2019

Adaptive Smoothing Parameter in Kernel Density Estimation and Parameter Estimation in Normal Mixture Distributions

Sabiha Mahzabeen

Minnesota State University, Mankato

Adaptive Smoothing Parameter in Kernel Density Estimation and Parameter Estimation in Normal Mixture Distributions

By

Sabiha Mahzabeen

A Thesis Submitted in Partial Fulfillment of the

Requirement for the degree of

Master of Science

in

Mathematics and Statistics

Minnesota State University, Mankato

Mankato, Minnesota.

May 2019

April 9, 2019

Adaptive Smoothing Parameter in Kernel Density Estimation and Parameter Estimation in Normal Mixture Distributions

Sabiha Mahzabeen

This thesis has been examined and approved by the following members of the student's committee:

_____

Dr. Mezbahur Rahman
(Supervisor)

_____

Dr. Han Wu
(Committee Member)

_____

Dr. Iresha Premarathna
(Committee Member)

Table of Contents

List of Figures

List of Tables

Abstract

Kernel density estimation is a widely used tool in nonparametric density estimation procedures. Choice of a kernel function and a smoothing parameter are two important issues in implementing kernel density estimation procedures. In this paper, four different kernel functions are considered in implementing an adaptive selection procedure in choosing the smoothing parameter. In simulation, a skewed bimodal density which is a mixture of two normal distributions is considered along with the standard normal and the standard exponential densities.

In skewed bimodal data, parameter estimation is also explored in the context of the parameter estimation in mixtures of normal distributions. Maximum likelihood estimation procedure is implemented in parameter estimation in mixtures of normal distributions.

Chapter 1: Introduction

In this paper I worked on approximation of the mean integrated squared errors for different kernels in density estimation to the best smoothing parameter. As stated in Silverman (1986), density estimation has become an integral part of nonparametric functional estimation procedures in statistics, where a density function is found from the observed data. One of the benefits with density estimation, is that it allows one to take a closer look at the data's properties. In Silverman (1986) and Hart (1997) many different methods for density estimation such as histograms, naive estimator, nearest neighbor method, orthogonal series estimator, and kernel density estimators are shown. In this paper the method that will be examined is the kernel density estimation method. The kernel method is based off a kernel, $K(u)$, which is a symmetric density function. This method is a way of using a weighted average based on the importance of the observed data and its closeness to the estimated point to give a better estimate of the observed data's properties, as shown in Hart (1997). The effectiveness of kernel method can be linked to the choice of smoothing parameter, bandwidth, or window width h, depending on the literature that is used. A smoothing parameter can be very large or small depending on the observed data. The best smoothing parameter is an h value that will minimize the estimation error associated with the data and the true distribution from which the data is derived. According to Silverman (1986), in terms of $\widehat{f(x)}$ as an estimator of $f(x)$ the mean integrated square error, MISE, is the most

popular method of finding the most accurate estimator, so minimizing the error with respect to h will give the best smoothing parameter. In this paper, it will not only find the best choice of h from the MISE, but h_a, which will be the smoothing parameter estimate for minimizing the approximate MISE, AMISE.

Chapter 2: Background

For estimating density function $\hat{f}$, we need some basic definitions such as what is probability density function, what is density estimation and why we are using nonparametric method instead of parametric.

2.1: Probability Density Function

Probability density function is a fundamental concept in Statistics. Consider any random quantity X that has probability density function f. Specifying the function f gives a natural description of the distribution of X, allows probabilities associated with X to be found from the relation

$$P(a < X < b) = \int_a^b f(x)dx \quad for\ all\ a < b$$

Suppose we have a set of observed data points assumed to be a sample from an unknown probability density function.

Mathematical Expression:

$$f(X_0) = dF(X_0)/dx$$

$$f(X_0) = \lim_{h \to 0} \frac{F(X_0 + h) - F(X_0 - h)}{2h}$$

$$f(X_0) = \lim_{h \to 0} \frac{1}{2h} P(X_0 - h < X < X_0 + h)$$

2.2: Density Estimation

Density estimation is the construction of an estimate of the density function from the observed data. Basically, we are interested in determining an unknown function f, given only random samples or observations distributed according to this function. More formally, the goal of density estimation is to infer the probability density function, or PDF, from observations of a random variable.

2.3: Parametric and Non parametric method

We have two different types of methods for density estimation. One approach to density estimation is parametric. Assume that the data are drawn from one of a known parametric family of distributions for example the normal distribution with mean μ and σ^2. The density f, underlying the data could then be estimated by finding estimates of μ and σ^2. From the data and substituting these estimates into the formula for the normal density.

Another approach to density estimation is nonparametric. Nonparametric statistics are not based on parameterized families of probability distribution. Nonparametric statistics make no assumptions about the probability distributions of the variables being assessed. Nonparametric models are extremely useful to get from discrete data to probability to density functions or distributions.

Chapter 3: Nonparametric Methods to Estimate Density Estimation

when a dataset is available, the distribution in which the data set is derived from is a key question. In parametric estimation, a distribution is postulated and then the parameters are estimated, but if the postulated distribution is not right, then the analysis became questionable. But in nonparametric density estimation, no distributional assumptions are made. Hence it became a popular tool in density estimation. There are several nonparametric density estimation procedures. Some of the procedures are given below:

1. Histogram
2. Naïve Estimators
3. Kernel Density Estimation
4. Orthogonal series methods
5. Nearest neighbor methods
6. Variable Kernel Methods
7. Maximum Penalized Likelihood estimators
8. General Weights function estimators
9. Spline Methods.

3.1 Histogram

The oldest and most widely used density estimator is the histogram. Histogram is a discrete approximation. Given an origin x_0 and a bin width h we define the bins of the histogram to be the intervals $[x_0 + mh,\ x_0 + (m+1)h)$ for positive and negative integers m. The intervals have been chosen closed on the left open on the right for definiteness.

The histogram is then defined by

$$\hat{f}^{hist}(x) = \frac{1}{nh}(no.of\ X_i\ in\ same\ bin\ as\ x)$$

$$= \frac{1}{n} \times \frac{\#\ of\ X_i\ in\ same\ bin\ as\ x}{Width\ of\ bin\ containing\ x}$$

Note that, to construct the histogram, we must choose both an origin and a bin width. It is the choice of bin width which, primarily, controls the amount of smoothing inherent in the procedure.

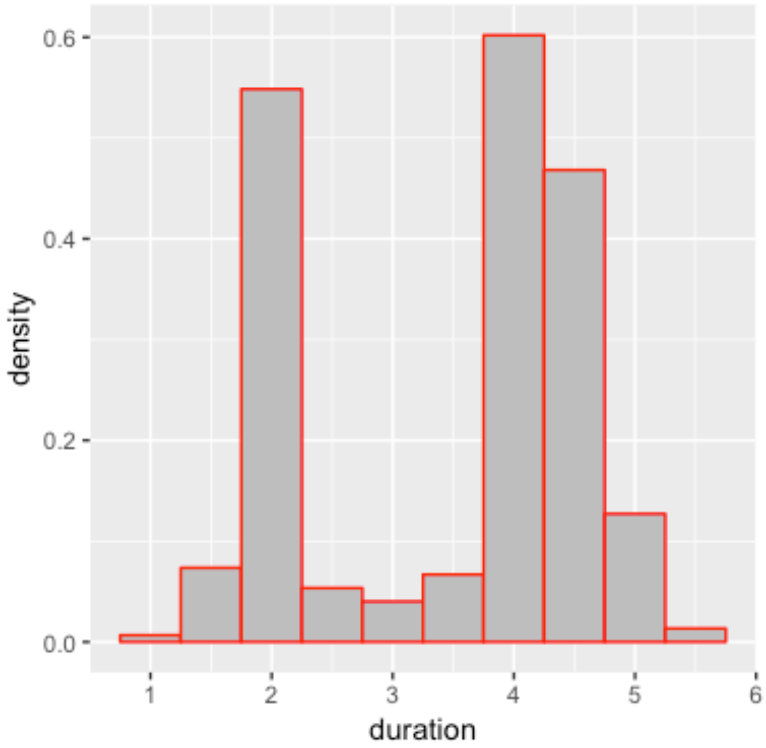


Figure 3.1 (a) Histogram with bin width 0.5

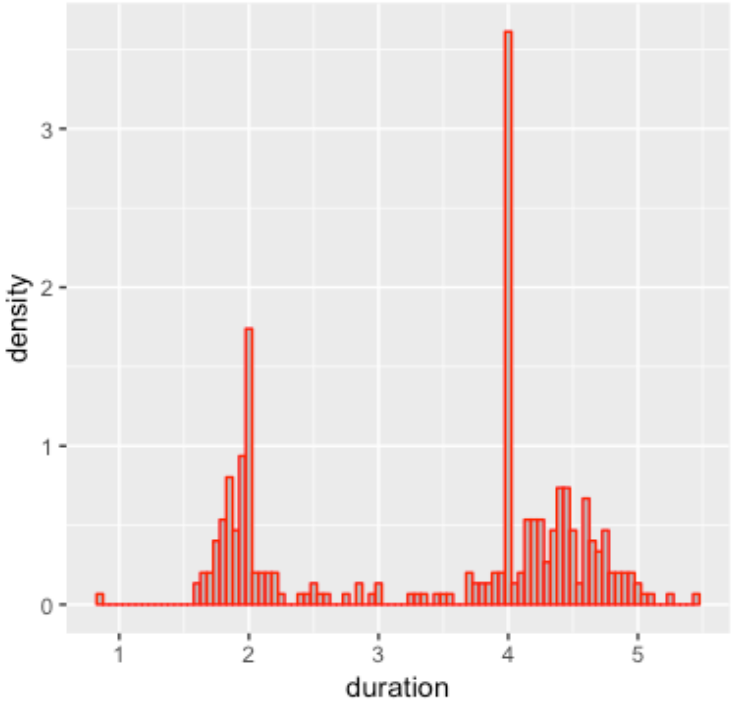


Figure 3.1 (b) Histogram with bin width 0.05

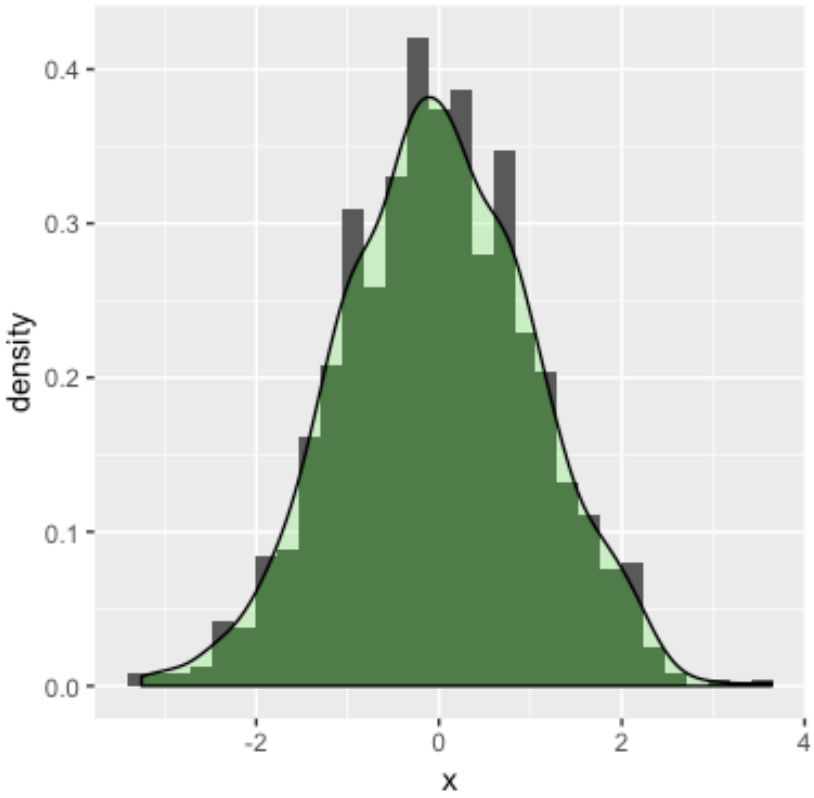


Figure 3.1 (c) Histogram with curve

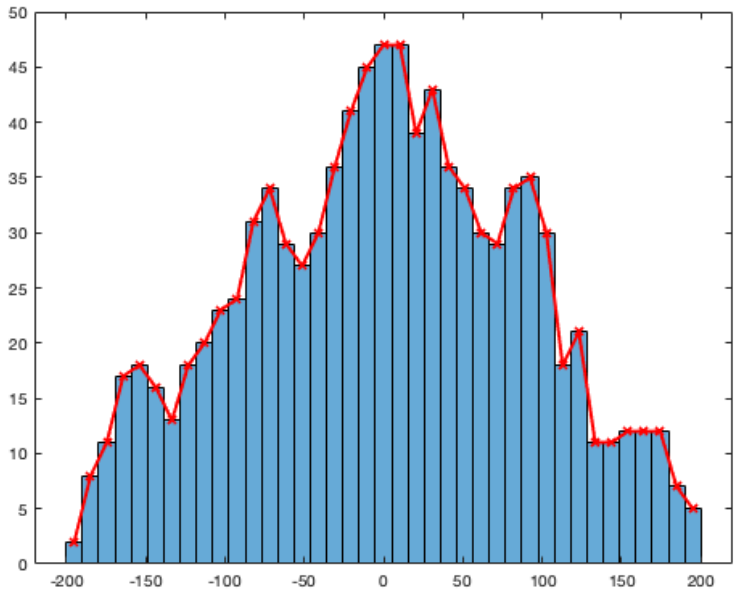


Figure 3.2: Center value of each bin with density

The histogram method has numbers of advantages. It is easy to implement and provides results which are straightforward to visualize and intuitive to interpret particularly in univariate case. The density estimate depends on the starting position of the bins. For multivariate data, the density estimate is also affected by the orientation of the bins. The discontinuity of histograms causes extreme difficulty if derivatives of the estimates are required. However, histograms have many disadvantages, which motivated the developed of more advanced methods.

3.2: Naïve Estimator

The Naïve estimator was introduced by Fix and Hodges (1951) in an unpublished report. In (1986) Silverman calls the naïve estimator, addresses the choice of bin locations. From the definition of a probability density, if the random variable X has density f, then

$$f(x) = \lim_{h \to 0} \frac{1}{2h} P(x - h < X < x + h)$$

For any given h, we can of course estimate $P(x - h < X < x + h)$ by the proportion of the sample falling in the interval $(x - h, x + h)$ .Thus a natural estimator $\hat{f}$ of the density is given by choosing a small number h and setting

$$\hat{f}(x) = \frac{1}{2hn} [no. of\ X_{1,\ldots\ldots} X_n\ falling\ in\ \ (x - h, x + h)]$$

We call this the naïve estimator.

To express the estimator more transparently, define the weight function w by

$$w(x) = \begin{cases} \frac{1}{2} & if\ |x| < 0 \\ 0 & otherwise \end{cases} \qquad (3.1)$$

Then it is easy to see that the naïve estimator can be written as

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} w\left(\frac{x - X_i}{h}\right)$$

It follows from (3.1) that the estimate is constructed by placing a 'box' of width $2h$ and height $(2nh)^{-1}$ on each observation and then summing to obtain the estimate. This interpretation is useful in deriving the Kernel estimator, which we are going to discuss below.

3.3: Kernel Density Estimation

The univariate kernel density estimation (KDE) is a nonparametric way to estimate the probability density function $f(x)$ of a random variable X, is a fundamental data smoothing problem where inferences about the population are made, based on a finite data sample. Let $(X_1, X_2, \ldots\ldots\ldots, X_n)$ be a univariate independent and identically distributed sample drawn from some distribution

with an unknown density f. We are interested in estimating the shape of this function f.

Kernel Estimator: Kernel estimators can be viewed as a generalization of the naïve estimator. Replace w by a kernel function K which satisfies the condition

$$\int_{-\infty}^{\infty} K(x)dx = 1$$

Usually, but not always, K wil be a symmetric probability density function, the normal density, for instance, or the weight function w used in the definition of the naïve estimator. By analogy with the definition of the naïve estimator, the kernel estimator with kernel K is defined by

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)$$

where h is the window width, also called the smoothing parameter or bandwidth by some authors and $X_1, X_2, \ldots\ldots, X_n$ is the sample data and x is at which value the function is estimated. In section 4, Kernel Density Estimation procedure is elaborated as the focus of this paper.

As the naïve estimator can be considered as a sum of 'boxes' centered at the observations, the kernel function K determines the shape of the bumps while the window width h determines their width. In Fig 3.3, where the individual bumps $n^{-1}h^{-1}K\{(x - X_i/h)\}$ are shown as well as the estimate $\hat{f}$ constructed by adding them up.

The effect of varying the window width is shown in Fig 3.4. The limit as h tends to zero is a sum of Dirac delta function spikes at the observations, while as h becomes large all detail, spurious or otherwise obscured.

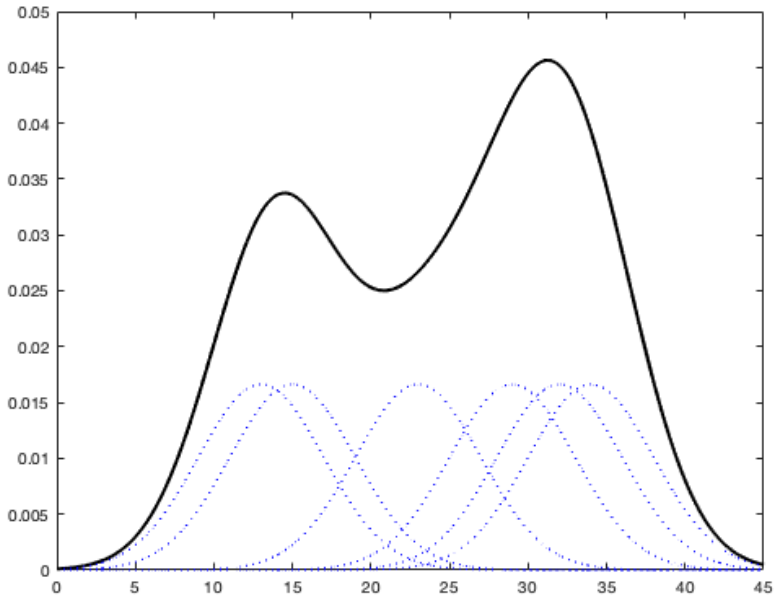


Figure 3.3: Kernel estimate showing individual kernels with bandwidth 4

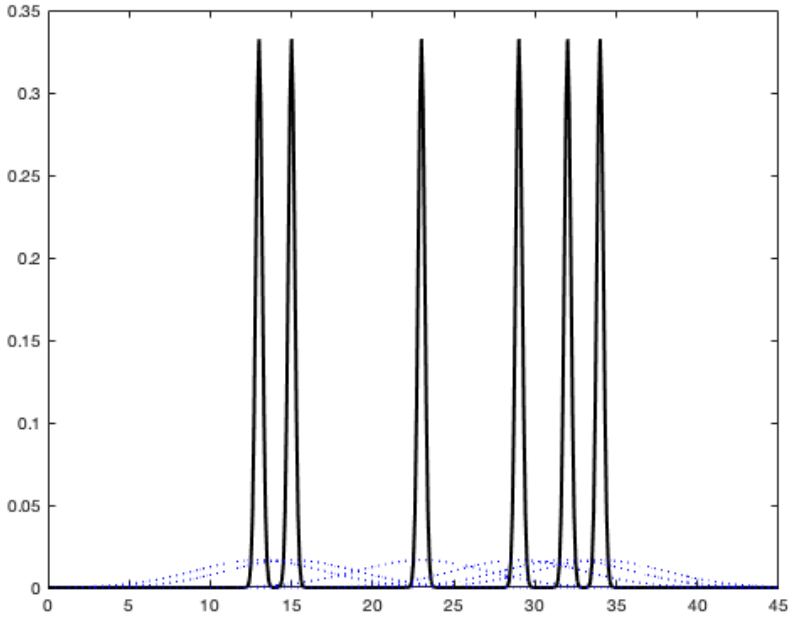


Figure 3.4(a): Kernel estimate showing individual kernels with bandwidth 0.2

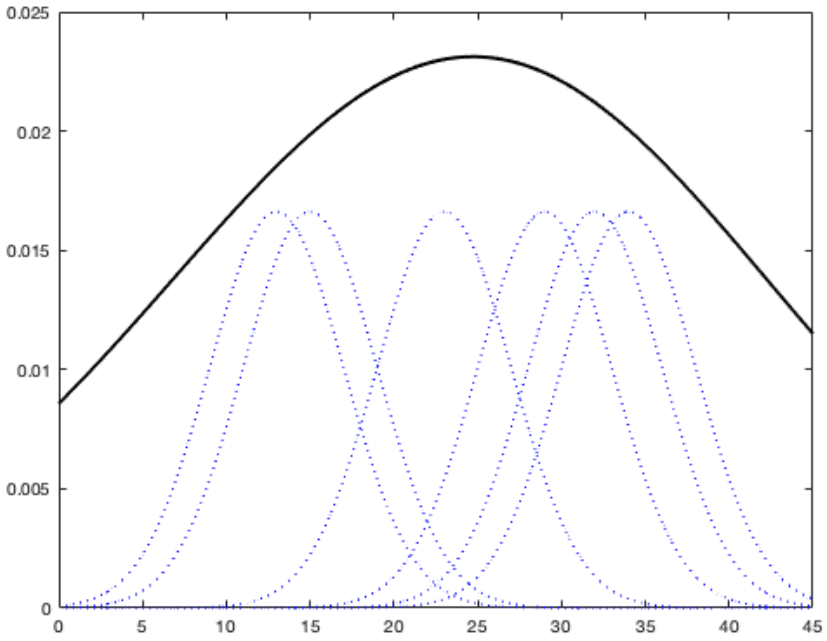


Figure 3.4(b): Kernel estimate showing individual kernels with bandwidth 15

Chapter 4: Kernel Density Estimation (KDE) Implementation

As stated in the introduction, kernel density function is a board field used extensively in Statistics. Throughout this paper $X_1, X_2, \ldots\ldots, X_n$ will be independent random sample approximated by kernel estimation in the form of a continuous density function, $\hat{f}(x)$. In this chapter, the focus will not only be the basis of kernel density estimation, the kernel, but also the importance of the smoothing parameters to the calculation of the density estimation.

The kernel estimation method is based off a kernel function, K(u), where $u = \frac{(x-X_i)}{h}$ and the kernel estimate of $f(x)$ can be expressed as the following,

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)$$

According to Hart (1997), a kernel function must satisfy the following equations which allow for the minimization of the mean integrated squared error.

$$\int K(u)du = 1 \qquad \int uK(u)du = 0$$

$$\int u^2K(u)du = k_2 \neq 0 < \infty \qquad \int K^2(u)du = I_2 < \infty$$

The first two equations make it so that the kernel must be a probability density function and have a mean of zero. Satisfying these two properties allows the function to be symmetric and a maximum at zero, which are ideal properties for estimating data that has an unknown distribution. The first two equations must be finite to allow for the calculation of the AMISE. B.W. Silverman (1986) states that the kernel functions must also be continuous and differentiable, so that $\hat{f}(x)$ can also take on these properties. These properties are satisfied by many density functions, therefore, there is a list of some of the most popular choices as expressed in Silverman (1986)

Here we will consider only Gaussian, Epanechnikov, Quartic, and Triweight kernels for the sake of time. The kernel functions and their key characteristics are given in Table 4.1. h_a indicates the optimized smoothing parameter for a minimized approximate integrated squared error (AMISE) that was explained in section 4.1.

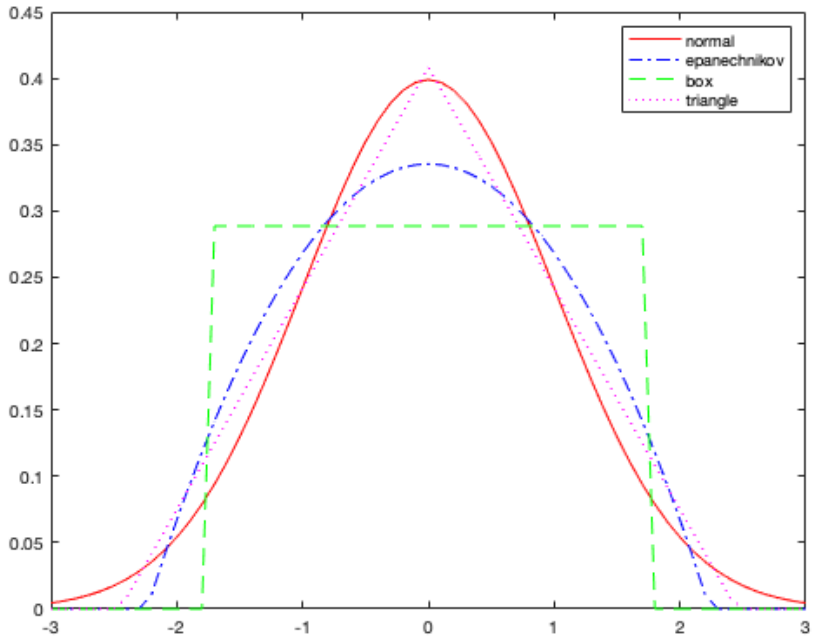


Figure 4: Different Kernel Functions

4.1: Smoothing Parameter or Bandwidth

A smoothing parameter can be very large or small depending on the observed data. The best parameter is an h value that will minimize the error associated with the data. According to Silverman (1986), in terms $\hat{f}(x)$ of as an estimator of $f(x)$ the mean integrated square error, MISE, is the most popular method of finding the most accurate estimator, so minimizing the error with respect to h will give the best smoothing parameter. In this paper optimum h denoting by h_a will be obtained by minimizing the approximate MISE, AMISE.

The effectiveness of the kernel estimation method comes down to the selection of the smoothing parameter. One must be very careful in selecting the ideal h value. If a smoothing parameter is chosen that is too large, this can cover up

the features of a distribution. However, if an h value is chosen that is too small, this can overemphasize the data variability.

Therefore, in most cases the more structure a graph has, the smaller the h value. Alternatively, the flatter the graph the larger the h value will be. The best choice of the smoothing parameter hinges on the sample size and the following three factors as described in Hart (1997):

1. The smoothness of the density function.
2. The distribution of the points of estimation.
3. The amount of variability among the data.

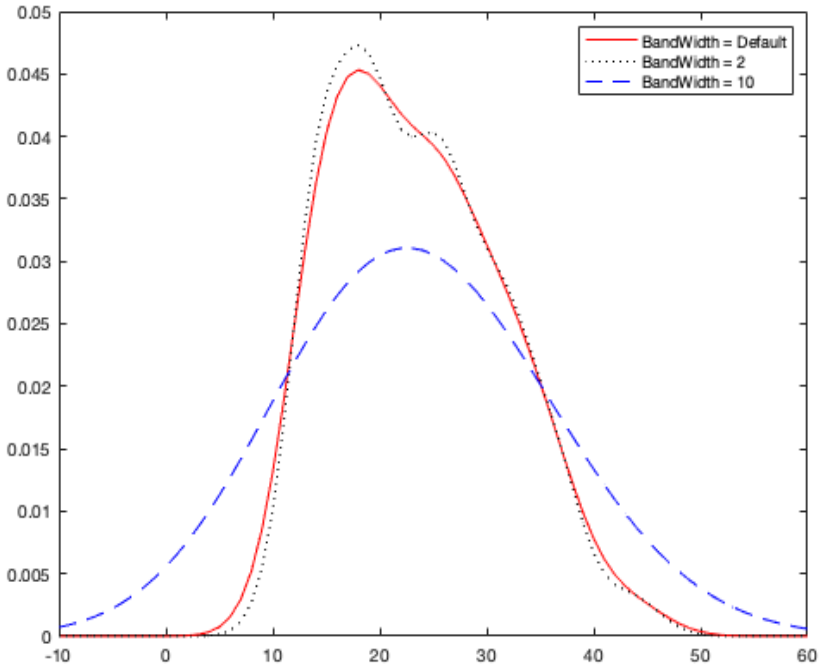


Figure 4.1: Kernel functions with different bandwidths

4.2: Bias and Asymptotically Unbiased Estimators

As stated in Silverman (1986), mean integrated squared error is the most popular used measure of accuracy of $\hat{f}(x)$ and can be expressed as following:

$$\text{MSE } (\hat{f}(x)) = E\{\hat{f}(x) - f(x)\}^2$$

We can write as $\text{MSE } (\hat{f}(x)) = E\{\hat{f}(x) - f(x)\}^2 + var\ \hat{f}(x)$

We want to optimize the mean integrated squared error (MISE) as

$$MISE = \int MSE\left(\hat{f}(x)\right) dx$$

$$MISE = E \int \{\hat{f}(x) - f(x)\}^2 dx + \int var\ \hat{f}(x)\ dx \qquad (4.2.1)$$

The bias (or a bias function) of an estimator is the difference between this estimator's expected value and the true value of the parameter being estimated.

An asymptotically unbiased estimators are operators whose bias goes to zero as the sample size goes to infinity. In other word if $\widehat{\theta_n}$ is an estimator of θ using a sample size n, then we say this estimator is asymptotically unbiased if $\lim_{n\to\infty} E[\widehat{\theta_n}\,|\theta]=\theta$.

Variance gives a measure of how the data distributes itself about the mean or expected value and also looks at all the data points determining distribution.

Considering $X_{1,}X_{2,\ldots\ldots\ldots,}X_n$ are iid sample from an unknown density function p. In here we will focus on given point x_0 and will analyze the quality of estimator $\widehat{p_n}(x_0)$.

$$E(\widehat{p_n}(x_0)) - p(x_0) = E\left(\frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{X_i - x_0}{h}\right)\right) - p(x_0)$$

$$= \frac{1}{h} E\left(K\left(\frac{X_i - x_0}{h}\right)\right) - p(x_0)$$

$$= \frac{1}{h} \int K\left(\frac{x - x_0}{h}\right) p(x)dx - p(x_0)$$

We do change the variable $y = \frac{x - x_0}{h}$ so that $dy = \frac{dx}{h}$ and after that we have

$$= \int K\left(\frac{x - x_0}{h}\right) p(x) \frac{dx}{h} - p(x_0)$$

$$= \int K(y)p(x_0 + hy)dy - p(x_0)$$

Now by Taylor expansion, when h is small,

$$p(x_0 + hy) = p(x_0) - hy.p'(x_0) + \frac{1}{2}h^2y^2p''(x_0) + o(h^2)$$

Note that $o(h^2)$ means that it is smaller order term compared to h^2 when $h \to 0$.

$E(\widehat{p_n}(x_0))$ - $p(x_0) = \int K(y)p(x_0 - hy)dy - \ p(x_0)$

$=\int K(y)[\ p(x_0) - hy.p'(x_0) + \frac{1}{2}h^2y^2p"(x_0) + o(h^2)]\ dy - p(x_0)$

$= p(x_0)\ \int K(y)dy - \ hp'(x_0) \int y\ K(y)dy + \frac{1}{2}h^2y^2p"(x_0) \int y^2K(y)$

$$dy + \ o(h^2) - \ p(x_0)$$

$= \ \frac{1}{2}h^2y^2p"(x_0)\ \int y^2K(y)dy + o(h^2)$

$= \frac{1}{2}h^2p"(x_0)\ \mu_k + \ o(h^2)$, where $\mu_k = \ \int y^2K(y)dy$

Now the bias of the KDE is

$$\text{bias } (\widehat{p_n}(x_0)) = \frac{1}{2}h^2p"(x_0)\mu_k + \ o(h^2) \qquad (4.2.2)$$

This means that when we have $h \to 0$, the bias is shrinking at a rate of $O(h^2)$ and from (4.2.2) we see the bias of KDE is caused by the curvature (second derivative) of the density function. Namely, the bias will be very large at a point where the density function curves a lot (e.g., a very peaked bump). This makes sense because for such a structure, KDE tends to smooth it too much, making the density function smoother (less curved) than it used to be.

For variance , $Var(\widehat{p_n}(x_0)) = Var\left(\frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{X_i - x_0}{h}\right)\right)$

$= \frac{1}{nh^2}\, Var\left(K\left(\frac{X_i - x_0}{h}\right)\right) \leq \frac{1}{nh^2}\, E\left(K^2\left(\frac{X_i - x_0}{h}\right)\right)$

$= \frac{1}{nh^2} \int K^2\left(\frac{x - x_0}{h}\right) p(x)dx$

$= \frac{1}{nh} \int K^2(y) p(x_0 + hy)dy$ Where, $y = \frac{x - x_0}{h}$ and $dy = \frac{dx}{h}$

$= \frac{1}{nh} \int K^2(y)[p(x_0) + hyp'(x_0) + o(h)]dy$

$= \frac{1}{nh}\left(p(x_0) \cdot \int K^2(y)dy + o\left(\frac{1}{nh}\right)\right.$

$= \frac{1}{nh}\, p(x_0)\, {\sigma_k}^2 + o\left(\frac{1}{nh}\right)$ Where, ${\sigma_k}^2 = \int K^2(y)dy.$ (4.2.3)

Therefore, the variance shrinks at rate $O\left(\frac{1}{nh}\right)$ when $n \to \infty$ and $h \to 0$. An interesting fact from the variance is that at point where the density value is large, the variance is also large.

4.3: The bias-variance trade-off

Since the smoothing parameter is chosen from a function of n, the sample size will affect the bias of the solution through the smoothing parameter. In Silverman (1986), it can be seen that by taking larger sample sizes, it will decrease the smoothing parameter, which will help reduce the bias and will adjust the weight function to obtain asymptotically unbiased estimates. Decreasing h increases the integrated variance, however increasing h will increase the bias. Therefore, in the selection of the smoothing parameter there is balance of systematic and random error that takes place. The method of minimizing the error that will be focused on in this paper is minimizing the approximate mean integrated square error.

4.4: Approximated MISE (AMISE)

As stated in Silverman (1986), mean integrated squared error is the most popular used measure of accuracy of $\hat{f}(x)$ and can be expressed as following:

$$\text{MISE}(\hat{f}(x)) = \int E\left\{\hat{f}(x) - f(x)\right\}^2 dx$$

It has been shown in Parzen (1962) that from the MISE the minimization of the AMISE is defined by:

$$\text{AMISE}\left(\hat{f}(x)\right) = \frac{1}{4}h^4 k_2^2 \int \left(f^{(2)}(x)\right)^2 dx + \frac{1}{nh}\int K^2(u)\, du \qquad (4.4.1)$$

From the AMISE it is crucial that the optimal value of h is found. It was also shown in Parzen (1962) that this value is:

$$h_a = k_2^{-\frac{2}{5}}\{\int K^2(u)\,du\}^{\frac{1}{5}}\left\{\int \left(f^{(2)}(x)\right)^2 dx\right\}^{-\frac{1}{5}} n^{-\frac{1}{5}} \qquad (4.4.2)$$

If the density function $f(x)$ of our observed data is known, these approximation equations may be solved by substitution. All components of this equation have already solved for except $\int \left(f^{(2)}(x)\right)^2 dx$. In most cases, the density function of the observed data will be unknown. Therefore $f^{(2)}(x)$ will also be unknown. In this case, an estimate is needed for $f^{(2)}(x)$ and as Rahman et al. (1996) stated, this will be defined as the following:

$$\hat{f}^{(2)}(x) = \frac{1}{nh}\sum_{i=1}^{n} K^{(2)}\left(\frac{x - X_i}{h}\right)$$

For this section, the Gaussian, Epanechnikov, Quartile and Triweight kernels and a known distribution will be used to find the AMISE and h_a. As an example, it will be shown how the AMISE and h_a can be found from the known distribution was $N(\mu, \sigma^2)$.

Since the distribution is known, the density function can be expressed as the following:

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-\mu)^2} \quad (4.4.3)$$

Taking the first derivative of equation (4.4.3)

$$f^{(1)}(x) = -\frac{1}{\sqrt{2\pi}\sigma^3}\ (x-\mu) e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \quad (4.4.4)$$

Similarly taking the second derivative of equation (4.4.4)

$$f^{(2)}(x) = -\frac{1}{\sqrt{2\pi}\sigma^5}\ e^{-\frac{1}{2\sigma^2}(x-\mu)^2}[(x-\mu)^2 - \sigma^2] \quad (4.4.5)$$

Then, the integration of this equation (4.4.5) squared will give us the following:

$$\int \left(f^{(2)}(x)\right)^2 dx = \frac{1}{\sqrt{\pi}}\frac{3}{8}\frac{1}{\sigma^5} \quad (4.4.6)$$

Now for the Gaussian Kernel the Adaptive smoothing parameter using (4.4.6),

Where, $k_2 = 1$ and $I_2 = \int K^2(u)du = \frac{1}{2\sqrt{\pi}}$.

$$h_a = k_2^{-\frac{2}{5}}\{\int K^2(u)\,du\}^{\frac{1}{5}}\left\{\int \left(f^{(2)}(x)\right)^2 dx\right\}^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = (1)^{-\frac{2}{5}}\left(\frac{1}{2\sqrt{\pi}}\right)^{\frac{1}{5}}\left(\frac{1}{\sqrt{\pi}}\frac{3}{8}\frac{1}{\sigma^5}\right)^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = \left(\frac{3n}{4}\right)^{-\frac{1}{5}}\ \sigma$$

For the Epanechnikov Kernel the Adaptive smoothing parameter using (4.4.6), Where,

$$k_2 = \frac{1}{2} \text{ and } I_2 = \int K^2(u)du = \frac{3}{5}$$

$$h_a = k_2^{-\frac{2}{5}}\{\int K^2(u)\,du\}^{\frac{1}{5}}\left\{\int \left(f^{(2)}(x)\right)^2 dx\right\}^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = \left(\frac{1}{2}\right)^{-\frac{2}{5}}\left(\frac{3}{5}\right)^{\frac{1}{5}}\left(\frac{1}{\sqrt{\pi}}\frac{3}{8}\frac{1}{\sigma^5}\right)^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = \left(\frac{n}{40}\frac{1}{\sqrt{\pi}}\right)^{-\frac{1}{5}}\sigma$$

For the Quartic Kernel the Adaptive smoothing parameter using equation (4.4.6),

$$k_2 = \frac{1}{7} \text{ and } I_2 = \int K^2(u)du = \frac{5}{7}$$

$$h_a = k_2^{-\frac{2}{5}}\{\int K^2(u)\,du\}^{\frac{1}{5}}\left\{\int \left(f^{(2)}(x)\right)^2 dx\right\}^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = \left(\frac{1}{7}\right)^{-\frac{2}{5}} \left(\frac{5}{7}\right)^{\frac{1}{5}} \left(\frac{1}{\sqrt{\pi}}\frac{3}{8}\frac{1}{\sigma^5}\right)^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = \left(\frac{3n}{280\sqrt{\pi}}\right)^{-\frac{1}{5}} \sigma$$

Finally, for the Triweight Kernel the Adaptive smoothing parameter using equation (4.4.6),

$$k_2 = \frac{1}{9} \text{ and } I_2 = \int K^2(u)du = \frac{350}{429}$$

$$h_a = k_2^{-\frac{2}{5}}\left\{\int K^2(u)\,du\right\}^{\frac{1}{5}}\left\{\int \left(f^{(2)}(x)\right)^2 dx\right\}^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = \left(\frac{1}{9}\right)^{-\frac{2}{5}} \left(\frac{350}{429}\right)^{\frac{1}{5}} \left(\frac{1}{\sqrt{\pi}}\,\frac{3}{8}\,\frac{1}{\sigma^5}\right)^{-\frac{1}{5}} n^{-\frac{1}{5}}$$

$$h_a = \left(\frac{143n}{25200\sqrt{\pi}}\right)^{-\frac{1}{5}} \sigma$$

Table 4.1: Kernel functions and key characteristics

Kernel	Function	k_2	I_2	h_a
Gaussian	$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$	1	$\frac{1}{2\sqrt{\pi}}$	$\left(\frac{3n}{4}\right)^{-\frac{1}{5}} \sigma$
Epanechnikov	$K(u) = \frac{3}{4}(1-u^2) I_{(\lvert u\rvert<1)}$	$\frac{1}{5}$	$\frac{3}{5}$	$\left(\frac{n}{40\sqrt{\pi}}\right)^{-\frac{1}{5}} \sigma$
Quadratic	$K(u) = \frac{15}{16}(1-u^2)^2 I_{(\lvert u\rvert<1)}$	$\frac{1}{7}$	$\frac{5}{7}$	$\left(\frac{3n}{280\sqrt{\pi}}\right)^{-\frac{1}{5}} \sigma$
Triweight	$K(u) = 35/32(1-u\text{^}2)\text{^}3 I_{(\lvert u\rvert<1)}$	$\frac{1}{9}$	$\frac{350}{429}$	$\left(\frac{143n}{25200\sqrt{\pi}}\right)^{-\frac{1}{5}} \sigma$

4.5: Adaptive Smoothing Parameter

If the density function $f(x)$ of our observed data is known, these approximation equations may be solved by substitution as in section (4.4). In section (4.4), it is considered that the population from which the data is obtained is known to be Normal Distribution. All components of this equation have already solved for except $\int (f^{(2)}(x))^2 \, dx$. In most cases, the density function of the observed data will be unknown. Therefore $f^2(x)$ will also be unknown. In this case, an estimate

is needed for $f^2(x)$ and as Rahman et al. (1996) stated, this will be defined as the following:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right) \qquad (4.5.1)$$

For the Gaussian kernel $\int \left(f^{(2)}(x)\right)^2 dx$ is estimated by equation (4.5.1) as shown by B.W Silverman (1986) to be the following:

$$\int \left(f^{(2)}(x)\right)^2 dx$$

$$= \frac{3}{8n^2h^9\sqrt{\pi}}\left[h^4 \sum_{i=1}^{n}\sum_{j=1}^{n} e^{-\frac{(x_i-x_j)^2}{4h^2}} - h^2 \sum_{i=1}^{n}\sum_{j=1}^{n}\left(x_i - x_j\right)^2 e^{-\frac{(x_i-x_j)^2}{4h^2}} \right.$$

$$\left. + \frac{1}{12}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(x_i - x_j\right)^4 e^{-\frac{(x_i-x_j)^2}{4h^2}} \right] \qquad (4.5.2)$$

For the Epanechnikov kernel $\int \left(f^{(2)}(x)\right)^2 dx$ is estimated by equation (4.5.1) as shown by B. W. Silverman (1986) to the following:

$$\int \left(f^{(2)}(x)\right)^2 dx = \frac{9(U-L)}{4h^4} \qquad (4.5.2)$$

Where U stands for the upper limit and L stands for the lower limit of the integral which can be easily found while implementing the estimation process.

For the Quartic kernel $\int \left(f^{(2)}(x)\right)^2 dx$ is estimated by equation (4.5.1) as shown by B. W Silverman (1986) to be the following:

$$\int \left(f^{(2)}(x)\right)^2 dx = \frac{16(U-L)}{h^6} - \frac{96}{nh^8}\left[\frac{n}{3}(U^3-L^3) - (U^2-L^2)\sum_{i=1}^{n} x_i + (U-L)\sum_{i=1}^{n} {x_i}^2\right] + \frac{144}{n^2h^{10}}\left[\frac{n^2}{5}(U^5-L^5) - n(U^4-L^4)\sum_{i=1}^{n} x_i + \frac{2}{3}(U^3-L^3)(n\sum_{i=1}^{n} {x_i}^2 - 2\,(\sum_{i=1}^{n} x_i)^2) - 2(U^2-L^2)(\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} {x_i}^2) + (U-L)(\sum_{i=1}^{n} {x_i}^2)^2\right] \quad (4.5.3)$$

Where U stands for upper limit and L stands for lower limit of the integral which can be easily found implementing the estimation process.

For the Triweight kernel $\int \left(f^{(2)}(x)\right)^2 dx$ is estimated by equation (4.5.1) as shown by B. W Silverman (1986) to be the following:

$$\int \left(f^{(2)}(x)\right)^2 = D^2 \left[\frac{{C_4}^2}{9}(U^9 - L^9) + \frac{C_3 C_4}{4}(U^8 - L^8) + \frac{2C_2 C_4 + {C_3}^2}{7}(U^7 - L^7)\right]$$

$$+ \frac{C_1 C_4 + C_2 C_3}{3}(U^6 - L^6) + \frac{2C_0 C_4 + 2C_1 C_3 + {C_2}^2}{5}(U^5 - L^5)$$

$$+ \frac{C_0 C_3 + C_1 C_2}{2}(U^4 - L^4) + \frac{2C_0 C_2 + {C_1}^2}{3}(U^3 - L^3)$$

$$+ \frac{C_0 C_3 + C_1 C_2}{2}(U^4 - L^4) + \frac{2C_0 C_2 + {C_1}^2}{3}(U^3 - L^3)$$

$$+ C_0 C_1 (U^2 - L^2) + {C_0}^2 (U - L)$$

(4.5.4)

Where U stands for the upper limit and L stands for the lower limit of the integral which can be easily found while implementing the estimation process and

$$D = \frac{105}{16}\frac{1}{nh^5}$$

$$C_0 = -n + \frac{6}{n^2}\sum_{i=1}^{n} {x_i}^2 - \frac{5}{h^4}\sum_{i=1}^{n} {x_i}^4$$

$$C_1 = -\frac{12}{h^2}\sum_{i=1}^{n} x_i - \frac{20}{h^4}\sum_{i=1}^{n} {x_i}^3$$

$$C_2 = \frac{6n}{h^2} - \frac{30}{h^4}\sum_{i=1}^{n} {x_i}^2$$

$$C_3 = \frac{20}{h^4}\sum_{i=1}^{n} x_i$$

$$C_4 = -\frac{5n}{h^4}$$

Now substituting all the values in equation (4.5.2) we will estimate smoothing parameter h_a for four different kernels.

Chapter 5: Simulation Study

Thousand samples of sizes n = 20, 50, and 100 are taken from three different distributions, Standard Normal (representing symmetric distributions), Standard Exponential (representing skewed distributions), and a mixture of two normal Skewed Bimodal (representing multimodal and skewed distributions). Smoothing parameters, the resulting approximate mean integrated squares (AMISE), and the sum of the squared deviations of the estimated densities from the true densities are computed. For comparison, maximum likelihood estimates of the true densities also computed. Results are given in Tables 5.1 to 5.3 for the method assuming that, the true distribution is Normal Distribution and the results for the Adaptive Method are given in Tables 5.4 to 5.6.

In the tables, M_h represents mean of estimated h values, S_h represents the standard deviation of the estimated h values, M_{AMISE} represents mean of AMISE's, S_{AMISE} represents standard deviation of AMISE's, M_{DEVSQ} represents mean of the sum of the squared deviations from the true density from which the samples are generated, and S_{DEVSQ} represents standard deviation of the sum of the squared deviations from the true density from which the samples are generated.

Table: 5.1 Normal Density Method (Standard Normal Samples)

Statistic	Epanechnikov	Gaussian	Quartic	Triweight	MLE
		$n = 20$			
M_h	1.2802	0.5783	1.5166	1.7221	
S_h	0.2027	0.0916	0.2401	0.2727	
M_{AMISE}	0.0301	0.1753	0.0302	0.0303	
S_{AMISE}	0.0049	0.0288	0.0050	0.0050	
M_{DEVSQ}	2.1888	2.2819	2.1933	2.2066	1.9262
S_{DEVSQ}	1.9206	1.9737	1.9333	1.9410	2.8560
		$n = 50$			
M_h	1.0643	0.4807	1.2608	1.4317	
S_h	0.1048	0.0473	0.1241	0.1410	
M_{AMISE}	0.0142	0.0830	0.0143	0.0144	
S_{AMISE}	0.0014	0.0084	0.0014	0.0015	
M_{DEVSQ}	1.1179	1.1661	1.1212	1.1281	0.7011
S_{DEVSQ}	0.8940	0,9105	0.8982	0.9006	0.9443
		$n = 100$			
M_h	0.9296	0.4199	1.1013	1.2506	
S_h	0.0630	0.0285	0.0747	0.0848	
M_{AMISE}	0.0081	0.0473	0.0081	0.0082	
S_{AMISE}	0.0006	0.0032	0.0006	0.0006	
M_{DEVSQ}	0.6498	0.6797	0.6523	0.6566	0.3145
S_{DEVSQ}	0.4408	0.4476	0.4426	0.4436	0.3270

Table: 5.2 Normal Density Method (Standard Exponential Samples)

Statistic	Epanechnikov	Gaussian	Quartic	Triweight	MLE
		$n = 20$			
M_h	1.2330	0.5569	1.4606	1.6586	
S_h	0.3546	0.1602	0.4201	0.4770	
M_{AMISE}	0.0330	0.1924	0.0332	0.0333	
S_{AMISE}	0.0100	0.0564	0.0097	0.0098	
M_{DEVSQ}	19.1897	17.5698	18.6189	18.3474	2.2193
S_{DEVSQ}	3.9250	4.1563	3.9699	4.0064	4.2911
		$n = 50$			
M_h	1.0504	0.4745	1.2444	1.4130	
S_h	0.2063	0.0932	0.2444	0.2776	
M_{AMISE}	0.0148	0.0865	0.0149	0.0150	
S_{AMISE}	0.0029	0.0168	0.0029	0.0029	
M_{DEVSQ}	15.4504	13.9661	14.9439	14.6964	0.7730
S_{DEVSQ}	1.5063	1.6643	1.5426	1.5691	1.4913
		$n = 100$			
M_h	0.9255	0.4181	1.0964	1.2450	
S_h	0.1277	0.0577	0.1513	0.1718	
M_{AMISE}	0.0083	0.0482	0.0083	0.0083	
S_{AMISE}	0.0012	0.0067	0.0012	0.0012	
M_{DEVSQ}	13.3360	11.9986	12.8855	12.6632	0.3424
S_{DEVSQ}	0.9274	1.0174	0.9490	0.9643	0.6541

Table: 5.3 Normal Density Method (Skewed Bimodal Samples)

Statistic	Epanechnikov	Gaussian	Quartic	Triweight	MLE
		$n = 20$			
M_h	1.3955	0.6304	1.6532	1.8773	
S_h	0.1961	0.0886	0.2323	0.2638	
M_{AMISE}	0.0274	0.1599	0.0276	0.0277	
S_{AMISE}	0.0040	0.0232	0.0040	0.0040	
M_{DEVSQ}	3.2326	3.0272	3.1384	3.1014	7.6385
S_{DEVSQ}	1.4986	1.5707	1.5123	1.5230	4.9085
		$n = 50$			
M_h	1.1726	0.5297	1.3892	1.5775	
S_h	0.1034	0.0467	0.1224	0.1390	
M_{AMISE}	0.0129	0.0752	0.0130	0.0130	
S_{AMISE}	0.0012	0.0068	0.0012	0.0012	
M_{DEVSQ}	2.1124	1.9219	2.0385	2.0055	7.4941
S_{DEVSQ}	0.7503	0.8021	0.7635	0.7718	3.6442
		$n = 100$			
M_h	1.0199	0.4607	1.2082	1.3720	
S_h	0.0636	0.0287	0.0753	0.0855	
M_{AMISE}	0.0074	0.0431	0.0074	0.0075	
S_{AMISE}	0.0005	0.0027	0.0005	0.0005	
M_{DEVSQ}	1.4895	1.3326	1.4333	1.4067	8.2645
S_{DEVSQ}	0.4315	0.4602	0.4389	0.4435	3.0520

Table: 5.4 Adaptive Method (Standard Normal Samples)

Statistic	Epanechnikov	Gaussian	Quartic	Triweight	MLE
		$n = 20$			
M_h	0.5777	0.7053	0.2851	0.1503	
S_h	0.1042	0.1116	0.0443	0.0238	
M_{AMISE}	0.0329	0.4087	6.14E+05	3.37E+13	
S_{AMISE}	0.0070	0.0678	1.18E+05	1.87E+13	
M_{DEVSQ}	2.5032	4.1615	15.0734	35.0173	2.0342
S_{DEVSQ}	2.1393	3.4742	8.4550	16.3605	2.8517
		$n = 50$			
M_h	0.4677	0.4762	0.1634	0.0731	
S_h	0.0613	0.0502	0.0172	0.0077	
M_{AMISE}	0.0162	0.8228	1.86E+07	5.22+16	
S_{AMISE}	0.0034	0.0889	2.12E+06	1.63E+16	
M_{DEVSQ}	1.2822	2.6505	10.6717	28.2996	0.7275
S_{DEVSQ}	0.9555	1.6710	4.1598	8.3709	0.8670
		$n = 100$			
M_h	0.4028	0.3539	0.1079	0.0428	
S_h	0.0401	0.0248	0.0075	0.0030	
M_{AMISE}	0.0094	0.1463	2.35E+08	1.25E+19	
S_{AMISE}	0.0023	0.1032	1.69E+07	2.57E+18	
M_{DEVSQ}	0.7519	1.7882	7.9536	23.7098	0.3145
S_{DEVSQ}	0.5106	0.9178	2.3177	4.8244	0.3371

Table: 5.5 Adaptive Method (Standard Exponential Samples)

Statistic	Epanechnikov	Gaussian	Quartic	Triweight	MLE
		$n = 20$			
M_h	0.4543	0.6824	0.3431	0.1443	
S_h	0.1151	0.2044	0.2097	0.0426	
M_{AMISE}	0.0517	0.4505	3.20E+05	7.63E+13	
S_{AMISE}	0.0161	0.1427	2.06E+06	2.24E+13	
M_{DEVSQ}	16.1994	15.4121	37.2597	47.1878	2.6454
S_{DEVSQ}	6.3627	8.5469	39.3291	38.3307	7.3021
		$n = 50$			
M_h	0.3491	0.4716	0.2229	0.0716	
S_h	0.0646	0.0879	0.1071	0.0130	
M_{AMISE}	0.0400	0.8503	1.16E+06	1.14+17	
S_{AMISE}	0.0166	0.1600	3.11E+06	2.19E+16	
M_{DEVSQ}	11.3574	9.4704	19.9081	31.5157	0.7111
S_{DEVSQ}	2.2850	2.4371	17.1605	14.3509	1.3171
		$n = 100$			
M_h	0.2781	0.3494	0.1556	0.0417	
S_h	0.0309	0.0473	0.0667	0.0054	
M_{AMISE}	0.0605	1.5019	8.30E+06	2.69E+19	
S_{AMISE}	0.0240	0.2049	1.33E+07	4.11E+18	
M_{DEVSQ}	8.8679	6.8901	12.6999	26.0551	0.3512
S_{DEVSQ}	1.5539	1.4991	11.0465	8.5742	0.6851

Table: 5.6 Adaptive Method (Skewed Bimodal Samples)

Statistic	Epanechnikov	Gaussian	Quartic	Triweight	MLE
		$n = 20$			
M_h	0.6162	0.7708	0.3218	0.1644	
S_h	0.0989	0.1079	0.0497	0.0230	
M_{AMISE}	0.0317	0.3719	4.43E+05	5.70E+13	
S_{AMISE}	0.0080	0.0549	4.09E+05	8.50E+12	
M_{DEVSQ}	0.0080	0.0549	4.09E+05	8.50E+12	5.4833
S_{DEVSQ}	2.2053	3.4421	8.1741	13.4878	4.6916
		$n = 50$			
M_h	0.4854	0.5216	0.1831	0.0802	
S_h	0.0544	0.0462	0.0157	0.0071	
M_{AMISE}	0.0172	0.7486	1.36E+07	8.52E+1	
S_{AMISE}	0.0056	0.0670	2.65E+06	7.76E+16	
M_{DEVSQ}	1.8350	2.2870	8.6016	23.4679	3.3423
S_{DEVSQ}	0.8849	1.4803	3.4564	6.7987	1.3525
		$n = 100$			
M_h	0.4005	0.3866	0.1202	0.0468	
S_h	0.0374	0.0241	0.0068	0.0029	
M_{AMISE}	0.0140	1.3380	1.75E+08	1.96E+18	
S_{AMISE}	0.0088	0.842	2.25E+07	1.25E+18	
M_{DEVSQ}	1.2359	1.5941	6.5915	19.7756	2.9087
S_{DEVSQ}	0.5280	0.8308	1.8820	3.9582	0.6986

In this simulation Study, we can see that in Normal Density Method (Standard Normal Samples, Standard Exponential Samples, Skewed Bimodal Samples) Mean of estimated adaptive smoothing parameter is getting smaller value with large sample size (100) comparing to small sample size (20) for Epanechnikov kernel as well as other kernels such as Gaussian, Quartic and Triweight. Similarly, we can conclude for

$$S_h, M_{AMISE}, S_{AMISE}, M_{DEVSQ}, S_{DEVSQ}$$

Now in Adaptive Method (Standard Normal Samples, Standard Exponential Samples, Skewed Bimodal Samples) Mean of estimated adaptive smoothing parameter is getting smaller value with large sample size (100) comparing to small sample size (20) for Epanechnikov kernel as well as other kernels such as Quartic and Triweight except Gaussian Kernels. Similarly, we can conclude for

$$S_h, M_{AMISE}, S_{AMISE}, M_{DEVSQ}, S_{DEVSQ}.$$

If we know the distribution from which the data is selected then we can apply parametric method, MLE is giving the parametric estimate. After overall tables observation, we see in Gaussian and Epanechnikov kernel is giving us comparatively smaller values rather than Quartic and Triweight and if we think about Gaussian and Epanechnikov then Epanechnikov is better except some sporadic cases Gaussian gives us better results.

Maximum Likelihood Parameter Estimation in a Mixtures of Two Normal Densities are conducted using the following EM Algorithm given in Chapter 7 and has been discussed the estimation procedure for a Mixture of two normal densities are discussed in detail.

Chapter 6: Applications

A data set will be run through the program that were used in Chapter 4. The data set will be used is the waist circumference in cm from a set of 40 females that was reported by Triola(2008).This data set can be run through the iterative program, which will return the following results shown in table 6.1.

Table 6.1: Application Data Estimations

$\mathrm{K(u)}$	h	$\hat{h}_a$	$AMISE_{\hat{h}_a}$
Gaussian	7.8007	4.2391	0.0021
Epanechnikov	17.2693	13.4617	0.0024

In this table, Once the h values for each kernel are calculated, then the density function for each smoothing parameter of the observed data can be calculated. To do this, 1000 evenly distributed points within four standard deviations of the mean can be used in equation 3.2, $\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right)$. The density functions and histograms can be found and displayed as they were in the following graphs of the density estimations of the Gaussian kernel and Epanechnikov kernel.

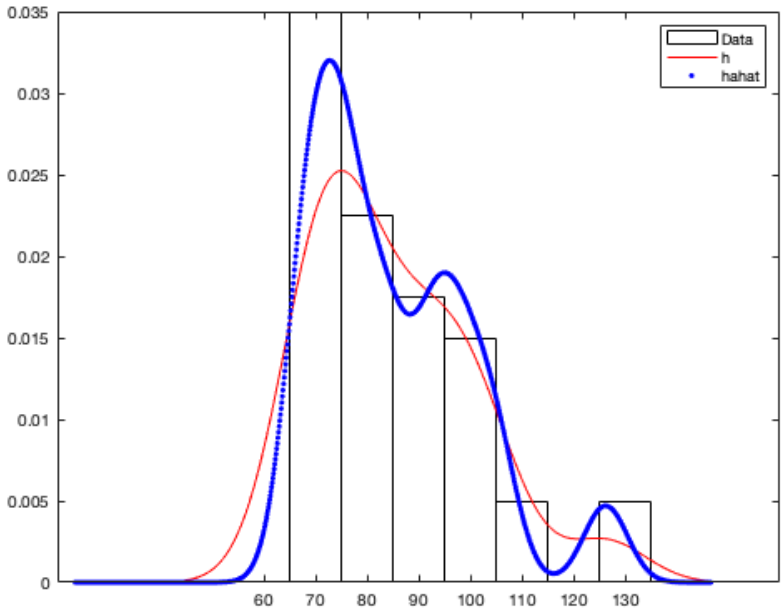


Figure 6.1: Gaussian Kernel Density Functions

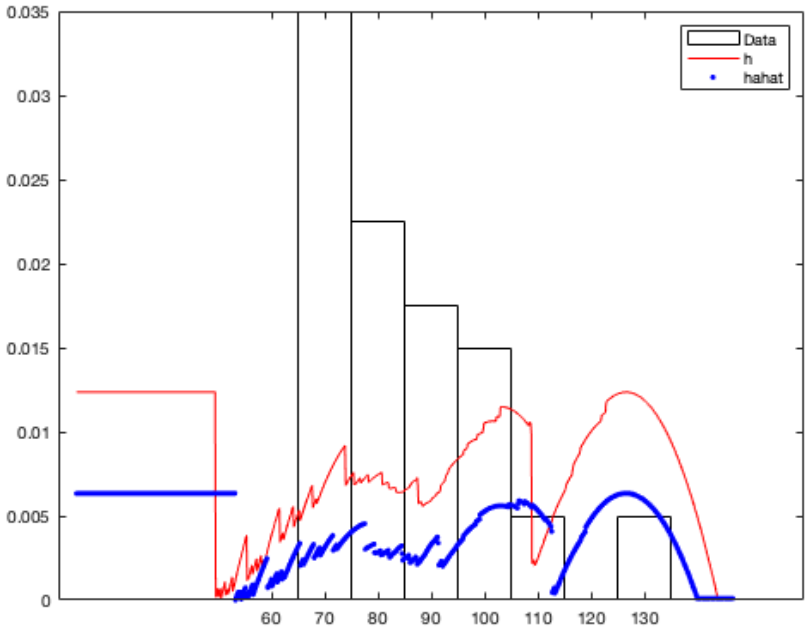


Figure 6.2: Epanechivok Kernel Density Functions

From the figures 6.1 and 6.2 we conclude that the Gaussian kernel does a much better job in displaying the observed data. Within the Gaussian figure, it can be seen that the two density functions display the entire structure of the data and demonstrates the difference the smoothing parameter value makes. In figure 6.1 smoothing parameter $\hat{h}_a$ value is smaller than h ,which has the steepest slopes and brings out all the peaks and valleys of the observed data but in figure 6.2 we can see that smoothing parameter h is giving better curve than $\hat{h}_a$.

Chapter 7: Maximum Likelihood Parameter Estimation for the Mixtures of Normal Distribution

7.1: Mixtures of normal distributions

A wide range of mixtures of normal distributions can be formed using linear combinations of independent normal densities as –

$$w_1\, N(\mu_1,\ {\sigma_1}^2)+w_1\, N(\mu_2,\ {\sigma_2}^2) + \cdots \ldots \ldots . + w_k\, N(\mu_k,\ {\sigma_k}^2) \qquad (7.1.1)$$

Where $0 < w_i < 1$ and $\sum_{i=1}^{k} w_i = 1$ and

$$N(\mu_i, {\sigma_i}^2) = \frac{1}{\sigma_i\sqrt{2\pi}}\, e^{-\frac{1}{2{\sigma_i}^2}(x-\mu_i)} \ , -\infty < x < \infty$$

For i =1,2,…..k .The mean of (7.1.1) can be written as $\mu = \sum_{i=1}^{k} w_i\mu_i$ and the standard deviation of (6.1) can be written as $\sigma = \sum_{i=1}^{k} c_i\sigma_i$. Eisenberger (1964) used a mixture of two normal distributions to give a genesis of bimodal distributions. Acuña (1999) discussed the modality of finite mixtures of normal densities. Marron and Wand (1992) used such mixtures of independent normal distributions in computing exact mean integrated squared error in kernel density estimation. Basford et al. (1997) used such finite mixtures of normal densities

in modelling the distribution of stamp paper thickness.

We know that a linear combination of independent normal distributions has a normal distribution. If the $X_i's$ are independent $N(\mu_i, {\sigma_i}^2)$ for i =1,2,……, k

then

$$C = \sum_{i=1}^{k} c_i X_i \sim N\left(\sum_{i=1}^{k} c_i \mu_i,\ \sum_{i}^{k} {c_i}^2 {\sigma_i}^2\right) \qquad (7.1.2)$$

Where the $c_i's$ are constants (Hogg and Craig (2005)). The median of the distribution in (7.1.2) is also equal to the mean $\sum_{i=1}^{k} c_i \mu_i$ but the same is not true for the distribution in (7.1.1). A general compounding of normal densities is also considered by Muirhead (1982, p. 33). Johnson (1987, p. 55) suggested the use of the mixtures of normal densities in simulating different bimodal densities. Rahman et al. (2006) computed quantiles for finite mixtures of normal distributions.

7.2: A Set of Mixtures of Independent Normal Distributions

Marron and Wand (1992) considered the following fourteen distributions given in Table 7.2.1. Figures of these fourteen distributions are given in Figure 7.2.1 and in Figure 7.2.14.

Table 7.2.1: Selected examples of normal mixture densities

# 1 Skewed unimodal	$\frac{1}{5}N(0,1)+\frac{1}{5}N\left(\frac{1}{2},\left(\frac{2}{3}\right)^2\right)+\frac{3}{5}N\left(\frac{13}{12},\left(\frac{5}{9}\right)^2\right)$
# 2 Strongly skewed	$\sum_{l=0}^{7}\frac{1}{8}\ N\left[\left(3\left(\left(\frac{2}{3}\right)^l\right)-1\right),\left(\frac{2}{3}\right)^{2l}\right]$
# 3 Kurtotic unimodal	$\frac{2}{3}N(0,1)+\frac{1}{3}\ N\left(0,\left(\frac{1}{10}\right)^2\right)$
# 4 Outlier	$\frac{1}{10}N(0,1)+\frac{9}{10}\ N\left(0,\left(\frac{1}{10}\right)^2\right)$
# 5 Bimodal	$\frac{1}{2}\ N\left(-1,\left(\frac{2}{3}\right)^2\right)+\frac{1}{2}\ N\left(1,\left(\frac{2}{3}\right)^2\right)$
# 6 Separated bimodal	$\frac{1}{2}\ N\left(-\frac{3}{2},\left(\frac{1}{2}\right)^2\right)+\frac{1}{2}\ N\left(\frac{3}{2},\left(\frac{1}{2}\right)^2\right)$
#7 Skewed bimodal	$\frac{3}{4}N(0,1)+\frac{1}{4}\ N\left(\frac{3}{2},\left(\frac{1}{3}\right)^2\right)$
# 8 Trimodal	$\frac{9}{20}\ N\left(-\frac{6}{5},\left(\frac{3}{5}\right)^2\right)+\frac{9}{20}\ N\left(\frac{6}{5},\left(\frac{3}{5}\right)^2\right)+\frac{1}{10}\ N\left(0,\left(\frac{1}{4}\right)^2\right)$
# 9 Claw	$\frac{1}{2}N(0,1)+\sum_{l=0}^{4}\frac{1}{10}N\left(\frac{l}{2}-1,\left(\frac{1}{10}\right)^2\right)$
# 10 Double claw	$\frac{49}{100}\ N\left(-1,\left(\frac{2}{3}\right)^2\right)+\frac{49}{100}\ N\left(1,\left(\frac{2}{3}\right)^2\right)$ $+\sum_{l=0}^{6}\frac{1}{350}N\left(\frac{l-3}{2},(\frac{1}{100})^2\right)$
# 11 Asymmetric claw	$\frac{1}{2}N(0,1)+\sum_{l=-2}^{2}\frac{2^{1-l}}{31}N\left(l+\frac{1}{2},(\frac{2^{-l}}{10})^2\right)$
	$\sum_{l=0}^{1}\frac{46}{100}N\left(2l-1,\left(\frac{2}{3}\right)^2\right)+\sum_{l=1}^{3}\frac{1}{300}N\left(\frac{l}{2},\left(\frac{1}{100}\right)^2\right)$

# 12 Asymmetric double claw	$+\sum_{l=1}^{3}\frac{7}{300}N\left(\frac{l}{2},\left(\frac{7}{100}\right)^2\right)$
#13 Smooth comb	$\sum_{l=0}^{5}\frac{2^{5-l}}{63}N\left(\frac{65-96\left(\frac{1}{2}\right)^l}{21},\frac{\left(\frac{32}{63}\right)^2}{2^{2l}}\right)$
#14 Discrete comb	$\sum_{l=0}^{2}\frac{2}{7}N\left(\frac{12l-15}{7},\left(\frac{2}{7}\right)^2\right)+\sum_{l=8}^{10}\frac{1}{21}N\left(\frac{2l}{7},\left(\frac{1}{21}\right)^2\right)$

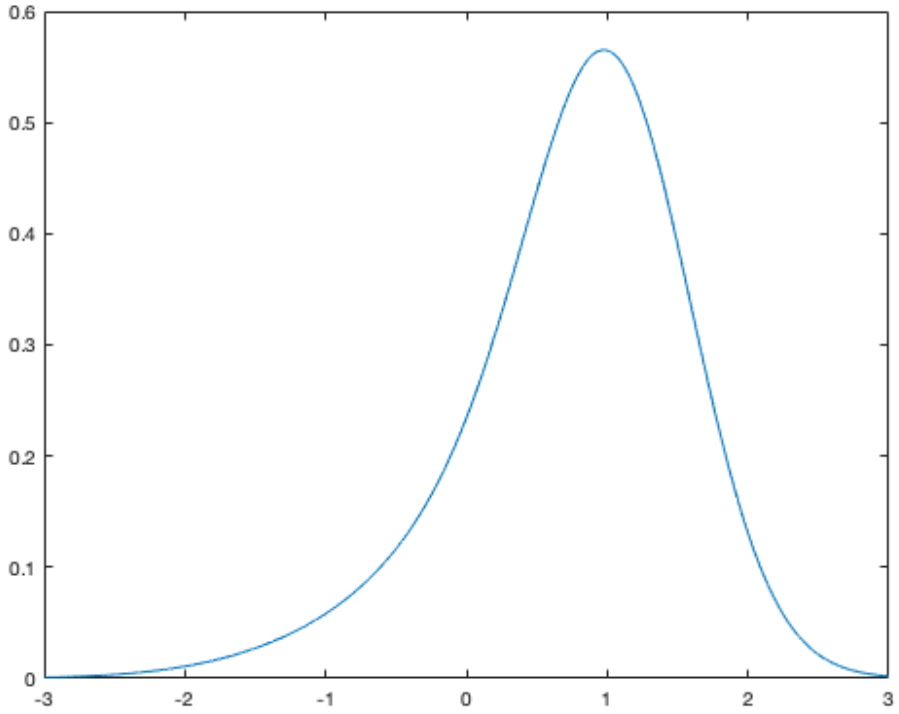
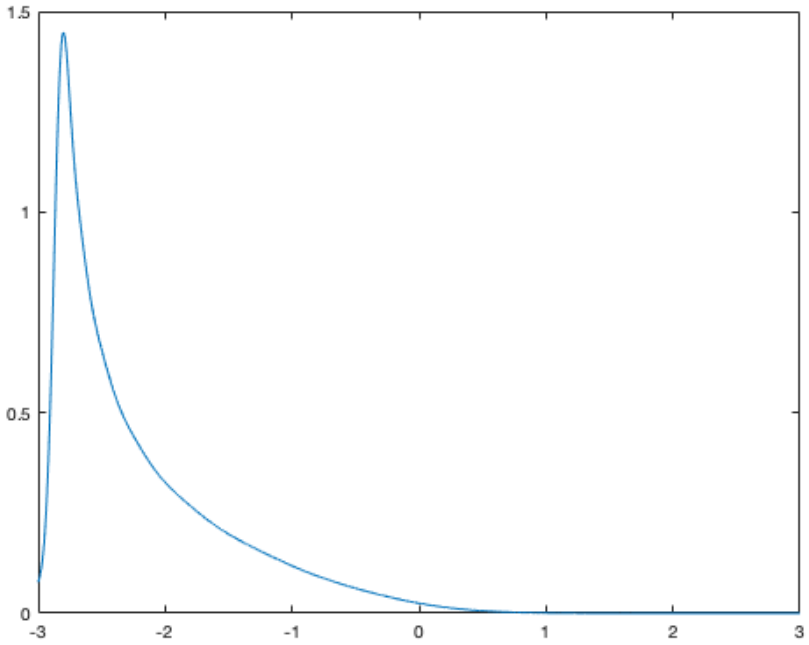


Figure 7.2.1: Skewed Unimodal Density



\

Figure 7.2.2: Strongly Skewed Density

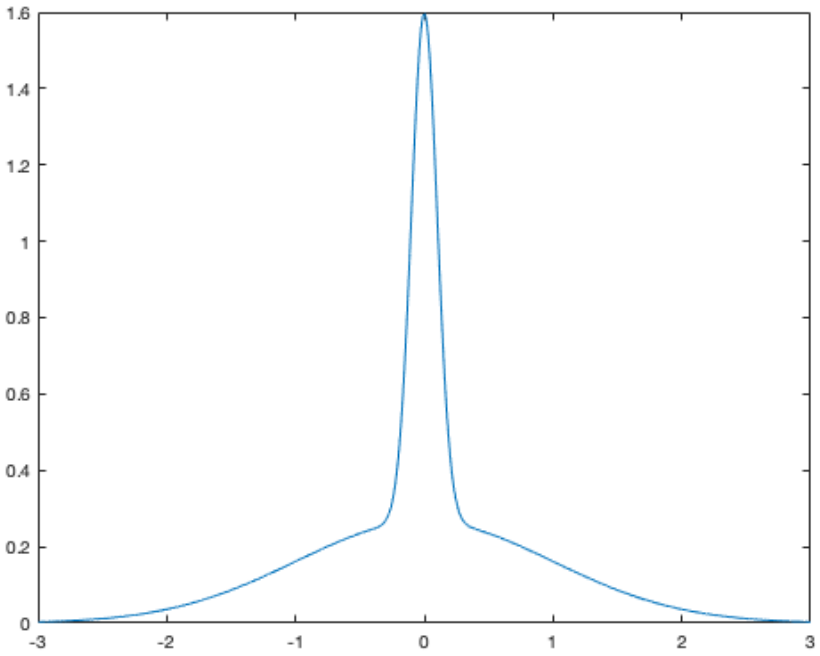


Figure 7.2.3: Kurtotic Density

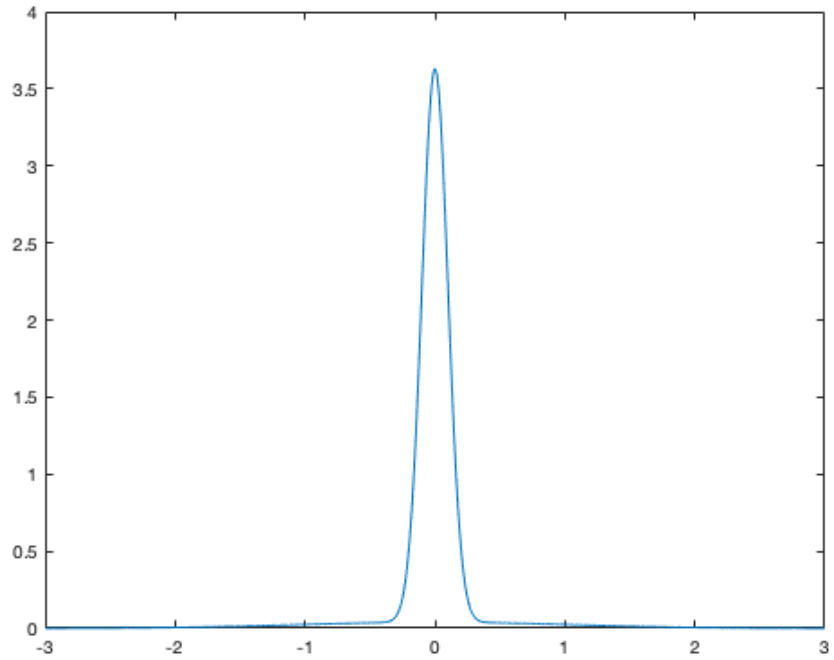


Figure 7.2.4: Outlier Density

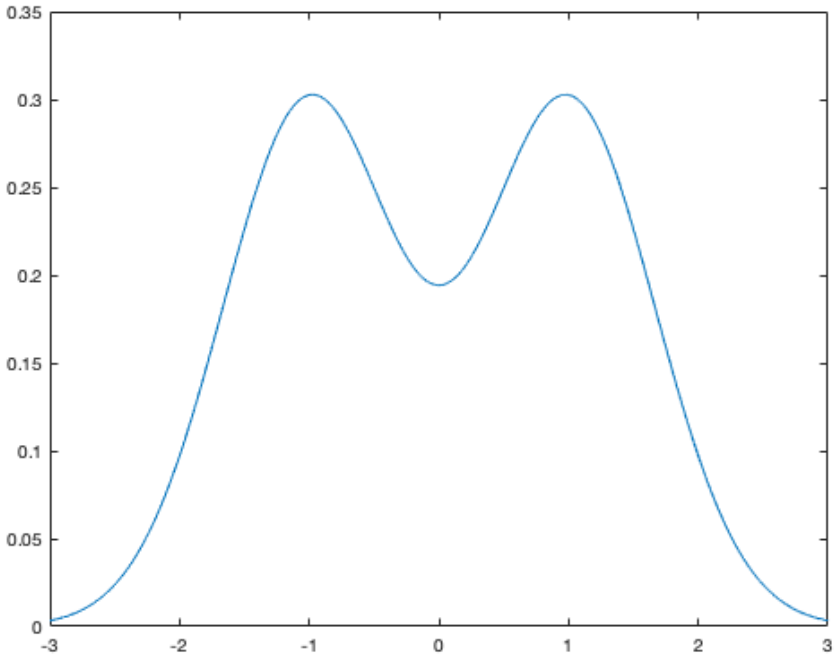


Figure 7.2.5: Bimodal Density

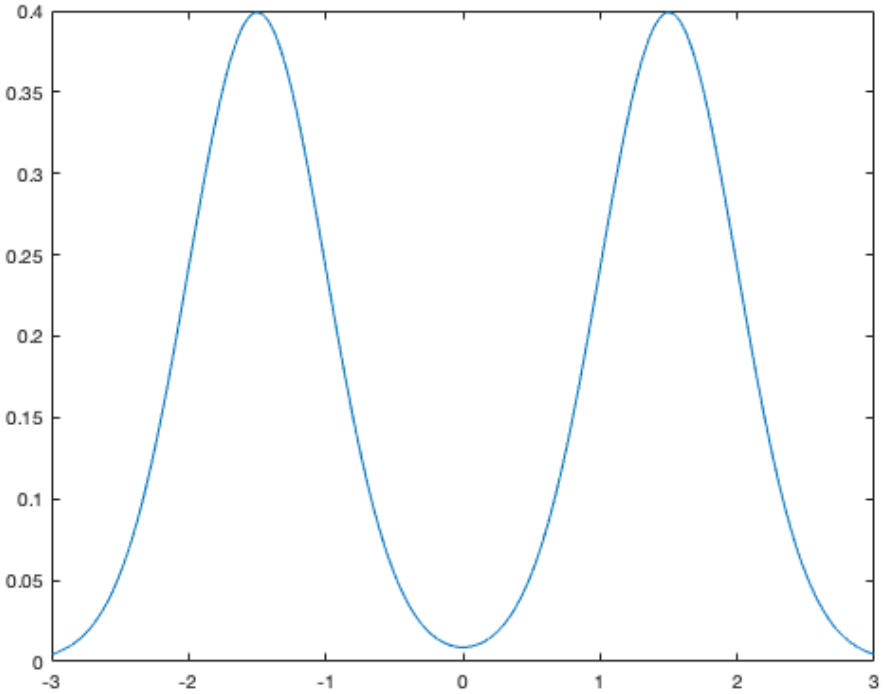


Figure 7.2.6: Separated Bimodal Density

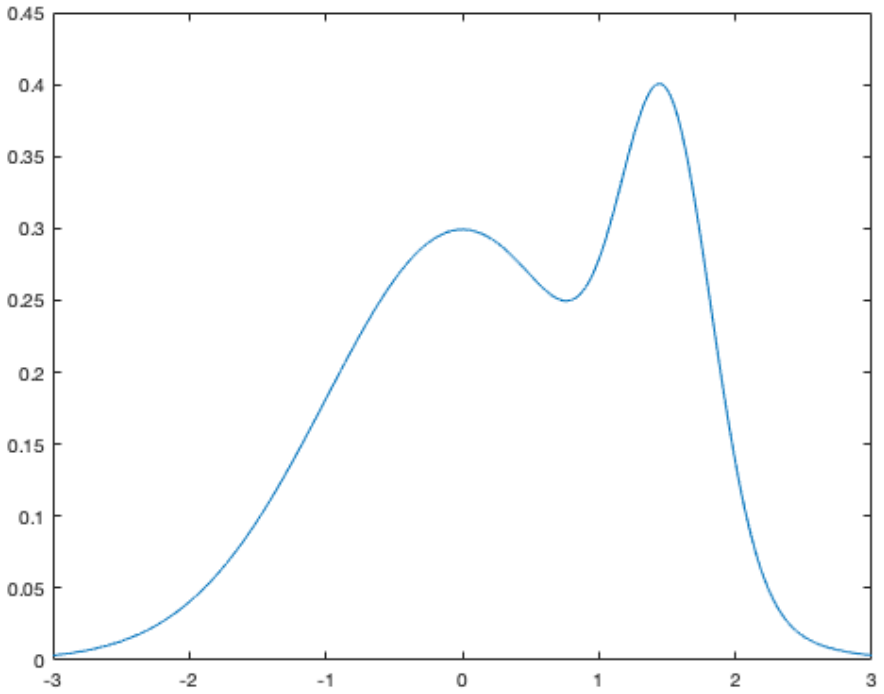


Figure 7.2.7: Asymmetric (Skewed) Bimodal Density

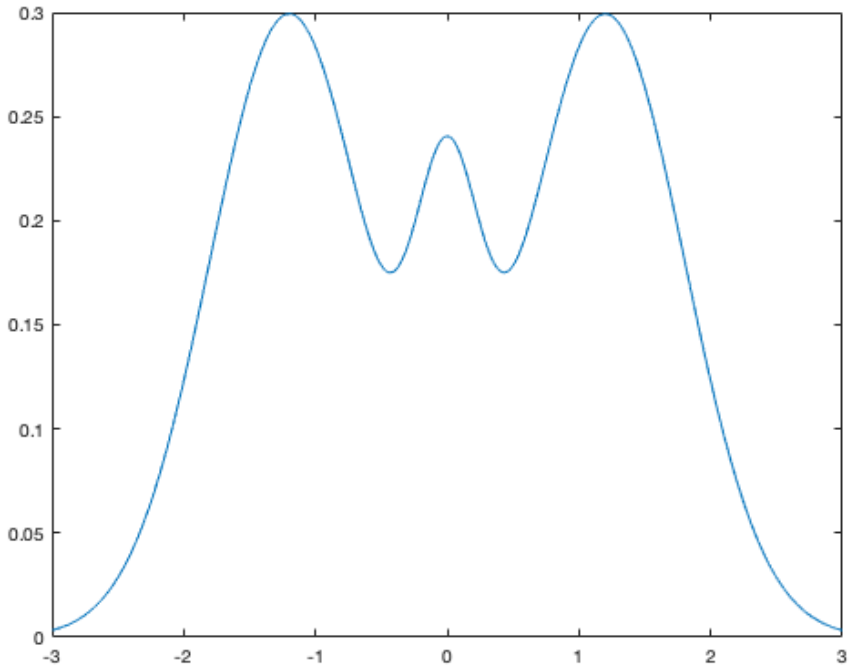


Figure 7.2.8: Trimodal Density

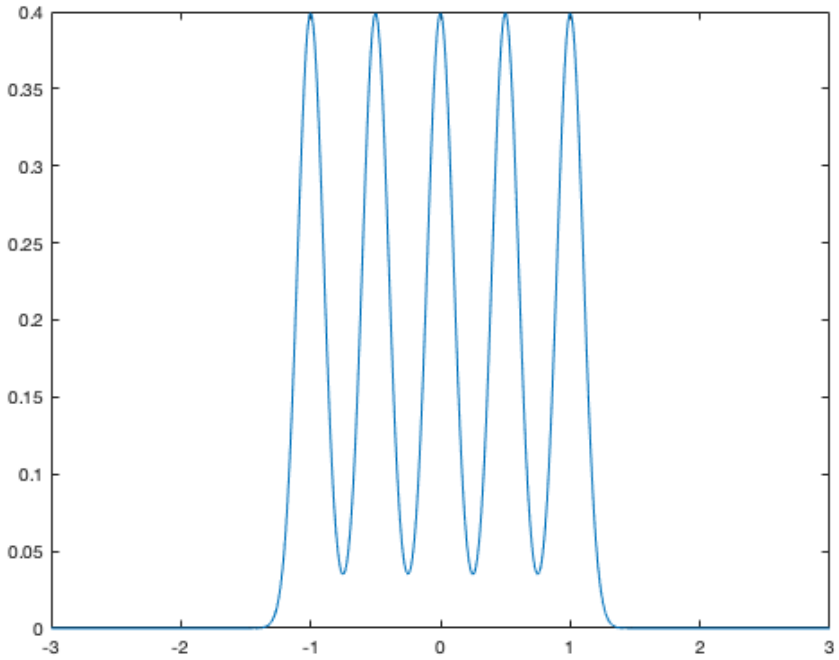


Figure 7.2.9: Claw Density

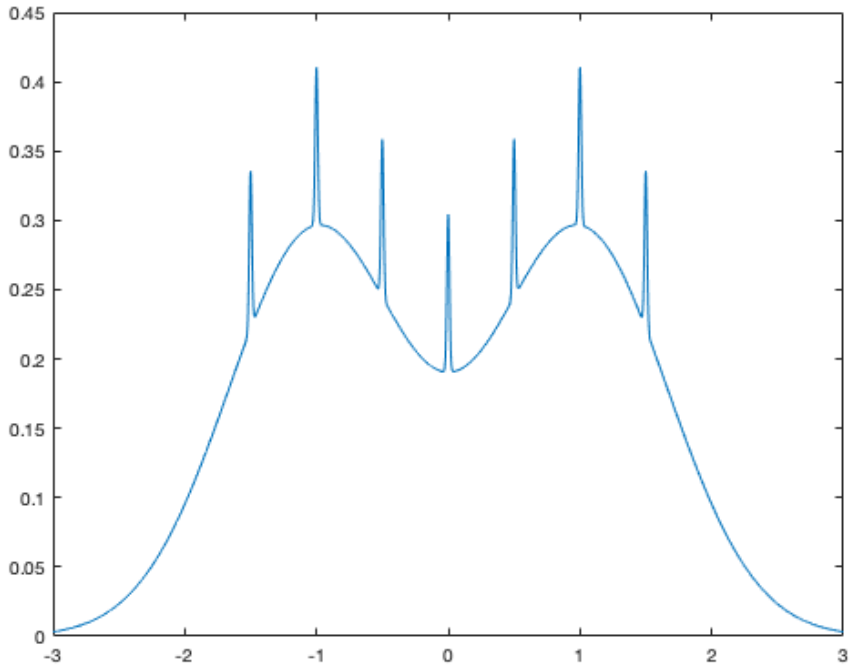


Figure 7.2.10: Double Claw Density

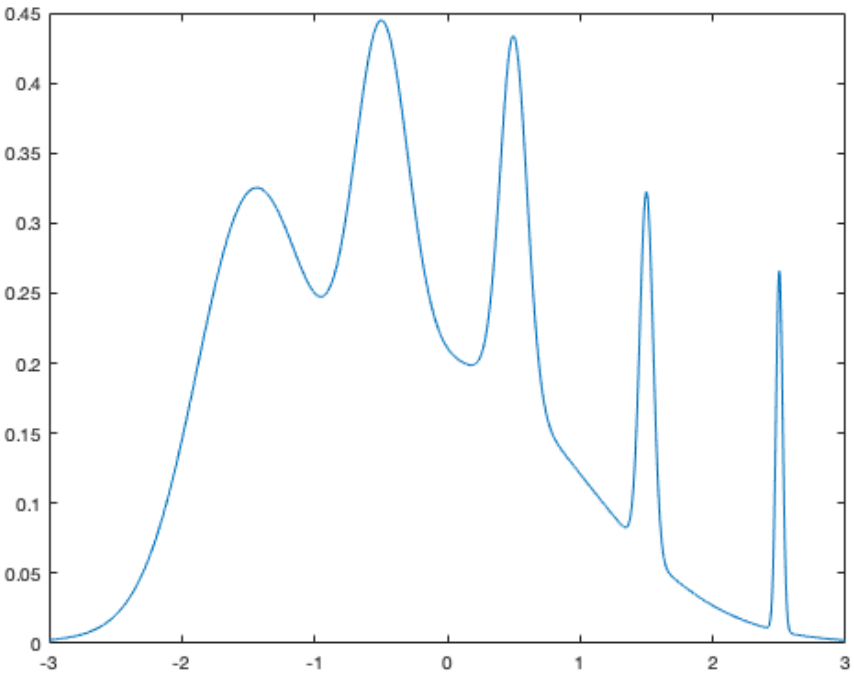


Figure 7.2.11: Asymmetric Claw Density

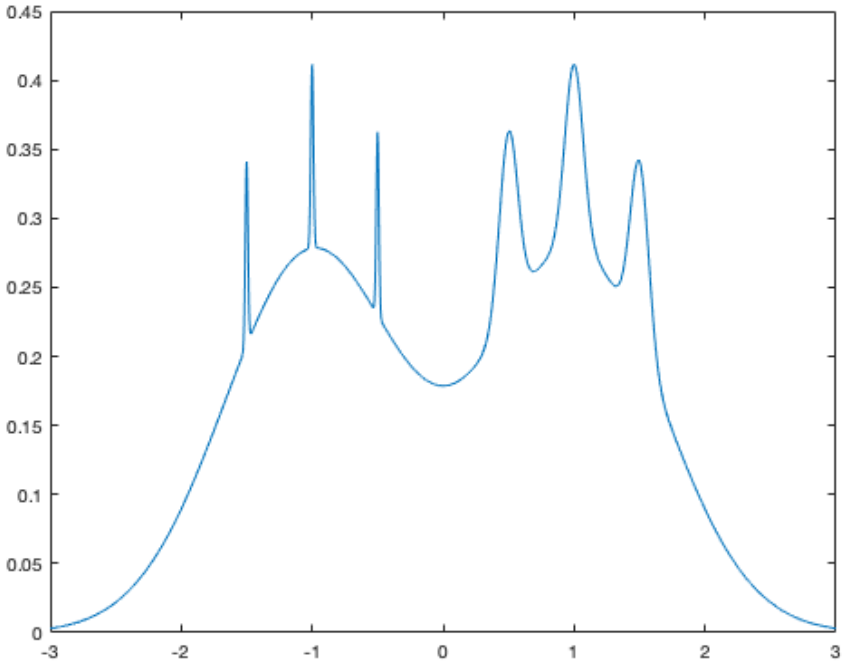


Figure 7.2.12: Asymmetric Double Claw Density

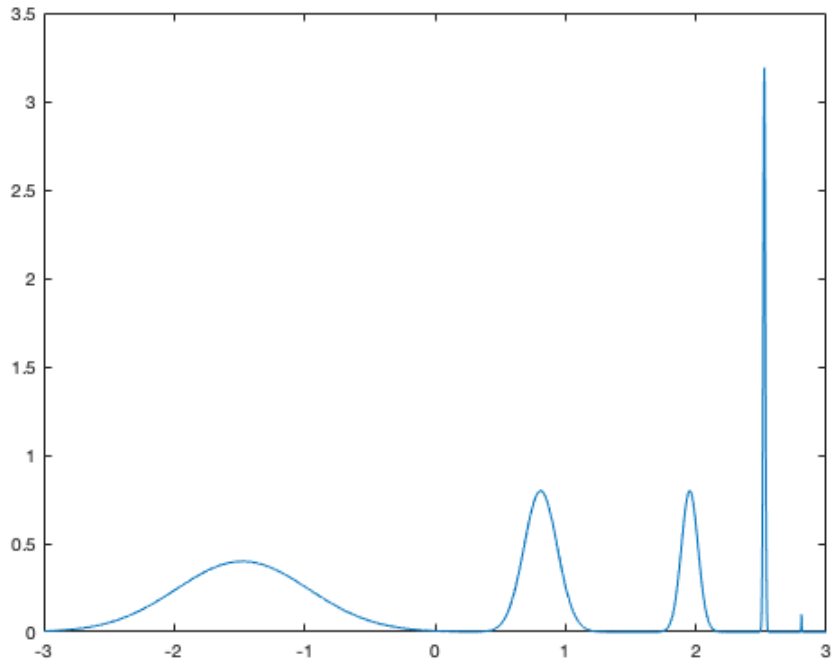


Figure 7.2.13: Smooth Comb Density

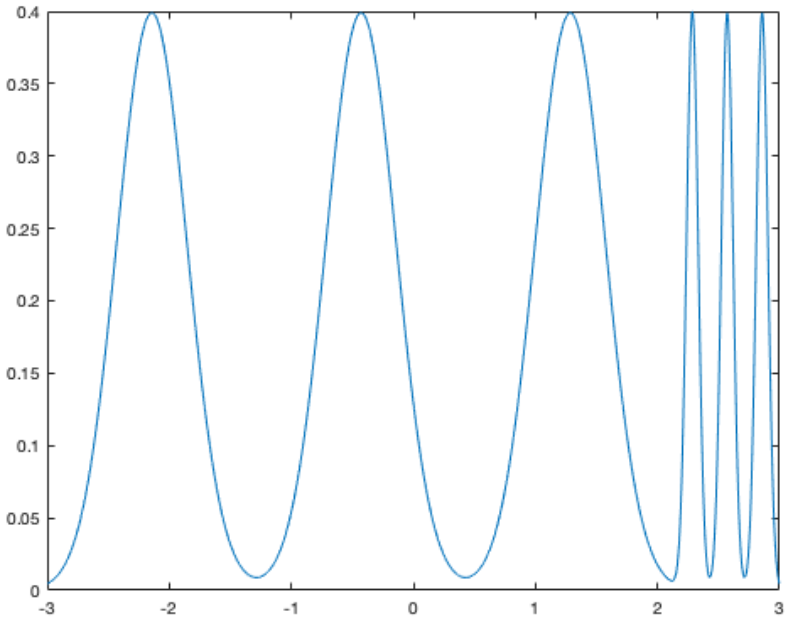


Figure 7.2.14: Discrete Comb Density

The skewed unimodal density, #2, is not far from Gaussian in shape, being only mildly skewed and resembles to the extreme value density in appearance. The next three densities are much further from the Gaussian. The strongly skewed density, #2, departs in the direction of skewness and resembles a lognormal density. The kurtotic unimodal density #3, is a departure in the direction of heavy kurtosis. The outlier density, #4, has a shape similar to the Gaussian, except that 10% of the observations are multiplied by 10, that is, are strong outliers.

The bimodal densities, # 5, #6 and #7 and the trimodal density, #8, represent important but simple departures from a unimodal shape. The claw density, #9, is of special interest because this is where the surprising result in regard to local

minima in the MISE (mean integrated squared error) occurs. The double claw density, #10, is essentially the same as #5, except that approximately 2% of the probability mass appears in the spikes. The asymptotic claw and double claw densities, #11 and #12, are modifications of #9 and #10, respectively. The smooth and discrete comb densities, #13 and #14, are enhancements of the basic idea of #6 and both have much different Fourier transform properties.

Here, we explore parameter estimation procedure for a mixture of two independent normal densities which will cover the scopes for the densities #3 Kurtotic unimodal, #4 Outlier, #5 Bimodal, #6 Separated bimodal, and #7 Skewed bimodal in Table 7.2.1.

7.3: Parameter Estimation in a Mixture of two Normal Densities using EM Algorithm

Let $x_1, x_2, \ldots, x_n$ be a random sample having density function

$$f(x) = \alpha \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2{\sigma_1}^2}(x-\mu_1)^2} + (1-\alpha)\frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{1}{2{\sigma_2}^2}(x-\mu_2)^2} \qquad (7.3.1)$$

Where $0 \leq \alpha \leq 1$. The parameters $\alpha, \mu_1, \mu_2, {\sigma_1}^2, {\sigma_2}^2$ can be estimated by an EM algorithm (see Lange [6], page 127). If

$$P_{mi} = \frac{\alpha_m \phi_{1,mi}}{\alpha_m \phi_{1,mi} + (1-\alpha_m)\phi_{2,mi}}$$

Where, $\phi_{1,mi} = \frac{1}{\sqrt{2\pi}\sigma_{1,m}} e^{-\frac{1}{2\sigma^2{}_{1,m}}(x_i-\mu_{1,m})^2}$

And $\phi_{2,mi} = \frac{1}{\sqrt{2\pi}\sigma_{2,m}} e^{-\frac{1}{2\sigma^2{}_{2,m}}(x_i-\mu_{2,m})^2}$

are the normal probability density functions (PDF). P_{mi} is the current posterior probability that observation i is taken from population (7.3.1), then EM algorithm updates are:

$$\alpha_{m+1} = \frac{1}{n}\sum_{i=1}^{n} P_{mi}$$

$$\mu_{1,(m+1)} = \frac{\sum_{i=1}^{n} P_{mi} x_i}{\sum_{i=1}^{n} P_{mi}}$$

$$\mu_{2,(m+1)} = \frac{\sum_{i=1}^{n} q_{mi} x_i}{\sum_{i=1}^{n} q_{mi}}$$

$$\sigma^2{}_{1,(m+1)} = \frac{\frac{1}{n}\sum_{i=1}^{n} P_{mi}(x_i - \mu_{1,(m+1)})^2}{\sum_{i=1}^{n} P_{mi}}$$

$$\sigma^2{}_{2,(m+1)} = \frac{\frac{1}{n}\sum_{i=1}^{n}(1-P_{mi})\,(x_i - \mu_{1,(m+1)})^2}{\sum_{i=1}^{n} q_{mi}}$$

Where $q_{mi} = 1 - P_{mi}$ and m indicates the iteration step.

7.4: Maximum Likelihood Estimation using Newton-Raphson Algorithm

The likelihood function for the density function in (7.3.1) can be written as

$$L = \prod_{i=1}^{n}\left[\alpha \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2{\sigma_1}^2}(x-\mu_1)^2} + (1-\alpha)\frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{1}{2{\sigma_2}^2}(x-\mu_2)^2}\right]$$

Where $0 \le \alpha \le 1$ $f(x_i;\mu_1,{\sigma_1}^2) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{\frac{-1}{2{\sigma_1}^2}(x_i-\mu_1)^2}$ and

$$f(x_i;\mu_2,{\sigma_2}^2) = \frac{1}{\sqrt{2\pi}\sigma_2} e^{\frac{-1}{2{\sigma_2}^2}(x_i-\mu_2)^2}$$

The log-likelihood function can be written as

$$\ln L = \sum_{i=1}^{n} \ln\left[\alpha \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2{\sigma_1}^2}(x-\mu_1)^2} + (1-\alpha)\frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{1}{2{\sigma_2}^2}(x-\mu_2)^2}\right] \qquad (7.4.1)$$

Where "ln" stands for natural logarithm. Then the first derivatives of the log-likelihood can be written as

$$\frac{\partial lnL}{\partial \alpha} = \sum_{i=1}^{n} \frac{f(x_i;\mu_1,{\sigma_1}^2) - f(x_i;\mu_2,{\sigma_2}^2)}{\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)}$$

$$\frac{\partial lnL}{\partial \mu_1} = \sum_{i=1}^{n} \frac{\alpha \, \frac{x_i - \mu_1}{{\sigma_1}^2} f(x_i; \mu_1, {\sigma_1}^2)}{\alpha f(x_i; \mu_1, {\sigma_1}^2) + (1-\alpha) f(x_i; \mu_2, {\sigma_2}^2)}$$

$$\frac{\partial lnL}{\partial \mu_2} = \sum_{i=1}^{n} \frac{(1-\alpha) \, \frac{x_i - \mu_2}{{\sigma_2}^2} f(x_i; \mu_2, {\sigma_2}^2)}{\alpha f(x_i; \mu_1, {\sigma_1}^2) + (1-\alpha) f(x_i; \mu_2, {\sigma_2}^2)}$$

$$\frac{\partial lnL}{\partial {\sigma_1}^2} = \sum_{i=1}^{n} \frac{\frac{\alpha}{2{\sigma_1}^4} ({\sigma_1}^2 - (x_i - \mu_1)^2) \, f(x_i; \mu_1, {\sigma_1}^2)}{\alpha f(x_i; \mu_1, {\sigma_1}^2) + (1-\alpha) f(x_i; \mu_2, {\sigma_2}^2)}$$

$$\frac{\partial lnL}{\partial {\sigma_2}^2} = \sum_{i=1}^{n} \frac{\frac{(1-\alpha)}{2{\sigma_2}^4} ({\sigma_2}^2 - (x_i - \mu_2)^2) \, f(x_i; \mu_2, {\sigma_2}^2)}{\alpha f(x_i; \mu_1, {\sigma_1}^2) + (1-\alpha) f(x_i; \mu_2, {\sigma_2}^2)}$$

The second derivatives are

$$\frac{\partial^2 lnL}{\partial \alpha^2} = \sum_{i=1}^{n} \frac{-\left(f(x_i; \mu_1, {\sigma_1}^2) - f(x_i; \mu_2, {\sigma_2}^2)\right)^2}{\left(\alpha f(x_i; \mu_1, {\sigma_1}^2) + (1-\alpha) f(x_i; \mu_2, {\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial \alpha \partial \mu_1} = \sum_{i=1}^{n} \frac{\frac{x_i - \mu_1}{{\sigma_1}^2} f(x_i; \mu_1, {\sigma_1}^2) \, f(x_i; \mu_2, {\sigma_2}^2)}{\left(\alpha f(x_i; \mu_1, {\sigma_1}^2) + (1-\alpha) f(x_i; \mu_2, {\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial \alpha \partial \mu_2} = \sum_{i=1}^{n} \frac{\frac{x_i - \mu_2}{{\sigma_2}^2} f(x_i; \mu_1, {\sigma_1}^2) \, f(x_i; \mu_2, {\sigma_2}^2)}{\left(\alpha f(x_i; \mu_1, {\sigma_1}^2) + (1-\alpha) f(x_i; \mu_2, {\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial\alpha\partial{\sigma_1}^2} = \sum_{i=1}^{n} \frac{\frac{1}{2{\sigma_2}^4}({\sigma_1}^2 - (x_i - \mu_2)^2)\, f(x_i;\mu_1,{\sigma_1}^2)\, f(x_i;\mu_2,{\sigma_2}^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial\alpha\partial{\sigma_2}^2} = \sum_{i=1}^{n} \frac{\frac{1}{2{\sigma_2}^4}({\sigma_2}^2 - (x_i - \mu_1)^2)\, f(x_i;\mu_1,{\sigma_1}^2)\, f(x_i;\mu_2,{\sigma_2}^2)}{(\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha)\, f(x_i;\mu_2,{\sigma_2}^2))^2}$$

$$\frac{\partial^2 lnL}{\partial{\mu_1}^2} = \sum_{i=1}^{n} \frac{-\frac{\alpha f(x_i;\mu_1,{\sigma_1}^2)}{{\sigma_1}^2}\{\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\left(1 - \frac{(x_i-\mu_1)^2}{{\sigma_1}^2}\right)\}}{(\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha) f(x_i;\mu_2,{\sigma_2}^2))^2}$$

$$\frac{\partial^2 lnL}{\partial\mu_1\partial\mu_2} = \sum_{i=1}^{n} \frac{-\alpha f(x_i;\mu_1,{\sigma_1}^2)\frac{x_i-\mu_1}{{\sigma_1}^2}(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\frac{x_i-\mu_2}{{\sigma_2}^2}}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial\mu_1\partial{\sigma_1}^2} = -\sum_{i=1}^{n} \frac{\frac{\alpha^2 f^2(x_i;\mu_1,{\sigma_1}^2)(x_i-\mu_1)}{{\sigma_2}^4}}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\alpha(1-\alpha) f(x_i;\mu_1,{\sigma_1}^2)(x_i-\mu_1)\, f(x_i;\mu_2,{\sigma_2}^2)(x_i-\mu_2)}{2{\sigma_2}^6}$$

$$-\sum_{i=1}^{n} \frac{(3{\sigma_1}^2 - (x_i-\mu_1)^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2) + (1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_2}^2} = -\sum_{i=1}^{n} \frac{\frac{x_i-\mu_1}{2{\sigma_1}^2{\sigma_2}^2}\alpha\,(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)({\sigma_2}^2-(x_i-\mu_2)^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial {\mu_2}^2} = \sum_{i=1}^{n} \frac{-\frac{(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)}{{\sigma_2}^2}\{(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)+\alpha f(x_i;\mu_1,{\sigma_1}^2)\left(1-\frac{(x_i-\mu_1)^2}{{\sigma_2}^2}\right)\}}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_1}^2} = \sum_{i=1}^{n} \frac{\frac{x_i-\mu_2}{2{\sigma_1}^4{\sigma_2}^2}\alpha\,(1-\alpha) f(x_i;\mu_1,{\sigma_1}^2) f(x_i;\mu_2,{\sigma_2}^2)({\sigma_1}^2-(x_i-\mu_1)^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_2}^2} = -\sum_{i=1}^{n} \frac{\frac{(1-\alpha)^2 f^2(x_i;\mu_2,{\sigma_2}^2)(x_i-\mu_2)}{{\sigma_2}^4}}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\alpha(1-\alpha) f(x_i;\mu_1,{\sigma_1}^2)\, f(x_i;\mu_2,{\sigma_2}^2)(x_i-\mu_2)}{2{\sigma_2}^6}$$

$$-\sum_{i=1}^{n} \frac{(3{\sigma_2}^2-(x_i-\mu_2)^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial ({\sigma_1}^2)^2} = \sum_{i=1}^{n} \frac{\frac{\alpha^2 f^2(x_i;\mu_1,{\sigma_1}^2)}{{\sigma_2}^4}({\sigma_1}^2-2(x_i-\mu_1)^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha) f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\alpha(1-\alpha)f(x_i;\mu_1,{\sigma_1}^2)\,f(x_i;\mu_2,{\sigma_2}^2)}{4{\sigma_1}^8}$$

$$+\sum_{i=1}^{n}\frac{3{\sigma_1}^4-6{\sigma_1}^2(x_i-\mu_1)^2+(x_i-\mu_1)^4}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha)f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial{\sigma_1}^2\partial{\sigma_2}^2}=-\sum_{i=1}^{n}\frac{\frac{\alpha\,(1-\alpha)f(x_i;\mu_2,{\sigma_2}^2)f(x_i;\mu_2,{\sigma_2}^2)}{4{\sigma_1}^4{\sigma_2}^4}({\sigma_1}^2-(x_i-\mu_1)^2)({\sigma_2}^2-(x_i-\mu_2)^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha)f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\partial^2 lnL}{\partial({\sigma_2}^2)^2}=\sum_{i=1}^{n}\frac{\frac{(1-\alpha)^2 f^2(x_i;\mu_2,{\sigma_2}^2)}{{\sigma_2}^4}({\sigma_2}^2-2(x_i-\mu_2)^2)}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha)f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

$$\frac{\alpha(1-\alpha)f(x_i;\mu_1,{\sigma_1}^2)\,f(x_i;\mu_2,{\sigma_2}^2)}{4{\sigma_2}^8}$$

$$+\sum_{i=1}^{n}\frac{3{\sigma_2}^4-6{\sigma_2}^2(x_i-\mu_2)^2+(x_i-\mu_2)^4}{\left(\alpha f(x_i;\mu_1,{\sigma_1}^2)+(1-\alpha)f(x_i;\mu_2,{\sigma_2}^2)\right)^2}$$

The likelihood equations can be written as

$$U_F = \begin{bmatrix} \frac{\partial lnL}{\partial \alpha} \\ \frac{\partial lnL}{\partial \mu_1} \\ \frac{\partial lnL}{\partial \mu_2} \\ \frac{\partial lnL}{\partial {\sigma_1}^2} \\ \frac{\partial lnL}{\partial {\sigma_2}^2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Then these nonlinear likelihood equations can be solved using the Newton-Raphson method as

$$B_F^{(l+1)} = B_F^{(l)} - I_F^{-1\,(l)} U_F^{(l)}$$

where F in suffix represents all five parameters, (l) stands for the iteration number and matrix of the second derivatives of the log likelihood I_F can be written as

$$I_F^{(l)} = \begin{bmatrix} \frac{\partial^2 lnL}{\partial \alpha^2} & \frac{\partial^2 lnL}{\partial \alpha \partial \mu_1} & \frac{\partial^2 lnL}{\partial \alpha \partial \mu_2} & \frac{\partial^2 lnL}{\partial \alpha \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \alpha \partial {\sigma_2}^2} \\ \frac{\partial^2 lnL}{\partial \alpha \partial \mu_1} & \frac{\partial^2 lnL}{\partial {\mu_1}^2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial \mu_2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_2}^2} \\ \frac{\partial^2 lnL}{\partial \alpha \partial \mu_2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial \mu_2} & \frac{\partial^2 lnL}{\partial {\mu_2}^2} & \frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_2}^2} \\ \frac{\partial^2 lnL}{\partial \alpha \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial ({\sigma_1}^2)^2} & \frac{\partial^2 lnL}{\partial {\sigma_1}^2 \partial {\sigma_2}^2} \\ \frac{\partial^2 lnL}{\partial \alpha \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial {\sigma_1}^2 \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial ({\sigma_2}^2)^2} \end{bmatrix}$$

And $$B_F^{(l)} = \begin{bmatrix} \hat{\alpha}^{(l)} \\ \hat{\mu}_1^{(l)} \\ \hat{\mu}_2^{(l)} \\ \hat{\sigma}_1^{2(l)} \\ \hat{\sigma}_2^{2(l)} \end{bmatrix}$$

7.5: Parameter Estimation Using EM Algorithm involving MLE

Let $x_1, x_2, \ldots, x_n$ be a random sample having density function

$$f(x) = \alpha \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{1}{2{\sigma_1}^2}(x-\mu_1)^2} + (1-\alpha)\frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{1}{2{\sigma_2}^2}(x-\mu_2)^2} \quad (7.5.1)$$

Where $0 \leq \alpha \leq 1$. The parameters $\alpha, \mu_1, \mu_2, {\sigma_1}^2, {\sigma_2}^2$ can be estimated by an EM algorithm (see Lange [6], page 127). If

$$P_{mi} = \frac{\alpha_m \phi_{1,mi}}{\alpha_m \phi_{1,mi} + (1-\alpha_m)\phi_{2,mi}}$$

Where $\phi_{1,mi} = \frac{1}{\sqrt{2\pi}\,\sigma_{1,m}}\, e^{-\frac{1}{2{\sigma^2}_{1,m}}(x_i-\mu_{1,m})^2}$

And $\Phi_{2,mi} = \frac{1}{\sqrt{2\pi}\,\sigma_{2,m}}\, e^{-\frac{1}{2{\sigma^2}_{2,m}}(x_i-\mu_{2,m})^2}$

are the normal probability density functions (PDF). P_{mi} is the current posterior probability that observation i is taken from population (7.5.1), then EM algorithm updates are

$$\alpha_{m+1} = \frac{1}{n}\sum_{i=1}^{n} P_{mi}$$

Where $q_{mi} = 1 - P_{mi}$ and m indicates the iteration step. Then the other four parameters are computed by maximizing the log-likelihood function for fixed α The likelihood equations can be written as

$$U_{R(m+1)} = \begin{bmatrix} \frac{\partial lnL}{\partial \mu_1} \\ \frac{\partial lnL}{\partial \mu_2} \\ \frac{\partial lnL}{\partial {\sigma_1}^2} \\ \frac{\partial lnL}{\partial {\sigma_2}^2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Where m stands for the EM algorithm iteration. Then these nonlinear likelihood equations can be solved using the Newton-Raphson method as

$$I_{R(m+1)}^{(l+1)} = B_{R(m+1)}^{(l)} - I_{R(m+1)}^{-1\,(l)} U_{R(m+1)}^{(l+1)}$$

where R in suffix represents reduced (four) parameters, (l) stands for the Newton-Raphson iteration number, and matrix of the second derivatives of the log likelihood $I_{R(m+1)}$can be written as

$$I_{R(m+1)} = \begin{bmatrix} \frac{\partial^2 lnL}{\partial {\mu_1}^2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial \mu_2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_2}^2} \\ \frac{\partial^2 lnL}{\partial \mu_1 \partial \mu_2} & \frac{\partial^2 lnL}{\partial {\mu_2}^2} & \frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_2}^2} \\ \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_1}^2} & \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial ({\sigma_1}^2)^2} & \frac{\partial^2 lnL}{\partial {\sigma_1}^2 \partial {\sigma_2}^2} \\ \frac{\partial^2 lnL}{\partial \mu_1 \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial \mu_2 \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial {\sigma_1}^2 \partial {\sigma_2}^2} & \frac{\partial^2 lnL}{\partial ({\sigma_2}^2)^2} \end{bmatrix}$$

And $$B_F^{(l)} = \begin{bmatrix} \hat{\alpha}^{(l)} \\ \hat{\mu}_1^{(l)} \\ \hat{\mu}_2^{(l)} \\ \hat{\sigma}_1^{2(l)} \\ \hat{\sigma}_2^{2(l)} \end{bmatrix}$$

7.6: Maximum Likelihood Estimation using Grid search

The likelihood function for the density function in (7.5.1) can be written as

$$L = \prod_{i=1}^{n} \left[\alpha \frac{1}{\sqrt{2\pi}\,\sigma_1} e^{-\frac{1}{2{\sigma_1}^2}(x-\mu_1)^2} + (1-\alpha) \frac{1}{\sqrt{2\pi}\,\sigma_2} e^{-\frac{1}{2{\sigma_2}^2}(x-\mu_2)^2} \right]$$

Where $0 \leq \alpha \leq 1$. The log-likelihood function can be written as

$$\ln L = \sum_{i=1}^{n} \ln \left[\alpha \frac{1}{\sqrt{2\pi}\,\sigma_1} e^{-\frac{1}{2{\sigma_1}^2}(x-\mu_1)^2} + (1-\alpha) \frac{1}{\sqrt{2\pi}\,\sigma_2} e^{-\frac{1}{2{\sigma_2}^2}(x-\mu_2)^2} \right]$$

Where ln stands for natural logarithm. Grid ranges are three standard deviation intervals determined by quick implementation of the EM algorithm method mentioned in Section (7.3). Note that in this range selection process, the following restrictions should be implemented, $0 \leq \alpha \leq 1$, ${\sigma_1}^2 > 0$ and ${\sigma_2}^2 > 0$

7.7: Simulation Study for Parameter Estimation in a Mixture of Two Normal Densities

Here, we present a simulation study for the Skewed Bimodal Distribution (#7 in Table 7.2.1) using both the methods mentioned above, using the normal probability density function (PDF) and the normal cumulative probability distribution function (CDF). Here, true values of the parameters are α = 0.75, $\mu_1 = 0$, $\mu_2 = 0$, ${\sigma_1}^2 = 1$, ${\sigma_2}^2 = (\frac{1}{3})^2$.

Simulations are performed for sample sizes n = 20, n = 50, n = 100, and n = 500. Simulations are conducted for 1000 samples for each sample sizes. Results are presented in Table 7.7.1 to 7.7.4, means of the estimates are represented as M_P, standard deviations are represented as S_P, and means of absolute biases are represented as M_B, standard deviations of the biases are represented as S_B, means of the sum of the log-likelihoods resulted for a set of estimated parameters as M_L, and standard deviations of the sum of the log-likelihoods resulted for a set of estimated parameters as S_L.

Table 7.7.1: EM Algorithm parameter estimates

	α	μ_1	μ_2	${\sigma_1}^2$	${\sigma_2}^2$
	n = 20		$M_L = -27.9339$		$S_L = 4.1494$
M_P	0.4842	0.1042	0.5830	1.0185	0.7255
S_P	0.0614	0.5063	0.5972	0.4141	0.5088
M_B	0.2658	0.4186	0.9289	0.3158	0.6216
S_B	0.0614	0.3030	0.5785	0.2683	0.5000
	n = 50		$M_L = -73.2497$		$S_L = 4.7819$
M_P	0.5001	0.1173	0.6373	1.1065	0.8430
S_P	0.0252	0.3531	0.4600	0.2695	0.4327
M_B	0.2499	0.3186	0.8646	0.2280	0.7331
S_B	0.0252	0.1920	0.4563	0.1787	0.4306
	n = 100		$M_L = -147.8127$		$S_L = 6.8254$
M_P	0.5024	0.1232	0.6293	1.1346	0.8768
S_P	0.0155	0.2981	0.3998	0.2114	0.3804
M_B	0.2476	0.2796	0.8710	0.2046	0.7659
S_B	0.0155	0.1606	0.3990	0.1447	0.3801
	n = 500		$M_L = -750.1973$		$S_L = 13.9221$
M_P	0.5010	0.1526	0.5963	1.2434	0.9593
S_P	0.0038	0.1905	0.2354	0.1018	0.2337
M_B	0.2490	0.2126	0.9037	0.2458	0.8482
S_B	0.0038	0.1198	0.2354	0.0959	0.2337

Table 7.7.2: Newton-Raphson MLE

	α	μ_1	μ_2	${\sigma_1}^2$	${\sigma_2}^2$
	n = 20		$M_L = -46.6694$		$S_L = 4.1919$
M_P	0.5533	–1.4373	– 0.5398	2.6945	2.3052
S_P	0.4043	62.5742	32.0165	31.9023	25.0328
M_B	0.3744	3.0462	2.5265	1.8251	2.1941
S_B	0.2487	62.5165	31.9818	31.8950	25.0328
	n = 50		$M_L = -84.9860$		$S_L = 2.4760$
M_P	0.4560	0.3739	–0.6470	1.4000	7.6320
S_P	0.4004	0.0490	0.4150	0.0579	1.7143
M_B	0.4264	0.6040	2.9292	0.4349	7.5209
S_B	0.2547	0.0488	0.4145	0.0579	1.7143
	n = 100		$M_L = -153.8408$		$S_L = 1.1730$
M_P	0.4368	0.2049	0.2280	1.5413	1.4435
S_P	0.3931	4.7842	4.1122	10.4728	7.6373
M_B	0.4336	0.5329	1.2720	0.5530	1.3324
S_B	0.2539	4.7588	4.1122	10.4722	0.2909
	n = 500		$M_L = -789.4095$		$S_L = 1.1082$
M_P	0.2244	0.3994	–2.3095	1.1980	12.6567
S_P	0.3020	0.0078	0.8298	0.0013	3.5440
M_B	0.5717	0.3994	3.8095	0.1981	12.5456
S_B	0.2013	0.0078	0.8298	0.0013	3.5440

Table 7.7.3: EM Algorithm in Newton-Raphson MLE

	α	μ_1	μ_2	${\sigma_1}^2$	${\sigma_2}^2$
	n = 20		$M_L = -35.4874$		$S_L = -85.4085$
M_P	0.5014	0.1035	0.4168	1.4802	1.4167
S_P	0.0377	5.5980	5.6998	5.1207	4.5231
M_B	0.2508	0.8038	1.4746	0.5964	1.3058
S_B	0.0180	5.5409	5.6112	5.1085	4.5230
	n = 50		$M_L = -96.0071$		$S_L = 3.0980$
M_P	0.5006	0.2512	0.3891	1.5500	1.4490
S_P	0.0342	7.5253	7.1144	5.6433	6.2826
M_B	0.2508	0.9082	1.5285	0.5904	1.3380
S_B	0.0223	7.4745	7.0364	5.6392	6.2826
	n = 100		$M_L = -158.1604$		$S_L = 2.4949$
M_P	0.4998	0.3173	0.4635	1.2064	1.2409
S_P	0.0074	1.7293	2.8566	0.5579	1.6282
M_B	0.2502	0.4267	1.2164	0.2159	1.1298
S_B	0.0074	1.7056	2.7847	0.5525	1.6282
	n = 500		$M_L = -753.9149$		$S_L = 0.1362$
M_P	0.5000	0.3761	0.3761	1.1961	1.1961
S_P	0.0000	0.0509	0.0509	0.0653	0.0653
M_B	0.2500	0.3761	1.1239	0.1961	1.0849
S_B	0.0000	0.0509	0.0509	0.0652	0.0653

Table 7.7.4: Grid search MLE

	α	μ_1	μ_2	${\sigma_1}^2$	${\sigma_2}^2$
	n = 20		$ML = -26.8678$		$S_L = 2.9535$
$\mathbf{M_P}$	0.4615	0.3456	0.4738	0.4430	0.5102
$\mathbf{S_P}$	0.1455	0.9399	0.8609	0.4092	0.4183
$\mathbf{M_B}$	0.2885	0.8694	1.0744	0.6324	0.4242
$\mathbf{S_B}$	0.1455	0.4963	0.7999	0.2788	0.3928
	n = 50		$M_L = -71.1160$		$S_L = 4.3460$
$\mathbf{M_P}$	0.5196	-0.1670	1.0099	0.6412	0.3805
$\mathbf{S_P}$	0.0592	0.5416	0.6453	0.2915	0.2744
$\mathbf{M_B}$	0.2304	0.4867	0.4989	0.4096	0.2734
$\mathbf{S_B}$	0.0592	0.2900	0.6385	0.2142	0.2707
	n = 100		$M_L = -739.8238$		$S_L = 12.8219$
$\mathbf{M_P}$	0.5281	-0.2652	1.1487	0.6994	0.3497
$\mathbf{S_P}$	0.0309	0.3941	0.4935	0.2124	0.2023
$\mathbf{M_B}$	0.2219	0.4202	0.3524	0.3288	0.2386
$\mathbf{S_B}$	0.0309	0.2213	0.4928	0.1713	0.2023
	n = 500		$M_L = -739.8238$		$S_L = 12.8219$
$\mathbf{M_P}$	0.5078	-0.3479	1.1335	0.9400	0.4602
$\mathbf{S_P}$	0.0083	0.1128	0.0398	0.0223	0.1029
$\mathbf{M_B}$	0.2422	0.3479	0.3665	0.0630	0.3491
$\mathbf{S_B}$	0.0083	0.1128	0.0398	0.0112	0.1029

In Tables 7.7.2 and 7.7.3 we noticed that estimates can be highly volatile as the implementations of systems of nonlinear equations using Newton-Raphson algorithm. In Tables 7.7.1 and 7.7.4 we noticed that all the estimates are consistent as the standard deviations of the estimates are lower for higher sample sizes. Grid search method resulted slightly higher means of sum of log-likelihood values indicating that if EM algorithm is implemented with higher convergence will suffice to get optimal estimate possible using maximizing the likelihood procedure. In all Tables 7.7.1 to 7.7.4 we noticed that the estimates are not asymptotically unbiased as expected. In absence of a better method, the EM algorithm method can conveniently be used.

Chapter 8: Conclusion and Future Studies

Since density estimation is a very important field in statistics, there is always new ways to improve the techniques of density estimation. This paper has only scratched the surface, since there are many kernels and this paper has only looked at four. These techniques could be used to analyze all of the other kernels as well. Another option with these four kernels is to analyze different normal family density functions as this was done in Marron and Wand (1992). Also, another iterative method may be explored to calculate the exact values of the Epanechnikov kernel that would decrease the computational time. Finally, the Epanechnikov kernel can be examined to see if there is a more accurate way of getting an approximate and exact estimate of the smoothing parameter, which in turn will give more desirable AMISE values.

In all four parameter estimation procedures, the estimates are not as accurate as desired as the biases and standard errors are high. In considering convenience EM algorithm method would serve the purpose. In parameter estimation in mixtures of normal densities, there is a wide opportunity for improvement. The matter of fact is that in estimation more than two parameters using maximum likelihood estimation process when system of equations are non-linear has ample scope of further study.

Appendix A: MATLAB Code Histogram, Kernel Functions and Bandwidth

Histogram Code:

```
vals= randn(1,1000)*100;
hist(vals,100)
hist(vals,10)
```

R Code Histogram:

```
library(MASS)
df<-geyser
str(df)
attach(df)
hist(duration)
ggplot(geyser)+geom_histogram(aes(x=duration),binwidth=0.
5,fill="grey",color="red")
ggplot(geyser)+geom_histogram(aes(x=duration),bandwidth=0
.05,fill="grey",color="red")
```

Kernel Density Code:

```
SixMPG = [13;15;23;29;32;34];

histogram(SixMPG)

pdSix = fitdist(SixMPG,'Kernel','BandWidth',4);
x = 0:.1:45;
ySix = pdf(pdSix,x);
plot(x,ySix,'k-','LineWidth',2)

% Plot each individual pdf and scale its appearance on the plot
hold on
for i=1:6
    pd = makedist('Normal','mu',SixMPG(i),'sigma',4);
    y = pdf(pd,x);
    y = y/6;
    plot(x,y,'b:')
end
hold off
```

Kernel Functions:

```
hname = {'normal' 'epanechnikov' 'box' 'triangle'};
colors = {'r' 'b' 'g' 'm'};
lines = {'-','-.','--',':'};

% Generate a sample of each kernel smoothing function and plot
data = [0];
figure
for j=1:4
    pd = fitdist(data,'kernel','Kernel',hname{j});
    x = -3:.1:3;
    y = pdf(pd,x);
    plot(x,y,'Color',colors{j},'LineStyle',lines{j})
    hold on
end
legend(hname)
hold off
```

Bandwidth of Kernel:

```
load carbig
% Set plot specifications
hname = {'normal' 'epanechnikov' 'box' 'triangle'};
colors = {'r' 'b' 'g' 'm'};
lines = {'-','-.','--',':'};

% Generate kernel distribution objects and plot
figure
for j=1:4
    pd = fitdist(MPG,'kernel','Kernel',hname{j});
    x = -10:1:60;
    y = pdf(pd,x);
    plot(x,y,'Color',colors{j},'LineStyle',lines{j})
    hold on
end
legend(hname)
hold off


%Bandwidth

pd1 = fitdist(MPG,'kernel');
pd2 = fitdist(MPG,'kernel','BandWidth',2);
pd3 = fitdist(MPG,'kernel','BandWidth',10);

% Compute each pdf
x = -10:1:60;
y1 = pdf(pd1,x);
y2 = pdf(pd2,x);


y3 = pdf(pd3,x);
```

```
% Plot each pdf
plot(x,y1,'Color','r','LineStyle','-')
hold on
plot(x,y2,'Color','k','LineStyle',':')
plot(x,y3,'Color','b','LineStyle','--')
legend({'BandWidth = Default','BandWidth = 2','BandWidth = 10'})
hold off
```

Appendix B: MATLAB Code for Substitution Method

Normal Density Method:

```
clc;
clear all;

%Gaussian Kernel (Sample Size 20)
 for k=1:1000
 n=20;
 m=1000;
 x=normrnd(0,1,n,1);
 x=sort(x);
 xl=mean(x)-4*std(x);
 xu=mean(x)+4*std(x);
 y=[xl:(xu-xl)/m:xu];
 %starting h
 h=(4/(3*n))^0.2;
 H0=h;
 AMISE=3*h^4/(32*sqrt(pi))+1/(2*n*h*sqrt(pi));
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
        cc=exp(-0.5*(y(j)-x(i))^2/h^2);
         s=s+cc;
     end
     f(j)=(1/sqrt(2*pi))*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
 f=f/fs;
 fh=f;
 ft=normpdf(y,0,1);
```

```
fl=normpdf(y,mean(x),sqrt((n-1)*var(x)/n));
 d1=0;
 d2=0;
 d3=0;
 for i=1:n
     for j=1:n
         ex=exp(-(x(i)-x(j))^2/(4*h^2));
         d1=d1+ex;
         d2=d2+(x(i)-x(j))^2*ex;
         d3=d3+(x(i)-x(j))^4*ex;
     end
 end
 Df2=3/(8*n^2*h^9*sqrt(pi))*(h^4*d1-h^2*d2+d3/12);
 ha=(1/(2*sqrt(pi)))^0.2*Df2^(-0.2)*n^(-0.2);
 h=ha;
 H(k)=ha;
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
        cc=exp(-0.5*(y(j)-x(i))^2/h^2);
         s=s+cc;
     end
     f(j)=(1/sqrt(2*pi))*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
 f=f/fs;
 fa=f;

 mfh(k)=sum((fh-ft).^2);
 mfa(k)=sum((fa-ft).^2);
 mfl(k)=sum((fl-ft).^2);
 end
 [H0 mean(H) std(H)]
 [mean(mfh) mean(mfa) mean(mfl)]
 [std(mfh) std(mfa) std(mfl)]

 %Gaussian Kernel (Sample Size 50)

 for k=1:1000
 n=50;
 m=1000;
 x=normrnd(0,1,n,1);
 x=sort(x);
 xl=mean(x)-4*std(x);
 xu=mean(x)+4*std(x);
 y=[xl:(xu-xl)/m:xu];
 %starting h
 h=(4/(3*n))^0.2;
 H0=h;
 AMISE=3*h^4/(32*sqrt(pi))+1/(2*n*h*sqrt(pi));
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
```

```
        cc=exp(-0.5*(y(j)-x(i))^2/h^2);
         s=s+cc;
    end
    f(j)=(1/sqrt(2*pi))*s/(n*h);
    fs=fs+f(j)*(y(2)-y(1));
end
f=f/fs;
fh=f;
ft=normpdf(y,0,1);
fl=normpdf(y,mean(x),sqrt((n-1)*var(x)/n));
d1=0;
d2=0;
d3=0;
for i=1:n
    for j=1:n
        ex=exp(-(x(i)-x(j))^2/(4*h^2));
        d1=d1+ex;
        d2=d2+(x(i)-x(j))^2*ex;
        d3=d3+(x(i)-x(j))^4*ex;
    end
end
Df2=3/(8*n^2*h^9*sqrt(pi))*(h^4*d1-h^2*d2+d3/12);
ha=(1/(2*sqrt(pi)))^0.2*Df2^(-0.2)*n^(-0.2);
h=ha;
H(k)=ha;
fs=0;
for j=1:m+1
    s=0;
    for i=1:n
       cc=exp(-0.5*(y(j)-x(i))^2/h^2);
        s=s+cc;
    end
    f(j)=(1/sqrt(2*pi))*s/(n*h);
    fs=fs+f(j)*(y(2)-y(1));
end
f=f/fs;
fa=f;

mfh(k)=sum((fh-ft).^2);
mfa(k)=sum((fa-ft).^2);
mfl(k)=sum((fl-ft).^2);
end
[H0 mean(H) std(H)]
[mean(mfh) mean(mfa) mean(mfl)]
[std(mfh) std(mfa) std(mfl)]


%Epanechnikov Kernel (Sample Size 20)

for k=1:1000
n=20;
m=1000;
x=normrnd(0,1,n,1);
x=sort(x);
xl=mean(x)-4*std(x);
```

```
xu=mean(x)+4*std(x);
 y=[xl:(xu-xl)/m:xu];

 %starting h

 h=(40*sqrt(pi)/n)^0.2;
 H0=h;
 AMISE=3*h^4/(800*sqrt(pi))+3/(5*n*h);
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
         if (y(j)-x(i))^2/h^2 < 1
             cc=1-(y(j)-x(i))^2/h^2;
         else
             cc=0;
         end
         s=s+cc;
     end
     s=n-s;
     f(j)=0.75*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
 f=f/fs;
fh=f;
 ft=normpdf(y,0,1);
 fl=normpdf(y,mean(x),sqrt((n-1)*var(x)/n));
 Df2=45/(2*h^6);
 ha=(0.2)^(-0.4)*(0.6)^0.2*Df2^(-0.2)*n^(-0.2);
 h=ha;
 H(k)=ha;
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
         if (y(j)-x(i))^2/h^2 < 1
             cc=1-(y(j)-x(i))^2/h^2;
         else
             cc=0;
         end
         s=s+cc;
     end

     f(j)=0.75*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
 f=f/fs;
 fa=f;

 mfh(k)=sum((fh-ft).^2);
 mfa(k)=sum((fa-ft).^2);
 mfl(k)=sum((fl-ft).^2);
 end
 [H0 mean(H) std(H)]
 [mean(mfh) mean(mfa) mean(mfl)]
```

```
[std(mfh) std(mfa) std(mfl)]

%Epanechnikov Kernel(Sample Size 50)

 for k=1:1000
 n=50;
 m=1000;
 x=normrnd(0,1,n,1);
 x=sort(x);
 xl=mean(x)-4*std(x);
 xu=mean(x)+4*std(x);
 y=[xl:(xu-xl)/m:xu];
 %starting h
 h=(40*sqrt(pi)/n)^0.2;
 H0=h;
 AMISE=3*h^4/(800*sqrt(pi))+3/(5*n*h);
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
         if (y(j)-x(i))^2/h^2 < 1
             cc=1-(y(j)-x(i))^2/h^2;
         else
             cc=0;
         end
         s=s+cc;
     end
     s=n-s;
     f(j)=0.75*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
 f=f/fs;
fh=f;
 ft=normpdf(y,0,1);
 fl=normpdf(y,mean(x),sqrt((n-1)*var(x)/n));
 Df2=45/(2*h^6);
 ha=(0.2)^(-0.4)*(0.6)^0.2*Df2^(-0.2)*n^(-0.2);
 h=ha;
 H(k)=ha;
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
         if (y(j)-x(i))^2/h^2 < 1
             cc=1-(y(j)-x(i))^2/h^2;
         else
             cc=0;
         end
         s=s+cc;
     end

     f(j)=0.75*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
 f=f/fs;
```

```
 fa=f;

 mfh(k)=sum((fh-ft).^2);
 mfa(k)=sum((fa-ft).^2);
 mfl(k)=sum((fl-ft).^2);
 end
 [H0 mean(H) std(H)]
 [mean(mfh) mean(mfa) mean(mfl)]
 [std(mfh) std(mfa) std(mfl)]



%Quadratic Kernel (Sample Size 20)

for k=1:1000
    n=20;
    m=1000;
    x=exprnd(1,n,1);
    x=normrnd(0,1,n,1);
    x=sort(x);
    xl=mean(x)-4*std(x);
    xu=mean(x)+4*std(x);
    y=[xl:(xu-xl)/m:xu];
    s=std(x);
    x1=sum(x);
    x2=sum(x.^2);

    %starting h

    h0(k)=1.06*s*n^(-0.2);
    part1=(xu-xl)*16/h0(k)^6;
    part2=(n*xu^3/3-xu^2*x1+xu*x2-n/3*xl^3+xl^2*x1-
xl*x2)*96/(n*h0(k)^8);
    part3=n^2/5*xu^5+4/3*x1^2*xu^3+xu*x2^2-n*x1*xu^4-
2*x1*x2*xu^2+2/3*n*x2*xu^3;
    part3=(part3-n^2/5*xl^5-
4/3*x1^2*xl^3+xl*x2^2+n*x1*xl^4+2*x1*x2*xl^2-2/3*n*x2*xl^3);
    part3=part3*144/(n^2*h0(k)^(10));
    der2=part1-part2+part3;
    h1(k)=(1/7)^(-0.4)*(5/7)^(0.2)*der2^(-0.2)*n^(-0.2);
    part1=(xu-xl)*16/h1(k)^6;
    part2=(n*xu^3/3-xu^2*x1+xu*x2-n/3*xl^3+xl^2*x1-
xl*x2)*96/(n*h1(k)^8);
    part3=n^2/5*xu^5+4/3*x1^2*xu^3+xu*x2^2-n*x1*xu^4-
2*x1*x2*xu^2+2/3*n*x2*xu^3;
    part3=(part3-n^2/5*xl^5-
4/3*x1^2*xl^3+xl*x2^2+n*x1*xl^4+2*x1*x2*xl^2-2/3*n*x2*xl^3);
    part3=part3*144/(n^2*h1(k)^(10));
    der2=part1-part2+part3;
    h2(k)=(1/7)^(-0.4)*(5/7)^(0.2)*der2^(-0.2)*n^(-0.2);
    part1=(xu-xl)*16/h2(k)^6;
    part2=(n*xu^3/3-xu^2*x1+xu*x2-n/3*xl^3+xl^2*x1-
xl*x2)*96/(n*h2(k)^8);
```

```
    part3=n^2/5*xu^5+4/3*x1^2*xu^3+xu*x2^2-n*x1*xu^4-
2*x1*x2*xu^2+2/3*n*x2*xu^3;
    part3=(part3-n^2/5*xl^5-
4/3*x1^2*xl^3+xl*x2^2+n*x1*xl^4+2*x1*x2*xl^2-2/3*n*x2*xl^3);
    part3=part3*144/(n^2*h2(k)^(10));
    der2=part1-part2+part3;
    h3(k)=(1/7)^(-0.4)*(5/7)^(0.2)*der2^(-0.2)*n^(-0.2);
    f0s=0;
    f1s=0;
    f2s=0;
    f3s=0;
    for j=1:m+1
        s0=0;
        for i=1:n
            if (y(j)-x(i))^2/h0(k)^2 < 1
                cc0=(1-(y(j)-x(i))^2/h0(k)^2)^2;
            else
                cc0=0;
            end
            s0=s0+cc0;
        end
        s1=0;
        for i=1:n
            if (y(j)-x(i))^2/h1(k)^2 < 1
                cc1=(1-(y(j)-x(i))^2/h1(k)^2)^2;
            else
                cc1=0;
            end
            s1=s1+cc1;
        end
        s2=0;
        for i=1:n
            if (y(j)-x(i))^2/h2(k)^2 < 1
                cc2=(1-(y(j)-x(i))^2/h2(k)^2)^2;
            else
                cc2=0;
            end
            s2=s2+cc2;
        end
        s3=0;
        for i=1:n
            if (y(j)-x(i))^2/h3(k)^2 < 1
                cc3=(1-(y(j)-x(i))^2/h3(k)^2)^2;
            else
                cc3=0;
            end
            s3=s3+cc3;
        end
        %     s=n-s;
        f0(j)=0.75*s0/(n*h0(k));
        f1(j)=0.75*s1/(n*h1(k));
        f2(j)=0.75*s2/(n*h2(k));
        f3(j)=0.75*s3/(n*h3(k));
        f0s=f0s+f0(j)*(y(2)-y(1));
        f1s=f1s+f1(j)*(y(2)-y(1));
```

```
        f2s=f2s+f2(j)*(y(2)-y(1));
        f3s=f3s+f3(j)*(y(2)-y(1));
    end
    f0=f0/f0s;
    f1=f1/f1s;
    f2=f2/f2s;
    f3=f3/f3s;
    ft=exppdf(y,1);
    ft=normpdf(y,0,1);
    fl=exppdf(y,mean(x));
    fl=normpdf(y,mean(x),std(x));



    mf0(k)=sum((f0-ft).^2);
    mf1(k)=sum((f1-ft).^2);
    mf2(k)=sum((f2-ft).^2);
    mf3(k)=sum((f3-ft).^2);
    mfl(k)=sum((fl-ft).^2);
end
[mean(h0) mean(h1) mean(h2) mean(h3)]
[std(h0) std(h1) std(h2) std(h3)]
[mean(mf0) mean(mf1) mean(mf2) mean(mf3) mean(mfl)]
[std(mf0) std(mf1) std(mf2) std(mf3) std(mfl)]

%Triweight Kernel (Sample Size 20)

for k=1:1000
    n=20;
    m=1000;
    x=exprnd(1,n,1);
    x=normrnd(0,1,n,1);
    x=sort(x);
    xl=mean(x)-4*std(x);
    xu=mean(x)+4*std(x);
    y=[xl:(xu-xl)/m:xu];
    s=std(x);
    x1=sum(x);
    x2=sum(x.^2);
    %starting h
    h0(k)=1.06*s*n^(-0.2);
    part1=(xu-xl)*16/h0(k)^6;
    part2=(n*xu^3/3-xu^2*x1+xu*x2-n/3*xl^3+xl^2*x1-
xl*x2)*96/(n*h0(k)^8);
    part3=n^2/5*xu^5+4/3*x1^2*xu^3+xu*x2^2-n*x1*xu^4-
2*x1*x2*xu^2+2/3*n*x2*xu^3;
    part3=(part3-n^2/5*xl^5-
4/3*x1^2*xl^3+xl*x2^2+n*x1*xl^4+2*x1*x2*xl^2-2/3*n*x2*xl^3);
    part3=part3*144/(n^2*h0(k)^(10));
    der2=part1-part2+part3;
    h1(k)=(1/7)^(-0.4)*(5/7)^(0.2)*der2^(-0.2)*n^(-0.2);
    part1=(xu-xl)*16/h1(k)^6;
    part2=(n*xu^3/3-xu^2*x1+xu*x2-n/3*xl^3+xl^2*x1-
xl*x2)*96/(n*h1(k)^8);
```

```
    part3=n^2/5*xu^5+4/3*x1^2*xu^3+xu*x2^2-n*x1*xu^4-
2*x1*x2*xu^2+2/3*n*x2*xu^3;
    part3=(part3-n^2/5*xl^5-
4/3*x1^2*xl^3+xl*x2^2+n*x1*xl^4+2*x1*x2*xl^2-2/3*n*x2*xl^3);
    part3=part3*144/(n^2*h1(k)^(10));
    der2=part1-part2+part3;
    h2(k)=(1/7)^(-0.4)*(5/7)^(0.2)*der2^(-0.2)*n^(-0.2);
    part1=(xu-xl)*16/h2(k)^6;
    part2=(n*xu^3/3-xu^2*x1+xu*x2-n/3*xl^3+xl^2*x1-
xl*x2)*96/(n*h2(k)^8);
    part3=n^2/5*xu^5+4/3*x1^2*xu^3+xu*x2^2-n*x1*xu^4-
2*x1*x2*xu^2+2/3*n*x2*xu^3;
    part3=(part3-n^2/5*xl^5-
4/3*x1^2*xl^3+xl*x2^2+n*x1*xl^4+2*x1*x2*xl^2-2/3*n*x2*xl^3);
    part3=part3*144/(n^2*h2(k)^(10));
    der2=part1-part2+part3;
    h3(k)=(1/7)^(-0.4)*(5/7)^(0.2)*der2^(-0.2)*n^(-0.2);
    f0s=0;
    f1s=0;
    f2s=0;
    f3s=0;
    for j=1:m+1
        s0=0;
        for i=1:n
            if (y(j)-x(i))^2/h0(k)^2 < 1
                cc0=(1-(y(j)-x(i))^2/h0(k)^2)^2;
            else
                cc0=0;
            end
            s0=s0+cc0;
        end
        s1=0;
        for i=1:n
            if (y(j)-x(i))^2/h1(k)^2 < 1
                cc1=(1-(y(j)-x(i))^2/h1(k)^2)^2;
            else
                cc1=0;
            end
            s1=s1+cc1;
        end
        s2=0;
        for i=1:n
            if (y(j)-x(i))^2/h2(k)^2 < 1
                cc2=(1-(y(j)-x(i))^2/h2(k)^2)^2;
            else
                cc2=0;
            end
            s2=s2+cc2;
        end
        s3=0;
        for i=1:n
            if (y(j)-x(i))^2/h3(k)^2 < 1
                cc3=(1-(y(j)-x(i))^2/h3(k)^2)^2;
            else
                cc3=0;
```

```
            end
            s3=s3+cc3;
        end
            s=n-s;
        f0(j)=0.75*s0/(n*h0(k));
        f1(j)=0.75*s1/(n*h1(k));
        f2(j)=0.75*s2/(n*h2(k));
        f3(j)=0.75*s3/(n*h3(k));
        f0s=f0s+f0(j)*(y(2)-y(1));
        f1s=f1s+f1(j)*(y(2)-y(1));
        f2s=f2s+f2(j)*(y(2)-y(1));
        f3s=f3s+f3(j)*(y(2)-y(1));
    end
    f0=f0/f0s;
    f1=f1/f1s;
    f2=f2/f2s;
    f3=f3/f3s;
     ft=exppdf(y,1);
    ft=normpdf(y,0,1);
     fl=exppdf(y,mean(x));
    fl=normpdf(y,mean(x),std(x));



    mf0(k)=sum((f0-ft).^2);
    mf1(k)=sum((f1-ft).^2);
    mf2(k)=sum((f2-ft).^2);
    mf3(k)=sum((f3-ft).^2);
    mfl(k)=sum((fl-ft).^2);
end
[mean(h0) mean(h1) mean(h2) mean(h3)]
[std(h0) std(h1) std(h2) std(h3)]
[mean(mf0) mean(mf1) mean(mf2) mean(mf3) mean(mfl)]
[std(mf0) std(mf1) std(mf2) std(mf3) std(mfl)]
```

Adaptive Method:

```
clc;
clear all;

% Gaussian Kernel

for k=1:1000
 n=20;
 m=1000;
 x=normrnd(0,1,n,1);
 x=sort(x);
 xl=mean(x)-4*std(x);
 xu=mean(x)+4*std(x);
 y=[xl:(xu-xl)/m:xu];
```

```
%starting h
h=(4/(3*n))^0.2;
H0=h;
AMISE=3*h^4/(32*sqrt(pi))+1/(2*n*h*sqrt(pi));
fs=0;
for j=1:m+1
    s=0;
    for i=1:n
       cc=exp(-0.5*(y(j)-x(i))^2/h^2);
        s=s+cc;
    end
    f(j)=(1/sqrt(2*pi))*s/(n*h);
    fs=fs+f(j)*(y(2)-y(1));
end
f=f/fs;
fh=f;
ft=normpdf(y,0,1);
fl=normpdf(y,mean(x),sqrt((n-1)*var(x)/n));
d1=0;
d2=0;
d3=0;
for i=1:n
    for j=1:n
        ex=exp(-(x(i)-x(j))^2/(4*h^2));
        d1=d1+ex;
        d2=d2+(x(i)-x(j))^2*ex;
        d3=d3+(x(i)-x(j))^4*ex;
    end
end
Df2=3/(8*n^2*h^9*sqrt(pi))*(h^4*d1-h^2*d2+d3/12);
ha=(1/(2*sqrt(pi)))^0.2*Df2^(-0.2)*n^(-0.2);
h=ha;
H(k)=ha;
fs=0;
for j=1:m+1
    s=0;
    for i=1:n
       cc=exp(-0.5*(y(j)-x(i))^2/h^2);
        s=s+cc;
    end
    f(j)=(1/sqrt(2*pi))*s/(n*h);
    fs=fs+f(j)*(y(2)-y(1));
end
f=f/fs;
fa=f;

mfh(k)=sum((fh-ft).^2);
mfa(k)=sum((fa-ft).^2);
mfl(k)=sum((fl-ft).^2);
end
[H0 mean(H) std(H)]
[mean(mfh) mean(mfa) mean(mfl)]
[std(mfh) std(mfa) std(mfl)]
```

```
% Epanechnikov Kernel

for k=1:1000
 n=20;
 m=1000;
 x=normrnd(0,1,n,1);
 x=sort(x);
 xl=mean(x)-4*std(x);
 xu=mean(x)+4*std(x);
 y=[xl:(xu-xl)/m:xu];

 %starting h

 h=(4/(3*n))^0.2;
 H0=h;
 AMISE=3*h^4/(32*sqrt(pi))+1/(2*n*h*sqrt(pi));
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
        cc=exp(-0.5*(y(j)-x(i))^2/h^2);
         s=s+cc;
     end
     f(j)=(1/sqrt(2*pi))*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
 f=f/fs;
 fh=f;
 ft=normpdf(y,0,1);
 fl=normpdf(y,mean(x),sqrt((n-1)*var(x)/n));
 d1=0;
 d2=0;
 d3=0;
 for i=1:n
     for j=1:n
         ex=exp(-(x(i)-x(j))^2/(4*h^2));
         d1=d1+ex;
         d2=d2+(x(i)-x(j))^2*ex;
         d3=d3+(x(i)-x(j))^4*ex;
     end
 end
 Df2=3/(8*n^2*h^9*sqrt(pi))*(h^4*d1-h^2*d2+d3/12);
 ha=(1/(2*sqrt(pi)))^0.2*Df2^(-0.2)*n^(-0.2);
 h=ha;
 H(k)=ha;
 fs=0;
 for j=1:m+1
     s=0;
     for i=1:n
        cc=exp(-0.5*(y(j)-x(i))^2/h^2);
         s=s+cc;
     end
     f(j)=(1/sqrt(2*pi))*s/(n*h);
     fs=fs+f(j)*(y(2)-y(1));
 end
```

```
f=f/fs;
fa=f;

mfh(k)=sum((fh-ft).^2);
mfa(k)=sum((fa-ft).^2);
mfl(k)=sum((fl-ft).^2);
end
[H0 mean(H) std(H)]
[mean(mfh) mean(mfa) mean(mfl)]
[std(mfh) std(mfa) std(mfl)]



%Quartic Kernel

h3=(280*sqrt(pi)/(3*n))^0.2*std(x);
sx1=sum(x);
sx2=sum(x.^2);
I3=16*(xu-xl)/h3^6-96/(n*h3^8)*(n/3*(xu^3-xl^3)-(xu^2-xl^2)*sx1+(xu-
xl)*sx2);
II=(xu^5-xl^5)*n^2/5-n*(xu^4-xl^4)*sx1+(xu^3-xl^3)*(n*sx2-
2*sx1^2)*2/3;
II=II-2*(xu^2-xl^2)*sx1*sx2+(xu-xl)*sx2^2;
I3=I3+144/(n^2*h3^(10))*II;
h3=(1/7)^(-0.4)*(5/7)^0.2*I3^(-0.2)*n^(-0.2);
fs3=0;
for j=1:m+1
    s3=0;
    for i=1:n
        if (y(j)-x(i))^2/h3^2 < 1
            cc3=(1-(y(j)-x(i))^2/h3^2)^2;
        else
            cc3=0;
        end
        s3=s3+cc3;
    end
    f3(j)=(15/16)*s3/(n*h3);
    fs3=fs3+f3(j)*(y(2)-y(1));
end
f3=f3/fs3;
H3(k)=h3;
I3=16*(xu-xl)/h3^6-96/(n*h3^8)*(n/3*(xu^3-xl^3)-(xu^2-xl^2)*sx1+(xu-
xl)*sx2);
II=(xu^5-xl^5)*n^2/5-n*(xu^4-xl^4)*sx1+(xu^3-xl^3)*(n*sx2-
2*sx1^2)*2/3;
II=II-2*(xu^2-xl^2)*sx1*sx2+(xu-xl)*sx2^2;
I3=I3+144/(n^2*h3^(10))*II;
A3(k)=0.25*h3^4*(1/7)^2*I3+5/(7*n*h3);

%Triweight Kernel

sx3=sum(x.^3);
sx4=sum(x.^4);
```

```
h4=(25200*sqrt(pi)/(143*n))^0.2*std(x);
D=105/(16*n*h4^3);
C0=-n+sx2*6/h4^2-sx4*5/h4^4;
C1=-sx1*12/h4^2+sx3*20/h4^4;
C2=6*n/h4^2-sx2*30/h4^4;
C3=sx1*20/h4^4;
C4=-5*n/h4^4;
II=C4^2/9*(xu^9-xl^9)+(xu^8-xl^8)*C3*C4/4+(xu^7-
xl^7)*(2*C2*C4+C3^2)/7;
II=II+(xu^6-xl^6)*(C1*C4+C2*C3)/3+(xu^5-
xl^5)*(2*C0*C4+2*C1*C3+C2^2)/5;
II=II+(xu^4-xl^4)*(C0*C3+C1*C2)/2+(xu^3-xl^3)*(2*C0*C2+C1^2)/3;
II=II+(xu^2-xl^2)*C0*C1+(xu-xl)*C0^2;
I4=D^2*II;
h4=(1/9)^(-0.4)*(350/429)^0.2*I4^(-0.2)*n^(-0.2);


fs4=0;
for j=1:m+1
    s4=0;
    for i=1:n
        if (y(j)-x(i))^2/h4^2 < 1
            cc4=(1-(y(j)-x(i))^2/h4^2)^3;
        else
            cc4=0;
        end
        s4=s4+cc4;
    end
    f4(j)=(35/32)*s4/(n*h4);
    fs4=fs4+f4(j)*(y(2)-y(1));
end
f4=f4/fs4;
H4(k)=h4;
D=105/(16*n*h4^3);
C0=-n+sx2*6/h4^2-sx4*5/h4^4;
C1=-sx1*12/h4^2+sx3*20/h4^4;
C2=6*n/h4^2-sx2*30/h4^4;
C3=sx1*20/h4^4;
C4=-5*n/h4^4;
II=C4^2/9*(xu^9-xl^9)+(xu^8-xl^8)*C3*C4/4+(xu^7-
xl^7)*(2*C2*C4+C3^2)/7;
II=II+(xu^6-xl^6)*(C1*C4+C2*C3)/3+(xu^5-
xl^5)*(2*C0*C4+2*C1*C3+C2^2)/5;
II=II+(xu^4-xl^4)*(C0*C3+C1*C2)/2+(xu^3-xl^3)*(2*C0*C2+C1^2)/3;
II=II+(xu^2-xl^2)*C0*C1+(xu-xl)*C0^2;
I4=D^2*II;
A4(k)=0.25*h4^4*(1/9)^2*I4+350/(429*n*h4);

mfh1(k)=sum((f1-ft).^2);
mfh2(k)=sum((f2-ft).^2);
mfh3(k)=sum((f3-ft).^2);
mfh4(k)=sum((f4-ft).^2);
mfl(k)=sum((fl-ft).^2);
end
[mean(H1) mean(H2) mean(H3) mean(H4)]
```

```
[std(H1) std(H2) std(H3) std(H4)]
mean(A1)
std(A1)
mean(A2)
std(A2)
mean(A3)
std(A3)
mean(A4)
std(A4)
AM=h^4/(sqrt(pi)*1^5)*3/800+3/(5*n*h)
[mean(mfh1) mean(mfh2) mean(mfh3) mean(mfh4) mean(mfl)]
[std(mfh1) std(mfh2) std(mfh3) std(mfh4) std(mfl)]
```

Appendix C: MATLAB Code for Application Data

```
x=[   67.2
      82.5
      66.7
      93.0
      82.6
      75.4
      73.6
      81.4
      99.4
      67.7
      100.7
      72.9
      85.0
      85.7
      126.0
      74.5
      74.5
      94.0
      92.8
      105.5
      75.5
      126.5
      70.0
      98.0
      104.7
      67.8
```

```
      99.3
      91.1
      74.5
      95.5
      79.5
      69.1
      105.5
      78.8
      85.7
      92.8
      72.7
      75.9
      68.6
      68.7];

n=length(x)
m=mean(x);
s=std(x);
mn=min(x)
mx=max(x)
x=sort(x);
x';
%Finding hahat and Gaussian Kernel
gha= s*(4/(3*n))^0.2;
h0=gha;
for p=1:100
    s1=0;
    s2=0;
    s3=0;
    for i=1:n
        for j=1:n
            c1=(x(i)-x(j))^2;
            c2=exp(-c1/(4*h0^2));
            s1=s1+c2;
            s2=s2+c1*c2;
            s3=s3+c1^2*c2;
        end
    end
    c3=3/(8*n^2*h0^9*sqrt(pi));
    f22=c3*(h0^4*s1-h0^2*s2+s3/12);
    h0=(1/(n*f22*2*sqrt(pi)))^0.2;
end
gha
ghahat=h0
gAMISEhat=h0^4/4*f22+1/(2*n*h0*sqrt(pi))
ghhat=h0

%Finding hahat and AMISE for Epaechnikov Kernel

eha=s*(40*sqrt(pi)/n)^0.2;
for k=1:1
    h0=eha;
    for l=1:1
        h0=(2*h0^6/(3*n))^0.2
    end
```

```
end
eha
ehahat=h0
eAMISEhat=h0^4/4/25*(45/(2*h0^6))+ 3/(5*n*h0)
%
% %Graphing
% %Bar Graph Grouping
c1=0;
c2=0;
c3=0;
c4=0;
c5=0;
c6=0;
c7=0;
c8=0;
for i=1:n
    if x(i)<= 65
        c1=c1+1;
    elseif x(i)<=75 & x(i)>65
        c2=c2+1;
    elseif x(i)<=85 & x(i)>75
        c3=c3+1;
    elseif x(i)<=95 & x(i)>85
        c4=c4+1;
    elseif x(i)<=105 & x(i)>95
         c5=c5+1;
    elseif x(i)<=115 & x(i)>105
         c6=c6+1;
    elseif x(i)<=125 & x(i)>115
         c7=c7+1;
    else
        c8=c8+1;
    end
end
bwidth=10;
yb=[c1/n/10  c2/n/10  c3/n/10  c4/n/10 c5/n/10 c6/n/10 c7/n/10
c8/n/10];
xb=[60 70 80 90 100 110 120 130];


%Sample X for 4 Standard Deviations
N=1000;
for i=1:N+1
    y(i)=m-4*s+(8*s)/N*(i-1);
end
y=y';

%Gaussian Function fhat for initial h, h and ha

for i= 1:N+1
    fgha=0;
    fghah=0;
    fghh=0;
    for j=1:n
        fgha=fgha+1/sqrt(2*pi)*exp(-((y(i)-x(j))/gha)^2/2);
```

```
        fghah=fghah+1/sqrt(2*pi)*exp(-((y(i)-x(j))/ghahat)^2/2);
        fghh=fghh+1/sqrt(2*pi)*exp(-((y(i)-x(j))/ghhat)^2/2);
    end
    fhgha(i)=fgha/n/gha;
    fhghah(i)=fghah/n/ghahat;
    fhghh(i)=fghh/n/ghhat;
end

%Epanechnikov Function fhat for initial h,h and ha
ehhat=h0;
F1=0;
c=0;
for i=1:N+1
    s=0;
    for j=1:n
        if (y(i)- x(j))^2/eha^2<1
            c=(y(i)-x(j))^2/eha^2;
        end
        s=s+c;
    end
    s=n-s;
    fheha(i)=s*3/4/n/eha;
    F1=F1+fheha(i)*(y(2)-y(1));
end
F2=0;
c=0;
for i=1:N+1
    s=0;
    for j=1:n
        if (y(i)-x(j))^2/ehahat^2<1
            c=(y(i)-x(j))^2/ehahat^2;
        end
        s=s+c;
    end
    s=n-s;
    fhehah(i)=s*3/4/n/ehahat;
    F2=F2+fhehah(i)*(y(2)-y(1));
end
F3=0;
c=0;
for i=1:N+1
    s=0;
    for j=1:n
        if(y(i)-x(j))^2/ehahat^2<1
            c=(y(i)-x(j))^2/ehahat^2;
        end
        s=s+c;
    end
    s=n-s;
    fhehah(i)=s*3/4/n/ehahat;
    F2=F2+fhehah(i)*(y(2)-y(1));
end
F3=0;
c=0;
for i=1:N+1
```

```
    s=0;
    for j=1:n
        if (y(i)-x(j))^2/ehhat^2<1
            c=(y(i)-x(j))^2/ehhat^2;
        end
        s=s+c;
    end
    s=n-s;
    fhehh(i)=s*3/4/n/ehhat;
    F3=F3+fhehh(i)*(y(2)-y(1));
end
fheha=fheha/F1;
fhehah=fhehah/F2;
fhehh=fhehh/F3;



%Graphs

bar(xb,yb,1,'w')
hold on
plot(y,fhgha,'r-')
hold on
plot(y,fhghah,'b.')
hold on
%plot(y,fhghh,'r-.')
legend('Data','h','hahat')

pause

bar(xb,yb,1,'w')
hold on
plot(y,fheha,'r-')
hold on
plot(y,fhehah,'b.')
hold on
%plot(y,fhehh,'r-.')
legend('Data','h','hahat')
```

Appendix D: MATLAB Code for Parameter Estimations

Mixture of Normal Distributions Graphs:

```
 x=[1:6000]/1000-3;


%#1(Skewed Unimodal)

y=0.2*normpdf(x,0,1)+0.2*normpdf(x,0.5,2/3)+0.6*normpdf(x,13/12,5/9);
plot(x,y)

%#2(Strongly Skewed)

p=length(x);
y1=normpdf(x,-3,1);
y2=normpdf(x,-1,2/3);
y3=normpdf(x,-15/9,4/9);
y4=normpdf(x,-19/9,8/27);
y5=normpdf(x,-65/27,16/81);
y6=normpdf(x,-211/81,32/243);
y7=normpdf(x,-665/243,64/729);
y8=normpdf(x,-2059/729,128/2187);
z=0;
for i=1:p
    z(i)=1/8*[y1(i)+y2(i)+y3(i)+y4(i)+y5(i)+y6(i)+y7(i)+y8(i)];
end
z;
plot(x,z)



% #3(Kurtotic Unimodal)

y=(2/3)*normpdf(x,0,1)+(1/3)*normpdf(x,0,0.1);
plot(x,y)

% #4(Outlier)

y=(1/10)*normpdf(x,0,1)+(9/10)*normpdf(x,0,0.1);
plot(x,y)

% #5(Bimodal)

y=(1/2)*normpdf(x,-1,2/3)+(1/2)*normpdf(x,1,2/3);
plot(x,y)



% #6 (Seperated Bimodal)
```

```
y=(1/2)*normpdf(x,-3/2,1/2)+(1/2)*normpdf(x,3/2,1/2);
plot(x,y)

% #7 (Skewed Bimodal)

y=(3/4)*normpdf(x,0,1)+(1/4)*normpdf(x,3/2,1/3);
plot(x,y)

% #8 (Trimodal Bimodal)
y=(9/20)*normpdf(x,-
6/5,3/5)+(9/20)*normpdf(x,6/5,3/5)+(1/10)*normpdf(x,0,1/4);
plot(x,y)

% #9(Claw)
p=length(x);
y1=normpdf(x,-1,1/10);
y2=normpdf(x,-1/2,1/10);
y3=normpdf(x,0,1/10);
y4=normpdf(x,1/2,1/10);
y5=normpdf(x,1,1/10);

z=0;
for i=1:p
    z(i)=(1/10)*(y1(i)+y2(i)+y3(i)+y4(i)+y5(i);
end
z;
plot(x,z)

% #10(Double Claw)
p=length(x);
y1=normpdf(x,-1,2/3)
y2=normpdf(x,1,2/3)
y3=normpdf(x,-3/2,1/100);
y4=normpdf(x,-1,1/100);
y5=normpdf(x,-1/2,1/100);
y6=normpdf(x,0,1/100);
y7=normpdf(x,1/2,1/100);
y8=normpdf(x,1,1/100);
y9=normpdf(x,3/2,1/100);

z=0;
for i=1:p

z(i)=[((49/100)*(y1(i)+y2(i)))+((1/350)*(y3(i)+y4(i)+y5(i)+y6(i)+y7(i
)+y8(i)+y9(i)))];
end
z;
plot(x,z)

% #11 (Asymmetric Claw)



x=[1:6000]/1000-3;
 p=length(x);
```

```
y1=normpdf(x,0,1);
y2=normpdf(x,-3/2,2/5);
y3=normpdf(x,-1/2,1/5);
y4=normpdf(x,1/2,1/10);
y5=normpdf(x,3/2,1/20);
y6=normpdf(x,5/2,1/40);
z=0;
for i=1:p
    z(i)=1/2*y1(i)+
8/31*y2(i)+4/31*y3(i)+2/31*y4(i)+1/31*y5(i)+1/62*y6(i);
end
z;
plot(x,z)
```

```
% #12(Asymmetric Double Claw)
p=length(x);
y1=normpdf(x,-1,2/3);
y2=normpdf(x,1,2/3);
y3=normpdf(x,-1/2,1/100);
y4=normpdf(x,-1,1/100);
y5=normpdf(x,-3/2,1/100);
y6=normpdf(x,1/2,7/100);
y7=normpdf(x,1,7/100);
y8=normpdf(x,3/2,7/100);
z=0;
for i=1:p

z(i)=[(46/100)*(y1(i)+y2(i))+(1/300)*(y3(i)+y4(i)+y5(i))+(7/300)*(y6(
i)+y7(i)+y8(i))];
end
z;
plot(x,z)
```

```
% #13(Smooth Comb)
p=length(x);
y1=normpdf(x,-31/21,32/63);
y2=normpdf(x,17/21,0.25*(32/63));
y3=normpdf(x,41/21,0.125*(32/63));
y4=normpdf(x,53/21,0.156*(32/63));
y5=normpdf(x,59/21,(1/256)*(32/63));
y6=normpdf(x,62/21,(1/1024)*(32/63));

z=0;
for i=1:p

z(i)=((32/63)*y1(i))+((16/63)*y2(i))+((8/63)*y3(i))+((4/63)*y4(i))+((
2/63)*y5(i))+((1/63)*y6(i));



end
```

```
z;
plot(x,z)



% #14(Discrete Comb)
p=length(x);
y1=normpdf(x,-15/7,2/7);
y2=normpdf(x,-3/7,2/7);
y3=normpdf(x,9/7,2/7);
y4=normpdf(x,16/7,1/21);
y5=normpdf(x,18/7,1/21);
y6=normpdf(x,20/7,1/21);

z=0;
for i=1:p
    z(i)=(2/7)*(y1(i)+y2(i)+y3(i))+(1/21)*(y4(i)+y5(i)+y6(i));
end
z;
plot(x,z)
```

MLE and EM algorithm code:

```
clc;
clear all;
n=50;
k=1;
for is=1:1000
    x=zeros(n,1);
    for i=1:n
        if rand() <= 0.75
            x(i)=normrnd(0,1);
        else
            x(i)=normrnd(3/2,1/3);
        end
    end
    d=4;
    a=0.5;
    m1=mean(x);
    m2=median(x);
    v1=var(x);
    v2=(n-1)*var(x)/n;
    [a m1 m2 v1 v2];
         for jj=1:9
             p1=zeros(n,1);
             p2=p1;
             p1=normpdf(x,m1,sqrt(v1));



             p2=normpdf(x,m2,sqrt(v2));
```

```
            P=zeros(n,1);
            P=a*p1./(a*p1+(1-a)*p2);
            Q=1-P;
            a=mean(P);
            m1=mean(P.*x);
            m2=mean(Q.*x);
            v1=mean(P.*(x-m1).^2);
            v2=mean(Q.*(x-m2).^2);
            [a m1 m2 v1 v2];
        end
    for jj=1:15
        R=[m1 m2 v1 v2]';
        for j=1:17
            I=zeros(d,d);
            U=zeros(d,1);
            for i=1:n
                f1=normpdf(x(i),m1,sqrt(v1));
                f2=normpdf(x(i),m2,sqrt(v2));
                D=a*f1+(1-a)*f2;
                if D <= 0.0001;
                    D=0.0001;
                end
                U(1)=U(1)+f1*a*((x(i)-m1)/v1)/D;
                U(2)=U(2)+f2*(1-a)*((x(i)-m2)/v2)/D;
                U(3)=U(3)-a*f1/(2*v1^2)*(v1-(x(i)-m1)^2)/D;
                U(4)=U(4)-(1-a)*f2/(2*v2^2)*(v2-(x(i)-m2)^2)/D;
                I(1,1)=I(1,1)-a*f1/v1^2*(v1*(a*f1+(1-a)*f2)-(1-
a)*f2*(x(i)-m1)^2)/D^2;
                I(1,2)=I(1,2)-a*(1-a)*f1*f2*(x(i)-m1)*(x(i)-
m2)/(v1*v2)/D^2;
                I(1,3)=I(1,3)-a*f1*(x(i)-m1)/(2*v1^3)*(2*a*f1*v1+(1-
a)*f2*(3*v1-(x(i)-m1)^2))/D^2;
                I(1,4)=I(1,4)+a*(1-a)*f1*f2*(x(i)-
m1)/(v1*2*v2^2)*(v2-(x(i)-m2)^2)/D^2;
                I(2,2)=I(2,2)-(1-a)*f2/v2^2*(v2*(a*f1+(1-a)*f2)-
a*f1*(x(i)-m2)^2)/D^2;
                I(2,3)=I(2,3)+a*(1-a)*f1*f2*(x(i)-
m2)/(v2*2*v1^2)*(v1-(x(i)-m1)^2)/D^2;
                I(2,4)=I(2,4)-(1-a)*f2*(x(i)-m2)/(2*v2^3)*(2*(1-
a)*f2*v2+3*a*f1*v2-a*f1*(x(i)-m2)^2)/D^2;
                I(3,3)=I(3,3)+a*f1/(4*v1^4)*(2*a*f1*v1*(v1-2*(x(i)-
m1)^2)+(1-a)*f2*(3*v1^2-6*(x(i)-m1)^2*v1+(x(i)-m1)^4))/D^2;
                I(3,4)=I(3,4)-(a*(1-a)*f1*f2*(v1-(x(i)-m1)^2)*(v2-
(x(i)-m2)^2)/(4*v1^2*v2^2))/D^2;
                I(4,4)=I(4,4)+(1-a)*f2/(4*v2^4)*(2*(1-a)*f2*v2*(v2-
2*(x(i)-m2)^2)+a*f1*(3*v2^2-6*(x(i)-m2)^2*v2+(x(i)-m2)^4))/D^2;
            end
            I(2,1)=I(1,2);
            I(3,1)=I(1,3);
            I(4,1)=I(1,4);
            I(3,2)=I(2,3);
            I(4,2)=I(2,4);
            I(4,3)=I(3,4);
            A=I;



            B=eye(4);
```

```
if A(1,1) < 0.0001 && A(1,1) >= 0
    A(1,1)=0.0001;
end
if A(1,1) > -0.0001 && A(1,1) < 0
    A(1,1)=-0.0001;
end
xx=A(1,1);
A(1,:)=A(1,:)/xx;
B(1,:)=B(1,:)/xx;
xx=A(2,1);
A(2,:)=A(1,:)*xx-A(2,:);
B(2,:)=B(1,:)*xx-B(2,:);
xx=A(3,1);
A(3,:)=A(1,:)*xx-A(3,:);
B(3,:)=B(1,:)*xx-B(3,:);
xx=A(4,1);
A(4,:)=A(1,:)*xx-A(4,:);
B(4,:)=B(1,:)*xx-B(4,:);
if A(2,2) < 0.0001 && A(2,2) >= 0
    A(2,2)=0.0001;
end
if A(2,2) > -0.0001 && A(2,2) < 0
    A(2,2)=-0.0001;
end
xx=A(2,2);
A(2,:)=A(2,:)/xx;
B(2,:)=B(2,:)/xx;
xx=A(3,2);
A(3,:)=A(2,:)*xx-A(3,:);
B(3,:)=B(2,:)*xx-B(3,:);
xx=A(4,2);
A(4,:)=A(2,:)*xx-A(4,:);
B(4,:)=B(2,:)*xx-B(4,:);
if A(3,3) < 0.0001 && A(3,3) >= 0
    A(3,3)=0.0001;
end
if A(3,3) > -0.0001 && A(3,3) < 0
    A(3,3)=-0.0001;
end
xx=A(3,3);
A(3,:)=A(3,:)/xx;
B(3,:)=B(3,:)/xx;
xx=A(4,3);
A(4,:)=A(3,:)*xx-A(4,:);
B(4,:)=B(3,:)*xx-B(4,:);
if A(4,4) < 0.0001 && A(4,4) >= 0
    A(4,4)=0.0001;
end
if A(4,4) > -0.0001 && A(4,4) < 0
    A(4,4)=-0.0001;
end
xx=A(4,4);
A(4,:)=A(4,:)/xx;
B(4,:)=B(4,:)/xx;



xx=A(3,4);
```

```
            A(3,:)=-A(4,:)*xx+A(3,:);
            B(3,:)=-B(4,:)*xx+B(3,:);
            xx=A(2,4);
            A(2,:)=-A(4,:)*xx+A(2,:);
            B(2,:)=-B(4,:)*xx+B(2,:);
            xx=A(1,4);
            A(1,:)=-A(4,:)*xx+A(1,:);
            B(1,:)=-B(4,:)*xx+B(1,:);
            xx=A(2,3);
            A(2,:)=-A(3,:)*xx+A(2,:);
            B(2,:)=-B(3,:)*xx+B(2,:);
            xx=A(1,3);
            A(1,:)=-A(3,:)*xx+A(1,:);
            B(1,:)=-B(3,:)*xx+B(1,:);
            xx=A(1,2);
            A(1,:)=-A(2,:)*xx+A(1,:);
            B(1,:)=-B(2,:)*xx+B(1,:);
            R=R-B*U;
            m1=R(1);
            m2=R(2);
            v1=R(3);
            v2=R(4);
        end
        p1=zeros(n,1);
        p2=p1;
        p1=normpdf(x,m1,sqrt(v1));
        p2=normpdf(x,m2,sqrt(v2));
        P=zeros(n,1);
        P=a*p1./(a*p1+(1-a)*p2);
        a=mean(P);
    end
    [a m1 m2 v1 v2];
    if min(abs([a m1 m2 v1 v2])) >= 0.00000001 && max(abs([a m1 m2 v1
v2])) <= 200
        PP(k,:)=[a m1 m2 v1 v2];
        BB(k,:)=abs(PP(k,:)-[0.75 0.00 1.50 1.00 (1/3)^2]);
        L(k)=sum(log(a*normpdf(x,m1,sqrt(v1))+(1-
a)*normpdf(x,m2,sqrt(v2))));
        k=k+1;
    end
         RR(:,is)=[a; R];
end
RR=RR';
II=[1:1000]';
RR=[II RR];
PPP=[mean(PP); std(PP); mean(BB)]
PPP=[mean(PP); std(PP); mean(BB)]
[mean(PP(:,1)) mean(PP(:,2)) mean(PP(:,3))  mean(PP(:,4))
mean(PP(:,5)) ]
[std(PP(:,1)) std(PP(:,2)) std(PP(:,3))  std(PP(:,4))  std(PP(:,5)) ]
[mean(BB)]
[std(BB)]
[mean(L) std(L)]
```

Bibliography

E. Fix and J. L. Hodges. Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

K. Lange. Statistics and Computing: Numerical Analysis for Statisticians, Springer, New York, 1999.

J. S. Marron and M. P. Wand. Exact mean integrated squared error. The Annals of Statistics, 20(2), 712-736, 1992.

M. Rahman, B. C. Arnold and D. V. Gokhale and Aman Ullah. ”Data-Based Selection of the Smoothing Parameter in Kernel Density Estimation Using Exact and Approximate MISE”, Technical Report No. 229, Department of Statistics, University of California Riverside, 1996.

B. W. Silverman. Density Estimation for Statistics and Data Analysis. Chapman and Hall, Bristol, 1986.

K. E. Basford, G. J. McLachlan, and M. G. York (1997). Modeling the distribution of stamp paper via finite normal mixtures: the 1872 Hidalgo stamp issue of Mexico revisited. Journal of Applied Statistics, 24, 169180.

I. Eisenberger (1964). Genesis of bimodal distributions, Technimetrics 6, 357363.

R. V. Hogg, J. W. McKean, and A. T. Craig (2005). Introduction to Mathematical Statistics, Prentice Hall, New Jersey, USA.

M. E. Johnson (1987). Multivariate Statistical Simulation, John Wiley & Sons, New York, USA.

Triola, Mario F. (2008). "Essentials of Statistics:, Third Edition, Pearson Education, Inc., Boston,MA.