# Kernel Smoothing Based on Bernstein Polynomials

SHANSHAN XIAN

A Thesis Submitted to the Faculty of Graduate Studies
in Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Department of Statistics
University of Manitoba
Winnipeg, Manitoba

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION

Kernel Smoothing
Based on Bernstein Polynomials

BY

Shanshan Xian

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

Of

Master of Science

# Contents

# Abstract

Kernel smoothing refers to a general class of techniques for non-parametric and semi-parametric estimation of function. In this thesis, we are looking for some adaptations of the kernel density estimator which attempt to overcome its limitations. We will introduce the new kernel estimation based on the application of Bernstein polynomials for approximating a bounded and continuous function. This kernel-based method can uncover structural features in the data which a traditional approach might not reveal. We are also interested in its graphical and numerical illustrations using Monte Carlo simulation.

**Key words**: kernel estimation, bandwidth, Bernstein polynomials, strong consistency, Monte Carlo simulation.

# Acknowledgements

First of all, I wish to express my sincere appreciation to my honorable supervisor, Dr. Jin Zhang, for his encouragement, guidance, patience and suggestions, and for his financial support from his research grants. This thesis would not have been possible without the support of his generous help.I am also very thankful to my committee members, Dr. Xikui Wang of the Department of Statistics, and Dr. Cam-Loi Huynh of the Department of Psychology, who offered advice and motivation. Their contribution makes this thesis more professional. I would also like to thank the faculty and staffs of the Department of Statistics for their wonderful teaching and unselfish help throughout my graduate studies.

Second of all, I am grateful to my family members and my boyfriend, who always offer endless support and love to me.

Finally, my fellows and friends, especially my 5-guy-group, also deserve special thanks for their views and tips in supplying the relevant literature.

# Chapter 1

# Introduction

In recent years, nonparametric estimation has become a popular topic in statistics and some applied fields. More and more statisticians and researchers focus on searching for the smoothing techniques of density and distribution functions.

Kernel smoothing is well-known in the nonparametric world. The basic ideas of kernel density estimation first appeared in a technical report by E. Fix and J. L. Hodges (1951). It conquers the discontinuity of histogram and constructs the continuous density estimation. The further developments of the kernel density estimation are given by Silverman (1986), Härdle (1991) and Wand and Jones (1995).

Bernstein polynomials are the important and interesting concrete operators on a space of continuous functions. They can be differentiated and integrated easily, and can be pieced together to form spline curves that can approximate any function to any accuracy desired. Babu, Canty and

Chaubey (2002) extended the application of Bernstein polynomials to the field of smoothing and derived the smooth density and distribution estimation.

In this thesis, we will introduce a new smoothing estimation based on kernel smoothing and Bernstein polynomials. In Chapter 2, we will review the knowledge of traditional kernel smoothing and the application of Bernstein polynomials on the smooth density estimation. Our new estimation is constructed in Chapter 3, and we will discuss its statistical properties and use some graphs to compare the three approaches for the normal mixture, Beta distribution and Gamma distribution. Comprehensive numerical comparisons based on Monte Carlo simulation are given in Chapter 4. At last, conclusions and further questions are discussed in Chapter 5.

# Chapter 2

# Preliminaries

## 2.1 Traditional Kernel Density Estimation

Suppose that $X_1, \ldots, X_n$ is a random sample from a continuous population $X$, having a probability density function(p.d.f.) $f$. The basic parametric approach of estimating $f$ is to assume that $f$ belongs to a parametric family of distribution, such as the normal or gamma family, and then estimate the unknown parameters using least square or maximum likelihood estimation. Sometimes, we do not have information of the population distribution. So we have to choose nonparametric methods which do not assume any pre-specified functional form for $f$. The most widely used nonparametric density estimation is *kernel smoothing*. This is usually formed by dividing the real line into intervals $[x-h, x+h)$, often called *bins*. Consider a reasonable way

to estimate $f(x)$:

$$\begin{aligned}\hat{f}(x) &= \frac{1}{\text{bin length} \cdot n}\sharp\{\text{observations that fall into a small bin around x}\} \\ &= \frac{1}{2hn}\sharp\{X_i \in [x-h, x+h)\} \qquad i = 1, \ldots, n\end{aligned}$$

Here $x$ is a point at which we want to estimate its probability density, and $h$ is a smoothing parameter called the *bandwidth* or *window width*. In the following, we will illustrate the univariate kernel density estimation.

### 2.1.1 Univariate Kernel Density Estimation

To begin with, the univariate kernel density estimator is defined as

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - X_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right) \tag{2.1}$$

where $K(\cdot)$ is a general p.d.f., and is called the *kernel*. The kernel is a symmetric function satisfying

$$\int K(x)dx = 1, \quad \int xK(x)dx = 0, \quad \int x^2K(x)dx = \mu_2(K) < \infty \tag{2.2}$$

Since the kernel is a density function, the kernel estimator $\hat{f}_h(x)$ is also a density function.

By definition, kernel density estimation can be seen as a weighting function to observations. Let us take the uniform kernel function and Epanechnikov kernel function as examples. We rewrite (2.1) as follows, which is called the uniform kernel function

$$K(u) = \frac{1}{2}I(|u| \leq 1)$$

where $u = (x - X_i)/h$. That is, the uniform kernel function assigns weight $1/2$ to each observation $X_i$ whose distance from $x$ is no more than $h$, and gives zero weight to points that are away from $x$. Furthermore, Epanechnikov kernel function is defined by

$$K(u) = \frac{3}{4}(1 - u^2)I(|u| \leq 1),$$

which gives more weight to observations $X_i$ that are closer to $x$, then less weight to those that are further away. This property of weighting function is shared by many other kernels, some of which are introduced in Table 2.1. The shapes of some kernel functions are shown in Figure 2.1.

| KERNEL | K(u) |
|---|---|
| Uniform | $\frac{1}{2}I(\|u\| \leq 1)$ |
| Triangle | $(1 - \|u\|)I(\|u\| \leq 1)$ |
| Epanechnikov | $\frac{3}{4}(1 - u^2)I(\|u\| \leq 1)$ |
| Quartic(Biweight) | $\frac{15}{16}(1 - u^2)^2 I(\|u\| \leq 1)$ |
| Triweight | $\frac{35}{32}(1 - u^2)^3 I(\|u\| \leq 1)$ |
| Gaussian | $\frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}u^2)$ |
| Cosine | $\frac{\pi}{4} \cos(\frac{\pi}{2}u)I(\|u\| \leq 1)$ |

Table 2.1: Kernel functions

Note that the choice of the kernel function is not particularly important. However, the choice of the value of bandwidth $h$ is a crucial problem. Figure 2.2 plots kernel density estimate for a simulated sample of size $n$ =300 from Gamma(2, 1) using the Gaussian kernel function with various values of $h$.

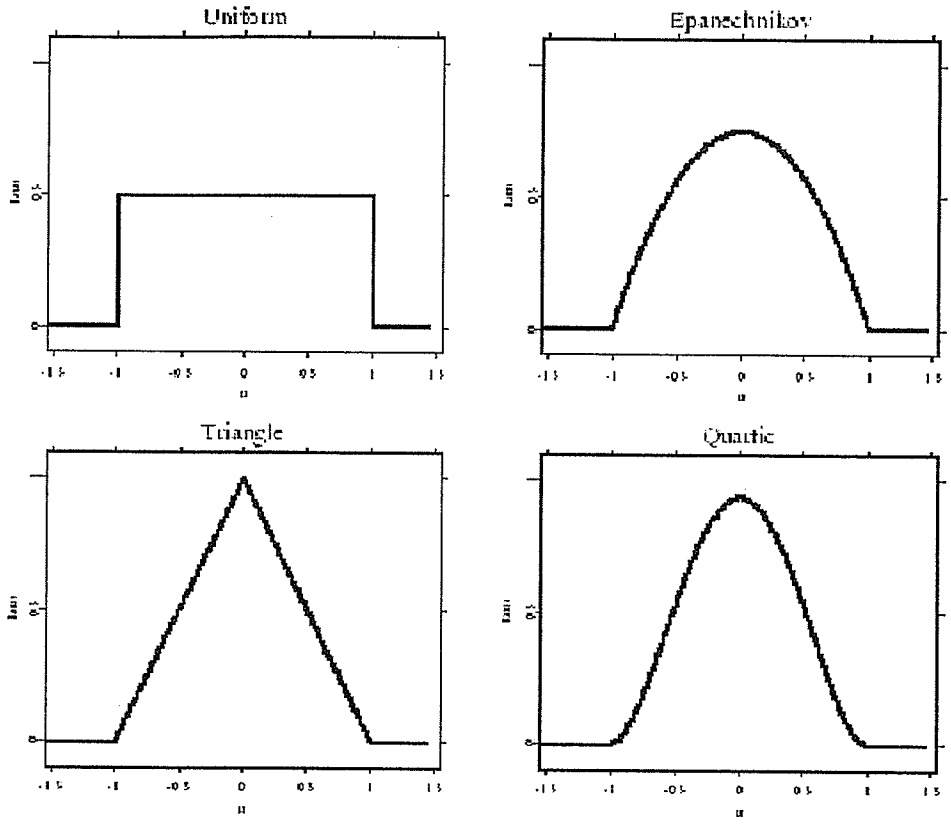


Figure 2.1: Some kernel functions

From the graph with $h = 0.1$ in Figure 2.2, it can be seen that the narrowness of the kernel means that this estimate pays too much attention to the particular data set at hand and does not allow for the variation across samples. Such an estimate is said to be *undersmoothed.* Conversely, from another kernel estimate based on the same data, but with $h = 0.8$, it is a much smoother estimate which is really too smooth since the peak has been smoothed away. This is an example of an *oversmoothed* case. In the rest

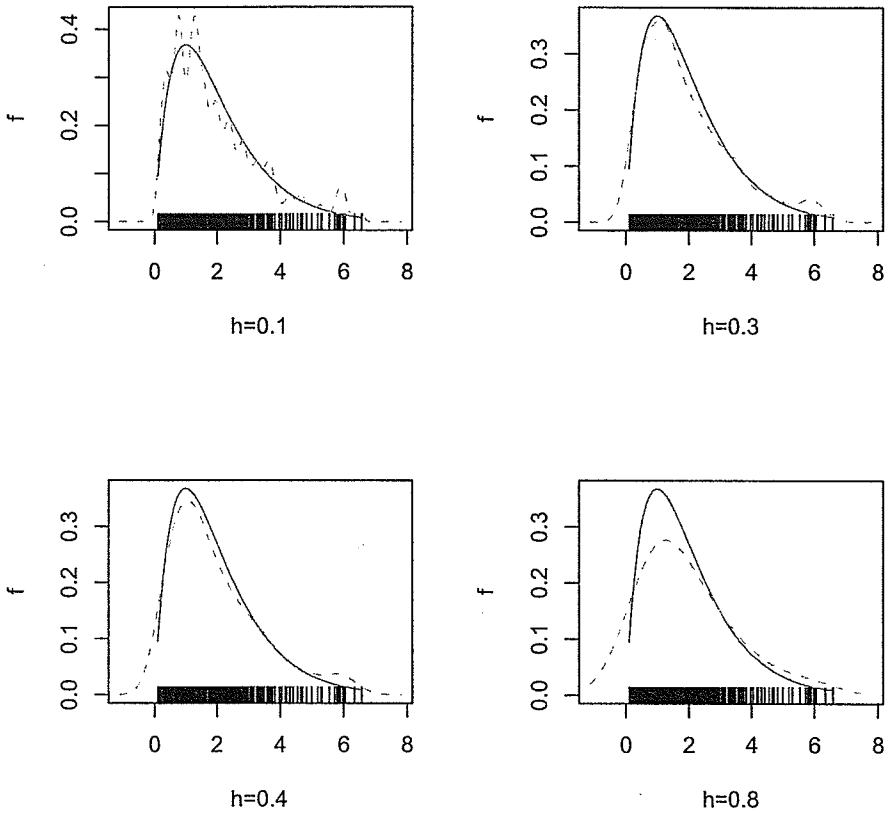


Figure 2.2: Kernel density estimates for $Gamma(2,1)$ with respective bandwidth $h$=0.1, $h$=0.3, $h$=0.4, $h$=0.8

——— true density, - - - - - kernel density estimate

of the graphs of Figure 2.2, reasonable results are reached with $h = 0.3$ and $h = 0.4$.

### 2.1.2 Statistical Properties

Now, we turn our attention to statistical properties of univariate kernel density estimation.

**Expectation**

$$\begin{aligned} E\{\hat{f}_h(x)\} &= E\left\{\frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right)\right\} = \frac{1}{h}E\left\{K\left(\frac{x-X}{h}\right)\right\} \\ &= \frac{1}{h}\int K\left(\frac{x-u}{h}\right) f(u)du \end{aligned}$$

We find that kernel density estimator is not an unbiased estimator of $f(x)$.

The bias of the estimator is

$$\begin{aligned} Bias\{\hat{f}_h(x)\} &= E\{\hat{f}_h(x) - f(x)\} \\ &= \frac{1}{h}\int K\left(\frac{x-u}{h}\right) f(u)du - f(x). \\ &= \int K(s)f(x-hs)ds - f(x) \end{aligned}$$

To simplify $Bias\{\hat{f}_h(x)\}$, taking the second-order Taylor expansion to $Bias\{\hat{f}_h(x)\}$, we have

$$Bias\{\hat{f}_h(x)\} = \frac{h^2}{2}f''(x)\mu_2(K) + o(h^2), \qquad as \quad h \to 0. \tag{2.3}$$

where $\mu_2(K)$ is defined in (2.2). Observe that bias is proportional to $h^2$, that means large value of $h$ implies large value of bias. Moreover, bias also depends on $f''(x)$, the curvature of the density at $x$.

**Variance**

$$\begin{aligned} Var\{\hat{f}_h(x)\} &= Var\left\{\frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right)\right\} = \frac{1}{nh^2} Var\left\{K\left(\frac{x-X}{h}\right)\right\} \\ &= \frac{1}{nh^2}\left(E\left\{K^2\left(\frac{x-X}{h}\right)\right\} - \left[E\left\{K\left(\frac{x-X}{h}\right)\right\}\right]^2\right) \end{aligned}$$

Taking

$$\frac{1}{nh^2} E\left\{K^2\left(\frac{x-X}{h}\right)\right\} = \frac{1}{n}\frac{1}{h^2}\int K^2\left(\frac{x-u}{h}\right) f(u)du$$

$$E\left\{K\left(\frac{x-X}{h}\right)\right\} = f(x) + o(h)$$

and similar variable substitution and Taylor expansion as in the derivation of the bias, it can be shown that

$$Var\{\hat{f}_h(x)\} = \frac{1}{nh} f(x) \parallel K \parallel_2^2 + o\left(\frac{1}{nh}\right), \qquad as \quad nh \to \infty. \tag{2.4}$$

Here, we denote $\parallel K \parallel_2^2 = \int K^2(s)ds$, the squared $L_2$ norm of $K$. Notice that variance goes down when the value of $h$ goes up. Now, we face a dilemma: we would like to keep both variance and bias as small as possible, but increasing $h$ will lower the variance while raising the bias. So we have to settle for finding the value of $h$ that yields (in some sense) the optimal compromise between variance and bias reduction.

**Mean Squared Error**

We have already known that choosing the bandwidth $h$ is an important problem in nonparametric density estimation, and variance and bias vary in

opposite direction of $h$. Minimizing the *mean squared error* (MSE) represents a trade-off to solve this dilemma. By definition,

$$\begin{aligned} MSE\{\hat{f}_h(x)\} &= E\{\hat{f}_h(x) - f(x)\}^2 \\ &= Var\{\hat{f}_h(x)\} + [Bias\{\hat{f}_h(x)\}]^2 \end{aligned}$$

From equations (2.3) and (2.4), it yields

$$MSE\{\hat{f}_h(x)\} = \frac{h^4}{4}[f''(x)]^2\mu_2^2(K) + \frac{1}{nh}f(x) \parallel K \parallel_2^2 + o(h^4) + o\left(\frac{1}{nh}\right)$$

We can see MSE goes to zero as $h \longrightarrow 0$ and $nh \longrightarrow \infty$. Hence, the kernel density estimator is indeed consistent.

**MISE and AMISE**

We observe that MSE depends on unknown $f(x)$ and $f''(x)$, that means, it is hard to obtain a good estimator of $h$. So we turn to deriving *mean integrated squared error* (MISE)-optimal bandwidth. In fact, the advantage of choosing MISE is that we can get a global measure of estimation. By definition, we have

$$MISE_{\hat{f}_h} = MISE(\hat{f}_h) = \int MSE\{\hat{f}_h(x)\}dx = \int E[\hat{f}_h(x) - f(x)]^2 dx. \quad (2.5)$$

We simplify the integral to obtain

$$\begin{aligned} MISE(\hat{f}_h) &= \frac{h^4}{4}\mu_2^2(K)\int\{f''(x)\}^2dx \\ &+ \frac{1}{nh} \parallel K \parallel_2^2 \int f(x)dx + o(h^4) + o\left(\frac{1}{nh}\right) \\ &= \frac{h^4}{4}\mu_2^2(K) \parallel f'' \parallel_2^2 + \frac{1}{nh} \parallel K \parallel_2^2 + o(h^4) + o\left(\frac{1}{nh}\right), \\ &\quad as\, h \longrightarrow 0, nh \longrightarrow \infty. \end{aligned}$$

Ignoring the higher order terms, the asymptotic MISE is

$$AMISE(\hat{f}_h) = \frac{h^4}{4}\mu_2^2(K) \parallel f'' \parallel_2^2 + \frac{1}{nh} \parallel K \parallel_2^2 . \tag{2.6}$$

Minimizing the AMISE with respect to $h$, we obtain the AMISE-optimal bandwidth

$$h_{opt} = \left( \frac{\parallel K \parallel_2^2}{\parallel f'' \parallel_2^2 \mu_2^2(K) n} \right)^{\frac{1}{5}} \propto n^{-\frac{1}{5}} \tag{2.7}$$

Here, there still exists an unknown term, $\parallel f'' \parallel_2^2$, but at least we can use $h_{opt}$ to do further research according to the statistical properties of kernel density estimator. Moreover, we are interested in at what speed the smoothers converge to the true curve if the number of observations increase. Inserting $h_{opt}$ in AMISE equation, we get $AMISE(\hat{f}_{h_{opt}}) \propto n^{-\frac{4}{5}}$, that is, if sample size $n$ gets larger and larger, AMISE is converging to zero at the rate $n^{-\frac{4}{5}}$.

**Other Properties**

To measure how close the estimated distribution is to the true distribution, there are good reasons for working with absolute error. Absolute error ignores the direction of error because positive and negative errors offset to a lower mean that reflects estimation bias and not error, such as the *mean integrated absolute error* (MIAE) given by

$$MIAE_{\hat{f}_h} = MIAE(\hat{f}_h) = E \int |\hat{f}_h(x) - f(x)| dx. \tag{2.8}$$

It includes the fact that MIAE is always defined when $\hat{f}_h(x)$ is a density, and invariance of MIAE under monotone transformations (see Devroye and

Györfi, 1985,p.1). However, the analysis of this quantity is substantially more complicated.

MISE and MIAE criteria can be applied to the cumulative distribution function (c.d.f.) to construct the reasonable basic measures of error.

$$MISE_{\hat{F}_h} = MISE\{\hat{F}_h\} = E\int [\hat{F}_h(x) - F(x)]^2 f(x)dx \tag{2.9}$$

$$MIAE_{\hat{F}_h} = MIAE\{\hat{F}_h\} = E\int |\hat{F}_h(x) - F(x)| f(x)dx \tag{2.10}$$

### 2.1.3 Smoothing Parameter Selection

When one estimates a continuous density from a data set, he often seeks to smooth the discrete data. The challenge in smoothing is to choose the best bandwidth that balances the desire to reduce the variance of the estimator (which needs lots of data points) yet capture significant small-scale features in the underlying distribution (which needs a narrow bandwidth). Currently the following methods are used to select the bandwidth for $\hat{f}_h(x)$:

1. Normal Reference Bandwidth
2. Cross-Validation Bandwidth
3. Plug-in Bandwidth

**Normal Reference Bandwidth** (Scott (1992))

If the population $X$ has a normal reference density $f$, the $h_{opt}$ in (2.7)

becomes

$$h_{NR} = \left( \frac{8\sqrt{\pi} \parallel K \parallel_2^2}{3\mu_2^2(K)n} \right)^{\frac{1}{5}} \sigma, \tag{2.11}$$

because $\parallel f'' \parallel_2^2 = \frac{3}{8\sqrt{\pi}}\sigma^{-5}$. Using sample standard deviation $s$ to estimate $\sigma$, and choosing a Gaussian kernel function, we get

$$\hat{h}_{NR} = \left( \frac{4\hat{\sigma}^5}{3n} \right)^{\frac{1}{5}} \approx 1.06\hat{\sigma}n^{-\frac{1}{5}} \tag{2.12}$$

where $\varphi$ is the p.d.f. of standard normal distribution, and $\parallel \varphi \parallel_2^2 = \frac{1}{2\sqrt{\pi}}$ and $\mu_2^2(\varphi) = 1$.

Normal reference bandwidth is sensitive to outliers, because a single outlier may cause a too large $\hat{\sigma}$ and hence implies a big bandwidth. We hope to find a more robust estimator. Thus we consider the interquartile range $IQR = X_{[0.75n]} - X_{[0.25n]}$. So, the widely known "rule-of-thumb" bandwidth (Silverman (1986)) is obtained by

$$\hat{h}_{rot} = 1.06 \cdot \min\{\hat{\sigma}, \frac{IQR}{\Phi^{-1}(\frac{3}{4}) - \Phi^{-1}(\frac{1}{4})}\}n^{-\frac{1}{5}} \tag{2.13}$$

where $\Phi^{-1}$ is the standard normal quantile function.

What we achieved by working under the normality assumption is an explicit, applicable formula for bandwidth selection. In practice, we do not know whether $X$ is normally distributed. If it is, then $\hat{h}_{NR}$ gives the optimal bandwidth. If not, then $\hat{h}_{NR}$ will give a bandwidth not too far from the optimum if the distribution of $X$ is not too different from the normal distribution. That's why normal reference bandwidth gives reasonable results for

all distributions that are unimodal, roughly symmetric and do not have tails that are too fat.

**Cross-Validation Bandwidth**

Cross-validation method is based on more involved mathematical arguments and requires considerably more computational effort. It is known that least squares cross-validation and biased cross-validation are frequently used to select the bandwidth.

*Least Squares Cross-validation* (Rudemo (1982) and Bowman (1984))

The optimal bandwidth for a kernel density estimate is typically calculated on the basis of an estimate for MISE or *integrated squared error* (ISE):

$$\begin{aligned} ISE(h) &= ISE\{\hat{f}_h(x)\} = \int \{\hat{f}_h(x) - f(x)\}^2 dx \\ &= \int \hat{f}_h^2(x)dx - 2E[\hat{f}_h(x)] + \int f^2(x)dx \end{aligned}$$

Apparently, ISE is an alternative distance measure between $\hat{f}_h$ and $f$. It is obvious that the third term of ISE is not determined by $h$, and can be ignored as far as minimization over $h$ is concerned. Moreover, $\int \hat{f}_h^2(x)dx$ can be obtained from the data. This leaves us with one term containing $h$ and unknown quantity $f$. The unbiased estimate of the expected value is allowed by

$$\widehat{E\{\hat{f}_h(X)\}} = \frac{1}{n}\sum_{i=1}^{n} \hat{f}_{h,-i}(X_i) \tag{2.14}$$

where

$$\hat{f}_{h,-i}(x) = \frac{1}{n-1}\sum_{j=1, i\neq j}^{n} K_h(x - X_j).$$

Here, $\hat{f}_{h,-i}(x)$ is the leave-one-out estimator, which suggests the $i^{th}$ observation is not used to estimate $\hat{f}_{h,-i}(X_i)$, so that $\hat{f}_{h,-i}(X_i)$ is independent on $X_i$.

Now, the least squares cross-validation criterion is given by

$$\begin{aligned} LSCV(h) &= \int \hat{f}_h^2(x)dx - 2E[\hat{f}_h(x)] \\ &= \int \hat{f}_h^2(x)dx - \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1, i\neq j}^{n} K_h(X_i - X_j). \end{aligned}$$

The estimate $\hat{h}_{LSCV}$ is obtained by minimizing LSCV(h). Studies have shown that the theoretical and practical performance of this bandwidth selector are somewhat disappointing. In particular, $\hat{h}_{LSCV}$ is highly variable. This leads to the proposal of several other hi-tech bandwidth selectors that aim to improve upon $\hat{h}_{LSCV}$.

*Biased Cross-Validation*(Scott and Terrell (1987))

Instead of using the exact MISE or ISE formula, biased cross-validation(BCV) is based on the formula for AMISE:

$$AMISE(\hat{f}_h) = \frac{h^4}{4}\mu_2^2(K) \parallel f'' \parallel_2^2 + \frac{1}{nh} \parallel K \parallel_2^2$$

The BCV objective function is obtained by replacing the unknown $\parallel f'' \parallel_2^2$ by the estimator

$$\begin{aligned} \widetilde{\parallel f'' \parallel_2^2} &= \parallel \hat{f}'' \parallel_2^2 - \frac{1}{nh^5} \parallel K'' \parallel_2^2 \\ &= \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} K_h'' \star K_h''(X_i - X_j) \end{aligned}$$

to give

$$BCV(h) = \frac{h^4}{4}\mu_2^2(K)\widetilde{\| f'' \|_2^2} + \frac{1}{nh} \| K \|_2^2 \tag{2.15}$$

$\hat{h}_{BCV}$ is the minimizer of $BCV(h)$. This method is really a hybrid of cross-validation and normal reference bandwidth selection, since it involves replacement of the unknown $\| f'' \|_2^2$ by the cross-validatory kernel estimator $\widetilde{\| f'' \|_2^2}$.

**Plug-in Bandwidth**

Generally speaking, plug-in method derives its name from the underlying principle: if there exists an expression involving an unknown parameter, replace the unknown parameter with an estimate.

*Direct Plug-in Rule* (Park and Marron (1992))

Also motivated by the formula for the AMISE-optimal bandwidth, each objective function is thus of the form

$$\psi(h) = \frac{h^4}{4}\mu_2^2(K)\hat{S}_D(\alpha) + \frac{1}{nh} \| K \|_2^2 \tag{2.16}$$

where

$$\hat{S}_D(\alpha) = n(n-1)^{-1}\alpha^{-5}\sum_i\sum_j L^{(4)}\alpha^{-1}(X_j - X_i) \tag{2.17}$$

(this derives from the estimate $n^{-1}\sum \hat{f}_\alpha^{(4)}(X_i)$ of $\| f'' \|_2^2$ and the subscript D standing for "diagonals in"). Notice that, importantly, $\alpha$ is another bandwidth differing from $h$, and also that $L$ is another symmetric density not

necessarily $K$. The objective function leads to the direct plug-in rule

$$h_{dpi} = \left( \frac{\| K \|_2^2}{\mu_2^2(K) \hat{S}_D(\alpha)} \right)^{1/5} n^{-1/5}. \tag{2.18}$$

Since $\hat{h}_{dpi}$ depends on the choice of the pilot bandwidth $\alpha$. One way of choosing $\alpha$ is to appeal to the formula for the AMSE-optimal bandwidth for estimation of $\hat{S}_D(\alpha)$. The value of $\alpha$ is

$$\alpha = D_1(L) \left( \| f''' \|_2^2 \right)^{-1/7} n^{-1/7} \tag{2.19}$$

where

$$D_1(L) = \{2L^{(4)}(0)/\sigma_L^2\}^{1/7}. \tag{2.20}$$

However, this rule for choosing $\alpha$ depends on an unknown density function, namely $\| f''' \|_2^2$. So it is apparent that the optimal bandwidth for estimating $\| f^{(r)} \|_2^2$ depends on $\| f^{(r+1)} \|_2^2$. The usual strategy for overcoming this problem is to estimate a $\| f^{(r)} \|_2^2$ using a quick and simple estimate, such as a version of the normal reference rule described above.

*Solve-the-Equation rule* (Sheather and Jones (1992))

Solve-the-equation rule requires that $h$ be chosen to satisfy the relationship

$$h = \left( \frac{\| K \|_2^2}{\mu_2^2(K) \hat{S}_D(\alpha(h))} \right)^{1/5} n^{-1/5}, \tag{2.21}$$

where the pilot bandwidth for the estimation of $\| f'' \|_2^2$ is a function $\alpha$ of $h$. The choice of $\alpha$ is given by

$$\alpha(h) = \left[ \frac{2L^{(4)}(0)\mu_2^2(K)}{\| K \|_2^2 \, \mu_2(L)} \right] \{ - \| \hat{f}''_{\alpha_1} \|_2^2 \, / \, \| \hat{f}'''_{\alpha_2} \|_2^2 \}^{1/7} h^{5/7} \tag{2.22}$$

where $\| \hat{f''}_{\alpha_1} \|_2^2$ and $\| \hat{f'''}_{\alpha_2} \|_2^2$ are kernel estimate of $\| f'' \|_2^2$ and $\| f''' \|_2^2$. The choice of $\alpha_1$ and $\alpha_2$ are made by minimizing AMSE.

Plug-in method improves the asymptotic rate of convergence of the estimated bandwidth to its theoretical (but practically unavailable) optimum value. Also it reduces the constant coefficient of the leading term in the expansion of the performance measure we use. Thirdly, it has a computational advantage and has reliable good characteristic to simulate the smooth densities.

## 2.2 Smooth Estimation Based on Bernstein Polynomials

### 2.2.1 Bernstein Polynomials

The Bernstein polynomials of degree $n$ are defined by

$$b_k(m, x) = \binom{m}{k} x^k (1-x)^{m-k}, \qquad x \in [0, 1].$$

for $k = 0, 1, \ldots, m$, where

$$\binom{m}{k} = \frac{m!}{k!(m-k)!}$$

There are $n+1$ $n$th-degree Bernstein polynomials. For mathematical convenience, we usually set $b_k(m, x) = 0$, if $k < 0$ or $k > m$.

By shifting and scaling, the Bernstein polynomials $b_k(m, x)$ can be used to approximate a function $f$ on any closed interval. The convergence of

Bernstein polynomials is slow for many practical approximation purposes, but they enjoy certain monotonicity and shape-preserving properties that make them useful.

### 2.2.2 Applications of Bernstein Polynomials

Babu, Canty and Chaubey (2002) considered the application of Bernstein polynomials for approximating a bounded and continuous function and showed that it can be naturally adapted for smooth estimation of a distribution function concentrated on the interval [0, 1] by a continuous approximation of the empirical distribution function. Suppose $X_1, \ldots, X_n$ are independent random variables, which have common c.d.f. $F$ and support $\mathbf{R}^+$. The estimator is based on smoothing the empirical distribution function, $F_{n_X}$ given by

$$F_{n_X}(x) = n^{-1} \sum_{i=1}^{n} I\{X_i \leq x\}.$$

Then, the following theorem is essential for using Bernstein Polynomials to smooth the empirical distribution function over the interval [0, 1].

**Theorem 2.2.1.** *(Feller, 1965, Theorem 1, Section VII.2).*
*If $u(x)$ is a bounded and continuous function on the interval [0, 1], then as $m \to \infty$*

$$u_m^*(x) = \sum_{k=0}^{m} u(k/m) b_k(m, x) \to u(x) \tag{2.23}$$

*uniformly for $x \in [0, 1]$.*

If c.d.f. $F$ has support [0, 1], Theorem 2.2.1 motivates the following smooth estimator which was introduced by Babu, Canty and Chaubey (2002)

$$\tilde{F}_{n,m}(x) = \sum_{k=0}^{m} F_{n_X}(k/m) b_k(m, x), \qquad x \in [0, 1]. \tag{2.24}$$

If $X$ is not in [0, 1], it is necessary to do some transformation such that $Y = g(X) \in [0, 1]$, where $g$ is an increasing differentiable function defined on the support of $F$. When $X$ is in compact interval [a, b], $a < b$, then $Y = (X - a)/(b - a)$ is valid to transform the random variable $X$ to $Y$. Similarly, if X has the support $[0, \infty)$ or $(-\infty, \infty)$, do the transformation $Y = X/(1 + X)$ or $Y = (1/2) + (1/\pi) \arctan X$, respectively.

For convenience, we call $\tilde{F}_{n,m}$ Babu distribution estimator.

### 2.2.3 Statistical Properties

Suppose that $x \in [0, 1]$. Then the $\tilde{F}_{n,m}$ has following properties (Babu *et al* ( 2002)):

1. $\tilde{F}_{n,m}$ is a polynomial in $x$, and has all the derivatives.
2. $\tilde{F}_{n,m}$ is continuous, $0 \leq \tilde{F}_{n,m}(x) \leq 1$ for $x \in [0, 1]$.
3. $\tilde{F}_{n,m}(x) = \sum_{k=0}^{m} f_{n_X}(k/m) B_k(m, x)$,

where

$$f_{n_X}(0) = 0, \quad f_{n_X}(k/m) = F_{n_X}(k/m) - F_{n_X}((k-1)/m), \quad k = 1, \ldots, m$$

and

$$B_k(m,x) = \sum_{j=k}^{m} b_j(m,x) = m\binom{m-1}{k-1}\int_0^x t^{k-1}(1-t)^{m-k}dt.$$

By definition, for all $k$, $f_{n_X}(k/m)$ is non-negative, and $B_k(m,x)$ is non-decreasing in $x$. Thus $\tilde{F}_{n,m}$ is non-decreasing in $x$.

4. The density $f$ of $F$ is given by

$$\begin{aligned}\tilde{f}_{n,m}(x) &= \sum_{k=1}^{m} f_{n_X}(k/m)\frac{d}{dx}B_k(m,x) \\ &= m\sum_{k=0}^{m-1} f_{n_X}((k+1)/m)b_k(m-1,x) \\ &= m\sum_{k=0}^{m-1}[F_{n_X}((k+1)/m) - F_{n_X}(k/m)]b_k(m-1,x) \quad (2.25)\end{aligned}$$

The following theorems show that Babu estimators $\tilde{F}_{n,m}$ and $\tilde{f}_{n,m}$ are strongly consistent.

**Theorem 2.2.2.** *(Babu, Canty and Chaubey (2002)) Let $F$ be a continuous probability distribution function on the interval [0, 1]. If $m, n \to \infty$, then*

$$\sup_{x\in[0,1]} |\tilde{F}_{n,m}(x) - F(x)| \to 0 \, a.s..$$

Recall that a function $h$ is said to be Lipschitz of order $\alpha$, if there exists a constant $C$ such that

$$|h(s) - h(t)| \leq C|s-t|^{\alpha}.$$

**Theorem 2.2.3.** *(Babu, Canty and Chaubey (2002)) If $f$ is Lipschitz of order 1, then for $2 \leq m \leq (n/\log n)$, we have a.s. as $n \to \infty$,*

$$\sup_{x\in[0,1]} |\tilde{f}_{n,m}(x) - f(x)| = O(m^{1/2}(n\log n)^{1/2}) + O(\sup_{x\in[0,1]} |F_m^{*'}(x) - f(x)|),$$

*where $F_m^{*'}(x)$ denotes the derivative of*

$$F_m^*(x) = \sum_{k=0}^{m} F(k/m) b_k(m,x) \tag{2.26}$$

*by definition (2.23). Consequently, if $m = o(n/\log n)$, then $\sup_{x\in[0,1]} |\tilde{f}_{n,m}(x) - f(x)| \to 0$ a.s. as $m, n \to \infty$.*

It is clear that the choice of $m$ is important in estimating the density and the distribution function. Various values of $m$ were tried and Babu *et al* suggested the choice of $m = n/\log n$.

The estimator proposed by Babu *et al.* (2002) improves the kernel density estimator to some extent. A simple illustrative example comes from estimating Gamma density in $[0, \infty)$. We plot the traditional kernel and Babu estimators based on Bernstein polynomials for sample sizes $n = 50, 200,$ and 700 for each of 10 simulated samples. The Gaussian kernel and normal reference bandwidth are used for kernel density estimator. As we saw in Figure 2.3, the traditional kernel estimator does not fit true density well due to smoothing, but the estimator based on the Bernstein polynomials can capture the density shape well.

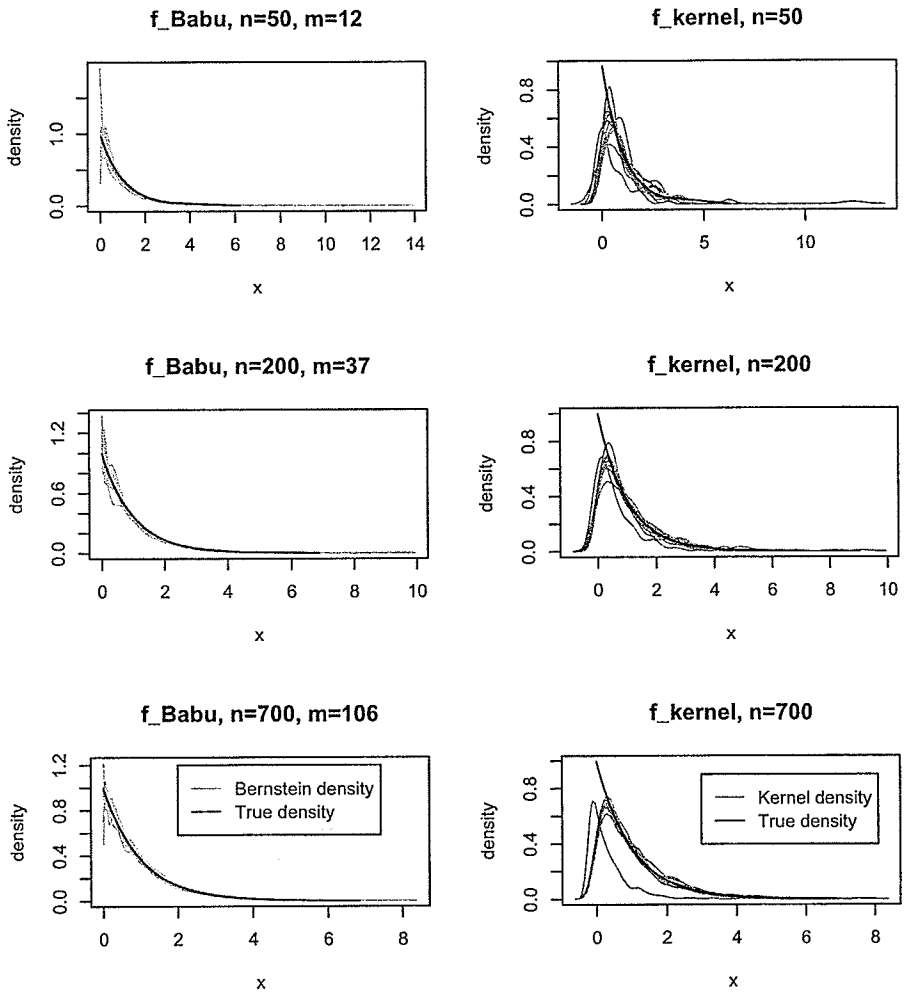


Figure 2.3: Bernstein and Kernel density estimates for $Exp(1)$

# Chapter 3

# Kernel Smoothing Based on Bernstein Polynomials

## 3.1 Motivation and Derivation

As indicated earlier, if we knew the support of population $X$, Babu, Canty and Chaubey (2002) suggested to use the following transformations in different cases to obtain the variables $Y_1, \ldots, Y_n$ with support [0, 1].

(1) Suppose X concentrates on a finite support [a, b], then $Y_i = (X_i - a)/(b - a)$, $i = 1, \ldots, n$ is widely used;

(2) For the non-negative support $[0, \infty)$, the transformation proposed is $Y_i = X_i/(1 + X_i)$;

(3) For the distributions on the interval $(-\infty, \infty)$, the transformed sample $Y_i = (1/2) + (1/\pi) \arctan X_i$ is acceptable.

However, there are many choices of transformation. Which one is the

best? Moveover, the support of $X$ is usually unknown. Then, how do we make an effective transformation $Y = g(X)$ in the general case?

First of all, from Theorem 2.2.1, we know that $u_m^*(x) \to u(x)$. If we suppose $u(x) = x$, that is $X \sim Unif(0,1)$, it is easy to prove $u_m^*(x) = u(x)$.

Second of all, we notice that $F_m^*(x) = F(x)$ and $\tilde{F}_{n,m}(x) \approx F(x)$ if $F(x) = x$, $x \in [0,1]$, or $X \sim Unif(0,1)$. Hence, $Y = g(X)$ will be close to the best transformation if $g(X)$ is close to the true underlying distribution $F(x)$, or if $Y$ is distributed closely to $Unif(0,1)$.

If $F_0(x)$ is a reasonable initial estimate of $F(x)$, an appropriate transformation will be $\hat{Y} = F_0(X)$, which is a data-driven transformation.

We use the kernel distribution function as the initial transformation, because kernel may extract all the important features of the data and is suitable for any type of support. We try to combine the kernel estimate and Bernstein polynomials to construct a new estimator.

Whether the original i.i.d. random variables $X_1, \ldots, X_n$ with finite or infinite support, do the following kernel transformation:

$$\hat{Y}_j = \hat{F}_h(X_j) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{K}\left(\frac{X_j - X_i}{h}\right), \qquad j = 1, \ldots, n$$

where, $\mathbb{K}$ is the kernel distribution function. Now we have the new i.i.d. variables $\hat{Y}_1, \ldots, \hat{Y}_n$, which have the support [0, 1]. The new estimator is still

based on smoothing the empirical distribution function, $F_{n_{\hat{Y}}}$ given by

$$\begin{aligned} F_{n_{\hat{Y}}}(y) &= n^{-1}\sum_{j=1}^{n} I\{\hat{Y}_j \le y\} \\ &= n^{-1}\sum_{j=1}^{n} I\{\hat{F}_h(X_j) \le y\} \\ &= n^{-1}\sum_{j=1}^{n} I\{X_j \le \hat{F}_h^{-1}(y)\} \\ &= F_{n_X}(\hat{F}_h^{-1}(y)), \qquad 0 \le y \le 1. \end{aligned}$$

Therefore, our new estimator $\hat{F}_{m,h}$ is obtained by:

$$\begin{aligned} \hat{F}_{m,h}(x) &= P(\widehat{X \le x}) = P(\hat{F}_h(\widehat{X}) \le \hat{F}_h(x)) \\ &= P(Y \le \widehat{\hat{F}_h}(x)) = \tilde{F}_{n,m}(\hat{F}_h(x)) \\ &= \sum_{k=0}^{m} F_{n_{\hat{Y}}}(k/m) b_k(m, \hat{F}_h(x)) \end{aligned}$$

## 3.2 Statistical Properties

Suppose that $x \in (-\infty, +\infty)$. Then the $\hat{F}_{m,h}$ has the following properties:

1. $\hat{F}_{m,h}$ is a polynomial in $\hat{F}_h(x)$, and has all the derivatives.
2. $\hat{F}_{m,h}$ is continuous, $0 \le \hat{F}_{m,h} \le 1$ for $x \in (-\infty, +\infty)$.
3. $\hat{F}_{m,h}(x) = \sum_{k=0}^{m} f_{n_{\hat{Y}}}(k/m) B_k(m, \hat{F}_h(x))$,

where

$$f_{n_{\hat{Y}}}(0) = 0,\ f_{n_{\hat{Y}}}(k/m) = F_{n_{\hat{Y}}}(k/m) - F_{n_{\hat{Y}}}((k-1)/m),\ k = 1, \ldots, m$$

and

$$B_k(m, \hat{F}_h(x)) = \sum_{j=k}^{m} b_j(m,x) = m\binom{m-1}{k-1}\int_0^{\hat{F}_h(x)} t^{k-1}(1-t)^{m-k}dt.$$

By definition, for all $k$, $f_{n_{\hat{Y}}}(k/m)$ is non-negative, and $B_k(m, \hat{F}_h(x))$ is non-decreasing in $\hat{F}_h(x)$. Thus $\hat{F}_{m,h}$ is non-decreasing in $\hat{F}_h(x)$.

4. The corresponding density estimator of $\hat{F}_{m,h}$ is given by

$$\hat{f}_{m,h}(x) = \tilde{f}_{n,m}(\hat{F}_h(x)) \cdot \hat{f}_h(x) \tag{3.1}$$

where $\tilde{f}_{n,m}(\cdot)$ is given in (2.25), and $\hat{f}_h(x)$ is the kernel density estimation:

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

Therefore, representation (3.1) leads to the explicit formula

$$\begin{aligned}\hat{f}_{m,h}(x) &= m\sum_{k=0}^{m-1}[F_{n_{\hat{Y}}}((k+1)/m) - F_{n_{\hat{Y}}}(k/m)]b_k(m-1, \hat{F}_h(x)) \\ &\quad \cdot\frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)\end{aligned}$$

By property 4, the new density estimator $\hat{f}_{m,h}$ can be viewed as adjusted kernel density estimator. The "adjusted factor" is $\tilde{f}_{n,m}(\hat{F}_h(x))$. Therefore, the new estimator is determined by the choice of both bandwidth and $m$.

Now we establish a strong convergence results for $\hat{F}_{m,h}$ and $\hat{f}_{m,h}$. Throughout the underlying theorems, we assume that a continuous probability distribution function $F$ has density $f$, and that $f$ is Lipschitz of order 1.

**Theorem 3.2.1.** *If $m, n \to \infty$ and $m \leq O(n^{1/2})$, then $\sup_{x\in(-\infty,+\infty)} |\hat{F}_{m,h}(x) - F(x)| \to 0$ a.s..*

**Proof.**

We first recall that $\hat{F}_{m,h}(x) = \tilde{F}_{n,m}(\hat{F}_h(x))$, we have

$$\begin{aligned}
& \sup_{x\in(-\infty,+\infty)} |\tilde{F}_{n,m}(\hat{F}_h(x)) - F(x)| \\
= & \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} F_{n_{\hat{Y}}}(k/m) b_k(m, \hat{F}_h(x)) - \sum_{k=0}^{m} (k/m) b_k(m, F(x))| \\
= & \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} F_{n_{\hat{Y}}}(k/m) b_k(m, \hat{F}_h(x)) - \sum_{k=0}^{m} F_{n_Y}(k/m) b_k(m, F(x)) \\
& + \sum_{k=0}^{m} F_{n_Y}(k/m) b_k(m, F(x)) - \sum_{k=0}^{m} (k/m) b_k(m, F(x))| \\
\leq & \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} F_{n_{\hat{Y}}}(k/m) b_k(m, \hat{F}_h(x)) - \sum_{k=0}^{m} F_{n_Y}(k/m) b_k(m, F(x))| \\
& + \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} F_{n_Y}(k/m) b_k(m, F(x)) - \sum_{k=0}^{m} (k/m) b_k(m, F(x))| \\
= & A_{n,m} + B_{n,m}, \; say.
\end{aligned}$$

Note that $F_{n_{\hat{Y}}}(k/m) = F_{n_Y}(k/m)$, so

$$\begin{aligned}
A_{n,m} &= \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} F_{n_Y}(k/m) b_k(m, \hat{F}_h(x)) - \sum_{k=0}^{m} F_{n_Y}(k/m) b_k(m, F(x))| \\
&= \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} [b_k(m, \hat{F}_h(x)) - b_k(m, F(x))] F_{n_Y}(k/m)| \\
&= \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} b'_k(m, \xi(x)) [\hat{F}_h(x) - F(x)] F_{n_Y}(k/m)| \\
&= \sup_{x\in(-\infty,+\infty)} |\tilde{f}_{n,m}(\xi(x)) \cdot (\hat{F}_h(x) - F(x))|
\end{aligned}$$

where $\xi(x)$ is the value between $\hat{F}_h(x)$ and $F(x)$.

By property of kernel, we know that $\sup_{x\in(-\infty,+\infty)} |\hat{F}_h(x) - F(x)| \to 0$

a.s. as $n \to \infty$. So our attention turns to prove $\tilde{f}_{n,m}(\xi(x))$ is bounded if we want to get $A_{n,m} \to 0$ a.s..

Recall that $Y = F(X) \sim Unif(0,1)$, we write

$$\begin{aligned}
\tilde{f}_{n,m}(\xi(x)) &= m\sum_{k=0}^{m-1}[F_{n_Y}((k+1)/m) - F_{n_Y}(k/m)]b_k(m-1,\xi(x)) \\
&= m\sum_{k=0}^{m-1}[F_{n_Y}((k+1)/m) - F((k+1)/m) - (F_{n_Y}(k/m) - F(k/m)) \\
&\quad +F((k+1)/m) - F(k/m)]b_k(m-1,\xi(x)) \\
&= m\sum_{k=0}^{m-1}[F_{n_Y}((k+1)/m) - (k+1)/m]b_k(m-1,\xi(x)) \\
&\quad -m\sum_{k=0}^{m-1}[F_{n_Y}(k/m) - k/m]b_k(m-1,\xi(x)) \\
&\quad +m\sum_{k=0}^{m-1}[(k+1)/m - k/m]b_k(m-1,\xi(x))
\end{aligned}$$

It is well known that the limit distribution of empirical distribution function follows the standard normal distribution, that is

$$n^{1/2}\frac{[F_n(x) - F(x)]}{\sqrt{F(x)(1-F(x))}} \to N(0,1)$$

in probability.

So it is easy to show that $F_{n_Y}(k/m) - F(k/m) = O_p\left(\frac{m}{\sqrt{n}}\right)$, $k = 0, 1, \ldots, m$. Therefore, when $m \leq O(n^{1/2})$ and $\sum_{k=0}^{m-1} b_k(m-1,\xi(x)) = 1$, then $\tilde{f}_{n,m}(\xi(x)) \leq O_p\left(\frac{m}{\sqrt{n}}\right) + O_p\left(\frac{m}{\sqrt{n}}\right) + 1$, that is $\tilde{f}_{n,m}(\xi(x))$ is bounded, as $n \to \infty$.

Furthermore,

$$\begin{aligned} B_{n,m} &= \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} F_{n_Y}(k/m) b_k(m, F(x)) - \sum_{k=0}^{m} (k/m) b_k(m, F(x))| \\ &= \sup_{x\in(-\infty,+\infty)} |\sum_{k=0}^{m} [F_{n_Y}(k/m) - k/m] b_k(m, F(x))| \\ &\leq \max_{0\leq k\leq m} |F_{n_Y}(k/m) - k/m| \cdot \sum_{k=0}^{m} b_k(m, F(x)) \\ &\leq \sup_{y\in[0,1]} |F_{n_Y}(y) - F(y)| \end{aligned}$$

By Glivenko-Cantelli theorem, it is straightforward to get $\sup_{y\in[0,1]} |F_{n_Y}(y) - F(y)| \to 0$ a.s. as $n \to \infty$. Therefore, we have $B_{n,m} \to 0$ a.s..

Now we can conclude that Theorem 3.2.1 is valid.

**Theorem 3.2.2.** *If $m, n \to \infty$ and $m \leq O(n^{1/2})$, then $\sup_{x\in(-\infty,+\infty)} |\hat{f}_{m,h}(x) - f(x)| \to 0$ a.s..*

**Proof.**

From equation 3.1, we have

$$\begin{aligned} \sup_{x\in(-\infty,+\infty)} |\hat{f}_{m,h}(x) - f(x)| &= \sup_{x\in(-\infty,+\infty)} |\tilde{f}_{n,m}(\hat{F}_h(x)) \cdot \hat{f}_h(x) - f(x)| \\ &= \sup_{x\in(-\infty,+\infty)} |(\tilde{f}_{n,m}(\hat{F}_h(x)) - 1)\hat{f}_h(x) + \hat{f}_h(x) - f(x)| \\ &\leq \sup_{x\in(-\infty,+\infty)} |(\tilde{f}_{n,m}(\hat{F}_h(x)) - 1)\hat{f}_h(x)| \\ &\quad + \sup_{x\in(-\infty,+\infty)} |\hat{f}_h(x) - f(x)| \\ &= C_{n,m} + D_{n,m}, \text{ say.} \end{aligned}$$

By property of kernel, we know that $D_{n,m} \to 0$ a.s. as $n \to \infty$. So we only

need to prove $\tilde{f}_{n,m}(\hat{F}_h(x)) \to 1$ uniformly in order to get $C_{n,m} \to 0$ a.s..

$$\begin{aligned}\tilde{f}_{n,m}(\hat{F}_h(x)) &= [\tilde{f}_{n,m}(\hat{F}_h(x)) - \tilde{f}_{n,m}(F(x))] + \tilde{f}_{n,m}(F(x)) \\ &= \tilde{f}'_{n,m}(\xi(x))[\hat{F}_h(x) - F(x)] + \tilde{f}_{n,m}(F(x))\end{aligned}$$

where $\xi(x)$ is the value between $\hat{F}_h(x)$ and $F(x)$.

We have known that $\sup_{x\in(-\infty,+\infty)} |\hat{F}_h(x) - F(x)| \to 0$ a.s. as $n \to \infty$, and $\tilde{f}'_{n,m}(\xi(x))$ is bounded. So, the first term of above equation goes to zero as $n \to \infty$.

By property of Babu estimator, $\tilde{f}_{n,m}(F(x)) \to 1$ a.s. as $n \to \infty$. Thus, $\tilde{f}_{n,m}(\hat{F}_h(x)) \to 1$ a.s. as $n \to \infty$.

Now we can conclude that Theorem 3.2.2 is valid.

## 3.3 Examples

Two examples will be given to illustrate the methods of different smoothing estimations. The software used for computation is *R*. The data set and programs are attached in Appendix I.

**Example 1**

This data set comprises the lengths in days of 86 spells of psychiatric treatment undergone by patients used as controls in a study of suicide risks (source: Copas and Fryer, 1980; Silverman, 1986). Figure 3.1(a) shows the plots of Babu density estimator based on Bernstein polynomials, traditional

kernel density estimator and kernel density estimator based on Bernstein polynomials using the Gaussian kernel and direct-plug-in window width by Park and Marron (1992).

The density for the length of treatment seems like a unimodal curve skewed to right. It is shown that traditional kernel density estimator slightly oversmooths the true density curve. Babu estimator undersmooths the curve too much. The magnitude of new kernel density estimator based on Bernstein polynomials looks moderate and well balances the two situations.

**Example 2**

This data set consists of 107 eruption lengths in minutes for the Old Faithful geyser in Yellowstone National Park, USA (source: Weisberg, 1980; Silverman, 1986). Figure 3.1(b) displays the plots of three smoothing methods mentioned in Example 1.

It is obvious that the density curve for eruption length is bimodal, and looks more like normal mixture. The traditional kernel density estimator intends to oversmooth the true density curve. Babu estimator oversmooth the curve too much, and can not capture the characteristics of bimodal curve. New kernel density estimator appears reasonable and robust.

## 3.4 Graphical Illustrations

For graphical illustrations, we obtain simulated samples from three kinds of distribution families: (1) normal mixture, (2) Beta distribution and (3) Gamma distribution, from $n$=50, 200 and 700. Figures 3.2–3.7 compare the smoothing estimations to the true density by graph.

(1) Traditional kernel estimation $\hat{f}_h$: $f_{kernel}$;

(2) Babu estimation based on Bernstein polynomials $\tilde{f}_{n,m}$: $f_{Babu}$;

(3) New kernel estimation based on Bernstein polynomials $\hat{f}_{m,h}$: $f_{new}$.

**Normal Mixture**

The family of normal mixture is so flexible that it can well approximate any density. For simplicity, we only consider $0.5N(0,1)+0.5N(\mu,\sigma^2)$ for each combination of $\mu = 0, 5, 15$ and $\sigma = 1, 1/3, 1/10$.

As we see, these three estimation methods work well when the distribution is unimodal. However, for the bimodal situation, the kernel estimation tends to oversmooth the peaks and valleys, and the Babu estimation only fit one peak well. Our new estimation looks more accurate, even the curve is sharp.

**Beta Distribution**

Beta density can take variant shapes due to different combinations of parameter $\alpha$ and $\beta$. For simplicity, we only consider a standard Beta distri-

bution $B(\alpha, \beta)$ for partial combinations of $\alpha, \beta = 0.8, 1, 2$ and 5, since the density is symmetric to the positions of $\alpha$ and $\beta$.

It is obvious that three approaches can not catch the shape of true density curve at the boundaries even for large sample sizes. For kernel based estimate, this is due to the bandwidth selection. In $R$, we chose the fixed bandwidth, which is not suitable for convex and monotone functions. Especially for Beta(0.8, 0.8), we need smaller bandwidth around the boundaries 0 and 1, than that in the body. So, the modification of bandwidth, such as variable bandwidth or adjusted bandwidth, will construct better estimator. On the other hand, to uniform density function, that is Beta(1, 1), the three estimation approaches fit as the nonlinear curves.

**Gamma Distribution**

Gamma distribution, say $\Gamma(s, t)$, is representative to the non-negative distribution family, where $s$ is shape parameter, and $t$ is scale parameter. Without loss of generality, we choose scale parameter $t = 1$, and only consider the cases of $s = 0.7, 1$, and 3.

It can be seen that when shape parameter $a < 1$, that is, the density curve goes up to infinity around boundary 0, but these three smoothing methods can not capture this infinite trend of the curve. When $a \geq 1$, we find that Babu and new kernel density estimator both fit better than traditional kernel density estimator, which is often oversmooths the curve.

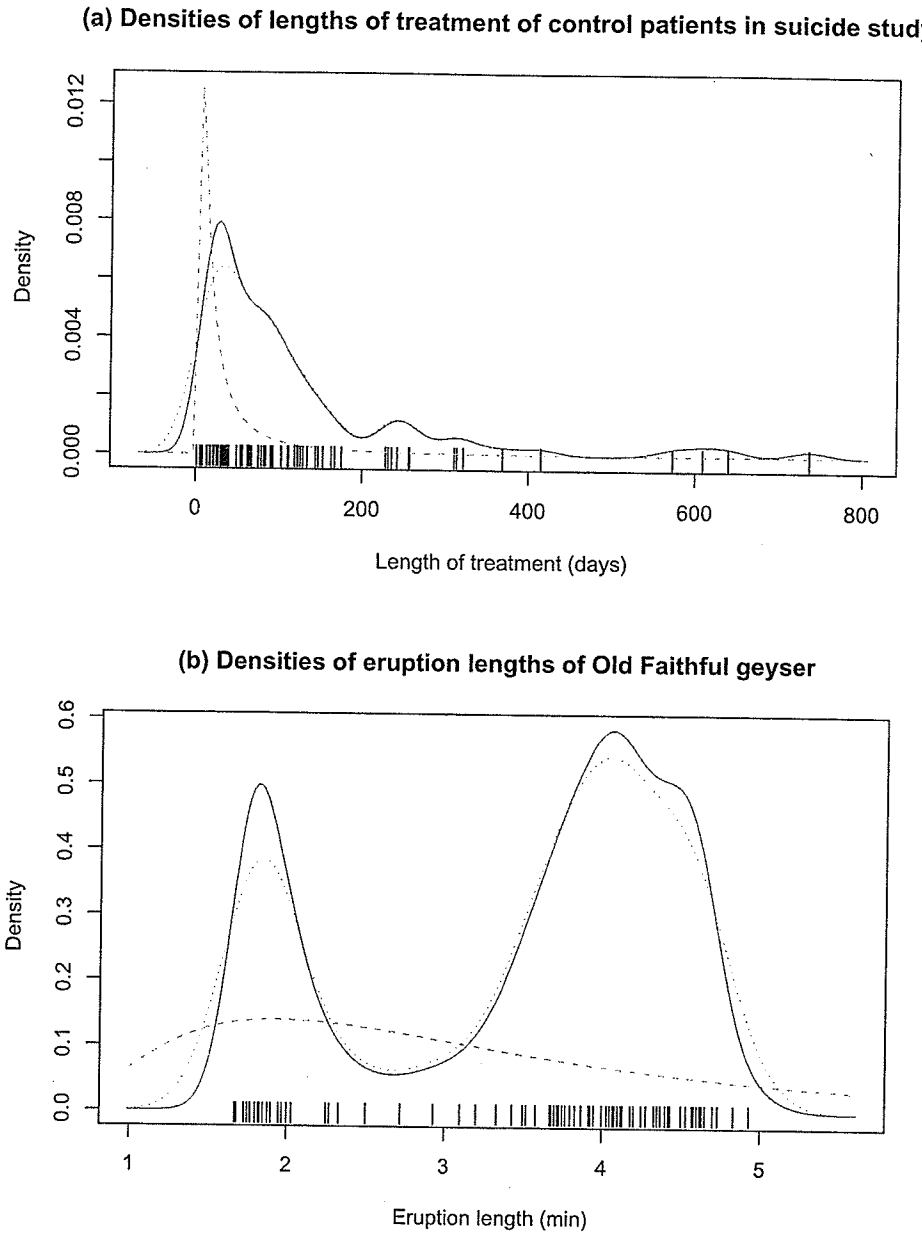


Figure 3.1: Bernstein and Kernel Density Estimations.

—— $f_{new}$; - - - - - $f_{Babu}$; ····· $f_{kernel}$

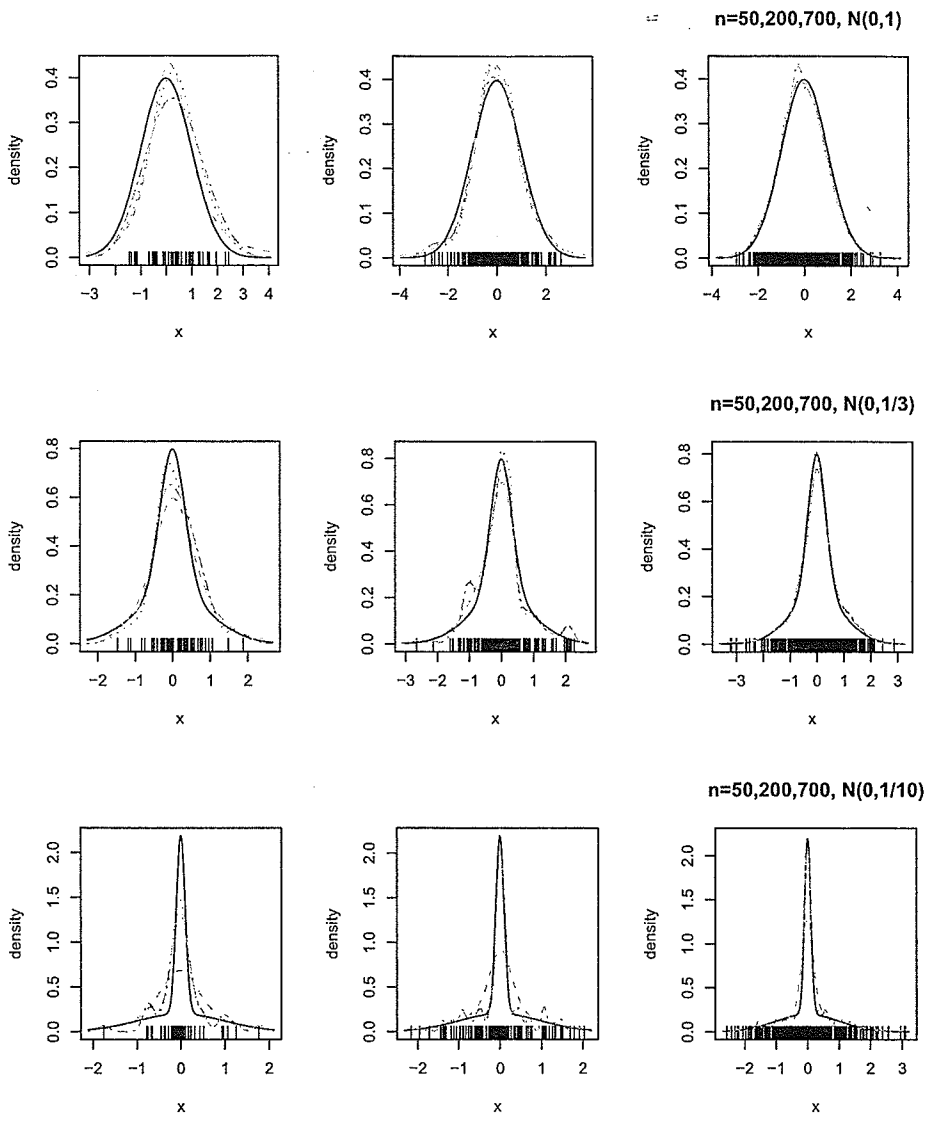


Figure 3.2: Bernstein and Kernel Density Estimations for Normal(0, 1), Normal(0, 1/3) and Normal(0, 1/10) for simulated samples of size $n$ =50,200 and 700.

—— $f_{true}$; — · — · — $f_{kernel}$; - - - - - $f_{Babu}$; ····· $f_{new}$

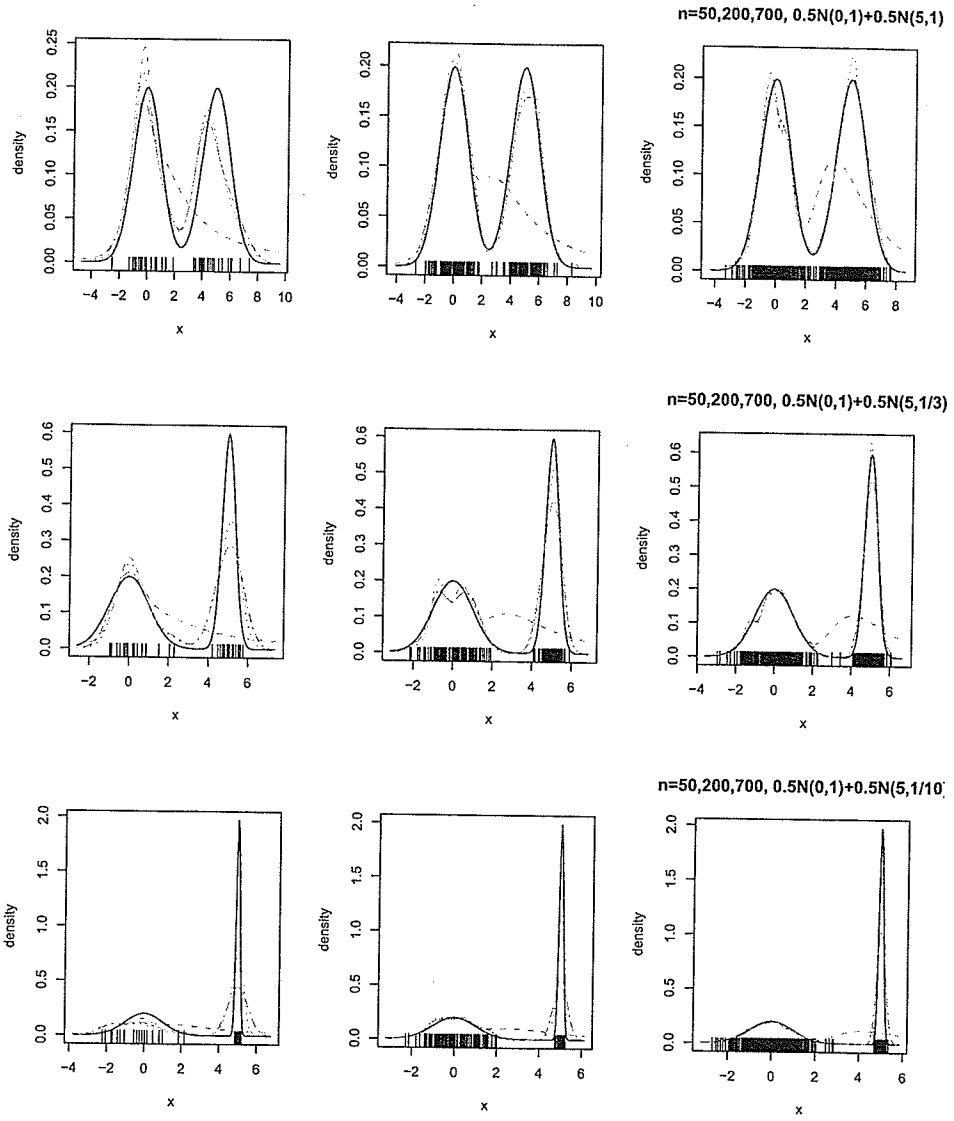


Figure 3.3: Bernstein and Kernel Density Estimations for $0.5N(0,1) + 0.5N(5,1)$, $0.5N(0,1) + 0.5N(5,1/3)$ and $0.5N(0,1) + 0.5N(5,1/10)$ for simulated samples of size $n$ =50,200 and 700.

—— $f_{true}$; — · — · — $f_{kernel}$; - - - - - $f_{Babu}$; ····· $f_{new}$

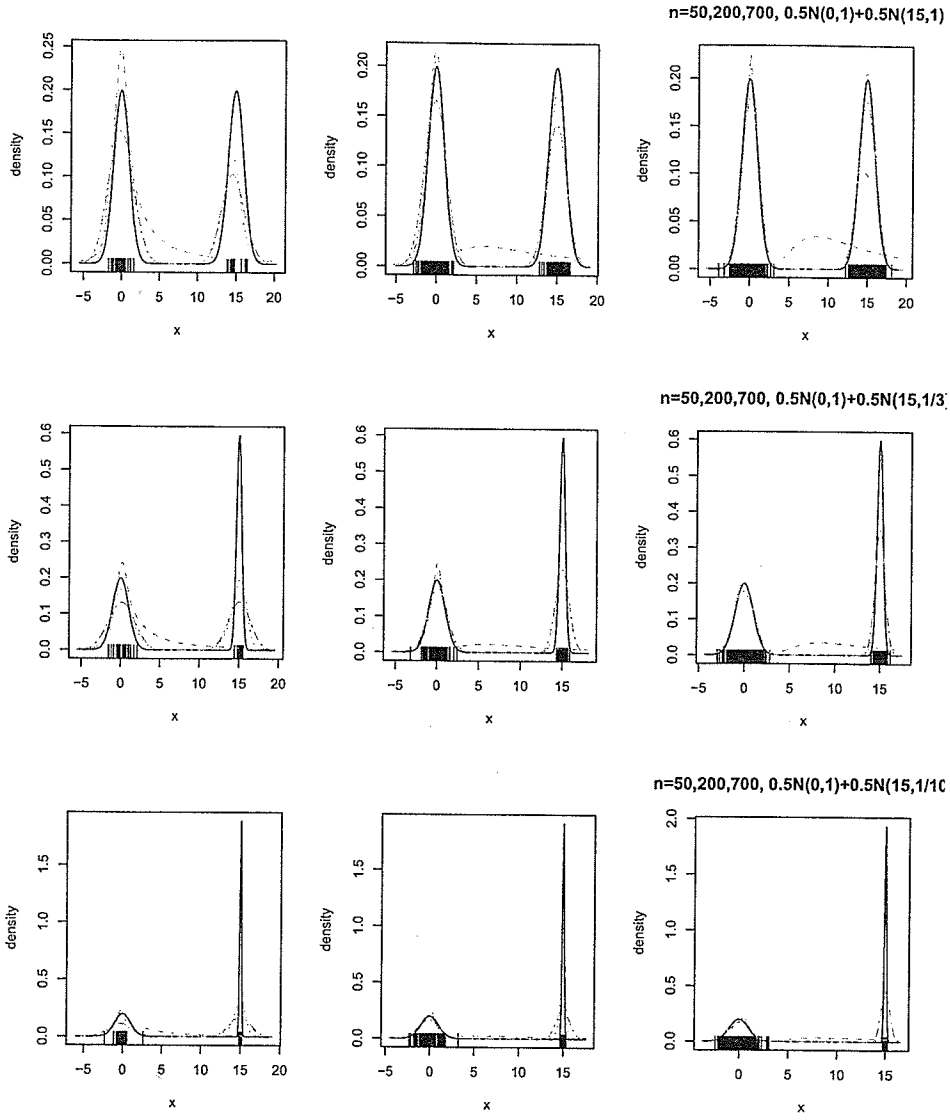


Figure 3.4: Bernstein and Kernel density estimations for $0.5N(0,1) + 0.5N(15,1)$, $0.5N(0,1) + 0.5N(15,1/3)$ and $0.5N(0,1) + 0.5N(15,1/10)$ for simulated samples of size $n$ =50,200 and 700.

—— $f_{true}$; — · — · — $f_{kernel}$; - - - - - $f_{Babu}$; ····· $f_{new}$

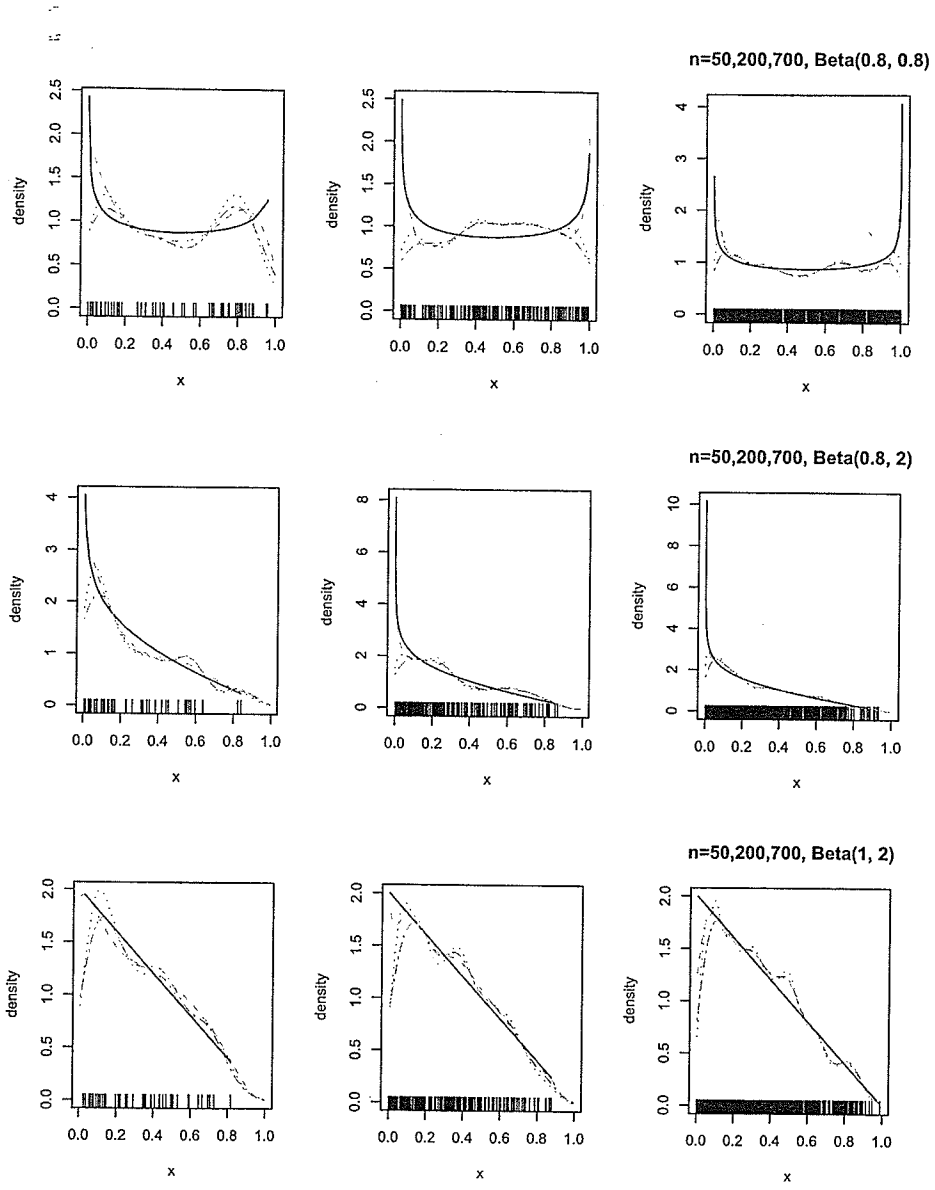


Figure 3.5: Bernstein and Kernel Density Estimations for Beta(0.8, 0.8), Beta(0.8, 2) and Beta(1, 2) for simulated samples of size $n$ =50,200 and 700.

—— $f_{true}$; — · — · — $f_{kernel}$; - - - - - $f_{Babu}$; ····· $f_{new}$

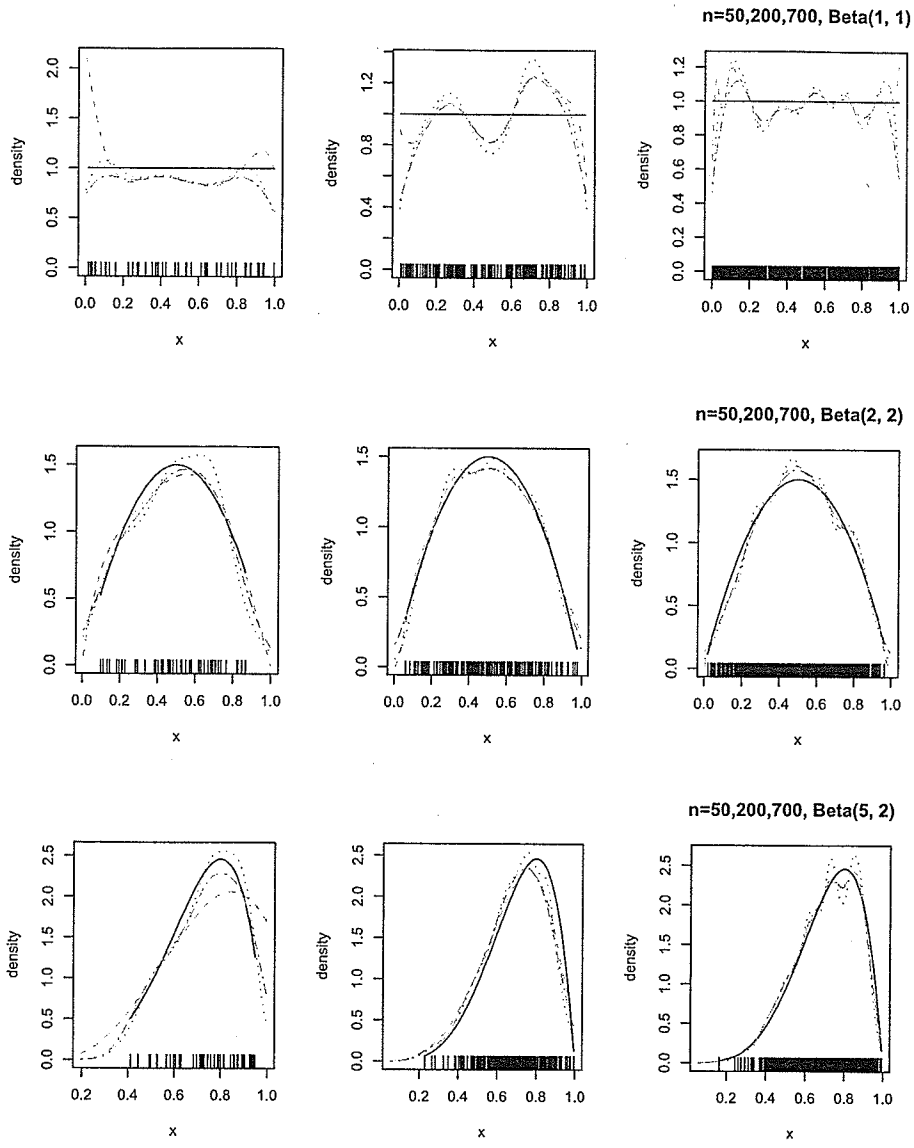


Figure 3.6: Bernstein and Kernel Density Estimations for Beta(1, 1), Beta(2, 2) and Beta(5, 2) for simulated samples of size $n$ =50,200 and 700.

—— $f_{true}$; — · — · — $f_{kernel}$; - - - - - $f_{Babu}$; ····· $f_{new}$

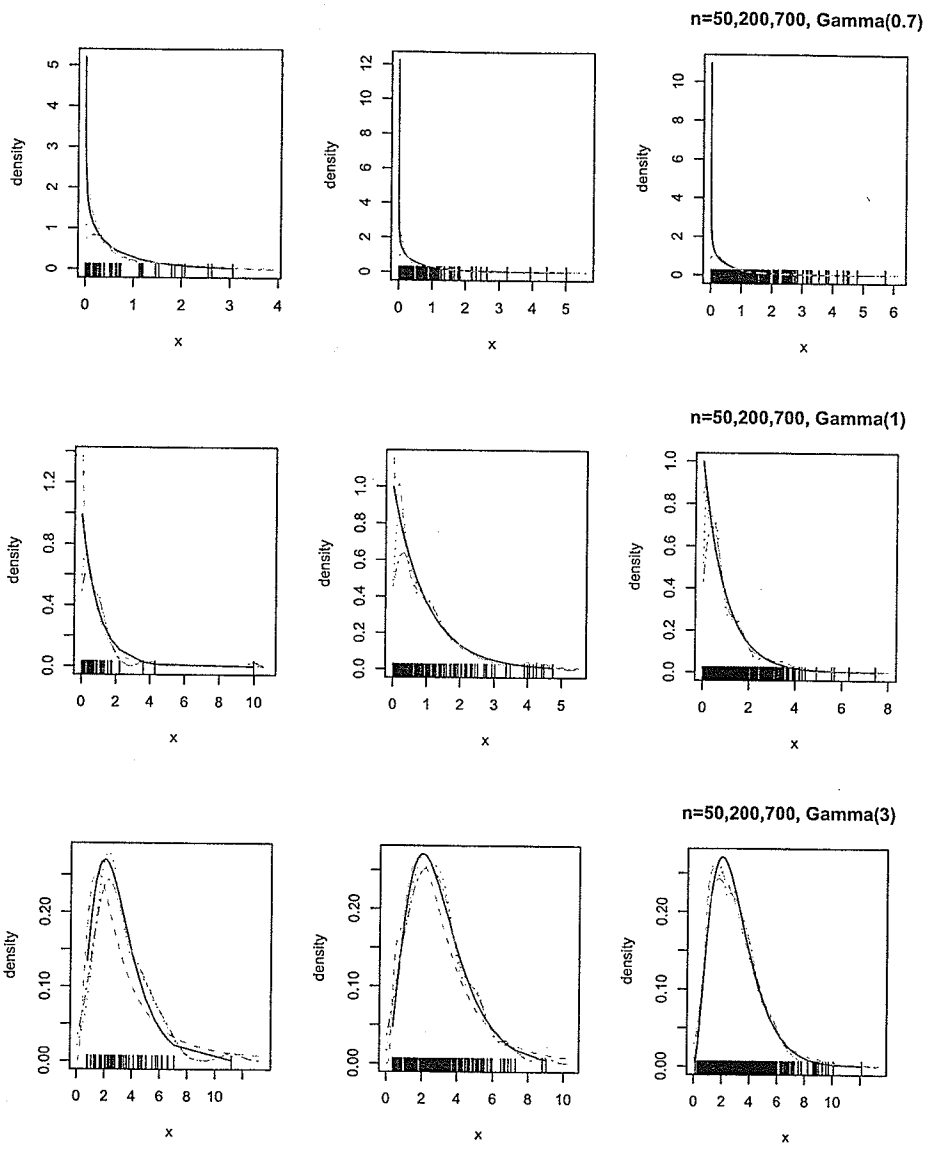


Figure 3.7: Bernstein and Kernel Density Estimations for $\Gamma(0.7)$, $\Gamma(1)$ and $\Gamma(3)$ for simulated samples of size $n$ =50,200 and 700.

—— $f_{true}$; — · — · — $f_{kernel}$; - - - - - $f_{Babu}$; ····· $f_{new}$

# Chapter 4

# Numerical Illustrations

## 4.1 Simulations

We have had the visual comparisons of three smoothing methods. Next, we will give the numerical illustrations in terms of the MISE and MIAE criteria. Using Monte Carlo simulation method, we can evaluate the $MISE_{\hat{f}}$, $MIAE_{\hat{f}}$ of

(1) Traditional kernel density estimation $\hat{f}_h$: $f_{kernel}$;

(2) Babu density estimation based on Bernstein polynomials $\tilde{f}_{n,m}$: $f_{Babu}$;

(3) New kernel density estimation based on Bernstein polynomials $\hat{f}_{m,h}$: $f_{new}$,

and the $MISE_{\hat{F}}$ and $MIAE_{\hat{F}}$ of

(1) Traditional kernel distribution estimation $\hat{F}_h$: $F_{kernel}$;

(2) Babu distribution estimation based on Bernstein polynomials $\tilde{F}_{n,m}$:

$F_{Babu}$;

(3) New kernel distribution estimation based on Bernstein polynomials $\hat{F}_{m,h}$: $F_{new}$.

We also give their efficiencies relative to the traditional kernel estimation.

In the following cases, $N = 5000$ Monte Carlo samples of size $n = 50, 200$ and 700 are generated from different distributions. The kernel estimation we will use is based on Gaussian kernel and direct-plug-in bandwidth introduced by Park and Marron (1992), which is considered as the best existing method. New kernel estimation based on Bernstein polynomials still uses the choice of $m = n/\log n$ in order to compare with Babu estimation clearly. The software used for computation is $R$.

In practice, it is very difficult to calculate integral to obtain the MISE and MIAE criteria. To avoid the numerical computation of integral, we develop the idea of double expectation to approximate $MISE_{\hat{f}}$, $MIAE_{\hat{f}}$, $MISE_{\hat{F}}$ and $MIAE_{\hat{F}}$.

Suppose $X, X_1, \ldots, X_n$ are i.i.d. random variables, having the common

p.d.f. $f(x)$, we have

$$\begin{aligned} MISE_{\hat{f}} &= E\int[\hat{f}(x)-f(x)]^2dx \\ &= E\int\frac{[\hat{f}(x)-f(x)]^2}{f(x)}f(x)dx \\ &= E\left\{E\left[\frac{[\hat{f}(X)-f(X)]^2}{f(X)}|X_1,\ldots,X_n\right]\right\} \\ &= E\left[\frac{[\hat{f}(X)-f(X)]^2}{f(X)}\right] \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\frac{[\hat{f}_{(i)}(X^{(i)})-f(X^{(i)})]^2}{f(X^{(i)})}. \end{aligned}$$

The last expression is based on Monte Carlo samples $X^{(i)}, X_1^{(i)}, \ldots, X_n^{(i)}$ $(i = 1, 2, \ldots, N)$ from $f(x)$. Using double expectation, the integral $\int[\hat{f}(x) - f(x)]^2dx$ under the first "E" (with respect to $X_1, \ldots, X_n$) is incorporated into the last "E" (with respect to $X, X_1, \ldots, X_n$).

Similarly,

$$\begin{aligned} MIAE_{\hat{f}} &= E\int|\hat{f}(x)-f(x)|dx \\ &= E\int\frac{|\hat{f}(x)-f(x)|}{f(x)}f(x)dx \\ &= E\left\{E\left[\left|\frac{\hat{f}(X)}{f(X)}-1\right||X_1,\ldots,X_n\right]\right\} \\ &= E\left|\frac{\hat{f}(X)}{f(X)}-1\right| \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{f}_{(i)}(X^{(i)})}{f(X^{(i)})}-1\right|, \end{aligned}$$

and

$$\begin{aligned} MISE_{\hat{F}} &= E\int[\hat{F}(x)-F(x)]^2 f(x)dx \\ &\approx \frac{1}{N}\sum_{i=1}^{N}[\hat{F}_{(i)}(X^{(i)})-F(X^{(i)})]^2 \end{aligned}$$

$$\begin{aligned} MIAE_{\hat{F}} &= E\int|\hat{F}(x)-F(x)|f(x)dx \\ &\approx \frac{1}{N}\sum_{i=1}^{N}|\hat{F}_{(i)}(X^{(i)})-F(X^{(i)})|. \end{aligned}$$

### 4.1.1 Normal Mixture

Tables 4.1––4.8 give the simulation results for the density of normal mixture $0.5N(0,1)+0.5N(\mu,\sigma^2)$ for each combination of $\mu$ =0, 5, 15 and $\sigma$ =1, 1/3, 1/10. In unimodal case with $\mu=0$, the new kernel-based estimator and the traditional kernel estimator have the same efficiency, except the standard normal case, whose MISE and MIAE value may be slightly enlarged. In bimodal case, it is clear that the new kernel-based estimator is the best one of the three estimators in that the reduction of MISE and MIAE criteria is significant, especially in $MISE_{\hat{f}}$ criterion. On the other hand, when $\mu$ is fixed and $\sigma$ goes to zero (the true density has sharp features which are hard to estimate), the new kernel estimator works stably. When $\sigma$ is fixed and $\mu$ goes to infinity, it still keeps the good performance.

### 4.1.2 Beta Distribution

Tables 4.9−−4.16 give the simulation results for the density of $Beta(\alpha, \beta)$ for partial combinations of $\alpha, \beta = 0.8, 1, 2$ and 5. It can be seen that Babu estimator performs best of three approaches, since the support of variables of standard Beta distribution is exact [0, 1], that means Babu method can use the original observations directly without any transformation. On the other hand, it is well-known that kernel-based estimates have boundary problems, which makes traditional and new kernel-based estimators have lower efficiencies in both MISE and MIAE crtieria. In $MISE_{\hat{f}}$ criterion, the difference between the efficiency of Babu estimator and kernel-based estimator (either traditional or new one) is obvious. But in other criteria, the difference is small.

### 4.1.3 Gamma Distribution

Tables 4.17−−4.24 give the simulation results for the density of Gamma$(s, t)$ with combinations $s$ =0.7, 1, 3 and $t$ =1. In $MISE_{\hat{f}}$ and $MISE_{\hat{F}}$ criteria, when $s \leq 1$, the efficiencies of Babu estimator relative to traditional kernel estimator are much better than those of the two kernel-based estimators, and the new kernel-based estimator is more effective than traditional one. When $s > 1$, the traditional kernel estimator turns to the best estimator, and the new kernel-based estimator is somewhat better than the Babu estimator. In $MIAE_{\hat{f}}$ and $MIAE_{\hat{F}}$ criteria, Babu estimator obtains the best efficiencies of

three approaches when $s \leq 1$, but the worst one when $s > 1$. The traditional and new kernel-based estimators obtain the equal efficiencies no matter what the value of $s$ is.

| | $MISE_{\hat{f}}$ | | |
|---|---|---|---|
| n | $MISE_{f_{kernel}}(MISE_{\hat{f}_h})$ | $MISE_{f_{Babu}}(MISE_{\tilde{f}_{n,m}})$ | $MISE_{f_{new}}(MISE_{\hat{f}_{m,h}})$ |
| | $\mu = 0 \quad \sigma = 1$ | | |
| 50 | 0.01090 | 0.00884 | 0.01410 |
| 200 | 0.00383 | 0.00380 | 0.00593 |
| 700 | 0.00148 | 0.00175 | 0.00285 |
| | $\mu = 0 \quad \sigma = 1/3$ | | |
| 50 | 0.02660 | 0.02620 | 0.02820 |
| 200 | 0.00909 | 0.00786 | 0.01090 |
| 700 | 0.00355 | 0.00259 | 0.00501 |
| | $\mu = 0 \quad \sigma = 1/10$ | | |
| 50 | 0.0994 | 0.420 | 0.0857 |
| 200 | 0.0324 | 0.278 | 0.0284 |
| 700 | 0.0119 | 0.133 | 0.0127 |
| | $\mu = 5 \quad \sigma = 1$ | | |
| 50 | 0.01090 | 0.0517 | 0.01050 |
| 200 | 0.00352 | 0.0414 | 0.00364 |
| 700 | 0.00138 | 0.0215 | 0.00171 |
| | $\mu = 5 \quad \sigma = 1/3$ | | |
| 50 | 0.05240 | 0.190 | 0.03270 |
| 200 | 0.01920 | 0.176 | 0.00770 |
| 700 | 0.00608 | 0.153 | 0.00334 |
| | $\mu = 5 \quad \sigma = 1/10$ | | |
| 50 | 0.429 | 0.688 | 0.3250 |
| 200 | 0.296 | 0.655 | 0.0751 |
| 700 | 0.165 | 0.623 | 0.0101 |
| | $\mu = 15 \quad \sigma = 1$ | | |
| 50 | 0.02450 | 0.0761 | 0.01440 |
| 200 | 0.00730 | 0.0610 | 0.00329 |
| 700 | 0.00214 | 0.0543 | 0.00145 |
| | $\mu = 15 \quad \sigma = 1/3$ | | |
| 50 | 0.1250 | 0.215 | 0.0864 |
| 200 | 0.0750 | 0.204 | 0.0142 |
| 700 | 0.0328 | 0.193 | 0.0031 |
| | $\mu = 15 \quad \sigma = 1/10$ | | |
| 50 | 0.597 | 0.720 | 0.5330 |
| 200 | 0.493 | 0.702 | 0.2700 |
| 700 | 0.422 | 0.688 | 0.0457 |

Table 4.1: $MISE_{\hat{f}}$'s of different density estimators for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| | Efficiency | | |
|---|---|---|---|
| n | $\frac{MISE_{f_{kernel}}}{MISE_{f_{kernel}}}$ | $\frac{MISE_{f_{kernel}}}{MISE_{f_{Babu}}}$ | $\frac{MISE_{f_{kernel}}}{MISE_{f_{new}}}$ |
| | $\mu = 0 \quad \sigma = 1$ | | |
| 50 | 1 | 1.240 | 0.775 |
| 200 | 1 | 1.010 | 0.646 |
| 700 | 1 | 0.847 | 0.519 |
| | $\mu = 0 \quad \sigma = 1/3$ | | |
| 50 | 1 | 1.01 | 0.942 |
| 200 | 1 | 1.16 | 0.834 |
| 700 | 1 | 1.37 | 0.709 |
| | $\mu = 0 \quad \sigma = 1/10$ | | |
| 50 | 1 | 0.2370 | 1.160 |
| 200 | 1 | 0.1170 | 1.140 |
| 700 | 1 | 0.0896 | 0.939 |
| | $\mu = 5 \quad \sigma = 1$ | | |
| 50 | 1 | 0.2100 | 1.040 |
| 200 | 1 | 0.0849 | 0.967 |
| 700 | 1 | 0.0642 | 0.809 |
| | $\mu = 5 \quad \sigma = 1/3$ | | |
| 50 | 1 | 0.2760 | 1.60 |
| 200 | 1 | 0.1090 | 2.49 |
| 700 | 1 | 0.0398 | 1.82 |
| | $\mu = 5 \quad \sigma = 1/10$ | | |
| 50 | 1 | 0.623 | 1.32 |
| 200 | 1 | 0.453 | 3.95 |
| 700 | 1 | 0.265 | 16.30 |
| | $\mu = 15 \quad \sigma = 1$ | | |
| 50 | 1 | 0.3220 | 1.70 |
| 200 | 1 | 0.1200 | 2.22 |
| 700 | 1 | 0.0394 | 1.48 |
| | $\mu = 15 \quad \sigma = 1/3$ | | |
| 50 | 1 | 0.581 | 1.44 |
| 200 | 1 | 0.368 | 5.29 |
| 700 | 1 | 0.170 | 10.50 |
| | $\mu = 15 \quad \sigma = 1/10$ | | |
| 50 | 1 | 0.829 | 1.12 |
| 200 | 1 | 0.702 | 1.83 |
| 700 | 1 | 0.613 | 9.24 |

Table 4.2: Efficiencies of different density estimators by $MISE_{\hat{f}}$ criterion for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| | $MIAE_{\hat{f}}$ | | |
|---|---|---|---|
| n | $MIAE_{f_{kernel}}(MIAE_{\hat{f}_h})$ | $MIAE_{f_{Babu}}(MIAE_{\tilde{f}_{n,m}})$ | $MIAE_{f_{new}}(MIAE_{\hat{f}_{m,h}})$ |
| | | $\mu = 0 \quad \sigma = 1$ | |
| 50 | 0.1970 | 0.2040 | 0.2080 |
| 200 | 0.1140 | 0.1450 | 0.1330 |
| 700 | 0.0699 | 0.0765 | 0.0892 |
| | | $\mu = 0 \quad \sigma = 1/3$ | |
| 50 | 0.2600 | 0.5060 | 0.257 |
| 200 | 0.1490 | 0.1290 | 0.155 |
| 700 | 0.0921 | 0.0737 | 0.103 |
| | | $\mu = 0 \quad \sigma = 1/10$ | |
| 50 | 0.422 | 0.621 | 0.398 |
| 200 | 0.248 | 0.495 | 0.238 |
| 700 | 0.157 | 0.319 | 0.156 |
| | | $\mu = 5 \quad \sigma = 1$ | |
| 50 | 0.2730 | 0.572 | 0.2570 |
| 200 | 0.1550 | 0.495 | 0.1510 |
| 700 | 0.0943 | 0.372 | 0.0981 |
| | | $\mu = 5 \quad \sigma = 1/3$ | |
| 50 | 0.419 | 0.672 | 0.344 |
| 200 | 0.247 | 0.725 | 0.171 |
| 700 | 0.137 | 0.615 | 0.110 |
| | | $\mu = 5 \quad \sigma = 1/10$ | |
| 50 | 0.629 | 0.715 | 0.609 |
| 200 | 0.428 | 0.660 | 0.265 |
| 700 | 0.386 | 0.644 | 0.131 |
| | | $\mu = 15 \quad \sigma = 1$ | |
| 50 | 0.530 | 0.766 | 0.3730 |
| 200 | 0.259 | 0.594 | 0.1550 |
| 700 | 0.132 | 0.544 | 0.0969 |
| | | $\mu = 15 \quad \sigma = 1/3$ | |
| 50 | 0.629 | 0.771 | 0.506 |
| 200 | 0.432 | 0.587 | 0.211 |
| 700 | 0.293 | 0.541 | 0.111 |
| | | $\mu = 15 \quad \sigma = 1/10$ | |
| 50 | 0.716 | 0.701 | 0.649 |
| 200 | 0.550 | 0.586 | 0.450 |
| 700 | 0.487 | 0.542 | 0.201 |

Table 4.3: $MIAE_{\hat{f}}$'s of different density estimators for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| n | Efficiency $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{kernel}}}$ | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{Babu}}}$ | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{new}}}$ |
|---|---|---|---|
| | $\mu = 0 \quad \sigma = 1$ | | |
| 50 | 1 | 0.966 | 0.944 |
| 200 | 1 | 0.783 | 0.856 |
| 700 | 1 | 0.914 | 0.784 |
| | $\mu = 0 \quad \sigma = 1/3$ | | |
| 50 | 1 | 0.513 | 1.010 |
| 200 | 1 | 1.160 | 0.958 |
| 700 | 1 | 1.250 | 0.892 |
| | $\mu = 0 \quad \sigma = 1/10$ | | |
| 50 | 1 | 0.680 | 1.06 |
| 200 | 1 | 0.501 | 1.04 |
| 700 | 1 | 0.492 | 1.00 |
| | $\mu = 5 \quad \sigma = 1$ | | |
| 50 | 1 | 0.477 | 1.060 |
| 200 | 1 | 0.314 | 1.030 |
| 700 | 1 | 0.254 | 0.962 |
| | $\mu = 5 \quad \sigma = 1/3$ | | |
| 50 | 1 | 0.623 | 1.22 |
| 200 | 1 | 0.341 | 1.45 |
| 700 | 1 | 0.223 | 1.25 |
| | $\mu = 5 \quad \sigma = 1/10$ | | |
| 50 | 1 | 0.879 | 1.03 |
| 200 | 1 | 0.648 | 1.61 |
| 700 | 1 | 0.599 | 2.94 |
| | $\mu = 15 \quad \sigma = 1$ | | |
| 50 | 1 | 0.692 | 1.42 |
| 200 | 1 | 0.435 | 1.67 |
| 700 | 1 | 0.242 | 1.36 |
| | $\mu = 15 \quad \sigma = 1/3$ | | |
| 50 | 1 | 0.816 | 1.24 |
| 200 | 1 | 0.736 | 2.04 |
| 700 | 1 | 0.542 | 2.65 |
| | $\mu = 15 \quad \sigma = 1/10$ | | |
| 50 | 1 | 1.020 | 1.10 |
| 200 | 1 | 0.939 | 1.22 |
| 700 | 1 | 0.898 | 2.42 |

Table 4.4: Efficiencies of different density estimators by $MIAE_{\hat{f}}$ criterion for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| | $MISE_{\hat{F}}$ | | |
|---|---|---|---|
| n | $MISE_{F_{kernel}}(MISE_{\hat{F}_h})$ | $MISE_{F_{Babu}}(MISE_{\tilde{F}_{n,m}})$ | $MISE_{F_{new}}(MISE_{\hat{F}_{m,h}})$ |
| | $\mu = 0 \quad \sigma = 1$ | | |
| 50 | 0.002580 | 0.002150 | 0.002680 |
| 200 | 0.000744 | 0.000654 | 0.000738 |
| 700 | 0.000239 | 0.000212 | 0.000226 |
| | $\mu = 0 \quad \sigma = 1/3$ | | |
| 50 | 0.002900 | 0.003400 | 0.002910 |
| 200 | 0.000783 | 0.000854 | 0.000760 |
| 700 | 0.000237 | 0.000233 | 0.000218 |
| | $\mu = 0 \quad \sigma = 1/10$ | | |
| 50 | 0.003240 | 0.00977 | 0.003130 |
| 200 | 0.000835 | 0.00457 | 0.000790 |
| 700 | 0.000238 | 0.00170 | 0.000223 |
| | $\mu = 5 \quad \sigma = 1$ | | |
| 50 | 0.002730 | 0.00827 | 0.002810 |
| 200 | 0.000744 | 0.00459 | 0.000730 |
| 700 | 0.000227 | 0.00212 | 0.000215 |
| | $\mu = 5 \quad \sigma = 1/3$ | | |
| 50 | 0.003850 | 0.01110 | 0.003300 |
| 200 | 0.001070 | 0.00927 | 0.000742 |
| 700 | 0.000315 | 0.00640 | 0.000228 |
| | $\mu = 5 \quad \sigma = 1/10$ | | |
| 50 | 0.00779 | 0.01210 | 0.006250 |
| 200 | 0.00415 | 0.01090 | 0.001300 |
| 700 | 0.00180 | 0.00922 | 0.000248 |
| | $\mu = 15 \quad \sigma = 1$ | | |
| 50 | 0.004110 | 0.02060 | 0.003470 |
| 200 | 0.001080 | 0.00960 | 0.000774 |
| 700 | 0.000288 | 0.00912 | 0.000206 |
| | $\mu = 15 \quad \sigma = 1/3$ | | |
| 50 | 0.00718 | 0.0202 | 0.005280 |
| 200 | 0.00321 | 0.0108 | 0.000983 |
| 700 | 0.00118 | 0.0113 | 0.000227 |
| | $\mu = 15 \quad \sigma = 1/10$ | | |
| 50 | 0.01100 | 0.0215 | 0.009940 |
| 200 | 0.00767 | 0.0111 | 0.003810 |
| 700 | 0.00549 | 0.0117 | 0.000588 |

Table 4.5: $MISE_{\hat{F}}$'s of different density estimators for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| Efficiency | | | |
|---|---|---|---|
| n | $\frac{MISE_{F_{kernel}}}{MISE_{F_{kernel}}}$ | $\frac{MISE_{F_{kernel}}}{MISE_{F_{Babu}}}$ | $\frac{MISE_{F_{kernel}}}{MISE_{F_{new}}}$ |
| $\mu = 0 \quad \sigma = 1$ | | | |
| 50 | 1 | 1.20 | 0.964 |
| 200 | 1 | 1.14 | 1.010 |
| 700 | 1 | 1.13 | 1.060 |
| $\mu = 0 \quad \sigma = 1/3$ | | | |
| 50 | 1 | 0.852 | 0.997 |
| 200 | 1 | 0.918 | 1.030 |
| 700 | 1 | 1.020 | 1.090 |
| $\mu = 0 \quad \sigma = 1/10$ | | | |
| 50 | 1 | 0.331 | 1.04 |
| 200 | 1 | 0.183 | 1.06 |
| 700 | 1 | 0.140 | 1.06 |
| $\mu = 5 \quad \sigma = 1$ | | | |
| 50 | 1 | 0.330 | 0.97 |
| 200 | 1 | 0.162 | 1.02 |
| 700 | 1 | 0.107 | 1.06 |
| $\mu = 5 \quad \sigma = 1/3$ | | | |
| 50 | 1 | 0.3470 | 1.17 |
| 200 | 1 | 0.1160 | 1.45 |
| 700 | 1 | 0.0492 | 1.38 |
| $\mu = 5 \quad \sigma = 1/10$ | | | |
| 50 | 1 | 0.642 | 1.25 |
| 200 | 1 | 0.382 | 3.19 |
| 700 | 1 | 0.195 | 7.27 |
| $\mu = 15 \quad \sigma = 1$ | | | |
| 50 | 1 | 0.1990 | 1.18 |
| 200 | 1 | 0.1130 | 1.40 |
| 700 | 1 | 0.0315 | 1.40 |
| $\mu = 15 \quad \sigma = 1/3$ | | | |
| 50 | 1 | 0.356 | 1.36 |
| 200 | 1 | 0.298 | 3.26 |
| 700 | 1 | 0.104 | 5.19 |
| $\mu = 15 \quad \sigma = 1/10$ | | | |
| 50 | 1 | 0.510 | 1.10 |
| 200 | 1 | 0.694 | 2.01 |
| 700 | 1 | 0.468 | 9.33 |

Table 4.6: Efficiencies of different density estimators by $MISE_{\hat{F}}$ criterion for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| | $MIAE_{\hat{F}}$ | | |
|---|---|---|---|
| n | $MIAE_{F_{kernel}}(MIAE_{\hat{F}_h})$ | $MIAE_{F_{Babu}}(MIAE_{\hat{F}_{n,m}})$ | $MIAE_{F_{new}}(MIAE_{\hat{F}_{m,h}})$ |
| | | $\mu = 0 \quad \sigma = 1$ | |
| 50 | 0.0399 | 0.0372 | 0.0404 |
| 200 | 0.0211 | 0.0205 | 0.0209 |
| 700 | 0.0119 | 0.0112 | 0.0113 |
| | | $\mu = 0 \quad \sigma = 1/3$ | |
| 50 | 0.041 | 0.0474 | 0.0413 |
| 200 | 0.021 | 0.0226 | 0.0205 |
| 700 | 0.012 | 0.0120 | 0.0116 |
| | | $\mu = 0 \quad \sigma = 1/10$ | |
| 50 | 0.0431 | 0.0795 | 0.0423 |
| 200 | 0.0214 | 0.0520 | 0.0210 |
| 700 | 0.0117 | 0.0311 | 0.0114 |
| | | $\mu = 5 \quad \sigma = 1$ | |
| 50 | 0.0402 | 0.0690 | 0.0401 |
| 200 | 0.0214 | 0.0516 | 0.0209 |
| 700 | 0.0119 | 0.0344 | 0.0115 |
| | | $\mu = 5 \quad \sigma = 1/3$ | |
| 50 | 0.0484 | 0.0787 | 0.0436 |
| 200 | 0.0261 | 0.0665 | 0.0209 |
| 700 | 0.0142 | 0.0561 | 0.0114 |
| | | $\mu = 5 \quad \sigma = 1/10$ | |
| 50 | 0.0702 | 0.0841 | 0.0636 |
| 200 | 0.0476 | 0.0718 | 0.0274 |
| 700 | 0.0305 | 0.0622 | 0.0117 |
| | | $\mu = 15 \quad \sigma = 1$ | |
| 50 | 0.0529 | 0.0996 | 0.0450 |
| 200 | 0.0269 | 0.0702 | 0.0210 |
| 700 | 0.0140 | 0.0628 | 0.0114 |
| | | $\mu = 15 \quad \sigma = 1/3$ | |
| 50 | 0.0692 | 0.0977 | 0.0584 |
| 200 | 0.0452 | 0.0725 | 0.0248 |
| 700 | 0.0269 | 0.0687 | 0.0116 |
| | | $\mu = 15 \quad \sigma = 1/10$ | |
| 50 | 0.0828 | 0.1030 | 0.0770 |
| 200 | 0.0655 | 0.0741 | 0.0472 |
| 700 | 0.0507 | 0.0687 | 0.0188 |

Table 4.7: $MIAE_{\hat{F}}$'s of different density estimators for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| Efficiency | | | |
|---|---|---|---|
| n | $\frac{MIAE_{F_{kernel}}}{MIAE_{F_{kernel}}}$ | $\frac{MIAE_{F_{kernel}}}{MIAE_{F_{Babu}}}$ | $\frac{MIAE_{F_{kernel}}}{MIAE_{F_{new}}}$ |
| $\mu = 0 \quad \sigma = 1$ | | | |
| 50 | 1 | 1.07 | 0.989 |
| 200 | 1 | 1.03 | 1.010 |
| 700 | 1 | 1.06 | 1.060 |
| $\mu = 0 \quad \sigma = 1/3$ | | | |
| 50 | 1 | 0.866 | 0.994 |
| 200 | 1 | 0.927 | 1.020 |
| 700 | 1 | 1.000 | 1.030 |
| $\mu = 0 \quad \sigma = 1/10$ | | | |
| 50 | 1 | 0.542 | 1.02 |
| 200 | 1 | 0.412 | 1.02 |
| 700 | 1 | 0.374 | 1.02 |
| $\mu = 5 \quad \sigma = 1$ | | | |
| 50 | 1 | 0.583 | 1.00 |
| 200 | 1 | 0.414 | 1.02 |
| 700 | 1 | 0.345 | 1.03 |
| $\mu = 5 \quad \sigma = 1/3$ | | | |
| 50 | 1 | 0.615 | 1.11 |
| 200 | 1 | 0.392 | 1.25 |
| 700 | 1 | 0.253 | 1.25 |
| $\mu = 5 \quad \sigma = 1/10$ | | | |
| 50 | 1 | 0.835 | 1.10 |
| 200 | 1 | 0.663 | 1.74 |
| 700 | 1 | 0.490 | 2.60 |
| $\mu = 15 \quad \sigma = 1$ | | | |
| 50 | 1 | 0.531 | 1.17 |
| 200 | 1 | 0.383 | 1.28 |
| 700 | 1 | 0.223 | 1.23 |
| $\mu = 15 \quad \sigma = 1/3$ | | | |
| 50 | 1 | 0.709 | 1.19 |
| 200 | 1 | 0.623 | 1.82 |
| 700 | 1 | 0.392 | 2.32 |
| $\mu = 15 \quad \sigma = 1/10$ | | | |
| 50 | 1 | 0.803 | 1.08 |
| 200 | 1 | 0.883 | 1.39 |
| 700 | 1 | 0.738 | 2.70 |

Table 4.8: Efficiencies of different density estimators by $MIAE_{\hat{F}}$ criterion for normal mixture $0.5N(0,1) + 0.5N(\mu, \sigma^2)$

| | $MISE_{\hat{f}}$ | | |
|---|---|---|---|
| n | $MISE_{f_{kernel}}(MISE_{\hat{f}_h})$ | $MISE_{f_{Babu}}(MISE_{\tilde{f}_{n,m}})$ | $MISE_{f_{new}}(MISE_{\hat{f}_{m,h}})$ |
| | $\alpha = 0.8 \quad \beta = 0.8$ | | |
| 50 | 0.1220 | 0.0645 | 0.1370 |
| 200 | 0.0911 | 0.0341 | 0.0861 |
| 700 | 0.0654 | 0.0179 | 0.0503 |
| | $\alpha = 0.8 \quad \beta = 2$ | | |
| 50 | 0.230 | 0.0812 | 0.2230 |
| 200 | 0.174 | 0.0499 | 0.1410 |
| 700 | 0.114 | 0.0234 | 0.0758 |
| | $\alpha = 1 \quad \beta = 2$ | | |
| 50 | 0.0912 | 0.0397 | 0.107 |
| 200 | 0.0492 | 0.0208 | 0.051 |
| 700 | 0.0273 | 0.0113 | 0.024 |
| | $\alpha = 1 \quad \beta = 1$ | | |
| 50 | 0.0603 | 0.0415 | 0.0815 |
| 200 | 0.0324 | 0.0225 | 0.0378 |
| 700 | 0.0185 | 0.0111 | 0.0186 |
| | $\alpha = 2 \quad \beta = 2$ | | |
| 50 | 0.0410 | 0.03870 | 0.0605 |
| 200 | 0.0149 | 0.01570 | 0.0263 |
| 700 | 0.0059 | 0.00859 | 0.0128 |
| | $\alpha = 5 \quad \beta = 2$ | | |
| 50 | 0.0729 | 0.08490 | 0.0957 |
| 200 | 0.0282 | 0.02410 | 0.0399 |
| 700 | 0.0106 | 0.00896 | 0.0185 |

Table 4.9: $MISE_{\hat{f}}$'s of different density estimators for Beta distribution, $B(\alpha, \beta)$

| Efficiency | | | |
|---|---|---|---|
| n | $\frac{MISE_{f_{kernel}}}{MISE_{f_{kernel}}}$ | $\frac{MISE_{f_{kernel}}}{MISE_{f_{Babu}}}$ | $\frac{MISE_{f_{kernel}}}{MISE_{f_{new}}}$ |
| $\alpha = 0.8 \quad \beta = 0.8$ | | | |
| 50 | 1 | 1.89 | 0.888 |
| 200 | 1 | 2.67 | 1.060 |
| 700 | 1 | 3.66 | 1.300 |
| $\alpha = 0.8 \quad \beta = 2$ | | | |
| 50 | 1 | 2.83 | 1.03 |
| 200 | 1 | 3.48 | 1.23 |
| 700 | 1 | 4.88 | 1.51 |
| $\alpha = 1 \quad \beta = 2$ | | | |
| 50 | 1 | 2.30 | 0.852 |
| 200 | 1 | 2.37 | 0.964 |
| 700 | 1 | 2.41 | 1.140 |
| $\alpha = 1 \quad \beta = 1$ | | | |
| 50 | 1 | 1.45 | 0.739 |
| 200 | 1 | 1.44 | 0.857 |
| 700 | 1 | 1.67 | 0.994 |
| $\alpha = 2 \quad \beta = 2$ | | | |
| 50 | 1 | 1.060 | 0.677 |
| 200 | 1 | 0.946 | 0.567 |
| 700 | 1 | 0.688 | 0.462 |
| $\alpha = 5 \quad \beta = 2$ | | | |
| 50 | 1 | 0.859 | 0.762 |
| 200 | 1 | 1.170 | 0.705 |
| 700 | 1 | 1.190 | 0.574 |

Table 4.10: Efficiencies of different density estimators by $MISE_{\hat{f}}$ criterion for Beta distribution, $B(\alpha, \beta)$

| | $MIAE_{\hat{f}}$ | | |
|---|---|---|---|
| n | $MIAE_{f_{kernel}}(MIAE_{\hat{f}_h})$ | $MIAE_{f_{Babu}}(MIAE_{\tilde{f}_{n,m}})$ | $MIAE_{f_{new}}(MIAE_{\hat{f}_{m,h}})$ |
| | $\alpha = 0.8 \quad \beta = 0.8$ | | |
| 50 | 0.226 | 0.1670 | 0.251 |
| 200 | 0.166 | 0.1210 | 0.169 |
| 700 | 0.122 | 0.0865 | 0.117 |
| | $\alpha = 0.8 \quad \beta = 2$ | | |
| 50 | 0.240 | 0.149 | 0.247 |
| 200 | 0.168 | 0.106 | 0.166 |
| 700 | 0.125 | 0.078 | 0.116 |
| | $\alpha = 1 \quad \beta = 2$ | | |
| 50 | 0.2080 | 0.145 | 0.2240 |
| 200 | 0.1360 | 0.102 | 0.1470 |
| 700 | 0.0918 | 0.074 | 0.0986 |
| | $\alpha = 1 \quad \beta = 1$ | | |
| 50 | 0.1870 | 0.1530 | 0.2200 |
| 200 | 0.1310 | 0.1120 | 0.1450 |
| 700 | 0.0924 | 0.0808 | 0.0983 |
| | $\alpha = 2 \quad \beta = 2$ | | |
| 50 | 0.1660 | 0.1570 | 0.1960 |
| 200 | 0.0976 | 0.0996 | 0.1280 |
| 700 | 0.0613 | 0.0722 | 0.0855 |
| | $\alpha = 5 \quad \beta = 2$ | | |
| 50 | 0.1860 | 0.2170 | 0.2050 |
| 200 | 0.1090 | 0.1020 | 0.1290 |
| 700 | 0.0684 | 0.0637 | 0.0893 |

Table 4.11: $MIAE_{\hat{f}}$'s of different density estimators for Beta distribution, $B(\alpha, \beta)$

| Efficiency | | | |
|---|---|---|---|
| n | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{kernel}}}$ | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{Babu}}}$ | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{new}}}$ |
| $\alpha = 0.8 \quad \beta = 0.8$ | | | |
| 50 | 1 | 1.35 | 0.902 |
| 200 | 1 | 1.37 | 0.979 |
| 700 | 1 | 1.42 | 1.050 |
| $\alpha = 0.8 \quad \beta = 2$ | | | |
| 50 | 1 | 1.61 | 0.973 |
| 200 | 1 | 1.59 | 1.020 |
| 700 | 1 | 1.61 | 1.080 |
| $\alpha = 1 \quad \beta = 2$ | | | |
| 50 | 1 | 1.44 | 0.927 |
| 200 | 1 | 1.34 | 0.927 |
| 700 | 1 | 1.24 | 0.931 |
| $\alpha = 1 \quad \beta = 1$ | | | |
| 50 | 1 | 1.22 | 0.848 |
| 200 | 1 | 1.17 | 0.904 |
| 700 | 1 | 1.14 | 0.940 |
| $\alpha = 2 \quad \beta = 2$ | | | |
| 50 | 1 | 1.050 | 0.847 |
| 200 | 1 | 0.980 | 0.760 |
| 700 | 1 | 0.848 | 0.717 |
| $\alpha = 5 \quad \beta = 2$ | | | |
| 50 | 1 | 0.858 | 0.908 |
| 200 | 1 | 1.080 | 0.849 |
| 700 | 1 | 1.070 | 0.766 |

Table 4.12: Efficiencies of different density estimators by $MIAE_{\hat{f}}$ criterion for Beta distribution, $B(\alpha, \beta)$

| | $MISE_{\hat{F}}$ | | |
|---|---|---|---|
| n | $MISE_{F_{kernel}}(MISE_{\hat{F}_h})$ | $MISE_{F_{Babu}}(MISE_{\tilde{F}_{n,m}})$ | $MISE_{F_{new}}(MISE_{\hat{F}_{m,h}})$ |
| | $\alpha = 0.8 \quad \beta = 0.8$ | | |
| 50 | 0.002520 | 0.002180 | 0.002740 |
| 200 | 0.000713 | 0.000654 | 0.000737 |
| 700 | 0.000232 | 0.000209 | 0.000224 |
| | $\alpha = 0.8 \quad \beta = 2$ | | |
| 50 | 0.002770 | 0.002320 | 0.002860 |
| 200 | 0.000844 | 0.000683 | 0.000790 |
| 700 | 0.000265 | 0.000207 | 0.000228 |
| | $\alpha = 1 \quad \beta = 2$ | | |
| 50 | 0.002570 | 0.002060 | 0.002700 |
| 200 | 0.000734 | 0.000638 | 0.000739 |
| 700 | 0.000227 | 0.000201 | 0.000219 |
| | $\alpha = 1 \quad \beta = 1$ | | |
| 50 | 0.002550 | 0.002200 | 0.002820 |
| 200 | 0.000718 | 0.000679 | 0.000767 |
| 700 | 0.000221 | 0.000211 | 0.000225 |
| | $\alpha = 2 \quad \beta = 2$ | | |
| 50 | 0.0383 | 0.0352 | 0.0393 |
| 200 | 0.0204 | 0.0191 | 0.0202 |
| 700 | 0.0113 | 0.0106 | 0.0110 |
| | $\alpha = 5 \quad \beta = 2$ | | |
| 50 | 0.002500 | 0.002160 | 0.002740 |
| 200 | 0.000666 | 0.000622 | 0.000709 |
| 700 | 0.000208 | 0.000197 | 0.000213 |

Table 4.13: $MISE_{\hat{F}}$'s of different density estimators for Beta distribution, $B(\alpha, \beta)$

| Efficiency | | | |
|---|---|---|---|
| n | $\frac{MISE_{F_{kernel}}}{MISE_{F_{kernel}}}$ | $\frac{MISE_{F_{kernel}}}{MISE_{F_{Babu}}}$ | $\frac{MISE_{F_{kernel}}}{MISE_{F_{new}}}$ |
| $\alpha = 0.8 \quad \beta = 0.8$ | | | |
| 50 | 1 | 1.16 | 0.919 |
| 200 | 1 | 1.09 | 0.968 |
| 700 | 1 | 1.11 | 1.040 |
| $\alpha = 0.8 \quad \beta = 2$ | | | |
| 50 | 1 | 1.19 | 0.968 |
| 200 | 1 | 1.24 | 1.070 |
| 700 | 1 | 1.28 | 1.170 |
| $\alpha = 1 \quad \beta = 2$ | | | |
| 50 | 1 | 1.25 | 0.954 |
| 200 | 1 | 1.15 | 0.993 |
| 700 | 1 | 1.13 | 1.040 |
| $\alpha = 1 \quad \beta = 1$ | | | |
| 50 | 1 | 1.16 | 0.906 |
| 200 | 1 | 1.06 | 0.935 |
| 700 | 1 | 1.05 | 0.985 |
| $\alpha = 2 \quad \beta = 2$ | | | |
| 50 | 1 | 1.09 | 0.974 |
| 200 | 1 | 1.07 | 1.010 |
| 700 | 1 | 1.07 | 1.020 |
| $\alpha = 5 \quad \beta = 2$ | | | |
| 50 | 1 | 1.16 | 0.913 |
| 200 | 1 | 1.07 | 0.940 |
| 700 | 1 | 1.05 | 0.975 |

Table 4.14: Efficiencies of different density estimators by $MIAE_{\hat{f}}$ criterion for Beta distribution, $B(\alpha, \beta)$

| | $MIAE_{\hat{F}}$ | | |
|---|---|---|---|
| n | $MIAE_{F_{kernel}}(MIAE_{\hat{F}_h})$ | $MIAE_{F_{Babu}}(MIAE_{\tilde{F}_{n,m}})$ | $MIAE_{F_{new}}(MIAE_{\hat{F}_{m,h}})$ |
| | | $\alpha = 0.8 \quad \beta = 0.8$ | |
| 50 | 0.0401 | 0.0357 | 0.0408 |
| 200 | 0.0210 | 0.0193 | 0.0209 |
| 700 | 0.0120 | 0.0109 | 0.0115 |
| | | $\alpha = 0.8 \quad \beta = 2$ | |
| 50 | 0.0418 | 0.0378 | 0.0419 |
| 200 | 0.0225 | 0.0198 | 0.0214 |
| 700 | 0.0126 | 0.0110 | 0.0115 |
| | | $\alpha = 1 \quad \beta = 2$ | |
| 50 | 0.0398 | 0.0340 | 0.0398 |
| 200 | 0.0215 | 0.0195 | 0.0211 |
| 700 | 0.0118 | 0.0107 | 0.0113 |
| | | $\alpha = 1 \quad \beta = 1$ | |
| 50 | 0.0395 | 0.0344 | 0.0405 |
| 200 | 0.0208 | 0.0194 | 0.0209 |
| 700 | 0.0119 | 0.0113 | 0.0117 |
| | | $\alpha = 2 \quad \beta = 2$ | |
| 50 | 0.0383 | 0.0352 | 0.0393 |
| 200 | 0.0204 | 0.0191 | 0.0202 |
| 700 | 0.0113 | 0.0106 | 0.0110 |
| | | $\alpha = 5 \quad \beta = 2$ | |
| 50 | 0.0398 | 0.0431 | 0.0404 |
| 200 | 0.0212 | 0.0210 | 0.0210 |
| 700 | 0.0115 | 0.0109 | 0.0111 |

Table 4.15: $MIAE_{\hat{F}}$'s of different density estimators for Beta distribution, $B(\alpha, \beta)$

| Efficiency | | | |
|---|---|---|---|
| n | $\frac{MIAE_{F_{kernel}}}{MIAE_{F_{kernel}}}$ | $\frac{MIAE_{F_{kernel}}}{MIAE_{F_{Babu}}}$ | $\frac{MIAE_{F_{kernel}}}{MIAE_{F_{new}}}$ |
| $\alpha = 0.8 \quad \beta = 0.8$ | | | |
| 50 | 1 | 1.12 | 0.983 |
| 200 | 1 | 1.09 | 1.000 |
| 700 | 1 | 1.10 | 1.050 |
| $\alpha = 0.8 \quad \beta = 2$ | | | |
| 50 | 1 | 1.11 | 0.997 |
| 200 | 1 | 1.14 | 1.050 |
| 700 | 1 | 1.15 | 1.090 |
| $\alpha = 1 \quad \beta = 2$ | | | |
| 50 | 1 | 1.17 | 1.00 |
| 200 | 1 | 1.10 | 1.02 |
| 700 | 1 | 1.10 | 1.04 |
| $\alpha = 1 \quad \beta = 1$ | | | |
| 50 | 1 | 1.15 | 0.975 |
| 200 | 1 | 1.07 | 0.994 |
| 700 | 1 | 1.06 | 1.020 |
| $\alpha = 2 \quad \beta = 2$ | | | |
| 50 | 1 | 1.09 | 0.974 |
| 200 | 1 | 1.07 | 1.010 |
| 700 | 1 | 1.07 | 1.020 |
| $\alpha = 5 \quad \beta = 2$ | | | |
| 50 | 1 | 0.924 | 0.987 |
| 200 | 1 | 1.010 | 1.010 |
| 700 | 1 | 1.060 | 1.040 |

Table 4.16: Efficiencies of different density estimators by $MIAE_{\hat{F}}$ criterion for Beta distribution, $B(\alpha, \beta)$

| | $MISE_{\hat{f}}$ | | |
|---|---|---|---|
| n | $MISE_{f_{kernel}}(MISE_{\hat{f}_h})$ | $MISE_{f_{Babu}}(MISE_{\tilde{f}_{n,m}})$ | $MISE_{f_{new}}(MISE_{\hat{f}_{m,h}})$ |
| | | $s = 0.7$ | |
| 50 | 0.320 | 0.1180 | 0.286 |
| 200 | 0.250 | 0.0756 | 0.196 |
| 700 | 0.194 | 0.0454 | 0.131 |
| | | $s = 1$ | |
| 50 | 0.0520 | 0.01760 | 0.0489 |
| 200 | 0.0321 | 0.01010 | 0.0244 |
| 700 | 0.0174 | 0.00476 | 0.0108 |
| | | $s = 3$ | |
| 50 | 0.00864 | 0.02070 | 0.01020 |
| 200 | 0.00305 | 0.00458 | 0.00418 |
| 700 | 0.00120 | 0.00127 | 0.00195 |

Table 4.17: $MISE_{\hat{f}}$'s of different density estimators for Gamma distribution, $\Gamma(s)$

| | Efficiency | | |
|---|---|---|---|
| n | $\frac{MISE_{f_{kernel}}}{MISE_{f_{kernel}}}$ | $\frac{MISE_{f_{kernel}}}{MISE_{f_{Babu}}}$ | $\frac{MISE_{f_{kernel}}}{MISE_{f_{new}}}$ |
| | | $s = 0.7$ | |
| 50 | 1 | 2.71 | 1.12 |
| 200 | 1 | 3.31 | 1.28 |
| 700 | 1 | 4.28 | 1.48 |
| | | $s = 1$ | |
| 50 | 1 | 2.95 | 1.06 |
| 200 | 1 | 3.18 | 1.31 |
| 700 | 1 | 3.65 | 1.61 |
| | | $s = 3$ | |
| 50 | 1 | 0.417 | 0.843 |
| 200 | 1 | 0.667 | 0.731 |
| 700 | 1 | 0.944 | 0.614 |

Table 4.18: Efficiencies of different density estimators by $MISE_{\hat{f}}$ criterion for Gamma distribution, $\Gamma(s)$

| | $MIAE_{\hat{f}}$ | | |
|---|---|---|---|
| n | $MIAE_{f_{kernel}}(MIAE_{\hat{f}_h})$ | $MIAE_{f_{Babu}}(MIAE_{\tilde{f}_{n,m}})$ | $MIAE_{f_{new}}(MIAE_{\hat{f}_{m,h}})$ |
| | | $s = 0.7$ | |
| 50 | 0.338 | 0.1610 | 0.325 |
| 200 | 0.244 | 0.1110 | 0.221 |
| 700 | 0.181 | 0.0769 | 0.157 |
| | | $s = 1$ | |
| 50 | 0.279 | 0.1630 | 0.272 |
| 200 | 0.190 | 0.1020 | 0.181 |
| 700 | 0.125 | 0.0734 | 0.119 |
| | | $s = 3$ | |
| 50 | 0.2040 | 0.360 | 0.216 |
| 200 | 0.1280 | 0.167 | 0.143 |
| 700 | 0.0769 | 0.086 | 0.094 |

Table 4.19: $MIAE_{\hat{f}}$'s of different density estimators for Gamma distribution, $\Gamma(s)$

| | Efficiency | | |
|---|---|---|---|
| n | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{kernel}}}$ | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{Babu}}}$ | $\frac{MIAE_{f_{kernel}}}{MIAE_{f_{new}}}$ |
| | | $s = 0.7$ | |
| 50 | 1 | 2.11 | 1.04 |
| 200 | 1 | 2.21 | 1.11 |
| 700 | 1 | 2.35 | 1.15 |
| | | $s = 1$ | |
| 50 | 1 | 1.71 | 1.03 |
| 200 | 1 | 1.87 | 1.05 |
| 700 | 1 | 1.70 | 1.05 |
| | | $s = 3$ | |
| 50 | 1 | 0.568 | 0.947 |
| 200 | 1 | 0.765 | 0.896 |
| 700 | 1 | 0.895 | 0.818 |

Table 4.20: Efficiencies of different density estimators by $MIAE_{\hat{f}}$ criterion for Gamma distribution, $\Gamma(s)$

| | $MISE_F$ | | |
|---|---|---|---|
| n | $F_{kernel}(\hat{F}_h)$ | $F_{Babu}(\tilde{F}_{n,m})$ | $F_{new}(\hat{F}_{Bk})$ |
| | $s = 0.7$ | | |
| 50 | 0.003490 | 0.002190 | 0.00755 |
| 200 | 0.001070 | 0.000670 | 0.00243 |
| 700 | 0.000353 | 0.000206 | 0.00143 |
| | $s = 1$ | | |
| 50 | 0.003030 | 0.002130 | 0.00494 |
| 200 | 0.000876 | 0.000659 | 0.00178 |
| 700 | 0.000259 | 0.000200 | 0.00062 |
| | $s = 3$ | | |
| 50 | 0.002640 | 0.005800 | 0.003350 |
| 200 | 0.000752 | 0.001340 | 0.000951 |
| 700 | 0.000218 | 0.000303 | 0.000612 |

Table 4.17: $MISE_F$'s of different density estimators for Gamma distribution, $\Gamma(s)$

| | Efficiency | | |
|---|---|---|---|
| n | $MISE_{F_{kernel}}/MISE_{F_{kernel}}$ | $MISE_{F_{kernel}}/MISE_{F_{Babu}}$ | $MISE_{F_{kernel}}/MISE_{F_{new}}$ |
| | $s = 0.7$ | | |
| 50 | 1 | 1.60 | 0.462 |
| 200 | 1 | 1.60 | 0.441 |
| 700 | 1 | 1.71 | 0.247 |
| | $s = 1$ | | |
| 50 | 1 | 1.42 | 0.613 |
| 200 | 1 | 1.33 | 0.491 |
| 700 | 1 | 1.30 | 0.418 |
| | $s = 3$ | | |
| 50 | 1 | 0.456 | 0.788 |
| 200 | 1 | 0.562 | 0.791 |
| 700 | 1 | 0.722 | 0.357 |

Table 4.18: Efficiencies of different density estimators by $MISE_F$ criterion for Gamma distribution, $\Gamma(s)$

| | $MIAE_F$ | | |
|---|---|---|---|
| n | $F_{kernel}(\hat{F}_h)$ | $F_{Babu}(\tilde{F}_{n,m})$ | $F_{new}(\hat{F}_{Bk})$ |
| | $s = 0.7$ | | |
| 50 | 0.0457 | 0.0366 | 0.0492 |
| 200 | 0.0253 | 0.0198 | 0.0254 |
| 700 | 0.0144 | 0.0112 | 0.0129 |
| | $s = 1$ | | |
| 50 | 0.0442 | 0.0363 | 0.0444 |
| 200 | 0.0229 | 0.0193 | 0.0237 |
| 700 | 0.0128 | 0.0110 | 0.0120 |
| | $s = 3$ | | |
| 50 | 0.0405 | 0.0657 | 0.0410 |
| 200 | 0.0211 | 0.0301 | 0.0210 |
| 700 | 0.0118 | 0.0142 | 0.0114 |

Table 4.17: $MIAE_F$'s of different density estimators for Gamma distribution, $\Gamma(s)$

| | Efficiency | | |
|---|---|---|---|
| n | $MIAE_{F_{kernel}}/MIAE_{F_{kernel}}$ | $MIAE_{F_{kernel}}/MIAE_{F_{Babu}}$ | $MIAE_{F_{kernel}}/MIAE_{F_{new}}$ |
| | $s = 0.7$ | | |
| 50 | 1 | 1.25 | 0.929 |
| 200 | 1 | 1.28 | 0.997 |
| 700 | 1 | 1.29 | 1.120 |
| | $s = 1$ | | |
| 50 | 1 | 1.22 | 0.994 |
| 200 | 1 | 1.19 | 0.967 |
| 700 | 1 | 1.16 | 1.070 |
| | $s = 3$ | | |
| 50 | 1 | 0.616 | 0.987 |
| 200 | 1 | 0.701 | 1.000 |
| 700 | 1 | 0.830 | 1.030 |

Table 4.18: Efficiencies of different density estimators by $MIAE_F$ criterion for Gamma distribution, $\Gamma(s)$

# Chapter 5

# Conclusions and Further Questions

## 5.1 Conclusions

From the previous chapters, we discussed the traditional kernel estimate, Babu estimate based on Bernstein polynomials and the new kernel estimate based on Bernstein polynomials. We learned that traditional kernel estimate is popular in nonparametric smoothing, but it often oversmooths the true density. Babu estimate works well in most cases, but it can not be used unless the support of the density is known. Furthermore, it is not able to catch the shape of the true density to the multimodal curve. The new kernel estimate is the tradeoff between the traditional kernel and the Babu estimate. It may overcome the problem of oversmoothness, and catch the shape of various true densities well, based on proper values of bandwidth and $m$. The simulated comparison among three approaches illustrates the

efficiencies for normal mixture, Beta distribution and Gamma distribution.

## 5.2 Further Questions

There are still many unsettled issues concerning the performance and practical implementation. Due to time limitation, we have to leave the following problems as further work.

(1) As we know, the derivation of new kernel estimation is given by

$$\hat{f}_X(x) = \tilde{f}_{n,m}(\hat{F}_h(x)) \cdot \hat{f}_h(x) \tag{5.1}$$

where $\hat{f}_h(x)$ is the kernel density estimation. New kernel density estimator can be considered as an improvement to initial traditional kernel density estimator $\hat{f}_h(x)$ using adjusted factor $\tilde{f}_{n,m}(\hat{F}_h(x))$. Therefore, if a better initial estimator is available, using Bernstein polynomials, we will construct a more accurate smoothing estimator.

We consider to replace the initial estimators $\hat{F}_h(x)$ and $\hat{f}_h(x)$ with $\hat{F}_{Bk}(x)$ and $\hat{f}_{Bk}(x)$ respectively in the above equation, then we would get the new estimator $\hat{\hat{f}}_{Bk}(x)$:

$$\hat{\hat{f}}_{m,h}(x) = \tilde{f}_{n,m}(\hat{F}_{m,h}(x)) \cdot \hat{f}_{m,h}(x) \tag{5.2}$$

Intuitively, $\hat{\hat{f}}_{m,h}(x)$ is better than $\hat{f}_{m,h}(x)$ although the computation is a bit complex.

(2) Theorem 2.2.1 is the application of the following lemma (Feller, 1965, Lemma 1, Section VII.1).

**Lemma 5.2.1.** *For $n = 1, 2, \ldots$ consider a family of distributions $F_{n,\theta}$ with expectation $\theta$ and variance $\sigma_n^2(\theta)$; here $\theta$ is a parameter varying in a finite or infinite interval. Suppose that $u$ is bounded and continuous, and that $\sigma_n^2(\theta) \to 0$ for each $\theta$. Then*

$$\boldsymbol{E}_{n,\theta}(u) = \int_{-\infty}^{+\infty} u(x)d(F_{n,\theta}(x)) \to u(\theta).$$

*The convergence is uniform in every finite interval in which $\sigma_n^2(\theta) \to 0$ uniformly.*

If $F_{n,\theta}, 0 \leq \theta \leq 1$ is a binomial distribution concentrated on the points $k/m (k = 0, \ldots, m)$, then $\sigma_n^2(\theta) = \theta(1-\theta)n^{-1} \to 0$ and so it derives Babu estimate and our new kernel estimate.

If $F_{n,\theta}, \theta \geq 0$ is a Poisson distribution attaching probability $e^{-m\theta}(m\theta)^k/k!$ to the point $k/m$, we have $\sigma_n^2(\theta) = \theta/n$ and so

$$u_m^*(\theta) = \sum_{k=0}^{\infty} u(\frac{k}{m}) p_k(m, \theta) \to u(\theta)$$

uniformly in every finite $\theta$-interval, where

$$p_k(m, \theta) = e^{-m\theta} \frac{(m\theta)^k}{k!}$$

Based on this formula, we can improve our traditional kernel density estimate to get

$$\tilde{F}_{n,m}(x) = \sum_{k=0}^{\infty} F_n(k/m) p_k(m, x), \qquad x \in [0, \infty).$$

The approach does not need to transfer the original observations into the support[0, 1], but it does for $[0, \infty)$. However, the difficulty is to compute the infinite summation.

# Chapter 6

# Appendix

### Appendix I

Appendix I shows the data set and $R$ programs of two examples in Chapter 3.

```
#-------------------------------------------------------------------
#          Data 1 (Silverman, 1986, p.8)
#-------------------------------------------------------------------
x < − c( 1, 1, 1, 5, 7, 8, 8, 13, 14, 14, 17, 18, 21, 21, 22, 25, 27, 27, 30, 30,
31, 31, 32, 34, 35, 36, 37, 38, 39, 39, 40, 49, 49, 54, 56, 56, 62, 63, 65, 65, 67,
75, 76, 79, 82, 83, 84, 84, 84, 90, 91, 92, 93, 93, 103, 103, 111, 112, 119, 122,
123, 126, 129, 134, 144, 147, 153, 163, 167, 175, 228, 231, 235, 242, 256, 256,
257, 311, 314, 322, 369, 415, 573, 609, 640, 737)      ##n = 86##
#-------------------------------------------------------------------
feg1 < − function(x)  {          ## x=obs. ##
n < − length(x)
fz < − density(x, width="SJ-dpi", n=200)
h< − fz$bw       ## find the direct-plug-in bandwidth ##
z< − fz$x
f0< − fz$y
Dbinom < − function(k, m, x) if (x> 0 & x< 1) dbinom(k, m, x) else 0
m< − floor(n/log(n))          ## take integer to m ##
fx< − Y1< − fy2< − fy1< − y1< − 1: 200
## create 200 elements in each variable ##
```

```
for(i in 1: n) y1[i] < − sum(pnorm((x[i]-x)/h))/n
for(i in 1: 200)  {
Y1[i] < − sum(pnorm((z[i]-x)/h))/n
fx[i] < − sum(dnorm((z[i]-x)/h))/h/n
}
y2< − x/(1+x)
Y2< − z/(1+z)
Fn1< − Fn2< − 1:(m+1)
for (k in 0: m)  {
Fn1[k+1]< − sum(y1<= k/m)/n
Fn2[k+1]< − sum(y2<= k/m)/n
}
for(i in 1: 200)  {
fy1[i]< − m*sum((Fn1[2:(m+1)]-Fn1[1:m])*Dbinom(0:(m-1), m-1, Y1[i]))*fx[i]
fy2[i]< − m*sum((Fn2[2:(m+1)]-Fn2[1:m])*Dbinom(0:(m-1), m-1, Y2[i]))/pi/(1+z[i]
^2)
}
plot(c(x, z, z, z), c(rep(0, n), fy1, fy2, f0),type="n", xlab="Length of treat-
ment (days)", ylab="Density")
lines(x, rep(0,n), type="p", pch="|")
lines(z, fy1, col=2, lty=7)      ## our new est'or! ##
lines(z, fy2, col=3, lty=8)      ## Babu est'or! ##
lines(z, f0, col=6, lty=9)      ## kernel est'or! ##
title(main="(a) Densities of lengths of treatment of control patients in sui-
cide study")
}
feg1(x)


#-------------------------------------------------------------------
#          Data 2 (Silverman, 1986, p.8)
#-------------------------------------------------------------------
x < − c( 4.37, 4.70, 1.68, 1.75, 4.35, 1.77, 4.25, 4.10, 4.05, 1.90, 4.00, 4.42,
1.83, 1.83, 3.95, 4.83, 3.87, 1.73, 3.92, 3.20, 2.33, 4.57, 3.58, 3.70, 4.25,
3.58, 3.67, 1.90, 4.13, 4.53, 4.10, 4.12, 4.00, 4.93, 3.68, 1.85, 3.83, 1.85,
3.80, 3.80, 3.33, 3.73, 1.67, 4.63, 1.83, 2.03, 2.72, 4.03, 1.73, 3.10, 4.62, 1.88,
3.52, 3.77, 3.43, 2.00, 3.73, 4.60, 2.93, 4.65, 4.18, 4.58, 3.50, 4.62, 4.03, 1.97,
```

```
4.60, 4.00, 3.75, 4.00, 4.33, 1.82, 1.67, 3.50, 4.20, 4.43, 1.90, 4.08, 3.43, 1.77,
4.50, 1.80, 3.70, 2.50, 2.27, 2.93, 4.63, 4.00, 1.97, 3.93, 4.07, 4.50, 2.25, 4.25,
4.08, 3.92, 4.73, 3.72, 4.50, 4.40, 4.58, 3.50, 1.80, 4.28, 4.33, 4.13, 1.95 )
##n = 107##
#-----------------------------------------------------------------------
feg2< − function(x)  {          ## x=obs. ##
n< − length(x)
fz< − density(x, width="SJ-dpi",n=200)
h< − fz$bw      ## find the direct-plug-in kernel bandwidth##
z< − fz$x
f0< − fz$y
Dbinom < − function(k, m, x) if (x> 0 & x< 1) dbinom(k, m, x) else 0
m< − floor(n/log(n))      ##take integer to m ##
fx< − Y1< − fy2< − fy1< − y1< − 1: 200
## create 200 elements in each variable ##
for(i in 1: n) y1[i] < − sum(pnorm((x[i]-x)/h))/n
for(i in 1: 200)   { Y1[i] < − sum(pnorm((z[i]-x)/h))/n
fx[i] < − sum(dnorm((z[i]-x)/h))/h/n
}
y2< − x/(1+x)
Y2< − z/(1+z)
Fn1 < − Fn2< − 1:(m+1)
for (k in 0: m)  {
Fn1[k+1]< − sum(y1<= k/m)/n
Fn2[k+1]< − sum(y2<= k/m)/n
}
for(i in 1: 200)  {
fy1[i]< − m*sum((Fn1[2:(m+1)]-Fn1[1:m])*Dbinom(0:(m-1),m-1,Y1[i]) )*fx[i]
fy2[i]< − m*sum((Fn2[2:(m+1)]-Fn2[1:m])*Dbinom(0:(m-1),m-1,Y2[i]) )/pi/(1+z[i]
^2)
}
plot(c(x, z, z, z), c(rep(0, n), fy1, fy2, f0), type="n", xlab="Eruption length
(min)", ylab="Density")
lines(x, rep(0,n), type="p", pch="|")
lines(z, fy1, col=2, lty=7)      ##our new est'or! ##
lines(z, fy2, col=3, lty=8)      ##Babu est'or! ##
lines(z, f0, col=6, lty=9)      ##kernel est'or! ##
title(main="(b) Densities of eruption lengths of Old Faithful geyser")
```

```
}
feg2(x)
```

### Appendix II

Appendix II is the *R* programs of MISE and MIAE simulations in Chapter 4.

```
#-------------------------------------------------------------
#          MISE_f̂ for normal mixture
#-------------------------------------------------------------
ffMISEnorm<- function(n, mu, sd)  {
## n=num. of obs., mu=mean, sd=standard deviation of normal dist. ##
unif<- runif(n+1)
x<- (unif<= 0.5)*rnorm(n+1)+(unif> 0.5)*rnorm(n+1, mu, sd)
X<- x[n+1]
x<- sort(x[1:n])
dX<- 0.5*dnorm(X)+0.5*dnorm(X, mu, sd)
## calculate MISE ##
Z1<- (fkernel(n, x, X)-dX)^2/dX
Z2<- (fbabu(n, x, X)-dX)^2/dX
Z3<- (fhat(n, x, X)-dX)^2/dX
return(c(Z1, Z2, Z3))
}

fkernel<- function(n, x, z)  {     ## x=obs, z=X ##
h<- density(x, width="SJ-dpi", n=200)$bw
fx<- sum(dnorm((z-x)/h))/h/n
return(fx)
}
fbp<- function(n, x, z)  {
m<- floor(n/log(n))
Fn<- 1:(m+1)
for (k in 0: m)  Fn[k+1]<- sum(x<= k/m)/n
```

```
fy< − m*sum((Fn[2:(m+1)]-Fn[1:m])*dbinom(0:(m-1), m-1, z))
return(fy)
}
fbabu< − function(n, x, z)  {
y2< − 0.5+(1/pi)*(atan(x))
Y2< − 0.5+(1/pi)*(atan(z))
fy2< − fbp(n, y2, Y2)/pi/(1+z^2)
return(fy2)
} fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
for(i in 1:n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
fy1< − fbp(n, y1, Y1)*fkernel(n, x, z)
return(fy1)
}
ffMISE < − function(N,mu,sd)  {      ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {      S < − rep(0, 3)
for (j in 1:N)  S < − S+ffMISEnorm(n[i], mu, sd)
D[i,  ] < − S/N
}
MISE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MISE, Ef)
}
ffMISE(5000, 0, 1)


#-------------------------------------------------------------
#          MIAE_f̂ for normal mixture
#-------------------------------------------------------------
ffMIAEnorm< − function(n, mu, sd)  {
## n=num. of obs., mu=mean, sd=standard deviation of normal dist. ##
unif< − runif(n+1)
```

```
x< − (unif<=0.5)*rnorm(n+1)+(unif>0.5)*rnorm(n+1, mu, sd)
X< − x[n+1] ; x < − sort(x[1:n])
dX< − 0.5*dnorm(X)+0.5*dnorm(X, mu, sd)
## calculate MIAE ##
Z1< − abs(fkernel(n, x, X)/dX-1)
Z2< − abs(fbabu(n, x, X)/dX-1)
Z3< − abs(fhat(n, x, X)/dX-1)
return(c(Z1, Z2, Z3))
}
fkernel< − function(n, x, z)  {      ## x=obs, z=X ##
h< − density(x, width="SJ-dpi", n=200)$bw
fx < − sum(dnorm((z-x)/h))/h/n
return(fx)
}
fbp< − function(n, x, z)  {
m< − floor(n/log(n))
Fn< − 1:(m+1)
for (k in 0: m)  Fn[k+1]< − sum(x<=k/m)/n
fy< − m*sum((Fn[2:(m+1)]-Fn[1:m])*dbinom(0:(m-1), m-1, z))
return(fy)
}
fbabu< − function(n,x,z)  {
y2< − 0.5+(1/pi)*(atan(x))
Y2< − 0.5+(1/pi)*(atan(z))
fy2< − fbp(n,y2,Y2)/pi/(1+z^2)
return(fy2)
}
fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
## find the direct-plug-in kernel bandwidth##
y1< − 1:n
for(i in 1: n)      y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
fy1< − fbp(n, y1, Y1)*fkernel(n, x, z)
return(fy1)
}
ffMIAE < − function(N,mu,sd)  {      ## N=num. of simulation ##
```

```
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {      S < − rep(0, 3)
for (j in 1:N)  S < − S+ffMIAEnorm(n[i], mu, sd)
D[i,  ] < − S/N
}
MIAE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MIAE, Ef)
}
ffMIAE(5000, 0, 1)
```

```
#-------------------------------------------------------------
#          MISE_F̂ for normal mixture
#-------------------------------------------------------------
FFMISEnorm< − function(n, mu, sd)  {
## n=num. of obs., mu=mean, sd=standard deviation of normal dist. ##
unif< − runif(n+1)
x< − (unif<= 0.5)*rnorm(n+1)+(unif> 0.5)*rnorm(n+1, mu, sd)
X< − x[n+1]
x< − sort(x[1:n])
FX< − 0.5*pnorm(X)+0.5*pnorm(X, mu, sd)
## calculate MISE ##
Z1< − (Fkernel(n, x, X)-FX)^2
Z2< − (Fbabu(n, x, X)-FX)^2
Z3< − (Fhat(n, x, X)-FX)^2
return(c(Z1, Z2, Z3))
}
Fkernel< − function(n, x, z)  {      ## x=obs, z=X ##
h< − density(x, width="SJ-dpi", n=200)$bw
Fx< − sum(pnorm((z-x)/h))/n
return(Fx)
}
Fbp< −function(n, x, z)  {
m< − floor(n/log(n))
Fn< − 1:(m+1)
```

```
for (k in 1:(m+1))  Fn[k]< −sum(x<= (k-1)/m)/n
Fy< − sum(Fn[1:(m+1)]*dbinom(0:m, m, z))
return(Fy)
}
Fbabu< − function(n, x, z)  {
y2< − 0.5+(1/pi)*(atan(x))
Y2< − 0.5+(1/pi)*(atan(z))
Fy2< − Fbp(n, y2, Y2)
return(Fy2)
}
Fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
for(i in 1: n)  y1[i]< − sum(pnorm((x[i]-x)/h))/n
Y1< − sum(pnorm((z-x)/h))/n
Fy1< − Fbp(n, y1, Y1)
return(Fy1)
}
FFMISE< − function(N, mu, sd)  {     ## N=num. of simulation ##
D< − matrix(0, 3, 3)
n< − c(50, 200, 700)
for (i in 1:3)  {     S < − rep(0, 3)
for (j in 1:N)  S < − S+FFMISEnorm(n[i], mu, sd)
D[i,  ]< − S/N
}
MISE< − cbind(n, signif(D, 3))
Ef< − cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MISE, Ef)
}
FFMISE(5000, 0, 1)
```

```
#-------------------------------------------------------------
#          MIAE_F̂ for normal mixture
#-------------------------------------------------------------
FFMIAEnorm< − function(n, mu, sd)  {
```

```
## n=num. of obs., mu=mean, sd=standard deviation of normal dist. ##
unif< − runif(n+1)
x< − (unif<=0.5)*rnorm(n+1)+(unif>0.5)*rnorm(n+1, mu, sd)
X < − x[n+1] ; x < − sort(x[1:n])
FX< − 0.5*pnorm(X)+0.5*pnorm(X, mu, sd)
## calculate MIAE ##
Z1< − abs(Fkernel(n, x, X)-FX)
Z2< − abs(Fbabu(n, x, X)-FX)
Z3< − abs(Fhat(n, x, X)-FX)
return(c(Z1, Z2, Z3))
}
Fkernel< − function(n, x, z)  {      ## x=obs, z=X ##
h< − density(x, width="SJ-dpi", n=200)$bw
Fx < − sum(pnorm((z-x)/h))/n
return(Fx)
}
Fbp< − function(n, x, z)  {
m< − floor(n/log(n))
Fn< − 1:(m+1)
for (k in 1: (m+1))  Fn[k]< −sum(x<=(k-1)/m)/n
Fy< − sum(Fn[1:(m+1)]*dbinom(0:m, m, z))
return(Fy)
}
Fbabu< − function(n, x, z)  {
y2< − 0.5+(1/pi)*(atan(x))
Y2< − 0.5+(1/pi)*(atan(z))
Fy2< − Fbp(n, y2, Y2)
return(Fy2)
}
Fhat< − function(n, x, z) {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
for(i in 1: n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
Fy1< − Fbp(n, y1, Y1)
return(Fy1)
}
```

```
FFMIAE <- function(N, mu, sd)  {      ## N=num. of simulation ##
D <- matrix(0, 3, 3)
n <- c(50, 200, 700)
for (i in 1:3)  {      S <- rep(0, 3)
for (j in 1:N)  S <- S+FFMIAEnorm(n[i], mu, sd)
D[i,  ] <- S/N
}
MIAE <- cbind(n, signif(D, 3))
Ef <- cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MIAE, Ef)
}
FFMIAE(5000, 0, 1)
```

```
#-------------------------------------------------------------
#          MISE_f for Beta Distribution
#-------------------------------------------------------------
ffMISEbeta<- function(n, a, b)  {
## n=num. of obs., a,b =para. of beta dist. ##
x<- rbeta(n+1, a, b)
X<- x[n+1]
x<- sort(x[1:n])
dX<- dbeta(X, a, b)
## calculate MISE ##
Z1<- (fkernel(n, x, X)-dX)^2/dX
Z2<- (fbabu(n, x, X)-dX)^2/dX
Z3<- (fhat(n, x, X)-dX)^2/dX
return(c(Z1, Z2, Z3))
}
fkernel<- function(n, x, z)  {      ## x=obs, z=X ##
h<- density(x, width="SJ-dpi", n=200)$bw
fx <- sum(dnorm((z-x)/h))/h/n
return(fx)
}
fbp<- function(n, x, z)  {
m<- floor(n/log(n))
Dbinom <- function(k, m, x)  if (x>0 & x<1)  dbinom(k, m, x)  else 0
```

```
Fn< − 1:(m+1)
for (k in 0: m)  Fn[k+1]< − sum(x<=k/m)/n
fy< − m*sum((Fn[2:(m+1)]-Fn[1:m])*Dbinom(0:(m-1), m-1, z))
return(fy)
}
fbabu< − function(n, x, z)  {
y2< − x
Y2< − z
fy2< − fbp(n, y2, Y2)
return(fy2)
}
fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
for(i in 1: n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
fy1< − fbp(n, y1, Y1)*fkernel(n, x, z)
return(fy1)
}
ffMISE < − function(N, a, b)  {      ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {     S < − rep(0, 3)
for (j in 1:N)   S < − S+ffMISEbeta(n[i], a, b)
D[i,  ] < − S/N
}
MISE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MISE, Ef)
}
ffMISE(5000, 1, 1)
```

```
#-------------------------------------------------------------
#          MIAE_f̂ for Beta Distribution
#-------------------------------------------------------------
```

```
ffMIAEbeta< − function(n,a,b)  {
## n=num. of obs., a,b=para. of beta dist. ##
x< − rbeta(n+1, a, b)
X< − x[n+1]
x< − sort(x[1:n])
dX< − dbeta(X, a, b)
## calculate MIAE ##
Z1< − abs(fkernel(n, x, X)/dX-1)
Z2< − abs(fbabu(n, x, X)/dX-1)
Z3< − abs(fhat(n, x, X)/dX-1)
return(c(Z1, Z2, Z3))
}
fkernel< − function(n, x, z)  {      ## x=obs, z=X ##
h< − density(x, width="SJ-dpi", n=200)$bw
fx < − sum(dnorm((z-x)/h))/h/n
return(fx)
}
fbp< − function(n, x, z)  {
m< − floor(n/log(n))
Dbinom < − function(k, m, x)  if (x>0 & x<1)  dbinom(k, m, x)  else 0
Fn< − 1:(m+1)
for (k in 0: m)  Fn[k+1]< − sum(x<=k/m)/n
fy< − m*sum((Fn[2:(m+1)]-Fn[1:m])*Dbinom(0:(m-1), m-1, z))
return(fy)
}
fbabu< − function(n, x, z)  {
y2< − x
Y2< − z
fy2< − fbp(n, y2, Y2)
return(fy2)
}
fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
for(i in 1: n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
fy1< − fbp(n, y1, Y1)*fkernel(n, x, z)
```

```
return(fy1)
}
ffMIAE < − function(N, a, b)  {      ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {       S < − rep(0,3)
for (j in 1:N)    S < − S+ffMIAEbeta(n[i], a, b)
D[i,  ] < − S/N
}
MIAE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MIAE, Ef)
}
ffMIAE(5000, 1, 1)
```

```
#————————————————————————————————————————
#          MISE_F̂ for Beta Distribution
#————————————————————————————————————————
FFMISEbeta< − function(n, a, b)
## n=num. of obs., a,b =para. of beta dist. ##
x< − rbeta(n+1, a, b)
X< − x[n+1]
x< − sort(x[1:n])
FX< − pbeta(X, a, b)
## calculate MISE ##
Z1< − (Fkernel(n, x, X)-FX)^2
Z2< − (Fbabu(n, x, X)-FX)^2
Z3< − (Fhat(n, x, X)-FX)^2
return(c(Z1, Z2, Z3))
}
Fkernel< − function(n, x, z)  {      ## x=obs, z=X ##
h< − density(x, width="SJ-dpi", n=200)$bw
Fx < − sum(pnorm((z-x)/h))/n
return(Fx)
}
Fbp< − function(n, x, z)  {
```

```
m< − floor(n/log(n))
Dbinom < − function(k, m, x)  if (x> 0 & x< 1)  dbinom(k, m, x)  else 0
Fn< − 1:(m+1)
for (k in 0: m)  Fn[k+1]< −sum(x<=k/m)/n
Fy< − sum(Fn[1:(m+1)]*Dbinom(0:m, m, z))
return(Fy)
}
Fbabu< − function(n, x, z)  {
y2< − x
Y2< − z
Fy2< − Fbp(n, y2, Y2)
return(Fy2)
}
Fhat< − function(n,x,z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
for(i in 1: n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
Fy1< − Fbp(n, y1, Y1)
return(Fy1)
}
FFMISE < − function(N, a, b)  {   ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {     S < − rep(0, 3)
for (j in 1:N)   S < − S+FFMISEbeta(n[i], a, b)
D[i,  ] < − S/N
}
MISE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MISE, Ef)
}
FFMISE(5000, 1, 1)
```

#

```
#          MIAE_F̂ for Beta Distribution
#-------------------------------------------------------------
FFMIAEbeta<- function(n, a, b) {
## n=num. of obs., a,b =para. of beta dist. ##
x<- rbeta(n+1, a, b)
X<- x[n+1]
x<- sort(x[1:n])
FX<- pbeta(X, a, b)
## calculate MIAE ##
Z1<- abs(Fkernel(n, x, X)-FX)
Z2<- abs(Fbabu(n, x, X)-FX)
Z3<- abs(Fhat(n, x, X)-FX)
return(c(Z1, Z2, Z3))
}
Fkernel<- function(n, x, z) {     ## x=obs, z=X ##
h<- density(x, width="SJ-dpi", n=200)$bw
Fx <- sum(pnorm((z-x)/h))/n
return(Fx)
}
Fbp<- function(n, x, z) {
m<- floor(n/log(n))
Dbinom <- function(k, m, x) if (x> 0 & x< 1) dbinom(k, m, x) else 0
Fn<- 1:(m+1)
for (k in 0: m) Fn[k+1]< -sum(x<=k/m)/n
Fy<- sum(Fn[1:(m+1)]*Dbinom(0:m, m, z))
return(Fy)
}
Fbabu<- function(n, x, z) {
y2<- x
Y2<- z
Fy2<- Fbp(n, y2, Y2)
return(Fy2)
}
Fhat<- function(n,x,z) {
x<- sort(x)
h<- density(x, width="SJ-dpi", n=200)$bw
y1<- 1:n
for(i in 1: n) y1[i] <- sum(pnorm((x[i]-x)/h))/n
```

```
Y1 < − sum(pnorm((z-x)/h))/n
Fy1< − Fbp(n, y1, Y1)
return(Fy1)
}
FFMIAE < − function(N, a, b)  {   ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {     S < − rep(0, 3)
for (j in 1:N)   S < − S+FFMIAEbeta(n[i], a, b)
D[i,  ] < − S/N
}
MIAE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  ,c(1, 1, 1)]/D, 3))
return(MIAE, Ef)
}
FFMIAE(5000, 1, 1)
```

```
#-------------------------------------------------------------
#          MISE_f for Gamma Distribution
#-------------------------------------------------------------
ffMISEgamma< − function(n, s)  {
## n=num. of obs., s=shape para., r=rate(=1/scale) of gamma dist. ##
x< − rgamma(n+1, s)
X< − x[n+1]
x< − sort(x[1:n])
dX< − dgamma(X, s)
## calculate MISE ##
Z1< − (fkernel(n, x, X)-dX)^2/dX
Z2< − (fbabu(n, x, X)-dX)^2/dX
Z3< −(fhat(n, x, X)-dX)^2/dX
return(c(Z1, Z2, Z3))
}
fkernel< − function(n, x, z)  {     ## x=obs, z=X ##
h< − density(x, width="SJ-dpi", n=200)$bw
fx < − sum(dnorm((z-x)/h))/h/n
return(fx)
```

```
}
fbp< − function(n, x, z)  {
m< − floor(n/log(n))
Dbinom < − function(k, m, x)  if (x> 0 & x< 1)  dbinom(k, m, x)  else
0
Fn< − 1:(m+1)
for (k in 0: m)  Fn[k+1]< −sum(x<=k/m)/n
fy< − m*sum((Fn[2:(m+1)]-Fn[1:m])*Dbinom(0:(m-1), m-1, z))
return(fy)
}
fbabu< − function(n, x, z)  {
y2< − x/(1+x)
Y2< − z/(1+z)
fy2< −fbp(n, y2, Y2)/(1+z)^2
return(fy2)
}
fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< −1:n
for(i in 1: n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
fy1< − fbp(n, y1, Y1)*fkernel(n, x, z)
return(fy1)
}
ffMISE < − function(N, s)  {      ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {      S < − rep(0, 3)
for (j in 1:N)  S < − S+ffMISEgamma(n[i], s)
D[i,  ] < − S/N
}
MISE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  , c(1, 1, 1)]/D, 3))
return(MISE, Ef)
}
ffMISE(5000, 3)
```

```
#-------------------------------------------------------------
#          MIAE_f̂ for Gamma Distribution
#-------------------------------------------------------------
ffMIAEgamma<- function(n, s)  {
## n=num. of obs., s=shape para., r=rate(=1/scale) of gamma dist. ##
x<- rgamma(n+1, s)
X<- x[n+1]
x<- sort(x[1:n])
dX<- dgamma(X, s)
## calculate MIAE ##
Z1<- abs(fkernel(n, x, X)/dX-1)
Z2<- abs(fbabu(n, x, X)/dX-1)
Z3<- abs(fhat(n, x, X)/dX-1)
return(c(Z1, Z2, Z3))
}
fkernel<- function(n, x, z)  {    ## x=obs, z=X ##
h<- density(x, width="SJ-dpi", n=200)$bw
fx <- sum(dnorm((z-x)/h))/h/n
return(fx)
}
fbp<- function(n, x, z)  {
m<- floor(n/log(n))
Dbinom <- function(k, m, x)  if (x> 0 & x< 1)  dbinom(k, m, x)  else
0
Fn<- 1:(m+1)
for (k in 0: m)  Fn[k+1]<-sum(x<=k/m)/n
fy<- m*sum((Fn[2:(m+1)]-Fn[1:m])*Dbinom(0:(m-1), m-1, z))
return(fy)
}
fbabu<- function(n, x, z)  {
y2<- x/(1+x)
Y2<- z/(1+z)
fy2<-fbp(n, y2, Y2)/(1+z)^2
return(fy2)
}
fhat<- function(n, x, z)  {
x<- sort(x)
h<- density(x, width="SJ-dpi", n=200)$bw
```

```
y1< −1:n
for(i in 1: n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
fy1< − fbp(n, y1, Y1)*fkernel(n, x, z)
return(fy1)
}
ffMIAE < − function(N, s)  {      ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)   {      S < − rep(0, 3)
for (j in 1:N)   S < − S+ffMIAEgamma(n[i], s)
D[i,  ] < − S/N
}
MIAE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[   ,c(1, 1, 1)]/D, 3))
return(MIAE, Ef)
}
ffMIAE(5000, 3)
```

```
#————————————————————————————————————————
#          MISE_F̂ for Gamma Distribution
#————————————————————————————————————————
FFMISEgamma< −function(n, s)  {
## n=num. of obs., s=shape para., r=rate(=1/scale) of gamma dist. ##
x< − rgamma(n+1, s)
X< − x[n+1]
x< − sort(x[1:n])
FX< − pgamma(X, s)
## calculate MISE ##
Z1< − (Fkernel(n, x, X)-FX)^2
Z2< − (Fbabu(n, x, X)-FX)^2
Z3< − (Fhat(n, x, X)-FX)^2
return(c(Z1, Z2, Z3))
}
Fkernel< − function(n, x, z)  {      ## x=obs, z=X ##
h< −density(x, width="SJ-dpi", n=200)$bw
```

```
Fx< − sum(pnorm((z-x)/h))/n
return(Fx)
}
Fbp< −function(n, x, z)  {
m< −floor(n/log(n))
Dbinom < − function(k, m, x)  if (x> 0 & x< 1)  dbinom(k, m, x)  else 0
Fn< − 1:(m+1)
for (k in 0: m)  Fn[k+1]< − sum(x<= k/m)/n
Fy< − sum(Fn[1:(m+1)]*Dbinom(0:m, m, z))
return(Fy)
}
Fbabu< − function(n, x, z)  {
y2< − x/(1+x)
Y2< − z/(1+z)
Fy2< − Fbp(n, y2, Y2)
return(Fy2)
}
Fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
for(i in 1: n)  y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
Fy1< − Fbp(n, y1, Y1)
return(Fy1)
}
FFMISE < − function(N, s)  {  ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)  {  S < − rep(0, 3)
for (j in 1:N)  S < − S+FFMISEgamma(n[i], s)
D[i,  ] < − S/N
}
MISE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[  ,c(1, 1, 1)]/D, 3))
return(MISE, Ef)
}
FFMISE(5000, 3)
```

```
#-------------------------------------------------------------------
#          MIAE_F̂ for Gamma Distribution
#-------------------------------------------------------------------
FFMIAEgamma< −function(n, s)  {
## n=num. of obs., s=shape para., r=rate(=1/scale) of gamma dist. ##
x< − rgamma(n+1, s)
X< − x[n+1]
x< − sort(x[1:n])
FX< − pgamma(X, s)
## calculate MIAE ##
Z1< − abs(Fkernel(n, x, X)-FX)
Z2< − abs(Fbabu(n, x, X)-FX)
Z3< − abs(Fhat(n, x, X)-FX)
return(c(Z1, Z2, Z3))
}
Fkernel< − function(n, x, z)  {      ## x=obs, z=X ##
h< −density(x, width="SJ-dpi", n=200)$bw
Fx< − sum(pnorm((z-x)/h))/n
return(Fx)
}
Fbp< −function(n, x, z)  {
m< −floor(n/log(n))
Dbinom < − function(k, m, x)  if (x> 0 & x< 1)  dbinom(k, m, x)  else 0
Fn< − 1:(m+1)
for (k in 0: m)  Fn[k+1]< − sum(x<= k/m)/n
Fy< − sum(Fn[1:(m+1)]*Dbinom(0:m, m, z))
return(Fy)
}
Fbabu< − function(n, x, z)  {
y2< − x/(1+x)
Y2< − z/(1+z)
Fy2< − Fbp(n, y2, Y2)
return(Fy2)
}
Fhat< − function(n, x, z)  {
x< − sort(x)
h< − density(x, width="SJ-dpi", n=200)$bw
y1< − 1:n
```

```
for(i in 1: n)    y1[i] < − sum(pnorm((x[i]-x)/h))/n
Y1 < − sum(pnorm((z-x)/h))/n
Fy1< − Fbp(n, y1, Y1)
return(Fy1)
}
FFMIAE < − function(N, s)    {      ## N=num. of simulation ##
D < − matrix(0, 3, 3)
n < − c(50, 200, 700)
for (i in 1:3)   {       S < − rep(0, 3)
for (j in 1:N)    S < − S+FFMIAEgamma(n[i], s)
D[i,   ] < − S/N
}
MIAE < − cbind(n, signif(D, 3))
Ef < − cbind(n, signif(D[   ,c(1, 1, 1)]/D, 3))
return(MIAE, Ef)
}
FFMIAE(5000, 3)
```

# Bibliography


[1] Bernstein, S.(1912) Démonstration du théorème de Weierstrass fondée sur le calcul des probabilities. *Comm. Soc. Math. Kharkov* **13**, 1-2.

[2] Bowman, A.W. (1984) An alternative method of cross-validation for the smoothing of density estimates. *Biometrika* **71**, 353-60.

[3] Devroye, L. and Györfi, L.(1985) *Nonparametric Density Estimation: The $L_1$ View.* Wiley, New York.

[4] Epanechnikov, V.A.(1969) Non-parametric estimation of a multivariate probability density. *Theory Probab. Appl.* **14**, 153-8.

[5] Feller, W. (1965) *An Introduction to Probability Theory and its Applications*, Vol. II. Wiley, New York.

[6] Fix,E. and Hodges, J.L.(1951) Discriminatory analysis–nonparametric discrimination: consistency properties. *Report No.4, project no. 21-29-004*, USAF School of Aviation Medicine, Randolph Field, Texas.

[7] Fix,E. and Hodges, J.L.(1989) Discriminatory analysis–nonparametric discrimination: consistency properties. *Internat. Statist. Rev.* **57**, 238-47.

[8] G.Jogesh Babu, Angelo J.Canty and Yogendra P. Chaubey (2002) Application of Bernstein Polynomials for smooth estimation of a distribution and density function. *Journal of Statistical Planning and Inference*, **105**, 377-392.

[9] Park, B.U. and Marron, J.S. (1992) On the use of pilot estimators in bandwidth selection. *J. Nonparametric Statist.* **1**, 231-40.

[10] Rudemo, M. (1982) Empirical choice of histograms and kernel density estimators. *Scand. J. Statist.* **9**, 65-78.

[11] Scott, D.W. and Terrell, G.R.(1987) Biased and unbiased cross-validation in density estimation. *Journal of the American Statistical Association*, **82**, 1131-1146.

[12] Scott, D.W. (1992) *Multivariate Density Estimation: Theory, Practice, and Visualization.* Wiley, New York.

[13] Silverman, B.W.(1986) *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, London.

[14] Sheather, S.J. and Jones, M.C. (1991) A reliable data-based bandwidth selection method for kernel density estimation. *J. Roy. Statist. Soc.* Ser. B **53**, 683-90.

[15] Wand,M.P. and Jones,M.C.(1995) *Kernel Smoothing.* Chapman and Hall, London.

[16] Zhang,J.(2005) Normal reference bandwidth based on quantile for kernel density estimation. (unpublished manuscript)