Kernel density estimation on positive data via the logKDE package for R

Andrew T. Jones[1], Hien D. Nguyen[2*], and Geoffrey J. McLachlan[1]

July 31, 2018

[1]School of Mathematics and Physics, University of Queensland, St. Lucia 4072, Queensland Australia. [2]Department of Mathematics and Statistics, La Trobe University, Bundoora 3086, Victoria Australia. *Corresponding author email: h.nguyen5@latrobe.edu.au.

Abstract

Kernel density estimators (KDEs) are ubiquitous tools for nonparametric estimation of probability density functions (PDFs), when data are obtained from unknown data generating processes. The KDEs that are typically available in software packages are defined, and designed, to estimate real-valued data. When applied to positive data, these typical KDEs do not yield *bona fide* PDFs. A log-transformation methodology can be applied to produce a nonparametric estimator that is appropriate and yields proper PDFs over positive supports. We call the KDEs obtained via this transformation log-KDEs. We derive expressions for the pointwise biases, variances, and mean-squared errors of the log-KDEs that are obtained via various kernel functions. Mean integrated squared error (MISE) and asymptotic MISE results are also provided and a plug-in rule for log-KDE bandwidths is derived. We demonstrate the log-KDEs methodology via our *R* package, `logKDE`. Real data case studies are provided to demonstrate the log-KDE approach.

Keywords: kernel density estimator; log-transformation; nonparametric; plug-in rule; positive data.

1 Introduction

Let X be a random variable that arises from a distribution that can be characterized by an unknown density function $f_X(x)$. Assume that (A1) X is supported on $\mathbb{R}$, and (A2) $f_X(x)$ is sufficiently continuously differentiable (i.e. $\int_{\mathbb{X}} \left|f^{(m)}(x)\right| \mathrm{d}x < \infty$, where $f^{(m)}(x)$ is the mth derivative of $f(x)$, for $m \le M \in \mathbb{N}$).

Let $\{X_i\}_{i=1}^n$ be an independent and identically distributed (IID) sample of random variables, where each X_i is identically distributed to X ($i \in [n] = \{1, \dots, n\}$). Under conditions (A1) and (A2), a common approach to estimating $f_X(x)$ is via the kernel density estimator (KDE)

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

which is constructed from the sample $\{X_i\}_{i=1}^n$. Here, $K(x)$ is a probability density function on $\mathbb{R}$ and is called the kernel function and $h > 0$ is referred to as the bandwidth. This approach was first proposed in the seminal work of [15].

Make assumptions (B1) $\int_{\mathbb{R}} K(x)\,\mathrm{d}x = 1$, (B2) $\int_{\mathbb{R}} xK(x)\,\mathrm{d}x = 0$, (B3) $\int_{\mathbb{R}} x^2 K(x)\,\mathrm{d}x = 1$, and (B4) $\int_{\mathbb{R}} K^2(x)\,\mathrm{d}x < \infty$, regarding the kernel function $K(x)$. Under conditions (A1) and (A2), [13] showed that (B1)–(B4) allowed for useful derivations of expressions for the mean squared error (MSE) and mean integrated squared error (MISE) between $f_X(x)$ and (1); see for example [19, Ch. 3] and [21, Ch. 24]. Furthermore, simple conditions can be derived for ensuring pointwise asymptotic unbiasedness and consistency of (1) (cf. [7, Sec. 32.7]). See [22] for further exposition regarding kernel density estimation (KDE).

The estimation of $f_X(x)$ by (1) has become a ubiquitous part of modern data analysis and visualization. The popularity of the methodology has made its implementation a staple in most available statistical software packages. For example, in the `R` statistical programming environment [14], KDE can be conducted using the core-package function `density`.

Unfortunately, when (A1) is not satisfied and is instead replaced by (A1*) X is supported on $(0, \infty)$, using a KDE of form (1) that is constructed from a kernel that satisfies (B1)–(B4) no longer provides a reasonable estimator of $f_X(x)$. That is, if $K(x) > 0$ for all $x \in \mathbb{R}$ and (B1)–(B4) are satisfied, then $\int_0^\infty \hat{f}_X(x)\,\mathrm{d}x < 1$ and thus (1) is no longer a proper probability density function (PDF) over $(0,\infty)$. For example, this occurs when $K(x)$ is taken to be the popular Gaussian kernel function. Furthermore, expressions for MSE and MISE between $f_Y(y)$ and (1) are no longer correct under (A1*) and (A2).

In [4], the authors proposed a simple and elegant solution to the problem of estimating $f_X(x)$ under (A1*) and (A2). Firstly, let $Y = \log X$, $Y_i = \log X_i$ ($i \in [n]$), and $f_Y(y)$ be the PDF of Y. Note that if X is supported on $(0,\infty)$ then the support of $f_Y(y)$ satisfies (A1). If we wish to estimate $f_Y(y)$, we can utilize a KDE of form (1), constructed from $\{Y_i\}_{i=1}^n$, with a kernel that satisfies (B1)–(B4). If $f_Y(y)$ also satisfies (A2), then we can calculate the MSE and MISE between $f_Y(y)$ and (1).

Let W be a random variable and $U = G(W)$, where $G(w)$ is a strictly increasing function. If the distribution of U and W can be characterized by the PDFs $f_U(u)$ and $f_W(w)$, respectively, then the change-of-variable formula yields: $f_W(w) = f_U(G(w))\,G^{(1)}(w)$ (cf. [1, Thm. 3.6.1]). By noting that $\mathrm{d}\log(x)/\mathrm{d}x = x^{-1}$, [4] used the aforementioned formula to derive the log kernel density estimator (log-KDE)

$$\begin{aligned}\hat{f}_{\log}(x) &= x^{-1}\hat{f}_Y(\log x)\\ &= \frac{1}{nh}\sum_{i=1}^n x^{-1}K\left(\frac{\log x - \log X_i}{h}\right)\\ &= \frac{1}{n}\sum_{i=1}^n L(x;X_i,h),\end{aligned}\tag{2}$$

where $L(x;z,h) = (xh)^{-1}K\left(\log\left[(x/z)^{1/h}\right]\right)$ is the log-kernel function with bandwidth h, at location parameter z. For any $z \in (0,\infty)$ and $h \in (0,\infty)$, $L(x;z,h)$ has the properties that (C1) $L(x;z,h) \geq 0$ for all $x \in (0,\infty)$ and (C2) $\int_0^\infty L(x;z,h)\,\mathrm{d}x = 1$, when (B1)–(B4) are satisfied.

By property (C2) we observe that $\int_0^\infty \hat{f}_X(x)\,\mathrm{d}x = 1$, thus making (2) a proper PDF on $(0,\infty)$. Furthermore, using the expressions for the MSE and MISE between $f_Y(y)$ and (1), we can derive the relevant quantities for (2) as well as demonstrate its asymptotic unbiasedness and consistency.

For every kernel function that satisfies (B1)–(B4), there is a log-kernel function that satisfies

Table 1: Pairs of kernel functions $K(y)$ and log-kernel functions $L(x;z,h)$, where $z \in (0,\infty)$ and $h \in (0,\infty)$.

Kernel	$K(y)$
Epanechnikov	$3\left(5-y^2\right)/\left(20\sqrt{5}\right)\mathbb{I}\left\{y \in \left(-\sqrt{5},\sqrt{5}\right)\right\}$
Gaussian	$(2\pi)^{-1/2}\exp\left(-y^2/2\right)$
Laplace	$\left(\sqrt{2}/2\right)\exp\left(-2^{1/2}\lvert y\rvert\right)$
Logistic	$\left(\pi/4\sqrt{3}\right)\operatorname{sech}^2\left(\pi y/2\sqrt{3}\right)$
Triangular	$\left(\sqrt{6}-\lvert y\rvert\right)6^{-1}\mathbb{I}\left\{y \in \left(-\sqrt{6},\sqrt{6}\right)\right\}$
Uniform	$\left(2\sqrt{3}\right)^{-1}\mathbb{I}\left\{y \in \left(-\sqrt{3},\sqrt{3}\right)\right\}$
Log-Kernel	$L(x;z,h)$
Log-Epanechnikov	$3\left(5-x^2\right)/\left(20\sqrt{5}xh\right)\mathbb{I}\left\{\log\left[(x/z)^{1/h}\right] \in \left(-\sqrt{5},\sqrt{5}\right)\right\}$
Log-Gaussian	$(2\pi)^{-1/2}(xh)^{-1}\exp\left(-\log^2\left[(x/z)^{1/h}\right]/2\right)$
Log-Laplace	$\left(\sqrt{2}/2\right)(xh)^{-1}\exp\left(-2^{1/2}\left\lvert\log\left[(x/z)^{1/h}\right]\right\rvert\right)$
Log-Logistic	$\left(\pi/4\sqrt{3}\right)(xh)^{-1}\operatorname{sech}^2\left(\pi\log\left[(x/z)^{1/h}\right]/2\sqrt{3}\right)$
Log-Triangular	$(xh)^{-1}\left(\sqrt{6}/6-\lvert x\rvert\right)6^{-1}\mathbb{I}\left\{\log\left[(x/z)^{1/h}\right] \in \left(-\sqrt{6},\sqrt{6}\right)\right\}$
Log-Uniform	$\left(2\sqrt{3}xh\right)^{-1}\mathbb{I}\left\{\log\left[(x/z)^{1/h}\right] \in \left(-\sqrt{3},\sqrt{3}\right)\right\}$

(C1) and (C2) and that generates a log-KDE that is a proper PDF over $(0,\infty)$. We have compiled an array of other potential pairs of kernel and log-kernel functions in Table 1. Throughout Table 1, the function $\mathbb{I}\{A\}$ takes value 1 if statement A is true and 0 otherwise.

Unfortunately, out of all of the listed function pairs from Table 1, only the Gaussian and log-Gaussian PDFs have been considered for use as kernel and log-kernel functions, respectively, for conducting log-KDE. The log-Gaussian PDF was used explicitly for the construction of log-KDEs in [4], and more generally, for conducting asymmetric KDE on the support $(0,\infty)$, in [11]. Other works that have considered the log-Gaussian PDF for conducting asymmetric KDE include [8], [10], [9], and [25]. In the *R* environment, asymmetric KDE with log-Gaussian PDF as kernels has been implemented through the `dke.fun` function from the package `Ake` [25]. For historical purposes, we also note the incidental use of the log-Gaussian PDF as a kernel function via transformation of variables in [6], [19, Sec. 2.10], [23], and [22, Sec. 2.10].

In this vignette, we make three main contributions. Firstly, we expand upon the theoretical results of [4], who derived expressions for the biases and variances between generic log-KDEs and their estimands. Here, we utilize general results for KDE of transformed data from [23] and [12].

We further derive a plug-in rule for the bandwidth h that is similar to the famous rule of [19, Sec. 3.4]. Secondly, we introduce the readers to our `R` package `logKDE`, which implements log-KDE in a manner that is familiar to users of the R base function `density`. The core functionalities of `logKDE` are described and example applications of the package are provided in order to familiarize the package to the reader. Thirdly, we perform an extensive Monte Carlo simulation study in order to demonstrate the relative advantages and disadvantages of the log-KDE approach via the different log-kernels from Table 1. We also compare the performance of log-KDE to other nonparametric density estimation techniques over $(0, \infty)$, such as asymmetric KDE estimation using the gamma kernel of [5] or the reciprocal inverse Gaussian kernel of [16], as well as the standard approach using KDEs of form (1).

The vignette proceeds as follows. Theoretical results for log-KDE are presented in Section 2. The `logKDE` package is introduced and described in Section 3. Numerical studies are conducted in Section 4. Conclusions are drawn in Section 5.

2 Theoretical Results

We start by noting that MSE and MISE expressions for the log-KDE with log-Gaussian kernels have been derived by [11]. The authors have also established the conditions for pointwise asymptotic unbiasedness and consistency for the log-Gaussian kernel. In the general case, informal results regarding expressions for the pointwise bias and variance, have been provided by [4]. In this section, we generalize the results of [11] and formalize the results of [4] via some previously known results from [23] and [22, Sec. 2.5]. In the sequel, we shall make assumptions (A1) and (A2) regarding $f_Y(y)$, (A1*) and (A2) regarding $f_X(x)$, and (B1)—(B4) regarding $K(y)$.

2.1 Pointwise Results

The following expressions are provided in [22, Sec. 2.5]. At any $y \in \mathbb{R}$, define the pointwise bias and variance between (1) and $f_Y(y)$ as

$$\begin{aligned}\text{Bias}\left[\hat{f}(y)\right] &= \mathbb{E}\left[\hat{f}(y)\right] - f_Y(y) \\ &= \frac{1}{2}h^2 f_Y^{(2)}(y) + o\left(h^2\right), \end{aligned} \tag{3}$$

and

$$\text{Var}\left[\hat{f}(y)\right] = \frac{1}{nh} f_Y(y) \int_{\mathbb{R}} K^2(z)\,\mathrm{d}z + o\left(\frac{1}{nh}\right), \tag{4}$$

respectively, where $a_n = o(b_n)$ as $n \to \infty$, if and only if $\lim_{n\to\infty} |a_n/b_n| = 0$. From expressions (3) and (4), and the change-of-variable formula, we obtain the following expressions for the bias, variance, and MSE between (2) and $f_X(x)$.

Proposition 1. *For any $x \in (0,\infty)$, the bias, variance, and MSE between (2) and $f_X(x)$ have the forms*

$$\begin{aligned}\mathit{Bias}\left[\hat{f}_{\log}(x)\right] &= \mathbb{E}\left[\hat{f}_{\log}(x)\right] - f_X(x) \\ &= \frac{h^2}{2}\left[f_X(x) + 3x f_X^{(1)}(x) + x^2 f_X^{(2)}(x)\right] + o\left(h^2\right), \end{aligned} \tag{5}$$

$$\mathit{Var}\left[\hat{f}_{\log}(x)\right] = \frac{1}{nhx} f_X(x) \int_{\mathbb{R}} K^2(z)\,dz + o\left(\frac{1}{nh}\right), \tag{6}$$

and

$$\begin{aligned}\mathit{MSE}\left[\hat{f}_{\log}(x)\right] &= \mathit{Var}\left[\hat{f}_{\log}(x)\right] + \mathit{Bias}^2\left[\hat{f}_{\log}(x)\right] \\ &= \frac{1}{nhx} f_X(x) \int_{\mathbb{R}} K^2(z)\,dz \\ &\quad + \frac{h^4}{4}\left[f_X(x) + 3x f_X^{(1)}(x) + x^2 f_X^{(2)}(x)\right]^2 \\ &\quad + o\left(\frac{1}{nh} + h^4\right), \end{aligned} \tag{7}$$

respectively.

Proof. For (5), we begin by noting that the change-of-variable formula and (3) and (4) implies that $\text{Bias}\left[\hat{f}_{\log}\left(x\right)\right] = x^{-1}\left[\mathbb{E}\left[\hat{f}\left(\log x\right)\right] - f_Y\left(\log x\right)\right] = \left(2x\right)^{-1} h^2 f_Y^{(2)}\left(\log x\right) + o\left(h^2\right)$ (cf. [12]). Now note that $f_Y^{(2)}\left(y\right) = e^y f_X\left(e^y\right) + 3e^{2y} f_X^{(1)}\left(e^y\right) + e^{3y} f_X^{(2)}\left(e^y\right)$ and make the substitution $y = \log x$ in order to obtain final expression.

For (6), we use the change-of-variable formula to obtain $\text{Var}\left[\hat{f}_{\log}\left(x\right)\right] = \text{Var}\left[x^{-1}\hat{f}\left(\log x\right)\right] = x^{-2}\text{Var}\left[\hat{f}\left(\log x\right)\right]$, which we then use (4) in order to get $\text{Var}\left[\hat{f}_{\log}\left(x\right)\right] = \left(nhx\right)^{-2} f_Y\left(\log x\right)\int_{\mathbb{R}} K^2\left(z\right)\mathrm{d}z + o\left(\left[nh\right]^{-1}\right)$. A final substitution of $x^{-1}f_X\left(x\right) = f_Y\left(\log x\right)$ yields the final expression. Expression (7) is obtained by definition. □

Let $h = h_n > 0$ be a positive sequence of bandwidths that satisfies the classical assumptions (D1) $\lim_{n\to\infty} h_n = 0$ and (D2) $\lim_{n\to\infty} nh_n = \infty$. That is, h_n approaches zero at a rate that is slower than n^{-1}. Under (D1) and (D2), we have obtain the pointwise unbiasedness and consistency of (2) as an estimator for $f_X\left(x\right)$.

Proposition 2. *For any $x \in (0,\infty)$, under (D1) and (D2), Bias $\left[\hat{f}_{\log}\left(x\right)\right] \to 0$ and MSE $\left[\hat{f}_{\log}\left(x\right)\right] \to 0$, as $n \to \infty$.*

Proof. Both results follow by making the substitution $h = h_n$ in (5) and (7), respectively, followed by evaluating the limits as $n \to \infty$. □

Remark 1. As noted by [4], the performance of the log-KDE method is most hindered by the behavior of the estimand $f_X\left(x\right)$, when $x = 0$, because of the $x^{-1}f_X\left(x\right)$ term in (6). If this expression is large at $x = 0$, then we can expect that the log-KDE will exhibit high levels of variability and a large number of observations n may be required in order to mitigate such effects. From the bias expressions (5), we also observe influences from expressions of form $xf_X^{(1)}\left(x\right)$ and and $x^2 f_X^{(2)}\left(x\right)$. This implies that there may be a high amount of bias when estimating $f_X\left(x\right)$ at values where x is large and $f_X\left(x\right)$ is either rapidly changing or the curvature of $f_X\left(x\right)$ is rapidly changing. Fortunately, in the majority of estimating problems over the domain $(0,\infty)$, both $f_X^{(1)}\left(x\right)$ and $f_X^{(2)}\left(x\right)$ tend to be decreasing in x, hence such effects should not be overly consequential, in general.

2.2 Integrated Results

We denote the asymptotic MISE between a density estimator and an estimand as the AMISE. From the general results of [23], we have the identity

$$\begin{aligned}\operatorname{MISE}\left[\hat{f}_{\log}\right] &= \int_0^\infty \operatorname{MSE}\left[\hat{f}_{\log}(x)\right]\mathrm{d}x \\ &= \operatorname{AMISE}\left[\hat{f}_{\log}\right] + o\left(\frac{1}{nh} + h^4\right),\end{aligned}$$

where

$$\begin{aligned}\operatorname{AMISE}\left[\hat{f}_{\log}\right] &= \frac{1}{nh}\mathbb{E}\left[X^{-1}\right]\int_{\mathbb{R}} K^2(z)\,\mathrm{d}z \\ &\quad + \frac{h^4}{4}\int_0^\infty \left[f_X(x) + 3xf_X^{(1)}(x) + x^2 f_X^{(2)}(x)\right]^2 \mathrm{d}x.\end{aligned}$$

By a standard argument

$$\begin{aligned}h^* &= \arg\inf_{h>0}\ \operatorname{AMISE}\left[\hat{f}_{\log}\right] \qquad (8)\\ &= \left[\frac{\mathbb{E}\left[X^{-1}\right]\int_{\mathbb{R}} K^2(z)\,\mathrm{d}z}{\int_0^\infty \left[f_X(x) + 3xf_X^{(1)}(x) + x^2 f_X^{(2)}(x)\right]^2 \mathrm{d}x}\right]^{1/5} n^{-1/5},\end{aligned}$$

and

$$\inf_{h>0}\ \operatorname{AMISE}\left[\hat{f}_{\log}\right] = \frac{5}{4}\left[\int_{\mathbb{R}} K^2(z)\,\mathrm{d}z\right]^{4/5} Jn^{-4/5}, \qquad (9)$$

where

$$J = \left(\mathbb{E}^4\left[X^{-1}\right]\int_0^\infty \left[f_X(x) + 3xf_X^{(1)}(x) + x^2 f_X^{(2)}(x)\right]^2 \mathrm{d}x\right)^{1/5}.$$

Using expression (8), we can derive a plugin bandwidth estimator for common interesting pairs of kernels $K(y)$ beand estimands $f_X(x)$. For example, we may particularly interesting in obtaining an optimal bandwidth h^* for scenario where we take $K(y)$ to be normal and $f_X(x)$ to be log-normal with scale parameter $\sigma^2 > 0$ and location parameter $\mu \in \mathbb{R}$. This scenario is analogous to the famous rule of thumb from [19, Sec. 3.4].

Proposition 3. *Let $K(y)$ be normal, as per Table 1 and let $f_X(x)$ be log-normal, with the form*

$$f_X(x) = \frac{1}{x\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left[\frac{\log x - \mu}{\sigma}\right]^2\right).$$

If we estimate $f_X(x)$ by a log-KDE of form (2), then the bandwidth that minimizes $AMISE\left[\hat{f}_{\log}\right]$ is

$$h^* = \left[\frac{8\exp\left(\sigma^2/4\right)}{\sigma^4 + 4\sigma^2 + 12}\right]^{1/5} \frac{\sigma}{n^{1/5}}, \tag{10}$$

and

$$\inf_{h>0} \; AMISE\left[\hat{f}_{\log}\right] = \frac{5}{16}\left(\frac{2}{\pi}\right)^{2/5} n^{-4/5} J,$$

where

$$J = \frac{1}{2}\frac{\exp\left(9\sigma^2/20\right)\left(\sigma^4 + 4\sigma^2 + 12\right)^{1/5}}{\pi^{1/10}\sigma}.$$

Proof. Via some calculus, we find that $\mathbb{E}\left[X^{-1}\right] = \exp\left(\sigma^2/2 - \mu\right)$, $\int_{\mathbb{R}} K^2(z)\,\mathrm{d}z = 1/\left(4\sqrt{\pi}\right)$, and

$$\int_0^\infty \left[f_X(x) + 3xf_X^{(1)}(x) + x^2 f_X^{(2)}(x)\right]^2 \mathrm{d}x = \frac{\exp\left(\sigma^2/4 - \mu\right)\left(\sigma^4 + 4\sigma^2 + 12\right)}{32\sigma^5}.$$

The desired results are obtained by substituting these expressions into expressions (8) and (9). □

Remark 2. In general, one does not know the true estimand $f_X(x)$, or else the problem of density estimation becomes trivialized. However, as a guideline, the log-normal density function can be taken as reasonably representative with respect to the class of densities over the $(0, \infty)$. As such, the plugin bandwidth estimator (10) can be used in order to obtain a log-KDE with reasonable AMISE value. Considering that the true parameter value σ^2 is also unknown, estimation of this quantity is also required before (10) can be made useful. If $\{X_i\}_{i=1}^n$ is a sample that arises from a log-normal density with parameters σ^2 and μ, then $\{Y_i\}_{i=1}^n$ ($Y_i = \log X_i$, $i \in [n]$) is a sample that arises from a normal density with the same parameters. Thus, faced with $\{X_i\}_{i=1}^n$, one may take the logarithmic transformation of the data and compute the sample variance of the data to use as an estimate for σ^2. Alternatively, upon taking the logarithmic transformation, any estimator for σ^2 with good properties can be used. For example, one can use the interquartile range divided by 1.349^2.

Rule (10) is by no means the only available technique for setting the bandwidth h when performing log-KDE. An alternative to using rule (10) is to utilize the classic rule from [19, Sec. 3.4], based on minimizing the AMISE with respect to the estimator of form (1) using normal kernels, for estimating normal densities. In the context of this paper, this rule is applied by firstly transforming the data $\{X_i\}_{i=1}^n$ to the log-transformed data $\{Y_i\}_{i=1}^n$ and then computing the bandwidth as $h = (4/3)^{1/5} \sigma n^{-1/5}$, where σ^2 is the variance of Y_i ($i \in [n]$). In general σ^2 is unknown and thus we must again estimate σ^2 by the sample variance or some other estimator of the variance with good properties.

Apart from the two aforementioned plugin bandwidth estimators, we can also utilize more computationally intensive methodology for choosing the bandwidth h, such as cross-validation (CV) procedures that are discussed in [19, Ch. 3] or the improved efficiency estimator of [18]. The implementations of each of the mentioned methods for bandwidth selection in the `logKDE` package are discussed in further detail in the following section.

3 The `logKDE` package

The `logKDE` package can be installed from github and loaded into an active `R` session using the following commands:

```
install.packages("logKDE", repos='http://cran.us.r-project.org')
```

The `logKDE` package seeks replicate the syntax and replicate the functionality of the KDE estimation function, `density`, built into the base *R* `stats` package. The two main functions included in the package, `logdensity` and `logdensity_fft`, both return Density objects, which are of the same form as those produced by `density`. This enables the efficient reuse of functions such as `plot` and `print`, included in the `stats` package. While the function parameters are very similar to those for `density`, there are a number of minor differences. The parameters of the function are as given below.

adjust The default value is 1. This parameter adjusts the calculated BW directly to enable to easy manipulation of the KDE smoothness.

weights By default all samples are weighted equally however if a vector of the length as the input vector is provided, the samples will be weighted accordingly.

n The number of points at which the KDE is computed. For `logdensity_fft` the value of n must be a power of 2 and the points will be evenly spaced on a log scale. In the case of `logdensity`, there is no restriction on the value of n and the points are evenly spaced on a linear scale.

from, to If unspecified the values of from and to are calculated as *cut* mutliplied by the bandwidth beyond the extremes of the data. However the lower bound will always be equal to or greater than 0.001. `from` and `to` can also be specified directly.

cut Defaults to 3. Determines the range over which the KDE is computed if not specified directly through `from` and `to`.

The `logKDE` package also provides two new bandwidth estimation functions, `bw.logCV` and `bw.logG`, as well as a range of Log-Domain kernel functions. These can be selected using the `kernel` and `bw` parameters in `logdensity` and `logdensity_fft`. These are described in detail the following sections.

3.1 Kernels

All of the kernels described in Table 1 are available in `logKDE`. They can be chosen via the `kernel` parameter in `logdensity` and `logdensity_fft`.

Epanechnikov `epanechnikov`

normal `normal`

Laplace `laplace`

logistic `logistic`

Triangular `triangular`

Uniform `uniform`

Note that `uniform` is referred to as "Rectangular" in `stats`. By default we set `kernel="Gaussian"` (i.e. normal). This choice was made to conform with the default settings of the `density` function.

3.2 Bandwidth selection

The available Bandwidth selection methods with the `logKDE` package include all of those from `stats` as well as two new bandwidth (BW) methods.

nrd0 Silverman

nrd Scott

ucv Unbiased cross-validation

bcv Biased cross-validation

bw.SJ Sheather-Jones

bw.logG Modified Silverman for left-truncated data.

bw.logCV Log-CV for left-truncated data.

The equation $h = 0.9n^{-1/5} \times \min\{\sigma, \mathrm{IQR}/1.34\}$ is used to compute the `nrd0` bandwidth [19]. The `nrd` bandwidth is the same as `nrd0`, except that 0.9 is replaced by 1.06. The bandwidths `ucv` and `bcv` are computed as per the descriptions of [17], performed on the log-transformed data. The `bw.SJ` bandwidth is computed as per the description of [18] `bw.logG` bandwidth utilizes Equation (10). Finally, `bw.logCV` computes unbiased cross-validation bandwidths using untransformed data, rather than the log-transformed data that are used by `ucv`; see [4] for details.

3.3 Plotting and visualization

The reuse of functionality from base `R` packages allows intuitive visualization of the densities estimated using `logKDE`. This is illustrated in the following simple example (see Figure 1):

```
> chisq10<-rchisq(100,10)

> plot(logdensity(chisq10))
```

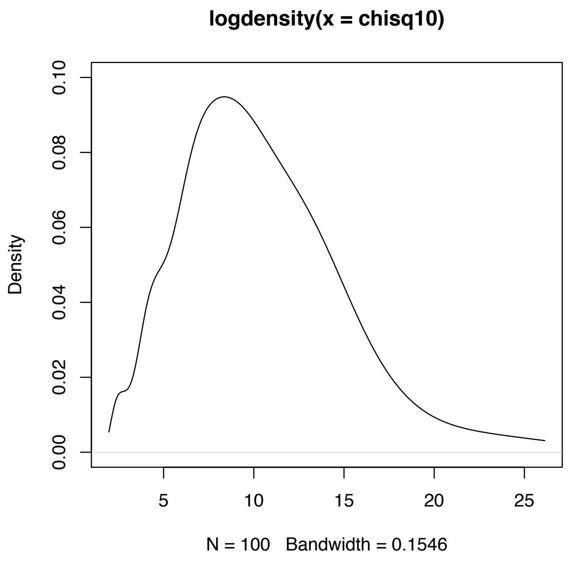

Figure 1: A basic example of the use of `logdensity` class from `R` package `logKDE`.

```
> print(fit1)

Call:

logdensity(x = chisq10)

Data: chisq10 (100 obs.); Bandwidth 'bw' = 0.1546

       x                y

 Min.   : 1.966   Min.   :0.003108

 1st Qu.: 8.007   1st Qu.:0.008810

 Median :14.048   Median :0.034050
```

```
Mean   :14.048   Mean   :0.040812
3rd Qu.:20.089   3rd Qu.:0.071367
Max.   :26.131   Max.   :0.094840
```

The shared syntax and class structure between `logdensity` and `density` allows for the simple creation of more complex graphical objects. Additionally, via a range of settings and options, different bandwidth and kernel preferences can be easily accessed, as can be seen in the following example (see Figure 2):

```
> fit<-logdensity(chisq10, bw ="logCV", kernel = "triangular")
> plot(fit, ylim=c(0, .1))
> grid(10,10,2)
> x<-(seq(min(chisq10), max(chisq10), 0.01))
> lines(x, dchisq(x,10), col =4)
```

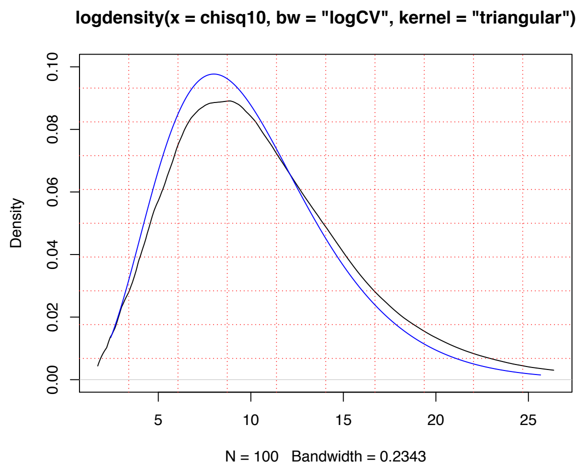

Figure 2: Another example of the use of `logdensity` class from `R` package `logKDE`. In this case, the bandwidth is selected using the CV method and a triangular kernel is used. The χ^2_{10} reference distribution is marked in blue, whereas the log-KDE is plotted in black.

4 Numerical Results

4.1 Simulation studies

Simulation studies for a range of scenarios were conducted. These scenarios correspond to those in [4]. The performance of the `logKDE` package was compared with those of the methods from `stats` and `Conake` [24]. For The first two packages, all available kernels were compared. For `Conake`, only the gamma and reciprocal inverse Gaussian (RIG) kernels were considered. The remaining available kernels from the `Conake` are the extended beta and the log-normal kernels. The extended beta kernel is only suitable for bounded interval domains, and the log-normal kernel is already considered in `logKDE`.

For each of the kernels, three different BW selection methods were compared. The Silverman method is as described in Section 3.2, the log-Silverman is our new method (`bw.logG`), and CV refers contextually and respectively to our untransformed unbiased-CV for `logKDE` kernel methods, the built-in CV method for kernels from `stats` (i.e. `bw.ucv`), or the built-in CV method for kernels from `Conake` (i.e. `cvbw`).

Random samples were drawn from three classes of test distributions: the log-normal, the left-truncated normal, and the Singh-Maddala [20] distributions. A number of different parameter values were used for each distribution. The log-normal samples were drawn from log-normal distributions with mean 0 and standard deviations of 0.5,0.1, and 2 depending on the simulation scenario. The left-truncated normal samples are taken from a normal distribution with a mean of 1 that has been truncated at 0. Standard deviations of 0.5,0.1, and 1.5 are used in each scenario. The Singh-Maddala distribution is as it was considered in [4], with a scale parameter of 0.193, shape parameter a set to 2.8, and shape parameter q set for each scenario as one of 1.145, 1.07 and 0.75. Sample sizes of 20, 50 and 100 were drawn from each distribution, in each scenario. For each scenario, 1000 replications of the sampling process were conducted and the average MISE and MIAE were calculated for each combination of the kernel density estimator, kernel type, and bandwidth estimator.

The average MISE values were calculated as

$$\frac{1}{M}\sum_{i=1}^{M}\int_{0}^{\infty}\left[\hat{f}(y_i)-f(y)\right]^2,$$

where y_{ij} is the ith replicate sample drawn from $f(y)$. Here, $M = 1000$ is the number of replications of the experiment, for each scenario. The integral is numerically computed using the `trapz` function from the R package `pracma` [2]. The average MIAE values were calculated as

$$\frac{1}{M}\sum_{i=1}^{M}\int_{0}^{\infty}\left|\hat{f}(y_i)-f(y)\right|.$$

The MIAE and MISE results are presented in Tables 2, 3, 4, 5, 6, and 7, where they are split up with respect to the sample sizes. The results for both MIAE and MISE largely followed the same pattern, as did the results for each of the sample sizes. No single estimator, bandwidth selection method, or kernel dominated across all scenarios, uniformly. Broadly, the `logKDE` package performed best in the log-Normal and Singh-Maddala distribution scenarios. The methods from the `Conake` package also performed well on the Singh-Maddala cases and on the left-truncated normal case, with standard deviation equal to 1.5. The standard KDE from `stats` generally performed well in the left-truncated normal cases, when the standard deviations are smaller.

4.2 Case Studies with real data

An illustrative example of the relative performance of the log-KDE method compared with standard KDE is provided using data taken from [26], These data comprise of 331 salaries of Major League Baseball players for the 1992 season. Both densities were constructed using the default settings of the respective packages and Gaussian kernels and both were estimated over the range [0.0001, 6500]. As can be seen in Figure 4, the log-KDE estimate is qualitatively closer to the histogram of the actual data, particularly for values that are close to the origin.

Another famous strictly positive data set is the daily ozone level data taken from a wider air-quality study [3]. The data consist of 116 daily measurements of ozone concentration in parts

[ht]

Table 2: MIAE calculated from a 1000 replicates with 20 observation each for each of the simulation scenarios and combination of estimator, kernel, and bandwidth selection method. The top 5 results for each scenario are in bold.

Package	Kernel	Bandwidth Method	Log-Normal			Truncated Normal			Singh-Maddala		
			0.5	1.0	2.0	0.5	1.0	1.5	0.75	1.07	1.145
logKDE	Epanechnikov	CV	0.3282	0.2992	0.3396	0.4127	0.3874	0.3662	0.3667	0.3661	0.3675
logKDE	Epanechnikov	log-Silverman	**0.2807**	0.2872	0.5807	0.3227	0.3196	0.3108	0.4390	0.5239	0.5375
logKDE	Epanechnikov	Silverman	**0.2820**	**0.2482**	**0.1969**	0.3597	0.3359	0.3079	**0.3141**	**0.3099**	**0.3003**
logKDE	Gaussian	CV	0.3257	0.2988	0.3453	0.4045	0.3769	0.3574	0.3633	0.3620	0.3617
logKDE	Gaussian	log-Silverman	**0.2763**	0.2774	0.5729	0.3228	0.3112	0.2992	0.4464	0.5359	0.5492
logKDE	Gaussian	Silverman	0.2854	**0.2537**	**0.2012**	0.3608	0.3355	0.3109	**0.3172**	**0.3141**	**0.3040**
logKDE	Laplace	CV	0.3359	0.3148	0.3745	0.4063	0.3735	0.3567	0.3717	0.3726	0.3644
logKDE	Laplace	log-Silverman	0.2854	0.2729	0.5602	0.3412	0.3182	0.2942	0.4898	0.5906	0.6116
logKDE	Laplace	Silverman	0.3155	0.2874	0.2248	0.3828	0.3592	0.3357	0.3474	0.3451	0.3349
logKDE	logistic	CV	0.3262	0.3021	0.3551	0.4021	0.3730	0.3541	0.3643	0.3632	0.3602
logKDE	logistic	log-Silverman	**0.2762**	0.2726	0.5684	0.3266	0.3095	0.2938	0.4592	0.5538	0.5687
logKDE	logistic	Silverman	0.2930	0.2635	**0.2079**	0.3661	0.3407	0.3178	**0.3253**	**0.3231**	**0.3125**
logKDE	triangular	CV	0.3266	0.2981	0.3401	0.4082	0.3811	0.3614	0.3640	0.3631	0.3646
logKDE	triangular	log-Silverman	**0.2781**	0.2816	0.5751	0.3220	0.3147	0.3046	0.4398	0.5259	0.5402
logKDE	triangular	Silverman	0.2822	**0.2493**	**0.1980**	0.3589	0.3345	0.3078	**0.3141**	**0.3103**	**0.3003**
logKDE	uniform	CV	0.3406	0.3106	0.3557	0.4311	0.4072	0.3834	0.3834	0.3833	0.3816
logKDE	uniform	log-Silverman	0.2959	0.3025	0.5886	0.3368	0.3401	0.3293	0.4575	0.5464	0.5599
logKDE	uniform	Silverman	0.2970	**0.2597**	**0.2044**	0.3781	0.3543	0.3224	**0.3304**	**0.3260**	**0.3196**
stats	Epanechnikov	CV	0.3469	0.3559	0.4158	**0.2698**	0.2924	0.2904	0.3825	0.3484	0.3530
stats	Epanechnikov	log-Silverman	0.3519	0.4043	0.6620	**0.2316**	**0.2376**	0.2282	0.4767	0.3798	0.3515
stats	Epanechnikov	Silverman	0.3376	0.3486	0.4417	0.2744	0.2596	0.2458	0.3741	0.3402	0.3313
stats	Gaussian	CV	0.3422	0.3540	0.3959	0.2764	0.2983	0.2969	0.3801	0.3471	0.3492
stats	Gaussian	log-Silverman	0.3387	0.3784	0.6504	**0.2383**	0.2413	0.2306	0.4474	0.3618	0.3393
stats	Gaussian	Silverman	0.3349	0.3432	0.4148	0.2813	0.2662	0.2496	0.3720	0.3414	0.3311
stats	rectangular	CV	0.3618	0.3692	0.4413	0.2789	0.3019	0.2993	0.3951	0.3599	0.3621
stats	rectangular	log-Silverman	0.3712	0.4317	0.6728	**0.2424**	0.2492	0.2399	0.5040	0.3991	0.3691
stats	rectangular	Silverman	0.3539	0.3640	0.4680	0.2865	0.2718	0.2591	0.3886	0.3536	0.3465
stats	triangular	CV	0.3437	0.3536	0.4016	0.2717	0.2938	0.2915	0.3805	0.3476	0.3519
stats	triangular	log-Silverman	0.3447	0.3894	0.6528	**0.2336**	**0.2381**	**0.2281**	0.4602	0.3704	0.3451
stats	triangular	Silverman	0.3349	0.3448	0.4239	0.2760	0.2614	0.2458	0.3720	0.3399	0.3303
Conake	gamma	CV	0.3433	0.2979	0.3794	0.3400	0.2896	0.2583	0.3912	0.4358	0.4571
Conake	gamma	log-Silverman	0.4335	0.2940	0.6809	0.3648	0.2448	**0.1923**	0.5268	0.4710	0.4672
Conake	gamma	Silverman	0.3661	0.2679	0.4152	0.3232	**0.2351**	**0.1929**	0.4077	0.4299	0.4449
Conake	RIG	CV	0.3406	0.3005	0.3555	0.3320	0.2794	0.2644	0.3925	0.4406	0.4617
Conake	RIG	log-Silverman	0.4115	**0.2574**	0.3412	0.3604	**0.2383**	**0.2003**	0.4979	0.4700	0.4729
Conake	RIG	Silverman	0.3655	0.2611	0.3335	0.3259	**0.2324**	**0.2017**	0.4194	0.4441	0.4569

[ht]

Table 3: MISE calculated from a 1000 replicates with 20 observations each for each of the simulation scenarios and combination of estimator, kernel, and bandwidth selection method. The top 5 results for each scenario are in bold.

Package	Kernel	Bandwidth Method	Log-Normal			Truncated Normal			Singh-Maddala		
			0.5	1.0	2.0	0.5	1.0	1.5	0.75	1.07	1.145
logKDE	Epanechnikov	CV	0.07795	0.10206	0.29039	0.12601	0.08973	0.09950	0.4900	0.5797	0.5024
logKDE	Epanechnikov	log-Silverman	**0.04775**	0.09375	0.26858	0.07742	0.06364	0.05306	0.5272	0.9819	1.1256
logKDE	Epanechnikov	Silverman	0.05416	**0.04854**	**0.07955**	0.10214	0.07983	0.06528	**0.2419**	**0.2902**	**0.3062**
logKDE	Gaussian	CV	0.07932	0.10109	0.29614	0.12574	0.08952	0.09759	0.4794	0.5386	0.5095
logKDE	Gaussian	log-Silverman	**0.04663**	0.08226	0.26044	0.07999	0.06431	0.05183	0.5510	1.0284	1.2007
logKDE	Gaussian	Silverman	0.05676	**0.05026**	**0.08000**	0.10487	0.08217	0.06718	0.2510	0.3046	0.3182
logKDE	Laplace	CV	0.09053	0.11080	0.35059	0.14292	0.10041	0.10573	0.5213	0.5560	0.5588
logKDE	Laplace	log-Silverman	**0.05202**	0.06884	0.24550	0.09759	0.07641	0.05736	0.6944	1.2725	1.5827
logKDE	Laplace	Silverman	0.07244	0.06265	0.09271	0.12804	0.09993	0.08421	0.3098	0.3923	0.4049
logKDE	logistic	CV	0.08211	0.10322	0.31327	0.12959	0.09188	0.09888	0.4879	0.5326	0.5243
logKDE	logistic	log-Silverman	**0.04715**	0.07570	0.25562	0.08471	0.06727	0.05268	0.5896	1.1003	1.3078
logKDE	logistic	Silverman	0.06095	0.05368	0.08310	0.11082	0.08696	0.07123	0.2665	0.3286	0.3408
logKDE	triangular	CV	0.07826	0.10070	0.28910	0.12488	0.08904	0.09748	0.4781	0.5450	0.4997
logKDE	triangular	log-Silverman	**0.04700**	0.08691	0.26171	0.07768	0.06343	0.05196	0.5330	0.9852	1.1532
logKDE	triangular	Silverman	0.05482	**0.04872**	**0.07900**	0.10270	0.08002	0.06569	**0.2441**	0.2946	**0.3082**
logKDE	uniform	CV	0.08280	0.11495	0.31449	0.13818	0.09713	0.12898	0.5471	0.6905	0.5314
logKDE	uniform	log-Silverman	0.05372	0.10892	0.27880	0.08390	0.07130	0.06017	0.5691	1.0592	1.1983
logKDE	uniform	Silverman	0.05983	0.05390	0.08938	0.11551	0.08740	0.07250	0.2662	0.3215	0.3469
stats	Epanechnikov	CV	0.06900	0.05916	0.17570	0.06135	0.04439	0.03452	0.2597	0.2959	0.3657
stats	Epanechnikov	log-Silverman	0.06792	0.09297	0.31656	**0.03710**	**0.02500**	**0.01822**	0.4815	0.3851	0.3443
stats	Epanechnikov	Silverman	0.06239	0.05982	0.19862	0.05825	0.03141	0.02122	**0.2421**	**0.2652**	**0.2797**
stats	Gaussian	CV	0.06802	0.05824	0.16631	0.06567	0.04725	0.03663	0.2580	0.2982	0.3646
stats	Gaussian	log-Silverman	0.06221	0.08342	0.31181	**0.04006**	**0.02605**	0.01859	0.4287	0.3481	0.3149
stats	Gaussian	Silverman	0.06188	0.05591	0.18711	0.06225	0.03310	0.02211	**0.2380**	**0.2679**	**0.2844**
stats	rectangular	CV	0.07394	0.06285	0.18532	0.06378	0.04624	0.03595	0.2750	0.3125	0.3832
stats	rectangular	log-Silverman	0.07597	0.10217	0.32065	**0.04014**	0.02724	0.02001	0.5321	0.4242	0.3826
stats	rectangular	Silverman	0.06852	0.06572	0.20957	0.06267	0.03413	0.02328	0.2622	**0.2863**	0.3102
stats	triangular	CV	0.06820	0.05820	0.16829	0.06283	0.04527	0.03514	0.2583	0.2975	0.3727
stats	triangular	log-Silverman	0.06477	0.08669	0.31257	**0.03804**	**0.02523**	**0.01818**	0.4478	0.3644	0.3285
stats	triangular	Silverman	0.06165	0.05720	0.18973	0.05947	0.03185	0.02135	**0.2390**	**0.2652**	**0.2813**
Conake	gamma	CV	0.07313	0.06264	0.09257	0.08004	0.04545	0.04119	0.3131	0.5192	0.5892
Conake	gamma	log-Silverman	0.10366	0.05921	0.23399	0.06605	**0.02355**	**0.01377**	0.5964	0.7255	0.7824
Conake	gamma	Silverman	0.07816	**0.05334**	0.09885	**0.05612**	**0.02368**	**0.01468**	0.4132	0.5591	0.5993
Conake	RIG	CV	0.07487	0.05968	0.08186	0.08118	0.04600	0.04148	0.3271	0.5479	0.6245
Conake	RIG	log-Silverman	0.10333	0.05497	**0.04973**	0.07130	0.02679	**0.01798**	0.6241	0.8121	0.8889
Conake	RIG	Silverman	0.08359	**0.05102**	**0.06137**	0.06218	0.02689	0.01870	0.4695	0.6680	0.7277

[ht]

Table 4: MIAE calculated from a 1000 replicates with 50 observations each for each of the simulation scenarios and combination of estimator, kernel, and bandwidth selection method. The top 5 results for each scenario are in bold.

Package	Kernel	Bandwidth Method	Log-Normal			Truncated Normal			Singh-Maddala		
			0.5	1.0	2.0	0.5	1.0	1.5	0.75	1.07	1.145
logKDE	Epanechnikov	CV	0.2179	0.2068	0.2635	0.2614	0.2780	0.2569	0.2537	0.2444	0.2219
logKDE	Epanechnikov	log-Silverman	0.1890	0.2473	0.7032	0.2151	0.2329	0.2314	0.2877	0.3608	0.3706
logKDE	Epanechnikov	Silverman	**0.1820**	**0.1598**	**0.1279**	0.2187	0.2270	0.2103	**0.2134**	**0.2115**	**0.2102**
logKDE	Gaussian	CV	0.2192	0.2090	0.2711	0.2590	0.2706	0.2535	0.2544	0.2431	0.2213
logKDE	Gaussian	log-Silverman	**0.1878**	0.2364	0.6997	0.2146	0.2273	0.2213	0.2928	0.3701	0.3784
logKDE	Gaussian	Silverman	**0.1865**	**0.1638**	**0.1320**	0.2200	0.2280	0.2105	**0.2167**	**0.2134**	**0.2137**
logKDE	Laplace	CV	0.2340	0.2246	0.3014	0.2643	0.2687	0.2599	0.2701	0.2526	0.2330
logKDE	Laplace	log-Silverman	0.1974	0.2193	0.6889	0.2282	0.2317	0.2130	0.3255	0.4110	0.4193
logKDE	Laplace	Silverman	0.2125	0.1862	0.1506	0.2388	0.2459	0.2274	0.2393	0.2356	0.2369
logKDE	logistic	CV	0.2225	0.2133	0.2812	0.2587	0.2675	0.2536	0.2582	0.2448	0.2236
logKDE	logistic	log-Silverman	0.1887	0.2278	0.6969	0.2171	0.2258	0.2155	0.3022	0.3831	0.3908
logKDE	logistic	Silverman	0.1938	**0.1701**	**0.1373**	0.2243	0.2321	0.2144	**0.2231**	**0.2189**	0.2204
logKDE	triangular	CV	0.2180	0.2072	0.2657	0.2596	0.2742	0.2547	0.2533	0.2432	0.2211
logKDE	triangular	log-Silverman	**0.1883**	0.2425	0.7002	0.2143	0.2298	0.2263	0.2887	0.3631	0.3725
logKDE	triangular	Silverman	**0.1834**	**0.1609**	**0.1293**	0.2182	0.2268	0.2093	**0.2137**	**0.2115**	**0.2108**
logKDE	uniform	CV	0.2263	0.2143	0.2736	0.2722	0.2917	0.2676	0.2646	0.2551	0.2312
logKDE	uniform	log-Silverman	0.1986	0.2570	0.7062	0.2270	0.2466	0.2445	0.3017	0.3776	0.3866
logKDE	uniform	Silverman	0.1932	**0.1677**	**0.1327**	0.2311	0.2382	0.2214	**0.2245**	**0.2235**	0.2211
stats	Epanechnikov	CV	0.2473	0.2659	0.3839	0.1661	0.1952	0.2061	0.2806	0.2647	0.2457
stats	Epanechnikov	log-Silverman	0.2575	0.3713	0.7308	**0.1415**	**0.1737**	0.1686	0.4197	0.2969	0.2877
stats	Epanechnikov	Silverman	0.2331	0.2626	0.4169	**0.1585**	**0.1826**	0.1766	0.2697	0.2528	0.2339
stats	Gaussian	CV	0.2466	0.2681	0.3607	0.1705	0.1988	0.2103	0.2802	0.2636	0.2467
stats	Gaussian	log-Silverman	0.2484	0.3427	0.7260	**0.1451**	**0.1758**	0.1687	0.3894	0.2839	0.2732
stats	Gaussian	Silverman	0.2327	0.2534	0.3869	0.1630	0.1854	0.1778	0.2670	0.2531	0.2350
stats	rectangular	CV	0.2568	0.2748	0.4070	0.1737	0.2009	0.2129	0.2891	0.2728	0.2544
stats	rectangular	log-Silverman	0.2714	0.3970	0.7348	**0.1507**	**0.1815**	0.1773	0.4474	0.3113	0.3047
stats	rectangular	Silverman	0.2431	0.2763	0.4437	0.1676	0.1910	0.1863	0.2813	0.2608	0.2455
stats	triangular	CV	0.2463	0.2658	0.3692	0.1673	0.1957	0.2070	0.2799	0.2641	0.2462
stats	triangular	log-Silverman	0.2526	0.3562	0.7271	**0.1422**	**0.1737**	**0.1677**	0.4035	0.2906	0.2806
stats	triangular	Silverman	0.2323	0.2574	0.3984	0.1595	0.1827	0.1760	0.2675	0.2526	0.2338
Conake	RIG	CV	0.2395	0.2080	0.3222	0.2131	0.1911	0.1701	0.2443	0.2297	**0.2119**
Conake	RIG	log-Silverman	0.3739	0.2206	0.3673	0.3315	0.1982	**0.1507**	0.3598	0.3359	0.3369
Conake	RIG	Silverman	0.3287	0.1938	0.3243	0.2983	0.1861	**0.1469**	0.2644	0.2736	0.2639
Conake	gamma	CV	0.2342	0.2037	0.3575	0.2159	0.1987	0.1681	0.2361	0.2382	**0.2081**
Conake	gamma	log-Silverman	0.3834	0.2612	0.7434	0.3260	0.2018	**0.1457**	0.4247	0.3779	0.3780
Conake	gamma	Silverman	0.3216	0.2044	0.3889	0.2859	0.1868	**0.1411**	0.2739	0.2871	0.2692

[ht]

Table 5: MISE calculated from a 1000 replicates with 50 observations each for each of the simulation scenarios and combination of estimator, kernel, and bandwidth selection method. The top 5 results for each scenario are in bold.

Package	Kernel	Bandwidth Method	Log-Normal			Truncated Normal			Singh-Maddala		
			0.5	1.0	2.0	0.5	1.0	1.5	0.75	1.07	1.145
logKDE	Epanechnikov	CV	0.0375	0.0474	0.1755	0.0464	0.0519	0.0398	0.2753	0.1975	0.1674
logKDE	Epanechnikov	log-Silverman	**0.0223**	0.0538	0.3183	0.0314	0.0295	0.0230	0.2134	0.4465	0.4845
logKDE	Epanechnikov	Silverman	**0.0232**	**0.0193**	**0.0393**	0.0348	0.0343	0.0261	**0.1098**	**0.1372**	0.1460
logKDE	Gaussian	CV	0.0391	0.0500	0.1824	0.0468	0.0556	0.0408	0.2852	0.2034	0.1709
logKDE	Gaussian	log-Silverman	0.**0222**	0.0479	0.3150	0.0319	0.0300	0.0223	0.2226	0.4728	0.5118
logKDE	Gaussian	Silverman	0.0245	**0.0200**	**0.0400**	0.0355	0.0356	0.0269	**0.1140**	**0.1433**	0.1526
logKDE	Laplace	CV	0.0471	0.0596	0.2135	0.0540	0.0712	0.0492	0.3431	0.2386	0.1959
logKDE	Laplace	log-Silverman	0.0253	0.0394	0.3052	0.0382	0.0366	0.0240	0.2796	0.5996	0.6555
logKDE	Laplace	Silverman	0.0324	0.0252	0.0478	0.0436	0.0457	0.0328	0.1419	0.1829	0.1924
logKDE	logistic	CV	0.0413	0.0534	0.1930	0.0485	0.0598	0.0427	0.2993	0.2133	0.1771
logKDE	logistic	log-Silverman	**0.0227**	0.0437	0.3125	0.0333	0.0314	0.0224	0.2379	0.5103	0.5526
logKDE	logistic	Silverman	0.0266	**0.0214**	0.0421	0.0375	0.0379	0.0285	0.1214	0.1537	0.1634
logKDE	triangular	CV	0.0380	0.0483	0.1771	0.0461	0.0541	0.0402	0.2777	0.1992	0.1680
logKDE	triangular	log-Silverman	**0.0222**	0.0510	**0.3153**	0.0312	0.0296	0.0225	**0.2158**	**0.4534**	0.4938
logKDE	triangular	Silverman	0.0235	**0.0194**	0.0392	0.0346	0.0346	0.0262	0.1106	0.1387	0.1476
logKDE	uniform	CV	0.0400	0.0501	0.1910	0.0503	0.0532	0.0423	0.2895	0.2122	0.1786
logKDE	uniform	log-Silverman	0.0248	0.0597	0.3216	0.0354	0.0324	0.0256	0.2316	0.4890	0.5294
logKDE	uniform	Silverman	0.0259	**0.0217**	0.0415	0.0392	0.0375	0.0289	0.1213	0.1529	0.1607
stats	Epanechnikov	CV	0.0355	0.0330	0.1806	0.0219	0.0175	0.0170	0.1336	0.1713	0.1504
stats	Epanechnikov	log-Silverman	0.0386	0.0861	0.3487	**0.0128**	0.0130	**0.0109**	0.4096	0.2447	0.2582
stats	Epanechnikov	Silverman	0.0299	0.0427	0.2029	**0.0164**	0.0145	0.0117	0.1201	0.1502	**0.1296**
stats	Gaussian	CV	0.0354	0.0327	0.1700	0.0237	0.0184	0.0178	0.1338	0.1709	0.1526
stats	Gaussian	log-Silverman	0.0352	0.0759	0.3470	**0.0137**	**0.0134**	0.0108	0.3524	0.2186	0.2269
stats	Gaussian	Silverman	0.0297	0.0381	0.1908	0.0178	0.0151	0.0118	**0.1158**	0.1502	**0.1312**
stats	rectangular	CV	0.0379	0.0352	0.1901	0.0234	0.0185	0.0180	0.1420	0.1816	0.1608
stats	rectangular	log-Silverman	0.0428	0.0953	0.3501	**0.0145**	0.0142	0.0120	0.4610	0.2697	0.2884
stats	rectangular	Silverman	0.0327	0.0474	0.2137	0.0182	0.0159	0.0130	0.1322	0.1617	0.1434
stats	triangular	CV	0.0353	0.0326	0.1725	0.0225	0.0177	0.0172	0.1337	0.1713	0.1517
stats	triangular	log-Silverman	0.0369	0.0797	0.3472	**0.0130**	**0.0130**	**0.0107**	0.3753	0.2315	0.2416
stats	triangular	Silverman	0.0296	0.0399	0.1937	0.0168	0.0146	0.0116	0.1175	0.1499	0.1297
Conake	RIG	CV	0.0366	0.0269	**0.0376**	0.0305	0.0193	0.0129	0.1193	**0.1371**	**0.1251**
Conake	RIG	log-Silverman	0.0892	0.0331	0.0424	0.0586	0.0158	**0.0090**	0.3276	0.3530	0.3921
Conake	RIG	Silverman	0.0695	**0.0287**	0.0367	0.0489	0.0146	**0.0088**	0.1879	0.2298	0.2418
Conake	gamma	CV	0.0345	0.0261	0.0589	0.0295	0.0188	0.0121	**0.1131**	**0.1438**	**0.1194**
Conake	gamma	log-Silverman	0.0847	0.0379	0.2269	0.0513	**0.0141**	**0.0073**	0.3846	0.3824	0.4177
Conake	gamma	Silverman	0.0614	0.0301	0.0654	0.0412	**0.0128**	**0.0073**	0.1826	0.2191	0.2178

[ht]

Table 6: MIAE calculated from a 1000 replicates with 100 observations each for each of the simulation scenarios and combination of estimator, kernel, and bandwidth selection method. The top 5 results for each scenario are in bold.

Package	Kernel	Bandwidth Method	Log-Normal			Truncated Normal			Singh-Maddala		
			0.5	1.0	2.0	0.5	1.0	1.5	0.75	1.07	1.145
logKDE	Epanechnikov	CV	0.1654	0.1600	0.28204	0.2115	0.2042	0.1864	0.1785	0.1904	0.1864
logKDE	Epanechnikov	log-Silverman	0.1537	0.2148	0.64859	0.1849	0.1868	0.1803	0.2054	0.2568	0.2782
logKDE	Epanechnikov	Silverman	**0.1447**	**0.1314**	**0.09499**	0.1864	0.1744	0.1621	**0.1552**	**0.1567**	**0.1558**
logKDE	Gaussian	CV	0.1660	0.1625	0.28869	0.2111	0.2005	0.1835	0.1805	0.1911	0.1878
logKDE	Gaussian	log-Silverman	0.1532	0.2041	0.64743	0.1851	0.1826	0.1735	0.2110	0.2620	0.2847
logKDE	Gaussian	Silverman	**0.1467**	**0.1336**	**0.09836**	0.1876	0.1744	0.1630	**0.1583**	**0.1599**	**0.1590**
logKDE	Laplace	CV	0.1768	0.1761	0.31960	0.2189	0.2021	0.1869	0.1952	0.2035	0.1994
logKDE	Laplace	log-Silverman	0.1589	0.1873	0.64383	0.1961	0.1831	0.1685	0.2378	0.2919	0.3183
logKDE	Laplace	Silverman	0.1639	0.1488	0.11150	0.2015	0.1854	0.1760	0.1785	0.1779	0.1785
logKDE	logistic	CV	0.1686	0.1665	0.29859	0.2124	0.1991	0.1832	0.1846	0.1941	0.1909
logKDE	logistic	log-Silverman	0.1537	0.1957	0.64661	0.1874	0.1804	0.1696	0.2192	0.2709	0.2949
logKDE	logistic	Silverman	**0.1514**	**0.1378**	**0.10246**	0.1910	0.1763	0.1660	**0.1639**	**0.1650**	**0.1646**
logKDE	triangular	CV	0.1652	0.1606	0.28369	0.2109	0.2022	0.1846	0.1788	0.1903	0.1865
logKDE	triangular	log-Silverman	0.1532	0.2104	0.64727	0.1845	0.1850	0.1770	0.2070	0.2577	0.2798
logKDE	triangular	Silverman	**0.1449**	**0.1318**	**0.09610**	0.1864	0.1740	0.1618	**0.1557**	**0.1575**	**0.1564**
logKDE	uniform	CV	0.1716	0.1649	0.29499	0.2183	0.2134	0.1940	0.1859	0.1971	0.1933
logKDE	uniform	log-Silverman	0.1598	0.2230	0.65068	0.1927	0.1955	0.1901	0.2138	0.2684	0.2919
logKDE	uniform	Silverman	**0.1525**	**0.1372**	**0.09897**	0.1935	0.1832	0.1696	**0.1642**	**0.1655**	**0.1644**
stats	Epanechnikov	CV	0.1939	0.2063	0.40386	0.1454	0.1478	0.1649	0.2157	0.2002	0.1960
stats	Epanechnikov	log-Silverman	0.2129	0.3420	0.69131	**0.1292**	**0.1294**	0.1451	0.3707	0.2765	0.2415
stats	Epanechnikov	Silverman	0.1859	0.2200	0.37107	**0.1378**	**0.1337**	0.1496	0.2128	0.1847	0.1814
stats	Gaussian	CV	0.1946	0.2079	0.37775	0.1487	0.1506	0.1670	0.2172	0.2013	0.1978
stats	Gaussian	log-Silverman	0.2058	0.3148	0.68575	**0.1322**	**0.1302**	0.1452	0.3414	0.2596	0.2293
stats	Gaussian	Silverman	0.1853	0.2130	0.34323	0.1412	0.1358	0.1501	0.2108	0.1855	0.1829
stats	rectangular	CV	0.2006	0.2130	0.42706	0.1497	0.1538	0.1699	0.2228	0.2082	0.2062
stats	rectangular	log-Silverman	0.2227	0.3643	0.69616	**0.1344**	0.1364	0.1516	0.3945	0.2906	0.2535
stats	rectangular	Silverman	0.1935	0.2304	0.39506	0.1432	0.1410	0.1574	0.2212	0.1933	0.1903
stats	triangular	CV	0.1938	0.2064	0.38690	0.1463	0.1483	0.1650	0.2160	0.2008	0.1970
stats	triangular	log-Silverman	0.2096	0.3283	0.68758	**0.1300**	**0.1288**	0.1444	0.3558	0.2681	0.2357
stats	triangular	Silverman	0.1853	0.2161	0.35434	0.1387	**0.1337**	0.1489	0.2113	0.1846	0.1817
Conake	gamma	CV	0.1781	0.1651	0.43095	0.1758	0.1511	**0.1327**	0.1833	0.1739	0.1746
Conake	gamma	log-Silverman	0.3564	0.2390	0.70793	0.3076	0.1839	**0.1235**	0.3920	0.3677	0.3521
Conake	gamma	Silverman	0.2975	0.1822	0.35370	0.2750	0.1685	**0.1172**	0.2596	0.2544	0.2551
Conake	RIG	CV	0.1845	0.1709	0.33632	0.1758	0.1455	0.1335	0.1904	0.1833	0.1783
Conake	RIG	log-Silverman	0.3549	0.2042	0.34623	0.3157	0.1839	**0.1290**	0.3465	0.3393	0.3305
Conake	RIG	Silverman	0.3098	0.1735	0.30024	0.2870	0.1709	**0.1235**	0.2611	0.2598	0.2608

[ht]

Table 7: MISE calculated from a 1000 replicates with 100 observations each for each of the simulation scenarios and combination of estimator, kernel, and bandwidth selection method. The top 5 results for each scenario are in bold.

Package	Kernel	Bandwidth Method	Log-Normal			Truncated Normal			Singh-Maddala		
			0.5	1.0	2.0	0.5	1.0	1.5	0.75	1.07	1
logKDE	Epanechnikov	CV	0.02251	0.02167	0.15999	0.03014	0.021007	0.017319	0.08046	0.13133	0.12
logKDE	Epanechnikov	log-Silverman	**0.01509**	0.04094	0.29621	0.02325	0.016432	0.013792	0.09799	0.21467	0.27
logKDE	Epanechnikov	Silverman	**0.01455**	**0.01254**	**0.01701**	0.02479	0.017641	0.015405	**0.05333**	**0.07652**	**0.08**
logKDE	Gaussian	CV	0.02340	0.02239	0.16602	0.03064	0.021111	0.017893	0.08343	0.13506	0.13
logKDE	Gaussian	log-Silverman	**0.01510**	0.03627	0.29459	0.02369	0.016451	0.013420	0.10431	0.22513	0.28
logKDE	Gaussian	Silverman	0.01527	**0.01297**	**0.01754**	0.02549	0.018184	0.015840	**0.05600**	0.08002	0.08
logKDE	Laplace	CV	0.02787	0.02649	0.20495	0.03524	0.024283	0.020674	0.10116	0.16196	0.15
logKDE	Laplace	log-Silverman	0.01691	0.02906	0.29020	0.02816	0.019370	0.014533	0.13725	0.28426	0.37
logKDE	Laplace	Silverman	0.01963	0.01607	0.02126	0.03084	0.023018	0.019627	0.07314	0.10047	0.10
logKDE	logistic	CV	0.02465	0.02354	0.17655	0.03181	0.021805	0.018747	0.08831	0.14186	0.13
logKDE	logistic	log-Silverman	0.01539	0.03265	0.29349	0.02479	0.016992	0.013475	0.11361	0.24195	0.31
logKDE	logistic	Silverman	0.01643	**0.01379**	**0.01857**	0.02688	0.019320	0.016811	**0.06045**	0.08570	0.09
logKDE	triangular	CV	0.02277	0.02182	0.16129	0.03020	0.020883	0.017422	0.08126	0.13251	0.12
logKDE	triangular	log-Silverman	**0.01505**	0.03912	0.29466	0.02322	0.016333	0.013549	0.09989	0.21691	0.27
logKDE	triangular	Silverman	**0.01473**	**0.01262**	**0.01708**	0.02487	0.017750	0.015457	**0.05403**	**0.07723**	**0.08**
logKDE	uniform	CV	0.02380	0.02328	0.17593	0.03222	0.023080	0.018377	0.08602	0.13786	0.13
logKDE	uniform	log-Silverman	0.01641	0.04462	0.29828	0.02565	0.018209	0.015273	0.10560	0.23341	0.29
logKDE	uniform	Silverman	0.01608	**0.01392**	**0.01957**	0.02716	0.019772	0.016972	**0.05961**	0.08496	0.08
stats	Epanechnikov	CV	0.02077	0.01887	0.19859	0.01564	0.011193	0.011201	0.06956	0.09447	0.09
stats	Epanechnikov	log-Silverman	0.02744	0.07517	0.32930	**0.01125**	**0.007626**	0.008731	0.33270	0.23999	0.18
stats	Epanechnikov	Silverman	0.01871	0.03179	0.18131	**0.01308**	**0.008193**	0.008980	0.07149	**0.07254**	**0.07**
stats	Gaussian	CV	0.02089	0.01875	0.18736	0.01677	0.011714	0.011438	0.07080	0.09644	0.09
stats	Gaussian	log-Silverman	0.02492	0.06554	0.32689	**0.01198**	**0.007756**	0.008566	0.28068	0.20460	0.16
stats	Gaussian	Silverman	0.01840	0.02807	0.16914	0.01400	0.008456	0.008931	0.06823	**0.07303**	**0.07**
stats	rectangular	CV	0.02228	0.02015	0.20836	0.01644	0.011859	0.011813	0.07559	0.10308	0.10
stats	rectangular	log-Silverman	0.03016	0.08345	0.33137	**0.01209**	0.008357	0.009463	0.37651	0.27086	0.20
stats	rectangular	Silverman	0.02030	0.03536	0.19192	0.01407	0.008980	0.009813	0.07825	0.08179	0.08
stats	triangular	CV	0.02074	0.01871	0.18997	0.01602	0.011299	0.011183	0.06997	0.09597	0.09
stats	triangular	log-Silverman	0.02626	0.06924	0.32748	**0.01147**	**0.007574**	0.008537	0.30312	0.22091	0.17
stats	triangular	Silverman	0.01850	0.02969	0.17219	0.01336	**0.008202**	0.008835	0.06969	**0.07231**	**0.07**
Conake	gamma	CV	0.02064	0.01771	0.08483	0.02139	0.010414	**0.007444**	0.06511	0.08493	0.08
Conake	gamma	log-Silverman	0.07629	0.03318	0.21035	0.04590	0.011330	**0.004934**	0.33498	0.36933	0.35
Conake	gamma	Silverman	0.05494	0.02633	0.05158	0.03791	0.009949	**0.004624**	0.16395	0.18950	0.19
Conake	RIG	CV	0.02236	0.01945	0.03737	0.02226	0.010750	0.007801	0.07328	0.09711	0.09
Conake	RIG	log-Silverman	0.08356	0.03115	0.03615	0.05304	0.012962	**0.006052**	0.30358	0.35950	0.35
Conake	RIG	Silverman	0.06461	0.02651	0.02938	0.04499	0.011617	**0.005684**	0.18301	0.22326	0.22

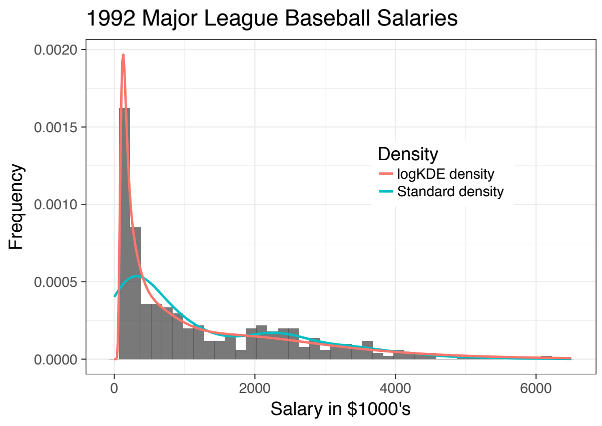

Figure 3: An example comparing KDEs on a real left-truncated dataset, namely the Baseball Salary data from [26].

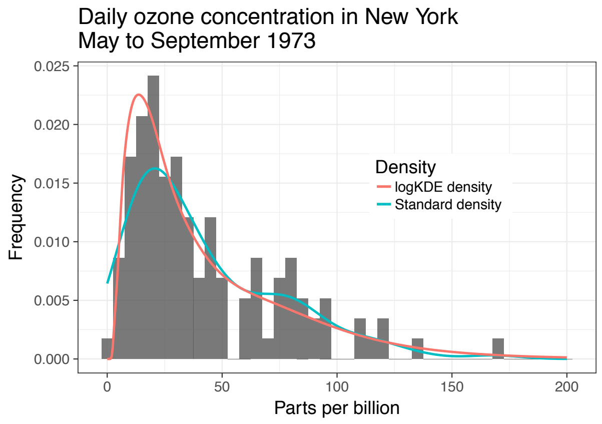

Figure 4: An example comparing KDEs on a real left-truncated dataset, in this case the daily ozone concentration data taken from the air quality dataset in [3].

per billion taken in New York City between May and September 1973. The default settings and Gaussian kernels were used for both estimators, which covered the range [0.0001, 200]. As with the baseball data, the fidelity of the kernel density estimate is improved close to the origin.

5 Conclusions

The log-KDE method works particularly well on left-truncated data with long right tails. The `Conake` package with a gamma kernel seems to be optimal for good for shorter tailed positive data. In addition, there is no substantial difference between the ordinary implementation and the FFT implementation of `logKDE`. As the FFT implementation is much faster, the function `logdensity_fft` is recommended except in cases where a high degree of accuracy is required.

References

[1] T Amemiya. *Introduction to Statistics and Econometrics.* Harvard University Press, Cambridge, 1994.

[2] Hans Werner Borchers. *pracma: Practical Numerical Math Functions*, 2017. R package version 2.0.7.

[3] John M Chambers, William S Cleveland, Beat Kleiner, and Paul A Tukey. *Graphical methods for data analysis.* Wadsworth, Belmont, CA, 1983.

[4] A Charpentier and E Flachaire. Log-transform kernel density estimation of income distribution. *L'Actualite Economique*, 91:141–159, 2015.

[5] S X Chen. Probability density function estimation using gamma kernels. *Annals of the Institute of Statistical Mathematics*, 52:471–480, 2000.

[6] J B Copas and M J Fryer. Density estimation and suicide risks in psychiatric treatment. *Journal of the Royal Statistical Society A*, 143:167–176, 1980.

[7] A DasGupta. *Asymptotic Theory Of Statistics And Probability.* Springer, New York, 2008.

[8] M Hirukawa and M Sakudo. Nonnegative bias reduction methods for density estimation using asymmetric kernels. *Computational Statistics and Data Analysis*, 75:112–123, 2014.

[9] G Igarashi. Weighted log-normal kernel density estimation. *Communications in Statistics - Theory and Methods*, 45:6670–6687, 2016.

[10] G Igarashi and Y Kakizawa. Bias corrections for some asymmetric kernel estimators. *Journal of Statistical Planning and Inference*, 159:37–63, 2015.

[11] X Jin and J Kawczak. Birnbaum-Saunders and lognormal kernel estimators for modelling durations in high frequency financial data. *Annals of Economics and Finance*, 4:103–124, 2003.

[12] J S Marron and D Ruppert. Transformations to reduce boundary bias in kernel density estimation. *Journal of the Royal Statistic Society B*, 56:653–671, 1994.

[13] E Parzen. On etimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[14] R Core Team. *R: a language and environment for statistical computing.* R Foundation for Statistical Computing, 2016.

[15] M Rosenblatt. Remarks on some nonparametric estimates of a density function. *Annals of Mathematical Statistics*, 27:832–835, 1956.

[16] O Scaillet. Density estimation using inverse and reciprocal inverse Gaussian kernels. *Journal of Nonparametric Statistics*, 16:217–226, 2004.

[17] David W. Scott and George R. Terrell. Biased and unbiased cross-validation in density estimation. *Journal of the American Statistical Association*, 82(400):1131–1146, 1987.

[18] S J Sheather and M C Jones. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society B*, 53:683–690, 1991.

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

[20] S. K. Singh and G. S. Maddala. A function for size distribution of incomes. *Econometrica*, 44(5):963–970, 1976.

[21] A van der Vaart. *Asymptotic Statistics*. Cambridge University Press, Cambridge, 1998.

[22] M P Wand and M C Jones. *Kernel Smoothing*. Springer, New York, 1995.

[23] M P Wand, J S Marron, and D Ruppert. Transformations in density estimation. *Journal of the American Statistical Association*, 86:343–353, 1991.

[24] W. E. Wansouwé, F. G. Libengué, and C. C. Kokonendji. *Conake: Continuous Associated Kernel Estimation*, 2015. R package version 1.0.

[25] W E Wansouwé, S M Some, and C C Kokonendji. Ake: an R package for discrete and continuous associated kernel estimations. *R Journal*, 8:258–276, 2016.

[26] Mitchell R Watnik. Pay for play: are baseball salaries based on performance? *Journal of Statistics Education*, 6(2), 1998.