ROBUST ESTIMATION OF THE NUMBER OF COMPONENTS FOR MIXTURES OF LINEAR REGRESSION MODELS

by

Meng Li

B.A., Zhengzhou University, China, 2012

A REPORT

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Statistics
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2014

Approved by:

Major Professor
Weixin Yao

Copyright

Abstract

In this report, we investigate a robust estimation of the number of components in the mixture of regression models using trimmed information criterion. Compared to the traditional information criterion, the trimmed criterion is robust and not sensitive to outliers. The superiority of the trimmed methods in comparison with the traditional information criterion methods is illustrated through a simulation study. A real data application is also used to illustrate the effectiveness of the trimmed model selection methods.

Table of Contents

List of Figures

List of Tables

Acknowledgments

My deepest gratitude goes first to my major professor, Dr. Weixin Yao, for his instructive advice and patient guidance.

Then my thanks would go to Dr. Abigail Jager and Dr. Christopher Vahl as being my committee members.

I would also like to express my gratitude to all those who helped me during the writing of this report. Especially my beloved Xiaobei, for his loving considerations and great confidence in me all through these years.

Finally, I owe my sincere gratitude to my roommate Sijia for her continuous support.

Chapter 1

Introduction

Within the family of mixture models, the mixture of linear regression models has been studied extensively. The mixture of linear regression models was first introduced by Goldfeld and Quandt (1973) as a very general form of switching regression. The unknown parameters were estimated based on moment-generating functions, from a likelihood point of view. Jones and McLachlan (1992) applied the mixture of regressions in a data analysis and used EM algorithm to fit these models.

Choosing the number of components for mixture models has long been considered as an important but very difficult research problem. Many methods have been proposed. See, for example, the AIC and BIC methods (Leroux, 1992), distance measures based methods (Chen and Kalbfleisch, 1996; James et al., 2001; Charnigo and Sun, 2004; Woo and Sriram, 2006; Ray and Lindsay, 2008), and hypothesis testing based methods (Chen, Chen, and Kalbfleisch, 2001, 2004). Hawkins et al. (2001) proposed choosing the number of components in the mixture of linear regression models using the likelihood equations. Recently, Chen and Li (2009) and Li and Chen (2010) proposed an EM test approach for testing the order of finite mixtures.

However, most of the above model selection methods are not robust in the presence of outliers. Even a single outlier can totally change the result. In this report, we mainly focus on the information criteria based model selection methods for mixtures of regressions and consider a robust version of these methods based on the trimmed likelihood estimate (TLE, Neykov et al., 2007). A simulation study and a real data application show that the new robust model selection methods work comparably to traditional information criteria based methods when the data are not contaminated but have superior performance when there are outliers.

The rest of the report is organized as follows. In Chapter 2, we give an introduction of four traditionally used information criteria for model selection and introduce their corresponding robust versions based on TLE. In Chapter 3, we use a simulation study and a real data application to demonstrate the effectiveness of the proposed robust model selection methods. A discussion chapter ends the report.

Chapter 2

Robust Model Selection Information Criteria for Mixtures of Regression

2.1 Introduction of mixtures of regressions

Let Z be a latent class variable with $P(Z = j|\mathbf{x}) = \pi_j$, $j = 1, \cdots, m$, where $\mathbf{x}$ is a p-dimensional vector and m is the number of components. Given $Z = j$, the response y depends on the $p-$dimensional predictor $\mathbf{x}$ in a linear way:

$$y = \mathbf{x}^T \boldsymbol{\beta}_j + \epsilon_j,$$

where $\boldsymbol{\beta}_j = (\beta_{1j}, \cdots, \beta_{pj})^T$ and $\epsilon_j \sim N(0, \sigma_j^2)$. Here, we assume that $\mathbf{x}$ includes both the constant 1 and predictors. The conditional distribution of Y given $\mathbf{x}$ without observing Z can be written as:

$$f(y|\mathbf{x}, \boldsymbol{\theta}) = \sum_{j=1}^{m} \pi_j \phi(y; \mathbf{x}^T \boldsymbol{\beta}_j, \sigma_j^2), \tag{2.1}$$

where $\phi(y; \mu, \sigma^2)$ denotes the normal density with mean μ and variance σ^2, and $\boldsymbol{\theta} = (\pi_1, \sigma_1^2, \boldsymbol{\beta}_1, \cdots, \pi_m, \sigma_m^2, \boldsymbol{\beta}_m)^T$.

If the number of components m in the mixture of linear regression models were known, $\boldsymbol{\theta}$ could be estimated by maximizing the log-likelihood

$$\ell(\boldsymbol{\theta}; \mathbf{x}) = \sum_{i=1}^{n} \log \sum_{j=1}^{m} \pi_j \phi(y_i; \mathbf{x}_i^T \boldsymbol{\beta}_j, \sigma_j^2). \tag{2.2}$$

That is,

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log \left\{ \sum_{j=1}^{m} \pi_j \phi(y_i; \mathbf{x}_i^T \boldsymbol{\beta}_j, \sigma_j^2) \right\}. \tag{2.3}$$

Note that the maximizer of (2.3) does not have an explicit solution and is usually estimated by the EM algorithm (Dempster et al., 1977).

2.2 Some information criteria for model selection

If the number of components m is unknown for mixture models, many methods have been proposed to determine the order m. Among them, information criteria have been popularly used to choose the number of components for mixture models due to their simplicity. If the log-likelihood (2.2) is treated as an objective function, one might tend to choose the model that maximizes the log-likelihood of the observed data. However, as pointed out by Celeux and Soromenho (1996), the log-likelihood is an increasing function of m. Therefore, the log-likelihood (2.2) can not be directly used to determine the number of components for mixture models. Many papers, such as Akaike (1974), Bozdogan (1993), and Rissanen (1986, 1987) have sought methods to remedy this problem by adding a penalty term to the log-likelihood.

Akaika's information criterion (AIC) is one of the most popular measures, and was proposed by Bozdogan and Sclove(1984) and Sclove (1987) in the mixture context. It takes the form:

$$AIC = -2\ell(\hat{\boldsymbol{\theta}}) + 2k,$$

where k is the number of parameters to be estimated and $\ell(\hat{\boldsymbol{\theta}})$ is the maximized value of the log-likelihood function for the estimated model.

Bozdogan (1992) provided an analytic extension of AIC, without violating Akaika's principle of minimizing the Kulback-Leibler information quantity. The new selection criterion, called consistent AIC (CAIC), is defined as:

$$CAIC = -2\ell(\hat{\boldsymbol{\theta}}) + k(\log n + 1),$$

where k is the number of free parameters to be estimated, and n is the sample size.

Bayesian information criterion (BIC), proposed by Schwarz (1978), is another commonly used criterion, an approximation to twice the log Bayes factor (Fraley and Raftery, 1998). The approximation relies on regularity conditions that do not hold in the mixture models setting, but BIC has been shown to provide a consistent estimate of the number of components in mixture models (Keribin, 2000). It is defined by

$$BIC = -\, 2\ell(\hat{\boldsymbol{\theta}}) + k \log n,$$

where n is the number of observations, or equivalently, the sample size.

AIC and BIC are typically derived from approximations based on asymptotic arguments (Kass and Raftery, 1995). They penalize the log-likelihood by an additive factor and are relatively simple to implement. Although there are theoretical limitations on the applicability of these two methods, they have been proven to work quite well for model selection in mixture models.

BIC works well with the case that each mixture component corresponds to a separate cluster. However, if the number of clusters in the data set is different from the number of components, Biernacki et al. (2000) proposed the integrated complete likelihood (ICL) criterion as a modification. Let z_{ij} be the component label indicator,

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

Then, the complete-data is $\mathbf{x}_c = (\mathbf{x}, \mathbf{z})$ and its complete log-likelihood is given by:

$$\ell_c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{z}) = \sum_{i=1}^{n} \log \prod_{j=1}^{m} \left\{\pi_j \phi(y_i; \mathbf{x}_i^T \boldsymbol{\beta}_j, \sigma_j^2)\right\}^{z_{ij}} = \sum_{i=1}^{n} \sum_{j=1}^{m} z_{ij} \log \left\{\pi_j \phi(y_i; \mathbf{x}_i^T \boldsymbol{\beta}_j, \sigma_j^2)\right\}.$$

ICL criterion penalizes the complexity of the mixture model, thus ensuring the partitioning of data with the greatest evidence.

$$ICL = -2\ell_c(\hat{\boldsymbol{\theta}} | \mathbf{x}, \hat{\mathbf{z}}) + k \log n,$$

where the missing data $\mathbf{z}$ have been replaced by their most probable values $\hat{\mathbf{z}}$, given the parameter estimate $\hat{\boldsymbol{\theta}}$. In the BIC approach, only the observed likelihood is maximized, however, in the ICL approach, the complete log-likelihood is used. ICL appears to be more robust than BIC to the violation of some of mixture model assumptions and can select a number of clusters leading to a sensible partition of the data.

The “mixtools” package (Benaglia et al., 2009) provides a set of functions to analyze various finite mixture models. These functions include both traditional methods like EM algorithms for univariate and multivariate normal mixtures, and newer methods reflecting recent research in finite mixture models. The basic goal of the tools in “mixtools” package is to discern subgroups of individuals by examining a sample of measurements. The command “regmixmodel.sel” in R package “mixtools” uses the above four information criteria to select the number of components for mixture models. In our examples of Chapter 3, we used “mixtools” to implement the above four information criteria.

2.3 Trimmed information criteria

It is well known that the maximum likelihood estimate (MLE) via the expectation-maximization (EM) algorithm works well for the finite mixture of distributions. However, it is sensitive to outliers. Even a single outlier can cause at least one of the component parameters to become arbitrarily large. Therefore, the traditional information criteria are sensitive to outliers in the data set. In this section, we consider a trimmed version of those information criteria to robustify the model selection procedures.

Assume that $(1-\alpha)\times 100\%$ of the observations in the data set are regular observations, and the remaining $\alpha \times 100\%$ are unpredictable outliers. The trimmed likelihood estimate (TLE) of mixture models, proposed by Neykov et al. (2007), only uses $(1-\alpha)\times 100\%$ of the data to fit the model, and removes the remaining $\alpha\times 100\%$ observations that are highly unlikely to occur if the fitted model were true. That is,

$$\max_{I_\alpha} \max_{\boldsymbol{\theta}} \sum_{i\in I_\alpha} f(y_i|\boldsymbol{\theta}), \tag{2.4}$$

where $f(y \mid \boldsymbol{\theta})$ is the density defined in (2.1), and I_α is the subset of $\{1,\ldots,n\}$ and only contains $\lfloor n(1-\alpha)\rfloor$ distinct elements of $\{1,\ldots,n\}$.

By combining the ideas of the trimmed likelihood estimate and the information criteria introduced in Section 2.2, we consider the trimmed versions of AIC, BIC, ICL, and CAIC to robustly estimate the number of components for the mixture of regression models as follows:

$$TAIC = -2\sum_{i\in\hat{I}_\alpha} f(y_i|\hat{\boldsymbol{\theta}}) + 2k, \tag{2.5}$$

$$TBIC = -2\sum_{i\in\hat{I}_\alpha} f(y_i|\hat{\boldsymbol{\theta}}) + k\log n, \tag{2.6}$$

$$TICL = -2\sum_{i\in\hat{I}_\alpha} f_c(y_i|\hat{\boldsymbol{\theta}},\hat{\mathbf{z}}) + k\log n, \tag{2.7}$$

$$TCAIC = -2\sum_{i\in\hat{I}_\alpha} f(y_i|\hat{\boldsymbol{\theta}}) + k(\log n + 1), \tag{2.8}$$

where $\hat{\boldsymbol{\theta}}$ is the trimmed likelihood estimator, $\hat{I}_\alpha$ is the corresponding index set, and $f_c(y_i|\boldsymbol{\theta},\mathbf{z}) = \sum_{j=1}^m z_{ij}\log\left\{\pi_j\phi(y_i;\mathbf{x}_i^T\boldsymbol{\beta}_j,\sigma_j^2)\right\}$.

The combinatorial nature of the TLE, that is, all possible $\binom{n}{\lfloor n(1-\alpha)\rfloor}$ partitions of the data have to be fitted by the MLE, makes the TLE computationally expensive, and infeasible for large data sets. The FAST-TLE algorithm (M$\ddot{u}$ller and Neykov, 2003; Neykov et al., 2007) was proposed to obtain an approximative TLE solution, which involves repeated iterations of a two-step procedure: a trial step followed by a refinement step. Next, we give the computation procedure to calculate (2.5)-(2.8) using FAST-TLE (Neykov et al., 2007).

1. Find an initial value for $\boldsymbol{\theta}$, denoted by $\boldsymbol{\theta}_0$. The initial value $\boldsymbol{\theta}_0$ might be found by fitting the mixture of linear regressions to a random subsample of size d from the data, where d is a value larger than k. In our report, we tried 40 initial values in total.

2. (a) For a given estimator $\hat{\boldsymbol{\theta}}$, sort $f(y|\hat{\boldsymbol{\theta}})$ as $f(y_{\nu(1)}|\hat{\boldsymbol{\theta}}) \geq f(y_{\nu(2)}|\hat{\boldsymbol{\theta}}) \geq \cdots \geq f(y_{\nu(n)}|\hat{\boldsymbol{\theta}})$, then $\{\nu(1),\cdots,\nu(\lfloor n(1-\alpha)\rfloor)\}$ forms the index set $\hat{I}_\alpha$.

   (b) Given an index set $\hat{I}_\alpha$, update the estimator of $\boldsymbol{\theta}$, which maximizes $\sum_{i\in\hat{I}_\alpha} f(y_i|\boldsymbol{\theta})$.

   (c) Iterate (a)-(b) until convergence.

3. *Repeat 1 and 2.*

4. Apply $\hat{\boldsymbol{\theta}}$ and $\hat{I}_\alpha$ to (2.5)-(2.8) to estimate the number of components.

We want to point out that we are not the first ones to use the trimmed information criteria. Neykov et al. (2007) have briefly discussed the trimmed BIC in their simulation study. Based on their limited empirical experience, the trimmed BIC works well and could robustly estimate the number of mixture components. In this report, we extend such trimmed idea to some other commonly used information criteria and give more simulation study and a real data application in next chapter to demonstrate the effectiveness of the trimmed information criteria.

Chapter 3

Simulation Study and Real Data Application

3.1 Simulation study

In this chapter, we investigate the effectiveness of the trimmed information criteria and compare them with the traditional information criteria for model selection for mixture models when outliers are present. To be more specific, the following four new methods are considered: trimmed AIC (TAIC), trimmed BIC (TBIC), trimmed CAIC (TCAIC), and trimmed ICL (TICL). The performance of the new methods are compared to AIC, BIC, CAIC, and ICL, whose likelihoods are calculated based on the maximum likelihood estimate (MLE). The trimming proportion α is set to be 5% for all information criteria. Similar to TLE, the proportion α is an important tuning parameter. Usually a conservative α is desired. In our simulation study, the proportion of outliers is never greater than 0.05. In Section 3.2, we use a real data set to illustrate how to data adaptively choose the α using the graphical tool proposed in Neykov et al. (2007).

To compare the performance of different model selection methods, we report the percentage of times when the number of components is correctly estimated. In addition, we also report the lower quartile (LQ), the median (MD), and the upper quartile (UQ) of the estimated number of components for each method.

We consider the following two mixture of linear regression models:

Example 1:

$$Y = \begin{cases} 0 + X_1 + X_2 + \epsilon_1, & \text{if } Z = 1, \\ 0 - X_1 - X_2 + \epsilon_2, & \text{if } Z = 2. \end{cases}$$

Example 2:

$$Y = \begin{cases} 3 + 3X_1 + 4X_2 + \epsilon_1, & \text{if } Z = 1, \\ 1 + X_1 + X_2 + \epsilon_2, & \text{if } Z = 2, \\ -1 - X_1 - X_2 + \epsilon_3, & \text{if } Z = 3, \\ -3 - 3X_1 - 4X_2 + \epsilon_4, & \text{if } Z = 4. \end{cases}$$

The mixing proportions are 0.4 and 0.6 in *Example 1*, and are all equal to 0.25 in *Example 2*. In both examples, $X_1 \sim N(0,1)$ and $X_2 \sim N(0,1)$ are uncorrelated, and the errors have the same distribution as $\epsilon \sim N(0,1)$. The sample sizes $n = 100$ and $n = 200$ are conducted over 200 repetitions, and the proportions of outliers are $\alpha = 0.05$, $\alpha = 0.03$, and $\alpha = 0.01$. Note that the trimming proportion is 0.05 for all trimmed information criteria. Using the above three proportions of outliers, we can check how the trimmed information criteria work for both the cases where the trimming proportions are correct and the cases where the trimming proportions are conservative.

Table 3.1 reports the percentage of times when the number of components in the mixture of linear regression models is correctly estimated. From Table 3.1, we can see that the percentages of correct estimates of number of components by TBIC, TCAIC, and TICL are much higher than the four traditional methods and TAIC. Therefore, both AIC and its trimmed version largely fail to estimate the correct number of components. But the trimmed versions of BIC, CAIC, and ICL can greatly improve the performance of their original versions when there are outliers. In addition, to our surprise, the traditional methods perform better when the sample size is small in our examples, but TBIC, TCAIC, and TICL show better results when the sample size is large.

To understand better how the methods perform, we also report the lower quartile, the median, and the upper quartile of the estimated numbers of components, in Table 3.2 and Table 3.3 for $n = 100$ and $n = 200$, respectively. Clearly, TBIC, TCAIC, and TICL give much better performance than the other methods. In addition, AIC, BIC, CAIC, ICL, and TAIC tend to overestimate the number of components.

		$n = 100$			$n = 200$		
		$\alpha = 0.05$	$\alpha = 0.03$	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.03$	$\alpha = 0.01$
Example 1							
	AIC	0%	0%	1.4%	0%	0%	2.5%
	BIC	3.6%	3.4%	19.4%	0%	0%	5%
	CAIC	6.6%	6%	24.6%	1%	0%	8.5%
	ICL	2.4%	3%	18.4%	0%	0%	5%
	TAIC	0%	0%	0%	4.5%	0%	0%
	TBIC	94%	92.4%	85.4%	99.5%	99.5%	99%
	TCAIC	96.4%	93.2%	90.2%	100%	100%	99.5%
	TICL	93%	90.6%	82.6%	99.5%	99%	97.5%
Example 2							
	AIC	5.5%	0%	2.5%	3%	0%	0%
	BIC	23%	7.5%	9%	20%	2%	0%
	CAIC	27.5%	10.5%	14%	25%	4%	0.5%
	ICL	23.5%	7.5%	9%	19.5%	2%	0%
	TAIC	2%	2%	1%	28%	21%	5%
	TBIC	86%	83%	75.5%	98.5%	99%	99%
	TCAIC	91%	85%	73.5%	100%	99%	98.5%
	TICL	86%	82%	74.5%	98.5%	99%	99%

Table 3.1: *Percentages of times when the number of components in the mixture of regression models is correctly estimated.*

		$\alpha = 0.05$			$\alpha = 0.03$			$\alpha = 0.01$		
		LQ	MD	UQ	LQ	MD	UQ	LQ	MD	UQ
$m = 2$										
	AIC	3	4	7	5	8	9	5	8	9
	BIC	3	3	3	3	3	3	3	3	3
	CAIC	3	3	3	3	3	3	2	3	3
	ICL	3	3	3	3	3	3	3	3	3
	TAIC	10	10	10	10	10	10	10	10	10
	TBIC	2	2	2	2	2	2	2	2	2
	TCAIC	2	2	2	2	2	2	2	2	2
	TICL	2	2	2	2	2	2	2	2	2
$m = 4$										
	AIC	5	5	7	5	6	8	5	6	7
	BIC	4	5	5	5	5	6	5	5	5
	CAIC	4	5	5	5	5	5	5	5	5
	ICL	4	5	5	5	5	6	5	5	5
	TAIC	9	10	10	9	10	10	9	10	10
	TBIC	4	4	4	4	4	4	4	4	4
	TCAIC	4	4	4	4	4	4	4	4	4
	TICL	4	4	4	4	4	4	4	4	4

Table 3.2: *The lower quartile (LQ), the median (MD), and the upper quartile (UQ) of the estimated numbers of components when $n = 100$.*

		$\alpha = 0.05$			$\alpha = 0.03$			$\alpha = 0.01$		
		LQ	MD	UQ	LQ	MD	UQ	LQ	MD	UQ
Example 1 ($m = 2$)										
	AIC	3	3	5	3	3	5	3	5	8
	BIC	3	3	3	3	3	3	3	3	4
	CAIC	3	3	3	3	3	3	3	3	4
	ICL	3	3	3	3	3	3	3	3	4
	TAIC	9	10	10	10	10	10	10	10	10
	TBIC	2	2	2	2	2	2	2	2	2
	TCAIC	2	2	2	2	2	2	2	2	2
	TICL	2	2	2	2	2	2	2	2	2
Example 2 ($m = 4$)										
	AIC	5	5	7	5	5	6	5	5.5	6
	BIC	4	5	5	5	5	5	5	5	6
	CAIC	4	5	5	5	5	5	5	5	6
	ICL	5	5	5	5	5	5	5	5	6
	TAIC	4	7	9	5	9	10	8	10	10
	TBIC	4	4	4	4	4	4	4	4	4
	TCAIC	4	4	4	4	4	4	4	4	4
	TICL	4	4	4	4	4	4	4	4	4

Table 3.3: *The lower quartile (LQ), the median (MD), and the upper quartile (UQ) of the estimated numbers of components when $n = 200$.*

3.2 Real data analysis

Next, we use the crabs data from R-package "MASS" as an example to compare different information criteria and their trimmed versions. Five morphological measurements were taken from 200 crabs of the species Leptograpsus variegatus, collected at Fremantle, Australia. Following García-Escudero et al. (2010), here we only focus on analyzing two variables, RW (rear width in mm) and CL (carapace length mm), with the objective of distinguishing between the two crabs sexes, without the other variables. The variable CL is considered as the response variable while RW is considered as the explanatory variable. The scatter plot of the data is shown in Figure 3.1, where squares and circles denote the two groups of crabs based on their sexes.

Let us suppose that the sexes of the crabs were unknown, and then estimate the number of components. The results are reported in the first row of Table 3.4 ($\alpha = 0$, without outliers). It can be seen that all methods, except for TAIC, provide correct estimates of the number of components, when there are no outliers in the data.

To check the robustness of different model selection methods, similar to McLachlan and Peel (2000), we artificially add some outliers to the original data set, for example, we randomly add 1%, 3% and 5% outliers into the original data set. Figure 3.1 shows an example when 5% outliers, denoted by dots, were added into the data. The model selection results are also reported in Table 3.4, from which we can see that our proposed TBIC, TCAIC, and TICL perform much better than the rest of the methods. In addition, it can be seen that AIC and TAIC tend to overestimate the number of components, which is consistent with the simulation results.

In all the above examples, we have fixed α in advance. To explore how to data adaptively select the trimming proportions in practical problems, we apply the graphical tool proposed in Neykov et al. (2007), that is, we fit the crabs data with a 2-component mixture of linear regression models using a grid points of α values, ranging from 0% to 50% in steps of 1%. Figure 3.2 shows the trimmed likelihood versus α's, when the actual percentages of

Table 3.4: *The estimated number of components for crabs data when using traditional and trimmed information criteria.*

	AIC	BIC	CAIC	ICL	TAIC	TBIC	TCAIC	TICL
α_0=0	2	2	2	2	4	2	2	2
α_0=0.01	5	3	3	3	2	2	2	2
α_0=0.03	6	3	3	3	9	2	2	2
α_0=0.05	5	3	3	3	7	2	2	2

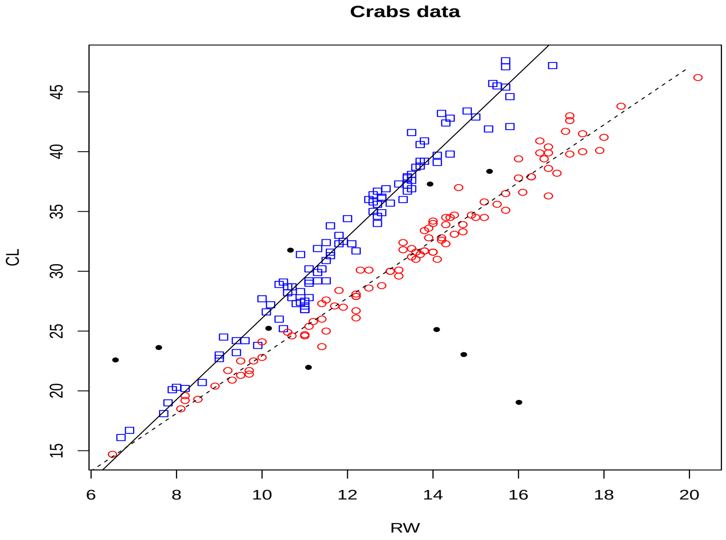

Figure 3.1: *Crabs data: squares and circles denote the two groups of crabs based on their sexes, and dots denote the randomly generated outliers.*

outliers are 1%, 3%, 5% and 10%, respectively, denoted by the vertical lines. The percentage of outliers can be estimated by the largest α at which the slope of curves changes. From Figure 3.2, we can see that the slope-changing locations are very close to the true percentages of outliers (corresponding to the vertical lines).

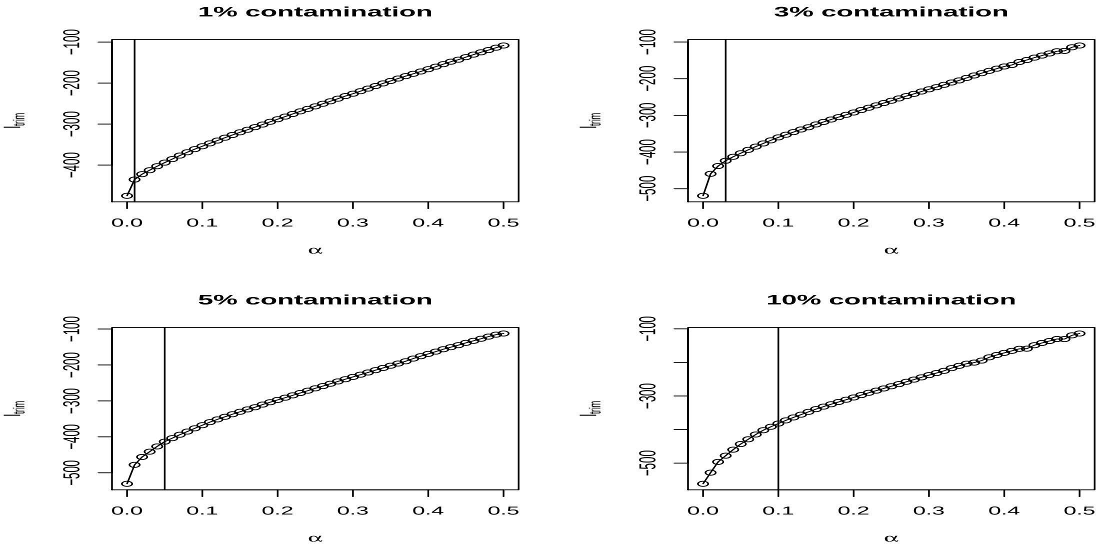

Figure 3.2: *Plot of trimmed likelihood versus the trimming proportion. The vertical line corresponds to the true percentages of outliers.*

Chapter 4

Discussion

In this report, we investigated some trimmed versions of information criteria to robustly estimate the number of components for mixtures of linear regression models when outliers are present in the data set. We demonstrated the superiority of the trimmed methods in comparison with the traditional methods when the data are contaminated using a simulation study and a real data example.

However, in this report, we mainly focus on the information criteria based on model selection methods. It requires more research to see whether we can use similar trimmed idea to robustify some other model selection methods, such as EM test (Chen and Li, 2009; Li and Chen, 2010). Chen and Khalili (2009) used a penalized likelihood to do order selection for mixture models. It will be interesting to see whether we can achieve robust order selection if we apply similar penalty functions to some robust mixtures of regression models (Neykov et al. 2007; Bai et al. 2012; Bashir and Carter, 2012; Yao et al. 2014; Song et al. 2014).

We have applied the graphical tool proposed in Neykov et al. (2007) to choose the trimming proportion α in the crabs data application. However, as pointed out by Neykov et al. (2007), such estimated trimming proportions tend to underestimate the true values in some cases. Therefore, it requires further study about how to data adaptively choose an optimal or a conservative trimming proportion α.

Bibliography

[1] Akaike, H. (1974). A new look at the statistical identification model. *IEEE Transactions on Automatic Control*, 19, 716-723.

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

[3] Bashir, S. and Carter, E. (2012). Robust mixture of linear regression models. *Communications in Statistics-Theory and Methods*, 41, 3371-3388.

[4] Benaglia, B., Chauveau, D., Hunter, D.R. and Young, D. (2009). Mixtools: An R Package for Analyzing Finite Mixture Models. *Journal of Statistical Software*, 32(6), 1-29.

[5] Biernacki, C., Celeux, G. and Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 719-725.

[6] Bozdogan, H. and Sclove, S. L. (1984). Multi-sample cluster analysis using Akaike's information criterion. *Annals of the Institute of Statistical Mathematics*, 36, 163-180.

[7] Bozdogan, H. (1992). Choosing the number of component clusters in the mixture-model using a new informational complexity criterion of the inverse-fisher information matrix. *In: Opitz, O., Lausen, B. and Klar, R. (Eds.), Information and Classification: Concepts, Methods and Applications, New York. Springer-Verlag*, 44-54.

[8] Bozdogan, H. (1993). Choosing the number of component clusters in the mixture-model using a new informational complexity criterion of the inverse-fisher information matrix.

In: Opitz, O., Lausen, B. and Klar, R. (Eds.), Studies in Classification, Data Analysis and Knowledge Organization, Springer-Verlag, Heidelberg, 40-54.

[9] Celeux G. and Soromenho G. (1996). An entropy criterion for assessing the number of clusters in a mixture model. *Journal of Classification*, 13, 195-212.

[10] Charnigo, R. and Sun, J. (2004). Testing homogeneity in a mixture distribution via the L2- distance between competing models. *Journal of the American Statistical Association*, 99, 488-498.

[11] Chen, H., Chen, J. and Kalbfleisch, J. D. (2001). A modified likelihood ratio test for homogeneity in finite mixture models. *Journal of the Royal Statistical Society, Series B*, 63, 19-29.

[12] Chen, H., Chen, J. and Kalbfleisch, J. D. (2004). Testing for a finite mixture model with two components. *Journal of the Royal Statistical Society, Series B*, 66, 95-115.

[14] Chen, J. and Kalbfleisch, J. D. (1996). Penalized minimum-distance estimates in finite mixture models. *The Canadian Journal of Statistics*, 24, 167-175.

[14] Chen, J. and Khalili, A. (2009). Order selection in finite mixture models with a nonsmooth penalty. *Journal of the American Statistical Association*, 104,187-196.

[15] Chen, J. and Li, P. (2009). Hypothesis test for normal mixture models: The EM approach. *The Annals of Statistics*, 37, 2523-2542.

[16] 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.

[17] Fraley, C. and Raftery, A. E. (1998). How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41, 578-588.

[19] García-Escudero, L. A., Gordaliza, A., Mayo-Iscar, A. and San Martín, R. (2010). Robust clusterwise linear regression through trimming. *Computational Statistics and Data Analysis*, 54, 3057-3069.

[19] Goldfeld, S. M. and Quandt, R. E. (1973). A Markov model for switching regressions. *Journal of Econometrics*, 1, 3-6.

[20] Hawkins D. S., Allen D. M. and Stomberg A. J. (2001). Determining the number of components in mixtures of linear models. *Computational Statistical and Data Analysis*, 38, 15-48.

[21] James, L. F., Priebe, C. E. and Marchette, D. J. (2001). Consistent estimation of mixture complexity. *The Annals of Statistics*, 29, 1281-1296.

[22] Jones, P. N. and Mclachlan G. J. (1992). Fitting finite mixture models in a regression context. *Australian Journal of Statistics*, 34, 233-240.

[23] Keribin, C. (2000). Consistent estimation of the order of mixture models. *The Indian Journal of Statistics*, 62, 49-66.

[24] Kass, R. E. and Raftery A. E. (1995). Bayes factors. *Journal of the American Statistical Association*, 90, 773-795.

[25] Leroux, B. G. (1992). Consistent estimation of a mixing distribution. *The Annals of Statistics*, 20, 1350-1360.

[26] Li, P. and Chen, J. (2010). Testing the order of a finite mixture. *The Journal of American Statistical Association*, 105, 1084-1092.

[27] McLachlan, G. J. and Peel, D. (2000). Finite mixture models. John Wiley and Sons, New York.

[28] Müller, C. and Neykov, N. (2003). Breakdown points of trimmed likelihood estimators and related estimators in generalized linear models. *Journal of Statistical Planning and Inference*, 116, 503-519.

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

[30] Ray, S. and Lindsay, B. G. (2008). Model selection in high-dimensions: A quadratic-risk based approach. *Journal of the Royal Statistical Society, Series B*, 70, 95-118.

[32] Rissanen, J. (1986). Stochastic complexity and modeling. *Annals of Statisics*, 14, 1080-1100.

[32] Rissanen, J. (1987). Stochastic complexity. *Journal of the Royal Statistical Society B*, 49, 223-239.

[33] Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6, 461-464.

[34] Sclove, S. L. (1987). Application of some model-selection criteria to some problems in multivariate analysis. *Psychometrika*, 52, 333-343.

[37] Song, W., Yao, W. and Xing, Y. (2014). Robust mixture regression model fitting by Laplace distribution. *Computational Statistics and Data Analysis*, 71, 128-137.

[37] Woo, M. and Sriram, T. N. (2006). Robust estimation of mixture complexity. *Journal of the American Statistical Association*, 101, 1475-1485.

[37] Yao, W., Wei, Y. and Yu, C. (2014). Robust mixture regression using t-distribution. *Computational Statistics and Data Analysis*, 71, 116-127.

Appendix A

R-Programs

```
rm(list=ls())

library(MASS)

library(gregmisc)

library(robustbase)

library(mixtools)


regmixEM<-function(y, x, lambda = NULL, beta = NULL, sigma = NULL, k = 2,

    addintercept = TRUE, arbmean = TRUE, arbvar = TRUE, epsilon = 1e-08,

    maxit = 10000, verb = FALSE)

{

    if (arbmean == FALSE && arbvar == FALSE) {

        stop(paste("Must change constraints on beta and/or sigma!",

            "\n"))

    }

    s = sigma

    if (addintercept) {

        x = cbind(1, x)

    }

    n <- length(y)
```

```
p <- ncol(x)
tmp <- regmix.init(y = y, x = x, lambda = lambda, beta = beta,
    s = s, k = k, addintercept = addintercept, arbmean = arbmean,
    arbvar = arbvar)
lambda <- tmp$lambda
beta <- tmp$beta
s <- tmp$s
k <- tmp$k
diff <- 1
iter <- 0
xbeta <- x %*% beta
res <- (y - xbeta)^2
if (arbmean == FALSE) {
    res <- sapply(1:k, function(i) res)
}
comp <- t((lambda/sqrt(2 * pi * s^2)) * t(exp(-t(t(res)/(2 *
    s^2)))))
obsloglik <- sum(log(apply(comp, 1, sum)))
ll <- obsloglik
z = matrix(nrow = n, ncol = k)
restarts <- 0
while (diff > epsilon && iter < maxit) {
    for (i in 1:n) {
        for (j in 1:k) {
            z.denom = c()
            for (h in 1:k) {
              z.denom = c(z.denom, (lambda[h]/lambda[j]) *
                (s[j * arbvar + (1 - arbvar)]/s[h * arbvar +
                  (1 - arbvar)]) * exp(-0.5 * ((1/s[h * arbvar +
                (1 - arbvar)]^2) * res[i, h] - (1/s[j * arbvar +
                (1 - arbvar)]^2) * res[i, j])))
            }
            z[i, j] = 1/sum(z.denom)
        }
    }
    z = z/apply(z, 1, sum)
    lambda.new <- apply(z, 2, mean)
    if (sum(lambda.new < 1e-08) > 0 || is.na(sum(lambda.new))) {
        sing <- 1
    }
    else {
        if (arbmean == FALSE) {
            if (addintercept) {
              beta.new <- lm(y ~ x[, -1], weights = apply(t(t(z)/(s^2)),
                1, sum))$coef
            }
            else beta.new <- lm(y ~ x - 1, weights = apply(t(t(z)/(s^2)),
              1, sum))$coef
        }
        else {
            if (addintercept) {
              lm.out <- lapply(1:k, function(i) lm(y ~ x[,
                -1], weights = z[, i]))
            }
            else lm.out <- lapply(1:k, function(i) lm(y ~
              x - 1, weights = z[, i]))
            beta.new <- sapply(lm.out, coef)
        }
        xbeta.new <- x %*% beta.new
        res <- (y - xbeta.new)^2
        if (arbmean == FALSE) {
            res <- sapply(1:k, function(i) res)
        }
        if (arbvar) {
            s.new <- sqrt(sapply(1:k, function(i) sum(z[,
              i] * (res[, i]))/sum(z[, i])))
        }
        else s.new <- sqrt(sum(z * res)/n)
        lambda <- lambda.new
        beta <- beta.new
        xbeta <- x %*% beta
        s <- s.new
        sing <- sum(s < 1e-08)
        comp <- lapply(1:k, function(i) lambda[i] * dnorm(y,
            xbeta[, i * arbmean + (1 - arbmean)], s[i * arbvar +
              (1 - arbvar)]))
        comp <- sapply(comp, cbind)
        compsum <- apply(comp, 1, sum)
        newobsloglik <- sum(log(compsum))
    }
    if (sing > 0 || is.na(newobsloglik) || newobsloglik <
        obsloglik || abs(newobsloglik) == Inf) {
        cat("Need new starting values due to singularity...",
            "\n")
        restarts <- restarts + 1
        if (restarts > 50)
            stop("Too many tries!")
        tmp <- regmix.init(y = y, x = x, k = k, addintercept = addintercept,
            arbmean = arbmean, arbvar = arbvar)
        lambda <- tmp$lambda
        beta <- tmp$beta
        s <- tmp$s
        k <- tmp$k
        diff <- 1
        iter <- 0
        xbeta <- x %*% beta
        res <- (y - xbeta)^2
        if (arbmean == FALSE) {
            res <- sapply(1:k, function(i) res)
        }
        comp <- t((lambda/sqrt(2 * pi * s^2)) * t(exp(-t(t(res)/(2 *
            s^2)))))
        obsloglik <- sum(log(apply(comp, 1, sum)))
        ll <- obsloglik
    }
    else {
        diff <- newobsloglik - obsloglik
        obsloglik <- newobsloglik
```

```
        ll <- c(ll, obsloglik)
        iter <- iter + 1
        if (verb) {
            cat("iteration=", iter, "diff=", diff, "log-likelihood",
              obsloglik, "\n")
        }
    }
}
scale.order = order(s)
sigma.min = min(s)
if (iter == maxit) {
    cat("WARNING! NOT CONVERGENT!", "\n")
}
cat("number of iterations=", iter, "\n")
if (arbmean == FALSE) {
    z = z[, scale.order]
    names(beta) <- c(paste("beta", ".", 0:(p - 1), sep = ""))
    colnames(z) <- c(paste("comp", ".", 1:k, sep = ""))
    a = list(x = x, y = y, lambda = lambda[scale.order],
        beta = beta, sigma = sigma.min, scale = s[scale.order]/sigma.min,
        loglik = obsloglik, posterior = z[, scale.order],
        all.loglik = ll, restarts = restarts, ft = "regmixEM")
    class(a) = "mixEM"
    a
}
else {
    rownames(beta) <- c(paste("beta", ".", 0:(p - 1), sep = ""))
```

```
        colnames(beta) <- c(paste("comp", ".", 1:k, sep = ""))
        colnames(z) <- c(paste("comp", ".", 1:k, sep = ""))
        a = list(x = x, y = y, lambda = lambda, beta = beta,
            sigma = s, loglik = obsloglik, posterior = z, all.loglik = ll,
            restarts = restarts, ft = "regmixEM")
        class(a) = "mixEM"
        a
    }
}

mixreg<-function(x,y,k, maxit=200){
  if(k>1){
    est=regmixEM(y,x,k=k,arbvar = 0,epsilon = 1e-08,maxit=maxit)
    }
  else{
    X=cbind(1,x);
    bet=ginv(t(X)%*%X)%*%t(X)%*%y;
    err=y-X%*%bet;sigma=sqrt(mean(err^2));
    lambda=1;
    est=list(beta=bet,sigma=sigma,lambda=lambda)
  }
  est
}


mixlinone<-function(x,y,bet,sig,pr,m=2, maxit=200)
{
 run=0;n=length(y);
 X=cbind(rep(1,n),x);
 if(length(sig)>1){  # unequal variances
  r=matrix(rep(0,m*n),nrow=n);pk=r;lh=0;
  for(j in 1:m){ # calculate likelihood: lh
   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^(-300),dnorm(r[,j],0,sig[j]))
   }
   pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n); # z_ij clasification probability

#M-steps
   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; #beta_j
    r[,j]= y-X%*%bet[j,];
    sig[j]=sqrt(t(pk[,j])%*%(r[,j]^2)/np[j]); # sigma_j
    lh=lh+pr[j]*dnorm(r[,j],0,sig[j]);
   }
   lh=sum(log(lh));dif=lh-plh;
```

```
      if(dif<10^(-5)|run>maxit){
        break
      }
    }
  }
  else{  # equal variances
    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^(-300),dnorm(r[,j],0,sig))
     }
     pk=pk/matrix(rep(apply(pk,1,sum),m),nrow=n);

#M-steps
     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<10^(-5)|run>maxit){
       break
     }
   }
    sig=sig*rep(1,m)
  }
  est=list(theta=matrix(c(bet,sig,pr),nrow=m),likelihood=lh,run=run,diflh=dif)
  est
}

##############################
#trimmed likelihood estimator
###################################
trimmixone<-function(x,y,k=2,alpha=0.9,bet,sig,pr){
    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;acc=10^(-4);
    obj=sum(log(lh[ind[1:n1]]));
    repeat
        {pobj=obj;run=run+1;
         x1=x[ind[1:n1],];y1=y[ind[1:n1]];
         fit=mixlinone(x1,y1,bet,sig,pr,k, maxit=200);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>300){break}
     }
     if(length(sig)<2)  sig=rep(sig,k);
     #theta=c(bet,sig,pr);
     est=list(beta=bet,lambda=pr,sigma=sig,loglik=obj,dif=dif,run=run)
     est
}

trimmix <-function(x,y,k=2,alpha=0.9,numini=20){
     n=length(y); x=matrix(x,nrow=n)
     a=dim(x);p=a[2]+1; n1=2*p;
     #temp=roblga(x,y,k,alpha,numstart=100);
     #bet=matrix(temp[1:(p*k)],nrow=k);
     #sig=mean(temp[(p*k+1):(p*k+k)]);
     #pr=temp[(p*k+k+1):(p*k+2*k)];
     #est=trimmixone(x,y,k,alpha,bet,sig,pr);
     #lh=est$likelihood;obj=rep(0,numini+1);obj[1]=lh;
     X=cbind(rep(1,n),x);
     tmp<- regmix.init(y,X,k = k,arbvar = 0)
     pr<- tmp$lambda
     bet<- tmp$beta
     sig<- tmp$s
     #pest=est;plh=lh;
     est=trimmixone(x,y,k,alpha,bet,sig,pr);
     lh=est$loglik;obj=rep(0,numini+1);obj[1]=lh;
     #if(lh<plh){est=pest;lh=plh;}

     for(i in seq(numini))
         {bet= matrix(rep(0,k*p),nrow=k);sig=0;
         for(j in seq(k))
            {ind=sample(1:n,n1); X=cbind(rep(1,n1),x[ind,]);
             bet[j,]=ginv(t(X)%*%X)%*%t(X) %*%y[ind];
             sig=sig+sum((y[ind] -X%*%bet[j,])^2);
         }
         pr=rep(1/k,k);sig=sig/n1/k;
         pest=est;plh=lh;
         est=trimmixone(x,y,k,alpha,bet,sig,pr);
         lh=est$loglik;obj[i+1]=lh;
         if(lh<plh){est=pest;lh=plh;}
     }
#est=list(theta=est$theta,loglik=est$likelihood,run=est$run,obj=obj)
est$obj=obj
```

```
est
}


robregmixmodel.sel<-function (x, y, k = 2, alpha=0.9,numini=10)
{
    aic <- NULL
    bic <- NULL
    caic<- NULL
    icl <- NULL
    AIC <- function(emout, p) {
        emout$loglik - (p - 1)
    }
    BIC <- function(emout, p, n) {
        emout$loglik - log(n) * (p - 1)/2
    }
    CAIC <- function(emout, p, n) {
        emout$loglik - (log(n) + 1) * (p - 1)/2
    }
    ICL <- function(emout, p, n) {
        BIC(emout, p, n) - sum(emout$lambda * log(emout$lambda))
    }

    n <- length(y)
    for (i in 1:k) {
         if (i == 1) {
             a <- ltsReg(x,y,nsamp=100,alpha=alpha)
             beta <- matrix(a$coef, ncol = 1)
             sd=sqrt(var(a$residuals[a$best]))
    loglik <- log(prod(dnorm(y[a$best], mean = a$fitted.values[a$best], sd = sd)))
    emout <- list(beta = beta, sigma = sd,
    lambda = 1, loglik = loglik)
    }
    else emout <- trimmix(x,y,k=i,alpha=alpha,numini=numini)
    P = length(emout$beta)+1+i
    aic[i] <- AIC(emout, p = P)
    bic[i] <- BIC(emout, p = P, n = n)
    caic[i] <- CAIC(emout, p = P, n = n)
    icl[i] <- ICL(emout, p = P, n = n)
    }
    out = rbind(aic, bic, caic, icl)
    Winner = apply(out, 1, function(x) (1:length(x))[x == max(x)])
    colnames(out) = 1:k
    rownames(out) = c("AIC", "BIC", "CAIC", "ICL")
    cbind(out, Winner)
}

#######
#Simulation study when there are m=4 components
 ###########################
set.seed(10^8)
bt=Sys.time()
repnum=200; m=4;
n=100; #n=200;
alpha=0.99       #trim proportion, alpha=0.97, alpha=0.99
```

```
numoutlier=round(n*(1-alpha))
bet=matrix(c(3,1,-1,-3,3,1,-1,-3,4,1,-1,-4),nrow=4);p=3;
pr=c(0.25,0.25,0.25,0.25); upk=10;
numcompaic=rep(0,repnum);numcompbic=numcompaic;
numcompcaic=numcompaic;numcompicl=numcompaic;
numcompaic1=rep(0,repnum);numcompbic1=numcompaic;
numcompcaic1=numcompaic;numcompicl1=numcompaic;
time1=c();time2=c();
save.image("E:\\Dropbox\\student\\Mengli\\simulation\\ex61.RData")
for(i in 1:repnum){
print(i)
  sig=1;e=rnorm(n,0,sig);
#u=runif(n,0,1);e=(u<=0.95)*rnorm(n,0,1)+(u>0.95)*rnorm(n,0,5);
  sig=1.4826*median(abs(e-median(e)))
#v=1; e=rt(n,v); sig=1.4826*median(abs(e-median(e)));
u=runif(n,0,1);x=cbind(rnorm(n,0,1),rnorm(n,0,1));
y=(u<=pr[1])*(bet[1,1]+bet[1,2]*x[,1]+bet[1,3]*x[,2]+e)+(u>pr[1]&u<pr[1]
+pr[2])*(bet[2,1]+bet[2,2]*x[,1]+bet[2,3]*x[,2]+e)+(u>pr[1]
+pr[2]&u<1-pr[4])*(bet[3,1]+bet[3,2]*x[,1]
+bet[3,3]*x[,2]+e)+(u>1-pr[4])*(bet[4,1]+bet[4,2]*x[,1]+bet[4,3]*x[,2]+e)
y[1:numoutlier]=y[1:numoutlier]+runif(numoutlier,20,30)
bt=Sys.time();est1=regmixmodel.sel(x,y,k=upk, arbvar = 0, maxit=200);
time1[i]=Sys.time()-bt
numcompaic[i]=est1[1,upk+1];numcompbic[i]=est1[2,upk+1];
numcompcaic[i]=est1[3,upk+1];numcompicl[i]=est1[4,upk+1];
bt=Sys.time();est2=robregmixmodel.sel(x,y,k=upk,alpha=0.95,numini=40);
time2[i]=Sys.time()-bt
```

```
numcompaic1[i]=est2[1,upk+1];numcompbic1[i]=est2[2,upk+1];
numcompcaic1[i]=est2[3,upk+1];numcompicl1[i]=est2[4,upk+1];
}
time=Sys.time()-bt

#######
#Simulation study m=2
###########################
set.seed(10^8)
bt=Sys.time()
repnum=200;n=200;m=2;
alpha=0.95       #trim proportion
numoutlier=round(n*(1-alpha))
bet=matrix(c(0,0,1,-1,1,-1),nrow=2);p=3;
pr=c(0.4,0.6); upk=10;
numcompaic=rep(0,repnum);numcompbic=numcompaic;
numcompcaic=numcompaic;numcompicl=numcompaic;
numcompaic1=rep(0,repnum);numcompbic1=numcompaic;
numcompcaic1=numcompaic;numcompicl1=numcompaic;
time1=c();time2=c();
for(i in seq(repnum)){
print(i)
sig=1;e=rnorm(n,0,sig);
#u=runif(n,0,1);e=(u<=0.95)*rnorm(n,0,1)+(u>0.95)*rnorm(n,0,5);
sig=1.4826*median(abs(e-median(e)))
#v=1; e=rt(n,v); sig=1.4826*median(abs(e-median(e)));
u=runif(n,0,1);x=cbind(rnorm(n,0,1),rnorm(n,0,1));
```

```
y=(u<=pr[1])*(bet[1,1]+bet[1,2]*x[,1]+bet[1,3]*x[,2]+e)+(u>pr[1])*(bet[2,1]
+bet[2,2]*x[,1]+bet[2,3]*x[,2]+e)
y[1:numoutlier]=y[1:numoutlier]+runif(numoutlier,7,10)
bt=Sys.time();est1=regmixmodel.sel(x,y,k=upk, arbvar = 0, maxit=200);
time1[i]=Sys.time()-bt
numcompaic[i]=est1[1,upk+1];numcompbic[i]=est1[2,upk+1];
numcompcaic[i]=est1[3,upk+1];numcompicl[i]=est1[4,upk+1];
bt=Sys.time();est2=robregmixmodel.sel(x,y,k=upk,alpha=0.95,numini=40);
time2[i]=Sys.time()-bt
numcompaic1[i]=est2[1,upk+1];numcompbic1[i]=est2[2,upk+1];
numcompcaic1[i]=est2[3,upk+1];numcompicl1[i]=est2[4,upk+1];
}
time=Sys.time()-bt
```