ECON 950 — Winter 2020
Prof. James MacKinnon

5. Kernel Density Estimation

Two useful books are Li and Racine (2007) and Henderson and Parmeter (2015).

The simplest way to estimate a CDF graphically is to use the **empirical distribution function**, or **EDF**.

Suppose we have a sample x_i, $i = 1, \ldots, N$, of realizations of a random variable X. Then the EDF at any point x is

$$\hat{F}(x) \equiv \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(x_i \leq x), \tag{1}$$

where $\mathbb{I}(\cdot)$ is the **indicator function**. This is not a smooth function of x.

The traditional way to estimate a probability density function graphically is to form a **histogram**. This would be the right thing to do if the data were discrete.

The interval containing the x_i is partitioned into a set of subintervals by a set of points z_j, $j = 1, \ldots, M$, with $z_j < z_{j+1}$ for all j, where typically $M \ll N$.

Like the EDF, the histogram is a locally constant function with discontinuities. Unlike the EDF, the histogram is discontinuous at the z_j, not the x_i.

Let j be such that $z_j \leq x < z_{j+1}$ for some x. Then the histogram is just the following estimate of the density function at x:

$$\hat{f}(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{\mathbb{I}(z_j \leq x_i < z_{j+1})}{z_{j+1} - z_j} \,. \tag{2}$$

The value of the histogram at x is the proportion of the sample points contained in the same bin as x, divided by the length of the bin.

A histogram is extremely dependent on the choice of the partitioning points z_j.

With just two z_j, the histogram would look like a uniform distribution with lower limit z_1 and upper limit z_2.

With a great many z_j, many bins would be empty. The remaining bins would contain spikes, because $z_{j+1} - z_j$ would tend to 0 as the partition became finer.

We want neither too few nor too many bins. To prove anything about asymptotic validity, we would need a rule for increasing the number of bins as $N \to \infty$.

5.1. Kernel estimation of distribution functions

The discontinuous indicator function $\mathbb{I}(x_i \leq x)$ in (1) can be interpreted as the CDF of a degenerate random variable which puts all its probability mass on x_i.

The EDF can be thought of as the unweighted average of these CDFs.

We can obtain a smooth estimator of the CDF by replacing the discontinuous function $\mathbb{I}(x \geq x_i)$ in (1) by a continuous CDF that has support in an interval containing x_i. This will give us a weighted average.

Let $K(z)$ be any continuous CDF corresponding to a distribution with mean 0. This function is called a **cumulative kernel**. It usually corresponds to a distribution with a density that is symmetric around the origin, such as the standard normal.

In order to be able to control the degree of smoothness of the estimate, we set the variance of the distribution characterized by $K(z)$ to 1 and introduce the **bandwidth parameter** h as a scaling parameter.

This gives the **kernel CDF estimator**

$$\hat{F}_h(x) = \frac{1}{N} \sum_{i=1}^{N} K\Big(\frac{x_i - x}{h}\Big). \tag{3}$$

This estimator depends on the cumulative kernel $K(\cdot)$ and the bandwidth h.

As $h \to 0$, a typical term of the summation on the right-hand side of (3) tends to $\mathbb{I}(x_i \geq x) = \mathbb{I}(x \leq x_i)$, and so $\hat{F}_h(x)$ tends to the EDF $\hat{F}(x)$ as $h \to 0$.

At the other extreme, as h becomes large, a typical term of the summation tends to the constant value $K(0)$, which makes $\hat{F}_h(x)$ very much too smooth.

In the usual case in which $K(z)$ corresponds to a symmetric distribution, $\hat{F}_h(x)$ tends to 0.5 as $h \to \infty$.

It has been shown that $h = 1.587 s N^{-1/3}$ is optimal for CDF estimation, where s is the standard deviation of the x_i.

Here "optimal" means that we minimize the **asymptotic mean integrated squared error**, or **AMISE**; see below.

5.2. Kernel estimation of density functions

For density estimation, we can choose $K(z)$ to be not only continuous but also differentiable. Then we define the **kernel function**, often simply called the **kernel**, as $k(z) \equiv K'(z)$.

If we differentiate equation (3) with respect to x, we obtain the **kernel density estimator**

$$\hat{f}_h(x) = \frac{1}{Nh} \sum_{i=1}^{N} k\Big(\frac{x_i - x}{h}\Big). \tag{4}$$

Notice that we divide by Nh rather than just N.

Like the kernel CDF estimator (3), the kernel density estimator (4) depends on the choice of kernel $k(\cdot)$ and the bandwidth h.

One very popular choice for $k(\cdot)$ is the **Gaussian kernel**, which is just the standard normal density $\phi(\cdot)$. It gives a positive (although perhaps very small) weight to every point in the sample.

Another commonly used kernel, which has certain optimality properties, is the **Epanechnikov kernel**,

$$k_1(z) = \frac{3(1 - z^2/5)}{4\sqrt{5}} \quad \text{for } |z| < \sqrt{5}, \quad 0 \text{ otherwise.} \tag{5}$$

This kernel gives a positive weight only to points for which $|(x_i - x)|/h < \sqrt{5}$.

Yet another popular kernel is the **biweight kernel**:

$$k_2(z) = \frac{15}{16}(1 - z^2)^2 \mathbb{I}(|z| \leq 1). \tag{6}$$

This is quite similar to the Epanechnikov kernel, but it squares the argument and involves different constants.

Three properties shared by all these kernels, and other **second-order kernels**, are

$$\kappa_0(k) \equiv \int_{-\infty}^{\infty} k(z)dz = 1, \tag{7}$$

$$\kappa_1(k) \equiv \int_{-\infty}^{\infty} z\,k(z)dz = 0, \tag{8}$$

and

$$\kappa_2(k) \equiv \int_{-\infty}^{\infty} z^2 k(z) dz < \infty. \tag{9}$$

The first property is shared by all PDFs.

The second property is that the kernel has first moment zero. It is satisfied by any kernel that is symmetric about zero.

The third property is that the kernel has finite variance. It is essential for estimates based on the kernel k to have finite bias.

The big difference between the Epanechnikov and Gaussian kernels is that the former is 0 for $|z| > \sqrt{5}$, while the latter is always positive.

It can be shown that, to highest order, the bias of the kernel estimator is

$$\mathrm{E}\big(\hat{f}_h(x) - f(x)\big) \cong \frac{h^2}{2} f''(x) \kappa_2(k), \tag{10}$$

where $f''(x)$ is the second derivative of the density $f(x)$. Recall from (9) that $\kappa_2(k)$ is the second moment of the kernel k.

Notice that the bias does not depend directly on the sample size. It only depends on N through h, which should become smaller as N increases.

Since bias is proportional to h^2, it may seem that we should make h very small. But that turns out to be desirable only when N is very large.

The shape of the density matters. If the slope of the density is constant, then $f''(x) = 0$, and there is no bias.

It can also be shown that, to highest order, the variance of $\hat{f}_h(x)$ is

$$\mathrm{E}\big(\hat{f}_h(x) - f(x)\big)^2 \cong \frac{1}{Nh} f(x) R(k), \tag{11}$$

where

$$R(k) = \int k^2(z) dz \tag{12}$$

measures the "difficulty" of the kernel.

Note that the variance depends inversely on both the sample size and the bandwidth.

It makes sense that the variance goes up as h goes down, because fewer observations are averaged to give us the estimate for any x.

In choosing h, there is a tradeoff between bias and variance. A larger h increases bias but reduces variance.

Making h larger is like making k larger in kNN estimation.

The **asymptotic mean squared error**, or **AMSE**, is

$$\begin{aligned} \text{AMSE}\big(\hat{f}_h(x)\big) &= \text{Bias}^2\big(\hat{f}_h(x)\big) + \text{Var}\big(\hat{f}_h(x)\big) \\ &\cong \tfrac{1}{4}\kappa_2^2(k)\big(f''(x)\big)^2 h^4 + (Nh)^{-1} f(x) R(k). \end{aligned} \tag{13}$$

If we held h fixed as $N \to \infty$, the first term (bias squared) would stay constant, and the second term (variance) would go to zero.

Thus we want to make h smaller as N increases. But we need to ensure that $Nh \to \infty$ as $N \to \infty$ to make the second term go away.

The AMSE depends on x, so it will be different in different parts of the distribution.

There is no law requiring h to be the same for all x, although using more than one value risks causing visible artifacts where h changes.

If our objective is simply to draw a picture that looks nice and accurately portrays the true distribution, we may well want to use more than one value of h, but we will have to smooth out the artifacts.

To get an overall result, it is common to consider the **asymptotic mean integrated squared error**, or **AMISE**:

$$\begin{aligned} \text{AMISE}\big(\hat{f}_h(x)\big) &= \int_{-\infty}^{\infty} \text{AMSE}\big(\hat{f}_h(z)\big) dz \\ &\cong \tfrac{1}{4} h^4 \kappa_2^2(k) R(f'') + \frac{R(k)}{nh}, \end{aligned} \tag{14}$$

where $R(f'')$ measures the "roughness" of $f(x)$. Note that $R(f'')$ should not be confused with $R(k)$!

Larger values of $R(f'')$ imply that the density is harder to estimate.

AMISE involves the same tradeoff between bias and variance as AMSE, but it does not depend on x because we have integrated it out.

The Epanechnikov kernel is optimal, in the sense that it minimizes AMISE. The efficiency of some other kernel, say $k_g(\cdot)$, relative to $k_1(\cdot)$ is

$$\frac{R(k_g)\kappa_2(k_g)^{1/2}}{R(k_1)}. \tag{15}$$

The quantity $\kappa_2(k_1)$ does not appear here, because $\kappa_2(k_1) = 1$. The loss in efficiency relative to Epanechnikov is roughly 0.61% for biweight and 5.13% for Gaussian.

5.3. Bandwidth selection

The choice of bandwidth is far more important than the choice of kernel.

The optimal bandwidth for minimizing AMSE is

$$h_{\text{opt}} = N^{-1/5}\left(\frac{f(x)R(k)}{\kappa_2^2(k)\big(f''(x)\big)^2}\right)^{1/5}. \tag{16}$$

and the optimal bandwidth for minimizing AMISE is

$$h_{\text{opt}} = N^{-1/5}\left(\frac{R(k)}{\kappa_2^2(k)R(f'')}\right)^{1/5}. \tag{17}$$

These results make it clear that h should get smaller as N gets larger, but quite slowly. For example,

$$\begin{aligned} N = 10 &\longrightarrow h \propto 0.63096 \\ N = 100 &\longrightarrow h \propto 0.39811 \\ N = 1000 &\longrightarrow h \propto 0.25119 \\ N = 10{,}000 &\longrightarrow h \propto 0.15849 \\ N = 100{,}000 &\longrightarrow h \propto 0.10000 \end{aligned}$$

Here N increases by a factor of 10,000, and h shrinks by a factor of just 6.3.

For any density and any kernel, one can figure out $R(k)$, $\kappa_2(k)$, and $R(f'')$. In the case of the Gaussian kernel and a normal density, these are

$$R(k) = (2\sqrt{\pi})^{-1}, \quad \kappa_2(k) = 1, \quad \text{and} \quad R(f'') = \frac{2}{8\sqrt{\pi}\sigma^5}. \tag{18}$$

Substituting these into expression (17) for h_{opt}, we find that

$$h_{\text{opt}} = \left(\frac{8\sqrt{\pi}\sigma^5}{6\sqrt{\pi}}\right)^{1/5} N^{-1/5} = \left(\frac{4}{3}\right)^{1/5} \sigma N^{-1/5} \cong 1.059\sigma N^{-1/5}. \tag{19}$$

This leads to **Silverman's rule-of-thumb bandwidth**:

$$h_{\text{rot}} = 1.059 s N^{-1/5}, \tag{20}$$

where s is the sample standard deviation.

When a distribution has heavy tails, s is not a very good measure of dispersion.

A more robust measure is the inter-quartile range. For a normal distribution, $\sigma = \text{IQR}/1.349$. Thus we can either replace s in the rule-of-thumb bandwidth by IQR/1.349, or (to avoid over-smoothing) replace it by $\min(s, \text{IQR}/1.349)$.

For the Epanechnikov kernel, the constant that corresponds to 1.059 is 1.049. It is a little bit smaller because of the greater efficiency of estimation based on the Epanechnikov kernel.

Of course, any rule-of-thumb bandwidth may fail badly if the distribution being estimated differs a lot from the normal. There are likely to be severe problems if the distribution is multi-modal.

We evidently need a way to estimate $R(f'')$.

H&P (2015) discuss various plug-in methods. All of these essentially require that we obtain a kernel estimate of $R(f'')$, which is then used to estimate the optimal bandwidth.

But estimating $R(f'')$ requires a bandwidth parameter, and estimating it requires another bandwidth parameter, and so on!

5.4. Some numerical examples

Figure 1 shows kernel density estimates for CRVE t statistics based on 5000 replications using a Gaussian kernel.

We would expect these t statistics to be more or less symmetrically distributed, but with variance greater than 1 and quite possibly kurtosis greater than $3\sigma^4$.

There are three estimated densities, using

$$h_{\text{rot}} = 1.059\, N^{-1/5}$$
$$h_{\text{big}} = 1.5 \times 1.059\, N^{-1/5}$$
$$h_{\text{small}} = 0.5 \times 1.059\, N^{-1/5}$$

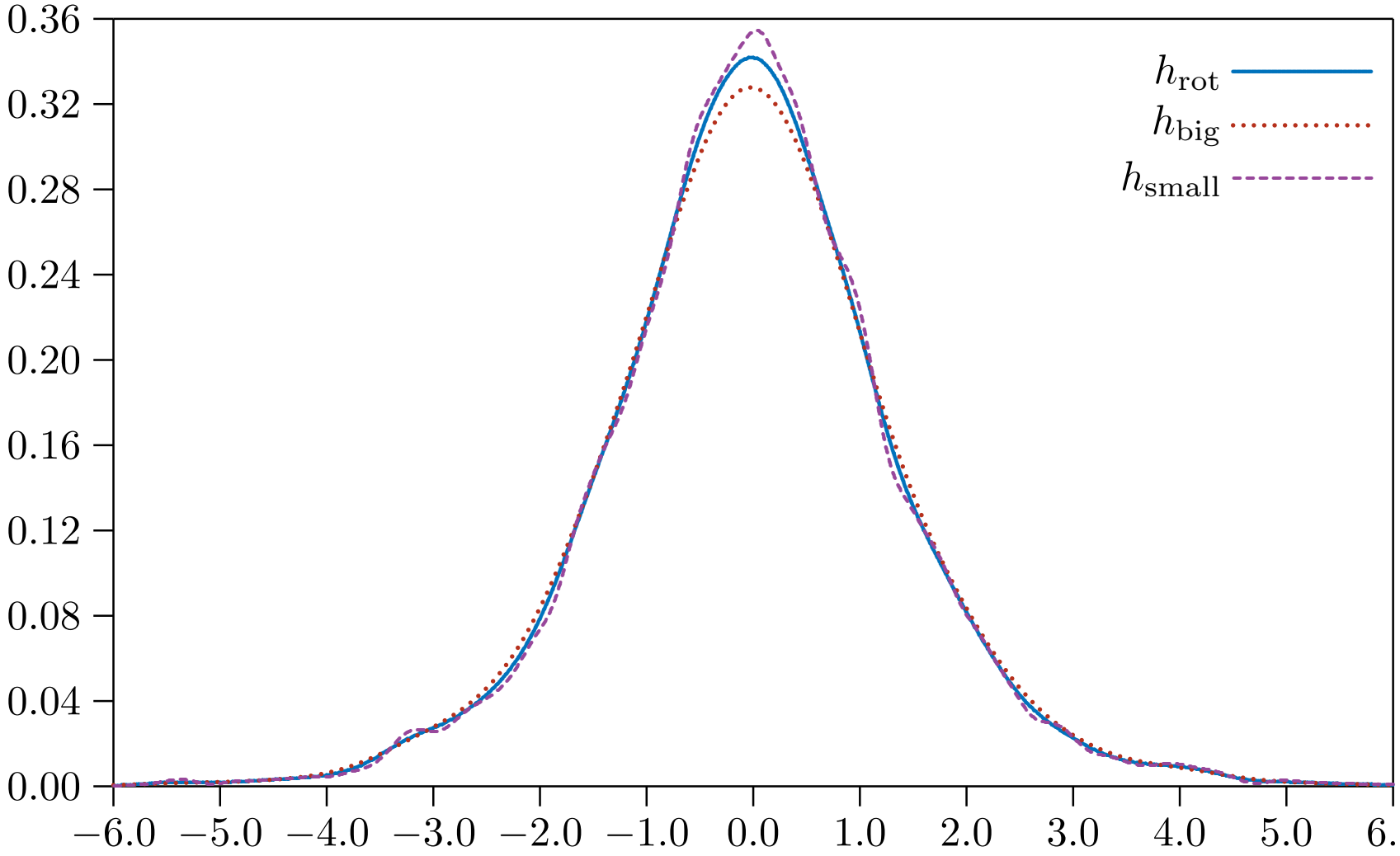

Figure 1. Kernel density estimates for CRVE t statistics

The figure is plotted for 601 values of t evenly spaced between -6.0 and 6.0.

It is clear from the figure that h_{small} is too small, because there are lots of wiggles.

It is not so obvious that h_{big} is too big.

All three estimates give us a pretty good idea of what the density looks like. h_{big} gives the nicest picture, but perhaps the peak is too low.

Figure 2 is similar, but it graphs the density of 4999 bootstrap (t^*) statistics. It seems even more obvious that h_{small} is too small.

In this case, we can generate as many observations as we like. With N sufficiently large, we should get essentially the same figure for any sensible value of h.

With real data, on the other hand, N may be too small to yield reliable estimates.

We can often obtain a density that looks nice by making h too large, but it may deviate a lot from the truth, especially in the tails and near the peak.

If multi-modal densities are possible and interesting, we do not want to make the bandwidth so large that the second mode disappears.

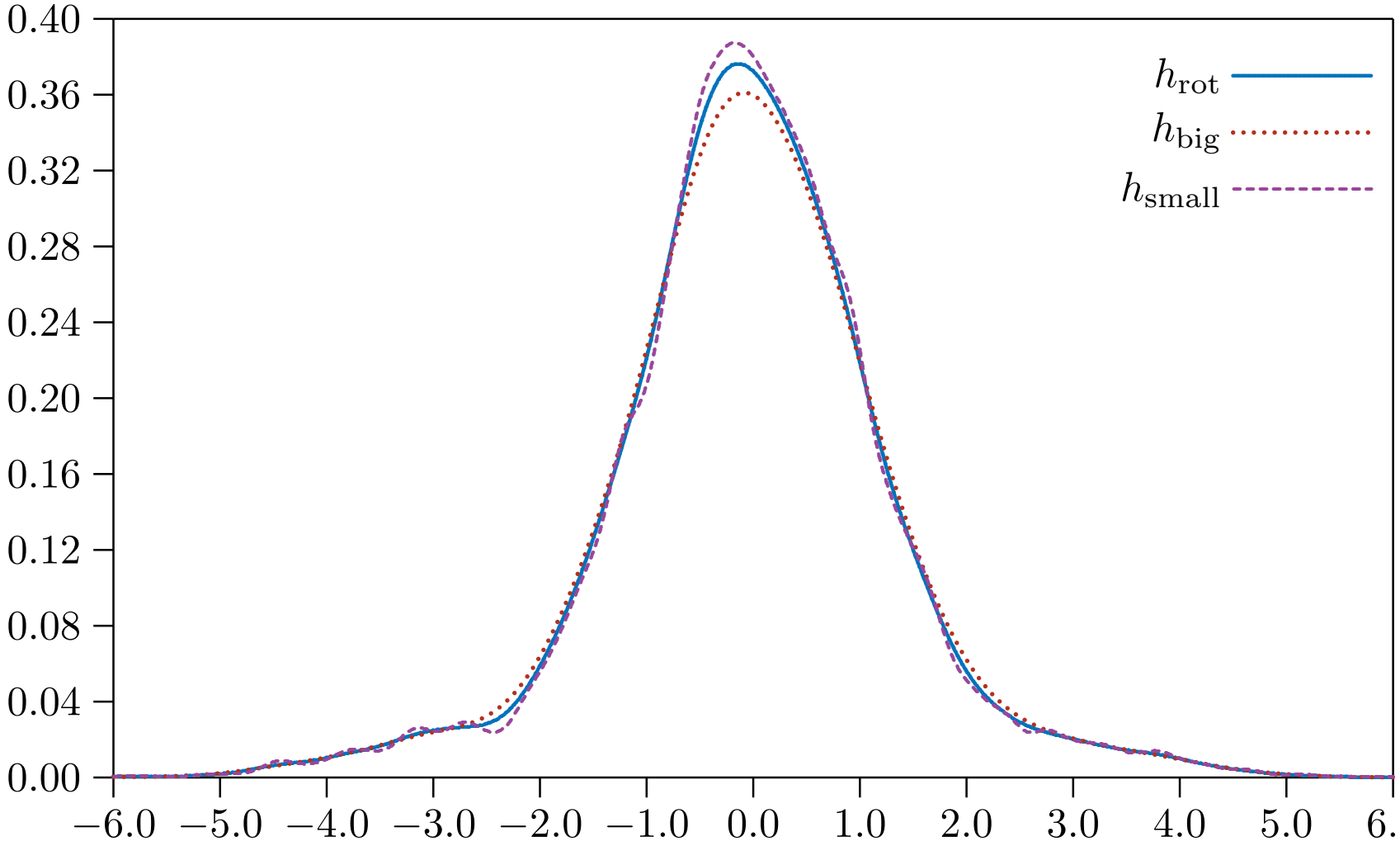

Figure 2. Kernel density estimates for bootstrap CRVE t^* statistics

5.5. Kernel regression

The simplest approach to nonparametric regression is **kernel regression**.

Suppose that two random variables Y and X are jointly distributed, and we wish to estimate the conditional expectation $\mu(x) \equiv \mathrm{E}(Y \mid x)$ as a function of x, using a sample of paired observations (y_i, x_i) for $i = 1, \ldots, n$.

For given x, consider the function $G(x)$ defined as

$$G(x) = \mathrm{E}\big(Y\,\mathbb{I}(X \leq x)\big) = \int_{-\infty}^{x} \int_{-\infty}^{\infty} y\, f(y, z)\, dy\, dz, \tag{21}$$

where $f(y, x)$ is the joint density of Y and X. Let $g(x) \equiv G'(x)$ denote the first derivative of $G(X)$.

A natural unbiased estimator of $G(x)$ is $\frac{1}{N} \sum_{i=1}^{N} y_i \mathbb{I}(x_i \leq x)$.

But this estimator, like the EDF, is discontinuous. We need to replace the indicator function by something smoother if we are to estimate the derivative of $G(x)$.

The simplest approach is to replace $\mathbb{I}(x_i \leq x)$ by a cumulative kernel. Thus we obtain the biased but smooth estimator

$$\hat{G}_h(x) = \frac{1}{N} \sum_{i=1}^{N} y_i \, K\Big(\frac{x - x_i}{h}\Big), \tag{22}$$

where K is a cumulative kernel (that is, the CDF of a distribution with mean 0 and variance 1), and h is a bandwidth parameter.

In order to obtain a kernel regression, we need to find the derivative of (22), say $\hat{g}_h(x)$, and estimate the marginal density of X.

This yields the **Nadaraya-Watson**, or **locally constant**, estimator

$$\hat{\mu}_h(x) = \frac{\sum_{i=1}^{N} y_i k_i(x)}{\sum_{i=1}^{N} k_i(x)}, \quad k_i(x) \equiv k\Big(\frac{x - x_i}{h}\Big), \tag{23}$$

where $k \equiv K'$ is a kernel function.

The numerator of (23) is a weighted average of the values of y_i in the neighborhood of x, and the denominator is a kernel estimate of the density of X at the point x.

The Nadaraya-Watson estimator is the solution to the estimating equation

$$\sum_{i=1}^{N} k_i(x)\big(y_i - \hat{\mu}_h(x)\big) = 0. \tag{24}$$

This is the empirical counterpart of a weighted average of the $y_i - \mu(x)$.

But the conditional expectation of y_i is not $\mu(x)$ but $\mu(x_i)$. This causes bias.

A better approximation is the two-term Taylor expansion $\mu(x) + \mu'(x)(x_i - x)$, in which both $\mu(x)$ and $\mu'(x)$ are unknown.

Both of these unknowns can be estimated simultaneously by solving the estimating equations

$$\sum_{i=1}^{N} k_i(x)\big(y_i - \mu(x) - \mu'(x)(x_i - x)\big) = 0 \tag{25}$$

and

$$\sum_{i=1}^{N} k_i(x)(x_i - x)\big(y_i - \mu(x) - \mu'(x)(x_i - x)\big) = 0. \tag{26}$$

The simplest way to solve these equations is to run the linear regression

$$k_i^{1/2}(x)\,y_i = \mu(x)k_i^{1/2}(x) + \mu'(x)(x_i - x)k_i^{1/2}(x) + \text{residual}, \tag{27}$$

so as to obtain the **locally linear estimator** of $\mu(x)$, which is just the first estimated coefficient, say $\hat{\mu}_h^{\text{LL}}$. Regression (27) is called a **local(ly) linear regression**.

We must run regression (27) for every value of x at which we wish to evaluate $\mu(x)$. How many observations it involves depends on the kernel and the bandwidth.

For a Gaussian kernel, regression (27) always has N observations. For an Epanechnikov kernel, it typically has a smaller (perhaps much smaller) number.

Observations near x get large weights, and observations far away get small weights. With kernels such as Epanechnikov, the latter actually get zero weights.

We could add additional terms, such as

$$\mu''(x)k_i^{1/2}(x)(x_i - x)^2, \tag{28}$$

to regression (27). This would give us a **locally quadratic estimator**. More generally, we would have a **locally polynomial estimator**.

It can be shown that, to second order, the bias of the locally constant estimator is

$$\begin{aligned} &\frac{\kappa_2(k)}{2f(x)}h^2\big(2\mu'(x)f'(x)+\mu''(x)f(x)\big) \\ &\quad=\frac{1}{2}\kappa_2(k)h^2\mu''(x)+\kappa_2(k)h^2\mu'(x)\frac{f'(x)}{f(x)}, \end{aligned} \tag{29}$$

where $\mu'(x)$ and $\mu''(x)$ denote the first and second derivatives of the conditional mean function evaluated at x.

The first term depends on the second derivative of $\mu(x)$, and the second term depends on the first derivative. So there will be no bias if $\mu(x)$ is a horizontal line.

Recall that $f(x)$ is the density of the x_i at x, and $f'(x)$ is its first derivative. Also, recall from (9) that $\kappa_2(k)$ is the second moment of the kernel k.

Thus bias will be larger for kernels with larger variance and when the density of x is changing more rapidly.

Similarly, to second order, the bias of the locally linear estimator is

$$\text{Bias}(\hat{\mu}_h^{\text{LL}})=\frac{1}{2}\kappa_2(k)h^2\mu''(x). \tag{30}$$

The bias of the LL estimator, expression (30), is equal to the first term of the bias of the LC estimator in the second line of (29).

The second term in (29) depends on $\mu'(x)$, but the common term depends only on $\mu''(x)$. So, as we might expect, the LL estimator is unbiased if $\mu(x)$ is linear.

To highest order, the variance of both estimators is the same:

$$\mathrm{Var}(\hat{\mu}_h^{\mathrm{LC}}) = \mathrm{Var}(\hat{\mu}_h^{\mathrm{LL}}) \cong \frac{\sigma^2 R(k)}{Nhf(x)}, \tag{31}$$

where σ^2 is the variance of the regression disturbances. Unlike the bias, this does not depend on the regression function we are estimating.

It is possible to find an optimal bandwidth by minimizing AMISE, but the result depends on:

- the sample size, with a factor of $N^{-1/5}$;
- the density of x, the regressor;
- the variance of the disturbances;

- the shape of the regression function; and
- the kernel.

In practice, people generally do not attempt to estimate the optimal bandwidth.

Instead, they typically use leave-one-out cross-validation. For the locally constant (LC) estimator, we choose h to minimize

$$\text{LSCV}(h) = \sum_{i=1}^{N} \big(y_i - \hat{\mu}_{-i}(x_i)\big)^2, \tag{32}$$

where

$$\hat{\mu}_{-i}(x_i) = \frac{\sum_{j \neq i}^{N} y_j k_h(x_j, x_i)}{\sum_{j \neq i}^{N} k_h(x_j, x_i)}. \tag{33}$$

For each i, this is just the kernel-weighted average of all the y_j for $j \neq i$.

5.6. A numerical example

I generated 400 observations from an artificial DGP that is linear for x_i below a certain value and quite nonlinear beyond that point.

I used an Epanechnikov kernel, with either a default bandwidth of $h = sN^{-1/5}$ or a value of h chosen by cross-validation.

The LC estimates totally miss the rightmost data points, and also perform poorly for the leftmost ones.

The reason is obvious: For the rightmost points, LC is taking a weighted average of points that are (almost) all to the left of them.

To avoid this, LSCV makes h quite small, which causes the fitted values to wiggle.

LL estimates are much more plausible than LC ones. h chosen by cross-validation is smaller than baseline value, but not much difference between two sets of estimates.

Values of the cross-validation function:

LC: baseline h (0.9803): 2.8525 optimal h (0.2498): 2.4960
LL: baseline h (0.9803): 2.4947 optimal h (0.6920): 2.4693

It is not a coincidence that the optimal bandwidth is much larger for LL regression than for LC regression.

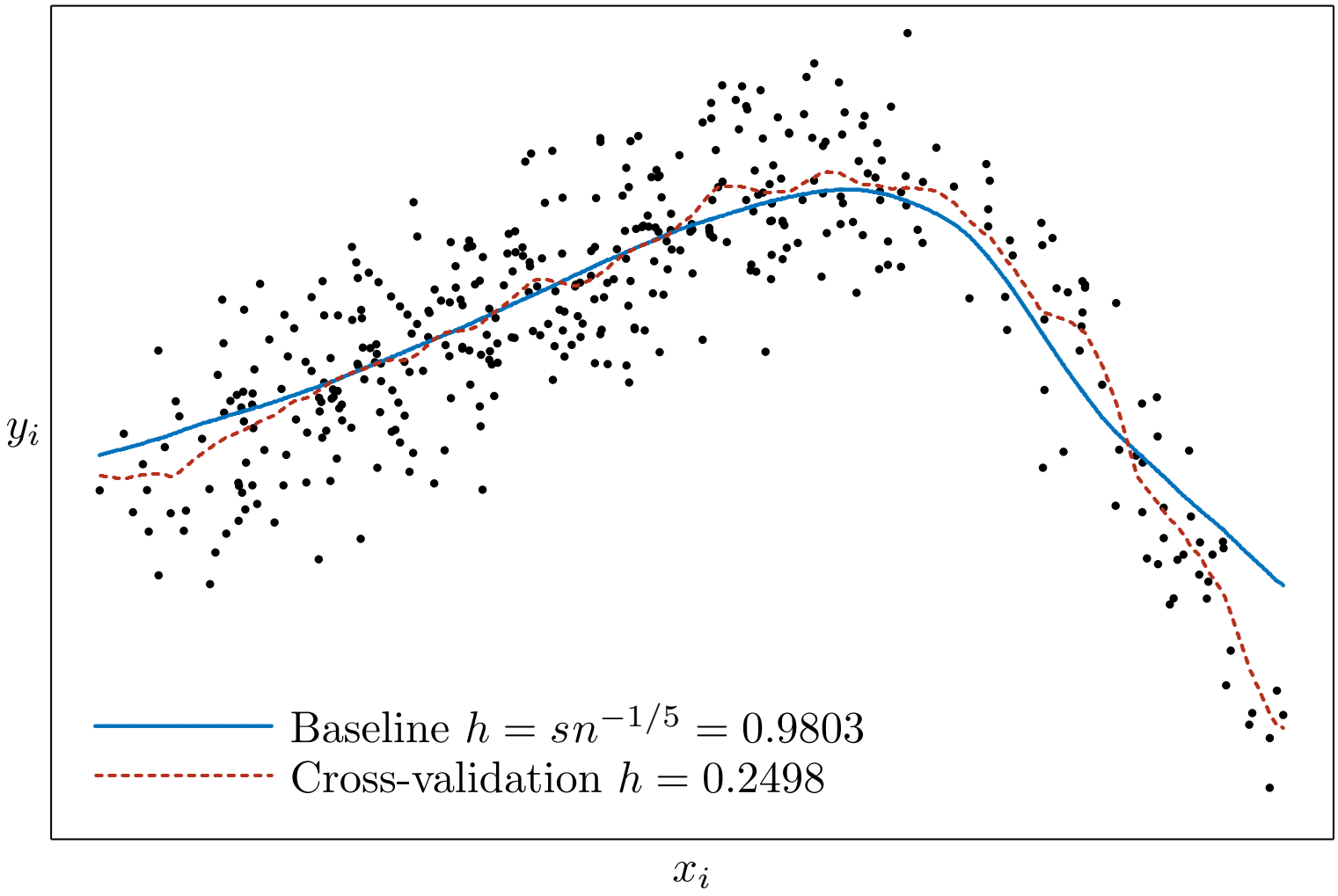

Figure 3. Locally constant kernel regression using simulated data

The choice of h involves a tradeoff between bias and variance. We saw in (31) that the variance of the two estimators is similar and declines with h.

We also saw that the bias of LC, in (29), is larger than the bias of LL, in (30). These both increase with h.

Therefore, the tradeoff favours making h larger for LL than for LC.

We can afford to make the bias term larger by making h larger when the bias term is smaller to begin with.

5.7. More on Kernels

We previously defined the Epanechnikov kernel as

$$k_1(z) = \frac{3(1 - z^2/5)}{4\sqrt{5}} \quad \text{for } |z| < \sqrt{5}, \;\; 0 \;\; \text{otherwise.} \tag{34}$$

ESL define it as

$$D(t) = \frac{3}{4}(1 - t^2) \quad \text{for } |t| < 1, \;\; 0 \;\; \text{otherwise.} \tag{35}$$

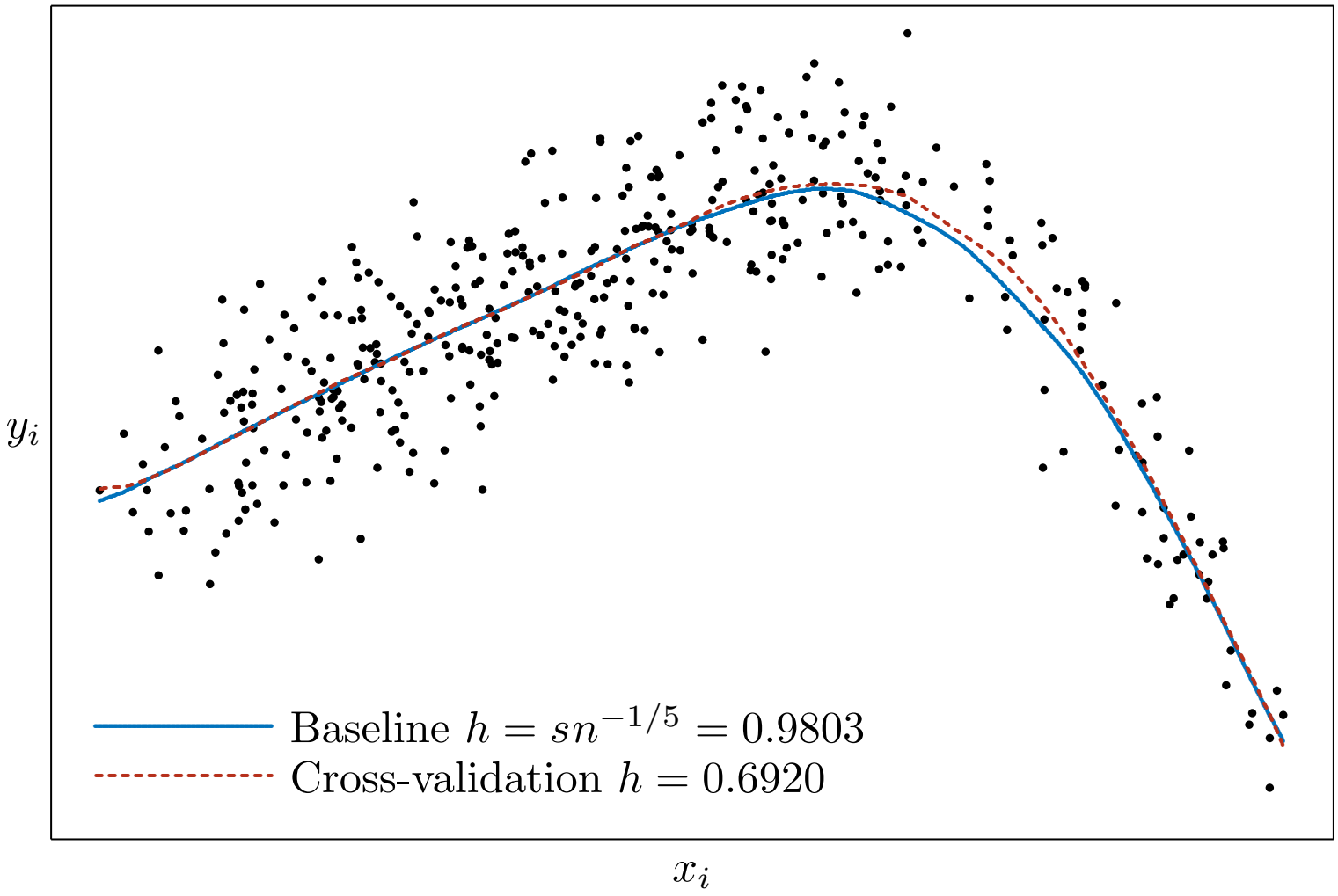

Figure 4. Locally linear kernel regression using simulated data

Thus $t = z/\sqrt{5}$ and $k_1(z) = D(t)/\sqrt{5}$.

We can obviously choose bandwidths so that kernel regression based on (34) and (35) are identical.

ESL define

$$K_\lambda(x_0, x) = D\left(\frac{|x - x_0|}{\lambda}\right), \tag{36}$$

whereas we used

$$k_1\left(\frac{x - x_0}{h}\right). \tag{37}$$

It seems odd that ESL use absolute values in (36) when the argument of $D(t)$ is squared.

A kernel that looks a lot like the Epanechnikov kernel but is differentiable is the **tri-cube kernel**

$$D(x) = \left(1 + |t^3|\right)^3 \quad \text{for } |t| < 1, \;\; 0 \;\; \text{otherwise.} \tag{38}$$

5.8. Local Regression in Higher Dimensions

It is easy to generalize Nadaraya-Watson kernel regression and locally linear or quadratic regression to more than one dimension, although it may not be a good idea for $p > 2$.

For example, we might have

$$\boldsymbol{b}(x) = [1 \;\; x_1 \;\; x_2]^\top, \tag{39}$$

for locally linear regression in two dimensions, or

$$\boldsymbol{b}(x) = [1 \;\; x_1 \;\; x_2 \;\; x_1^2 \;\; x_2^2 \;\; x_1 x_2]^\top, \tag{40}$$

for locally quadratic regression in two dimensions. We minimize

$$\sum_{i=1}^{N} K_\lambda(\boldsymbol{x}_0, \boldsymbol{x}_i)\big(y_i - \boldsymbol{b}^\top(\boldsymbol{x}_i)\boldsymbol{\beta}(\boldsymbol{x}_0)\big)^2 \tag{41}$$

and obtain the fitted values $\hat{f}(\boldsymbol{x}_0) = \boldsymbol{b}^\top(\boldsymbol{x}_0)\hat{\boldsymbol{\beta}}(\boldsymbol{x}_0)$.

The kernel is usually a radial Epanechnikov or tri-cube:

$$K_\lambda(\boldsymbol{x}_0, \boldsymbol{x}) = D\left(\frac{||\boldsymbol{x} - \boldsymbol{x}_0||}{\lambda}\right), \tag{42}$$

where the predictors should usually be standardized.

It is impossible to maintain both low bias and low variance, unless the sample has a great many points near every interesting value of $\boldsymbol{x}_0$. This is extremely difficult to achieve unless N is very large.

For bias to be small, we need all the points that get much weight to be near $\boldsymbol{x}_0$, which implies that λ must be small.

For the variance to be small, we need there to be a lot of points that are near $\boldsymbol{x}_0$, which implies that λ must be large.

The only way that λ can be small enough for low bias and large enough for low variance is if N increases exponentially in p.

5.9. Structured Local Regression Models

Unless p is very small, we need to impose structure on the model.

For example, we can use a **structured regression function** such as

$$f(\boldsymbol{x}) = \alpha + \sum_{j=1}^{p} g_j(x_j) + \sum_{k<\ell} g_{k\ell}(x_k, x_\ell) + \dots \tag{43}$$

This generalizes the partially linear regression model. Typically, there cannot be too many higher-order terms. For **additive models**, there are just the $g_j(x_j)$.

Instead of a nonlinear function for one variable and a linear model for all the others, (43) has many one-dimensional and two-dimensional nonlinear models to estimate.

This can be done iteratively. Consider the additive case. If we centre the data and all the $g_j(x_j)$ except $g_k(x_k)$ are assumed known, we can estimate an additive model by repeatedly running the local regression

$$y - \sum_{j \neq k} g_j(x_j) = g_k(x_k) + \text{resid}. \tag{44}$$

We cycle through j from 1 to p until convergence. We may have to estimate a lot of local regressions, but each one is just one-dimensional.

Another type of model is the **varying coefficient model**. Let z denote x_p and define $q \equiv p - 1$. Then consider the model

$$f(\boldsymbol{x}) = \beta_0(z) + \beta_1(z)x_1 + \ldots + \beta_q(z)x_q, \tag{45}$$

where there are now p nonlinear functions to estimate. Conditional on them, we simply have a linear regression model.

It can be fitted by locally weighted least squares. We minimize

$$\sum_{i=1}^{N} K_\lambda(z_0, z_i)\big(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}(z_0)\big)^2 \tag{46}$$

with respect to the vector $\boldsymbol{\beta}(z_0)$ for each value of z_0.

5.10. Local Likelihood

Any parametric model can be converted to a local one by using weights that vary across observations according to the value of $\boldsymbol{x}$.

In particular, it is easy to turn globally linear models into locally linear ones.

Suppose the model has a loglikelihood function

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{N} \ell(y_i, \boldsymbol{x}_i^\top \boldsymbol{\beta}). \tag{47}$$

An obvious example is a logit or probit model. Then we can estimate a locally linear version by maximizing

$$\ell\big(\boldsymbol{\beta}(\boldsymbol{x}_0)\big) = \sum_{i=1}^{N} K_\lambda(\boldsymbol{x}_0, \boldsymbol{x}_i) \ell\big(y_i, \boldsymbol{x}_i^\top \boldsymbol{\beta}(\boldsymbol{x}_0)\big). \tag{48}$$

Here we weight contributions to the loglikelihood instead of squared residuals.

We could also estimate a model with varying coefficients by maximizing

$$\ell\big(\boldsymbol{\theta}(z_0)\big) = \sum_{i=1}^{N} K_\lambda(z_0, z_i)\ell\big(y_i, \boldsymbol{x}_i^\top \boldsymbol{\theta}(z_0)\big) \tag{49}$$

with respect to the vector $\boldsymbol{\theta}(z_0)$ for each value of z_0; compare (46).

Consider the multiple logit model with J responses, where

$$\Pr(G = j \,|\, \boldsymbol{x}) = \frac{\exp(\beta_{j0} + \boldsymbol{x}^\top \boldsymbol{\beta}_j)}{1 + \sum_{k=1}^{J-1} \exp(\beta_{k0} + \boldsymbol{x}^\top \boldsymbol{\beta}_k)}, \tag{50}$$

where $\beta_{J0} = 0$ and $\boldsymbol{\beta}_J = \mathbf{0}$.

The local loglikelihood for this model is

$$\begin{aligned} &\sum_{i=1}^{N} K_\lambda(\boldsymbol{x}_0, \boldsymbol{x}_i) \Bigg(\beta_{g_i 0}(\boldsymbol{x}_0) + (\boldsymbol{x}_i - \boldsymbol{x}_0)^\top \boldsymbol{\beta}_{g_i}(\boldsymbol{x}_0) \\ &- \log\Bigg(1 + \sum_{k=1}^{J-1} \exp\left(\beta_{k0}(\boldsymbol{x}_0) + (\boldsymbol{x}_i - \boldsymbol{x}_0)^\top \boldsymbol{\beta}_{g_i}(\boldsymbol{x}_0)\right)\Bigg)\Bigg). \end{aligned} \tag{51}$$

Because the regressions are centred at $\boldsymbol{x}_0$, the posterior probabilities at $\boldsymbol{x}_0$ are simply

$$\widehat{\Pr}(G = j \,|\, \boldsymbol{x}_0) = \frac{\exp\big(\hat{\beta}_{j0}(\boldsymbol{x}_0)\big)}{1 + \sum_{k=1}^{J-1} \exp\big(\hat{\beta}_{k0}(\boldsymbol{x}_0)\big)}. \tag{52}$$

They do not depend on the vectors $\hat{\boldsymbol{\beta}}_j(\boldsymbol{x}_0)$.

Since $\widehat{\Pr}(G = j \,|\, \boldsymbol{x}_0)$ just depends on $\boldsymbol{x}_0$ and the coefficients $\hat{\beta}_{j0}$, $j = 1, J-1$, we can calculate its standard error using the delta method.

This model can be used for classification in low dimensions.