Please see the online version of this article for supplementary materials.

Fast Computation of Kernel Estimators

Vikas C. RAYKAR, Ramani DURAISWAMI, and Linda H. ZHAO

The computational complexity of evaluating the kernel density estimate (or its derivatives) at m evaluation points given n sample points scales quadratically as $O(nm)$—making it prohibitively expensive for large datasets. While approximate methods like binning could speed up the computation, they lack a precise control over the accuracy of the approximation. There is no straightforward way of choosing the binning parameters a priori in order to achieve a desired approximation error. We propose a novel computationally efficient ϵ-*exact* approximation algorithm for the univariate Gaussian kernel-based density derivative estimation that reduces the computational complexity from $O(nm)$ to linear $O(n + m)$. The user can specify a desired accuracy ϵ. The algorithm guarantees that the actual error between the approximation and the original kernel estimate will always be less than ϵ. We also apply our proposed fast algorithm to speed up automatic bandwidth selection procedures. We compare our method to the best available binning methods in terms of the speed and the accuracy. Our experimental results show that the proposed method is almost twice as fast as the best binning methods and is around five orders of magnitude more accurate. The software for the proposed method is available online.

Key Words: Bandwidth estimation; Binning; Fast Fourier transform; Kernel density derivative estimation; Kernel density estimation.

1. INTRODUCTION

Density estimation techniques are widely used in exploratory data analysis, data modeling, and various inference procedures in statistics and machine learning. The task of density estimation is to compute an estimate $\widehat{f}$ based on n iid samples $x_1, \ldots, x_n \in \mathbb{R}$ drawn from an unknown density f. One of the most popular nonparametric methods for density estimation is the *kernel density estimator* (KDE) (see Wand and Jones 1995 for a review)

$$\widehat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right), \tag{1.1}$$

Vikas C. Raykar is Research Scientist, CAD and Knowledge Solutions, Siemens Healthcare, Malvern, PA 19355 (E-mail: *vikasraykar@gmail.com*). Ramani Duraiswami is Associate Professor, Department of Computer Science, University of Maryland, College Park, MD 20742. Linda H. Zhao is Associate Professor, Department of Statistics, University of Pennsylvania, Philadelphia, PA 19104.

where K is the *kernel function* (such that $K(u) \geq 0$ and $\int_{\mathbb{R}} K(u) \, du = 1$) and $h \in \mathbb{R}^+$ is the *bandwidth* of the kernel. Naive direct evaluation of (1.1) at m different evaluation points $x_{e1}, \ldots, x_{em} \in \mathbb{R}$ requires $O(nm)$ kernel evaluations and $O(nm)$ multiplications and additions, making it prohibitively expensive for large datasets.

Estimation of the bandwidth (that is optimal in some sense) is also a computationally intensive task. A plethora of techniques have been proposed for automatic data-driven bandwidth selection (see Jones, Marron, and Sheather 1996 for a review). The most successful state-of-the-art methods rely on the estimation of general integrated squared *density derivative functionals*. This is the most computationally intensive task, the computational cost being $O(n^2)$. The core task contributing to this is the computation of an estimate of the density derivative. For this reason we will also address the density derivative estimation problems as well.

A simple estimator for the density derivative is obtained by taking the derivative of the kernel density estimate (Bhattacharya 1967; Schuster 1969). If the kernel K is differentiable r times, then the rth density derivative estimate $\widehat{f}^{(r)}(x)$ can be computed as

$$\widehat{f}^{(r)}(x) = \frac{1}{nh^{r+1}} \sum_{i=1}^{n} K^{(r)} \left(\frac{x - x_i}{h} \right), \tag{1.2}$$

where $K^{(r)}$ is the rth derivative of the kernel K. The direct approach of evaluating the density derivative (1.2) at m evaluation points given n sample points from the density also requires $O(nm)$ kernel evaluations and $O(nm)$ additions and multiplications.

The most commonly used method to reduce the computational burden for KDE is to use an *approximation* technique commonly known as *binning* (Scott 1981; Silverman 1982; Scott and Sheather 1985; Härdle and Scott 1992). The main idea behind binning is to subdivide the interval into an equally spaced mesh of G ($\ll n$) *grid points*, $g_1 < \cdots < g_G$, and replace the data by grid counts $c_1, \ldots, c_G$, where c_j is a weight chosen to represent the amount of data near g_j. By computing the kernel estimates based on the grid counts, the number of kernel evaluations and the number of additions and multiplications reduce to $O(Gm)$. If the evaluation points are also on a regular grid (this is usually the case for graphical analysis of data)—so that $m = G$—then further savings can be obtained by exploiting the fact that certain kernel evaluations are used repeatedly. This can reduce the number of kernel evaluations from $O(G^2)$ to $O(G)$. The number of additions and multiplications required is still $O(G^2)$—which can be further reduced to $O(G \log G)$ by using the fast Fourier transform (FFT) to perform the discrete convolution (Silverman 1982). Table 1 summarizes the time required by the different approximate algorithms. Further applications of these ideas, to more complex problems (including density derivative estimation and kernel regression), can be found in the works of Fan and Marron (1994), Turlach (1994), Wand (1994), Hall and Wand (1996), and Turlach and Wand (1996). Another class of methods for polynomial kernels has been proposed by Seifert et al. (1994).

The accuracy of the binned estimator has been previously studied by Hall (1982), Jones and Lotwick (1983), Scott and Sheather (1985), Jones (1989), Härdle and Scott (1992), Hall and Wand (1996), and González-Manteiga, Sánchez Sellero, and Wand (1996). The

Table 1. The computational complexity for various methods of evaluating the approximate kernel density estimates at m evaluation points given n sample points from the density.

Method	Number of kernel evaluations	Number of operations	Approximation error
Direct	$O(nm)$	$O(nm)$	Exact
Binning with $G \ll n$ bins	$O(Gm)$	$O(Gm)$	Accuracy increases with G
If evaluation points are equispaced			
Binning with $G = m$ bins	$O(G)$	$O(G^2)$	No mechanism to choose G
Binning with $G = m$ bins and FFT	$O(G)$	$O(G \log G)$	
Proposed method	$O(pn + pm)$		User controlled

NOTE: p is a constant that depends on the desired accuracy.

accuracy depends on the number of grid points G and can be made arbitrarily good by increasing G, at the expense of increase in the number of computations. However, there is no straightforward way of choosing G in order to achieve a desired approximation error. Also for a given G there are no good bounds on the approximation error. Hall and Wand (1996) proposed that using G between 100 and 500 should give a reasonably good approximation. For visual purposes Fan and Marron (1994) recommended using an equally spaced grid of 400 points. However, it should be clear that G depends heavily on the bandwidth of the kernel used (small bandwidths require a finer grid) and also on type of density (densities with sharp peaks require a much finer grid).

In this article we propose a very different computationally efficient approximation algorithm for the univariate kernel density estimation. It can be used to estimate the rth derivative of the density (note that $r = 0$ corresponds to the usual KDE). The algorithm reduces the computational complexity from $O(nm)$ to linear $O(n + m)$. The algorithm is based on the Taylor series expansion of the Gaussian kernel and retaining only the first few terms so that the error due to truncation is less than the desired error. The technique is inspired by the fast multipole methods used in computational physics (Greengard 1994).

Unlike binning, a desirable property of our proposed algorithm is that *we are able to control the approximation error*. Because of the well-known optimal properties of the original kernel estimators we have chosen to measure the errors between the proposed estimator to the original kernel estimator. The user specifies a desired accuracy ϵ. The algorithm guarantees that the actual error between the approximation and the original kernel estimate will always be less than ϵ (made more precise in Section 2). The constant hidden in $O(n + m)$ depends on the desired *accuracy*, ϵ, which can be *arbitrarily* small. More speedup can be obtained at the cost of reduced accuracy. In fact, for machine precision accuracy ($\epsilon < 10^{-16}$) there is no difference in the answers provided by the direct and the fast approximate algorithm. Our experimental results (see Table 2) show that for a wide range of densities the proposed method is almost twice as fast as the best binning methods and is around five orders of magnitude more accurate.

The binning methods achieve considerable savings by assuming that the evaluation points lie on an equally spaced grid. This property helps to reduce the number of kernel evaluations and, more importantly, to use the fast Fourier transform—which needs the

evaluation points to be on an equally spaced grid. The proposed algorithm works irrespective of whether the evaluation points are on an equally spaced grid or completely scattered. An equally spaced grid may be suitable for visualization purposes. For evaluation of kernel functionals the evaluation points are the data points themselves, which do not necessarily lie on an equally spaced grid. There have been some attempts to derive FFT-like algorithms for nonuniform grid (Potts and Steidl 2003).

The proposed method can also be used in conjunction with the binning methods, leading to further computational savings. It can be used to speed up nonparametric kernel regression methods. The estimation of the density derivative also comes up in various other applications (Singh 1977) like estimation of modes of densities (Fukunaga and Hostetler 1975) and estimation of the derivatives of the projection index in projection pursuit algorithms (Huber 1985).

The rest of the article is organized as follows. In Section 2 we make precise the notion of our ϵ-exact approximation. The proposed fast algorithm is described in detail in Sections 3 and 4 with most of the technical details described in the appendices available online as supplementary material. In Section 5 we show how the proposed procedure can be used to speed up bandwidth estimation. In Section 6 we show simulation results on the speedup achieved and the corresponding approximation error with points sampled from a series of density functions of Marron and Wand (1992) with varying degree of smoothness and complexity. Our experimental results show that to achieve a desired error our proposed algorithm is almost twice as fast as the best binning methods and is around five orders of magnitude more accurate.

2. GAUSSIAN KERNEL DENSITY DERIVATIVE ESTIMATION

A widely used kernel is the Gaussian kernel with zero mean and unit variance, that is, $K(u) = (1/\sqrt{2\pi})e^{-u^2/2}$. We have chosen to use the Gaussian kernel so that all the rth derivatives can be easily estimated through the rth derivative of the kernel estimate. A similar algorithm can be derived for other kernels as well. The rth derivative of the Gaussian kernel $K(u)$ is given by $K^{(r)}(u) = (-1)^r H_r(u)K(u)$, where $H_r(u)$ is the rth Hermite polynomial. The Hermite polynomials are a set of orthogonal polynomials. The first few Hermite polynomials are $H_0(u) = 1$, $H_1(u) = u$, and $H_2(u) = u^2 - 1$. Hence from (1.2), the density derivative estimate with the Gaussian kernel can be written as

$$\widehat{f}^{(r)}(x) = \frac{(-1)^r}{\sqrt{2\pi}nh^{r+1}} \sum_{i=1}^{n} H_r\left(\frac{x - x_i}{h}\right)e^{-(x-x_i)^2/2h^2}. \tag{2.1}$$

The computational complexity of evaluating the rth derivative of the density estimate from n data points at m evaluation points is thus $O(rnm)$. Let us say we have to estimate the density derivative at m evaluation points $x_{e1}, \ldots, x_{em}$. More generally, we need to evaluate the following sum:

$$\widehat{f}^{(r)}(x_{ej}) = \sum_{i=1}^{n} q_i H_r\left(\frac{x_{ej} - x_i}{h_1}\right)e^{-(x_{ej}-x_i)^2/h_2^2}, \qquad j = 1, \ldots, m, \tag{2.2}$$

where we have defined $q_i = (-1)^r/(\sqrt{2\pi} n h^{r+1})$, $h_1 = h$, and $h_2 = \sqrt{2}h$. A similar sum (with appropriately defined q_i, h_1, and h_2) is also involved in nonparametric kernel regression and the same algorithm can be used to accelerate it as well.

The computational complexity of evaluating (2.2) is $O(rnm)$. We present an ϵ-*exact* approximation algorithm that reduces the computational complexity to $O(prn + qpr^2m)$, where the constants p and q depend on the desired precision ϵ and the bandwidth h. For any given $\epsilon > 0$ the algorithm computes an approximation $\widehat{f}_\epsilon^{(r)}(x_{ej})$ such that for any x_{ej}

$$\left| \widehat{f}_\epsilon^{(r)}(x_{ej}) - \widehat{f}^{(r)}(x_{ej}) \right| \le Q\epsilon, \tag{2.3}$$

where $Q = \sum_{i=1}^n |q_i|$. We call $\widehat{f}_\epsilon^{(r)}(x_{ej})$ an ϵ-*exact* approximation to $\widehat{f}^{(r)}(x_{ej})$.[1]

3. THE PROPOSED FAST ALGORITHM

The proposed fast algorithm is based on separating the x_i and x_{ej} in the Gaussian kernel via factorization by Taylor series and retaining only the first few terms so that the error due to truncation of the infinite series is less than the desired error. For any nonnegative integer p the Gaussian function can be factorized around any point $x_* \in \mathbb{R}$ via the Taylor series as follows (see Appendix A in the supplementary material available online):

$$e^{-(x_{ej}-x_i)^2/h_2^2} = \sum_{k=0}^{p-1} \frac{2^k}{k!} \left[e^{-|x_i-x_*|^2/h_2^2} \left(\frac{x_i - x_*}{h_2} \right)^k \right]$$

$$\times \left[e^{-|x_{ej}-x_*|^2/h_2^2} \left(\frac{x_{ej} - x_*}{h_2} \right)^k \right] + error_p, \tag{3.1}$$

where $error_p$, the error due to retaining only the first p terms in the expansion, is bounded as

$$error_p \le \frac{2^p}{p!} \left(\frac{|x_i - x_*|}{h_2} \right)^p \left(\frac{|x_{ej} - x_*|}{h_2} \right)^p e^{-(|x_i-x_*|-|x_{ej}-x_*|)^2/h_2^2}. \tag{3.2}$$

The Hermite function can be factorized via the binomial theorem as follows (see Appendix B in the supplementary material available online):

$$H_r\left(\frac{x_{ej} - x_i}{h_1} \right) = \sum_{s=0}^{\lfloor r/2 \rfloor} \sum_{t=0}^{r-2s} a_{st} \left(\frac{x_i - x_*}{h_1} \right)^t \left(\frac{x_{ej} - x_*}{h_1} \right)^{r-2s-t}, \tag{3.3}$$

where

$$a_{st} = \frac{(-1)^{s+t} r!}{2^s s! t! (r - 2s - t)!}. \tag{3.4}$$

---

[1] For the task of density derivative estimation $Q = 1/(\sqrt{2\pi} h^{r+1})$. In practice we set $\epsilon = \sqrt{2\pi} h^{r+1} \epsilon'$, where ϵ' is our desired accuracy. This guarantees that the absolute value of the error between the direct implementation and the approximate computation will be always less than ϵ'.

Combining (3.1) and (3.3), $\widehat{f}^{(r)}(x_{ej})$ after ignoring the error term ($error_p$) can be approximated as

$$\widehat{f}_\epsilon^{(r)}(x_{ej}) = \sum_{k=0}^{p-1}\sum_{s=0}^{\lfloor r/2 \rfloor}\sum_{t=0}^{r-2s} a_{st} B_{kt} e^{-|x_{ej}-x_*|^2/h_2^2}\left(\frac{x_{ej}-x_*}{h_2}\right)^k\left(\frac{x_{ej}-x_*}{h_1}\right)^{r-2s-t}, \quad (3.5)$$

where we have defined

$$B_{kt} = \frac{2^k}{k!}\sum_{i=1}^{n} q_i e^{-|x_i-x_*|^2/h_2^2}$$

$$\times \left(\frac{x_i-x_*}{h_2}\right)^k\left(\frac{x_i-x_*}{h_1}\right)^t, \qquad k=0,\ldots,p-1, t=0,\ldots,r. \quad (3.6)$$

The coefficients B_{kt} do not depend on x_{ej} and hence can be evaluated separately with $O(prn)$ operations. Once B_{kt} are evaluated, computing $\widehat{f}_\epsilon^{(r)}(x_{ej})$ at m points requires $O(pr^2m)$ operations. Hence the number of computations required has reduced from $O(rnm)$ to $O(prn + pr^2m)$.

In order to maintain the desired error bound ϵ the truncation number p can be chosen using the bound (3.2) (see Appendix C in the supplementary material available online). Thus far, we have used the Taylor's series expansion about a certain point x_*. However, if we use one common x_*, we typically would require very high truncation number p since the Taylor's series gives good approximation only in a small open interval around x_*. So we uniformly subdivide the space into L intervals of length, say, $2r_x$. Each of the n data points $x_1,\ldots,x_n$ is assigned into one of the L clusters (the closest one), S_l for $l=1,\ldots,L$ with c_l being the center of each cluster. This step is similar to the binning idea. The aggregated coefficients B_{kt} are now computed for each cluster and the total contribution from all the clusters is summed up. Hence

$$\widehat{f}_\epsilon^{(r)}(x_{ej}) = \sum_{l=1}^{L}\sum_{k=0}^{p-1}\sum_{s=0}^{\lfloor r/2 \rfloor}\sum_{t=0}^{r-2s} a_{st} B_{kt}^l e^{-|x_{ej}-c_l|^2/h_2^2}\left(\frac{x_{ej}-c_l}{h_2}\right)^k\left(\frac{x_{ej}-c_l}{h_1}\right)^{r-2s-t}, \quad (3.7)$$

where

$$B_{kt}^l = \frac{2^k}{k!}\sum_{x_i\in S_l} q_i e^{-|x_i-c_l|^2/h_2^2}\left(\frac{x_i-c_l}{h_2}\right)^k\left(\frac{x_i-c_l}{h_1}\right)^t, \qquad l=1,\ldots,L. \quad (3.8)$$

Since the Gaussian function decays very rapidly a further speedup is achieved if we ignore all the points belonging to a cluster, if the cluster is greater than a certain distance from the target point, i.e., for x_{ej} we can ignore all points in the cluster c_l if $|x_{ej}-c_l| > r_{ex}$. The cluster cutoff radius r_{ex} depends on the desired error ϵ. Substituting $h_1 = h$ and $h_2 = \sqrt{2}h$ the final summation can be written as

$$\widehat{f}_\epsilon^{(r)}(x_{ej}) = \sum_{\forall l \text{ s.t. } |x_{ej}-c_l|\leq r_{ex}}\sum_{k=0}^{p-1}\sum_{s=0}^{\lfloor r/2 \rfloor}\sum_{t=0}^{r-2s} a_{st} B_{kt}^l e^{-|x_{ej}-c_l|^2/2h^2}$$

$$\times \left(\frac{x_{ej}-c_l}{h}\right)^{k+r-2s-t}, \quad (3.9)$$

where

$$B_{kt}^l = \frac{1}{k!} \sum_{x_i \in S_l} q_i e^{-|x_i - c_l|^2 / 2h^2} \left(\frac{x_i - c_l}{h} \right)^{k+t}. \tag{3.10}$$

Computing the coefficients B_{kt}^l for all the clusters is $O(prn)$, since each point belongs to only one cluster. Evaluation of $\widehat{f}_\epsilon^{(r)}(x_{ej})$ at m points is $O(qpr^2m)$, where we define $q \ (\leq L)$ to be the maximum number of neighbor clusters which influence x_{ej}. Hence the total computational complexity is $O(prn + qpr^2m)$. For each cluster we need to store all the pr coefficients. Hence the memory needed is of the order of $O(prL + n + m)$.

Finally, given any $\epsilon > 0$, we want to choose the following free parameters: r_x (the interval radius), r_{ex} (the cutoff radius for each cluster), and p (the truncation number) such that for any point x_{ej}, $|\widehat{f}_\epsilon^{(r)}(x_{ej}) - \widehat{f}^{(r)}(x_{ej})| \leq Q\epsilon$, where $Q = \sum_{i=1}^n |q_i|$. Appendix C in the supplementary material available online describes in detail how to choose these parameters. The final algorithm is summarized in the next section.

4. ALGORITHM: FAST KERNEL DENSITY DERIVATIVE ESTIMATION

Given the following inputs:

(1) $x_1, \ldots, x_n$, n points drawn from an unknown density f,

(2) $x_{e1}, \ldots, x_{em}$, m points where we want to evaluate the density derivative,

(3) $r \geq 0$, the order of density derivative ($r = 0$ corresponds to the density estimate),

(4) $h > 0$, the bandwidth of the Gaussian kernel, and

(5) $\epsilon > 0$, the desired approximation error,

the following algorithm computes an ϵ-exact approximation $\widehat{f}_\epsilon^{(r)}(x_{ej})$ to the Gaussian kernel-based density derivative estimate

$$\widehat{f}^{(r)}(x_{ej}) = \frac{(-1)^r}{\sqrt{2\pi} n h^{r+1}} \sum_{i=1}^n H_r \left(\frac{x_{ej} - x_i}{h} \right) e^{-(x_{ej} - x_i)^2 / 2h^2}, \tag{4.1}$$

such that for any x_{ej}, the error $|\widehat{f}_\epsilon^{(r)}(x_{ej}) - \widehat{f}^{(r)}(x_{ej})| \leq \epsilon$.

1. Scale all the $n + m$ points $x_1, \ldots, x_n, x_{e1}, \ldots, x_{em}$ to lie in the unit interval $[0, 1]$. Scale the bandwidth h appropriately.

2. Define $q = (-1)^r / \sqrt{2\pi} n h^{r+1}$ and set $\epsilon' = \epsilon / n |q|$.

3. Choose the interval radius $r_x = h/2$. Subdivide the unit interval into L intervals (S_l for $l = 1, \ldots, L$) of length $2r_x$. Let c_l be the center of each interval. Assign each x_i to the closest interval based on its distance to the interval center c_l.

4. Choose the cutoff radius $r_{ex} = r_x + 2h \sqrt{\ln(\sqrt{r!}/\epsilon')}$.

5. Choose the truncation number p numerically such that

$$\frac{\sqrt{r!}}{p!}\left(\frac{r_x b}{h^2}\right)^p e^{-(r_x - b)^2/4h^2} \le \epsilon', \quad \text{where}$$

$$b = \min\left(r_{ex}, \frac{r_x + \sqrt{r_x^2 + 8ph^2}}{2}\right). \tag{4.2}$$

6. For each S_l compute the aggregated coefficients B_{kt}^l for $k = 0, \ldots, p-1$ and $t = 0, \ldots, r$:

$$B_{kt}^l = \frac{1}{k!} \sum_{x_i \in S_l} q e^{-|x_i - c_l|^2/2h^2}\left(\frac{x_i - c_l}{h}\right)^{k+t}. \tag{4.3}$$

7. Compute the coefficients a_{st} for $s = 0, \ldots, \lfloor r/2 \rfloor$ and $t = 0, \ldots, r$:

$$a_{st} = \frac{(-1)^{s+t} r!}{2^s s! t! (r - 2s - t)!}. \tag{4.4}$$

8. The approximate kernel density derivative at point x_{ej} is computed as

$$\widehat{f}_\epsilon^{(r)}(x_{ej}) = \sum_{\forall l \text{ s.t. } |x_{ej} - c_l| \le r_{ex}} \sum_{k=0}^{p-1} \sum_{s=0}^{\lfloor r/2 \rfloor} \sum_{t=0}^{r-2s} a_{st} B_{kt}^l e^{-|x_{ej} - c_l|^2/2h^2}$$

$$\times \left(\frac{x_{ej} - c_l}{h}\right)^{k+r-2s-t}. \tag{4.5}$$

9. Rescale back the estimate.

5. FAST BANDWIDTH SELECTION

A plethora of techniques have been proposed for automatic data-driven bandwidth selection (see Jones, Marron, and Sheather 1996 for a review). The most successful state-of-the-art methods rely on the estimation of general integrated squared *density derivative functionals*. This is the most computationally intensive task, the computational cost being $O(n^2)$, where n is the number of sample points. One of the most successful among all the current methods is the *solve-the-equation plug-in* method (Sheather and Jones 1991). This involves the numerical solution of a nonlinear equation. Iterative methods to solve this equation repeatedly use the density derivative functional estimator for different bandwidths which further increases the computational burden. The general integrated squared density derivative functional is defined as $R(f^{(s)}) = \int_{\mathbb{R}} [f^{(s)}(x)]^2 dx$. Using integration by parts, this can be written in the following form (Wand and Jones 1995):

$$R(f^{(s)}) = (-1)^s \int_{\mathbb{R}} f^{(2s)}(x) f(x) dx. \tag{5.1}$$

More specifically, for even $r = 2s$ we are interested in estimating density functionals of the form

$$\Phi_r = \int_{\mathbb{R}} f^{(r)}(x) f(x) dx = E\big[f^{(r)}(X)\big]. \tag{5.2}$$

An estimator for Φ_r is

$$\widehat{\Phi}_r = \frac{1}{n} \sum_{i=1}^{n} \widehat{f}^{(r)}(x_i), \tag{5.3}$$

where $\widehat{f}^{(r)}(x_i)$ is the estimate of the rth derivative of the density $f(x)$ at $x = x_i$. Using a kernel density derivative estimate for $\widehat{f}^{(r)}(x_i)$, we have

$$\widehat{\Phi}_r = \frac{1}{n^2 h^{r+1}} \sum_{i=1}^{n} \sum_{j=1}^{n} K^{(r)}\left(\frac{x_i - x_j}{h}\right). \tag{5.4}$$

Computing $\widehat{\Phi}_r$ requires $O(rn^2)$ computations and hence can be very expensive if a direct algorithm is used. The proposed algorithm can be used directly to reduce this to $O((pr + qpr^2)n)$.

6. EXPERIMENTAL RESULTS

We demonstrate the speedup achieved of the proposed algorithm on the mixture of normal densities used by Marron and Wand (1992). The family of normal mixture densities is extremely rich and any density can be approximated arbitrarily well by a member of this family. Fifteen such densities were proposed by Marron and Wand (1992) as a typical representative of the densities likely to be encountered in real data situations.

We sample n points from each density. In order to compare against the binning methods the kernel density estimates are evaluated at an equally spaced grid of m points. We compare the proposed fast algorithm against the binned approximation and the naive direct exact implementation in terms of the computation time and the approximation error. The proposed fast algorithm was programmed in C++ with MATLAB bindings and was run on a 2.4 GHz processor with 2 GB of RAM. The code is provided as a supplemental material. For binning we use functions from the general purpose statistical smoothing library written in MATLAB by J. S. Marron.[2] We have modified the code so that the discrete convolution in binning can be done using the fast Fourier transform (FFT), which for large number of bins is much faster than the function provided by Marron. The speedup and the actual error depend on the number of data points n, the number of evaluation points m, the bandwidth of the kernel h, the order of the density derivative r, the density type, and for the proposed algorithm the desired error ϵ.

6.1 SPEEDUP AS A FUNCTION OF n—THE NUMBER OF SAMPLE POINTS

For the first set of experiments we fix the evaluation grid and plot the computation time and the approximation error as a function of the number of sample points. Figure 1(a) plots the time (in seconds) required to compute the KDE as a function of n—the number of sample points—for the direct, binned, binned with FFT, and the proposed fast method. The

---

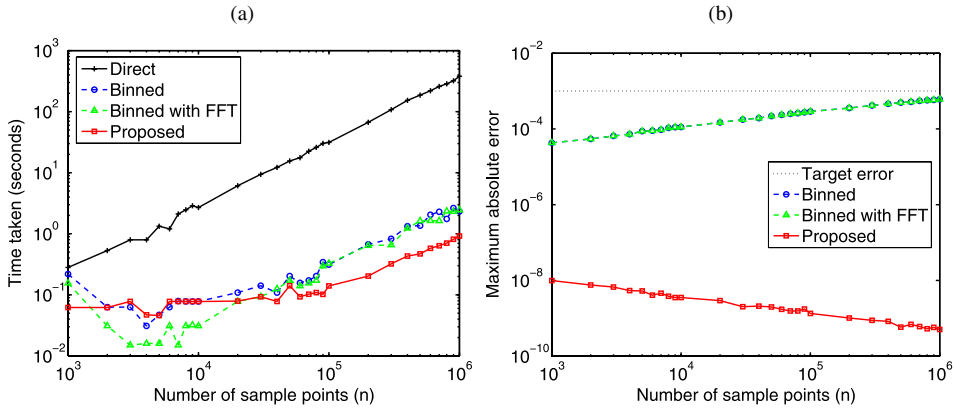[2]Can be downloaded from *http://www.unc.edu/~marron/marron_software.html*.

Figure 1.   (a) The time (in seconds) required to compute the KDE as a function of n—the number of sample points—for the direct, binned approximation, binned approximation with FFT, and the proposed fast method. The KDE was estimated at an equally spaced grid of $m = 1000$ points. The points were sampled from the strongly skewed density of Marron and Wand (1992). (b) The maximum absolute error between the direct implementation (exact) and the fast approximate algorithms. For the proposed algorithm the desired accuracy was set at 10^{-3}. The binned FFT has the same error as the binned method.

KDE was estimated at an equally spaced grid of $m = 1000$ points. The points were sampled from the strongly skewed density of Marron and Wand (1992). The same bandwidth chosen using the solve-the-equation plug-in method of Jones, Marron, and Sheather (1996) was used for all the methods. For this experiment we do not report the time taken to compute the optimal bandwidth. For each n, we use the optimal bandwidth selected by our method. Choosing the optimal bandwidth is also a computationally intensive task and we separately report the results for bandwidth estimation in Section 6.6.

The approximate methods (binned and proposed) are both significantly faster than the naive direct computation. For smaller number of points ($n < 10{,}000$) all the approximate methods take almost the same time. However, asymptotically as n increases, the proposed method is faster than the binned approximation. For example, at $n = 10^6$ sample points the direct computation takes 379.27 sec, while the binned approximation takes 2.32 sec (a speedup of 164), and the proposed method takes only 0.92 sec (a speedup of 412). Note that FFT does not help in speeding up the binned method since the evaluation grid is small. The benefits of FFT can be seen only for large m (see Section 6.3).

Figure 1(b) plots the maximum absolute error between the direct exact implementation and the fast approximate algorithms. For the proposed algorithm we set the desired accuracy to 10^{-3}. More precisely we set $\epsilon = \sqrt{2\pi} h^{r+1} 10^{-3}$ (see (2.3)). This guarantees that the absolute value of the error between the direct implementation and the approximate computation will be always less than 10^{-3}, as validated from the plot. In fact, the error bound is very conservative (the error is of the order of 10^{-8} almost toward machine level precision) and in fact we can safely set a much lower accuracy. The binned method gives no guarantees on the error. For the binning methods we observe that as n increases the error increases and binning does not provide an automatic mechanism to control this error.

Table 2. The time (in seconds) required to compute the kernel density estimate for the direct, binned FFT, and the proposed fast method for all the 15 densities of Marron and Wand (1992). The speedup achieved and the maximum absolute error between the direct implementation (exact) and the fast approximate algorithms are also shown. The KDE was evaluated at an equally spaced grid of $m = 1000$ points using $n = 100,000$ points sampled from each density. For the proposed algorithm we set the desired accuracy to 10^{-3}.

Density	Direct Time (sec)	Binned FFT Time (sec)	Binned FFT Speedup	Binned FFT Error	Proposed Time (sec)	Proposed Speedup	Proposed Error	Speedup over binned FFT
1	21.41	0.31	68.39	6.2e–006	0.13	171.26	4.4e–010	2.50
2	29.66	0.33	90.41	2.5e–005	0.14	211.83	2.9e–010	2.34
3	35.89	0.20	176.80	1.9e–003	0.16	228.60	2.4e–011	1.29
4	23.89	0.16	152.17	1.6e–003	0.11	217.18	3.5e–011	1.43
5	28.70	0.20	141.39	4.9e–003	0.16	183.99	3.5e–011	1.30
6	28.78	0.16	183.32	1.2e–005	0.09	309.47	2.0e–010	1.69
7	22.05	0.16	140.43	1.2e–005	0.08	282.65	1.5e–010	2.01
8	26.91	0.09	286.24	2.7e–005	0.06	427.10	1.6e–010	1.49
9	29.75	0.20	146.55	2.2e–005	0.13	238.00	1.5e–010	1.62
10	31.66	0.11	290.42	7.6e–004	0.08	400.71	1.9e–011	1.38
11	25.33	0.31	80.92	2.8e–005	0.19	134.73	2.0e–010	1.66
12	36.98	0.31	118.16	1.2e–003	0.20	182.19	3.2e–011	1.54
13	37.30	0.17	216.84	2.3e–004	0.08	478.17	4.7e–011	2.21
14	29.11	0.11	264.63	2.6e–003	0.08	373.19	1.5e–011	1.41
15	27.52	0.22	126.22	1.2e–003	0.14	195.14	1.6e–011	1.55

6.2 SPEEDUP FOR DIFFERENT DENSITIES

Table 2 shows the same results for all the 15 densities of Marron and Wand (1992). The KDE was evaluated at an equally spaced grid of $m = 1000$ points using $n = 100,000$ points sampled from each density. For a wide range of densities the proposed method is almost twice as fast as the best binning method and is around five orders of magnitude more accurate.

6.3 SPEEDUP AS A FUNCTION OF m—THE NUMBER OF EVALUATION POINTS

For the second set of experiments we fix $n = 10,000$ and plot the time and error as a function of m—the number of points on the evaluation grid. Figure 2(a) shows the running time (in seconds) for both the direct and the fast methods as a function of m. As can be seen from the plot, the time required for the binned method grows as $O(m^2)$ and can exceed the direct computation if a very fine evaluation grid is needed (large m). In such situations FFT helps to speed up computations reducing the time to $O(m \log m)$, almost linear in m. The proposed method takes linear $O(m)$ time. While the binned FFT appears to be the fastest method asymptotically, it can achieve an error comparable to the proposed method only for very large m (see Figure 2(b)). Also note that the FFT can be used only for an equally spaced evaluation grid while the proposed method does not require the evaluation points to be equispaced. The error bound for the proposed method is very conservative and we can obtain an improvement in the speedup by decreasing the required accuracy (see Section 6.4).
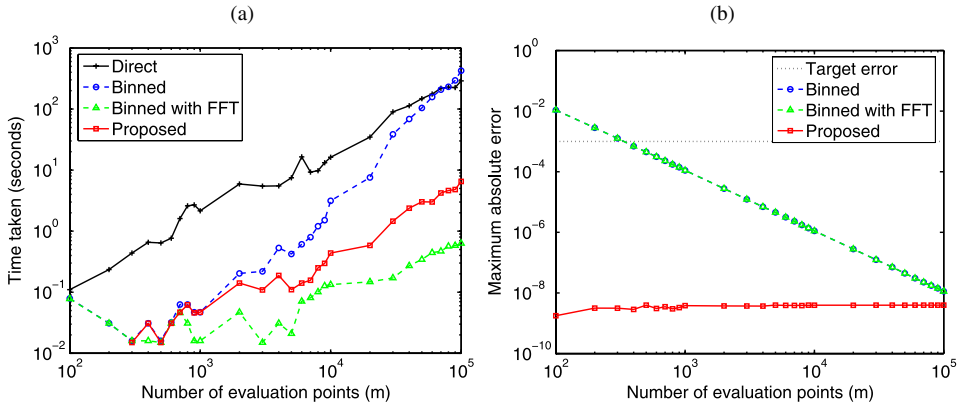
Figure 2.   (a) The time (in seconds) required to compute the KDE as a function of m—the number of grid points—for the direct, binned, binned FFT, and the proposed fast method. The KDE was estimated using $n = 10,000$ points sampled from the strongly skewed density of Marron and Wand (1992). (b) The maximum absolute error between the direct implementation (exact) and the fast approximate algorithms. For the proposed algorithm the desired accuracy was set at 10^{-3} (shown as a dotted line). The binned FFT has the same error as the binned method.

6.4   SPEEDUP AS A FUNCTION OF ϵ—THE DESIRED ERROR

Figure 3(a) shows the running time (in seconds) for both the direct and the proposed method as a function of ϵ—the desired accuracy—for the KDE evaluated at $m = 10,000$ points from $n = 10,000$ sample points. The time taken by the proposed method decreases as the desired error ϵ increases. An increase in speedup is obtained at the cost of reduced accuracy.
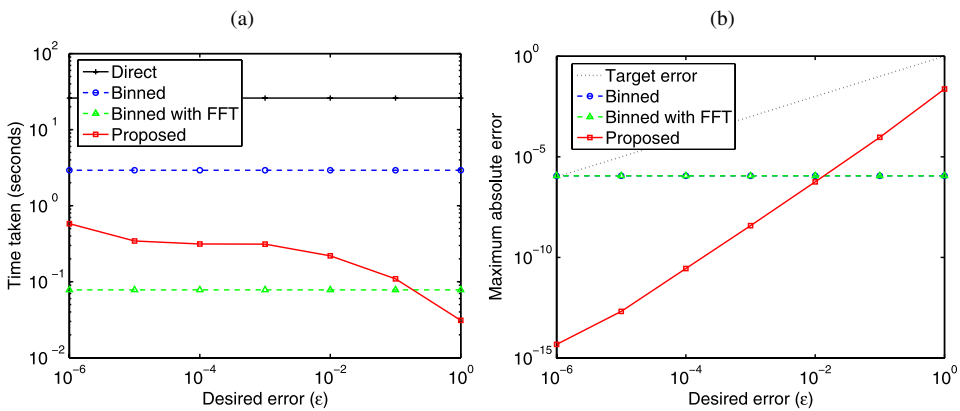

Figure 3.   (a) The time (in seconds) required to compute the KDE for the direct, binned, binned FFT, and the proposed fast method as a function of ϵ—the target accuracy for the proposed method. The KDE was evaluated at $m = 10,000$ points using $n = 10,000$ points sampled from the strongly skewed density of Marron and Wand (1992). (b) The maximum absolute error between the direct implementation (exact) and the fast approximate algorithms. The binned FFT has the same error as the binned method.
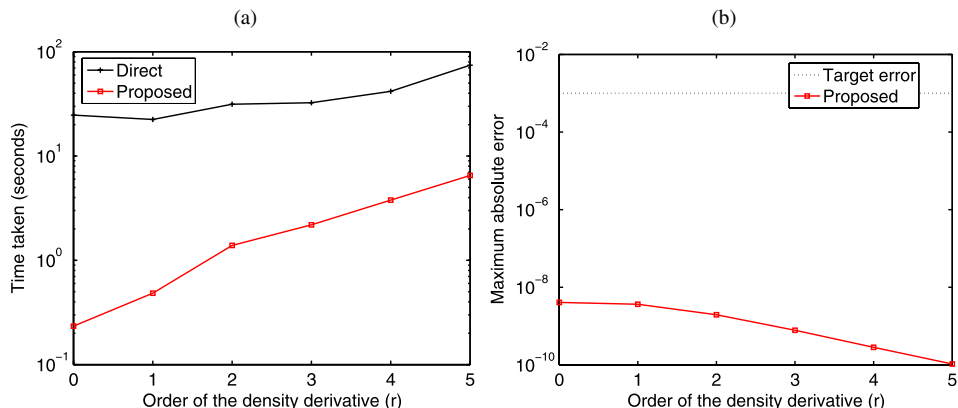
Figure 4.   (a) The time (in seconds) required to compute the kernel density derivative for the direct and the proposed fast method as a function of r—order of the derivative. The density derivative was evaluated at $m = 10{,}000$ points using $n = 10{,}000$ points sampled from the strongly skewed density of Marron and Wand (1992). (b) The maximum absolute error between the direct implementation (exact) and the fast approximate algorithms. For the proposed algorithm the desired accuracy was set at 10^{-3} (shown as a dotted line).

6.5   SPEEDUP FOR DENSITY DERIVATIVE ESTIMATION

The proposed method can be used to estimate the kernel density derivative estimate. Figure 4(a) plots the running time (in seconds) for both the direct and the proposed method as a function of r—the order of the density derivative. Note that $r = 0$ corresponds to the kernel density estimate. We could not compare with the binning methods as there was no code readily available. However, in terms of the speedup we expect performance similar to that of the density estimate. There could be a difference in the approximation error.

6.6   SPEEDUP FOR BANDWIDTH ESTIMATION

The solve-the-equation plug-in method of Jones, Marron, and Sheather (1996) was implemented in MATLAB with the core computational task of computing the density derivative written in C++. We sampled $n = 50{,}000$ points from each density. The optimal bandwidth was estimated using both the direct methods and the proposed fast method. For the fast method we used $\epsilon = 10^{-2}$. Table 3 shows the speedup achieved and the absolute relative error. Since the proposed method uses an approximation to compute the density derivative functional, we show the absolute relative error for the estimated bandwidth— defined as $|(h_{direct} - h_{fast})/h_{direct}|$, where h_{direct} is the bandwidth estimated using the direct method and h_{fast} is the bandwidth estimated using the proposed fast method. We obtained speedups in the range 85 to 150 with absolute relative error of 10^{-3} to 10^{-5}.

7. CONCLUSION

In this article we proposed a fast approximate algorithm for kernel density (and derivative) estimation which reduced the computational complexity from $O(nm)$ to linear $O(n + m)$. Unlike binning, a desirable property of our proposed algorithm is that we are able to control the approximation error. The user specifies a desired accuracy ϵ. The algo-

Table 3.   The bandwidth estimated using the solve-the-equation plug-in method for the 15 normal mixture densities of Marron and Wand. h_{direct} and h_{fast} are the bandwidths estimated using the direct and the proposed fast method, respectively. The running time in seconds for the direct and the fast methods is shown. The absolute relative error is defined as $|(h_{direct} - h_{fast})/h_{direct}|$. In the study $n = 50{,}000$ points were sampled from the corresponding densities. For the fast method we used $\epsilon = 10^{-2}$.

	Direct		Proposed			
Density	h_{direct}	Time (sec)	h_{fast}	Time (sec)	Speedup	Error
1	0.122159	5049.30	0.122343	39.27	128.59	1.5e–003
2	0.081035	6124.13	0.081150	56.39	108.60	1.4e–003
3	0.019657	9840.98	0.019659	87.69	112.23	1.4e–004
4	0.020450	8700.64	0.020454	79.31	109.70	1.7e–004
5	0.012897	7515.09	0.012905	58.39	128.70	6.1e–004
6	0.097448	7287.58	0.097592	67.91	107.32	1.5e–003
7	0.091950	7258.11	0.092009	65.19	111.34	6.4e–004
8	0.074210	6974.06	0.074306	50.83	137.21	1.3e–003
9	0.081219	6712.83	0.081307	79.72	84.21	1.1e–003
10	0.024390	8198.69	0.024394	66.14	123.96	1.6e–004
11	0.090918	7042.25	0.091066	71.27	98.82	1.6e–003
12	0.032118	11,111.61	0.032128	99.05	112.19	3.3e–004
13	0.044978	9645.44	0.044995	75.55	127.67	3.7e–004
14	0.028111	12,040.91	0.028115	91.11	132.16	1.1e–004
15	0.022927	12,377.81	0.022928	86.75	142.68	7.0e–005

rithm guarantees that the actual error between the approximation and the original kernel estimate will always be less than ϵ. The proposed algorithm works irrespective of whether the evaluation points are on an equally spaced grid or completely scattered. Our experimental results show that the proposed method is faster and more accurate than the widely used binning methods. We also apply our proposed fast algorithm to speed up the best automatic bandwidth selection procedure. The proposed method can also be used in conjunction with the binning methods, leading to further computational savings. Also the same algorithm can be used to speed up nonparametric kernel regression methods.

SUPPLEMENTAL MATERIALS

Online appendix: The appendix contains the detailed derivations of the factorization of the Gaussian function (Appendix A) and the Hermite polynomial (Appendix B). Appendix C describes how to choose the various parameters to achieve the desired error bound. (FastKernelEstimator_Appendix.pdf)

MATLAB package: The package contains the MATLAB code for the proposed algorithm for the fast kernel density derivative estimation and also for the bandwidth selection method of Sheather and Jones (1991). The core computation is written in C++ with a MATLAB wrapper. The package includes the compiled dll files for windows platform. The C++ source code files are also available online at *http://www.umiacs.umd.edu/~vikas/Software/optimal_bw/optimal_bw_code.htm* under the GNU Lesser General Public License. (FastKernelEstimator_MatlabPackage.zip)

REFERENCES

Bhattacharya, P. K. (1967), "Estimation of a Probability Density Function and Its Derivatives," *Sankhya, Ser. A*, 29, 373–382. [206]

Fan, J., and Marron, J. (1994), "Fast Implementations of Nonparametric Curve Estimators," *Journal of Computational and Graphical Statistics*, 3 (1), 35–56. [206,207]

Fukunaga, K., and Hostetler, L. (1975), "The Estimation of the Gradient of a Density Function, With Applications in Pattern Recognition," *IEEE Transactions on Information Theory*, 21 (1), 32–40. [208]

González-Manteiga, W., Sánchez Sellero, C., and Wand, M. P. (1996), "Accuracy of Binned Kernel Functional Approximations," *Computational Statistics and Data Analysis*, 22, 1–16. [206]

Greengard, G. (1994), "Fast Algorithms for Classical Physics," *Science*, 265 (5174), 909–914. [207]

Hall, P. (1982), "The Influence of Rounding Errors on Some Nonprametric Estimators of a Density and Its Derivatives," *SIAM Journal of Applied Mathematics*, 42, 390–399. [206]

Hall, P., and Wand, M. (1996), "On the Accuracy of Binned Kernel Density Estimators," *Journal of Multivariate Analysis*, 56 (2), 165–184. [206,207]

Härdle, W., and Scott, D. W. (1992), "Smoothing in Low and High Dimensions by Weighted Averaging Using Rounded Points," *Computational Statistics*, 7, 97–128. [206]

Huber, P. J. (1985), "Projection Pursuit," *The Annals of Statistics*, 13, 435–475. [208]

Jones, M. C. (1989), "Discretized and Interpolated Kernel Density Estimates," *Journal of the American Statistical Association*, 84, 733–741. [206]

Jones, M. C., and Lotwick, H. W. (1983), "On the Errors Involved in Computing the Empirical Characteristic Function," *Journal of Statistical and Computer Simulation*, 17, 133–149. [206]

Jones, M. C., Marron, J. S., and Sheather, S. J. (1996), "A Brief Survey of Bandwidth Selection for Density Estimation," *Journal of the American Statistical Association*, 91 (433), 401–407. [206,212,214,217]

Marron, J. S., and Wand, M. P. (1992), "Exact Mean Integrated Squared Error," *The Annals of Statistics*, 20 (2), 712–736. [208,213-217]

Potts, D., and Steidl, G. (2003), "Fast Summation at Nonequispaced Knots by NFFTs," *SIAM Journal of Scientific Computing*, 24, 2013–2037. [208]

Schuster, E. F. (1969), "Estimation of a Probability Density Function and Its Derivatives," *The Annals of Mathematical Statistics*, 40 (4), 1187–1195. [206]

Scott, D. (1981), "Using Computer-Binned Data For Density Estimation," in *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, ed. W. Eddy, New York: Springer-Verlag, pp. 282–294. [206]

Scott, D. W., and Sheather, S. J. (1985), "Kernel Density Estimation With Binned Data," *Communications in Statistics—Theory and Methods*, 14, 1353–1359. [206]

Seifert, B., Brockmann, M., Engel, J., and Gasser, T. (1994), "Fast Algorithms for Nonparametric Curve Estimation," *Journal of Computational and Graphical Statistics*, 3 (2), 192–213. [206]

Sheather, S., and Jones, M. (1991), "A Reliable Data-Based Bandwidth Selection Method for Kernel Density Estimation," *Journal of Royal Statistical Society, Ser. B*, 53 (3), 683–690. [212,218]

Silverman, B. (1982), "Algorithm AS 176: Kernel Density Estimation Using the Fast Fourier Transform," *Applied Statistics*, 31 (1), 93–97. [206]

Singh, R. S. (1977), "Applications of Estimators of a Density and Its Derivatives to Certain Statistical Problems," *Journal of the Royal Statistical Society, Ser. B*, 39 (3), 357–363. [208]

Turlach, B. (1994), "Fast Implementation of Density-Weighted Average Derivative Estimation," in *Computationally Intensive Statistical Methods. Computing Science and Statistics*, Vol. 26, eds. J. Sall and A. Lehman, Fairfax Station, VA: Interface Foundation of North America, pp. 28–33. [206]

Turlach, B., and Wand, M. (1996), "Fast Computation of Auxiliary Quantities in Local Polynomial Regression," *Journal of Computational and Graphical Statistics*, 5 (4), 337–350. [206]

Wand, M. (1994), "Fast Computation of Multivariate Kernel Estimators," *Journal of Computational and Graphical Statistics*, 3 (4), 433–445. [206]

Wand, M. P., and Jones, M. C. (1995), *Kernel Smoothing*, London: Chapman & Hall. [205,212]