Chapter 2

Kernel Regression

2.1 Introduction

Regression analysis is a method for quantifying the relationship between a target or dependent variable, y, and one or more predictor variables (also called explanatory or independent variables, or covariates), x. We start with the case in which there is a single predictor, and denote the observations as $(x_1, y_1),\ (x_2, y_2), ...,\ (x_n, y_n)$.

Parametric modelling is carried out by examining the scatterplot of the data, postulating a parametric model, such as a simple linear model:

$$y_i = \theta_1 + \theta_2\, x_i + e_i, \quad i = 1, 2..., n,$$

or a more complex linear model, such as a cubic relationship:

$$y_i = \theta_1 + \theta_2\, x_i + \theta_3\, x_i^2 + \theta_4\, x_i^3 + e_i, \quad i = 1, 2..., n,$$

where the e_i are independently and identically distributed (iid) with $E(e_i) = 0$ and $\mathrm{Var}(e_i) = \sigma^2$. The θ–parameters can be estimated using the method of least squares, which is described in Appendix C, and σ^2 using the mean square of the estimated residuals.

Parametric modelling enjoys a number of advantages. Firstly, the method is easy to apply and to understand. The properties of the estimators are known and there is a methodology available to compute confidence intervals, to carry out hypothesis tests, diagnostic checking etc.. But the parametric approach suffers one major shortcoming, namely its lack of flexibility.

The nonparametric approach avoids making assumptions about the specific form of the relationship between the x and y. Instead one assumes that

$$y_i = m(x_i) + e_i,$$

where $m(x_i)$ represents some smooth function, and that the e_i are iid with $E(e_i) = 0$ and $\mathrm{Var}(e_i) = \sigma^2$. Figure 2.1 illustrates the two approaches.

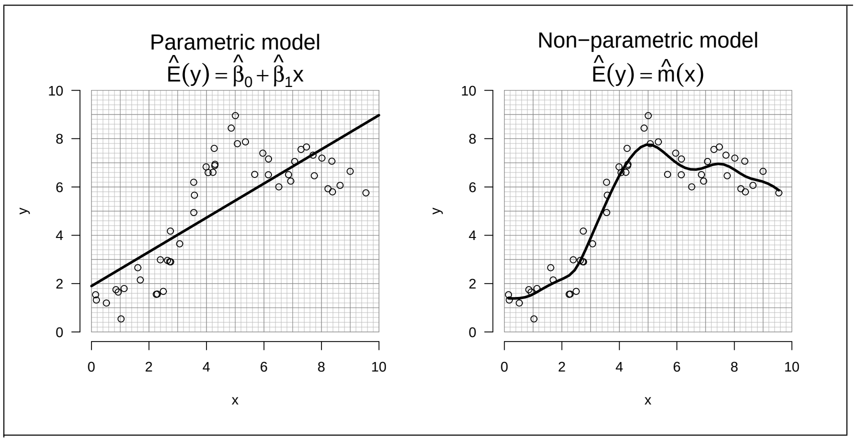

Figure 2.1: *Parametric and nonparametric models.*

2.2 Moving Averages

While the parametric model is determined by the parameters θ (and σ^2), the nonparametric approach tries to fit some function $m(x)$ to the data (Figure 2.1). The idea is to estimate $m(x)$ by averaging over the y–values "near" the point x. Define $J(x,h) = \{i : |x_i - x| < h\}$ as the set of indices whose corresponding x–values are "near" x (in this case closer than h). Let $n(x,h)$ be the number of indices in $J(x,h)$. Then, the local average that is used to estimate $m(x)$ at a point x is given by

$$\hat{m}(x) = \begin{cases} \frac{1}{n(x,h)} \sum\limits_{i \in J(x,h)} y_i & \text{for} \quad n(x,h) \neq 0 \\ \text{not defined} & \text{for} \quad n(x,h) = 0 \end{cases} .$$

Alternatively, and equivalently, $\hat{m}(x)$ can be expressed as a weighted average of all the y–values. The weighting function here is given by

$$w(x - x_i, h) = \begin{cases} 1 & \text{if} \quad |x - x_i| < h \\ 0 & \text{if} \quad |x - x_i| \geq h \end{cases} ,$$

and the estimator is given by

$$\hat{m}(x) = \begin{cases} \frac{\sum\limits_{i=1}^{n} w(x-x_i,h) y_i}{\sum\limits_{i=1}^{n} w(x-x_i,h)} & \text{for} \quad \sum\limits_{i=1}^{n} w(x - x_i, h) \neq 0 \\ \text{not defined} & \text{for} \quad \sum\limits_{i=1}^{n} w(x - x_i, h) = 0 \end{cases}$$

Figure 2.2 shows $\hat{m}(x)$ and the weights needed to estimate $m(4)$ using moving averages with different bandwidths. It is obvious that increasing the bandwidth smoothes the overall shape of $\hat{m}$, however, due to the rectangular weighting function, in detail $\hat{m}$ is not smooth but has the shape of a step function.

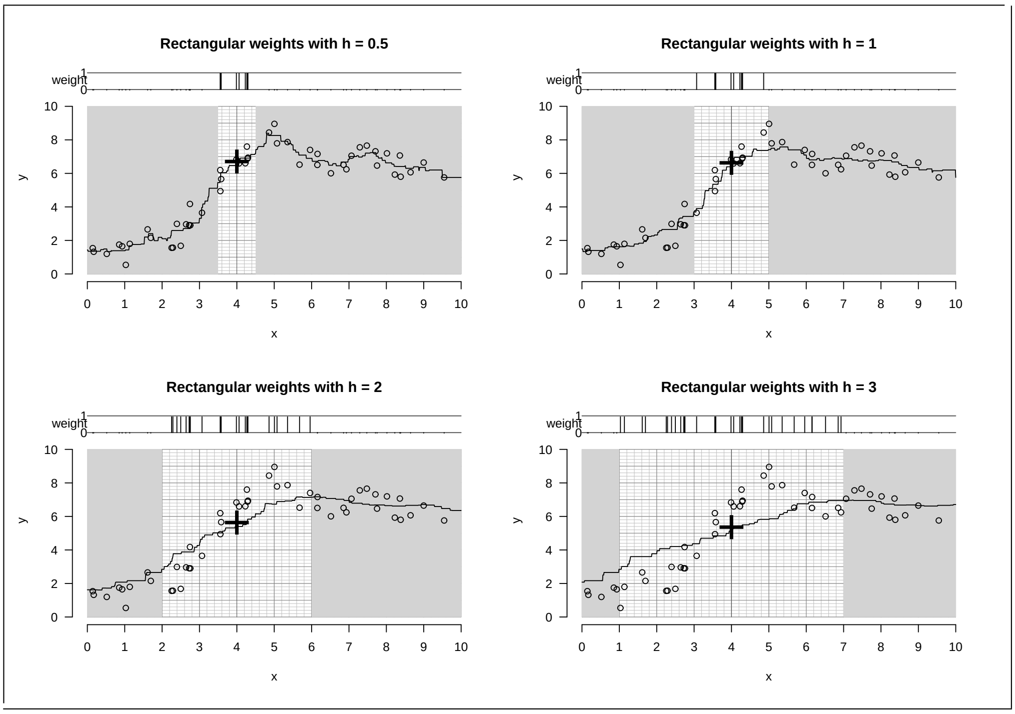

Figure 2.2: *Local averaging with a rectangular weighting function and different bandwidths.*

The advantage of the representation as weighted average is that one can also use other weighting functions instead of the rectangular weighting function used above, for example the Gaussian weighting function which is given below.

Rectangular weighting function: $w(z,h) = \begin{cases} 1 & \text{for} \quad |z| < h \\ 0 & \qquad \text{otherwise} \end{cases}$

Gaussian weighting function: $w(z,h) = \frac{1}{\sqrt{2\pi}\,h}\, e^{-\frac{1}{2}\left(\frac{z}{h}\right)^2}\,, \qquad -\infty < z < \infty$

For convenience we will sometimes use the following briefer notation:

$$w_i = w(x - x_i, h) \quad \text{and} \quad v_i = w_i / \sum_{j=1}^{n} w_j\,, \quad i = 1, 2, \ldots, n,$$

though it must be kept in mind that the weights, w_i, and the normalized weights, v_i, are functions of x, the values of the covariate, $x_1, x_2, \ldots, x_n$, and the bandwidth, h. The estimator can be written as

$$\hat{m}(x) = \frac{\sum\limits_{i=1}^{n} w_i y_i}{\sum\limits_{i=1}^{n} w_i} = \sum_{i=1}^{n} v_i\, y_i \qquad (2.1)$$

The estimator $\hat{m}(x)$ based on the Gaussian weighting function and the respective weights w_i for different x–values and bandwidths h are given in Figure 2.3. Here, due to the shape of the Gaussian weighting function $\hat{m}$ is also smooth in detail.

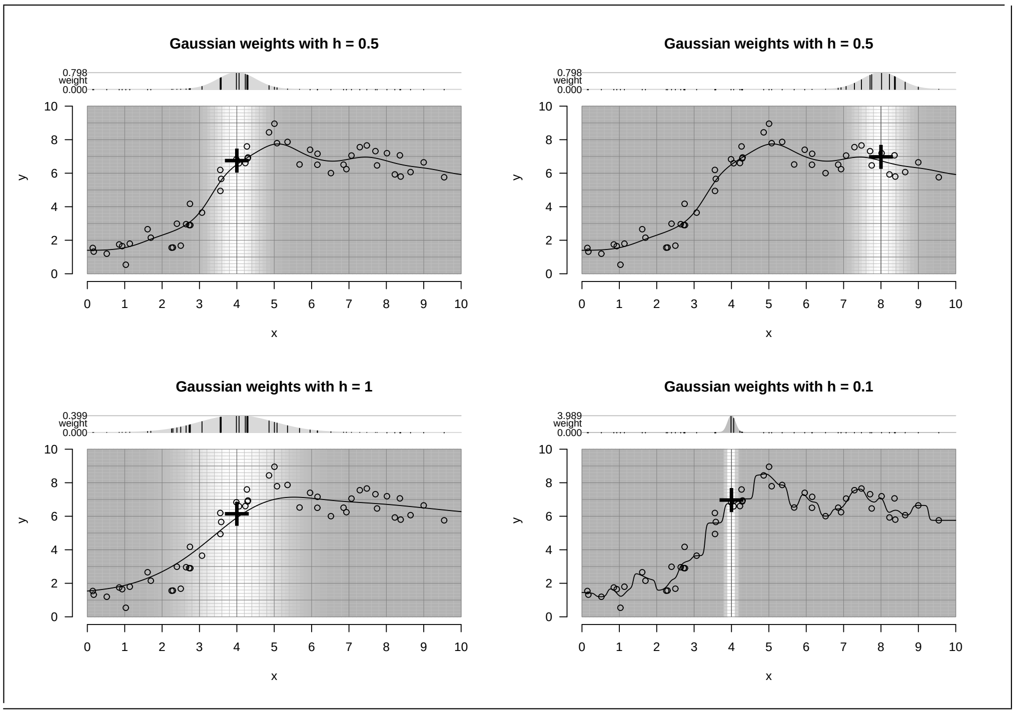

Figure 2.3: *Local averaging with a Gaussian weighting function and different bandwidths.*

2.3 Properties of $\hat{m}(x)$

The expectation of $\hat{m}(x)$ is given by

$$\mathrm{E}\,\hat{m}(x) = \mathrm{E}\sum_{i=1}^{n} v_i\, y_i = \sum_{i=1}^{n} v_i \mathrm{E}\, y_i = \sum_{i=1}^{n} v_i\, m(x_i) \tag{2.2}$$

It follows that, in general, $\mathrm{E}\hat{m}(x) \neq m(x)$, i.e. $\hat{m}(x)$ is a biased estimator of $m(x)$. As will be shown later, the bias of $\hat{m}(x)$ depends on the following factors:

— the shape of $m(x)$ in the neighbourhood of x,

— the number of x_i–values "near" x,

— the weighting function, and especially the bandwidth h.

Figure 2.4, which displays $m(x)$ and $E\,\hat{m}(x)$ for different sample sizes and bandwidths, illustrates the effect of the above factors.

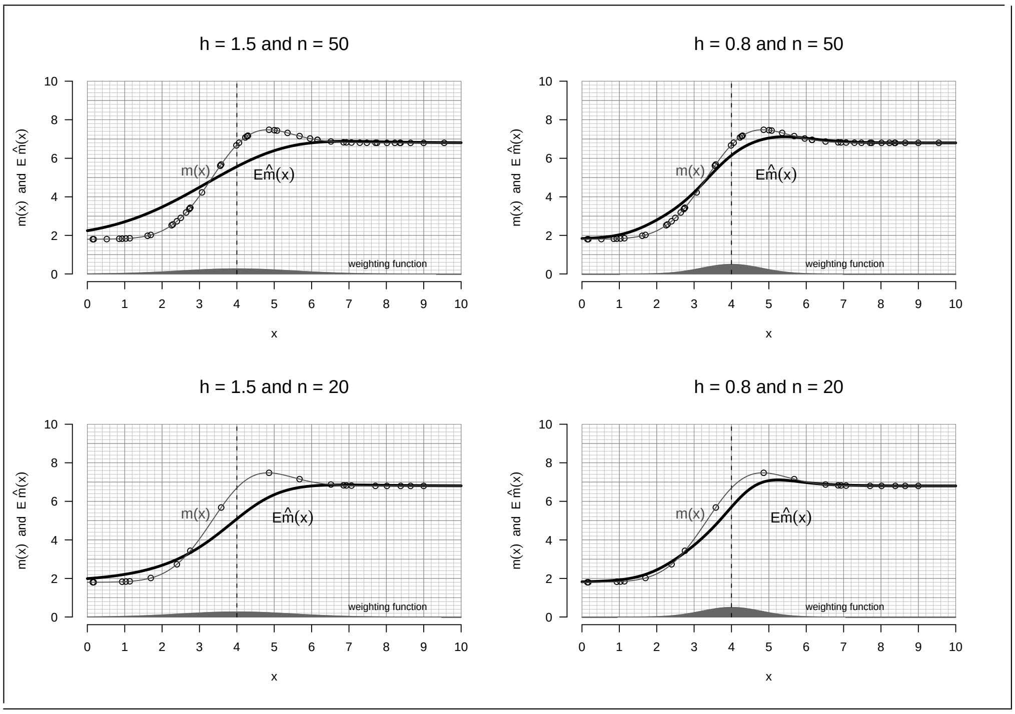

Figure 2.4: *$m(x)$ and $E\,\hat{m}(x)$ for different sample sizes and bandwidths.*

Notice that the bias is greatest near $x = 4$, where the curvature of $m(x)$ is greatest. Just as in the case of kernel density estimators, $\hat{m}$ tends to "erode the hills and fill in the valleys". Secondly, note that, at the values of x at which $\hat{m}(x)$ is biased (e.g. $x = 4$) the bias increases as the number of observations near x decreases. Finally, the bias rises with increasing bandwidth.

Note that $E\,\hat{m}(x)$ can only be computed if $m(x)$ is known — which is not the case in general — and that it does not depend on the y_i–values but on the $m(x_i)$–values. For that reason the points shown in Figure 2.4 mark the $(x_i, m(x_i))$–combinations instead of the (x_i, y_i)–combinations which were given in the preceding figures.

The variance of $\hat{m}(x)$ is obtained by making use of the assumption that the e_i, and hence the y_i, are independently distributed.

$$\mathrm{Var}\,(\hat{m}(x)) = \mathrm{Var}\left(\sum_{i=1}^{n} v_i\, y_i\right) = \sum_{i=1}^{n} v_i^2\, \mathrm{Var}\,(y_i)$$

The variance of y_i is given by

$$\mathrm{Var}(y_i) = \mathrm{Var}(m(x_i) + e_i) = \mathrm{Var}(e_i) = \sigma^2$$

and thus

$$\mathrm{Var}\,(\hat{m}(x)) = \sum_{i=1}^{n} v_i^2\, \sigma^2\,. \tag{2.3}$$

In general $\mathrm{Var}(\hat{m}(x))$ depends on the following factors (for a derivation of these results see Chapter 2.5):

— the variance of the residuals σ^2; it is directly proportional to σ^2,

— the sample size n; it decreases as n increases,

— the precise positions of the x_i–values "near" x,

— the shape of the weighting function,

— the bandwidth h; it decreases with increasing h.

Figure 2.5, which displays $m(x)$ and $\mathrm{Var}(\hat{m}(x))$ for different sample sizes and bandwidths, illustrates the effect of some of these factors (an additional analysis of the influence of the bandwidth h on the variance is covered in Assignment 4).

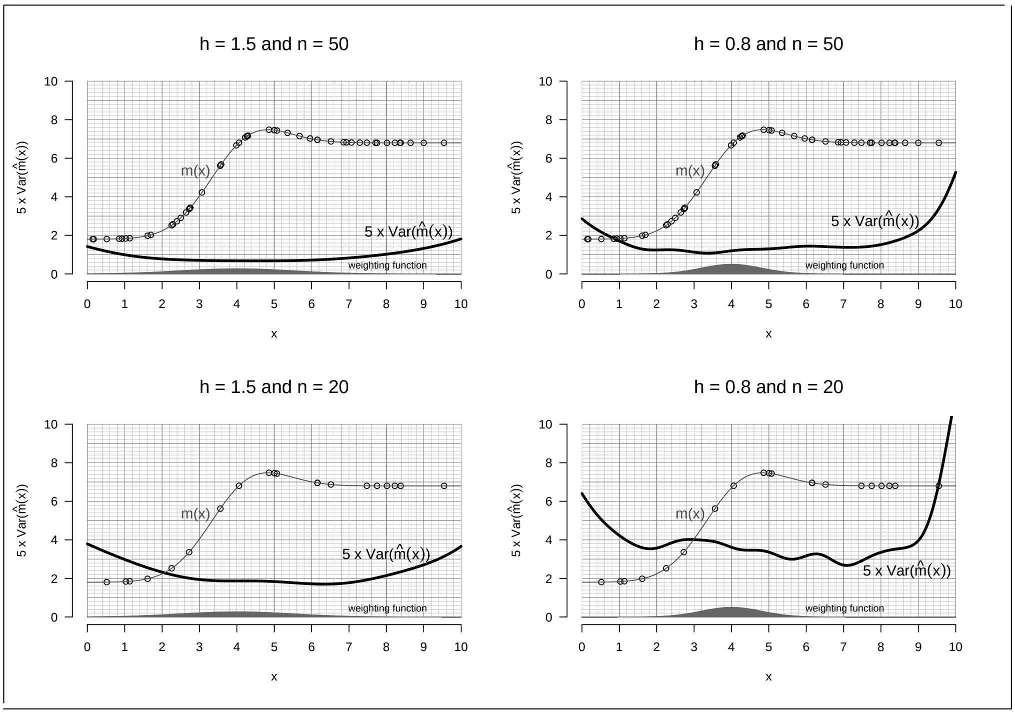

Figure 2.5: *Variance of $\hat{m}(x)$ for different sample sizes and bandwidths.*

Notice that $\mathrm{Var}(\hat{m}(x))$ increases substantially at the "ends"; that is because there are fewer observations near these values of x. Furthermore, the variance decreases with increasing sample size n and bandwidth h.

Analogue to the expectation, the variance $Var(\hat{m}(x))$ can only be computed if $m(x)$ is known and does not depend on the y_i–values but on the $m(x_i)$–values. Therefore the points given in Figure 2.5 mark the $(x_i, m(x_i))$–combinations instead of the (x_i, y_i)–combinations again.

Although $\hat{m}(x)$, based on moving averages, provides a flexible estimator of $m(x)$, it can be improved in certain respects. One is the "end effect": $\hat{m}(x)$ tends to be "too horizontal" at the edges. This leads to a high bias unless $m(x)$ is approximately constant at the ends. The next section describes an alternative estimator which has the additional advantage of being (asymptotically) "design adaptive", which means that its bias does not depend on the pattern of the design points, i.e. the positions of the x_is (see e.g. Fan (1992),(1993)).

2.4 Local Linear Regression

The method of moving averages for nonparametric regression can be extended in several ways. E.g. consider the following approaches for estimating $m(x)$:

(a) Fit the model $y_i = \theta_1 + e_i$ using the method of weighted least squares with weights $w_i = w(x_i - x, h)$. Here one estimates θ_1 by minimizing $\sum_{i=1}^{n} w_i e_i^2$. This yields $\hat{m}(x) = \hat{\theta}_1$ and is precisely the "moving average" estimator discussed in the previous section.

(b) Fit the model $y_i = \theta_1 + \theta_2 x_i + e_i$ using the method of weighted least squares, i.e. estimate θ_1 and θ_2 by minimizing $\sum_{i=1}^{n} w_i e_i^2$. This yields $\hat{m}(x) = \hat{\theta}_1 + \hat{\theta}_2 x$.

(c) Fit the model $y_i = \theta_1 + \theta_2 x_i + \theta_3 x_i^2 + e_i$ using the method of weighted least squares, i.e. estimate θ_1, θ_2 and θ_3 by minimizing $\sum_{i=1}^{n} w_i e_i^2$. This yields $\hat{m}(x) = \hat{\theta}_1 + \hat{\theta}_2 x + \hat{\theta}_3 x^2$.

For a short introduction to the method of weighted least squares see Appendix C. Note that in local linear regression the θ–parameters are different for each x, i.e. the local regression curve has to be refitted for each value of x!

Figure 2.6 illustrates the three approaches to estimate $m(2)$ and $m(8)$ using a rectangular weighting function (top) and a Gaussian weighting function (bottom). In addition, $\hat{m}$ is given for all cases. Again it becomes clear that it is not reasonable to apply a rectangular weighting function in nonparametric regression since the resulting estimator $\hat{m}$ is not a smooth function.

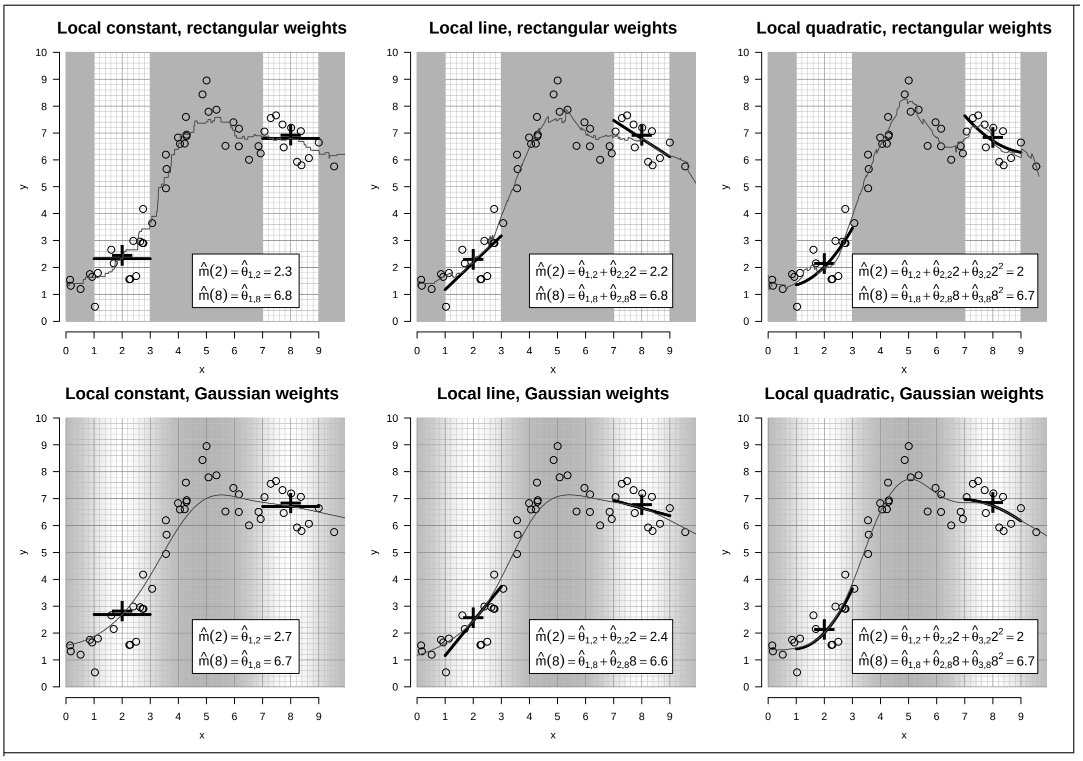

Figure 2.6: *Estimating $m(2)$ and $m(8)$ in local linear regression using a constant (left), a linear function (middle) and a quadratic function (right) with a rectangular (top) and a Gaussian weighting function (bottom).*

Of course it is possible to use a local cubic function instead of a local straight line or quadratic. The formulae are easy to derive — in fact they can be regarded as a special case of the local multivariate regression procedure. However, in practice the main gain is in going from a local constant to a local straight line and there is seldom any advantage in using a higher degree polynomial in the context of local regression. We therefore restrict our attention to (b).

It is also possible, and usual, to use alternative weighting functions, such as the tricube weighting function; it is simply a matter of using the appropriate weights w_i when estimating the parameters using the method of weighted least squares.

The computations can be carried out more efficiently if one uses a slightly different representation of the model (b), namely

$$y_i = \theta_1 + \theta_2(x_i - x) + e_i \ , \qquad i = 1, 2, ..., n \ .$$

By centering the observations around the point x, the "local regression line" estimator of $m(x)$ becomes simply $\hat{m}(x) = \hat{\theta}_1$, where

$$\begin{pmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \end{pmatrix} = (X'WX)^{-1}X'Wy \ ,$$

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \ , \quad X = \begin{pmatrix} 1 & (x_1 - x) \\ 1 & (x_2 - x) \\ \vdots & \vdots \\ 1 & (x_n - x) \end{pmatrix} \ , \quad W = \begin{pmatrix} w_1 & 0 & \dots & 0 \\ 0 & w_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & w_n \end{pmatrix} \ ,$$

and where $w_i = w(x_i - x, h)$ is the weight of the i-th observation at the point x. For a derivation of the estimator $\hat{\theta}$ see Appendix C. Note that the weighting function is quite general in this formula — one can use a rectangular, Gaussian or any other kernel — and that with this representation, too, the parameters θ_1 and θ_2 are different for each x and so $\hat{\theta}$ has to be recomputed for each value of x.

An advantage of using this matrix formulation is that it is easy to generalize to the case in which there are several covariates. Consider, for example, the case with two covariates where the observations are denoted by (x_{11}, x_{21}, y_1), (x_{12}, x_{22}, y_2), ..., (x_{1n}, x_{2n}, y_n). The model is given by

$$y_i = m(x_{1i}, x_{2i}) + e_i \ , \qquad i = 1, 2, ..., n \ .$$

We can now fit local planes to the observations:

$$y_i = \theta_1 + \theta_2 x_{1i} + \theta_3 x_{2i} + e_i \ , \quad i = 1, 2, .., n \ .$$

Again it is computationally convenient to use a centered model instead to estimate $m(x_1, x_2)$:

$$y_i = \theta_1 + \theta_2(x_{1i} - x_1) + \theta_3(x_{2i} - x_2) + e_i \ , \quad i = 1, 2, .., n \ .$$

The resulting estimator is given by $\hat{m}(x_1, x_2) = \hat{\theta}_1$, where

$$\begin{pmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \hat{\theta}_3 \end{pmatrix} = (X'WX)^{-1}X'Wy,$$

and

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, X = \begin{pmatrix} 1 & (x_{11} - x_1) & (x_{21} - x_2) \\ 1 & (x_{12} - x_1) & (x_{22} - x_2) \\ \vdots & \vdots & \vdots \\ 1 & (x_{1n} - x_1) & (x_{2n} - x_2) \end{pmatrix}, W = \begin{pmatrix} w_{11}w_{21} & 0 & \dots & 0 \\ 0 & w_{12}w_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \dots & w_{1n}w_{2n} \end{pmatrix},$$

$w_{1i} = w(x_{1i} - x_1, \ h_1)$ and $w_{2i} = w(x_{2i} - x_2, \ h_2)$. One must now select two bandwidths here, one for each of the two covariates.

It is also possible to use a more general weighting function instead of the product of two individual weighting functions used above. Secondly it is also possible to use different kernels for each covariate, e.g. a rectangular kernel for x_1 and a Gaussian for x_2. This type of generalization is rarely used in practice because there is seldom a good reason to do so.

2.4.1 The fitted values and the residuals

The estimated residuals are useful to assess certain aspects of the fit of the model, e.g. to check for heteroscedasticity, and to model their distribution for the purpose of computing confidence intervals for $m(x)$, and for predictions based on the model. To compute the residuals we need the "fitted values". We consider again the case in which there is a single covariate and a local line is fitted.

The residuals, e_i, are estimated using

$$\hat{e}_i = y_i - \hat{m}(x_i)\,,\ i = 1, 2, \dots, n\,.$$

Now $\hat{m}(x_i) = (1 \quad (x_i - x_i)) \begin{pmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \end{pmatrix} = \hat{\theta}_1$, where $\hat{\theta}_1$ changes for each i, $i = 1, 2, \dots, n$. In particular, the matrix of weights W changes for each i. Let W_i be the matrix of weights associated with x_i, i.e. W_i is a diagonal matrix with entries $w(x_j - x_i, h)$, $j = 1, 2, \dots, n$. Then $\hat{\theta}_1$ is the first row of the matrix $X(X'W_iX)^{-1}X'W_i$, say $(s_{i1}, s_{i2}, \dots, s_{in})$, multiplied by y. It follows that

$$\hat{m}(x_i) = (s_{i1}, s_{i2}, \dots, s_{in}) \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \qquad \text{and so}$$

$$\hat{m} = \begin{pmatrix} \hat{m}(x_1) \\ \hat{m}(x_2) \\ \vdots \\ \hat{m}(x_n) \end{pmatrix} = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1n} \\ s_{21} & s_{22} & \dots & s_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ s_{n1} & s_{n2} & \dots & s_{nn} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = Sy$$

.

Thus the fitted values are simply linear functions of the observed y–values. We say that the y–values have been linearly filtered. The matrix S is called smoother matrix and is closely connected to the concept of degrees of freedom of a model, i.e. the equivalent number of parameters of the model. Among other possibilities one can define the degrees of freedom of a nonparametric smoother as the trace of the smoother matrix, i.e. $df(\hat{m}) = tr(S)$. For a more detailed discussion of the concept of degrees of freedom in nonparametric regression see Chapter 3.

Given the smoother matrix S, the residuals are obtained as follows:

$$\hat{e} = y - \hat{m} = y - Sy = (I - S)y \quad ,\text{where} \quad \hat{e} = (\hat{e}_1,\ \hat{e}_2,\ ...,\ \hat{e}_n)'\,.$$

2.5 Asymptotic properties of kernel smoothers

We have seen that $\hat{m}(x)$ can be represented as a linear function of the y–values, i.e. in the form

$$\hat{m}(x) = \sum_{i=1}^{n} v_i \, y_i \ ,$$

where $v_i = \frac{w_i}{w_.}$, $w_i = w(x - x_i,\ h)$ and $w_. = \sum\limits_{i=1}^{n} w_i$. Note that the v_i depend on x, and that $\sum\limits_{i=1}^{n} v_i = 1$.

We also showed that

$$E\hat{m}(x) = \sum_{i=1}^{n} v_i m(x_i) \ ,$$

$$\mathrm{Var}(\hat{m}(x)) = \sum_{i=1}^{n} v_i^2 \sigma^2 \ .$$

We now consider the asymptotic properties of $\hat{m}(x)$, i.e. its properties as the sample size n increases indefinitely. These properties will depend on which values of x arise as n increases. If, for example, the values of x are all taken to be equal to some constant, c, then the performance of $\hat{m}(x)$ will only improve for $x = c$ (and perhaps for x "near" c) as n becomes large. It will not improve for x–values which are "far away" from c. On the other hand if the x–values are assumed to be spread uniformly over an interval (a, b) as n increases then the performance of $\hat{m}(x)$ is expected to improve for all $x \in (a, b)$. In what follows we will suppose that the x–values will be generated independently according to some density function $f(x)$.

For the local linear case it can be shown (see e.g. Ruppert and Wand (1994)) that, as n becomes large, one has

$$\begin{aligned}
\mathrm{E}\,(\hat{m}(x)) &\approx m(x) + \frac{h^2}{2}\, k_2 m''(x) \ , && (2.4)\\
\mathrm{Bias}\,(\hat{m}(x)) &\approx \frac{h^2}{2}\, k_2 m''(x), && (2.5)\\
\mathrm{Var}\,(\hat{m}(x)) &\approx \frac{\sigma^2}{nh^2}\, j_2\, \frac{1}{f(x)} \ , && (2.6)
\end{aligned}$$

where $k_2 = \int z^2 K(z) dz,\ j_2 = \int K^2(z) dz$. $K(z)$ is the kernel, i.e. $w(t,h) = \frac{1}{h}\ K\left(\frac{t}{h}\right)$.

The above expressions are important because they indicate how each component of the estimator affects the bias and the variance of the estimator.

The asymptotic bias depends on

— the bandwidth; it increases with increasing h,

— the variance of the kernel, k_2,

— the curvature of m at the point x, $m''(x)$; the bias is small if $m(x)$ is a straight line near the point x.

We are free to choose the bandwidth and the kernel but not $m''(x)$, because that is a property of the unknown function, m, that we are attempting to estimate.

The asymptotic variance depends on

— the residual variance, σ^2,

— the bandwidth; it decreases with increasing h,

— the sample size; increasing n reduces the variance,

— j_2, a property of the kernel,

— the pdf $f(x)$; $\text{Var}\,(\hat{m}(x))$ is smallest where $f(x)$ has a maximum, i.e. where the most x–values are available.

For a given sample size, n, we are free to select h and the kernel. Given that one uses a “reasonable” kernel, the most important issue is that of determining a suitable value of h. Note that by decreasing h we reduce the bias but increase the variance.

The asymptotic mean squared error, given by

$$\text{MSE}(\hat{m}(x)) = \frac{h^4}{4}\, k_2^2 (m''(x))^2 + \frac{\sigma^2}{nh}\, j_2\, \frac{1}{f(x)}\ , \tag{2.7}$$

provides a measure that takes both the bias and the variance into account.

Ideally we would wish to select the bandwidth that minimizes the MSE. In fact one can derive an expression for the optimal bandwidth (see Assignment 5) but the expression depends on a number of unknown factors — the curvature of $m(x)$, the residual variance and $f(x)$. One would need to estimate these components to estimate the optimal value of h. This is not easy although so-called “plug-in” estimators have been developed to do this (see for example Bowman and Azzalini (1997), Section 4.5.). We will consider an alternative method of estimating the optimal bandwidth, as described in the following section.

2.6 Cross-Validation

When fitting a model $y_i = m(x_i) + e_i$ to given data (x_1,y_1), ..., (x_n,y_n) one is interested in assesing the accuracy of the resulting fit $\hat{m}(x)$. Assume that (after fitting our model) we somehow obtain additional observations (x_1^*,y_1^*), ..., (x_m^*,y_m^*). These new observations could be used to assess the fit. One could, for example, compute the residual sum of squares (RSS) for the new observations:

$$\text{RSS} = \sum_{i=1}^{m} (y_i^* - \hat{m}(x_i^*))^2 \tag{2.8}$$

One way of obtaining "fresh observations" with which to asses the fit is to split the original sample with n observations into two subsamples: One subsample of size $n-k$, called the *calibration sample*, is used to estimate $\hat{m}$. The other sample of size k, called the *validation sample*, is used for assessing the fit.

In practice it is not the MSE but the prediction squared error (PSE) that is used to assess the fit of $\hat{m}$:

$$\text{PSE} = \text{E}(y - \hat{m}(x))^2 \tag{2.9}$$

However, the value for h which minimizes the PSE also minimizes the MSE:

$$\begin{aligned}
\text{PSE} &= \text{E}(y - \hat{m}(x))^2 = \text{E}(y - m(x) + m(x) - \hat{m}(x))^2 \\
&= \underbrace{\text{E}(y - m(x))^2}_{=\sigma^2} + \underbrace{\text{E}(m(x) - \hat{m}(x))^2}_{=MSE(\hat{m}(x))} + 2\text{E}\overbrace{\underbrace{(y - m(x))}_{=0}(m(x) - \hat{m}(x))}^{independence} \\
&= \sigma^2 + \text{MSE}(\hat{m}(x))
\end{aligned}$$

The third term above is equal to zero because the random variables $y - m(x)$ and $m(x) - \hat{m}(x)$ are independent, and $\text{E}(y - m(x)) = 0$. The indepence follows from the fact that $\hat{m}(x)$ is computed using the calibration sample, whereas y is from the (independent) validation sample. The PSE can be estimated for different values of h by using cross–validation.

A clever idea in this context (generally attributed to John Tukey) is the one–item–out cross–validation. In this case k (the size of the validation sample) equals one.

Then, the cross–validation criterion (CV) used for assesing the accuracy is defined as:

$$CV = \frac{1}{n}\sum_{i=1}^{n} \hat{e}_i^{*2} = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{m}_{-i}(x_i))^2$$

where $\hat{m}_{-i}$ denotes the estimator of m based on the original sample with one observation omitted, namely the observation (x_i,y_i). Assuming that $\text{MSE}(\hat{m}(x)) \approx \text{MSE}(\hat{m}_{-i}(x))$ one has that

$$
\begin{aligned}
\mathrm{E}(CV) &= \frac{1}{n}\sum_{i=1}^{n}\mathrm{E}(CV_i)\,,\\
\mathrm{E}(CV_i) &= \mathrm{E}(y_i-\hat{m}_{-i}(x_i))^2 = \mathrm{E}(y_i-m(x_i)+m(x_i)-\hat{m}_{-i}(x_i))^2\\
&= \underbrace{\mathrm{E}(y_i-m(x_i))^2}_{=\sigma^2}+\underbrace{\mathrm{E}(m(x_i)-\hat{m}_{-i}(x_i))^2}_{\approx \mathrm{MSE}(\hat{m}(x_i))}+\underbrace{\mathrm{E}(y_i-m(x_i))(m(x_i)-\hat{m}_{-i}(x_i))}_{=0\text{ (as above)}}\,,\\
\mathrm{E}(CV_i) &\approx \sigma^2+\mathrm{MSE}(\hat{m}(x_i))\,,\\
\mathrm{E}(CV) &\approx \sigma^2+\frac{1}{n}\sum_{i=1}^{n}\mathrm{MSE}(\hat{m}(x_i)) = \frac{1}{n}\sum_{i=1}^{n}\mathrm{PSE}(\hat{m}(x_i)) \qquad (2.10)
\end{aligned}
$$

Thus $\mathrm{E}(CV)$ is approximately equal to the average PSE, where the average is taken over the observed values $x_1, x_2, ..., x_n$. The cross–validation criterion, CV, is an (approximately) unbiased estimator of this average. The optimal bandwidth is estimated using that value of h that minimizes CV. The **R** library "sm" offers the function "hcv" to carry out the optimization, which is computationally demanding when the sample size is large.

Figure 2.7 displays a plot of the CV criterion for samples of different sizes. The samples for the middle and right-hand panels were obtained by removing observations from the sample in the left-hand panel. In each case the bandwidth that led to the smallest CV was used to compute the estimates $\hat{m}(x)$ that are displayed in the bottom three panels. Notice that, as n decreases, so the estimated optimal value of the bandwidth increases.

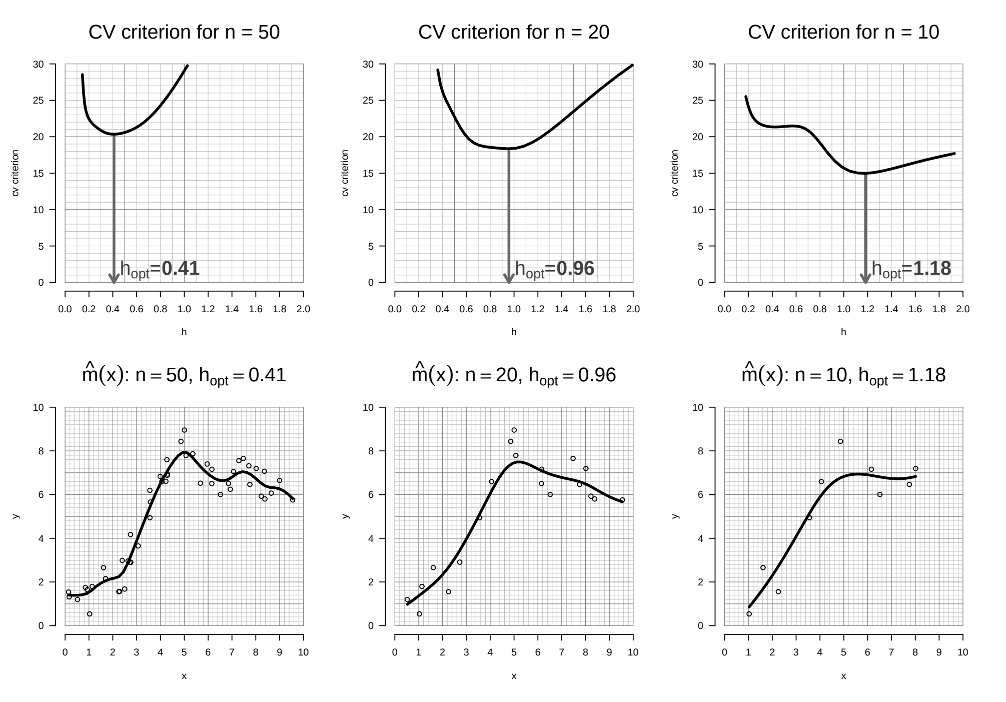

Figure 2.7: *Cross validation estimates of the optimal h and the corresponding $\hat{m}(x)$ for different sample sizes.*

2.7 Kernels with Variable Bandwidth

Figure 2.7 illustrates the point that the optimal value of h depends on the sample size. Other things being equal, the larger n the smaller the optimal value of h. Taking this idea one step further, it makes sense to use different bandwidths for different values of x, depending on how much sample information is available “near” x. Figure 2.8 illustrates this point.

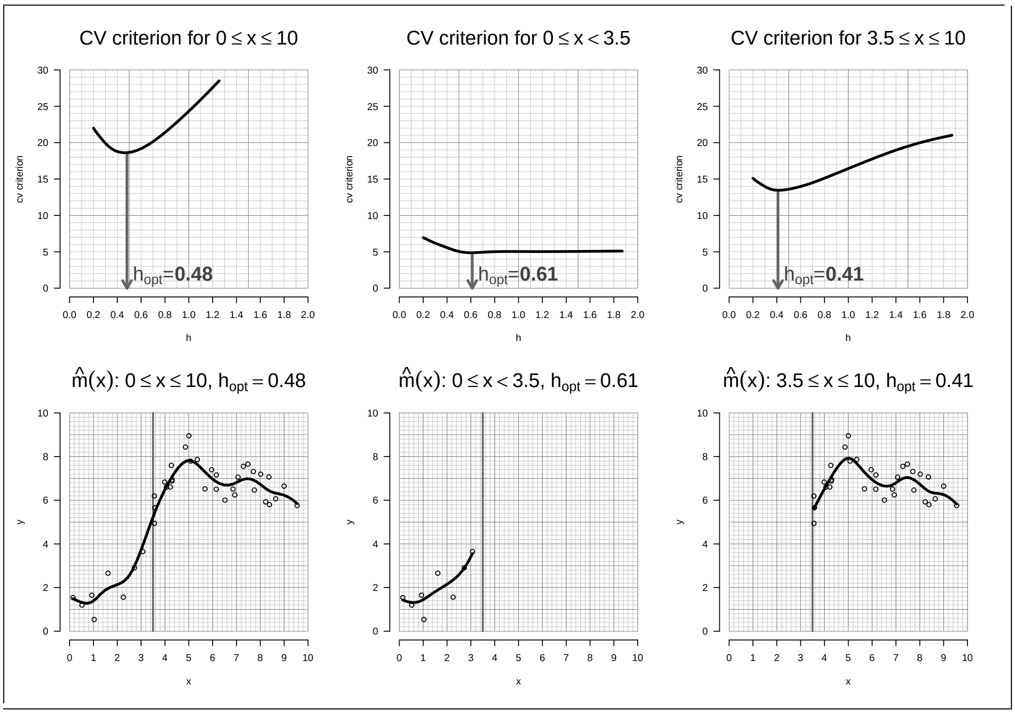

Figure 2.8: *Cross validation-based estimates for different intervals.*

The bottom left panel shows a sample of observations and $\hat{m}(x)$ using $h = 0.48$, the value that minimized the CV criterion (top left panel). This sample was constructed to have fewer observations in the interval $0 \leq x < 3.5$ than in the interval $3.5 \leq x \leq 10$. If we restrict our attention to the observations in the interval $0 \leq x < 3.5$ (bottom middle panel) then the optimal bandwidth is $h = 0.61$ whereas for the interval $3.5 \leq x \leq 10$ (bottom right panel) it is $h = 0.41$. Thus the value $h = 0.48$ that we obtain for the entire interval is “too small” for the sparsely sampled region on the left, and “too large” for the densely sampled region on the right.

It is a compromise that arises because we are using a single bandwidth for the entire curve.

The explanation for the behaviour of the estimated optimal bandwidth in Figure 2.8 is that the sample size n directly affects the variance of $\hat{m}(x)$ and thus also the mean squared error. The higher the sample size, the smaller the variance and its impact on the MSE for a given bandwidth h. This leads to a smaller optimal bandwidth. Generally speaking, a large n leads to a smaller optimal bandwidth. Similarly, the optimal local bandwidth depends on the "local sample size", that is the number of points near the x at which we are estimating $m(x)$.

The above discussion is intended to motivate the option of using different bandwidths for different parts of the curve, in other words to use variable bandwidth techniques. One of the most popular of those techniques is "`loess`" that is implemented by the **R** function with the same name (see `help(loess)` for implementation details). This method is based on the bandwidth of the form

$$h(x) \propto d_k(x), \tag{2.11}$$

where $d_k(x)$ represents the distance to the k–th nearest observed x–value.

Thus $d_k(x)$ is a measure of the density of points "near" x; if $d_k(x)$ is very small then there are (at least) k points near x, but if $d_k(x)$ is large the k-th nearest point is "far away" from x. In the former case we would wish to use a small bandwidth and in the latter case a large bandwidth. The choice of k determines what we regard as "near" or "far". In practice on specifies the "span", $f = \frac{k}{n}$, rather than k, i.e. the fraction of the sample size that is to be used to fit a local regression. The tricube kernel is often used to construct the weighting function:

$$K(z) = \begin{cases} (1-|z|^3)^3 & \text{for } |z| < 1 \\ 0 & \text{otherwise} \end{cases}$$

Figure 2.9 displays some variable-bandwidth estimates based on different spans, f. Also shown is the fixed-bandwidth estimate based on the CV-optimal bandwidth. Comparing the left two panels one can see that the two estimates are very similar for $x \geq 3.5$, but that the variable-bandwidth estimate is a little smoother for $x < 3.5$, the interval in which there are fewer observations.

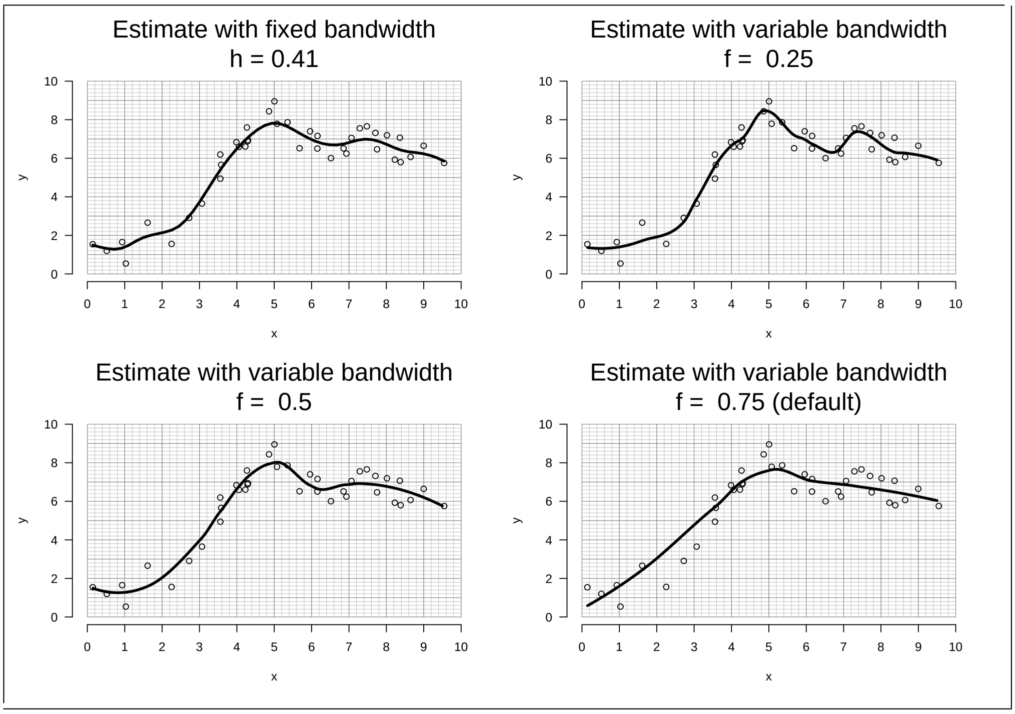

Figure 2.9: *Fixed– and variable–bandwidth estimators.*