A note on the use of binning in kernel smoothing

Dinh Tuan PHAM

Laboratory of Modeling and Computation, IMAG - C.N.R.S.

B.P. 53X, 38041 Grenoble cédex, France

Abstract

The aim of this paper is to provide a general binning scheme for fast computation of kernel and local polynomial estimation, both in the density and regression context. It is shown that binning is equivalent to using a different equivalent kernel. A method to design binning scheme for which this equivalent kernel approximates a given scaled kernel is proposed.

Keywords Binning. Fast Computation. Fast Fourier Transform. Discretization. Kernel Estimation. Local Polynomial Regression. Smoothing.

1 Introduction

Smoothing techniques are powerful and widely used methods in nonparametric estimation. The most common problems in this area are the density estimation and the nonlinear regression (or estimation of conditional expectation). For such problems (and others) the kernel method is among the most popular. However, this method is also quite costly in term of computation, in the density estimation or in the regression with irregular design. Indeed, consider, as example, the kernel density estimator defined by

$$\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K\Big(\frac{x - X_i}{h}\Big) \tag{1.1}$$

where $X_1, \ldots X_n$ are the data, K is the kernel and h denotes the bandwidth parameter. Computing this estimator at m points would require nm kernel evaluations. It is true that this numbers can be much reduced if one uses a kernel with compact support so that for a given x, many of the indices i would be such that $(x - X_i)/h$ fall outside the support of K. But then one also needs to perform a test to see if this is the case.

Recently, there are much interest in the binning technique (Jones and Lotwick, 1983, 1994, Jones, 1989, 1992, Scott and Sheather, 1985, Hall and Wand, 1993, Fan and Marron, 1994, ...) which can speed up considerably the computation. Binning, in its simplest form, can be describes as follows: consider a set of equi-spaced grid points $\{jb, j \in \mathbb{Z}\}$ with a spacing b (called bin-width), then replace each data point X_i by a grid point $\tilde{X}_i = j_i b$ closest to it. One thus get a new data set $X_1^\dagger, \ldots, X_n^\dagger$ for which n_j of them equal jb, $j \in \mathbb{Z}$. The "binned" density estimator at lb is then

$$\hat{f}^\dagger(lb) = \sum_{j=-\infty}^{\infty} \frac{1}{h} K\Big[(l-j)\frac{b}{h}\Big] \frac{n_j}{n}. \tag{1.2}$$

Thus the sequence $\hat{f}^\dagger(lb), l \in \mathbb{Z}$ is obtained as the discrete convolution of the sequences $h^{-1}K(jb/h), j \in \mathbb{Z}$ and $n_j/n, j \in \mathbb{Z}$. The advantage of binning stems essentially from the fact that the $h^{-1}K(jb/h)$ need to be computed only once and, assuming that K has compact support, only a small number p among them are nonzero and need to be evaluated. As for the discrete convolution, it is rather cheap. Indeed, $\hat{f}^\dagger(lb)$ need to be computed only at m of consecutive indices for which it can be nonzero, hence the convolution requires about mp multiplications and additions. The number m roughly equals to R_n/b where $R_n = \max(X_1, \ldots, X_n) - \min(X_1, \ldots, X_n)$ and if b is chosen to be a fixed fraction of h, p would be fixed. Thus mp would be of the same order as R_n/h. Since in any reasonable bandwidth choice procedure, nh would go to infinity with n faster than R_n, the total number of operations is of *lower* order than n.

Note The above discrete convolution may be performed through the fast Fourier transform (FFT). This might reduce further the computations, but not always so. Indeed one needs to apply the FFT on two sequences of length N, the smallest power of 2 not less than $m+p$. (One has to work with sequence of length at least $m+p$ in order that circular convolution equal ordinary convolution, at the point of interest.) The number of operations for these FFT is about $2N \log_2 N$. Compare with mp, the number of operations for computing convolution directly, the FFT method would be

advantageous only if p is fairly large, considering its overhead. The use of FFT can nevertheless be of interest when the Fourier transform of K is known analytically and when many different values of h need to be considered in a choice of bandwidth procedure (see Silverman, 1982)

We have just described a simple binning scheme in density estimation context. It is easy to generalize this idea to kernel and local polynomial kernel estimation of regression function (see Fan and Marron, 1994). Some details will be given in next section. A different generalization consists in changing the way the data are binned. In "linear binning" one splits each X_i into two fractional weights associated with its two closest grid points. This scheme has been proposed in Jones and Lotwick (1983, 1984) and has been proved to be superior than the above simple binning, at least in the density estimation context. Later Pham (1995) introduced a general binning scheme and interpret it as a pre-smoothing followed by a discrete sampling.

In this paper, we will explore the above interpretation of binning in a general context and not restricted to density estimation. We will show that the general binning scheme introduced in Pham (1995) can be extended to regression problems and furthermore they are actually equivalent to a kernel estimator sampled at a equi-spaced grid, albeit with a somewhat different kernel. Thus there is no need to make a separate study of "binned" kernel estimator, as in Hall and Wand (1993) since the theory of kernel estimation is well known. Note however that in kernel estimation the choice of bandwidth parameter is quite important and the equivalent kernel associated with binning varies with it in a complicated way and not through simple rescaling as in the case without binning. Therefore it would be of interest to retain the structure of a family of kernels which are related through rescaling. In this respect we propose a binning scheme such that the equivalent kernel approaches closely $K(\cdot/h)/h$ for a given K and h. We shall discuss how such goal can be achieved. Thus, we regard a binned kernel estimator as an approximation to a given non binned one. This point of view is often adopted in the literature. Even if binned kernel estimator has been studied as a nonparametric estimator in its own right, it

is still usually in reference to a nominal kernel estimator.

2 A general binning scheme for kernel estimation

We shall consider both the density estimation and the nonlinear regression problems. For simplicity we consider only the univariate case although it is easy to generalize the procedure to multivariate data. In density estimation, one observes a sample $X_1, \ldots, X_n$ from a distribution on the real line admitting a density f. The kernel estimator of f is defined by (1.1). In nonlinear regression, one observes n pairs $(X_1, Y_1), \ldots, (X_n, Y_n)$ obeying the model

$$Y_i = m(X_i) + \epsilon_i$$

where $m : \mathbb{R} \mapsto \mathbb{R}$ is the regression function and ϵ_i are the residuals. Two situations can be considered: the fixed design case in which the $X_1, \ldots, X_n$ are fixed numbers and the ϵ_i are independent identically distributed (iid) random variables, and the random design case in which the pairs $(X_1, \epsilon_1), \ldots, (X_n, \epsilon_n)$ are iid random vectors with $\mathrm{E}(\epsilon_i | X_i) = 0$. In the fixed design case, we are interested only in the case where the $X_1, \ldots, X_n$ are not equi-spaced (otherwise there is no need of binning) and we assume that there sample distribution converge to some distribution having a density f, as $n \to \infty$. Then the arguments and results for the fixed design case will be no different than those for the random design case, interpreting f as the marginal density of the X_i.

The Nadaraya-Watson kernel estimator for $m(x)$ is

$$\hat{m}(x) = \Big[\frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K\Big(\frac{x - X_i}{h}\Big) Y_i\Big] \Big/ \hat{f}(x) \tag{2.1}$$

where $\hat{f}(x)$, K and h are as before. Note that the numerator can be viewed as an estimator of $m(x)f(x)$. Another method to estimate $m(x)$ is the kernel local polynomial regression. Restricting to polynomial of first order (linear), this estimator is given by (Fan and Marron, 1994)

$$\hat{m}(x) = \frac{S_2(x)T_0(x) - S_1(x)T_1(x)}{S_2(x)S_0(x) - S_1(x)^2} = \frac{T_0(x) - S_1(x)T_1(x)/S_2(x)}{S_0(x) - S_1(x)^2/S_2(x)} \tag{2.2}$$

where

$$S_l(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K\Big(\frac{x - X_i}{h}\Big)\Big(\frac{x - X_i}{h}\Big)^l, \qquad l = 0, 1, 2,$$

$$T_l(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K\Big(\frac{x - X_i}{h}\Big)\Big(\frac{x - X_i}{h}\Big)^l Y_i, \qquad l = 0, 1.$$

Note that $S_0(x)$ and $T_0(x)$ are precisely $\hat{f}(x)$ and the numerator of (2.1). Thus in this method, the estimators are still simple functions of certain statistics obtained through convolution with respect to some kernel. More precisely, let $K_{l,h}(x) = h^{-1}K(x/h)(x/h)^l$ and define μ_n, ν_n the discrete measures with masses $1/n$ and Y_i/n at the points X_i, $i = 1, \ldots, n$ (μ_n is in fact the sample distribution of the X_i), then one has

$$S_l = K_{l,h} * \mu_n, \qquad T_l = K_{l,h} * \nu_n, \tag{2.3}$$

$*$ denoting the convolution operator.

We now show that binning amounts to smoothing out μ_n and ν_n through convolution and then discretizing the result to obtain an approximating measure having masses (only) on a regular grid. To see this, consider the simple binning scheme described in the introduction. For definiteness, when a data point is exactly in the middle of a grid (i.e. equals $(j + \frac{1}{2})b$ for some j), we shall replace it by the grid point on the left (i.e. jb). Then it is easy to see that the ratio n_j/n in (1.2) is given by $\sum_{i=1}^{n} k(j - X_i/b)$, with k being the indicator of the interval $[-\frac{1}{2}, \frac{1}{2})$ (lower bound included, upper bound excluded). Thus, the sequence $n_j/n, j \in \mathbb{Z}$ can be viewed as resulting from two operations: a pre-smoothing in which measure μ_n is convolved with the kernel $k_b = k(\cdot/b)/b$ and a discretization in which the above convolution $k_b * \mu_n$ is replaced by the discrete sequence $b\ (k_b * \mu_n)(jb)$ associated with the grid points jb, $j \in \mathbb{Z}$. We shall denote by D_b the operator associated with the above discretization, which replaces a function by the sequence of b times its values at jb, $j \in \mathbb{Z}$. Here, the factor b has been introduced in order that for a function g, the measure associated with its discretized sequence $D_b g$, that is the one having masses at the grid points given by the values of the sequence, converges generally to the

measure with density g, as the grid size goes to zero. With this notation, the binned kernel estimator (1.2) of the introduction can be defined by

$$D_b \hat{f}^\dagger = D_b[K_{0,h} * D_b(k_b * \mu_n)].$$

We have written $D_b \hat{f}^\dagger$ to emphasize the fact that the binned estimator is always computed only at the grid points. The operator D_b does *not* commute with $*$ so that the order of operation in the above right hand side is important and thus parentheses and brackets have been introduced to avoid any confusion.

The above formula suggests a natural generalization of the binning scheme. Choose a bin-width b and a kernel k_b, then replace the statistics S_l, T_l by

$$S_l^\dagger = D_b[K_{l,h} * D_b(k_b * \mu_n)], \qquad T_l^\dagger = D_b[K_{l,h} * D_b(k_b * \nu_n)]. \tag{2.4}$$

Note that if we take k_b to be the triangular kernel defined by $k_b(x) = 1 - |x/b|$ if $|x/b| < 1$, $= 0$ otherwise, then the resulting binning scheme can be seen to be the same as the linear binning introduced by Jones and Lotwick (1983, 1984) and Fan and Marron (1994). But there is no reason, other than computational convenience and accuracy, which require us to use this kernel. We have taken k_b as a scaled version of some kernel k but in fact it is not even necessary, since we have the freedom of choice of k. The reason for this choice is that most popular kernels have support in $[-1, 1]$ so that k_b would have support in $[-b, b]$. This is important since for computational speed k_b should have support $[-b, b]$ or a small "multiple" of it and the use of a scaled kernel achieves this in an automatic way.

3 The equivalent kernel and approximation

Let us look again at the formula (2.4). The purpose of binning is to reduce the computation through discretization. But one can't discretize a discrete measure. One can only discretize a continuous function and the smoother it is the smaller the error. Therefore one has to smooth μ_n and ν_n through the convolution with k_b before discretization. The smoothing should be sufficient to minimize the discretization

error, but not too heavy to avoid distorting the data. As formula (2.4) shows, there is a second smoothing induced by the convolution with the kernel $K_{l,h}$. Thus, the smoothing introduced by the first, which we called the pre-smoothing, can be compensated for by the second (by using a smaller h for example). In fact, the two stages of smoothing can be combined and thus is equivalent to a single smoothing. To show this, we shall need the following result.

Lemma 3.1 *Let K, g be two continuous integrable functions on $\mathbb{R}$ and define D_b as above. Then*

$$D_b(K * D_b g) = D_b[(D_b K) * g]$$

Proof

By definition $(K * D_b g)(x) = \sum_{j=-\infty}^{\infty} K(x - jb) b g(jb)$. Hence, taking $x = lb$,

$$\begin{aligned}(K * D_b g)(lb) &= b \sum_{j=-\infty}^{\infty} K[(l-j)b] g(jb) \\ &= b \sum_{j=-\infty}^{\infty} K(jb) g[(l-j)b] \\ &= \sum_{j=-\infty}^{\infty} bK(jb) g(lb - jb).\end{aligned}$$

The last right hand side is precisely the function $(D_b K) * g$ evaluated at the point lb. The result follows. ∎

The above result shows that

$$S_l^\dagger = D_b[(D_b K_{l,h}) * k_b * \mu_n], \qquad T_l^\dagger = D_b[(D_b K_{l,h}) * k_b * \nu_n]$$

which appear as the convolution of μ_n and ν_n with respect to the kernel $(D_b K_{l,h}) * k_b$ followed by a discretization. Thus, apart from the final discretization, the whole process amounts to a smoothing through the equivalent kernel

$$K_{l,h}^b = (D_b K_{l,h}) * k_b \tag{3.1}$$

The theory of kernel estimation is well known and one can apply it to derive the properties for $S_l^\dagger$, $T_l^\dagger$, in the same way as for S_l, T_l by just replacing $K_{l,h}$ by the

above equivalent kernel. Note however that in the asymptotic setting, one lets h go to zero with n and the $K_{l,h}$ are scaled versions of a same kernel whereas the equivalent kernels $K^b_{l,h}$ depend on both h and b. Nevertheless, if b is taken to be a *fixed* fraction of h, then $K^b_{l,h}$ would depend only on h and it can be easily seen that the resulting family of kernels are scaled version of a single one. Since the ratio b/h can be anything, we conclude that the asymptotic formula for the mean and variance of $S^\dagger_l$, $T^\dagger_l$ still apply (using the equivalent kernel) as soon as $h \to 0$, $nh \to \infty$ and b/h remains bounded (the case $b/h \to 0$ is clearly admissible since then the estimator approaches the non-binned one)

At this point, it is worthwhile to ask what does the equivalent kernel $K^b_{l,h}$ look like. Clearly, its shape depends not only on $K_{l,h}$ but also on the binning kernel k_b. For the latter, we shall consider two cases: the rectangular kernel corresponding to simple binning and the triangular kernel, corresponding to linear binning. For k_b equal to $1/b$ times the indicator of the interval $[-\frac{b}{2}, \frac{b}{2})$, it is easily seen that $K^b_{l,h}$ is a step function taking the value $K_{l,h}(jb)$ in the interval $[(j-\frac{1}{2})b, (j+\frac{1}{2})b)$, $j \in \mathbb{Z}$. Thus $K^b_{l,h}$ is indeed an approximation to $K_{l,h}$, but a rather crude one. By the mean value Theorem, the approximation error can be seen to be $O(b/h)$ as $b/h \to 0$ provided that $K_{l,1}$ admits a bounded first derivative. For the triangular kernel: $k_b(x) = (1 - |x/b|)/b$ for $|x| < b$, $= 0$ otherwise, it can be easily seen that $K^b_{l,h}$ is a piecewise linear continuous function taking the value $K_{l,h}(jb)$ at the points jb, $j \in \mathbb{Z}$. Thus, $K^b_{l.h}$ is simply the linear interpolation of the function $K_{l,h}$, based on these points. This is clearly a better approximation to $K_{l,h}$: the theory of linear interpolation shows that the error is $O(b^2/h^2)$ as $b/h \to 0$, provided that $K_{l,1}$ admits a bounded second derivative.

From the above considerations, if one is not much concerned with kernel shape (which, as is well known, has little influence on the performance of the estimator), one might be satisfied with the binning scheme (2.4). We want to emphasize here that when computing the bias and variance of the estimator, one should use the equivalent kernel in place of the nominal kernel. More often though, one works

with a family of kernel depending on some band-with parameter which controls the smoothing. With binning, one has two parameters b and h which can both play this role. If one would like the kernel $K_{l,h}^b$ to be a scaled version of a common kernel, then b and h should vary in such a way that their ratio remains the same. But this would defeat the purpose of binning. For computational speed, the "binned data" $k_b * \mu_n$ and $k_b * \nu_n$ should be computed *only once* and efficiently. Thus b is kept fixed and the control of smoothing rests on h only. But then the family of equivalent kernels $K_{l,h}^b$ varies with h (for fixed b) in a somewhat different way than that of a family of scaled kernels. We don't think however that this has an appreciable effect on the choice of kernel, if b is much smaller that the lowest value of h that one might consider. Another point deserving mention is that if h is selected by the "plugged in" method in which the integrated mean squared errors of the estimator is estimated by the asymptotic formula, then there would be some extra complexities due to the unusual form of the equivalent kernel.

By the above reasons or by familiarity and simplicity, one might prefer to stick with the original family of scaled kernel $K_{l,h}$. Note that when binning is first invented, the purpose is to reduce the computations, not to introduce new estimators. Thus binned kernel estimator is often viewed as approximations to a kernel estimator and has been studied as such. Even if many authors have studied it as a nonparametric estimator in its own right, it is still in reference to a nominal kernel estimator. Therefore, it is of interest to design binning scheme such that the equivalent kernel approximates a given kernel. Before doing that, we note that there exists situations where exact equality can be achieved, at least for $l = 0$. For example, take h to be a multiple of b and let $K_{0,h}$ and k_b both be scaled versions of the rectangular kernel. Then it is seen from the above computations that $K_{0,h}^b$ is the same as $K_{0,h}$. The same result holds for $K_{0,h}$ and k_b being scaled versions of the triangular kernel. At this point, one may ask if one would ever take $K_{0,h}$ and k_b to be scaled version of a same function, since then why just simply "bin" the data with the bin-with $b = h$ and thus dispense with the subsequent discrete

convolution? There are two reasons for not doing so. Firstly, taking $b = h$ would mean that the estimate will be computed only at resolution h which can be far too coarse for later uses. Although one can compute $D_b(k_h * \mu_n)$ in the same way as one computes $D_b(k_b * \mu_n)$ since k_h is just another kernel similar to k_b, the computation is not efficient when the discretization interval b is much smaller than the support of the kernel k_h, as is the case here. Secondly, the optimal bandwidth is usually not known and has to be determined through a bandwidth selection procedure which requires computing the estimate with several different bandwidths. The advantage of the binning scheme described in section 2 is that one needs to computation of $D_b(k_b * \mu)$ once and estimators with different bandwidths h can all be obtained through discrete convolution. Note that the equivalent kernel would then coincide with the original one only if h is a multiple of b, but this is not a great restriction since b is usually small and one can only compute the estimator for a finite number of choices of band-width anyway.

The situations where the equivalent kernel coincides with the original one are rare. But it is the equivalent kernel that counts and this kernel depends on $K_{l,h}$ only through the sequence $bK_{l,h}(jb)$ which can be chosen freely and not necessarily comes from the discretization of the kernel $K_{l,h}$. Thus one may consider a general binning scheme in which instead of (2.4), $S_l^\dagger$ and $T_l^\dagger$ are computed as

$$S_l^\dagger = d_{l,h,b} * D_b(k_b * \mu_n), \qquad T_l^\dagger = d_{l,h,b} * D_b(k_b * \mu_n) \tag{3.2}$$

where $d_{l,h,b} = \sum_{j=-\infty}^{\infty} a_j \delta_{jb}$, $a_j = a_j(l,h,b)$, $j \in \mathbb{Z}$ being a sequence of real numbers only a finite number of which are non zero and δ_x denoting the Dirac measure centered at x. Then one can try to adjust the sequence a_j so that the equivalent kernel comes sufficiently close to some scaled kernel $K_{l,h}$. Now, using the same computations as in the proof of Lemma 3.1, one has, for any continuous function g,

$$\Big[\sum_{j=-\infty}^{\infty} a_j \delta_{jb}\Big] * D_b g = \sum_{r=-\infty}^{\infty} \Big[\sum_{j=-\infty}^{\infty} a_j g(rb - jb)\Big] b\delta_{rb} = D_b\Big[\sum_{j=-\infty}^{\infty} a_j g(\cdot - jb)\Big].$$

Thus, taking $g = k_b * \mu_n$ or $g = k_b * \mu_n$, it is seen that $S_l^\dagger$ and $T_l^\dagger$, as defined in (3.2)

equal respectively $D_b(K^b_{l,h} * \mu_n)$ and $D_b(K^b_{l,h} * \mu_n)$ with $K^b_{l,h}$ given by

$$K^b_{l,h} = \sum_{j=-\infty}^{\infty} a_j(l,h,b) k_b(\cdot - jb) = \Big[\sum_{j=-\infty}^{\infty} a_j(l,h,b)\delta_{jb} \Big] * k_b \tag{3.3}$$

The last equation defines the equivalent kernel, which should approximate $K_{l,h}$.

The problem of finding the coefficients a_j in (3.3) can be formulated as an approximation problem. Assuming for simplicity that k and K have support $[-1,1]$, one may impose the condition $a_j = 0$ for $|j| \geq h/b$, since k_b and $K_{l,h}$ have support $[-b,b]$ and $[-h,h]$. One may then try to minimized the L^2 distance from the equivalent kernel $K^b_{l,h}$ to the given kernel $K_{l,h}$. This makes sense since we know that the standard deviation of S_l and T_l are proportional to the L^2 norm of the kernel $K_{l,h}$. One is then led to a quadratic minimization problem with respect to the coefficients a_j. Since it is known that that the asymptotic mean and bias of the S_l and T_l are directly proportional to the total mass and the first non zero moment of K_l, one may further introduce the constraints that the equivalent kernel (3.3) admit the same total mass and first non zero moment. These constraints are linear in terms of the a_j and can be easily implemented.

The above procedure is however unnecessarily complex. In practice, b is generally much smaller than h and thus taking $a_j = K_{l,h}(jb)b$ in (3.3) already yields an equivalent kernel close to $K_{l,h}$. Therefore a small correction to those a_j may suffice for practical purpose. The correction we propose consists in re-normalizing the a_j and changing the smoothing parameter to $h' \neq h$. The reasoning behind this is that the binning stage introduces some smoothing, so one just smoothes a little less in the discrete convolution stage to correct the binning effect. To determine the right amount of smoothing, we look at the mean and bias of the binned estimator. Explicitly, we take

$$a_j = \frac{1}{C} K\Big(j\frac{b}{h'}\Big)\Big(j\frac{b}{h'}\Big)^l \frac{b}{h'} \tag{3.4}$$

and determine C, h' such that $\int K^b_{l,h}(x)x^r dx = \int K(x/h)(x/h)^l x^r dx/h$, for $r = 0$, 2 if l is even and $r = 1$, 3 if l is odd (we assume here that K is an even function).

Since k and hence k_b integrates to one, the above conditions amount to, for l even,

$$C = \Big[\sum_{j=-\infty}^{\infty} K\Big(j\frac{b}{h'}\Big)\Big(j\frac{b}{h'}\Big)^l \frac{b}{h'}\Big] \Big/ \Big[\int K(x)x^l dx\Big] \tag{3.5}$$

$$\frac{1}{C}\Big[\sum_{j=-\infty}^{\infty} K_l\Big(j\frac{b}{h'}\Big)\Big(j\frac{b}{h'}\Big)^l \frac{b}{h'}\Big(j^2 + \int k(x)x^2dx\Big)b^2\Big] = h^2 \int K(x)x^{l+2}dx.$$

Taking into account of (3.5), the last equation can be rewritten as

$$\Big(\frac{h'}{h}\Big)^2 = C\,\frac{\int K(x)x^{l+2}dx - (b/h)^2 \int K(x)x^l dx \int k(x)x^2dx}{\sum_j K(jb/h')(jb/h')^{l+2}(b/h')}. \tag{3.6}$$

In the case l is odd, we get the same set of equations but with l replaced by $l+1$.

Equations (3.5) – (3.6) are nonlinear with respect to C, h'. However, their right hand sides vary little when h' changes so that one can solve them by the fixed point iteration. One starts with $h' = h$, computes C and then h' by the right hand sides of (3.5) and (3.6) and repeats this process until convergence. In practice, one may be satisfied with only one iteration together with an extra computation for C, in order that $\int K^b_{l,h}(x)dx$, or $\int K^b_{l,h}(x)xdx$ in the case l odd, has the right value. Thus, one begins by "binning" the data with a small bin-width b. Then to obtain the kernel estimator with a kernel k and a band-width h, one computes h' by (3.5) and (3.6) with the h' in the right hand side set to h, then C by (3.5) with the new found h' and finally performs the discrete convolution of the "binned" data and the a_j, as given by (3.4). The computations in (3.5, (3.6) are quite fast since they involve mainly summations of a few terms. Note that h' and C, as defined, depend on l, but one may adopt a simpler procedure which makes the above correction only when computing $S_0^\dagger$ and $T_0^\dagger$ but keeps $S_1^\dagger$, $S_2^\dagger$, $T_1^\dagger$ unchanged, i.e. given by (2.4). (This amounts to taking $h' = h$ and $C = 1$.) Such procedure is justified by the fact that the terms $S_1(x)T_1(x)/S_2(x)$ and $S_1(x)^2/S_2(x)$ in the second right hand side of (2.2) are correction terms and thus need not be computed with great accuracy. Indeed, it can be easily shown that $S_1(x)$, $T_1(x)$ are $O(h)$ while $S_0(x)$, $T_0(x)$, $S_2(x)$, $T_2(x)$ are $O(1)$ as $h \to 0$, hence an error in $S_1(x)$, $S_2(x)$, $T_l(x)$ will be attenuated by a factor converging to zero at least as fast as h.

4 Some numerical results

We have made some numerical computations to assess the accuracy of the proposed correction. As smoothing kernel, we take the density of the family of symmetric β distributions on $[-1, 1]$, given by

$$K(x) = \frac{\Gamma(2\beta)}{\Gamma^2(\beta)\ 2^{2\beta-1}}(1-x^2)^{\beta-1} \quad \text{for } |x| < 1, \qquad = 0 \quad \text{else.}$$

This family contains a number of popular kernels: Epanechnikov ($\beta = 2$), bi-weight ($\beta = 3$) and tri-weight ($\beta = 4$). Gaussian kernel can also be approximated by taking β very large. However, we will do calculation for exact Gaussian kernel rather than using the above family. (In fact, as can be seen later, the tri-weight kernel already yields very similar results as the Gaussian kernel). The Gaussian kernel has infinite support so that some truncation is required. In our calculations, the truncation point is chosen such the kernel mass is at least .9999

For each of the above kernels and for different ratios h/b, we compute the following characteristics of the equivalent kernel: the total mass, the second moment, the squared L^2 norm. They are directly related to the asymptotic bias and variance of the (Nadaraya-Watson) estimator. For simplicity, we will report only the laters divided by the corresponding values of the referenced non binned estimator. The reason for using such relative characteristics is that they will not depend on the density and regression function but only on the kernel. Further, values close to 1 of such relative characteristics indicate good agreement between the equivalent and the reference kernel. As another measure of the agreement, we consider the L^2 distance between these two kernels, divided by the L^2 norm of the reference kernel. Note that these measures depends only on the ratio b/h, which controls the computational speed. Indeed, apart from the "binning" stage which needs to be done only once and is rather fast anyway, the discrete convolution needed for obtaining the estimator requires an amount of computation roughly proportional to h/b times the number of bins in the data range.

The results are reported in table 1. In this table the first row corresponds

h/b	2	3	4	5	6	8	10	14
	Epanechnikov kernel							
relative	0.9583	0.9815	0.9896	0.9933	0.9954	0.9974	0.9983	0.9991
bias	1.1333	1.0297	1.0110	1.0052	1.0029	1.0011	1.0006	1.0002
relative	0.9583	0.9815	0.9896	0.9933	0.9954	0.9974	0.9983	0.9991
variance	0.9542	0.9885	0.9955	0.9978	0.9987	0.9995	0.9997	0.9999
relative	0.0625	0.0278	0.0156	0.0100	0.0069	0.0039	0.0025	0.0013
distance	0.0462	0.0159	0.0081	0.0049	0.0033	0.0018	0.0011	0.0006
	bi-weight kernel							
relative	1.2181	1.1146	1.0681	1.0447	1.0314	1.0179	1.0115	1.0059
bias	1.1235	1.0315	1.0113	1.0050	1.0026	1.0009	1.0004	1.0001
relative	0.9040	0.9504	0.9707	0.9808	0.9865	0.9923	0.9950	0.9975
variance	0.9575	0.9940	0.9999	1.0010	1.0011	1.0009	1.0007	1.0004
relative	0.1085	0.0531	0.0308	0.0200	0.0140	0.0079	0.0051	0.0026
distance	0.0906	0.0416	0.0241	0.0157	0.0110	0.0063	0.0040	0.0021
	tri-weight kernel							
relative	1.4047	1.1772	1.0976	1.0617	1.0425	1.0237	1.0151	1.0077
bias	1.0257	1.0014	1.0001	1.0000	1.0000	1.0000	1.0000	1.0000
relative	0.8618	0.9329	0.9611	0.9747	0.9823	0.9900	0.9935	0.9967
variance	1.0620	1.0147	1.0073	1.0044	1.0030	1.0016	1.0010	1.0005
relative	0.1484	0.0646	0.0366	0.0235	0.0164	0.0093	0.0060	0.0030
distance	0.1709	0.0477	0.0246	0.0153	0.0104	0.0058	0.0037	0.0019
	Gaussian kernel							
h/b	2/3	1	4/3	5/3	2	8/3	10/3	14/3
relative	1.3685	1.1666	1.0933	1.0587	1.0413	1.0222	1.0142	1.0065
bias	0.9485	1.0001	1.0003	1.0005	1.0001	1.0003	1.0001	1.0001
relative	0.8682	0.9263	0.9563	0.9714	0.9798	0.9886	0.9926	0.9963
variance	1.0799	1.0010	0.9999	0.9995	0.9999	0.9995	0.9996	0.9995
relative	0.1252	0.0746	0.0431	0.0279	0.0195	0.0110	0.0071	0.0036
distance	0.1292	0.0432	0.0214	0.0129	0.0087	0.0047	0.0030	0.0016

Table 1: Relative asymptotic bias and variance of the binned estimator to the referenced estimator and the relative L^2 distance from the equivalent to the referenced kernel. Linear binning is used, without (1st row) and with (2nd row) correction.

to the linear binning without correction while the second corresponds to the same linear binning but with correction. The correction is as described in previous section with only *one* iteration step.

It can be seen from table 1 that the agreement between the equivalent and

the reference kernel is quite good in general. Correction, which is quite cheap, makes it even better. An exception is the case of the Epanechnikov kernel with $h/b = 2$. Here correction does not improve the agreement between the kernels, especially in term of bias and variance. We suspect that because the agreement is not good initially, a one-step correction won't be enough to improve it. It should be noted that the ratio $h/b = 2$ is very low which shouldn't be used in practice. This ratio corrresponds to a bin-width $b = 0.5\,h$, larger than the standard deviation of the Epanechnikov kernel scaled by h, which is $h/\sqrt{5} = 0.4472\,h$. Note further that for the bi-weight, tri-weight and Gaussian kernels, the agreement between the kernels is also not good at the lowest ratio h/b, but a one-step correction has been able to improve it significantly. We suspect that it is because these kernels are smoother and closer than the Gaussian kernel than the Epanechnikov kernel. Our correction is based on the renormalization and rescaling, and the only family of densities invariant with respect to these operation is the family of Gaussian densities. Indeed, table 1 shows that our correction works best with the Gaussian kernel. It should be noted that for this kernel we have used a ratio h/b one third of those for the β family of kernels. This is because the Gaussian kernel (with unit standard deviation) has a support roughly $[-3, 3]$ while the β family has support $[-1, 1]$. In fact, different kernels should be used with different values of band-width h. Numerical computations based on the asymptotic formula for bias and variance show that for a band-width h of the Gaussian kernel, one must use a band-width of $2.2138\,h$, $2.62262\,h$ and $2.97811\,h$ for the Epanechnikov, bi-weight and tri-weight kernels, respectively, in order to achieve roughly the same (i.e. up to the efficiency factor) asymptotic bias and variance.

The practical implication of the above numerical result is that if $h_{\min}$ denote the smallest band-width the (optimal) band-width selection might choose, then with a bin-width about $h_{\min}/3$ in the case of the Epanchinov kernel or $h_{\min}$ in the case of the Gaussian kernel, the corrected binning scheme, described at the end of section 3, yields quite good agreement between the equivalent kernel and the referenced kernel.

REFERENCES

[1] Fan, J. & Marron, J. S. (1994) Fast implementation of non parametric curve estimators. *J. Comp. Graphical Statist.*, **3**, 35–56.

[2] Hall, P. & Wand, M. P. (1993) On the accuracy of binned kernel density estimators. Working Paper 93–003, Australian Graduate School of Management, University of New South Wales.

[3] Jones, M. C. (1989) Discretized and interpolated kernel density estimates. *J. Amer. Statist. Assoc*, **84**, 733–741.

[4] Jones, M. C. (1992) Differences and derivatives in kernel estimation. *Metrika*, **39**, 335–340.

[5] Jones, M. C. & Lotwick, H. W. (1983) On the errors involving in computing the empirical characteristic function. *J. Statist. Comp. and Simul.* **13**, 173–149.

[6] Jones, M. C. & Lotwick, H. W. (1984) A remark on algorithm AS 176. Kernel density estimation using the fast Fourier transform. Remark AS R50, *Appl. Statist.* **33**, 120–122.

[7] Pham, D. T. (1995) On the discretisation error in the computation of the empiical characteristic function. To appear in *J. Statist. Comp. Simul.*

[8] Scott, D. W. & Sheather, S. J. (1985) Kernel density estimation with binned data. *Commun. Statist. – Theory Meth.*, **14**, 1353–1359.

[9] Silverman, B. W (1982) *Density Estimation for Statistics and Data Analysis*. London: Chapman and Hall.