ECO 523 - Class 2

Nonparametric Econometrics

Carolina Caetano*

Contents

1 Bandwidth Choice

The choice of the bandwidth is fundamental. Particularly when we really are interested in $g(x)$ for a specific x. When we are aggregating the $g(x)$ it becomes somewhat less important, because the aggregation cancels out a lot of the bias, and divides the variance by n. We saw how the behavior of the bandwidth as the sample size increases affects the asymptotic bias and variance. In finite samples, however, the theorem provides no guidance for the choice of h. We would like to choose the best possible bandwidth at the available sample size. An agreed upon strategy is to minimize some form of squared errors.

*Special thanks to Tiago Tavares for help with plots and editing.

1.1 Silverman's rule-of-thumb and the plug-in method

This strategy aims at minimizing the weighted integrated squared errors. The weight is necessary because the terms in question are often not integrable. Hence,

$$\begin{aligned} WIMSE &= \min_{h_1,\ldots,h_q} \int \left[Bias(\hat{g}(x))^2 + Var(\hat{g}(x))\right] \nu(x)dx \\ &= \min_{h_1,\ldots,h_q} \int \left\{ \left[\sum_{s=1}^{q} h_s^2 B_s(x) \left[\frac{1}{f(x)}\frac{\partial f(x)}{\partial x_s}\frac{\partial g(x)}{\partial x_s} + \frac{1}{2}\frac{\partial^2 g(x)}{\partial x_s^2}\right]\right]^2 \right. \\ &\qquad \left. + \frac{1}{nh_1\ldots h_q}\frac{\sigma^2(x)}{f(x)}\left(\int k(v)^2 dv\right)^q \right\} \nu(x)dx \end{aligned}$$

where $WIMSE$ stands for weighted integrated mean square error. Observe that even when we would like to estimate only $g(x)$ for a particular x, the only way we can deal with the bandwidth issue is by looking at how would we act in the case of estimation at all points.

The trick is to make $h_1 = a_1 h, \ldots, h_q = a_q h$:

$$\begin{aligned} WIMSE &= \min_{h_1,\ldots,h_q} \int \left\{ h^4 \left[\sum_{s=1}^{q} a_s^2 B_s(x) \left[\frac{1}{f(x)}\frac{\partial f(x)}{\partial x_s}\frac{\partial g(x)}{\partial x_s} + \frac{1}{2}\frac{\partial^2 g(x)}{\partial x_s^2}\right]\right]^2 \right. \\ &\qquad \left. + \frac{1}{nh^q}\frac{1}{a_1\ldots a_q}\frac{\sigma^2(x)}{f(x)}\left(\int k(v)^2 dv\right)^q \right\} \nu(x)dx \\ &= \min_h h^4 \int \left[\sum_{s=1}^{q} a_s^2 B_s(x) \left[\frac{1}{f(x)}\frac{\partial f(x)}{\partial x_s}\frac{\partial g(x)}{\partial x_s} + \frac{1}{2}\frac{\partial^2 g(x)}{\partial x_s^2}\right]\right]^2 \nu(x)dx \\ &\qquad + \frac{1}{nh^q}\int \frac{1}{a_1\ldots a_q}\frac{\sigma^2(x)}{f(x)}\left(\int k(v)^2 dv\right)^q \nu(x)dx \\ &= \min_h h^4 C_1 + \frac{1}{nh^q} C_2 \end{aligned}$$

By taking the first order condition:

$$\begin{aligned} 4h^3 C_1 - \frac{q}{nh^{q+1}} C_2 &= 0 \\ \implies h = \left[\frac{qC_2}{4C_1}\right]^{\frac{1}{4+q}} \frac{1}{n^{\frac{1}{4+q}}} &= Cn^{-\frac{1}{4+q}} \end{aligned}$$

We should keep going on, now plugging in h back in the original equation and minimizing this all over again for each a_s. This cannot be done because we don't know $B_s(x)$, $\sigma^2(x)$, $f(x)$ etc. These terms can be estimated, and the optimal $a_1, \ldots, a_q$ calculated by some numerical mechanism in a computer. This is, unfortunately, very cumbersome. It is a valid method whenever the next strategy we will discuss (cross-validation) is even more computer intensive. Nevertheless, the above equation is still informative as it gives us a

rough idea on the bandwidth choice. The point of showing you this approach is so that you know Silverman's rule-of-thumb:

$$h_s \propto n^{-\frac{1}{4+q}}$$

Which is sometimes useful for a first approximation. For instance, in the univariate case, a good bandwidth with which to start might be

$$\tilde{h} = \frac{2\,\mathrm{SD}(X)}{n^{1/5}}$$

where $\mathrm{SD}(X)$ is the estimated standard deviation of X.

1.2 Least Squares Cross-Validation

This is the most commonly used method in practice. The strategy is the following:

1. For each observation, try to predict its value using the other observations. Let the bandwidth be fixed at the theoretical levels $h_1, \ldots, h_q$. This is known as the "leave-one-out" method. Hence,

$$\hat{g}_{-i}(X_i) = \frac{\sum_{j \neq i} Y_j K\left(\frac{X_j - X_i}{\mathbf{h}}\right)}{\sum_{j \neq i} K\left(\frac{X_j - X_i}{\mathbf{h}}\right)}$$

   where we substituted the Y_j's, the X_j's and X_i, but left the bandwidths as they are. Do this for all the observations in the sample.

2. Calculate the mean squared errors of these predictions:

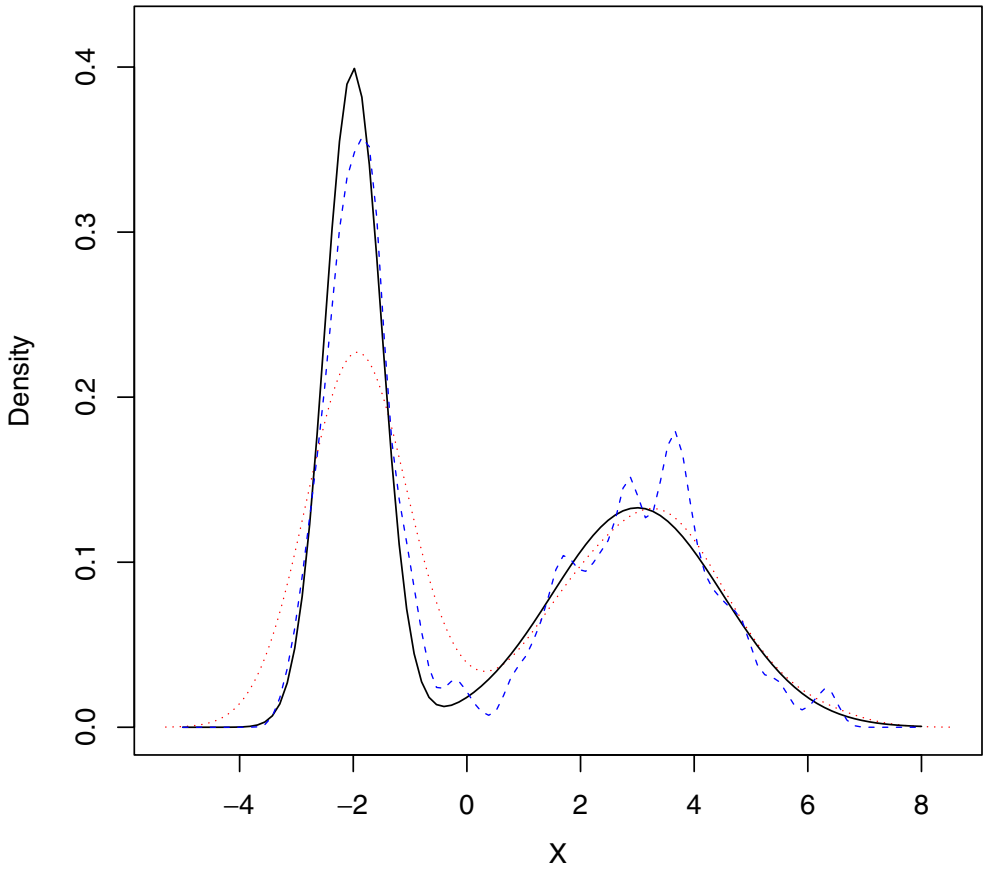
$$CV(h_1, \ldots, h_q) = \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \hat{g}_{-i}(X_i)\right)^2$$

3. Have a computer find the $h_1, \ldots, h_q$ that minimizes this function.

This is an excellent method, but it has some practical problems because of boundary effects. It is a good idea in practice to restrict the search for bandwidths to a subset of the possibilities, avoiding excessively big bandwidths (so that almost every point becomes a boundary point), and also those that are so small that there is almost no data weighted by the kernel. A good first idea of where the bandwidth should be is to look at Silverman's rule-of-thumb bandwidth. The advantage of cross-validations is that most of the burden is computational, and it does not require a deep understanding of the econometrics involved. The computational burden amounts to one regression per observation and then a numerical optimization process in q dimensions.

In practice, researchers often avoid doing the numerical optimization step. Instead, they calculate $CV(h_1, \ldots, h_q)$ for many different values of $h_1, \ldots, h_q$, and then choose the $h_1, \ldots, h_q$ that yielded the least among the CV's. In order to make an educated guess of the range where to look for the optimal bandwidth, it is smart to test bandwidths of magnitude around the one prescribed by Silverman's rule-of-thumb approach. The computational burden of this procedure is n regressions times the number of bandwidth vectors surveyed. This is easy to program, and if q is not too large, this method can be quite effective.

Figure 1.2: Plug-in versus least squares cross-validation density estimates. The true density is the solid line, the dotted line is the plug-in density (the bandwidth is large, as evidenced by the over-smooth resulting curve), and the dashed line is the least squares cross-validation density (the bandwidth is small, as evidenced by the non-smooth resulting curve).



Notice that the original intent was to estimate $g(x)$ at a particular x. However, the cross-validation method is global, so the optimal h thus chosen is the best h in a global sense. For a particular x, the cross-validation bandwidth is likely not optimal. See for

example figure 1.2. The cross-validation method tried to pick the best bandwidth overall, but in order to fit the peak region, a smaller bandwidth was necessary. Hence, the resulting fit is clearly not optimal for x's in the smoother part of the function. There, a larger bandwidth would have been a better choice. This problem is often very pronounced when trying to choose the bandwidth for estimation near the boundary (not shown in figure 1.2). For estimation near the boundary, as a general rule, (1) do not use kernel regression. Instead, use local polynomials, which will be taught later in this class, and (2) when using local polynomials, after finding the cross-validation bandwidth, inflate the cross-validation bandwidth near the boundary points, and deflate it near the interior points.

In spite of this, in practice the goal is not really to choose the optimal bandwidth, but rather a bandwidth in the right scale, so that the resulting curve is neither too smooth, nor too rough. If the function g is somewhat smooth, the cross-validation bandwidth should be a good choice. In figure 1.2 the curve is not very smooth, and though the cross-validation fit is worse than the plug-in in the smoother part of g, the cross-validation fit is generally closer to the true g, whereas the plug-in grossly misestimates g in the peak area. Also, observe how the estimation biases somewhat cancel each other-out. Because of this, if the estimated g are plugged into an average, as it is often the case, then (1) the bandwidth choice matters less, because of the cancellation of the bias, and (2) smaller bandwidths are usually better, because the biases are smaller overall (see this point in more detail in the next section). Sure, the smaller bandwidths cause the variance of the estimation of $g(x)$ at a particular x to increase, but the average divides the variance by n, so we prioritize bias reduction over variance in such cases.

It is common practice to report results with the cross-validation bandwidth and also with some other bandwidths.

2 Inference

In order to test hypotheses about nonparametric estimators, it is necessary to estimate the variance. The bias also plays an important role, and should be estimated. However, the bias estimation issue is often bypassed using a trick called "undermoothing."

2.1 Undersmoothing for bias reduction

If $\sqrt{nh_1 \dots h_q}h_s^2 \to 0$ for all s, the bias is asymptotically negligible. A bandwidth which satisfies this condition is rarely optimal from the mean squared errors point-of-view, but if the sample is somewhat large we prefer not to deal with the bias. This strategy of choosing bandwidths such that $\sqrt{nh_1 \dots h_q}h_s^2 \to 0$ is known as **undersmoothing**, because it often means that we have to make the bandwidth smaller than the optimal, and therefore the

resulting estimated curve would be less smooth than it should have been.

In practice, undersmoothing can be simulated by choosing the optimal bandwidth by any of the available techniques, and then picking a bandwidth that is slightly smaller. The effects of this approach can be seen in figure 1.2: undersmoothing creates a curve which is likely rougher than the true g. However, this strategy yields small biases overall. Over-smoothing (choosing larger-than-optimal bandwidths for a given point) can result in gross miss-estimation in less smooths parts of the function.

2.2 The Plug-in method of variance estimation

Now we turn to the estimation of the variance. The plug-in method consists of substituting the variance terms by estimators. hence:

$$\hat{V}(\hat{g}(x)) = \frac{1}{nh_1 \dots h_q} \frac{\hat{\sigma}^2(x)}{\hat{f}(x)} \left(\int k(v)^2 dv \right)^q$$

Then,

$$\hat{f}(x) = \frac{1}{nh_1 \dots h_q} \sum_{i=1}^{n} K\left(\frac{X_i - x}{\mathbf{h}} \right).$$

To estimate $\sigma^2(x)$, observe that

$$\sigma^2(x) = E(u^2 \mid X = x)$$

This is a nonparametric function which can be estimated. If we knew the values of the u_i, we could do

$$\tilde{\sigma}^2(x) = \sum_{i=1}^{n} k_h(X_i - x) u_i^2$$

However, since we do not know u_i, we will substitute it with estimated values:

$$\hat{\sigma}^2(x) = \sum_{i=1}^{n} k_h(X_i - x) \left[Y_i - \hat{g}(X_i) \right]^2$$

The estimator $\hat{\sigma}^2(x)$ can be shown to be a consistent estimator of $\sigma^2(x)$ (you have to under-smooth for this), but we will not cover this result in this course. It can be computationally intensive, though, because it requires the estimation of n regressions in order to acquire the $\hat{g}(X_i)$.

2.3 The Bootstrap

If we would like to test hypotheses using the estimators of $\hat{g}(x)$ and $\hat{V}(\hat{g}(x))$, the t-statistic

$$t = \frac{\hat{g}(x) - g(x)}{\sqrt{\hat{V}(\hat{g}(x))}}$$

would not be adequate in smaller samples, even with undersmoothing. The fact is that there is still bias, and that may not be negligible, so that t cannot be compared to the $N(0,1)$ distribution.

The bootstrap can solve this problem, and it consists in the following approach:

1. Given the full sample $\{(Y_1, X_1), \ldots, (Y_n, X_n)\}$, draw R samples of size n from the original, with reposition. Index the new samples by r.

2. Calculate the

$$t_r = \frac{\hat{g}_r(x) - \hat{g}(x)}{\sqrt{\hat{V}_r(\hat{g}(x))}}$$

   where $\hat{g}_r(x)$ is simply the estimator of $g(x)$ applying the formula for $\hat{g}(x)$ in the sample r, and $\hat{V}_r(\hat{g}(x))$ is the estimator of the variance of $\hat{g}(x)$ using the formula for $\hat{V}(\hat{g}(x))$ in each sample r. Make sure that you undersmooth in those estimators.

3. Use the t_r to calculate critical values. For example, the critical values for a 5% significance level dual-sided test are given by the 2.5% and the 97.5% quantiles among the t_r's. Let the critical values obtained be $c^*_{2.5}$ and $c^*_{97.5}$. Observe that these critical values are not necessarily symmetrical. The test of $H_0 : g(x) = 3$ will reject H_0 if

$$\frac{\hat{g}(x) - 3}{\sqrt{\hat{V}(\hat{g}(x))}} < c^*_{2.5} \quad \text{or} \quad \frac{\hat{g}(x) - 3}{\sqrt{\hat{V}(\hat{g}(x))}} > c^*_{97.5}.$$

4. Using the critical values, you can build confidence intervals. For example, the 95% confidence interval is:

$$\left[\hat{g}(x) + c^*_{2.5}\sqrt{\hat{V}_r(x)}\ ,\ \hat{g}(x) + c^*_{97.5}\sqrt{\hat{V}_r(x)}\right].$$

The recommendation for how many samples to generate (how large is R) is the following: as many as is feasible. There is the variance generated by the fact that the original sample is not the population, and the variance generated by the samples themselves. You cannot do anything about the first variation, but you want to keep the second variation to a minimum.

Unfortunately, the bootstrap can be very hard computationally. The estimation of the t_r's requires $R(n+1)+1$ regressions, and Rn weighted averages. Because of this, it is common to see researchers do the bootstrap in the following way:

1. For each sample r, calculate $\hat{g}_r(x)$.
2. Calculate the variance of the $\hat{g}_r(x)$ by doing

$$\hat{V}_1(x)\frac{1}{R-1}\sum_{r=1}^{R}(\hat{g}_r(x)-\hat{g}(x))^2$$

3. Perform hypothesis tests using this variance. For example, the test of $H_0 : g(x) = 3$ will reject H_0 if

$$\frac{\hat{g}(x)-3}{\sqrt{\hat{V}_1(x)}} < -1.96 \quad \text{or} \quad \frac{\hat{g}(x)-3}{\sqrt{\hat{V}(\hat{g}(x))}} > 1.96.$$

Though this method is correct asymptotically, it has a tendency to underestimate $V(x)$. This often happens when you bootstrap a non-pivotal statistic, as is the case with $\hat{g}(x)$, which is not pivolat because its asymptotic variance is unknown. t in the other hand is pivotal, because if we undersmooth, $t \xrightarrow{d} N(0,1)$. However, the computational burden of the second approach is very small, because it requires only $R+1$ regressions, and no knowledge about the behavior of $\hat{g}(x)$ other than its formula. If you must apply this method, make sure to exaggerate in the number of repetitions and to be very conservative with respect to the standard errors that you find.

3 Local Polynomial

3.1 The univariate case

Remember that the Nadaraya-Watson estimator of $g(x)$ is the weighted mean of the Y_i's, and the weights depend on the distance between the X_i and x. One way to think about this is as if in a neighborhood of x we assumed that g is constant. In this case,

$$g(X_i) = g(x)$$

Then the kernel regression is really an estimator of the model:

$$Y_i = g(x) + u$$

for X_i in a neighborhood of x. This problem can be solved by the estimation of b_0 in the econometric model

$$Y_i = b_0 + u.$$

Then, $\hat{g}(x) = \hat{b}_0$, which can be obtained by doing an average, or more generally a weighted average, of the Y_i corresponding to X_i in a neighborhood of x.

We can generalize this by thinking that, in a neighborhood of x, g is not constant, but a polynomial. One justification for this way of thinking is the Taylor expansion. Let's begin with the case where X_i is univariate. Provided g is $p+1$ times continuously differentiable, then

$$g(X_i) = g(x) + g^{(1)}(x)(X_i - x) + \frac{1}{2}g^{(2)}(x)(X_i - x)^2 + \cdots + \frac{1}{p!}g^{(p)}(x)(X_i - x)^p + Reminder$$

where *Reminder* is $o_p(||X_i - x||^{p+1})$. Hence,

$$Y_i = g(x) + g^{(1)}(x)(X_i - x) + \frac{1}{2}g^{(2)}(x)(X_i - x)^2 + \cdots + \frac{1}{p!}g^{(p)}(x)(X_i - x)^p + Reminder + u$$

If we were to approximate this function, we could do so by estimating the econometric model

$$Y_i = b_0 + b_2(X_i - x) + b_3(X_i - x)^2 + \cdots + b_p(X_i - x)^p + u$$

by a weighted regression using the weights given by the kernel. If the kernel has bounded support, then $X_i - x$ is at most h.[1]. Hence, the bias caused by the omission of *Reminder* is of the order of at most h^{p+1}. Therefore, $\hat{b}_0$ is an estimator of $g(x)$ with a bias of order at most h^{p+1}. If we simply estimated the mean, then the bias would be of order h^2, which is much larger. If you look back to your notes, you will see that h^2 is indeed the order of the bias of the constant kernel regression.

In order to perform the weighted regression, it is necessary to solve the problem:

$$\min_{b_0,b_1,\ldots,b_p} \sum_{i=1}^{n} k\left(\frac{X_i - x}{h}\right)(Y_i - b_0 - b_1(X_i - x) - b_2(X_i - x)^2 - \cdots - b_p(X_i - x)^p)^2.$$

[1] This is without loss of generality. Even if the kernel has a larger (but still finite) support, the maximum distance $X_i - x$ it is still proportional to h.

In matrix format, suppose that

$$\hat{b} = \begin{bmatrix} b_0 \\ \vdots \\ b_p \end{bmatrix}, Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix}, \mathcal{X} = \begin{bmatrix} 1 & (X_1 - x) & (X_1 - x)^2 & \dots & (X_1 - x)^p \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & (X_n - x) & (X_n - x)^2 & \dots & (X_n - x)^p \end{bmatrix},$$

$$K = \begin{bmatrix} k\left(\frac{X_1 - x}{h}\right) & & 0 \\ & \ddots & \\ 0 & & k\left(\frac{X_n - x}{h}\right) \end{bmatrix}$$

Then,

$$b = (\mathcal{X}^T K \mathcal{X})^{-1} \mathcal{X}^T K Y.$$

which is the formula of the weighted least squares regression. Remember that $\hat{b}_0$ is the estimator of $g(x)$, and therefore, if $e_1 = (1, 0, \dots, 0)^T$ is the first canonical vector of dimension $p + 1$, then

$$\hat{g}(x) = e_1^T (\mathcal{X}^T K \mathcal{X})^{-1} \mathcal{X}^T K Y.$$

The way to program this estimator is straight-forward from its definition. In a matrix-based software, you just have to program the formula for $\hat{g}(x)$. In ready-to-use packages, simply run a weighted regression of the Y_i onto $1, X_i - x, (X_i - x)^2, \dots, (X_i - x)^p$, using the $K\left(\frac{X_i - x}{h}\right)$ as weights, and then get the estimator of the coefficient of the constant.

The choice of h in the local polynomial can be made in exactly the same way it is done in the constant kernel case. The local polynomial regression requires that one also choose p. The following asymptotic results will help guide the choice of p.

3.1.1 Asymptotic behavior of the univariate local polynomial estimator

Theorem: Suppose that

1. Let $\mathcal{G}$ denote the class of Borel measurable functions that have a finite second moment. Assume that $g(\cdot) = E(Y \mid X = \cdot)$ belongs to $\mathcal{G}$.
2. $E(Y^2) < \infty$, and define $\sigma^2(x) := Var(u \mid X = x)$.
3. X is a random variable with a density function $f_X(x) = f(x)$ twice continuously differentiable in x.
4. Let x be such that $f(x) > 0$.
5. g is $p + 1$ times continuously differentiable in x.

6. The kernel $k(\cdot)$ has a compact support, is bounded, symmetric, integrates to 1, and has a second moment which is different from zero.

7. The sample (Y_i, X_i), $i = 1, \ldots, n$ is i.i.d.

8. As $n \to \infty$, $h \to 0$, $nh \to \infty$ and $\sqrt{nh}\, h^{p+1}$ is bounded.

Then,

$$\sqrt{nh}\,(\hat{g}(x) - g(x) - B(x)) \xrightarrow{d} N\left(0, \frac{\sigma^2(x)}{f(x)}\, e_1^T \Gamma^T \Delta \Gamma e_1\right)$$

where if p is odd:

$$B(x) = c_{1,p}\, h^{p+1} \left[\frac{g^{(p+1)}(x)}{(p+1)!}\right] e_1^T \Gamma^{-1} \begin{bmatrix} \gamma_{p+1} \\ \vdots \\ \gamma_{2p+1} \end{bmatrix}$$

and if p is even:

$$B(x) = c_{2,p}\, h^{p+2} \left[\frac{g^{(p+1)}(x)}{(p+1)!}\frac{f'(x)}{f(x)}\right] e_1^T \Gamma^{-1} \begin{bmatrix} \gamma_{p+2} \\ \vdots \\ \gamma_{2p+2} \end{bmatrix}$$
$$+ c_{3,p}\, h^{p+2} \left[\frac{g^{(p+2)}(x)}{(p+2)!}\right] e_1^T \Gamma^{-1} \Gamma_{(+1)} \Gamma^{-1} \begin{bmatrix} \gamma_{p+1} \\ \vdots \\ \gamma_{2p+1} \end{bmatrix}$$

where

$$\Gamma = \begin{bmatrix} \gamma_0 & \cdots & \gamma_p \\ \vdots & & \vdots \\ \gamma_p & \cdots & \gamma_{2p} \end{bmatrix}, \Gamma_{(+1)} = \begin{bmatrix} \gamma_1 & \cdots & \gamma_{p+1} \\ \vdots & & \vdots \\ \gamma_{p+1} & \cdots & \gamma_{2p+1} \end{bmatrix}, \Delta = \begin{bmatrix} \delta_0 & \cdots & \delta_p \\ \vdots & & \vdots \\ \delta_p & \cdots & \delta_{2p} \end{bmatrix}$$
$$\gamma_j = \int v^j k(v) dv \text{ and } \delta_j = \int v^j k(v)^2 dv.$$

***

In order to understand this result, we must first analyze the assumptions:

1. Assumptions 1, 2 and 3 are the same as in the kernel regression case.

2. Assumption 4 is substantially weaker, because it does not require that x be an interior point. More about boundary estimation later.

3. Assumption 5 imposes a stronger restriction on differentiability. This is a theoretical assumption.

4. Assumption 6 requires now that the kernel have a compact support. This was not necessary before, and is necessary in order to guarantee that among all the X_i that the kernel will use for the weighted regression, the maximum absolute value of $X_i - x$ is proportional to h.

5. The requirements in the bandwidth are weaker than in the kernel case.

Now to the understanding of the theorem. We begin with the bias terms. As you can see, the bias has a different formula depending on whether p is even or odd. However, the first thing you must realize is that the bias is always proportional to h elevated to an even power. Therefore, the order of the bias is the same for each odd p as it is to the next even number. This means, for example, that the order of the bias is the same if we have a third degree polynomial or a second degree polynomial. It may seem as if one should choose the even degree then. However, one should do exactly the opposite, as will be shown soon. First, let's understand the bias.

The $p+1$'th derivative of g should affect the bias, because of the Taylor expansion reminder. This is exactly true when p is odd. However, when p is even, the $h^{p+1}\left[\frac{g^{(p+1)}(x)}{(p+1)!}\right]\ldots$ term in the bias is eliminated. The reason is because of the symmetry of the kernel, which causes all $\gamma_j = \int u^j k(u)du$ and $\delta_j = \int u^j k(j)^2 du$ to be zero when j is odd. If p is even, then $e_1^T \Gamma^{-1} \left(\gamma_{p+1}, \ldots, \gamma_{2p+1}\right)^T = 0$.

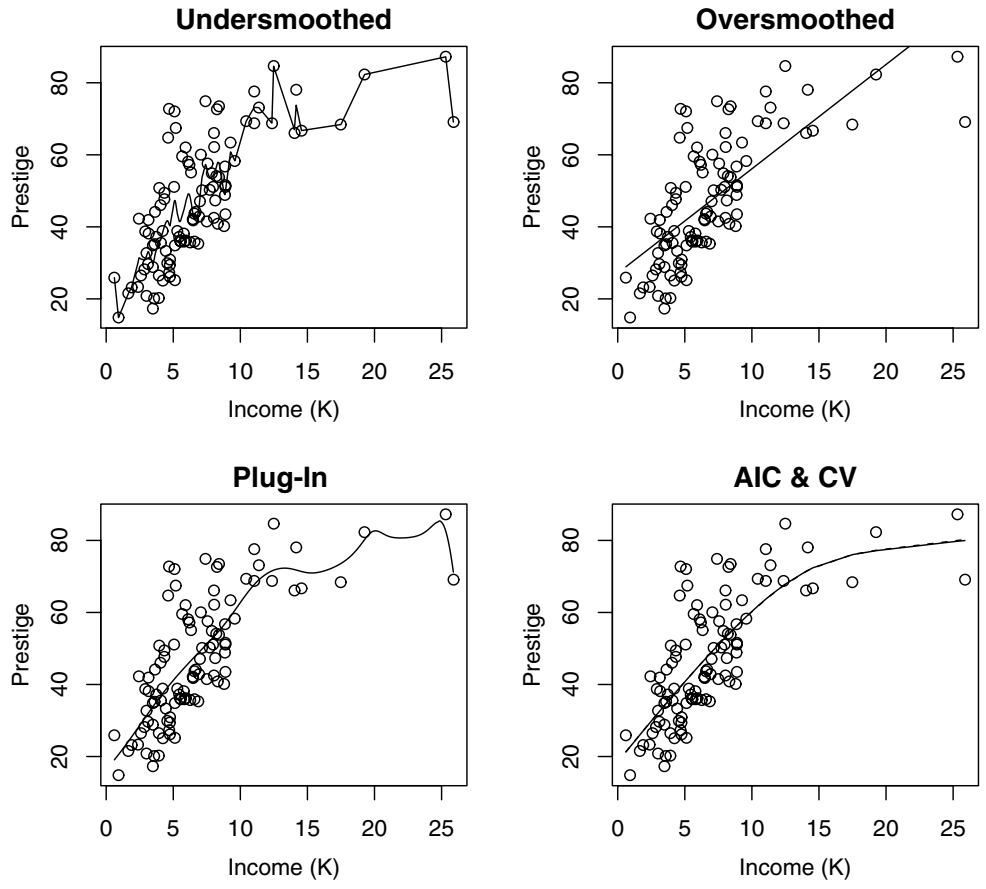
When p is even, the $p+1$'th derivative of $g(x)$ affects the bias when it is interacted with $f'(x)$. This is analogous to the term involving $g'(x)f'(x)$ in the Nadaraya-Watson estimator, and the intuition is the same: when the data is not balanced to the sides of x and $g^{p+1}(x)$ is large, then the estimation of $g^p(x)$ will be biased, and therefore so will be the estimation of $g(x)$.

Estimators with a bias that depends on $f'(x)$ are sometimes known as "non-adaptive," because they cannot adapt to data imbalances and are therefore prone to biases that can usually be avoided. The local polynomial with p odd does not suffer from this problem. It is easy to see that if we use an odd degree polynomial, it will tend to naturally pick up the slopes of $g(x)$ and its derivatives, while the even p local polynomial will tend to be symmetrical around x. Therefore, it is better to pick odd polynomials. Though there is no gain in the order of the bias with respect to the next smaller even polynomial degree, the bias itself is not affected by imbalances in both the function and the data distribution.

The finite sample variance of the local polynomial tends to be higher than that of the Nadaraya-Watson estimator. This happens because the local polynomial estimates a regression with p explanatory variables and a constant, and has therefore less degrees of

freedom than the simple constant kernel regression. The asymptotic variance term is the same for p even or odd, and is in fact very similar to the Nadaraya-Watson estimator's variance. In the local linear regression case, $e_1^T \Gamma^T \Delta \Gamma e_1 = (\int k(v)^2 dv)^q$, so the variance is exactly the same as in the Nadaraya-Watson case. The intuition of its behavior is therefore the same, and we will not repeat it. There is indeed no reason why they should be asymptotically different, since as $n \to \infty$, the degrees of freedom difference becomes irrelevant.

Figure 3.1.1: Local linear kernel estimates with varying window widths. Bandwidths are undersmoothed ($0.1\sigma n - 1/5$), oversmoothed ($103\sigma n - 1/5$), AICC and CV ($3.54\sigma n - 1/5$, $3.45\sigma n - 1/5$), and plug-in ($1.08\sigma n - 1/5$).



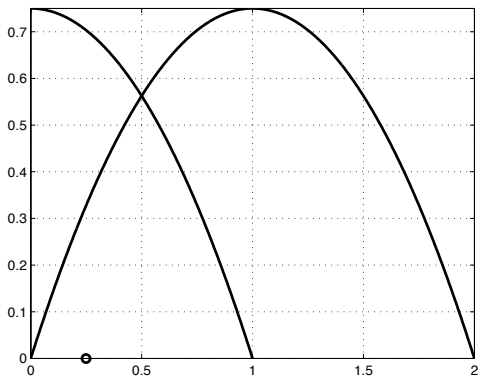
Hence, the recommendation for the practice when choosing p is to use as high a degree as possible, taking in consideration that there are still enough degrees of freedom. In practice, the local linear regression does already a masterful job of reducing the bias in comparison with the Nadaraya-Watson estimator. When choosing between an even degree and the

next odd degree, it is recommended that in general you choose the next odd degree. For example, between a degree 3 polynomial and a degree 2 polynomial, it is better to choose the degree 3 one. Sure, the variance is smaller with the degree 2 polynomial, and the bias is of the order h^4 in both cases. However, the degree 3 polynomial is adaptive, and will therefore not be affected by data imbalances to the sides of x. If the sample is large enough, this is doubtless the best choice. You should only choose the even degree polynomial if you have reasons to believe that g is fairly flat around x.

Are local polynomials always preferred to the Nadaraya-Watson estimator? Generally yes, but not always.

Let's first consider the boundary points issue. As you can see the theorem is equally valid if x is a boundary or an interior point. However, for finite samples, the kernel range is cut when x approximates to the boundary. In the boundary extremum, the kernel puts positive weight in exactly half of the area that it would otherwise do at an interior point (*vide* figure 1). This means that for the same bandwidth, an interior point estimation will use more data points than a boundary point estimation. This causes the variance to

Figure 3.1.1: Issues with boundary points for an Epanechnikov kernel



increase considerably. For the same reason, the bias at the boundary has a tendency to decrease, because the cuts in the kernel range generate an overrepresentation of observations that are closer to x. This issue is shared by the Nadaraya-Watson estimator. However, the Nadaraya-Watson is strongly biased at the boundary when g is sharply increasing or decreasing. This is a problem of the same nature as data imbalance caused by large $f'(x)$, because at the boundary there is data only to one side. Hence, this reason would have us preferring the local polynomial every time (for the same reason that we preferred the odd degree polynomials to the even degree ones, except there the bias reduction gain is of smaller order.)

However, there is another reason that causes the variance of the local polynomial to increase even more close to the boundary. When x is close to the boundary, the terms $X_i - x$ have increasingly all the same signal. When the boundary is small, the $X_i - x$ become dangerously closer to a constant. As a result, the constant and the $X_i - x$ terms behave as if they had a multicolinearity problem, which increases the variance further. The Nadaraya-Watson estimator does not suffer from this problem. Hence, if you believe that g is fairly flat close to x and you fear that you do not have much data close to the boundary, the Nadaraya-Watson estimator may do a better job than the local polynomial.

3.2 The multivariate local polynomial

The multivariate case is conceptually only a simple extension of the univariate case. Notation-wise, though, it can be very cumbersome, as is the case, for example, with the multivariate Taylor expansion. Masry (1996) developed the famous proof of the convergence of the local polynomial in the general case (which is in effect an exercise in notation), and you can find the results in that paper. In the interest of simplicity, we will develop the multivariate local linear case. The bias reduction effects of the linear case are already considerably large, and the local linear is often preferable to the local quadratic (as discussed in the last section). Since it is unlikely that in practice you will decide to apply the multivariate local polynomial estimator with a degree of 3 or more, the following results will probably suffice for any application you'll ever do.

The expansion is now

$$g(X_i) = g(x) + \nabla^{(1)} g(x)^T (X_i - x) + \frac{1}{2}(X_i - x)^T \nabla^{(2)} g(x)(X_i - x) + \dots$$

We can therefore estimate the model

$$g(X_i) = b_0 + b_{11}(X_{i1} - x_1) + b_{12}(X_{i2} - x_2) + \cdots + b_{1q}(X_{iq} - x_q) + u$$

with a weighted least squares regression. This is the same as solving the problem

$$\min_{b_0, b_{11}, \dots, b_{1q}} \sum_{i=1}^{n} K\left(\frac{X_i - x}{\mathbf{h}}\right) (Y_i - b_0 - b_{11}(X_{i1} - x_1) - b_{12}(X_{i2} - x_1) - \cdots - b_{1q}(X_{iq} - x_q))^2.$$

In matrix format, suppose that

$$\hat{b} = \begin{bmatrix} b_0 \\ b_{11} \\ \vdots \\ b_{1q} \end{bmatrix}, Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix}, \mathcal{X} = \begin{bmatrix} 1 & (X_{11} - x_1) & (X_{12} - x_2) & \dots & (X_{1q} - x_q) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & (X_{n1} - x_1) & (X_{n2} - x_1) & \dots & (X_{nq} - x_q) \end{bmatrix},$$

$$K = \begin{bmatrix} K\left(\frac{X_1 - x}{\mathbf{h}}\right) & & 0 \\ & \ddots & \\ 0 & & K\left(\frac{X_n - x}{\mathbf{h}}\right) \end{bmatrix}$$

Then,

$$b = (\mathcal{X}^T K \mathcal{X})^{-1} \mathcal{X}^T K Y.$$

which is the formula of the weighted least squares regression. Remember that $\hat{b}_0$ is the estimator of $g(x)$, and therefore, if $e_1 = (1, 0, \dots, 0)^T$ is the first canonical vector of dimension $p + 1$, then

$$\hat{g}(x) = e_1^T (\mathcal{X}^T K \mathcal{X})^{-1} \mathcal{X}^T K Y.$$

and to estimate the derivative of g with respect to x_s, we can do

$$\hat{g}^{(s)}(x) = e_{s+1}^T (\mathcal{X}^T K \mathcal{X})^{-1} \mathcal{X}^T K Y.$$

In practice, the only difference between estimating the univariate and the multivariate local linear regression is the kernel, which has to be multivariate, and the matrix $\mathcal{X}$.

3.2.1 Asymptotic behavior of the multivariate local linear estimator

Theorem: Suppose that

1. Let $\mathcal{G}$ denote the class of Borel measurable functions that have a finite second moment. Assume that $g(\cdot) = E(Y \mid X = \cdot)$ belongs to $\mathcal{G}$.
2. $E(Y^2) < \infty$, and define $\sigma^2(x) := Var(u \mid X = x)$.
3. X is a random vector with a density function $f_X(x) = f(x)$.
4. Let x be such that $f(x) > 0$.
5. g is twice continuously differentiable in x.

6. The kernel $k(\cdot)$ has a compact support, is bounded, symmetric, integrates to 1, and has a second moment which is different from zero.

7. The sample (Y_i, X_i), $i = 1, \ldots, n$ is i.i.d.

8. As $n \to \infty$, $h_s \to 0$ (for all $s = 1, \ldots, q$), $nh_1 \ldots h_q \to \infty$ and $(nh_1 \ldots h_q) \sum_{s=1}^{q} h_s^4$ is bounded.

Then,

$$\sqrt{nh_1 \ldots h_q} \left(\hat{g}(x) - g(x) - \sum_{s=1}^{q} h_s^2 B_s(x) \right) \xrightarrow{d} N \left(0, \frac{\sigma^2(x)}{f(x)} \left[\int k(v)^2 dv \right]^q \right)$$

where

$$B_s(x) = \frac{\int v^2 k(v) dv}{2} \left[\frac{\partial^2 g(x)}{\partial x_s^2} \right]$$

and

$$\sqrt{nh_1 \ldots h_q}\, h_s \left(\hat{g}_s^{(1)}(x) - g_s^{(1)}(x) \right) \xrightarrow{d} N \left(0, \frac{\sigma^2(x)}{f(x)} \frac{\left[\int k(v)^2 dv \right]^{q-1} \int v^2 k(v)^2 dv}{\left[\int v^2 k(v) dv \right]^2} \right).$$

The asymptotic variance of the local linear regression is similar to the Nadaraya-Watson, as was discussed in the univariate case, with the slight increase due to the use of $q-1$ extra regressors. This increase is more pronounced depending on the concavity of the kernel. The more concave the kernel, the smaller the amount of data considered, which increases the variance of estimation. This issue is magnified as more dimensions are added.

Observe also that the bias in the local linear case is the same as in the Nadaraya-Watson, except that it does not have the term that involves the first derivatives. This is natural, since the linear expansion is fitting those, as was discussed in the univariate case.

Finally, observe that it is possible to estimate the partial derivatives by looking at the estimators of the coefficient of the linear term. This estimator converges at a slower speed than the estimator of $g(x)$, as it is often the case with derivative estimators.