Semi-parametric mixtures of partially linear models

by

Ruan Jean du Randt

Submitted in partial fulfillment of the requirements for the degree
Masters of Science eScience
in the Faculty of Natural and Agricultural Science
University of Pretoria, Pretoria

2022

Semi-parametric mixtures of partially linear models

by

Ruan Jean du Randt

E-mail: u17015881@tuks.co.za

Abstract

This mini-dissertation considers semi-parametric finite mixtures of partially linear models with Gaussian errors and focuses on the estimation procedure for such models. The semi-parametric structure allows for flexible modelling of the expected value of the response variable. These models are used in cases where the regression structure include both parametric and non-parametric covariate structures. We demonstrate the properties of the profile likelihood expectation maximisation algorithm (PL-EM) using a simulation study. The estimation algorithm is also demonstrated on real data. Overall, the estimation procedure is adequate in estimating the parameters of the mixtures of partially linear models from the results obtained in both the simulation study and the real-world application.

Keywords: EM algorithm, local kernel regression, non-parametric, profile likelihood, semi-parametric.

Supervisors : Prof. S. M. Millard
Prof. F. H. J. Kanfer

Department : Department of Statistics

Degree : Master of Science eScience

Acknowledgements

The support of the DSI-NICIS National e-Science Postgraduate Teaching and Training Platform (NEPTTP) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the NEPTTP.

Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1

Introduction

1.1 Motivation

Partially linear models (PLMs), introduced by Engle et al. (1986), are used in cases where the regression structure includes both parametric and non-parametric covariate structures. These models extend the linear model by adding a covariate that is not parametrically related to the response variable. This is especially useful when working with data that is known to have a non-parametric relationship and accommodates the introduction of linear dependent covariates.

Since the introduction of PLMs numerous studies have been conducted on the model. Heckman (1986) proposed that the non-parametric component can be estimated using spline functions, Robinson (1988) considered the multi-variate version of the model, Boente et al. (2006) used Pearson residuals and regression splines to introduce a robust method for estimating PLMs. The PLM for a dependent variable Y can be defined as

$$Y = \boldsymbol{x}^T\boldsymbol{\beta} + g(U) + \epsilon \tag{1.1}$$

where

Y is the response variable,

$\boldsymbol{x}^T$ is the transpose of a p-dimensional vector of explanatory variables,

$\boldsymbol{\beta}$ is a p-dimensional vector of regression coefficients,

$g(\cdot)$ is an unknown but smooth function,

U is an univariate explanatory variable, and

$\epsilon \sim N(0, \sigma^2)$ is the random error term.

Considering a population of homogeneous data, which has no underlying classes, the PLM is especially effective when modeling samples from such a population. However, in fields such as Biology (see Mamitsuka et al., 2018), climate (see Mallya et al., 2015), business (see Janka and Guenther, 2018), and many other, sub-populations are common, which results in a heterogeneous population. In statistics and other disciplines mixture modeling has received much attention for analysing heterogeneous data. Peel and MacLahlan (2000) contains a systematic review of finite mixture models.

Finite mixture models were extended by Goldfeld and Quandt (1973) to model regression data. Mixture regression models have been implemented in fields such as Biology (see Wang et al., 1996) and Medicine (see Green and Richardson, 2002), because they are able to model heterogeneous data. Mixture models are utilised when it is suspected that data can be clustered into underlying classes, or groups. These clusters of observations usually share some meaningful pattern of response on the covariates. The probability that an observation belongs to a certain class is obtained by using the observed data. These probabilities of belonging are then used to identify the latent classes. Underlying, or latent classes within the population are referred to as components in mixture regression. When considering K of these classes, it is possible to model these with K linear regression components, using a finite mixture regression model. A mixture of linear

regressions for the response variable Y is defined as

$$Y = \begin{cases} \boldsymbol{x}^T\boldsymbol{\beta}_1 + \epsilon_1 & \text{with probability } \pi_1 \\ \boldsymbol{x}^T\boldsymbol{\beta}_2 + \epsilon_2 & \text{with probability } \pi_2 \\ \dots & \\ \boldsymbol{x}^T\boldsymbol{\beta}_K + \epsilon_K & \text{with probability } \pi_K \end{cases}$$

where

Y the response variable,

$\boldsymbol{x}^T$ the transpose of a p-dimensional vector of explanatory variables,

$\boldsymbol{\beta}_k$ a p-dimensional vector of regression coefficients of the k^{th} component for $k = 1, 2, \dots, K$,

π_k are the mixing probabilities $0 < \pi_k < 1$ for all $k = 1, 2, \dots, K$ and $\sum_{k=1}^{K} \pi_k = 1$,

ϵ_k are the random error terms.

The model is a mixture of Gaussian distributions regression models when the component distribution of $Y \sim N(\boldsymbol{x}^T\boldsymbol{\beta}_k, \sigma_k^2)$ for $k = 1, 2, \dots, K$. In this mini-dissertation we consider the combination of the PLMs and the mixture of Gaussian distributions regression models that is based on the work of Wu and Liu (2017). The model use PLMs to model the expected value of the normal distribution instead of a linear regression model.

The Gaussian mixture regression model uses an Expectation Maximisation (EM) algorithm to determine the estimates for the parameters. The EM-algorithm, first introduced by Dempster et al. (1977), is used when working with incomplete data to obtain the maximum likelihood parameter estimates. The algorithm consist of two steps; the expectation step (E-step) and the maximisation step (M-step). The two steps are repeated until the algorithm convergence occurs.

This mini-dissertation considers a modified algorithm that incorporates the concepts from EM-algorithm, local kernel regression, and the profile likelihood. The algorithm

accommodates the non-parametric part of the model whilst keeping the desired properties of the original EM-algorithm to estimate the parametric part of the model. The modified algorithm is applied to simulation data in the form of a Monte Carlo simulation and a real world example in the form of a Miles Per Gallon dataset obtained from Dua and Graff (2017). Additionally, the results also explore methods to determine the number of components and bandwidth choice of the model.

1.2 Objectives

The objectives of this mini-dissertation are to:

- formulate the mixtures of PLMs,
- provide an estimation procedure for the mixtures of PLMs,
- apply the mixtures of PLMs on a simulated dataset using the estimation procedure,
- apply the mixtures of PLMs on a real world example, and to
- determine and discuss the performance of the mixtures of PLMs.

1.3 Outline

The structure of this mini-dissertation is as follows:

- **Chapter 2** focuses on the formation of the model and the profile likelihood expectation maximisation algorithm to estimate the model parameters. We also address the number of components and the choice of bandwidth that is needed in the mixtures of PLM.

- **Chapter 3** provides a simulation study to investigate the properties of the estimation procedure using Monte Carlo simulations.
- **Chapter 4** consists of a real-world application of the mixtures of PLM.
- **Chapter 5** is composed of a conclusion on the research done, provides recommendations, and ideas on future research works.

Chapter 2

Mixtures of partially linear models

2.1 Introduction

This chapter considers the semi-parametric mixtures of PLMs introduced by Wu and Liu (2017). This model combines the PLM (Equation 1.1) and the Gaussian mixture regression model by considering a PLM as the expected value of the Gaussian mixture regression model. Hence, the model is able to model both heterogeneous data and semi-parametric data, simultaneously. Section 2.2 defines the mixtures of PLMs. This is followed by the derivation of the proposed profile likelihood EM-algorithm as the estimation procedure of the model in Section 2.3. Section 2.4 addresses the problem of selecting the number of components by investigating the BIC method. Section 2.5 discusses the use of cross-validation to determine the optimal bandwidth for the model. A brief synopsis of the chapter is provided in Section 2.6.

2.2 Model definition

The mixture of PLMs is defined as

$$Y|_{\boldsymbol{X}=\boldsymbol{x},U=u} \sim \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}^T\boldsymbol{\beta}_k + g_k(u), \sigma_k^2), \tag{2.1}$$

where

Y is the response variable,

$\boldsymbol{X}$ is a p-dimensional vector of explanatory variables,

U is an univariate explanatory variable,

K denotes the number of components,

$k = 1, 2, \ldots, K$,

$\boldsymbol{\beta}_k$ is the regression coefficients of the k^{th} component,

$g_k(\cdot)$ is the unknown smooth function of the k^{th} component,

π_k is the mixing probability of component k, $0 < \pi_k < 1$ for all values of k and $\sum_{k=1}^{K} \pi_k = 1$,

σ_k^2 is the variance of component k (see Wu and Liu, 2017).

For a random sample of size n, let $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_n)^T$ a $n \times 1$ vector, $\boldsymbol{u} = (u_1, u_2, \ldots, u_n)^T$ a $n \times 1$ vector, $\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \\ \ldots \\ \boldsymbol{x}_n^T \end{pmatrix}$ a $p \times n$ matrix and $\boldsymbol{\beta}_k$ a $p \times 1$ vector of the regression coefficients for the k^{th} component. Then, a random sample from this model would be

$$Y_i|_{\boldsymbol{x}_i,u_i} \sim N(\boldsymbol{x}_i^T\boldsymbol{\beta}_k + g_k(u_i), \sigma_k^2) \quad \text{with probability } \pi_k,$$

where $i = 1, 2, \ldots, n$ denotes the i^{th} observation of the random sample.

2.3 Model estimation

Let $\mu_{i,k} = \boldsymbol{x}_i^T\boldsymbol{\beta}_k + g_k(u_i)$ for the i^{th} observation, $i = 1, 2, \ldots, n$, and k^{th} component, $k = 1, 2, \ldots, K$, and $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_K)$ where $\boldsymbol{\theta}_k = (\pi_k, \boldsymbol{\beta}_k, \sigma_k^2, g_k(\cdot))$. The mixture distribution of y_i is then given by

$$f_Y(y_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \phi\left(y_i|\mu_{i,k}, \sigma_k^2\right).$$

Then the likelihood of the data $(y_i, \boldsymbol{x}_i, u_i, i = 1, 2, \ldots, n)$ can be written as

$$\begin{aligned} L(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u}) &= \prod_{i=1}^{n} f_Y(y_i|\boldsymbol{\theta}), \\ &= \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \, \phi(y_i|\mu_{i,k}, \sigma_k^2), \end{aligned}$$

it then follows that the log-likelihood is given by

$$l(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u}) = \sum_{i=1}^{n} \ln \left[\sum_{k=1}^{K} \pi_k \, \phi(y_i|\mu_{i,k}, \sigma_k^2) \right]. \tag{2.2}$$

A profile likelihood EM algorithm (PL-EM) is used to estimate this semi-parametric model (see Wu and Liu, 2017). The profile log-likelihood is derived by first fixing $\boldsymbol{\beta}_k$ and constructing the least-favourable curve $g_k(\cdot)$ (see Severini and Wong, 1992). For a given $\boldsymbol{\beta}_k$, let $\mu_{i,k} = \boldsymbol{x}_i^T\boldsymbol{\beta}_k + g_k(u_i)$. The profile log-likelihood that corresponds to Equation 2.2 can then be written as

$$l_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u}) = \sum_{i=1}^{n} \ln \left[\sum_{k=1}^{K} \pi_k \, \phi(y_i|\mu_{i,k}, \sigma_k^2) \right], \tag{2.3}$$

where $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_K)$. The joint distribution of $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_n)$ and $\boldsymbol{\Delta} = (\Delta_1, \Delta_2, \ldots, \Delta_n)$ is given by

$$\begin{aligned} \mathbb{P}\left[\boldsymbol{Y}, \boldsymbol{\Delta}|\boldsymbol{\theta}\right] &= \mathbb{P}\left[\boldsymbol{Y}|\boldsymbol{\Delta}, \boldsymbol{\theta}\right] \mathbb{P}\left[\boldsymbol{\Delta}|\boldsymbol{\theta}\right], \\ &= \prod_{i=1}^{n} \mathbb{P}\left[y_i|\Delta_i, \boldsymbol{\theta}\right] \mathbb{P}\left[\Delta_i|\boldsymbol{\theta}\right], \end{aligned}$$

where $\boldsymbol{\Delta}$ is the observed component allocations (see Huang et al., 2013). Also, $\mathbb{P}\left[y_i|\Delta_i, \boldsymbol{\theta}\right] = \phi\left(y_i|\mu_{i,k}, \sigma_k^2\right)$ and $\mathbb{P}\left[\Delta_i = k|\boldsymbol{\theta}\right] = \pi_k$. Then

$$\mathbb{P}\left[\boldsymbol{Y}, \boldsymbol{\Delta}|\boldsymbol{\theta}\right] = \prod_{i=1}^{n}\prod_{k=1}^{K}[\pi_k \phi\left(y_i|\mu_{i,k}, \sigma_k^2\right)]^{I(\Delta_i=k)},$$

where $I(\cdot)$ is an indicator function. Define $\boldsymbol{Z} = (z_{ik})$, a binary indicator function for observation i to be observed from component k, with

$$z_{ik} = \begin{cases} 1, & \text{if observation } i \text{ is from component } k \\ 0, & \text{otherwise} \end{cases}.$$

Now the complete data profile log-likelihood that corresponds to profile log-likelihood, Equation 2.3, is given by

$$l_{\boldsymbol{\beta}}^{c}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{X}, \boldsymbol{u}) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}\left[\ln\left(\pi_k\right) + \ln\left(\phi(y_i|\mu_{i,k}, \sigma_k^2)\right)\right], \tag{2.4}$$

where $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_K)$. In the E-step of the algorithm the responsibility of each observation belonging to a specific component, r_{ik}, is calculated. This is estimated by taking the expected value of the latent variable (see Huang et al., 2013), that is

$$\begin{aligned} r_{ik} &= \mathbb{E}[z_{ik}], \\ &= \mathbb{P}\left[z_{ik} = 1|y_i, \boldsymbol{x}_i, u_i, \boldsymbol{\theta}_k\right], \\ &= \frac{\mathbb{P}\left[z_{ik} = 1, y_i|\boldsymbol{x}_i, u_i, \boldsymbol{\theta}_k\right]}{\mathbb{P}\left[y_i|\boldsymbol{x}_i, u_i, \boldsymbol{\theta}_k\right]}, \\ &= \frac{\mathbb{P}\left[y_i|z_{ik} = 1, \boldsymbol{x}_i, u_i, \boldsymbol{\theta}_k\right]\mathbb{P}\left[z_{ik} = 1|\boldsymbol{x}_i, u_i, \boldsymbol{\theta}_k\right]}{\mathbb{P}\left[y_i|\boldsymbol{x}_i, u_i, \boldsymbol{\theta}_k\right]}, \\ &= \frac{\pi_k\, \phi\left(y_i|\mu_{i,k}, \sigma_k^2\right)}{\sum_{k=1}^{K}\pi_k\, \phi\left(y_i|\mu_{i,k}, \sigma_k^2\right)}. \end{aligned} \tag{2.5}$$

Substituting z_{ik} in Equation 2.4 with $\mathbb{E}[z_{ik}] = r_{ik}$ gives the function required for the M-step, that is

$$Q_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u}) = \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}\left[\ln\left(\pi_k\right) + \ln\left(\phi(y_i|\mu_{i,k}, \sigma_k^2)\right)\right]. \tag{2.6}$$

The estimates of $(\pi_k, \boldsymbol{\beta}_k, \sigma_k^2)$ is obtained by maximising Equation 2.6 (see Wang et al., 2016). Firstly, maximising with respect to π_k with the constraint $\sum_{k=1}^{K} \pi_k = 1$ requires Lagrangian multipliers. That is maximising

$$Q^*_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u}) = \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}\left[\ln(\pi_k) + \ln\left(\phi(y_i|\mu_{i,k}, \sigma_k^2)\right)\right] + \lambda\left(\sum_{k=1}^{K}\pi_k - 1\right).$$

Equating the partial derivatives of $Q^*_{\boldsymbol{\beta}}$ in terms of π_k and λ and setting equal to zero yields

$$\begin{aligned} \frac{\partial Q^*_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u})}{\partial \pi_k} &= \sum_{i=1}^{n}\frac{r_{ik}}{\pi_k} + \lambda = 0, \\ \frac{\partial Q^*_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u})}{\partial \lambda} &= \sum_{k=1}^{K}\pi_k - 1 = 0. \end{aligned} \tag{2.7}$$

Summing Equation 2.7 over k and multiplying by π_k gives

$$\sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik} + \lambda\sum_{k=1}^{K}\pi_k = 0.$$

Since, $\sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik} = \sum_{i=1}^{n} 1 = n$ and $\sum_{k=1}^{K}\pi_k = 1$ it follows that $\lambda = -n$. Solving for π_k by substituting $\lambda = -n$ into Equation 2.7 gives

$$\begin{aligned} \sum_{i=1}^{n}\frac{r_{ik}}{\pi_k} - n &= 0, \\ \pi_k &= \sum_{i=1}^{n}\frac{r_{ik}}{n}. \end{aligned} \tag{2.8}$$

In order to estimate the non-parametric component a local profile likelihood function is required (see Huang et al., 2013). This is given by

$$Q_{\boldsymbol{\beta},h}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u}, t) = \sum_{i=1}^{n}\sum_{k=1}^{K} r_{ik}\left[\ln(\pi_k) + \ln\left(\phi(y_i|\boldsymbol{x}_i^T\boldsymbol{\beta}_k + g_k(t), \sigma_k^2)\right)\right] K_h(u_i - t), \tag{2.9}$$

where h is the bandwidth and $t \in \mathbb{R}$. Considering only the $g_k(t)$ terms for component k in Equation 2.9 gives

$$Q^*_{\boldsymbol{\beta},h}(\boldsymbol{\theta}|\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{u}, t) = \sum_{i=1}^{n} r_{ik}\left(-\frac{1}{2}\left(\frac{y_i - \boldsymbol{x}_i^T\boldsymbol{\beta}_k - g_k(t)}{\sigma_k}\right)^2\right) K_h(u_i - t),$$

partially differentiating in terms of $g_k(t)$ and setting equal to zero gives

$$\begin{aligned}\frac{\partial Q^*_{\boldsymbol{\beta},h}(\boldsymbol{\theta}|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{u},t)}{\partial\left(g_k(t)\right)} &= \sum_{i=1}^{n} r_{ik}K_h(u_i-t)\left(\frac{y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(t)}{\sigma_k}\right) &= 0,\\ \sum_{i=1}^{n} r_{ik}K_h(u_i-t)\left(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k\right) - g_k(t)\sum_{i=1}^{n} r_{ik}K_h(u_i-t) &= 0.\end{aligned}$$

The estimate of $g_k(\cdot)$ at point t is then simply given by

$$g_k(t) = \frac{\sum_{i=1}^n r_{ik}K_h(u_i-t)(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k)}{\sum_{i=1}^n r_{ik}K_h(u_i-t)}.$$

Selecting t from the $\boldsymbol{u}$, say $u_j \in \boldsymbol{u}$, then

$$g_k(u_j) = \frac{\sum_{i=1}^n r_{ik}K_h(u_i-u_j)(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k)}{\sum_{i=1}^n r_{ik}K_h(u_i-u_j)}. \tag{2.10}$$

In order to write the estimates in matrix notation, define the smoothing matrix $\boldsymbol{S}$ such that its elements are given by

$$(\boldsymbol{S})_{ij} = \frac{r_{ik}K_h(u_i-u_j)}{\sum_{i=1}^n r_{ik}K_h(u_i-u_j)}, \qquad \boldsymbol{S} = (\boldsymbol{S}_1\ \boldsymbol{S}_2\ \ldots\ \boldsymbol{S}_n), \tag{2.11}$$

where $\boldsymbol{S}_i$ denotes the i^{th} column vector of the $n \times n$ matrix $\boldsymbol{S}$ (see Severini and Wong, 1992). Then the following matrix expression for $\boldsymbol{G}$ is obtained

$$\boldsymbol{G} = \boldsymbol{S}(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{\beta}) \tag{2.12}$$

where $\boldsymbol{G} = (g_k(u_1), g_k(u_2), \ldots, g_k(u_n))$. Now, considering only the $\boldsymbol{\beta}_k$ terms in Equation 2.6 and for component k gives the following function to be maximised

$$Q^{**}_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{u}) = \sum_{i=1}^{n} r_{ik}\left(-\frac{1}{2}\left(\frac{y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)}{\sigma_k}\right)^2\right),$$

given the estimates obtained in Equation 2.10, gives

$$Q^{**}_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{u}) = \sum_{i=1}^{n} r_{ik}\left(-\frac{1}{2}\left(\frac{y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k}{\sigma_k} - \frac{1}{\sigma_k}\left(\frac{\sum_{j=1}^n r_{jk}K_h(u_j-u_i)(Y_j-\boldsymbol{x}_j^T\boldsymbol{\beta}_k)}{\sum_{j=1}^n r_{jk}K_h(u_j-u_i)}\right)\right)^2\right).$$

Taking the partial derivative in terms of $\boldsymbol{\beta}_k$ and setting equal to zero gives

$$\begin{aligned}
\frac{\partial Q^{**}_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{u})}{\partial \boldsymbol{\beta}_k} = \sum_{i=1}^{n} r_{ik}\left(\frac{y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)}{\sigma_k}\right)\left(\boldsymbol{x}_i^T-\left[\sum_{j=1}^{n}(\boldsymbol{S})_{ji}\boldsymbol{x}_j^T\right]\right) &= 0,\\
\sum_{i=1}^{n} r_{ik}\left(\frac{y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)}{\sigma_k}\right)\left(\boldsymbol{x}_i^T-\boldsymbol{S}_i^T\boldsymbol{X}\right) &= 0,\\
\sum_{i=1}^{n} r_{ik}\left(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)\right)\left(\boldsymbol{x}_i^T-\left(\boldsymbol{X}^T\boldsymbol{S}_i\right)^T\right) &= 0,\\
\sum_{i=1}^{n} r_{ik}\left(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)\right)\left(\boldsymbol{x}_i-\boldsymbol{X}^T\boldsymbol{S}_i\right)^T &= 0, \qquad (2.13)
\end{aligned}$$

Writing equation 2.13 in matrix form yields

$$\begin{aligned}
\left(\boldsymbol{X}-\boldsymbol{S}\boldsymbol{X}\right)^T\boldsymbol{R}_k\left(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{\beta}_k-\boldsymbol{G}\right) &= 0,\\
\left(\boldsymbol{X}-\boldsymbol{S}\boldsymbol{X}\right)^T\boldsymbol{R}_k\left(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{\beta}_k-\boldsymbol{S}\left(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{\beta}_k\right)\right) &= 0,\\
\left((\boldsymbol{I}-\boldsymbol{S})\boldsymbol{X}\right)^T\boldsymbol{R}_k\left((\boldsymbol{I}-\boldsymbol{S})(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{\beta}_k)\right) &= 0,\\
\tilde{\boldsymbol{X}}^T\boldsymbol{R}_k\tilde{\boldsymbol{Y}}-\tilde{\boldsymbol{X}}^T\boldsymbol{R}_k\tilde{\boldsymbol{X}}\boldsymbol{\beta}_k &= 0,
\end{aligned}$$

where $\tilde{\boldsymbol{X}}=(\boldsymbol{I}-\boldsymbol{S})\boldsymbol{X}$, $\tilde{\boldsymbol{Y}}=(\boldsymbol{I}-\boldsymbol{S})\boldsymbol{Y}$, $\boldsymbol{R}_k=\text{diag}(r_{1k}, r_{2k}, \ldots, r_{nk})$ and $\boldsymbol{I}$ denotes an $n\times n$ identity matrix. The estimate for $\boldsymbol{\beta}_k$ is then given by

$$\begin{aligned}
\tilde{\boldsymbol{X}}^T\boldsymbol{R}_k\tilde{\boldsymbol{X}}\boldsymbol{\beta}_k &= \tilde{\boldsymbol{X}}^T\boldsymbol{R}_k\tilde{\boldsymbol{Y}},\\
\boldsymbol{\beta}_k &= \left(\tilde{\boldsymbol{X}}^T\boldsymbol{R}_k\tilde{\boldsymbol{X}}\right)^{-1}\tilde{\boldsymbol{X}}^T\boldsymbol{R}_k\tilde{\boldsymbol{Y}} \qquad (2.14)
\end{aligned}$$

Next, considering only the σ_k^2 terms in Equation 2.6 and for k gives

$$Q^{***}_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{u})=\sum_{i=1}^{n} r_{ik}\left(-\frac{1}{2}\ln(\sigma_k^2)-\frac{1}{2}\left(\frac{y_i-\boldsymbol{x}_i\boldsymbol{\beta}_k-g_k(u_i)}{\sigma_k}\right)^2\right),$$

partially differentiating in terms of σ_k^2 and setting equal to zero gives

$$\begin{aligned}
\frac{\partial Q^{***}_{\boldsymbol{\beta}}(\boldsymbol{\theta}|\boldsymbol{Y},\boldsymbol{X},\boldsymbol{u})}{\partial \sigma_k^2} = \sum_{i=1}^{n} -\frac{r_{ik}}{2\sigma_k^2}+\frac{1}{2}r_{ik}\left(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)\right)^2\frac{1}{\sigma_k^4} &= 0,\\
\sum_{i=1}^{n} -r_{ik}+r_{ik}\left(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)\right)^2\frac{1}{\sigma_k^2} &= 0.
\end{aligned}$$

The estimate for σ_k^2 is then given by

$$\sigma_k^2 = \frac{\sum_{i=1}^{n} r_{ik}\left(y_i-\boldsymbol{x}_i^T\boldsymbol{\beta}_k-g_k(u_i)\right)^2}{\sum_{i=1}^{n} r_{ik}} \qquad (2.15)$$

For the purposes of this mini-dissertation convergence is determined by comparing the difference of the current log-likelihood and the previous iteration's log-likelihood. If this value is smaller than 1^{-10} then the algorithm converged. A summary of the algorithm is given below in Algorithm 2.1 (see Wu and Liu, 2017).

1. Choose a set of initial values for $\boldsymbol{\theta}^{(old)}$, that is $\pi_1^{(old)}, \ldots, \pi_K^{(old)}$, $\boldsymbol{\beta}_1^{(old)}, \ldots, \boldsymbol{\beta}_K^{(old)}$, $g_1^{(old)}(\cdot), \ldots, g_K^{(old)}(\cdot)$ and $\sigma_1^{2(old)}, \ldots, \sigma_K^{2(old)}$.

2. In the E-Step, determine the responsibilities, $r_{ik}^{(new)}$ for $i = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, K$, by evaluating

$$r_{ik}^{(new)} = \frac{\pi_k^{(old)}\, \phi\left(y_i | \boldsymbol{x}_i^T \boldsymbol{\beta}_k^{(old)} + g_k^{(old)}(u_i), \sigma_k^{2(old)}\right)}{\sum_{k=1}^{K} \pi_k^{(old)}\, \phi\left(y_i | \boldsymbol{x}_i^T \boldsymbol{\beta}_k^{(old)} + g_k^{(old)}(u_i), \sigma_k^{2(old)}\right)}.$$

3. In the M-Step, update the unknown parameters estimates $\pi_k^{(new)}$, $g_k^{(new)}(u_i)$, $\boldsymbol{\beta}_k^{(new)}$ and $\sigma_k^{2(new)}$ for $i = 1, 2, \ldots, n$ and $k = 1, 2, \ldots, K$ by using

$$\pi_k^{(new)} = \sum_{i=1}^{n} \frac{r_{ik}^{(new)}}{n}$$

$$g_k^{(new)}(u_j) = \frac{\sum_{i=1}^{n} r_{ik}^{(new)} K_h(u_i - u_j)(y_i - \boldsymbol{x}_i^T \boldsymbol{\beta}_k^{(old)})}{\sum_{i=1}^{n} r_{ik}^{(new)} K_h(u_i - u_j)}$$

$$\boldsymbol{\beta}_k^{(new)} = \left(\tilde{\boldsymbol{X}}^T \boldsymbol{R}_k^{(new)} \tilde{\boldsymbol{X}}\right)^{-1} \tilde{\boldsymbol{X}}^T \boldsymbol{R}_k^{(new)} \tilde{\boldsymbol{Y}}$$

$$\sigma_k^{2(new)} = \frac{\sum_{i=1}^{n} r_{ik}^{(new)} \left(y_i - \boldsymbol{x}_i^T \boldsymbol{\beta}_k^{(new)} - g_k^{(new)}(u_i)\right)^2}{\sum_{i=1}^{n} r_{ik}^{(new)}}$$

4. Let $\boldsymbol{\theta}_k^{(old)} = \boldsymbol{\theta}_k^{(new)}$ for $k = 1, 2, \ldots, K$

5. Repeat (2) to (4) until convergence.

Algorithm 2.1: Profile Likelihood Expectation Maximisation algorithm.

2.4 Selecting the number of components

One of the most persistent problems with mixture models is that of finding the number of components associated with the dataset. The choice of the number of components in cluster analysis usually occur when determining the number of clusters there are in the data (see McLachlan and Rathnayake, 2014). Two main methods are proposed to determine the number of components of a mixture model. The first is to use a penalised form of the log-likelihood function. This method incorporates the well-known information criteria to penalise the model for the number of parameters because the likelihood increases as the number of components increase (see McLachlan et al., 2019). The second method uses a likelihood ratio test (see Chen et al., 2001; Frühwirth-Schnatter, 2006). The primary problem with this method constitutes that the distribution under the null hypothesis is not valid.

For the purpose of this research the Bayesian Information Criteria (BIC) is used to determine the number of components. The BIC can be defined as follows (see Schwarz, 1978)

$$BIC \;=\; -2l + df\,\ln(n),$$

where l is the log-likelihood, n the sample size, and df the number of estimable parameters i.e. the degree of freedom for the model. The non-parametric part of the model contains an unknown number of degrees of freedom. For the purposes of this model define the effective degree of freedom for the non-parametric part as in Fan et al. (2001) and Wang et al. (2016).

$$df_{np} \;=\; \tau_K\, h^{-1}\, |\Omega| \left(K(0) - \frac{1}{2} \int K^2(t) dt \right),$$

where $|\Omega|$ denotes the length of the support of U, and

$$\tau_K \;=\; \frac{K(0) - 0.5 \int K^2(t)\, dt}{\int \left(K(t) - 0.5[K * K(t)] \right)^2 dt}$$

where $K * K(\cdot)$ is the convolution of $K(\cdot)$. Therefore, the following degrees of freedom for the model in 2.1 is obtained

$$df_{model} = K \times df_{np} + (p+1)K - 1.$$

where K denotes the number of components and $K(\cdot)$ denotes the kernel function.

2.5 Choice of bandwidth

The estimates contained in the PL-EM algorithm depend on the bandwidth chosen in the initialisation. This research chooses bandwidth through the usage of manifold cross-validation (CV) (see Wu and Liu, 2017). For a given number of components, K, partition D, the complete dataset, into J parts, say D_j, $j = 1, 2, \ldots, J$. For each of these subsets use D_j as testing data and the remaining data is used as the training data. The training data, denoted by $D_{(-j)}$, is then used to obtain estimates for $\hat{\boldsymbol{\theta}}_{k(-j)}$. These estimates are then used to calculate the value of $g_k(u)$ in the points of the D_j dataset. Lastly, the criterion to evaluate the CV is obtained by

$$CV_{crit} = \sum_{j=1}^{J} \sum_{i \in D_j} (y_i - \hat{y}_i)^2, \tag{2.16}$$

where

$$\hat{y}_i = \sum_{k=1}^{K} \hat{r}_{ik} \left(\boldsymbol{x}_i^T \hat{\boldsymbol{\beta}}_{k(-j)} + \hat{g}_k(u_i) \right) \tag{2.17}$$

and $\hat{r}_{ik}$ is the probability that observation i belong to group k. In order to obtain the bandwidth one selects the value that minimizes the CV_{crit} value obtained in Equation 2.16.

2.6 Summary

This chapter considered semi-parametric mixtures of PLMs, introduced by Wu and Liu (2017), in Section 2.2. The model is an extension of the Gaussian mixture regression model where there is a PLM in the mean parameter. Section 2.3 provides an estimation procedure for the various parameters in the form of a PL-EM algorithm that incorporates the concepts from EM algorithm, local kernel regression, and the profile likelihood. An overview of the algorithm is provided at the end of Section 2.3. The selection of the number of components is discussed in Section 2.4 and the BIC method is chosen to identify the number of components. Section 2.5 describes how CV can be used to determine the optimal bandwidth for the model.

The next chapter will discuss the estimation procedure of the model, Equation 2.1, using a simulation study.

Chapter 3

Simulation study

3.1 Introduction

This chapter investigates the performance of the PL-EM estimation procedure for a finite sample semi-parametric mixtures of PLMs using Monte Carlo simulations. Section 3.2 describes the setting of the simulation and provides a method to assess the performance of the non-parametric function. Scatter plots of the simulated covariates and the response variable are used to assist with determining if the mixtures of PLM can be applied to an arbitrary dataset. Section 3.3 present and discuss the results obtained after implementing the PL-EM algorithm on the simulation data. Lastly, Section 3.4 provides a summary of the work done in this chapter.

3.2 Simulation framework

Considering a 2-component mixture of PLMs with

$$\pi_1 = 0.35 \text{ and } \pi_2 = 0.65,\ \sigma_1^2 = 0.04 \text{ and } \sigma_2^2 = 0.09$$

$$m_1(x,u) = 5x + 2\cos^2(\pi u),\ m_2(x,u) = -x + \exp(2u-1)$$

where $m_i(x, u)$ is the mean function of the i^{th} component. In this simulation, $\beta_1 = 5$ and $\beta_2 = 1$ and $g_1(u) = 2\cos^2(\pi u)$ and $g_2(u) = \exp(2u - 1)$. The initial values and function definitions are chosen to be similar to that of Wu and Liu (2017). Here x and u are generated from a univariate uniform distribution on $[0, 1]$. Choosing 500 samples of sizes $n = 200$ and $n = 500$ (see Wu and Liu, 2017). A question arised on the method to determine if a component should be considered in the non-parametric component of the model or in the linear component. Therefore, to attempt to find an answer, consider the scatter plots of the two covariates with respect to the response variable for the simulated data (see Figures 3.1 and 3.2). This will provide a guide when the model is implemented on real world data. Note that for the purpose of this exploration analysis, a random sample of size 500 is chosen.

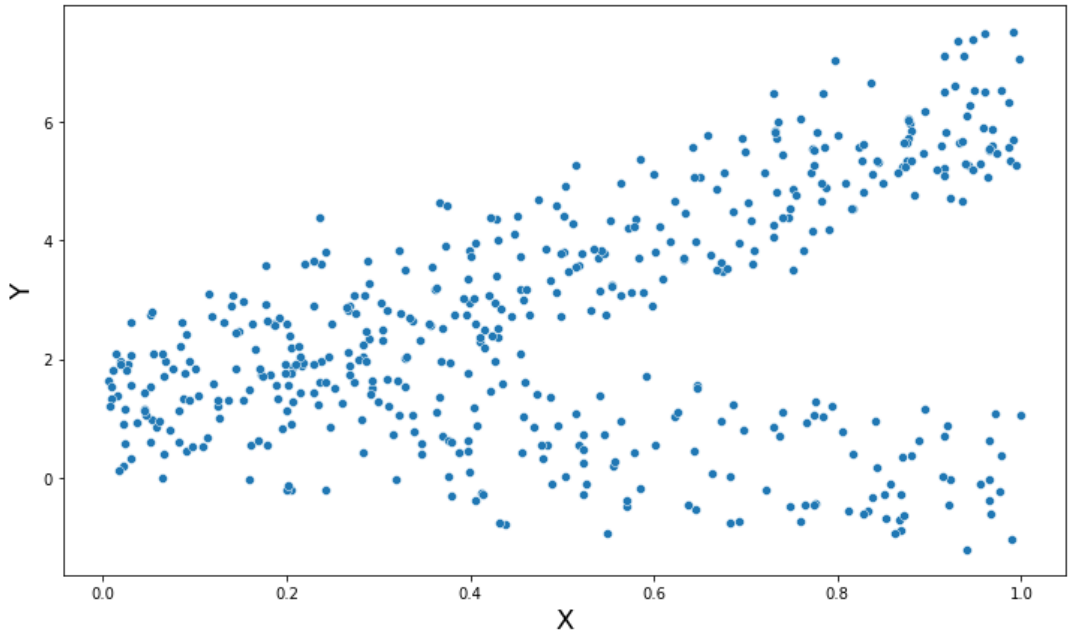


Figure 3.1: Scatter plot of Y versus X of the simulated data when $n = 500$.

Firstly, considering the two variables that are expected to be linear, X and Y. Figure 3.1 displays the scatter plot of the Y versus X variables of the simulation data. The plot

reveals that two different components affect the response variable. The one component appears to have a positive linear relationship and the other a negative, as expected. This implies that when determining the linear component, one needs to find a covariate that is linear in the parameters with respect to the response variable.

The second part would be the ability to determine the non-parametric covariate. This is done by considering the variables that are expected to have such a relationship, Y and U. The scatter plot of the Y versus U of the simulated data is shown in Figure 3.2. Unlike the previous case, it is not as clear to see that there are two components from this plot. However, there does appear to be a relationship between Y and U.

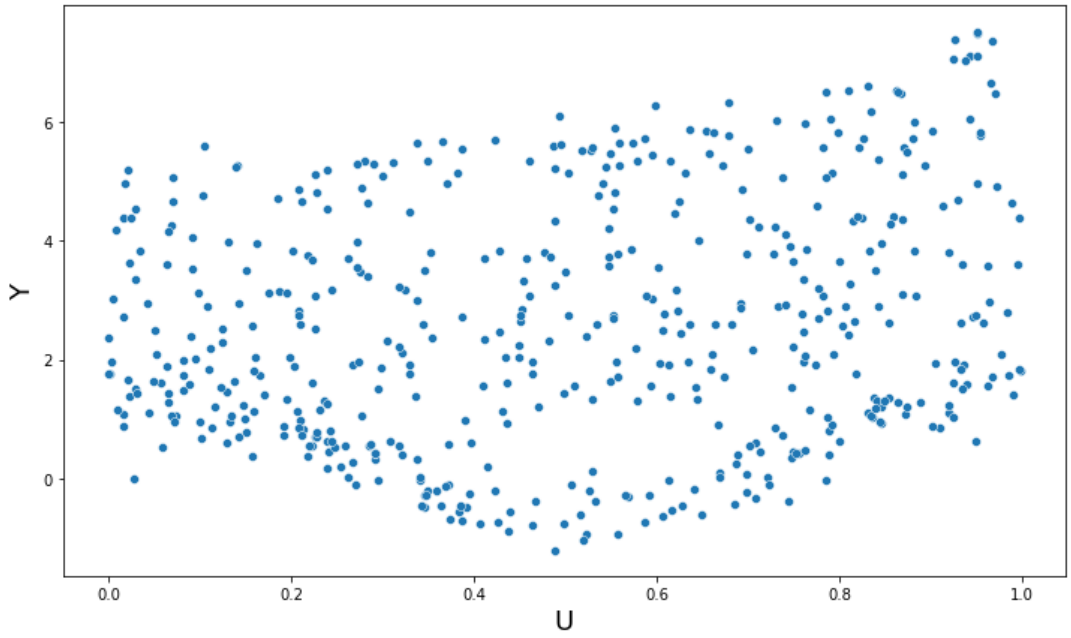


Figure 3.2: Scatter plot of Y versus U of the simulated data when $n = 500$.

Although, this does not provide a clear method of determining the non-parametric covariate of the model. Therefore, for the purpose of this mini-dissertation, the non-parametric covariate will be the variable that is not linear in the parameters with respect to the response variable.

The simulation uses square root of the average squared errors (RASE) to assess the performance of the estimator $\hat{g}_k(\cdot)$, that is

$$RASE_g = \sqrt{N^{-1}\sum_{k=1}^{K}\sum_{i=1}^{N}[\hat{g}_k(u_i) - g_k(u_i)]^2}, \tag{3.1}$$

where $i = 1, \ldots, N$ and u_i are points divided equally in the range of the covariate u. The kernel function used in the simulation is the Epanechnikov kernel, $K(u) = \frac{3}{4}(1-u^2)_+$, and as per convention the number of grid points are chosen to be $N = 100$.

3.3 Results

3.3.1 Number of components

In order to evaluate the efficiency of using the BIC to identify the number of components using 1 to 5 components (similar to Wu and Liu, 2017), we fit the mixtures of PLMs using five distinct bandwidths and compare the criterion for each dataset. Table 3.1 reports the frequencies of the selected K's over the 500 simulations, for a given bandwidth. This is done by selecting the number of components, $K = 1, 2, 3, 4,$ or 5, that minimises the BIC score for each of the simulated datasets using the five h values.

The last column of Table 3.1 contains the component chosen when considering all of the bandwidths. In the 500 simulations the BIC selected two components 93.6% and 96% for $n = 200$ and $n = 500$, respectively, as indicated in the last column. The lowest proportion is given when the bandwidth is the largest and even in that case, two components where chosen 90.4% and 93.2% for $n = 200$ and $n = 500$, respectively. This is still quite accurate considering the sample sizes and the number of degrees of freedom in the model. These results indicate that using the BIC to determine the number of components works reasonably well.

Table 3.1: Number of times K were selected by the BIC method.

$n = 200$	$h = 0.06$	$h = 0.09$	$h = 0.12$	$h = 0.15$	$h = 0.18$	$\min_h BIC_h$
$K = 1$	0	0	0	0	0	0
$K = 2$	479	476	472	461	452	468
$K = 3$	19	23	23	33	40	27
$K = 4$	2	0	4	1	4	1
$K = 5$	0	1	1	5	4	4
$n = 500$	$h = 0.04$	$h = 0.06$	$h = 0.08$	$h = 0.10$	$h = 0.12$	$\min_h BIC_h$
$K = 1$	0	0	0	0	0	0
$K = 2$	498	488	482	477	466	480
$K = 3$	2	12	16	20	28	18
$K = 4$	0	0	1	0	4	2
$K = 5$	0	0	1	3	2	0

3.3.2 Choice of bandwidth

Next, we use 5-fold CV to determine the optimal bandwidth. Here, for a given n, several datasets are simulated, for each of these datasets a bandwidth is chosen by using the CV_{crit} value in Equation 2.16. The optimal bandwidth, $\hat{h}$, for a given n is then determined by taking the average of all of bandwidths obtained. For simplicity the value is rounded to the nearest second decimal. The performance of the method is illustrated by taking three bandwidths: $\frac{2\hat{h}}{3}$, $\hat{h}$, and $\frac{3\hat{h}}{2}$, and calculating the mean squared error (MSE) for all of the parameters and the RASE for the non-parametric estimates. The notation MSE_{π_1} is used to indicate the MSE of the parameter π_1, a similar notation is used for the other parameters.

Table 3.2: Square errors of the parameter estimates for different bandwidths and sample sizes.

n	h	MSE_{π_1}	MSE_{β_1}	MSE_{β_2}	$MSE_{\sigma_1^2}$	$MSE_{\sigma_2^2}$	$RASE_g$
	0.08	0.001208	0.009643	0.01771	0.007665	0.01885	0.1488
200	0.12	0.001225	0.009744	0.01225	0.005140	0.01455	0.1438
	0.18	0.001242	0.01015	0.01130	0.01707	0.01354	0.1812
	0.055	0.0004511	0.003532	0.005459	0.003174	0.007650	0.09894
500	0.08	0.0004522	0.003409	0.004462	0.002078	0.005943	0.08923
	0.12	0.0004540	0.003418	0.004029	0.002073	0.005327	0.09873

Table 3.2 contains the MSE and RASE results for the different sample sizes under various h lengths using all 500 simulations. The method shows that the RASE is the smallest for the optimal bandwidth, in the case of $n = 200$ and $n = 500$. The MSE of β_1 and β_2 mostly decrease as the bandwidth increases when $n = 500$. This is because the estimation procedure adds more weight to the regression coefficients as the bandwidth increases. However, this is not the case for the smaller sample of $n = 200$. We intend to investigate the smaller samples in future work. The results indicate that using the CV method to determine the bandwidth will provide the optimal bandwidth that minimises the RASE.

3.3.3 Parameter estimates

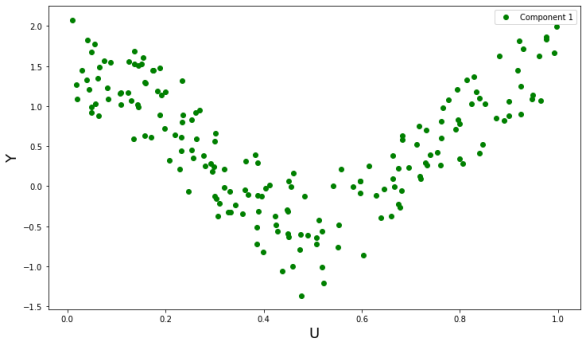
In order to show the effectiveness of the PL-EM algorithm one of the Monte Carlo simulations, selected randomly, is again fitted to the model and a 95% confidence interval is obtained for the non-parametric function using the bootstrap procedure. Note that the confidence interval becomes wider at the boundaries of the function. This is due to the method that is used to estimate the non-parametric function.

The parameter estimates under the optimal bandwidth are $\hat{\beta}_1 = 4.98$, and $\hat{\beta}_2 = -0.97$. Also, $\hat{\sigma}_1^2 = 0.038$, $\hat{\sigma}_2^2 = 0.093$, $\hat{\pi}_1 = 0.36$, and $\hat{\pi}_2 = 0.64$. These are sufficiently close to the true parameters used to generate the dataset.

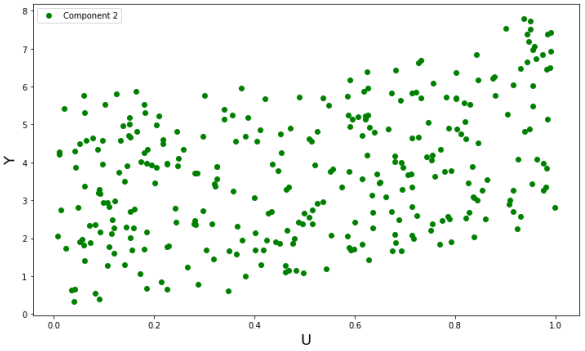
Figure 3.3 show the results of the estimated non-parametric part of the model after using hard classification. From the scatter plots in Figures 3.3a and 3.3b there appears to be a significant difference between the two components, as expected. The components show a resemblance of the functions defined to simulate the model.

The results in Figures 3.3c and 3.3d show the real and estimated $g_i(\cdot)$'s for both of the components along with 95% confidence intervals obtained using a bootstrap method. The true function values are within a 95% confidence interval of the estimated function values for the majority of the points. The confidence intervals become wider at the boundaries of the estimated range due to the nature of kernel estimation. The widening is evident in both components and is also the area where the real function values are not always contained within the interval. However, the PL-EM algorithm estimates the parameters of the model quite well.

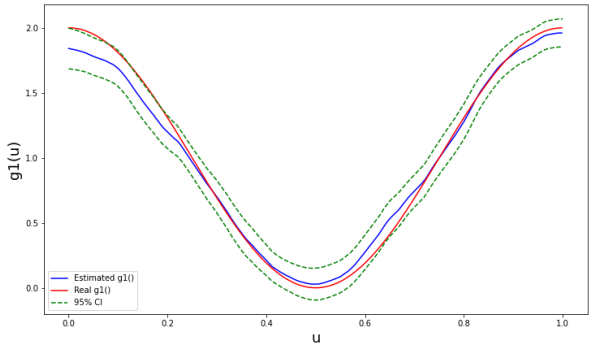
It is expected that the non-parametric part of the model should be equal to the response variable without the linear component. Therefore, when considering only the observations that the model classified into a specific component, it is evident that the PL-EM algorithm works in estimating the model. Figures 3.3e and 3.3f show the results of the points classified into a component without their linear part along with the real and estimated g_i's. When comparing the two components it is clear that the mixing probability of the first is smaller than that of the second, as expected. The results also show that the estimation error at the edges of the range is less of a problem, since the data is still quite centered around the estimated function values in these points.



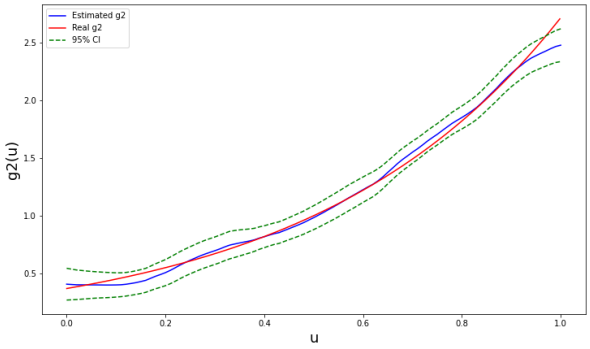
(a) Scatter plot of Y versus U of the observations hard classified into component 1.



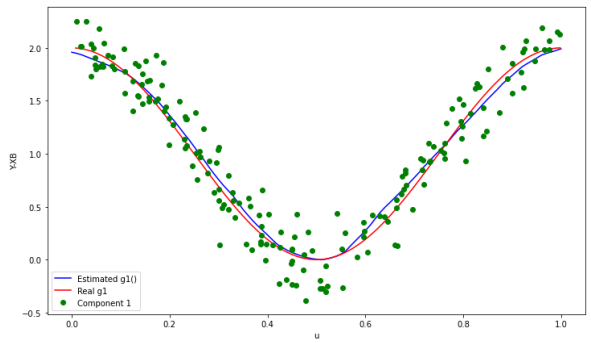
(b) Scatter plot of Y versus U of the observations hard classified into component 2.



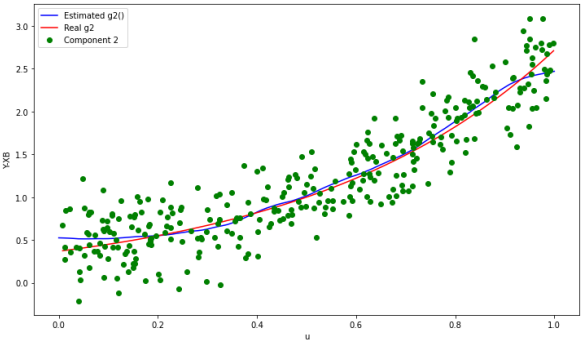
(c) The true and estimated non-parametric function values, $g_1(u)$, and a 95% CI for component 1.



(d) The true and estimated non-parametric function values, $g_2(u)$, and a 95% CI for component 2.



(e) The true and estimated non-parametric function values with the response variable without the estimated linear results for component 1.



(f) The true and estimated non-parametric function values with the response variable without the estimated linear results for component 2.

Figure 3.3: Results for the non-parametric part of the model estimation.

The results show that the model fits the data quite well. Thus, with the consideration of all the results, it can be concluded that the estimation procedure, the PL-EM algorithm, is effective at determining the parameters for the mixtures of PLMs. The algorithm works sufficiently well with a moderate sample size. The study is done as a replication of the work by Wu and Liu (2017) and therefore there are no comparitive studies done.

3.4 Summary

In this chapter, the performance of a finite sample semi-parametric mixtures of PLMs was investigated using Monte Carlo simulations. Section 3.2 describes the setting of the simulation by indicating all of the parameters used to simulate the data. Also, showing scatter plots of the simulated data to assist with determining the variables needed to implement the mixtures of PLM to an arbitrary dataset. The Section concludes by providing a method to assess the performance of the non-parametric function.

Section 3.3 contains three parts. Firstly, it determines the number of components in the model using the BIC illustrates the effectiveness thereof by fitting the model using various bandwidths. The results show that using the BIC to determine the number of components works reasonably well. Secondly, Section 3.3 indicates the effectiveness of the CV method to determine the optimal bandwidth. The performance of the method is illustrated by taking three bandwidths: $\frac{2\hat{h}}{3}$, $\hat{h}$, and $\frac{3\hat{h}}{2}$, where $\hat{h}$ is the optimal bandwidth as per the CV method. The results indicate that using the CV method to determine the bandwidth will provide the optimal bandwidth tat minimises the RASE. The section concludes by illustrating the results of using the PL-EM algorithm as the estimation procedure. The chapter shows that the BIC, CV, and PL-EM estimation procedure work sufficiently well on simulated data. The next chapter will implement the estimation

procedure on a real-world example.

Chapter 4

Application on real data

4.1 Introduction

This chapter considers a real-world application of the mixtures of PLMs by applying the model to a Miles Per Gallon dataset obtained from Dua and Graff (2017). A description of the data is given in Section 4.2. This is followed by an exploratory analysis of the data in Section 4.2. Section 4.4 discuss the results obtained from fitting the model to the data and Section 4.5 contains a synopsis of the work done in this chapter.

4.2 Description

The data describe the engines of various vehicles used in city driving. The dataset contains 398 instances and 8 attributes. The attributes are described in Table 4.1. The data do not contain any missing values.

Table 4.1: Description of the attributes used in the mpg dataset

Attribute	Description	Type
Miles per Gallon	Number of miles that can be driven per gallon of fuel	Continuous
Cylinders	Number of cylinders the vehicle has	Discrete
Displacement	The engine size in cubic inches	Continuous
Weight	The weight of the vehicle in pounds	Continuous
Acceleration	Time, in seconds, to get from 0-60 mph	Continuous
Model year	Year the vehicle was released	Discrete
Horsepower	Measures the engine power output	Continuous
Origin	An indication of where the vehicle was made	Discrete

The attributes differ significantly in magnitude and scale. Therefore, for the purposes of this mini-dissertation the data is first standardised, that is

$$X_{new} = \frac{X - \bar{X}}{\sigma_X}. \tag{4.1}$$

The resultant data are used to implement a mixture of PLM. However, since *Cylinders*, *Model year* and *Cylinders* are discrete random variables they are not studied further. The aim is to evaluate the effectiveness of the estimation procedure and therefore we decided to exclude the variables. Additionally, the three attributes are not linearly related to any other attributes. Thus, only the remaining five attributes will be considered further to fit the model. This assists to reduce the computational constraints.

Table 4.2: Descriptive statistics of the standardised attributes

Standardised Attribute	Median	Minimum	Maximum
Miles per Gallon	-0.089	-1.851	0.712
Displacement	-0.415	-1.208	0.777
Weight	-0.205	-1.607	0.750
Acceleration	-0.015	-2.733	0.538
Horsepower	-0.285	-1.519	0.559

Table 4.2 contains the descriptive statistics of the standardised attributes. Note that all of the attributes are positive skew. This is expected as most motor vehicles contain small engines and there are only a few with large engines. This is considered when implementing the model. The skewness might have an effect on the estimation procedure or indicate that the procedure is not affected by skewed data.

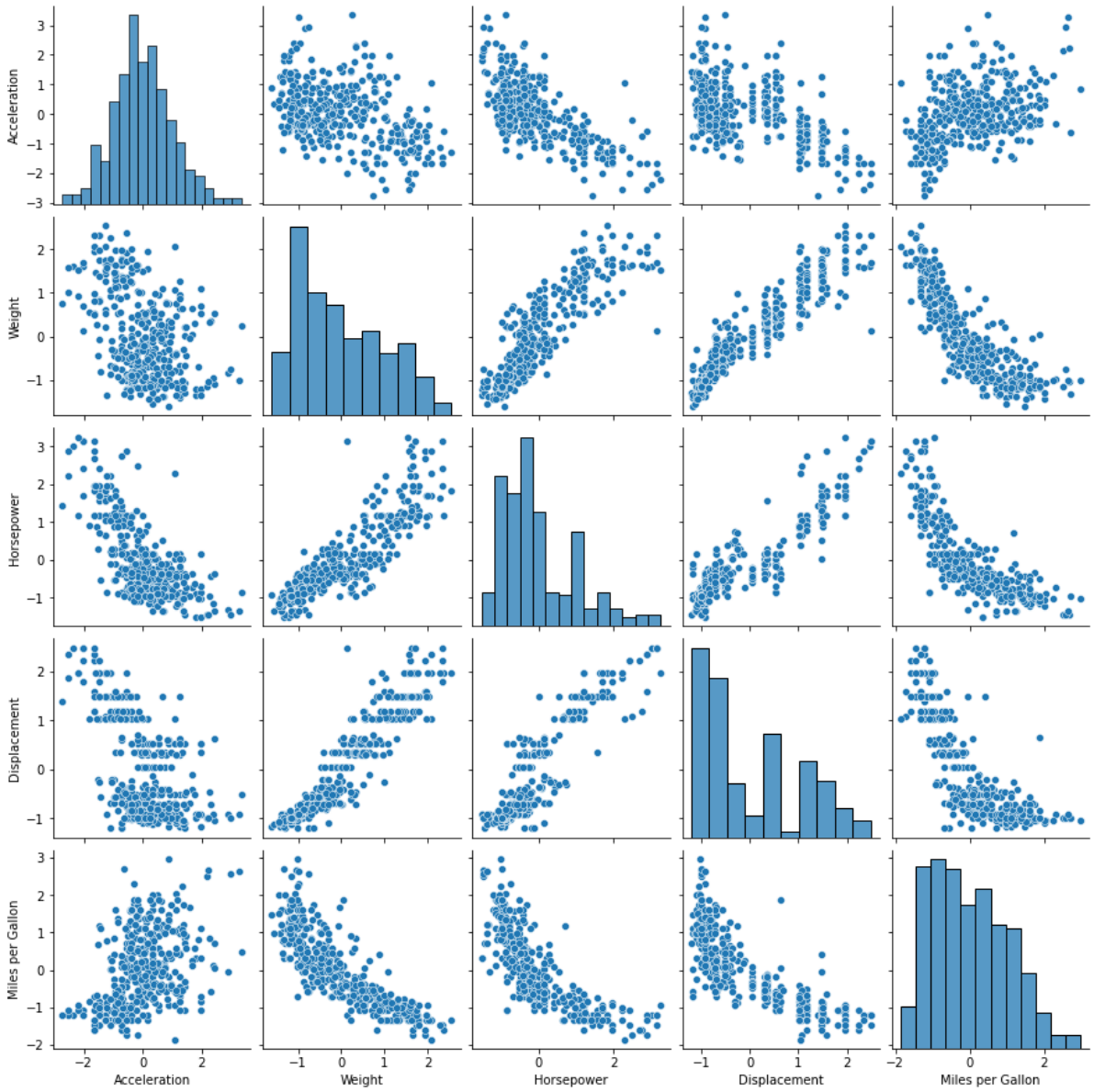


Figure 4.1: Pair plot of the standardised attributes

4.3 Exploratory analysis

Figure 4.1 shows the scatter plot relationships between the standardised variables. In order to fit the mixtures of PLMs model it is required that there is a linear relationship in the parameters between one or more covariates and the response variable, as well as a non-parametric relationship between a single covariate and the response variable. This leads to using *Horsepower* as the response variable, $Y = Horsepower$.

In Figure 4.2, it is clear that there is a linear relationship between *Horsepower* and *Weight*. Figure 4.2 also illustrates the skewness of the data. The majority of the data points are located in the bottom left quadrant of the scatter plot. Due to the linear nature it is decided *Weight* can be used as a linear covariate to estimate the response, *Horsepower*, that is $X_1 = Weight$. However, from this figure, it is not yet clear if there are different components in the model.

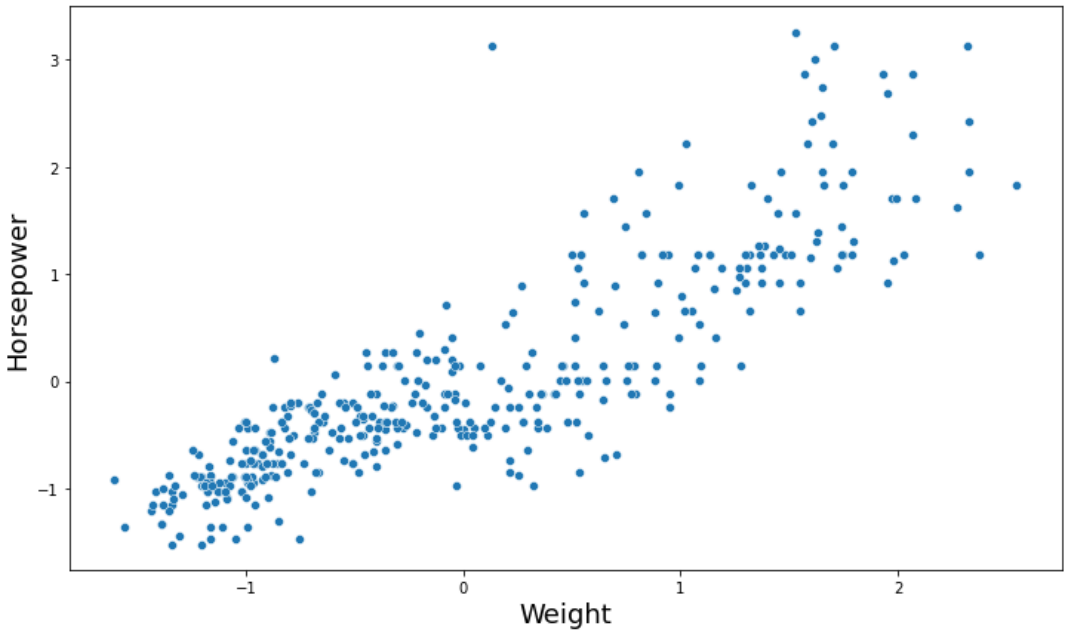


Figure 4.2: Scatter plot of Horsepower versus Weight.

The skewness in the data is even more evident when comparing *Horsepower* and *Displacement* in Figure 4.3. Here even more of the data are contained in the bottom left quadrant. Also, note that the scatter plot indicates a straight line relationship between *Horsepower* and *Displacement*. Again, the linearity leads to using *Displacement* as a linear covariate in the model, that is $X_2 = Displacement$.

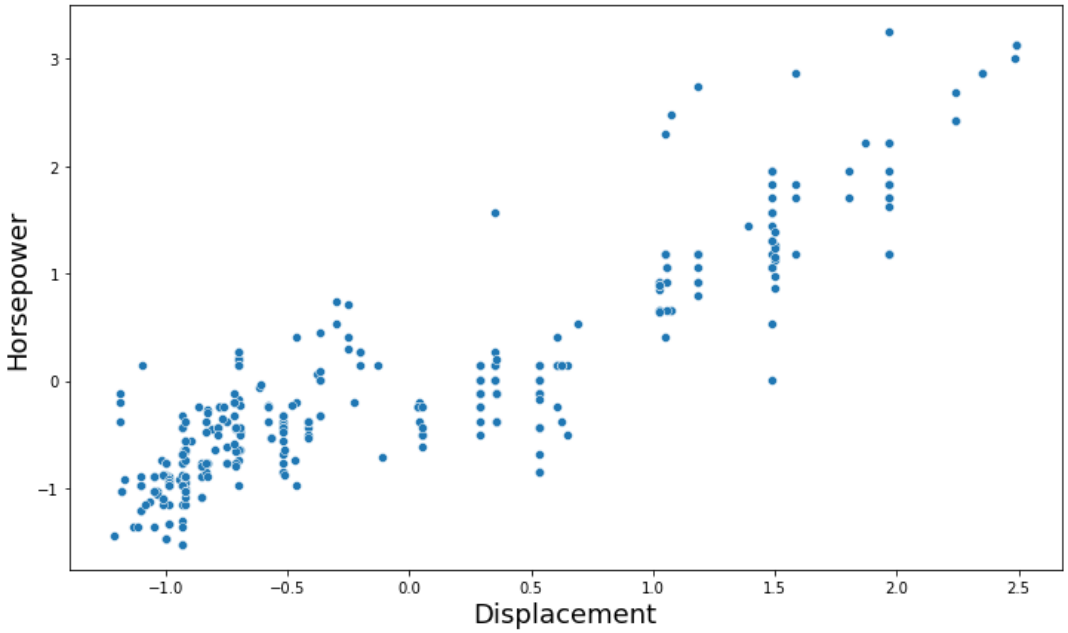


Figure 4.3: Scatter plot of Horsepower versus Displacement.

After evaluating the scatter plot of *Horsepower* versus *Miles per Gallon*, see Figure 4.4, it is clear that the relationship appear to be non-parametric. This leads to using *Miles per Gallon* as the non-parametric covariate in the model, that is $U = Miles\ per\ Gallon$. Figure 4.4 does not appear to be as skew as the other two components in the model. This could possibly impact the estimation procedure by creating a more constant estimate for the non-parametric covariate.

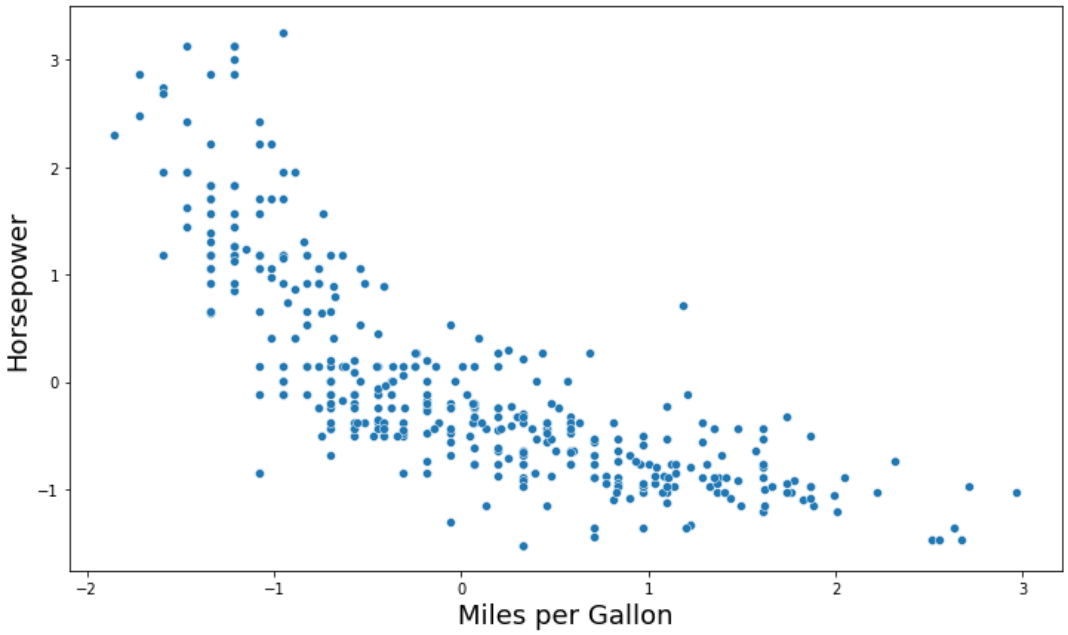


Figure 4.4: Scatter plot of Horsepower versus Miles per Gallon.

4.4 Model fit and results

Firstly, the number of components to be used in the model needs to be determined. Since this depends on the bandwidth chosen in the model it is decided that for simplicity this mini-dissertation will assume $h = 0.8$ when determining the number of components. The results indicated the following: for $K = 1$ the BIC $= -773.489$, for $K = 2$ the BIC $= -673.414$, for $K = 3$ the BIC $= -698.433$, for $K = 4$ the BIC $= -723.452$ and for $K = 5$ the BIC $= -748.47$. This concludes that the the number of components in the model is two, $K = 2$. Therefore, the following mixtures of PLM model is obtained

$$Y|_{X_1=x_1,X_2=x_2,U=u} \sim \sum_{k=1}^{2} \pi_2 N\{\beta_{k1}x_1 + \beta_{k2}x_2 + g_k(u), \sigma_k^2\}.$$

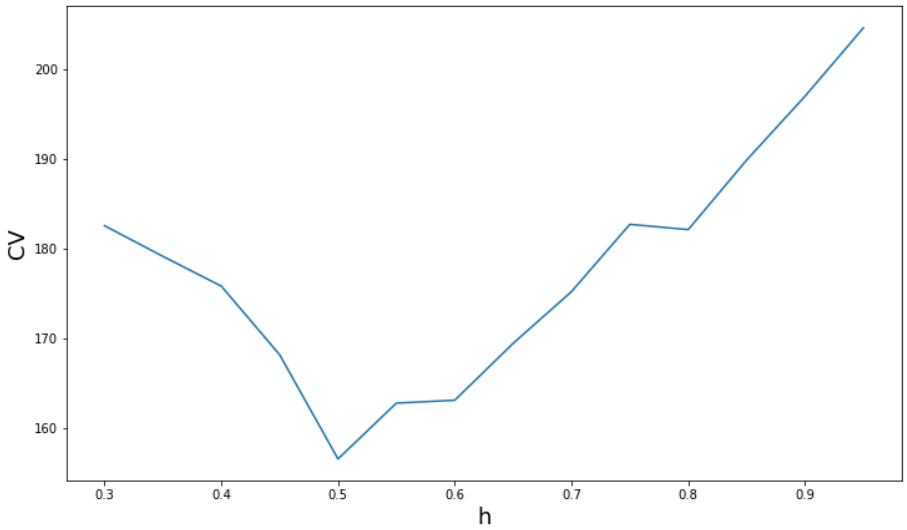
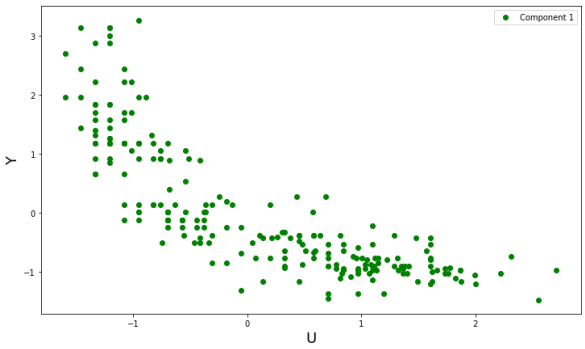
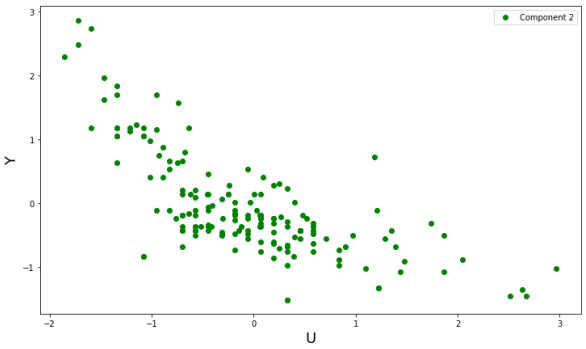


Figure 4.5: Plot showing the value of the CV value for different bandwidths

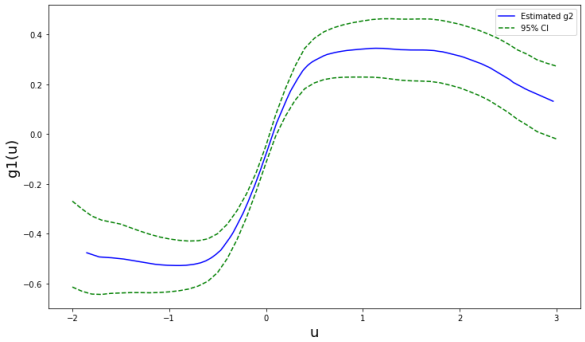
The optimal bandwidth can be found by applying the CV method discussed in Chapter 2, Section 2.5. The results can be seen in Figure 4.5. Here it is evident that the optimal bandwidth for the model is $\hat{h} = 0.5$. Applying the PL-EM algorithm to the data, the following results were obtained: the parameter estimates under the optimal bandwidth is $\hat{\beta}_{11} = 1.48$, $\hat{\beta}_{12} = 0.03$, $\hat{\beta}_{21} = 0.04$, and $\hat{\beta}_{22} = 0.46$. Also, $\hat{\sigma}_1^2 = 0.05$, $\hat{\sigma}_2^2 = 0.15$, $\hat{\pi}_1 = 0.53$, and $\hat{\pi}_2 = 0.47$. Comparing the estimated parameters of the two components it is evident that the two components are vastly different in terms of the linear covariates. The results are illustrated using a hard classification method, meaning that if an observation has a belonging greater than 0.5 in component one, then it is classified to be from component one, the same for component two. This is done to illustrate the effectiveness of the estimation procedure.



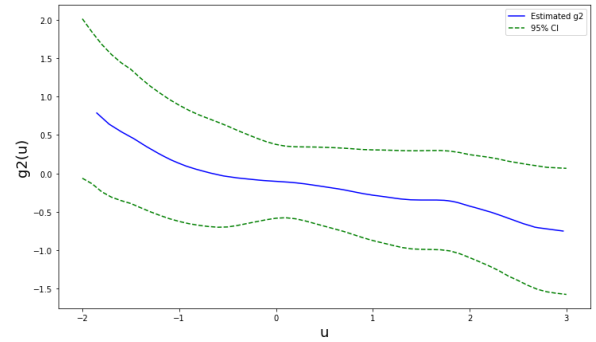
(a) Scatter plot of Y versus U of the observations hard classified into component 1.



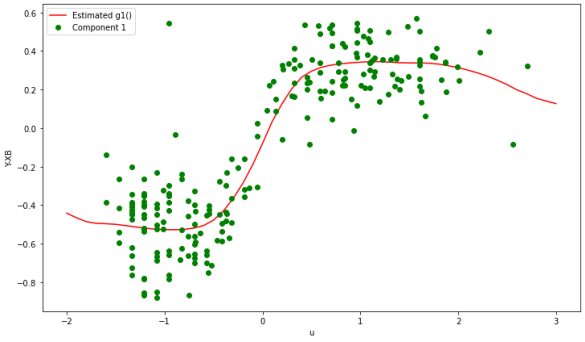
(b) Scatter plot of Y versus U of the observations hard classified into component 2.



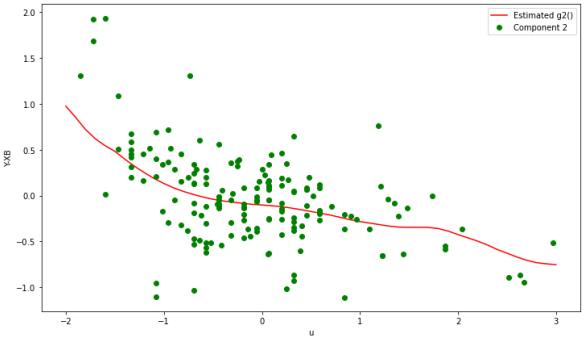
(c) The estimated non-parametric function values, $g_1(u)$, and a 95% confidence interval for component 1.



(d) The estimated non-parametric function values, $g_2(u)$, and a 95% confidence interval for component 2.



(e) The estimated non-parametric function values and the response variable without the estimated linear results for component 1.



(f) The estimated non-parametric function values and the response variable without the estimated linear results for component 2.

Figure 4.6: Results for the non-parametric part of the model estimation.

The estimated non-parametric part of the model is found in Figure 4.6. The scatter plots of the observations assigned to the two components, Figures 4.6a and 4.6b, show that component one uses less miles per gallon than component two. This is seen as there are more values clustered in the range where $0.5 < u < 2$ for component one, where the horsepower is low, than for the second component where the cluster is found between $-1 < u < 0.5$, where the horsepower is higher. This indicates that one would expect component one's engines to be smaller than those of the second component.

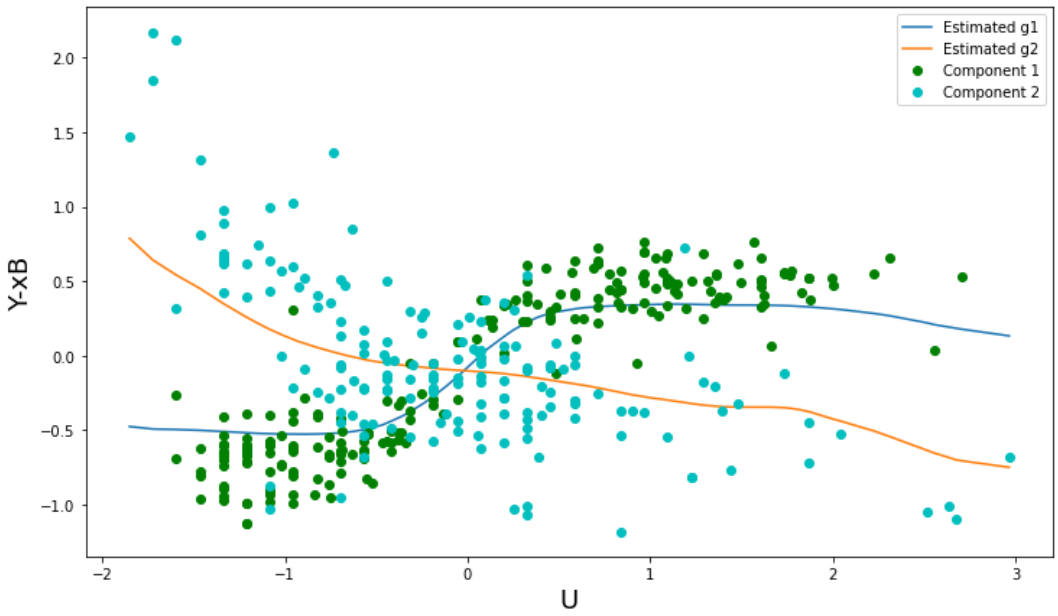


Figure 4.7: The estimated non-parametric function values and the response variable without the estimated linear results for both components.

Figures 4.6c and 4.6d show the estimates of the two functions as well as the 95% confidence intervals, obtained by applying a bootstrap method. Similar to the simulation study in Chapter 3 the confidence interval is wider at the edges. Once again this is the nature of using kernel estimation to determine unknown functions. Figure 4.6c shows a very small confidence interval at the area where $u = 0$. This is due to the lack of

observations assigned to component one in that area. However, even though component two has more observations around this point, the confidence interval is also smaller compared to the rest of the interval.

Similarly to the simulation study, again it is expected that the non-parametric covariate should be equal to the response variable without the linear component. These results are found in Figures 4.6e and 4.6f. Both components show the accuracy of the model quite well. From Figure 4.6e it is clear to see why the confidence interval decreases that significantly at $u = 0$ as there is almost no observations there. There is similar occurrence with component two, only at the edges. This explains why component two's confidence interval does not increase as significantly at the two edges than expected. Therefore, from the results it is clear that the PL-EM estimates the non-parametric part of the model effectively and that for the given data the results work sufficiently well as seen in Figure 4.7.

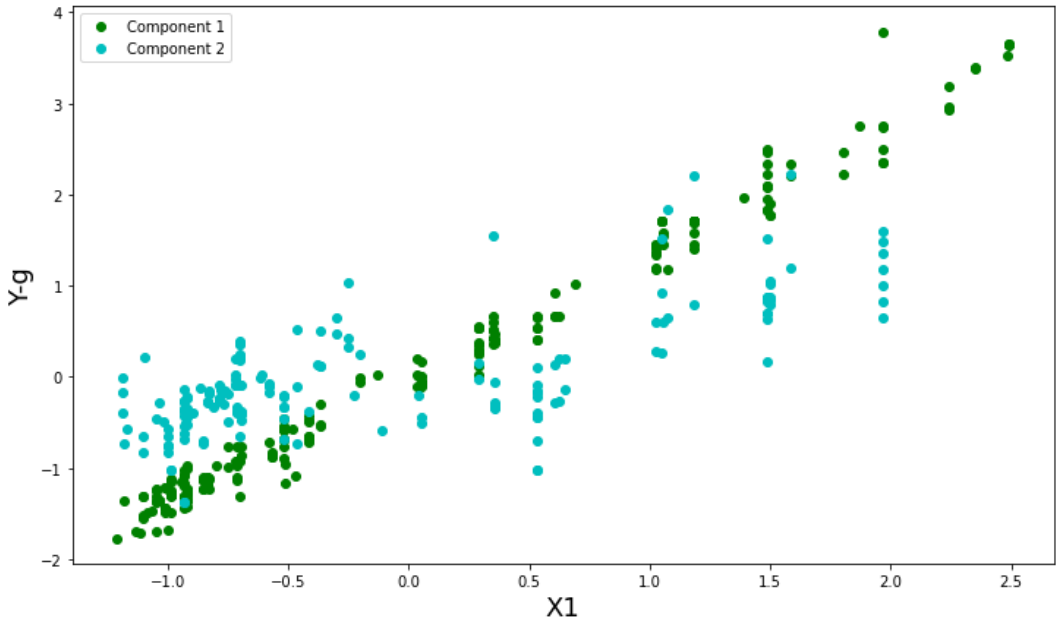


Figure 4.8: The estimated observations of X_1 and the response variable without the estimated non-parametric results for both components.

Figures 4.8 and 4.9 show the component of the linear covariate in relationship to the response without the non-parametric covariate. The results for X_1 indicates that the relationship between *Horsepower* and *Displacement* is quite strong for the first component. The second component shows a linear nature. However, the results for *Weight*, X_2, is not that clear. This is due to the impact the first covariate has on the response variable. There is a clear linear relationship in both components with regards to X_2 without the non-parametric part of the model.

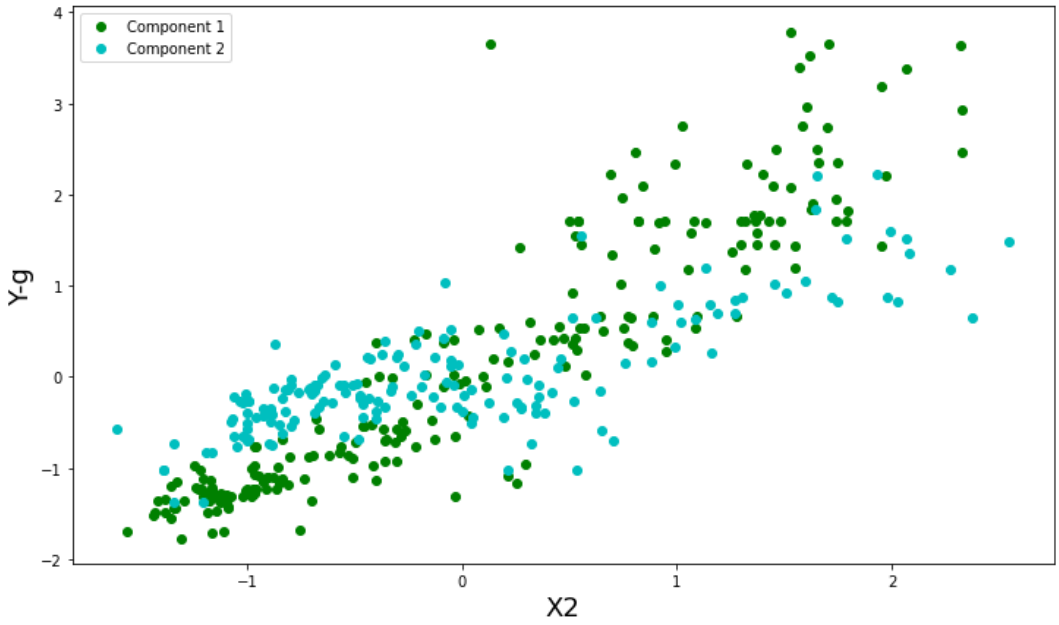


Figure 4.9: The estimated observations of X_2 and the response variable without the estimated non-parametric results for both components.

Considering the full results it is clear that the model fits the data quite well and that the estimation procedure is adequate in determining the parameter estimates. There is an evident non-parametric relationship between *Horsepower* and *Miles per Gallon*. There are also linear covariates, *Displacement* and *Weight*, that are clearly linearly related to *Horsepower*. These show that the semi-parametric mixtures of PLMs fit the data sufficiently well and that the estimation procedure, PL-EM algorithm, is adequate to obtain the parameter estimates and determine the g functions for each component. The procedure is also not affected by the skewness in the data.

4.5 Summary

In this chapter a real-world application of the mixtures of PLMs by applying the model to a Miles Per Gallon dataset obtained from Dua and Graff (2017) was considered. The description of the data, given in Section 4.2, indicated that not all of the features will be used in the fitting of the model and provided insight on the skewness of the data. Section 4.2's exploration showed that the model should be using *Horsepower* as the response variable. After fitting the model to the data in Section 4.4 it is clear to see that the model fits the data sufficiently and that the estimation procedure, PL-EM algorithm, is adequate to obtain the parameter estimates and determine the g functions for each component. The next chapter will conclude the mini-dissertation.

Chapter 5

Conclusions

This chapter provides a brief summary of the exploration of mixtures of partially linear models. Section 5.1 gives an overview of the work done in this research and Section 5.2 indicates any future developments that can be implemented to improve the model.

5.1 Concluding summary

There are a few objectives this mini-dissertation needed to address. The first is done by defining the mixtures of PLMs, proposed by Wu and Liu (2017). This model combines the PLM and the Gaussian mixture regression model by considering a PLM for the expected value of the Gaussian mixture regression model. Hence, the model is able to model both heterogeneous data and semi-parametric data, simultaneously. The estimation procedure is derived to be the PL-EM algorithm, which is a modified EM-algorithm that incorporates the concepts from the EM algorithm, local kernel regression, and a profile likelihood function. The algorithm provides a method of estimating the regression component parameters, non-parametric functions, and mixing coefficients. Parameters such as bandwidth and number of components are estimated by employing the BIC and

CV_{crit}, respectfully.

The mixtures of PLMs and the PL-EM algorithm are illustrated using a Monte Carlo simulation. Scatter plots of the simulated data are obtained to assist with determining the variables needed to implement the mixtures of PLMs to an arbitrary dataset. The number of components are determined using the BIC and the effectiveness thereof is illustrated by fitting the model using various bandwidths. The results show that using the BIC to determine the number of components works reasonably well. The effectiveness of using the CV method to determine the optimal bandwidth is discussed and the results indicate that using the CV method to determine the bandwidth will provide an optimal bandwidth that minimises the RASE. A simulation study is also used to compute the results of using the PL-EM algorithm as the estimation procedure.

A real-world application of the mixtures of PLMs by applying the model to a Miles Per Gallon dataset obtained from Dua and Graff (2017) is provided to facilitate the implementation of the proposed method. The results show that the estimation procedure works well in determining the estimates of the mixtures of PLMs. The estimation procedure can be used in many different ways. It is recommended to use it for estimating the mixture of PLMs model or the normal Gaussian mixture model. One can also use it for the single component PLM. The procedure is fairly adaptable and should be a single algorithm used, instead of various different ones. Care should be taken since the EM algorithm could converge to a local maxima instead of the global maxima.

5.2 Future work

This mini-dissertation considered the mixtures of PLMs and demonstrate the properties of the PL-EM algorithm. The algorithm is implemented on simulated data and real-world data to illustrate the adequacy of the estimation procedure. Considering this

mini-dissertation interesting future work could include:

- implementing a more comprehensive study using a wider range of sample sizes and determining the effect this will have on the MSE of the linear coefficients. This mini-dissertation considers moderate sample sizes of 200 and 500. Further investigation must be conducted on the effectiveness of the estimation procedure using even smaller sample sizes. Different ratios of the optimal bandwidth can also be implemented to determine the effectiveness of the PL-EM algorithm.

- considering other accuracy measures when the errors of the model are not Gaussian errors, for example, using the mean absolute error instead of the RASE for the non-parametric function.

- investigating the effect of co-linearity on the estimation procedure, the CV method, and the BIC method.

Bibliography

G. Boente, X. He, and J. Zhou. Robust estimates in generalized partially linear models. *The Annals of Statistics*, 34(6):2856–2878, 2006.

H. Chen, J. Chen, and J. D. Kalbfleisch. A modified likelihood ratio test for homogeneity in finite mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(1):19–29, 2001.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

R. F. Engle, C. W. Granger, J. Rice, and A. Weiss. Semiparametric estimates of the relation between weather and electricity sales. *Journal of the American statistical Association*, 81(394):310–320, 1986.

J. Fan, C. Zhang, and J. Zhang. Generalized likelihood ratio statistics and Wilks phenomenon. *The Annals of statistics*, 29(1):153–193, 2001.

S. Frühwirth-Schnatter. *Finite mixture and Markov switching models*, volume 425. Springer, 2006.

S. M. Goldfeld and R. E. Quandt. A Markov model for switching regressions. *Journal of econometrics*, 1(1):3–15, 1973.

P. J. Green and S. Richardson. Hidden Markov models and disease mapping. *Journal of the American statistical association*, 97(460):1055–1070, 2002.

N. E. Heckman. Spline smoothing in a partly linear model. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(2):244–248, 1986.

M. Huang, R. Li, and S. Wang. Nonparametric mixture of regression models. *Journal of the American Statistical Association*, 108(503):929–941, 2013.

M. Janka and T. W. Guenther. Management control of new product development and perceived environmental uncertainty: Exploring heterogeneity using a finite mixture approach. *Journal of Management Accounting Research*, 30(2):131–161, 2018.

G. Mallya, S. Tripathi, and R. S. Govindaraju. Probabilistic drought classification using gamma mixture models. *Journal of Hydrology*, 100(526):116–126, 2015.

H. Mamitsuka, C. DeLisi, and M. Kanehisa. *Data mining for systems biology: methods and protocols*. Springer, 2018.

G. J. McLachlan and S. Rathnayake. On the number of components in a Gaussian mixture model. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(5):341–355, 2014.

G. J. McLachlan, S. X. Lee, and S. I. Rathnayake. Finite mixture models. *Annual review of statistics and its application*, 6(1):355–378, 2019.

D. Peel and G. MacLahlan. *Finite Mixture Models*. John & Sons, 2000.

P. M. Robinson. Root-N-consistent semiparametric regression. *Econometrica: Journal of the Econometric Society*, 56(4):931–954, 1988.

G. Schwarz. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

T. A. Severini and W. H. Wong. Profile likelihood and conditionally parametric models. *The Annals of Statistics*, 20(4):1768–1802, 1992.

P. Wang, M. L. Puterman, I. Cockburn, and N. Le. Mixed Poisson regression models with covariate dependent rates. *Biometrics*, 52(2):381–400, 1996.

S. Wang, M. Huang, X. Wu, W. Yao, et al. Mixture of functional linear models and its application to CO2-GDP functional data. *Computational Statistics & Data Analysis*, 100(97):1–15, 2016.

X. Wu and T. Liu. Estimation and testing for semiparametric mixtures of partially linear models. *Communications in Statistics-Theory and Methods*, 46(17):8690–8705, 2017.

Appendix A

Code

A.1 Generate data

```
import numpy as np
from numpy.random import uniform

def gen_data(N):
    X = uniform(size=(N, 1))
    U = uniform(size=(N, 1))
    return X[:,0], U[:,0]
sig1 = 0.3
sig2 = 0.2
# 200
n = 200
X200 = np.zeros((n,500))
U200 = np.zeros((n,500))
Y200 =np.zeros((n,500))
```

```
for j in range(500):
    n = 200
    X200[:,j], U200[:,j] = gen_data(n)
    U1 = np.exp(2*U200[:,j] - 1)
    U2 = 2*np.cos(np.pi * U200[:,j])**2
    m1 = 5*X200[:,j] + U1
    m2 = -1*X200[:,j] + U2
    for i in range(n):
        p = uniform()
        if p < 0.65:
            Y200[i,j] = np.random.normal(m1[i],sig1)
        else:
            Y200[i,j] = np.random.normal(m2[i],sig2)

# 500
n = 500
X500 = np.zeros((n,500))
U500 = np.zeros((n,500))
Y500 =np.zeros((n,500))
for j in range(500):
    n = 500
    X500[:,j], U500[:,j] = gen_data(n)
    U1 = np.exp(2*U500[:,j] - 1)
    U2 = 2*np.cos(np.pi * U500[:,j])**2
    m1 = 5*X500[:,j] + U1
```

```
    m2 = -1*X500[:,j] + U2
    for i in range(n):
        p = uniform()
        if p < 0.65:
            Y500[i,j] = np.random.normal(m1[i],sig1)
        else:
            Y500[i,j] = np.random.normal(m2[i],sig2)


# 1000
n = 1000
X1000 = np.zeros((n,500))
U1000 = np.zeros((n,500))
Y1000 =np.zeros((n,500))
for j in range(500):
    n = 1000
    X1000[:,j], U1000[:,j] = gen_data(n)
    U1 = np.exp(2*U1000[:,j] - 1)
    U2 = 2*np.cos(np.pi * U1000[:,j])**2
    m1 = 5*X1000[:,j] + U1
    m2 = -1*X1000[:,j] + U2
    for i in range(n):
        p = uniform()
        if p < 0.65:
            Y1000[i,j] = np.random.normal(m1[i],sig1)
        else:
```

```
Y1000[i,j] = np.random.normal(m2[i],sig2)
```

A.2 Functions

```
from scipy.stats import norm
#import math
import numpy as np
np.seterr(invalid='ignore')
#import Generate_Data as GD
def kern(x):
    k = max(0.75*(1-x**2), 0)
    return k

def Kh1(x1,h):
    x = x1/h
    result = kern(x)/h
    return result



def get_r(n, C, X, B, G, Y, sig, pi):
    Sumi = np.zeros((n,C))
    r = np.zeros((n, C))
    for c in range(C):
        XB = np.matmul(X,B[:,c]) + G[:,c]
        Sumi[:,c] = pi[c]*norm.pdf(Y,loc=XB, scale= sig[c])
    S1 = np.sum(Sumi,axis=1)
```

```
    for c in range(C):
        r[:,c] = np.divide(Sumi[:,c],S1)
    return r


def get_S(r, n, C, KH):
    S = np.zeros((n,n,C))
    S_top = np.zeros((n,n))
    S_bot = np.zeros((n,1))
    for c in range(C):
        for i in range(n):
            S_top[i,:] = r[:,c]*KH[i,:]
        S_bot = np.sum(S_top, axis = 1)
        S[:,:,c] = np.divide(S_top.T, S_bot).T
        if S[:,:,c].any() > 1:
            print("Error")
    return S


def get_Beta(X, Y, W, C, B, S, n):
    Idn = np.identity(n)
    for i in range(C):
        IS = (Idn-S[:,:,i])
        X = np.matmul(IS, X)
        Y = np.matmul(IS, Y)
```

```
            XW = np.matmul(X.T,W[:,:,i])
            XWX = np.matmul(XW,X)
            XWY = np.matmul(XW,Y)

            try:
                B[:,i] = np.matmul(np.linalg.inv(XWX),(XWY))
                pass
            except:
                print("LinAlgError")
                pass

    return B

def get_sig(X, B, Y, G, C, n, r):
    sigma = np.zeros((C))
    for c in range(C):
        XB = np.matmul(X,B[:,c])
        YXB = Y[:]-XB[:]
        YXBG = YXB  - G[:,c]
        S2 = np.power(YXBG,2)
        S3 = r[:,c] * S2
        S4 = np.sum(S3)
        S5 = np.sum(r[:,c])
        sigma[c] = np.sqrt(S4/S5)
    return sigma
```

```
def Likelihood(Y, X, G, sigma, pi, B, C, n):
    L=0
    for i in range(n):
        inner = 0
        for c in range(C):
            XB = np.matmul(X[i],B[:,c]) + G[i,c]
            inner += pi[c]*norm.pdf(Y[i],loc=XB, scale= sigma[c])
        L += np.log(inner)
    return L

def EM1(Y, X, n, C, ite):
    pi = np.zeros((C))
    bnum = np.shape(X)
    B = np.zeros((bnum[1], C))
    sig = np.zeros((C))
    for i in range(C):
        pi[i] = 1/C
        sig[i] = 0.02 + C
        for l in range(bnum[1]):
            B1 = X[:,l].mean()
            B[l, i] = B1 +i

    G = np.zeros((n,C))
    S = np.zeros((n,n,C))
```

```
    W = np.zeros((n,n,C))
    Like = np.empty(())

    for j in range(ite):
        r = get_r(n, C, X, B, G, Y, sig, pi)
        for i in range(C):
            pi[i] = np.sum(r[:,i])/n
            W[:,:,i] = np.diag(r[:,i])
        B = get_Beta(X, Y, W, C, B, S, n)
        sig = get_sig(X, B, Y, G, C, n, r)
        L = Likelihood(Y, X, G, sig, pi, B, C, n)
        #print(L)
        if j > 0:
            if np.abs(L - Like[-1]) < 1e-5:
                Like = np.append(Like, L)
                return r, pi, B, sig

        Like = np.append(Like, L)
    return r, pi, B, sig

def isNaN(num):
    if float('-inf') < num < float('inf'):
        return False
    else:
        return True
```

```
def PL_EM(Y, X, n, C, ite, U, h):
    O = np.ones((n,1))
    if X.shape[1] == 1:
        X_new = np.stack((O[:,0],X[:,0]), axis=1)
    else:
        X_new = np.stack((O[:,0],X[:,0], X[:,1]), axis=1)
    del O
    tempr, pi, B, sigma = EM1(Y, X_new, n, C, ite)
    del tempr
    X = X.reshape(n,X.shape[1])
    #print(X.shape)
    G = np.zeros((n,C))
    for c in range(C):
        G[:,c] = B[0,c]
    B = np.delete(B,0,0)
    W = np.zeros((n,n,C))
    KH = np.zeros((n,n))
    for i in range(n):
        for j in range(n):
            temp = U[i]-U[j]
            KH[i,j] = Kh1(temp, h)
    #print("\nDone KH")
    Like = np.empty(())
    G2 = G
```

```
sigma2 = sigma
pi2 = pi
B2 = B
for j in range(ite):
    r = get_r(n, C, X, B, G, Y, sigma, pi)
    for i in range(C):
        pi[i] = np.sum(r[:,i])/n
        W[:,:,i] = np.diag(r[:,i])
    S = get_S(r, n, C, KH)
    B = get_Beta(X, Y, W, C, B, S, n)
    for c in range(C):
        YXB = (Y-np.matmul(X,B[:,c]))
        G[:,c] = np.matmul(S[:,:,c],YXB)
    sigma = get_sig(X, B, Y, G, C, n, r)
    L = Likelihood(Y, X, G, sigma, pi, B, C, n)

    for i in np.argsort(pi):
        G2[:,i] = G[:,i]
        sigma2[i] = sigma[i]
        pi2[i] = pi[i]
        B2[:,i] = B[:,i]

    G = G2
    sigma = sigma2
    pi = pi2
```

```
            B = B2


            if j > 0:
                if np.abs(L - Like[-1]) < 1e-10:
                    Like = np.append(Like, L)
                    return r, pi, B, sigma, G, KH, S, Like


            Like = np.append(Like, L)
        return r, pi, B, sigma, G, KH, S, Like

def get_G_CV(U_test, U_train, r, h, n_test, n_train, Y_train,
             X_train, B_train, C=2):
    X_train = X_train.reshape(n_train,2)
    G = np.zeros((n_test, C))
    KH = np.zeros((n_train, n_test))
    for i in range(n_train):
        for j in range(n_test):
            temp = U_train[i]-U_test[j]
            KH[i,j] = Kh1(temp, h)

    S = np.zeros((n_train,n_test,C))
    S_top = np.zeros((n_train,n_test))
    S_bot = np.zeros((n_test,1))
    for c in range(C):
```

```
        for i in range(n_test):
            S_top[:,i] = r[:,c]*KH[:,i]


        for i in range(n_test):
            S_bot[i] = np.sum(S_top[:,i])
        for i in range(n_test):
            S[:,i,c] = np.divide(S_top[:,i], S_bot[i])
    for c in range(C):
            YXB = (Y_train-np.matmul(X_train,B_train[:,c]))
            G[:,c] = np.matmul(S[:,:,c].transpose(),YXB)
    return G
```