1-x2+e2;
x=cbind(x1,x2);y=y1*p1+y2*p2;

if(dist==6)   # 6: normal+0.05outlier
{nout=round(n*0.95)+1;
x[nout:n,]=20;y[nout:n]=100;}
# Initial Values
prob=c(0.5,0.5);
b1=c(-0.5,-0.9,-0.9);
b2=c(0.5,-0.5,0.5);
beta=rbind(b1,b2);
clust=kmeans(y,2)
sig01=sd(y[clust$cluster==1])
sig02=sd(y[clust$cluster==2])
if(is.na(sig01)){sig01=0};
if(is.na(sig02)){sig02=0};
if((sig01==0)|(sig02=0)){sig0=max(sig01,sig02)} else
{sig0=(sig01+sig02)/2};
group=2;
X=cbind(rep(1,n),x);m=2;
maxv=seq(0.6,15,length=30);
likhod=flag=rep(0,30);
theta=array(0,dim=c(2,5,30));vv=1;
for(v in maxv){
bet=beta;sig=sig0;pr=prob;
run=0;n=length(y);X=cbind(rep(1,n),x);
lh=-10^10;a=dim(x);p=a[2]+1;
pk=u=r=matrix(rep(0,m*n),nrow=n);
```

```
for(j in seq(m)){
r[,j]=(y-X%*%bet[j,])/sig;}

repeat{
run=run+1;prelh=lh;for(j in seq(m)){
pk[,j]=pr[j]*den(y, X%*%bet[j,],sig,v);
u[,j]=(2*v)/(1+r[,j]^2)}
lh=sum(log(apply(pk,1,sum)));
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
dif=lh-prelh;
if(!is.finite(lh)){flag[v]=1; break}
if(dif<1e-6|run>500){flag[v]=0; break}

np=apply(pk,2,sum);pr=np/n;
for(j in seq(m))
{ w=diag(pk[,j]*u[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]=y-X%*%bet[j,];}
sig=sqrt(sum(pk*r^2*u)/sum(pk));r=r/sig;}
sig=sig*rep(1,m);
theta[,,vv]= matrix(c(bet,pr,sig),nrow=m);
likhod[vv]=lh;vv=vv+1;}
if(all(flag==0)){break}}

fv=which(likhod==max(likhod))
if(maxv[fv]<15)
{result=theta[,,fv];df=maxv[fv];} else
{df=15;
run=0;n=length(y);
X=cbind(rep(1,n),x);
r=matrix(rep(0,m*n),nrow=n);pk=r;lh=0
for(j in seq(m)){
r[,j]=y-X%*%bet[j,];
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));

repeat{
prest=c(bet,sig,pr);
run=run+1;plh=lh;
for(j in seq(m)){
pk[,j]=pr[j]* pmax(10^(-6),dnorm(r[,j],0,sig))}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
np=apply(pk,2,sum);pr=np/n;
for(j in seq(m)){w=diag(pk[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
```

```
r[,j]= y-X%*%bet[j,];}
sig=sqrt(sum(pk*(r^2))/n);
lh=0;
for(j in seq(m)){
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));dif=lh-plh;
if((dif<1e-6)|(run>500)){break}}
sig=sig*rep(1,m)
result=matrix(c(bet,pr,sig),nrow=m)}

btemp=c(result[1,4], result[1,1:3], result[2,4], result[2,1:3]);
btemp=slab(btemp);
prob=c(btemp[1],btemp[5]);
beta=rbind(btemp[2:4],btemp[6:8]);
rest7=cbind(beta,prob)

b107[k]=rest7[1,1]
b117[k]=rest7[1,2]
b127[k]=rest7[1,3]
b207[k]=rest7[2,1]
b217[k]=rest7[2,2]
b227[k]=rest7[2,3]
pi17[k]=rest7[1,4]
df7[k]=df}

tb10=0;  tb11=1;   tb12=1;
tb20=0;  tb21=-1;   tb22=-1; tpi1=0.25;

RL=cbind(b107-tb10,b117-tb11,b127-tb12,b207-tb20,b217-tb21,
b227-tb22,pi17-tpi1);
BV[seq((dist-1)*7+1,dist*7),1:2]=cbind(apply(RL,2,function(x){mean(x^2)}),
apply(RL,2,mean))
print(BV[seq((dist-1)*7+1,dist*7),1:2])

# Degree of freedom
dff=cbind(mean(df7),median(df7))
print(dff)}
round(BV,3)

## Robust Mixture Regression By Type VII Distribution with MCD
# Definition of type vii density

den=function(y,mu,sig,v)
{est=gamma(v)*sig^(-1)/((pi)^(1/2)*gamma(v-1/2)*
(1+(y-mu)^2/sig^2)^v);est}
```

```
BV=matrix(0,nrow=42,ncol=2)
n=200;    # Sample Size
total=200;
for(dist in seq(6)){
set.seed(98765)
b107=b117=b127=b207=b217=b227=pi17=df7=rep(0,total)
for(k in seq(total)){
repeat{
n=200;u=runif(n,0,1);   # A random number for assigning groups
p1=(u<=0.25);     # probability of group 1
p2=1-p1;          # probability of group 2

x1=rnorm(n,0,1);x2=rnorm(n,0,1);

e1n=rnorm(n,0,1);  # 1: normal,
e2n=rnorm(n,0,1);

e1L=rexp(n,sqrt(2))-rexp(n,sqrt(2)); # 2: laplace,
e2L=rexp(n,sqrt(2))-rexp(n,sqrt(2));

e1t1=rt(n,1);   # 3: t(1),
e2t1=rt(n,1);

e1t3=rt(n,3);   # 4: t(3),
e2t3=rt(n,3);

u=runif(n,0,1)
e1mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)  # 5: normal_mixture,
e2mn=(u<0.95)*rnorm(n,0,1)+(u>=0.95)*rnorm(n,0,5)

e1=e1n*((dist==1)|(dist==6))+e1L*(dist==2)+e1t1*(dist==3)
+e1t3*(dist==4)+e1mn*(dist==5)
e2=e2n*((dist==1)|(dist==6))+e2L*(dist==2)+e2t1*(dist==3)
+e2t3*(dist==4)+e2mn*(dist==5)

y1=0+x1+x2+e1;y2=0-x1-x2+e2;
x=cbind(x1,x2);y=y1*p1+y2*p2;

if(dist==6)   # 6: normal+0.05outlier
{nout=round(n*0.95)+1;
x[nout:n,]=20;y[nout:n]=100;}

# Initial Values
prob=c(0.5,0.5);
```

```
b1=c(0.1,0.9,0.9);
b2=c(-0.1,-0.9,-0.9);
beta=rbind(b1,b2);
clust=kmeans(y,2)
sig01=sd(y[clust$cluster==1])
sig02=sd(y[clust$cluster==2])
if(is.na(sig01)){sig01=0};
if(is.na(sig02)){sig02=0};
if((sig01==0)|(sig02=0)){sig0=max(sig01,sig02)} else
{sig0=(sig01+sig02)/2};
group=2;
mv=lev.p(x,method="mcd");
TX=cbind(x[,1]-mv$mean[1],x[,2]-mv$mean[2]);
ind=which(diag(TX%*%solve(mv$covar)%*%t(TX))>=qchisq(0.975,2));
if(length(ind)>0){
x=x[-ind,];y=y[-ind];}
x1=x[,1];x2=x[,2];
n=length(y);X=cbind(rep(1,n),x);
m=2;maxv=seq(0.6,15,length=30);
likhod=flag=rep(0,30);
theta=array(0,dim=c(2,5,30));vv=1;
for(v in maxv){
bet=beta;sig=sig0;
pr=prob;run=0;n=length(y);
X=cbind(rep(1,n),x);lh=-10^10;
a=dim(x);p=a[2]+1;
pk=u=r=matrix(rep(0,m*n),nrow=n);

for(j in seq(m)){
r[,j]=(y-X%*%bet[j,])/sig;}
repeat{run=run+1;prelh=lh;
for(j in seq(m)){
pk[,j]=pr[j]*den(y, X%*%bet[j,],sig,v);u[,j]=(2*v)/(1+r[,j]^2)}
lh=sum(log(apply(pk,1,sum)));
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
dif=lh-prelh;
if(!is.finite(lh)){flag[v]=1; break}
if(dif<1e-6|run>500){flag[v]=0; break}
np=apply(pk,2,sum);pr=np/n;
for(j in seq(m)){
w=diag(pk[,j]*u[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]=y-X%*%bet[j,];}
sig=sqrt(sum(pk*r^2*u)/sum(pk));r=r/sig;}
sig=sig*rep(1,m);
```

```
theta[,,vv]= matrix(c(bet,pr,sig),nrow=m);
likhod[vv]=lh;vv=vv+1;}
if(all(flag==0)){break}}

fv=which(likhod==max(likhod))
if(maxv[fv]<15)
{result=theta[,,fv];df=maxv[fv];} else
{df=15;run=0;n=length(y);X=cbind(rep(1,n),x);
r=matrix(rep(0,m*n),nrow=n);pk=r;lh=0
for(j in seq(m))
{r[,j]=y-X%*%bet[j,];
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));

repeat
{prest=c(bet,sig,pr);
run=run+1;plh=lh;
for(j in seq(m)){
pk[,j]=pr[j]* pmax(10^(-6),dnorm(r[,j],0,sig))}
pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);
np=apply(pk,2,sum);pr=np/n;
for(j in seq(m)){
w=diag(pk[,j]);
bet[j,]=ginv(t(X)%*%w%*%X)%*%t(X)%*%w%*%y;
r[,j]= y-X%*%bet[j,];}
sig=sqrt(sum(pk*(r^2))/n);lh=0;
for(j in seq(m)){
lh=lh+pr[j]*dnorm(r[,j],0,sig);}
lh=sum(log(lh));dif=lh-plh;
if((dif<1e-6)|(run>500)){break}}
sig=sig*rep(1,m)
result=matrix(c(bet,pr,sig),nrow=m)}

btemp=c(result[1,4], result[1,1:3], result[2,4], result[2,1:3]);
btemp=slab(btemp);
prob=c(btemp[1],btemp[5]);
beta=rbind(btemp[2:4],btemp[6:8]);
rest7=cbind(beta,prob)

b107[k]=rest7[1,1]
b117[k]=rest7[1,2]
b127[k]=rest7[1,3]
b207[k]=rest7[2,1]
b217[k]=rest7[2,2]
b227[k]=rest7[2,3]
```

```
pi17[k]=rest7[1,4]
df7[k]=df}

tb10=0;  tb11=1;   tb12=1;
tb20=0;  tb21=-1;   tb22=-1; tpi1=0.25;

RL=cbind(b107-tb10,b117-tb11,b127-tb12,b207-tb20,b217-tb21,
b227-tb22,pi17-tpi1);
BV[seq((dist-1)*7+1,dist*7),1:2]=cbind(apply(RL,2,function(x){mean(x^2)}),
apply(RL,2,mean))
print(BV[seq((dist-1)*7+1,dist*7),1:2])

# Degree of freedom
dff=cbind(mean(df7),median(df7))
print(dff)}
round(BV,3)
```