Journal of Statistical Software

January 2022, Volume 101, Issue 2. doi: 10.18637/jss.v101.i02

lpdensity: Local Polynomial Density Estimation and Inference

Matias D. Cattaneo
Princeton University

Michael Jansson
University of California, Berkeley

Xinwei Ma
University of California, San Diego

Abstract

Density estimation and inference methods are widely used in empirical work. When the underlying distribution has compact support, conventional kernel-based density estimators are no longer consistent near or at the boundary because of their well-known boundary bias. Alternative smoothing methods are available to handle boundary points in density estimation, but they all require additional tuning parameter choices or other typically *ad hoc* modifications depending on the evaluation point and/or approach considered. This article discusses the R and Stata package **lpdensity** implementing a novel local polynomial density estimator proposed and studied in Cattaneo, Jansson, and Ma (2020, 2022), which is boundary adaptive and involves only one tuning parameter. The methods implemented also cover local polynomial estimation of the cumulative distribution function and density derivatives. In addition to point estimation and graphical procedures, the package offers consistent variance estimators, mean squared error optimal bandwidth selection, robust bias-corrected inference, and confidence bands construction, among other features. A comparison with other density estimation packages available in R using a Monte Carlo experiment is provided.

Keywords: kernel-based nonparametrics, local polynomial, density estimation, bandwidth selection, bias correction, robust inference, boundary carpentry, R, Stata.

1. Introduction

Nonparametric estimation of a probability density function (PDF), as well as its associated cumulative distribution function (CDF) or higher-order derivatives thereof, plays an important role in empirical work across many disciplines. Sometimes these quantities are the main objects of interest, while in other cases they are useful ingredients in forming more complex nonparametric or semiparametric statistical procedures. See Wand and Jones (1995) and

Fan and Gijbels (1996) for classical textbook introductions to kernel-based density and local polynomial methods.

This article discusses the main methodological and numerical features of the software package **lpdensity**, available in both R (R Core Team 2021) and Stata (StataCorp 2019), which implements the local polynomial smoothing approach proposed and studied in Cattaneo, Jansson, and Ma (2020, 2022) for estimation of and inference on a smooth CDF, PDF, and derivatives thereof. In a nutshell, the idea underlying this estimation approach is to first approximate the discontinuous empirical CDF using local polynomial methods, and then employ that smoothed approximation to construct estimators of the distribution function, density function, and higher-order derivatives.

The resulting local polynomial density estimator is intuitive and easy to implement, and exhibits several interesting theoretical and practical features. For example, it does not require pre-binning or any other complicated pre-processing of the data, and enjoys all of the celebrated features associated with local polynomial regression estimation (Fan and Gijbels 1996). In particular, it automatically adapts to the (possibly unknown) boundaries of the density's support, a feature that is unavailable for most other density estimators in the literature. See Karunamuni and Albert (2005) for a review on this topic. Two exceptions are the local polynomial density estimators of Cheng, Fan, and Marron (1997) and Zhang and Karunamuni (1998), which require pre-binning of the data or, more generally, pre-estimation of the density near the boundary, thereby introducing additional tuning parameters that need to be chosen for implementation. In contrast, the density estimator implemented in the **lpdensity** package requires choosing only one tuning parameter: the bandwidth entering the local polynomial approximation, for which the package also offers data-driven selectors. Furthermore, following the results in Calonico, Cattaneo, and Farrell (2018, 2022), robust bias-corrected inference methods are also implemented, which allow using mean squared error (MSE) optimal or the integrated mean squared error (IMSE) optimal bandwidth choices when forming confidence intervals or conducting hypothesis testing.

The software implementation covers smooth estimation of the distribution and density function, and derivatives thereof, for any polynomial order at both interior and boundary points. Cattaneo, Jansson, and Ma (2020, 2022) give formal large-sample statistical results for these estimators, including (i) asymptotic expansions of the leading bias and variance, (ii) asymptotic pointwise and uniform normal approximations, (iii) consistent standard error estimators, (iv) consistent data-driven bandwidth selection based on asymptotic MSE expansions of the point estimators, and (v) asymptotically valid uniform inference and confidence bands. Importantly, all these results apply to both interior and boundary points simultaneously. We briefly summarize these results in the upcoming sections, and illustrate them numerically, including a comparison with other methods available in R.

In the remaining of this article, we focus on the R implementation of the software package **lpdensity** available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/package=lpdensity`, but all functionalities are also available in Stata. See Appendix B for more details. The R package includes the following two main functions.

- `lpdensity()`: This function implements the local polynomial approximation to the empirical distribution function for a grid of evaluation points, and offers smooth point estimators of the CDF, PDF, and derivatives thereof. The function takes the bandwidth for each evaluation point as given, and employs the companion function `lpbwdensity()`

for data-driven bandwidth selection whenever the bandwidth is not provided. Inference is implemented by using robust bias correction methods (Calonico, Cattaneo, and Farrell 2018, 2022), and both pointwise confidence intervals and uniform confidence bands are supported. Standard inference methods assuming undersmoothing or ignoring smoothing bias are also available.

- `lpbwdensity()`: This function offers pointwise and integrated MSE-optimal bandwidth selectors for the local polynomial CDF, PDF, and higher-order derivatives estimators implemented in `lpdensity()`. The selectors are rate-optimal for both interior and boundary evaluation points. Under an additional condition on the local polynomial fit discussed below, they are also consistent and hence (I)MSE-optimal. Both rule-of-thumb and direct plug-in implementations are available.

In addition, the methods `coef()`, `confint()`, `plot()`, `print()`, `summary()` and `vcov()` are supported for objects returned by `lpdensity()`, and the methods `coef()`, `print()` and `summary()` are supported for objects returned by `lpbwdensity()`. In particular, the function `plot()`, building on the **ggplot2** package in R (Wickham 2016), can be used to plot the estimated CDF, PDF, or higher-order derivatives for graphical illustration. This function takes the output from `lpdensity()`, and plots both point estimates and confidence intervals/bands for a collection of grid points.

There are several other packages and functions available for kernel-based density estimation in R. Table 1 gives a summary of their functionalities. As shown in that table, the package **lpdensity** is the first to offer consistent estimation of the CDF, PDF and density derivatives for both interior and boundary points, higher-order bias reduction, and valid inference both pointwise (confidence intervals) and uniformly (confidence bands). Section 4 compares the numerical performance of these packages in a simulation study.

This article continues as follows. Section 2 provides a brief, self-contained overview of the main ideas underlying the local polynomial estimators implemented in the package **lpdensity**. Section 3 illustrates the main features of our package. Section 4 showcases its finite-sample performance and compares it with other R packages implementing kernel-based density estimators. Section 5 concludes. We also include two appendices. Appendix A discusses in more detail our data-driven bandwidth selectors, and Appendix B illustrates the Stata implementation of the **lpdensity** package. Installation details, scripts replicating the numerical results reported herein, links to software repositories, and other companion information, can be found in the package's website: `https://nppackages.github.io/lpdensity/`.

2. Methodology and implementation

This section offers a brief overview of the main methods implemented in the R and Stata package **lpdensity**. For formal results, including assumptions, proofs and any other technical details see Cattaneo, Jansson, and Ma (2020, 2022, hereafter CJM).

We assume that $X_1, X_2, \ldots, X_n$ is a random sample from the random variable $X \in \mathcal{X}$, where $F(x)$ denotes its smooth CDF, $f(x)$ denotes its smooth PDF, and $\mathcal{X} \subseteq \mathbb{R}$ denotes its (possibly restricted) support, which can be bounded or unbounded. As it is well known, conventional kernel density estimators will be biased at or near boundary points, and other density estimators must be used if the goal is to estimate a density function on a compact support

Package Function	Density derivative	Valid for boundary	Higher-order bias reduction	Standard error	Valid inference	Confidence bands
KernSmooth (Wand and Ripley 2021)						
`bkde`	×	×	×	×	×	×
`locpoly`	✓	✓	✓	×	×	×
ks (Duong 2007, 2021)						
`kdde`	✓	×	×	×	×	×
`kde`	×	×	×	×	×	×
np (Hayfield and Racine 2008; Racine and Hayfield 2021)						
`npudens`	×	×	×	✓	–	×
`npuniden.boundary`	×	✓	×	✓	–	×
nprobust (Calonico, Cattaneo, and Farrell 2019, 2020)						
`kdrobust`	×	×	×	✓	✓	×
plugdensity (Herrmann and Mächler 2011)						
`plugin.density`	×	×	×	×	×	×
`stats::density`	×	×	×	×	×	×
lpdensity						
`lpdensity`	✓	✓	✓	✓	✓	✓

Table 1: Comparison of R packages and functions. ✓ indicates the feature is available, × indicates the feature is not available, and – indicates that inference is available and valid if undersmoothing is used but that is not the default in the package (and hence inference is invalid by default).

(Karunamuni and Albert 2005, and references therein). The package **lpdensity** implements a simple, easy-to-interpret and boundary adaptive density estimator based on local polynomial methods. As a by-product, the package also offers a smooth local polynomial estimate of the CDF as well as density derivatives. To cover all cases in an unified way, we employ the notation $g^{(\nu)}(x) = \partial^\nu g(x)/\partial x^\nu$ and $g(x) = g^{(0)}(x)$ for any smooth function $g(\cdot)$, and define $F(x) = F^{(0)}(x)$, $f(x) = F^{(1)}(x)$, and derivatives of the density function as $f^{(\nu-1)}(x) = F^{(\nu)}(x)$ with $\nu = 1, 2, \ldots$, with $f(x) = f^{(0)}(x)$.

2.1. Local polynomial distribution and density estimation

To describe the estimators implemented in the package **lpdensity**, consider first the weighted empirical distribution function

$$\hat{F}(x) = \frac{1}{n}\sum_{j=1}^{n} W_j \mathbb{1}(X_j \leq x),$$

where $\mathbb{1}(\cdot)$ is the indicator function, and the weights W_j are introduced for empirical applications such as missing data or counterfactual comparison (see CJM for examples). We assume these weights are normalized so that $\sum_{j=1}^{n} W_j/n = 1$. The package **lpdensity** allows for a possibly estimated weighting scheme embedded in $\hat{F}(x)$, although for simplicity we will assume that each $W_i = 1$ throughout this article. That is, $\hat{F}(x)$ is taken to be the standard root-n consistent empirical distribution function estimator of $F(x)$.

As an alternative to conventional kernel density estimators, consider an estimator that first smooths out $\hat{F}(x)$ using local polynomials, and then constructs an estimator of $f(x)$ (and its derivatives). For $x \in \mathcal{X}$, the estimator implemented in the **lpdensity** package is

$$\begin{aligned}\widehat{\boldsymbol{\beta}}_p(x) &= \underset{\mathbf{b}\in\mathbb{R}^{p+1}}{\arg\min} \sum_{i=1}^{n} \left(\hat{F}(X_i) - \mathbf{r}_p(X_i - x)^\top \mathbf{b}\right)^2 K\left(\frac{X_i - x}{h}\right) \\ &= \left[\tfrac{1}{0!}\hat{F}_p(x) \quad \tfrac{1}{1!}\hat{f}_p(x) \quad \tfrac{1}{2!}\hat{f}_p^{(1)}(x) \quad \cdots \quad \tfrac{1}{p!}\hat{f}_p^{(p-1)}(x)\right]^\top,\end{aligned}$$

where $\mathbf{r}_p(u) = (1, u, u^2, \ldots, u^p)^\top$ is the p-th order polynomial expansion, $K(\cdot)$ is a kernel function such as the uniform or triangular kernel, and h is a positive vanishing bandwidth sequence. The estimator approximates the discontinuous empirical CDF $\hat{F}(x)$ by a smooth local polynomial expansion using the weighting scheme implied by the kernel function, localized around the evaluation point x according to the bandwidth h. CJM showed that

$$\widehat{\boldsymbol{\beta}}_p(x) \xrightarrow{\mathbb{P}} \boldsymbol{\beta}_p(x) = \left[\tfrac{1}{0!}F(x) \quad \tfrac{1}{1!}f(x) \quad \tfrac{1}{2!}f^{(1)}(x) \quad \cdots \quad \tfrac{1}{p!}f^{(p-1)}(x)\right]^\top,$$

as $h \to 0$ and $nh^{2p-1} \to \infty$, where $\xrightarrow{\mathbb{P}}$ denotes convergence in probability. This implies that the least squares coefficients $\widehat{\boldsymbol{\beta}}_p(x)$ are consistent estimators of the CDF, PDF, and derivatives thereof at the evaluation point x.

Therefore, the generic local polynomial distribution estimator takes the form:

$$\hat{F}_p^{(\nu)}(x) = \hat{f}_p^{(\nu-1)}(x) = \nu!\mathbf{e}_\nu^\top \widehat{\boldsymbol{\beta}}_p(x), \qquad 0 \le \nu \le p,$$

where $\mathbf{e}_\nu$ denotes the conformable unit vector that extracts the $(\nu+1)$-th estimated coefficient. This estimator is implemented in the function `lpdensity()`, given a choice of evaluation point x, polynomial degree p, derivative order ν, kernel function K, and bandwidth h. In particular, the local polynomial density estimator is $\hat{f}_p(x) = \hat{F}_p^{(1)}(x) = \mathbf{e}_1^\top \widehat{\boldsymbol{\beta}}_p(x)$, which is implemented via the default `lpdensity(..., p = 2, v = 1)`, employing a quadratic approximation to the empirical distribution function to construct the density estimator $\hat{f}_2(x)$. Similarly, higher-order derivatives of the CDF can be estimated through $\hat{f}_p^{(\nu-1)}(x) = \hat{F}_p^{(\nu)}(x) = \nu!\mathbf{e}_\nu^\top \widehat{\boldsymbol{\beta}}_p(x)$ for $2 \le \nu \le p$. We recommend using a local polynomial that is one order higher than the derivative to be estimated, $p = \nu + 1$, or more generally to set $p - \nu$ odd. Of course, it is possible to achieve more bias reduction by increasing the local polynomial order p.

The `lpdensity()` function employs the triangular kernel by default. Other available options include the uniform kernel and the Epanechnikov kernel. Generally speaking, the uniform kernel delivers a smaller asymptotic variance but a larger asymptotic bias for the resulting point estimator. It is possible to reduce its asymptotic bias by using a kernel that is more concentrated around the origin, such as the triangular kernel, at the cost of increasing the asymptotic variance. The choice of the kernel, however, does not affect the orders of the bias and the variance, and hence this is less of a concern compared to bandwidth selection, which we discuss in more detail below. In addition, Cattaneo, Jansson, and Ma (2022) show that more efficiency gains can be achieved by first including a higher-order polynomial term and then partialling out this term using a minimum distance approach.

The rest of this section outlines the main properties, both statistical and numerical, of the estimator $\hat{F}_p^{(\nu)}(x)$, and discusses other related issues such as bandwidth selection, variance

estimation, and valid (robust bias-corrected) inference. While the smooth CDF estimator $\hat{F}_p(x) = \hat{F}_p^{(0)}(x)$ is useful, the density and higher-order derivatives estimators are perhaps more relevant for empirical work. In particular, as mentioned before, the density estimator is intuitive and very easy to implement, while also being boundary adaptive. Thus, the estimator $\hat{f}_p(x)$ can be computed for all evaluation points on the (possibly restricted) support of X in an automatic and straightforward way. This explains why the main functions in the package **lpdensity** refer to density estimation.

2.2. Mean squared error

The estimator $\hat{F}_p^{(\nu)}(x)$ can be written in the familiar weighted least squares form: $\widehat{\boldsymbol{\beta}}_p(x) = (\mathbf{X}^\top\mathbf{W}\mathbf{X})^{-1}(\mathbf{X}^\top\mathbf{W}\mathbf{Y})$ with $\mathbf{X}$ the usual polynomial design matrix and $\mathbf{W}$ a diagonal matrix consisting of kernel weights. The only difference relative to standard local polynomial regression is that here the "dependent variable" is estimated: $\mathbf{Y} = [\hat{F}(X_1), \hat{F}(X_2), \ldots, \hat{F}(X_n)]^\top$, where $\hat{F}(x)$ is the possibly weighted empirical CDF. Unlike other local polynomial density estimators proposed in the literature (e.g., Cheng, Fan, and Marron 1997; Zhang and Karunamuni 1998), the estimator $\hat{F}_p^{(\nu)}(x)$ does not require pre-binning or any other pre-processing of the data beyond constructing the empirical distribution function. As a result, this estimation approach removes the need of choosing the number, position, and length of the bins in a preliminary histogram estimate.

CJM obtained a general stochastic approximation to the bias and variance of the estimator $\hat{F}_p^{(\nu)}(x)$, $\nu = 0, 1, 2, \ldots, p$, for all evaluation points $x \in \mathcal{X}$. Here we discuss the leading case of density estimation and derivatives thereof. For any choice of polynomial order p, and $1 \leq \nu \leq p$, the variance and bias of $\hat{F}_p^{(\nu)}(x)$ are approximately

$$\begin{aligned}
\mathsf{Var}[\hat{F}_p^{(\nu)}(x)] &= \frac{1}{nh^{2\nu-1}}\mathsf{V}_{\nu,p}(x), \\
\mathsf{Bias}[\hat{F}_p^{(\nu)}(x)] &= h^{p-\nu+1}\left[F^{(p+1)}(x)\mathsf{B}_{1,\nu,p}(x) + h \cdot F^{(p+2)}(x)\mathsf{B}_{2,\nu,p}(x)\right],
\end{aligned}$$

where $\mathsf{V}_{\nu,p}(x)$, $\mathsf{B}_{1,\nu,p}(x)$, and $\mathsf{B}_{2,\nu,p}(x)$ denote quantities that can be constructed directly using only the data, choice of (preliminary) bandwidth, evaluation point, polynomial order, derivative order, and kernel function. That is, all these quantities are in pre-asymptotic form, which has been shown to offer higher-order distributional refinements in the context of local polynomial regression (Calonico, Cattaneo, and Farrell 2018, 2022). Furthermore, it can be shown that $\mathsf{V}_{\nu,p}(x)$, $\mathsf{B}_{1,\nu,p}(x)$, and $\mathsf{B}_{2,\nu,p}(x)$ converge (in probability) to well-defined non-random limits.

Since the above approximations are in pre-asymptotic form and are valid for all evaluation points, we can define a generic pointwise MSE-optimal bandwidth choice as

$$h_{\mathtt{MSE},p}(x) = \operatorname*{arg\,min}_{h>0} \mathsf{MSE}[\hat{F}_p^{(\nu)}(x)] = \operatorname*{arg\,min}_{h>0} \left\{\mathsf{Var}[\hat{F}_p^{(\nu)}(x)] + \mathsf{Bias}[\hat{F}_p^{(\nu)}(x)]^2\right\}.$$

The optimal bandwidth also depends on ν, but we suppress this in the notation to conserve notation. Under standard regularity conditions, $h_{\mathtt{MSE},p}(x)$ is MSE optimal in rates for all evaluation points and choices of p and ν; and is MSE-optimal in constants if either (i) $p - \nu$ is odd or (ii) x is a boundary point. See Appendix A for more details.

We also define the IMSE-optimal bandwidth choice as follows:

$$h_{\texttt{IMSE},p} = \arg\min_{h>0} \int \mathsf{MSE}[\hat{F}_p^{(\nu)}(x)] w(x) \mathrm{d}x,$$

where $w(x)$ denotes a user-chosen weighting scheme. Dependence on ν is again suppressed to ease notation. In practice, the integral will be approximated using the grid points specified in `lpdensity()` or `lpbwdensity()`, allowing for both a uniform weighting ($w(x) = 1$) as well as the empirical distribution weighting.

The MSE-optimal and IMSE-optimal bandwidth selectors, $h_{\texttt{MSE},p}(x)$ and $h_{\texttt{IMSE},p}$, are carefully developed so that they automatically adapt to boundary points, while also retaining their main theoretical features (e.g., rate optimality). In practice, these bandwidths can be computed after replacing the unknown quantities by estimates thereof. We discuss implementation details below.

2.3. Point estimation and robust bias-corrected inference

Both $h_{\texttt{MSE},p}(x)$ and $h_{\texttt{IMSE},p}$, as well as their feasible counterparts, denoted by $\hat{h}_{\texttt{MSE},p}(x)$ and $\hat{h}_{\texttt{IMSE},p}$, can be used to construct MSE-optimal or IMSE-optimal point estimators for the PDF or its derivatives. The package **lpdensity** also allows for CDF estimation and computes an (I)MSE-optimal bandwidth, though we do not provide the details here to conserve space: some stochastic approximations change in non-trivial ways because the CDF estimator is $\sqrt{n}$-consistent. See CJM for more details.

As it is well known in the nonparametric literature, standard Wald-type inference is not valid when an (I)MSE-optimal bandwidth is used to construct the nonparametric point estimator. To be specific, the following distributional approximation holds for the standard Wald-type test statistic based on the local polynomial density estimator constructed using an (I)MSE-optimal bandwidth:

$$\mathsf{T}_{\nu,p}(x) = \frac{\hat{F}_p^{(\nu)}(x) - F^{(\nu)}(x)}{\sqrt{\mathsf{Var}[\hat{F}_p^{(\nu)}(x)]}} \rightsquigarrow \mathcal{N}(\mathsf{bias},\ 1), \qquad 1 \leq \nu \leq p,$$

where $\rightsquigarrow$ indicates convergence in distribution, and $\mathcal{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2. The bias term cannot be dropped in general: if the point estimator $\hat{F}_p^{(\nu)}(x)$ is constructed using an (I)MSE-optimal bandwidth, inference based on the usual "point estimator $\pm$ $z_{1-\alpha/2}$ $\times$ standard error" confidence interval is invalid due to the presence of an asymptotic bias (z_α denotes the α quantile of the standard normal distribution). The mechanical solution to this inferential problem is to undersmooth the point estimator $\hat{F}_p^{(\nu)}(x)$ using an *ad hoc* bandwidth h smaller than $h_{\texttt{MSE}}(x)$ or $h_{\texttt{IMSE}}$. Of course, the function `lpdensity()` allows for this approach by simply running bandwidth selection, estimation, and inference in separate steps (see Section 3 for an illustration).

Calonico, Cattaneo, and Farrell (2018, 2022) showed that undersmoothing is suboptimal for inference under the same assumptions employed to construct an (I)MSE-optimal bandwidth. Instead, it is demonstrably better, in terms of higher-order distributional approximations and asymptotic coverage properties, to employ robust bias correction (RBC). The idea is to bias correct the point estimator and then adjust the variance accordingly. Heuristically, and

abusing notation for simplicity, this leads to the Wald-type test statistic

$$\mathsf{T}^{\mathtt{RBC}}_{\nu,p}(x) = \frac{\hat{F}^{(\nu),\mathtt{BC}}_p(x) - F^{(\nu)}(x)}{\sqrt{\mathsf{Var}[\hat{F}^{(\nu),\mathtt{BC}}_p(x)]}} \rightsquigarrow \mathcal{N}(0,1), \qquad \hat{F}^{(\nu),\mathtt{BC}}_p(x) = \hat{F}^{(\nu)}_p(x) - \mathsf{Bias}[\hat{F}^{(\nu)}_p(x)],$$

which has a valid standard normal distribution even when an MSE, IMSE or a cross-validation-type bandwidth for $\hat{F}^{(\nu)}_p(x)$ is used. Confidence intervals with correct asymptotic coverage can be constructed by inverting the test statistic $\mathsf{T}^{\mathtt{RBC}}_{\nu,p}(x)$. In particular, it can be shown that a RBC confidence interval is equivalent to employing the test statistic $\mathsf{T}_{\nu,p+1}(x) = \mathsf{T}^{\mathtt{RBC}}_{\nu,p}(x)$ for a particular choice of parameters/implementation. Therefore, the function `lpdensity()` employs an RBC test statistic by default, assuming an (I)MSE-optimal or cross-validation-based bandwidth for the p-th order point estimator is used, denoted generically by h_p, and therefore forms the test statistic

$$\mathsf{T}^{\mathtt{RBC}}_{\nu,p}(x) \equiv \mathsf{T}_{\nu,p+1}(x;h_p) = \frac{\hat{F}^{(\nu)}_{p+1}(x;h_p) - F^{(\nu)}(x)}{\sqrt{\mathsf{Var}[\hat{F}^{(\nu)}_{p+1}(x;h_p)]}},$$

and associated confidence intervals

$$\mathsf{CI}^{\mathtt{RBC}}_{\nu,p}(x) \equiv \mathsf{CI}_{\nu,p+1}(x;h_p) = \left[\hat{F}^{(\nu)}_{p+1}(x;h_p) \quad \pm \quad z_{1-\alpha/2}\sqrt{\mathsf{Var}[\hat{F}^{(\nu)}_{p+1}(x;h_p)]}\right].$$

The notation makes the bandwidth explicit to distinguish the two polynomial degrees used in constructing the point estimator and the RBC confidence interval/test statistic: (i) a p-th order polynomial is used for point estimation (and bandwidth selection), and (ii) a $(p+1)$-th order polynomial is used for inference.

More generally, the package **lpdensity** implements confidence intervals of the form:

$$\mathsf{CI}^{\mathtt{RBC},q}_{\nu,p}(x) \equiv \mathsf{CI}_{\nu,q}(x;h_p) = \left[\hat{F}^{(\nu)}_{q}(x;h_p) \quad \pm \quad z_{1-\alpha/2}\sqrt{\mathsf{Var}[\hat{F}^{(\nu)}_{q}(x;h_p)]}\right],$$

with q determining the inference approach. The above confidence interval is thus based on inverting the statistic $\mathsf{T}^{\mathtt{RBC},q}_{\nu,p}(x) \equiv \mathsf{T}_{\nu,q}(x;h_p)$, and by default we set $q = p+1$. CJM formally showed that the RBC confidence intervals have asymptotically correct coverage:

$$\lim_{n\to\infty} \mathbb{P}\left[F^{(\nu)}(x) \in \mathsf{CI}^{\mathtt{RBC},q}_{\nu,p}(x)\right] = 1-\alpha, \qquad \forall x \in \mathcal{X}.$$

In addition to pointwise confidence intervals, the **lpdensity** package also offers uniform confidence bands for the CDF, PDF, or derivatives thereof. The uniform confidence band for $F^{(\nu)}(x)$ takes a similar form,

$$\mathsf{CB}^{\mathtt{RBC},q}_{\nu,p}(\mathcal{G}) \equiv \mathsf{CB}_{\nu,q}(\mathcal{G};h_p) = \left\{\left[\hat{F}^{(\nu)}_{q}(x;h_p) \quad \pm \quad z_{\mathcal{G},1-\alpha/2}\sqrt{\mathsf{Var}[\hat{F}^{(\nu)}_{q}(x;h_p)]}\right], \quad x \in \mathcal{G}\right\},$$

with two noticeable differences. First, the confidence band no longer depends on the evaluation point, but rather on a collection of evaluation points, $\mathcal{G}$. Second, the critical value also changes, which is now denoted by $z_{\mathcal{G},1-\alpha/2}$. In practice, the new critical value can be obtained by first simulating a suitable Brownian bridge on the grid $\mathcal{G}$, and then computing the upper α quantile of the supremum of the simulated process. The option `CIuniform = TRUE`

enables estimation and reporting of the uniform confidence band, which is turned off by default. CJM established a uniformly valid distributional approximation for the stochastic process $\{\mathsf{T}_{\nu,p}^{\mathtt{RBC},q}(x) : x \in \mathcal{G}\}$, and proved that a nominal $1-\alpha$ level RBC confidence band is asymptotically valid:

$$\lim_{n\to\infty} \mathbb{P}\left[F^{(\nu)}(x) \in \mathsf{CI}_{\nu,p}^{\mathtt{RBC},q}(\mathcal{G}),\ \forall x \in \mathcal{G}\right] = 1-\alpha.$$

See Cattaneo, Jansson, and Ma (2022) for technical details, regularity conditions, and additional discussions.

Robust bias correction methods lead to confidence intervals/bands that will not be centered at the density point estimates because of the recentering introduced by the bias correction. That is, different polynomial orders are used for constructing point estimates and confidence intervals/bands. Setting q and p to be equal delivers confidence intervals/bands that are centered at the point estimates, but requires undersmoothing for valid inference (i.e., an (I)MSE-optimal bandwdith cannot be used). Hence the bandwidth would need to be specified manually when $q = p$, and the point estimates will no longer be (I)MSE-optimal. Sometimes the point estimates may even lie outside of the confidence intervals/bands, which can happen if the underlying distribution exhibits high curvature at some evaluation point(s). One possible solution in this case is to increase the polynomial order p or to employ a smaller bandwidth.

2.4. Bandwidth selection

The package **lpdensity** implements several bandwidth selectors through `lpbwdensity()`, including MSE-optimal and IMSE-optimal plug-in rules, as well as rule-of-thumb bandwidth selectors based on a normal reference model. We only outline the main aspects of bandwidth selection here, but further details are given in Appendix A.

To introduce our bandwidth selectors, recall that the quantities $\mathsf{V}_{\nu,p}(x)$, $\mathsf{B}_{1,\nu,p}(x)$, and $\mathsf{B}_{2,\nu,p}(x)$ are given in pre-asymptotic form, and hence they can be computed from the data directly given a pilot/preliminary bandwidth. As a consequence, to construct the (I)MSE-optimal bandwidth, the only unknown quantities are $F^{(p+1)}(x)$ and $F^{(p+2)}(x)$, which can be consistently estimated using the local polynomial density derivative estimators implemented in `lpdensity()` with a pilot/preliminary bandwidth. To be more precise, the MSE-optimal bandwidth is estimated by

$$\hat{h}_{\mathtt{MSE},p}(x) = \operatorname*{arg\,min}_{h>0} \left\{ \mathsf{Var}[\hat{F}_p^{(\nu)}(x)] + \widehat{\mathsf{Bias}}[\hat{F}_p^{(\nu)}(x)]^2 \right\},$$

with $\widehat{\mathsf{Bias}}[\hat{F}_p^{(\nu)}(x)]$ constructed by replacing $F^{(p+1)}(x)$ and $F^{(p+2)}(x)$ with their estimated counterparts. Similarly, the IMSE-optimal bandwidth selector is given by

$$\hat{h}_{\mathtt{IMSE},p} = \operatorname*{arg\,min}_{h>0} \sum_{g_j\in\mathcal{G}} \left\{ \mathsf{Var}[\hat{F}_p^{(\nu)}(g_j)] + \widehat{\mathsf{Bias}}[\hat{F}_p^{(\nu)}(g_j)]^2 \right\},$$

where $\mathcal{G}$ is the collection of grid points specified in the function (by default, $\mathcal{G}$ takes on nineteen quantile-spaced values over the support of the data).

2.5. CDF estimation

While the estimator $\hat{F}_p^{(\nu)}(x)$ is valid for all $\nu \geq 0$, our discussion so far focused on the case $\nu \geq 1$ because the resulting estimators of the density ($\nu = 1$) and its derivatives ($\nu \geq 2$)

are the main focus of the package. Nevertheless, as a by-product, CJM developed analogous estimation, bandwidth selection and RBC inference results for the smooth CDF estimator $\hat{F}_p(x) = \hat{F}_p^{(0)}(x)$. These results are also implemented in the package **lpdensity** via the option `v = 0`. For example, CDF estimation using a local constant approximation is obtained using `lpdensity(..., p = 0, v = 0)`, which employs the corresponding MSE-optimal bandwidth (`bwselect = "mse-dpi"`) and a local linear approximation for inference (`q = p + 1`) by default.

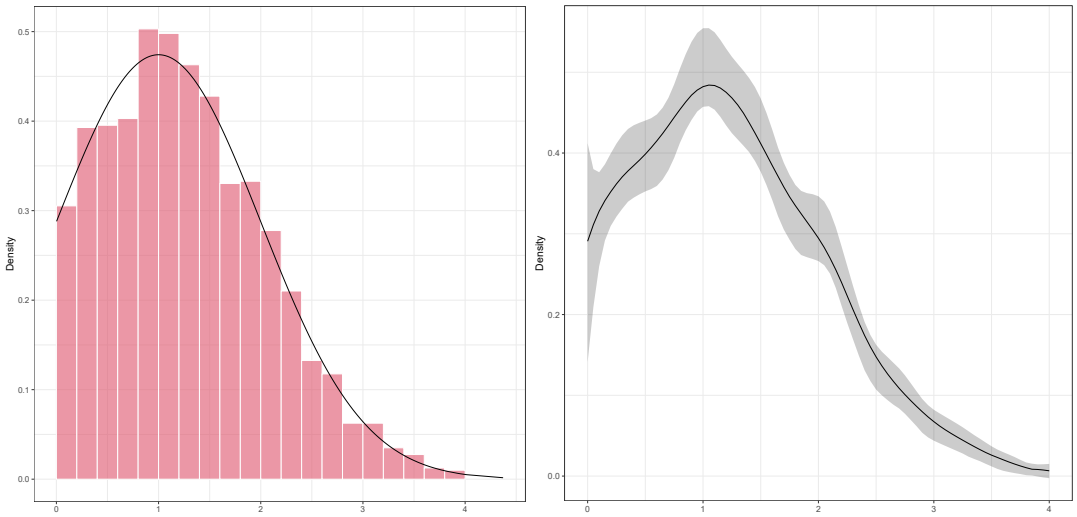
3. Implementation and numerical illustration

We showcase some of the main features of the **lpdensity** package. The data consists of 2 000 observations simulated from the normal distribution $\mathcal{N}(1, 1)$ truncated from below at 0. We create a discontinuity in density at $x = 0$ to illustrate the performance of our procedure at boundaries. Panel (a) of Figure 1 plots a histogram estimate and the true density function.

```
R> set.seed(42)
R> data <- rnorm(4000, mean = -1)
R> data <- data[data < 0]
R> data <- -1 * data[1:2000]
```

3.1. Function `lpdensity()`

The function `lpdensity()` provides both point estimates as well as RBC inference (confidence intervals and bands) employing the local polynomial density estimator, given a grid of points and a bandwidth choice. If the latter are not provided, then by default the function chooses



(a) Histogram. (b) Density plot.

Figure 1: Histogram of the simulated data and the density plot.

nineteen quantile-spaced grid points over the support of the data and computes $\hat{h}_{\texttt{MSE},p}(x)$ at each point.

The following command estimates the density function (`v = 1`, the default) with fixed bandwidth `bw = 0.5` at points $0, 0.5, \ldots, 4$, using a local quadratic approximation (`p = 2`, the default) to the empirical distribution function. RBC confidence intervals over the grid are also computed, in this case using a local cubic approximation (`q = 3`, the default).

```
R> model1 <- lpdensity(data, bw = 0.5, grid = seq(0, 4, 0.5))
R> summary(model1)

Call: lpdensity

Sample size                                  (n=)    2000
Polynomial order for point estimation        (p=)    2
Density function estimated                   (v=)    1
Polynomial order for confidence interval     (q=)    3
Kernel function                                      triangular
Bandwidth selection method                           user provided

================================================================================
                                        Point      Std.       Robust B.C.
Index     Grid      B.W.    Eff.n       Est.      Error      [ 95% C.I. ]
================================================================================
1       0.0000    0.5000      355     0.2908    0.0436     0.1413 ,  0.4121
2       0.5000    0.5000      799     0.3986    0.0147     0.3525 ,  0.4402
3       1.0000    0.5000      919     0.4822    0.0160     0.4572 ,  0.5545
4       1.5000    0.5000      820     0.4116    0.0150     0.3767 ,  0.4675
5       2.0000    0.5000      564     0.2946    0.0137     0.2662 ,  0.3465
--------------------------------------------------------------------------------
6       2.5000    0.5000      320     0.1475    0.0099     0.1071 ,  0.1626
7       3.0000    0.5000      147     0.0674    0.0069     0.0438 ,  0.0821
8       3.5000    0.5000       59     0.0259    0.0045     0.0120 ,  0.0369
9       4.0000    0.5000       15     0.0065    0.0022    -0.0027 ,  0.0151
================================================================================
```

The first part of the output provides basic information on the options specified in the function. For example, the default estimand is the density function, indicated by `Density function estimated (v=) 1`. The rest of the output gives estimation results, including (i) `Grid`: the grid points; (ii) `B.W.`: the bandwidths; (iii) `Eff.n`: the effective sample size for each grid point; (iv) `Point Est.`: the point estimates using polynomial order `p`, and the associated standard errors under `Std. Error`; (v) `Robust B.C.[95% C.I.]`: robust bias-corrected 95% confidence intervals. Point estimates, standard errors, and other information can be easily extracted for further statistical analysis. The output is stored in a standard matrix, and can be accessed with the following:

```
R> model1$Estimate
```

When the argument `grid` is suppressed, the evaluation points will be the $0.05, 0.1, \ldots, 0.9, 0.95$ quantiles computed from the data. Conventional inference results (i.e., without robust bias correction) can be obtained by setting `q = p`. For example (output is suppressed):

```
R> summary(lpdensity(data, bw = 0.5, p = 2, q = 2))
```

It is also possible to suppress the argument `bw`, and the function will select the bandwidth automatically by minimizing (an estimated approximation to) the mean squared error, employing `lpbwdensity()`. Other bandwidth selection methods are available; we will illustrate data-driven bandwidth selection procedures in an upcoming subsection.

The method `summary()` takes six additional arguments. The first one, `alpha`, specifies the (one minus) nominal coverage of the confidence interval, with default being 0.05. Another argument is `sep`, which controls the horizontal separator. The default value is 5, and hence a dashed line is drawn after every five grid points. This feature can be suppressed by setting it to 0. Sometimes it may be desirable to report only a subset of the estimates, which can be done by using either the `grid` or the `gridIndex` option. The `grid` option allows reporting results for a selected set of grid points originally specified in the `lpdensity()` function, while `gridIndex` helps achieve the same goal by specifying the indices of the grid points. The last two options are related to confidence bands. By setting `CIuniform = TRUE`, a uniform confidence band, instead of pointwise confidence intervals, will be reported. Because the critical values have to be simulated in this case, the number of simulations used is controlled by the option `CIsimul` (its default value is 2 000). The following example produces the 99% confidence band for four grid points 0, 0.5, 1 and 2, with dashed lines appearing after every three grid points. (Fixing the random seed allows reproducing the simulated critical values and the confidence intervals.)

```
R> set.seed(123)
R> summary(model1, alpha = 0.01, sep = 3, grid = c(0, 0.5, 1, 2),
+    CIuniform = TRUE)

Call: lpdensity

Sample size                                (n=)    2000
Polynomial order for point estimation      (p=)    2
Density function estimated                 (v=)    1
Polynomial order for confidence interval   (q=)    3
Kernel function                                    triangular
Bandwidth selection method                         user provided

================================================================================
                                      Point     Std.        Robust B.C.
Index     Grid      B.W.    Eff.n      Est.     Error    [ Unif. 99% C.I. ]
================================================================================
1       0.0000    0.5000      355    0.2908    0.0436     0.0606 ,  0.4927
2       0.5000    0.5000      799    0.3986    0.0147     0.3263 ,  0.4664
3       1.0000    0.5000      919    0.4822    0.0160     0.4283 ,  0.5835
--------------------------------------------------------------------------------
5       2.0000    0.5000      564    0.2946    0.0137     0.2423 ,  0.3704
================================================================================
```

Another important argument in `lpdensity()` is `scale`, which scales the point estimates and standard errors. This is particularly useful if only part of the data is used. For example, assume one would like to estimate the PDF using the two subsamples $\{X_i : X_i < 1.5\}$ and $\{X_i : X_i > 1.5\}$ separately. Simply splitting the data will not give consistent estimates, as it produces conditional (rather than marginal) density estimates:

```
R> lpdensity(data[data < 1.5], bw = 0.5, grid = 1.5)$Estimate[, "f_p"]
R> lpdensity(data[data > 1.5], bw = 0.5, grid = 1.5)$Estimate[, "f_p"]
R> dnorm(1.5, mean = 1, sd = 1) / pnorm(0, mean = 1, sd = 1,
+    lower.tail = FALSE)

[1] 0.6755464
[1] 1.222052
[1] 0.4184555
```

The previous commands give point estimates 0.676 and 1.222, which are far from the true value 0.418. To have consistent estimates, we need to scale the estimates by the proportion of the data used for estimation:

```
R> lpdensity(data[data < 1.5], bw = 0.5, grid = 1.5,
+    scale = sum(data < 1.5)/2000)$Estimate[, "f_p"]
R> lpdensity(data[data > 1.5], bw = 0.5, grid = 1.5,
+    scale = sum(data > 1.5)/2000)$Estimate[, "f_p"]

[1] 0.4303231
[1] 0.443605
```

3.2. Function `plot()`

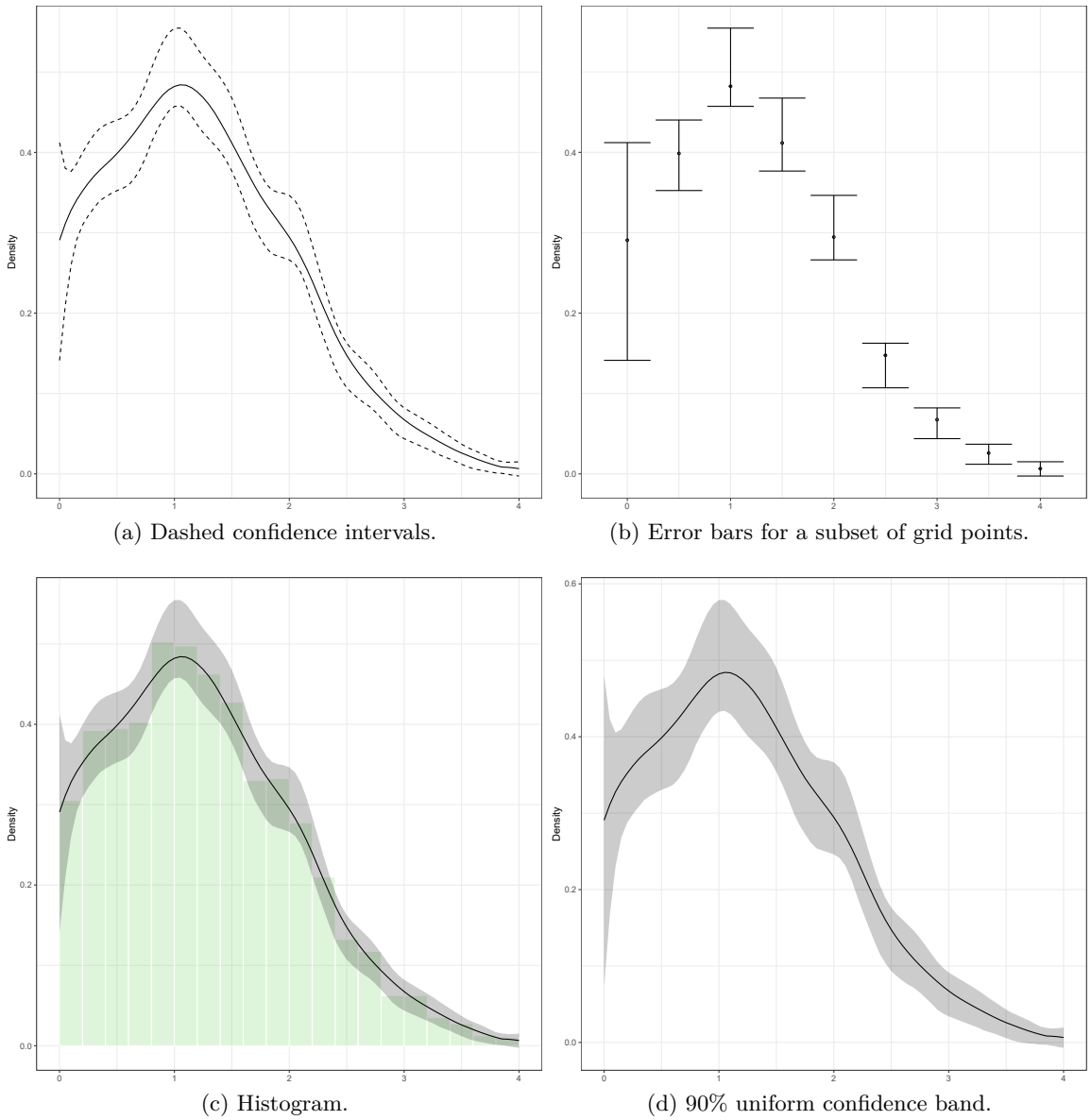
The function `plot()`, along with many other methods, is supported. This function takes the output from `lpdensity()` and produces plots of point estimates and robust bias-corrected confidence intervals/bands over the grid of evaluation points selected. Panel (b) of Figure 1 shows how plots can be easily generated.

```
R> model2 <- lpdensity(data, bw = 0.5, grid = seq(0, 4, 0.05))
R> plot(model2) + theme(legend.position = "none")
```

The confidence intervals/bands are not centered at the point estimates in general. As described in Section 2, by default the point estimates are constructed using MSE-optimal bandwidths, which implies the smoothing bias is non-negligible and hence valid inference should be based on robust bias-corrected confidence intervals.

The function `plot()` allows for customization: Figure 2 illustrates some of the features. For Panel (d), we again fix the random seed to reproduce the simulated critical values and the confidence band.

```
R> plot(model2, CItype = "line") + theme(legend.position = "none")
R> plot(model2, type="points", CItype="ebar", grid = seq(0, 4, 0.5)) +
+    theme(legend.position = "none")
```



(a) Dashed confidence intervals.

(b) Error bars for a subset of grid points.

(c) Histogram.

(d) 90% uniform confidence band.

Figure 2: Density plots with different specifications.

```
R> plot(model2, hist = TRUE, histData = data, histBreaks = seq(0, 4, 0.2)) +
+    theme(legend.position = "none")
R> set.seed(123)
R> lpdensity.plot(model2, alpha = 0.1, CIuniform = TRUE) +
+    theme(legend.position = "none")
```

3.3. Function `lpbwdensity()`

The function `lpbwdensity()` implements four bandwidth selectors, (i) MSE-optimal plug-in bandwidth selector, denoted by `"mse-dpi"` (this is the default option), (ii) IMSE-optimal

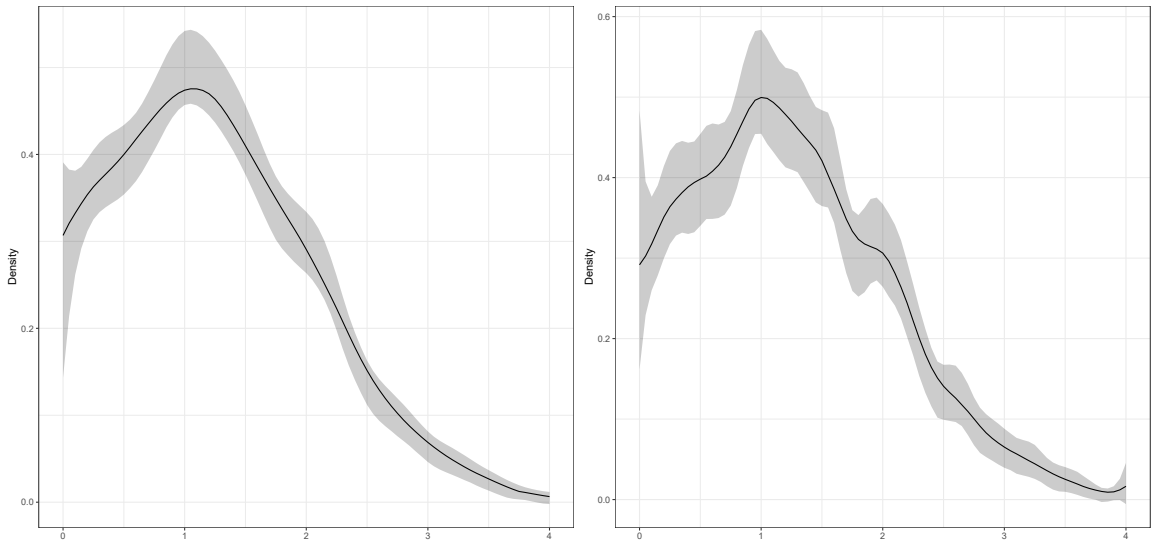
plug-in bandwidth selector, denoted by `"imse-dpi"`, (iii) rule-of-thumb bandwidth selector with a normal reference model, denoted by `"mse-rot"`, and (iv) integrated rule-of-thumb bandwidth selector, denoted by `"imse-rot"`. We illustrate some of the main features of `lpbwdensity()` with the same simulated data used previously.

By default, `lpbwdensity()` computes the MSE-optimal bandwidth for estimating the PDF with a local quadratic regression and triangular kernel, on 19 quantile-spaced grid points: `lpbwdensity(..., p = 2, v = 1, bwselect = "mse-dpi", kernel = "triangular")`. The output resembles that of `lpdensity()`, and provides basic information for the data and options specified, as well as a matrix with three columns: (i) `Grid` for grid of evaluation points, (ii) `B.W.` for estimated bandwidths, and (iii) `Eff.n` for effective sample size at each grid point given the estimated bandwidth. The following is an example with a user-chosen grid of evaluation points.

```
R> model1bw <- lpbwdensity(data, grid = seq(0, 4, 0.5))
R> summary(model1bw)

Call: lpbwdensity

Sample size                              (n=)    2000
Polynomial order for point estimation    (p=)    2
Density function estimated               (v=)    1
Kernel function                                  triangular
Bandwidth selection method                       mse-dpi

================================
Index     Grid      B.W.   Eff.n
================================
1       0.0000    0.4064     287
2       0.5000    0.6266     933
3       1.0000    0.4721     872
4       1.5000    0.6474    1048
5       2.0000    1.0662    1216
--------------------------------
6       2.5000    0.5835     385
7       3.0000    0.5991     175
8       3.5000    0.6458      80
9       4.0000    0.6170      22
================================
```

The estimated bandwidths from this function can be used as input for `lpdensity()`, but constructing bandwidths in a separate step is redundant: bandwidth selection can be specified directly through the option `bwselect` in `lpdensity()`. For example, the following first computes the IMSE-optimal bandwidth and then estimates the density function:

```
R> model5 <- lpdensity(data, grid = seq(0, 4, 0.5), bwselect="imse-dpi")
R> summary(model5)
```



(a) IMSE-bandwidth (b) Under-smoothing

Figure 3: Density plot with IMSE-optimal bandwidth and under-smoothing.

It may be helpful to estimate bandwidths in a separate, first step so that they can be modified prior to estimation and inference (e.g., to implement *ad hoc* undersmoothing). To show this procedure, we reproduce Panel (b) of Figure 1 with the estimated IMSE-optimal bandwidth as well as *ad hoc* under-smoothing (where the IMSE-optimal bandwidth is divided by 2). See the following code and Figure 3.

```
R> model6bwIMSE <- lpbwdensity(data, grid = seq(0, 4, 0.05),
+    bwselect = "imse-dpi")
R> model6 <- lpdensity(data, grid = seq(0, 4, 0.05),
+    bw = model6bwIMSE$BW[, "bw"])
R> plot(model6) + theme(legend.position = "none")
R> model7 <- lpdensity(data, grid = seq(0, 4, 0.05),
+    bw = model6bwIMSE$BW[, "bw"] / 2)
R> plot(model7) + theme(legend.position = "none")
```

To prevent the estimated bandwidth from being too small, the default implementation in the `lpbwdensity()` function requires the local neighborhood around the evaluation point to contain at least $20 + p + 1$ (unique) observations. If the resulting neighborhood is not large enough, then the bandwidth is enlarged until the minimum number of observations is met. The default values can be changed through the options `nLocalMin`, controlling the minimum number of observations in each local neighborhood, and `nUniqueMin`, controlling the minimum number of unique observations in each local neighborhood. This minimal local sample size checking feature can be turned off by setting `regularize=FALSE`. Finally, the package **lpdensity** also includes checks and adjustments for repeated observations of the variable X in the data. This feature can be turned off by setting `massPoints=FALSE`.

4. Simulation evidence and comparison with other R packages

We illustrate the finite-sample performance of our **lpdensity** package in a simulation study, and compare it with other R packages implementing kernel-based density estimation procedures. The functions/packages we consider are: `bkde()` and `locpoly()` in the **KernSmooth** package (Wand and Ripley 2021), `kdde()` and `kde()` in the **ks** package (Duong 2007, 2021), `npudens()` and `npuniden.boundary()` in the **np** package (Hayfield and Racine 2008; Racine and Hayfield 2021), `kdrobust()` in the **nprobust** package (Calonico, Cattaneo, and Farrell 2019, 2020), `plugin.density()` in the **plugdensity** package (Herrmann and Mächler 2011), as well as the built-in density estimator `stats::density()`.

Table 1 provides a brief summary of their main features. First, three packages offer valid density estimates at (or near) boundaries, including **KernSmooth**, **np** and our **lpdensity**. However, only **KernSmooth** and **lpdensity** provide automatic boundary carpentry, while **np** requires specifying boundary kernels. Second, only two packages, **KernSmooth** and **lpdensity**, support higher-order bias reduction. Third, statistical inference is available in **np**, **nprobust**, and **lpdensity**. However, among these three packages, only **nprobust** and **lpdensity** account for the possibly leading smoothing bias when constructing test statistics/confidence intervals using (I)MSE-optimal bandwidths, and **nprobust** is not valid at or near boundary points. In addition, our **lpdensity** package is the only one that supports constructing uniform confidence bands. Fourth, only three packages, **KernSmooth**, **ks** and **lpdensity**, offer density derivative estimation. In summary, the **lpdensity** package provides valid density and derivatives estimation for both interior and boundary evaluation points, allows higher-order bias reduction through the use of higher-order local polynomial approximations, and offers several (I)MSE-optimal bandwidth selection methods. For statistical inference, **lpdensity** takes into account the possibly leading smoothing bias, and hence delivers (asymptotically) valid testings and confidence intervals/bands, both pointwise and uniformly over evaluation points.

We now describe our simulation design. The data consists of a random sample of size $n = 1\,000$, generated either from the normal distribution $\mathcal{N}(1, 1)$ truncated below at 0 (column "Truncated Normal"), or the exponential distribution with a scale parameter of 1 (column "Exponential"). We consider the estimation of the PDF at three evaluation points: $x = 1.5$, $x = 0.2$ and $x = 0$, corresponding to interior, near boundary and boundary regions, respectively. We employ 2 000 Monte Carlo repetitions. Simulation results are reported in Table 2. For the point estimate, we report its bias (column "Bias"), standard deviation (column "SD") and root mean squared error (column "RMSE"). Whenever available, we also report the empirical coverage probability of a nominal 95% confidence interval (column "EC") as well as its average length (column "IL").

At the interior evaluation point, all procedures perform similarly in terms of RMSE and empirical coverage. Point estimates obtained using `lpdensity()` have relatively small RMSEs, and the corresponding RBC confidence intervals exhibit satisfactory coverage properties. When the evaluation point is close to or exactly at the boundary, most packages or functions are no longer valid, and hence we only report simulation results for `locpoly()`, `npuniden.boundary()`, and `lpdensity()`. In such cases, `lpdensity()` delivers points estimates and confidence intervals with excellent finite-sample performance.

	Truncated Normal						Exponential					
	h	Bias	SD	RMSE	EC	IL	h	Bias	SD	RMSE	EC	IL
Interior ($x = 1.5$)												
`bkde`		0.008	0.019	0.020				0.009	0.013	0.016		
`locpoly`	0.172	0.004	0.023	0.023			0.100	0.001	0.024	0.024		
`kdde`	0.172	0.004	0.023	0.023			0.100	0.001	0.024	0.024		
`kde`	0.172	0.004	0.023	0.023			0.100	0.001	0.024	0.024		
`npudens`	0.102	0.000	0.035	0.035	0.964	0.140	0.143	0.003	0.023	0.023	0.949	0.089
`npuniden.boundary`	0.231	0.008	0.023	0.024	0.948	0.091	0.147	0.003	0.021	0.021	0.962	0.084
`kdrobust`	0.609	0.011	0.016	0.019	0.936	0.081	0.633	0.009	0.013	0.016	0.941	0.063
`plugin.density`	0.144	0.003	0.026	0.026			0.071	0.000	0.029	0.029		
`density`	0.179	0.004	0.022	0.023			0.185	0.004	0.017	0.018		
`lpdensity`($h_{\texttt{MSE}}$)	0.785	0.008	0.021	0.022	0.957	0.102	0.680	0.006	0.015	0.017	0.949	0.083
`lpdensity`($h_{\texttt{IMSE}}$)	0.623	0.007	0.019	0.020	0.947	0.112	0.687	0.007	0.014	0.016	0.948	0.083
Near boundary ($x = 0.2$)												
`locpoly`	0.172	0.033	0.024	0.041			0.100	0.020	0.045	0.049		
`npuniden.boundary`	0.230	0.019	0.026	0.032	0.877	0.099	0.147	0.018	0.043	0.046	0.931	0.172
`lpdensity`($h_{\texttt{MSE}}$)	1.149	0.022	0.044	0.049	0.948	0.118	0.903	0.009	0.045	0.046	0.938	0.153
`lpdensity`($h_{\texttt{IMSE}}$)	0.621	0.001	0.030	0.030	0.950	0.117	0.687	0.001	0.040	0.040	0.944	0.156
Boundary ($x = 0$)												
`locpoly`	0.172	0.139	0.016	0.140			0.100	0.548	0.034	0.549		
`npuniden.boundary`	0.230	0.054	0.033	0.063	0.506	0.117	0.147	0.091	0.083	0.124	0.548	0.242
`lpdensity`($h_{\texttt{MSE}}$)	0.686	0.010	0.058	0.059	0.944	0.348	0.807	0.045	0.087	0.098	0.932	0.511
`lpdensity`($h_{\texttt{IMSE}}$)	0.621	0.007	0.055	0.055	0.955	0.343	0.687	0.026	0.082	0.086	0.952	0.514

Table 2: Simulation results. Empty cells correspond to features that are not readily available without modifying the source code. For the case of "Near boundary" and "Boundary" we only consider software packages/functions that are valid for those cases. Default options for each package/function are used whenever possible. Results are based on 2 000 simulations with a sample size of 1 000. Column "Truncated Normal": The $\mathcal{N}(1, 1)$ distribution truncated from below at 0. Column "Exponential": The exponential distribution with a scale parameter 1.

5. Conclusion

We gave an introduction to the general purpose software package **lpdensity**, which offers local polynomial regression based estimation and inference procedures for a cumulative distribution function, probability density function, and higher-order derivatives thereof. This package is available in both R and Stata statistical platforms, and further details can be found at https://nppackages.github.io/lpdensity/.

Acknowledgments

We thank Sebastian Calonico, David Drukker, Yingjie Feng, the editor, and two anonymous reviewers for thoughtful comments on our software implementation and article. We are also grateful to many users who provided valuable feedback. Cattaneo gratefully acknowledges financial support from the National Science Foundation (SES-1459931 and SES-1947805). Jansson gratefully acknowledges financial support from the National Science Foundation (SES-1459967 and SES-1947662) and the research support of CREATES.

References

Calonico S, Cattaneo MD, Farrell MH (2018). "On the Effect of Bias Estimation on Coverage Accuracy in Nonparametric Inference." *Journal of the American Statistical Association*, **113**(522), 767–779. doi:10.1080/01621459.2017.1285776.

Calonico S, Cattaneo MD, Farrell MH (2019). "**nprobust**: Nonparametric Kernel-Based Estimation and Robust Bias-Corrected Inference." *Journal of Statistical Software*, **91**(8), 1–33. doi:10.18637/jss.v091.i08.

Calonico S, Cattaneo MD, Farrell MH (2020). ***nprobust**: Nonparametric Robust Estimation and Inference Methods Using Local Polynomial Regression and Kernel Density Estimation*. R package version 0.4.0, URL https://CRAN.R-project.org/package=nprobust.

Calonico S, Cattaneo MD, Farrell MH (2022). "Coverage Error Optimal Confidence Intervals for Local Polynomial Regression." *Bernoulli*. Forthcoming.

Cattaneo MD, Jansson M, Ma X (2020). "Simple Local Polynomial Density Estimators." *Journal of the American Statistical Association*, **115**(531), 1449–1455. doi:10.1080/01621459.2019.1635480.

Cattaneo MD, Jansson M, Ma X (2022). "Local Regression Distribution Estimators." *Journal of Econometrics*. doi:10.1016/j.jeconom.2021.01.006. Forthcoming.

Cheng MY, Fan J, Marron JS (1997). "On Automatic Boundary Corrections." *The Annals of Statistics*, **25**(4), 1691–1708. doi:10.1214/aos/1031594737.

Duong T (2007). "**ks**: Kernel Density Estimation and Kernel Discriminant Analysis for Multivariate Data in R." *Journal of Statistical Software*, **21**(7), 1–16. doi:10.18637/jss.v021.i07.

Duong T (2021). ***ks**: Kernel Smoothing*. R package version 1.13.2, URL https://CRAN.R-project.org/package=ks.

Fan J, Gijbels I (1996). *Local Polynomial Modelling and Its Applications*. Chapman & Hall/CRC, New York.

Hayfield T, Racine JS (2008). "Nonparametric Econometrics: The **np** Package." *Journal of Statistical Software*, **27**(5), 1–32. doi:10.18637/jss.v027.i05.

Herrmann E, Mächler M (2011). ***plugdensity**: Plug-in Kernel Density Estimation*. R package version 0.8-3, URL https://CRAN.R-project.org/package=plugdensity.

Karunamuni RJ, Albert T (2005). "On Boundary Correction in Kernel Density Estimation." *Statistical Methodology*, **2**, 191–212. doi:10.1016/j.stamet.2005.04.001.

Racine JS, Hayfield T (2021). ***np**: Nonparametric Kernel Smoothing Methods for Mixed Data Types*. R package version 0.60-11, URL https://CRAN.R-project.org/package=np.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

StataCorp (2019). *Stata Statistical Software: Release 16*. StataCorp LLC, College Station. URL http://www.stata.com/.

Wand MP, Jones MC (1995). *Kernel Smoothing*. Chapman & Hall/CRC, New York.

Wand MP, Ripley BD (2021). **KernSmooth**: *Functions for Kernel Smoothing Supporting Wand & Jones (1995)*. R package version 2.23-20, URL https://CRAN.R-project.org/package=KernSmooth.

Wickham H (2016). **ggplot2**: *Elegant Graphics for Data Analysis*. Springer-Verlag, New York. URL https://ggplot2.tidyverse.org/.

Zhang S, Karunamuni RJ (1998). "On Kernel Density Estimation Near Endpoints." *Journal of Statistical Planning and Inference*, **70**(1), 301–316. doi:10.1016/s0378-3758(97)00187-0.

A. Details on bandwidth selection

We provide more methodological details on bandwidth selectors implemented thorough the function `lpbwdensity()`. We continue to focus on the case of $1 \leq \nu \leq p$, and therefore do not discuss bandwidth selection for CDF estimation. See CJM for details.

A.1. Rule-of-thumb bandwidths

Recall that, in the definition of $\mathsf{Var}[\hat{F}_p^{(\nu)}(x)]$ and $\mathsf{Bias}[\hat{F}_p^{(\nu)}(x)]$, we introduced pre-asymptotic quantities $\mathsf{V}_{\nu,p}(x)$, $\mathsf{B}_{1,\nu,p}(x)$ and $\mathsf{B}_{2,\nu,p}(x)$. For the rule-of-thumb bandwidth selectors, we consider a normal reference model, hence all evaluation points are interior. Then, those quantities have well-defined limits, which can be computed using features of the underlying distribution (such as normal densities and higher-order derivatives), p, ν, and the kernel function. We denote the rule-of-thumb bandwidth by $\hat{h}_{\texttt{ROT},p}$. An integrated version can be constructed accordingly, and is denoted by $\hat{h}_{\texttt{IROT},p}$.

Given x, p and ν, the rate at which the MSE-optimal bandwidth $h_{\texttt{MSE}}$ shrinks to zero depends on whether $p-\nu$ is even or odd, and whether x is interior or boundary. This is summarized in Panel (a) of Table 3. We also include the rate at which the rule-of-thumb bandwidths shrinks in Panel (b). (The notation $\hat{h} \asymp_{\mathbb{P}} n^{-1/\gamma}$ indicates that both $n^{1/\gamma}\hat{h}$ and $n^{-1/\gamma}\hat{h}^{-1}$ are bounded in probability.) Note that the (I)ROT-optimal bandwidths have the correct rate of convergence, except when $p-\nu$ is even and x is near boundary.

A.2. (I)MSE-optimal bandwidths

We now discuss some implementation details of the MSE-optimal bandwidth, which will also apply to the construction of the IMSE-optimal bandwidth. First, the unknown higher-order derivatives $F^{(p+1)}(x)$ and $F^{(p+2)}(x)$ are replaced by consistent estimates, $\hat{F}_{p+2}^{(p+1)}(x;\hat{h}_{\texttt{IROT},p+1,p+2})$ and $\hat{F}_{p+3}^{(p+2)}(x;\hat{h}_{\texttt{IROT},p+2,p+3})$, respectively. Here we augment the subscript of bandwidths with one additional argument, since the bandwidth depends on both the polynomial order as well as the order of derivative. For example, $\hat{h}_{\texttt{IROT},p+1,p+2}$ is an estimated bandwidth using a normal reference model, which is IMSE-optimal for a local polynomial regression of order $p+2$ when estimating the $(p+1)$-th derivative of $F(x)$.

The next step is to construct the pre-asymptotic quantities $\mathsf{V}_{\nu,p}(x)$, $\mathsf{B}_{1,\nu,p}(x)$ and $\mathsf{B}_{2,\nu,p}(x)$, which require a preliminary bandwidth. We use $\hat{h}_{\texttt{IROT},1,2}$, so those quantities are $\mathsf{V}_{\nu,p}(x;\hat{h}_{\texttt{IROT},1,2})$, $\mathsf{B}_{1,\nu,p}(x;\hat{h}_{\texttt{IROT},1,2})$ and $\mathsf{B}_{2,\nu,p}(x;\hat{h}_{\texttt{IROT},1,2})$. Then, the MSE-optimal bandwidth is

$$\hat{h}_{\texttt{MSE},p} = \operatorname*{arg\,min}_{h>0} \left\{ \widehat{\mathsf{Var}}[\hat{F}_p^{(\nu)}(x)] + \widehat{\mathsf{Bias}}[\hat{F}_p^{(\nu)}(x)]^2 \right\},$$

with $\widehat{\mathsf{Var}}[\hat{F}_p^{(\nu)}(x)] = \frac{1}{nh^{2\nu-1}}\mathsf{V}_{\nu,p}(x;\hat{h}_{\texttt{IROT},1,2})$ and

$$\widehat{\mathsf{Bias}}[\hat{F}_p^{(\nu)}(x)] = h^{p-\nu+1}\Big[\hat{F}_{p+2}^{(p+1)}(x;\hat{h}_{\texttt{IROT},p+1,p+2})\mathsf{B}_{1,\nu,p}(x;\hat{h}_{\texttt{IROT},1,2}) + h\cdot\hat{F}_{p+3}^{(p+2)}(x;\hat{h}_{\texttt{IROT},p+2,p+3})\mathsf{B}_{2,\nu,p}(x;\hat{h}_{\texttt{IROT},1,2})\Big].$$

Under regularity conditions, it can be shown that $\hat{h}_{\texttt{MSE},p}$ is rate consistent (see Panel (c) of Table 3). Under the assumption that either (i) x is near boundary, or (ii) $p-\nu$ is odd, it

	x interior	x boundary
$p-\nu$ odd	$\gamma = 2p+1$	$\gamma = 2p+1$
$p-\nu$ even	$\gamma = 2p+3$	$\gamma = 2p+1$

(a) $h_{\mathtt{MSE},p} \asymp n^{-1/\gamma}$

	x interior	x boundary
$p-\nu$ odd	$\gamma = 2p+1$	$\gamma = 2p+1$
$p-\nu$ even	$\gamma = 2p+3$	$\gamma = 2p+3$

(b) $\hat{h}_{\mathtt{ROT},p} \asymp_{\mathbb{P}} n^{-1/\gamma}$ and $\hat{h}_{\mathtt{IROT},p} \asymp_{\mathbb{P}} n^{-1/\gamma}$

	x interior	x boundary
$p-\nu$ odd	$\gamma = 2p+1$	$\gamma = 2p+1$
$p-\nu$ even	$\gamma = 2p+3$	$\gamma = 2p+1$

(c) $\hat{h}_{\mathtt{MSE},p} \asymp_{\mathbb{P}} n^{-1/\gamma}$

Table 3: Bandwidths rates for $1 \leq \nu \leq p$.

is possible to show a stronger result: $\hat{h}_{\mathtt{MSE},p}/h_{\mathtt{MSE},p} \xrightarrow{\mathbb{P}} 1$, so that the MSE-optimal bandwidth selector is consistent both in *rate* and *constant*. What happens for interior x with $p-\nu$ even? In this case $\mathsf{B}_{1,\nu,p}(x;\hat{h}_{\mathtt{IROT},1,2}) \xrightarrow{\mathbb{P}} 0$, and $\mathsf{B}_{2,\nu,p}(x;\hat{h}_{\mathtt{IROT},1,2})$ captures only part of the leading bias. As a result, $\hat{h}_{\mathtt{MSE},p}$ has the correct rate of convergence, but is not consistent for $h_{\mathtt{MSE},p}$ in the strong sense.

B. Stata Implementation

We discuss the Stata implementation of our **lpdensity** package, which offers two commands, `lpdensity` for estimation of and inference on the CDF, PDF, and their higher-order derivatives, and `lpbwdensity` for data-driven bandwidth selection. The plotting features employ the built-in command `twoway`.

The command `lpdensity` provides point estimation and robust confidence intervals/bands employing the local polynomial density estimator, given a grid of points and a bandwidth choice. We generate 2 000 observations from the normal distribution $\mathcal{N}(1,1)$ truncated below at 0. Although the same seed, 42, as in R is used, observations generated in Stata are generally different due to the different random number generators used by the statistical platforms.

The following command estimates the density function (`v(1)`, the default) with fixed bandwidth `bw(0.5)` over the grid of evaluation points $0, 0.5, \ldots, 4$, using a local quadratic approximation (`p(2)`, the default) to the empirical distribution function. Robust bias-corrected confidence intervals over the grid are computed using a local cubic approximation (`q(3)`, the default).

```
. set seed 42
. set obs 4000
. gen data = rnormal(1, 1)
. drop if data <= 0
. drop if _n > 2000
. gen grid = -0.5 + 0.5 * _n if _n <= 9
. lpdensity data, grid(grid) bw(0.5)

Local Polynomial Density Estimation and Inference.
```

```
Sample size                                (n=)              2000
Polynomial order for point estimation      (p=)                 2
Density function estimated                 (v=)                 1
Polynomial order for confidence interval   (q=)                 3
Kernel function                                          triangular
Bandwidth selection method                                 mse-dpi

------------------------------------------------------------------------------
                                        Point      Std.          Robust B.C.
 Index     Grid      B.W.    Eff.n      Est.      Error             95% C.I.
------------------------------------------------------------------------------
   1     0.0000    0.5000      366    0.2815    0.0403    0.1064    0.3547
   2     0.5000    0.5000      814    0.4230    0.0153    0.3899    0.4829
   3     1.0000    0.5000      897    0.4680    0.0158    0.4455    0.5414
   4     1.5000    0.5000      834    0.4056    0.0146    0.3594    0.4486
   5     2.0000    0.5000      607    0.3209    0.0143    0.2981    0.3810
------------------------------------------------------------------------------
   6     2.5000    0.5000      307    0.1406    0.0100    0.1065    0.1632
   7     3.0000    0.5000      117    0.0516    0.0061    0.0303    0.0642
   8     3.5000    0.5000       43    0.0199    0.0038    0.0070    0.0292
   9     4.0000    0.5000       12    0.0130    0.0077   -0.0006    0.0432
------------------------------------------------------------------------------
```

Coverage of the robust confidence interval can be specified through `level()`. For example, to report nominal 99% confidence intervals, one can use

```
. lpdensity data, grid(grid) bw(0.5) level(99)
```

When the argument `grid()` is suppressed, the evaluation points will be the $0.05, 0.1, \ldots, 0.9, 0.95$ quantiles computed from the data. Conventional inference results (i.e., without robust bias correction) can be obtained by setting `q()` to be the same as `p()`.

```
. lpdensity data, bw(0.5)
. lpdensity data, bw(0.5) q(2)
```

In Stata, graphical illustration of the estimates can be obtained using the option `plot`. The following plots the estimated density function on a fine grid, which resembles Panel (b) of Figure 1.

```
. capture drop grid
. gen grid = -0.05 + 0.05 * _n if _n <= 81
. lpdensity data, grid(grid) bw(0.5) plot
```

The same figure can be produced by first storing estimation results and then calling the `twoway` command directly. For example,

```
. lpdensity data, grid(grid) bw(0.5) genvars(lpdTemp)
. twoway                                                                ///
```

```
>   (rarea lpdTemp_CI_l lpdTemp_CI_r lpdTemp_grid, sort color(red%30)) ///
>   (line lpdTemp_f_p lpdTemp_grid,                                    ///
>     sort lcolor(red) lwidth("medthin") lpattern(solid)),             ///
>   legend(off) title("lpdensity (p=2, q=3, v=1)", color(gs0))        ///
>   xtitle("data") ytitle("")
. drop lpdTemp_*
```

To further illustrate, the following generates analogues of Panel (c) and (d) of Figure 2.

```
. lpdensity data, grid(grid) bw(0.5) plot histogram
. lpdensity data, grid(grid) bw(0.5) plot ciuniform level(90)
```

Before closing this appendix, we illustrate the bandwidth selector `lpbwdensity`. By default, this command computes the MSE-optimal bandwidth for estimating the PDF with a local quadratic regression and triangular kernel:

```
. capture drop grid
. gen grid = -0.5 + 0.5 * _n if _n <= 9
. lpbwdensity data, grid(grid)

Bandwidth Selection for Local Polynomial Density Estimation.

 Sample size                                (n=)            2000
 Polynomial order for point estimation      (p=)               2
 Density function estimated                 (v=)               1
 Kernel function                                        triangular
 Bandwidth selection method                                mse-dpi

--------------------------------
 Index    Grid      B.W.   Eff.n
--------------------------------
   1    0.0000    0.3812     258
   2    0.5000    0.6167     947
   3    1.0000    0.5254     946
   4    1.5000    0.7212    1168
   5    2.0000    0.5599     667
--------------------------------
   6    2.5000    0.4835     298
   7    3.0000    0.4925     114
   8    3.5000    1.1727     183
   9    4.0000    0.7140      22
--------------------------------
```

Finally, the following computes the IMSE-optimal bandwidth for density estimation.

```
. lpbwdensity data, grid(grid) bwselect(imse-dpi)
```

Affiliation:

Matias D. Cattaneo
Department of Operations Research and Financial Engineering
Princeton University
227 Sherrerd Hall
Princeton, New Jersey 08544, United States of America
E-mail: cattaneo@princeton.edu
URL: https://cattaneo.princeton.edu/

Michael Jansson
Department of Economics
University of California, Berkeley
530 Evans Hall #3880
Berkeley, California 94720, United States of America
E-mail: mjansson@econ.berkeley.edu
URL: https://eml.berkeley.edu/~mjansson/

Xinwei Ma
Department of Economics
University of California, San Diego
9500 Gilman Dr. #0508
La Jolla, California 92093, United States of America
E-mail: x1ma@ucsd.edu
URL: https://sites.google.com/view/xinweima/

Journal of Statistical Software https://www.jstatsoft.org/
published by the Foundation for Open Access Statistics https://www.foastat.org/
January 2022, Volume 101, Issue 2 *Submitted:* 2019-07-13
doi:10.18637/jss.v101.i02 *Accepted:* 2021-02-22