Local regression I

Patrick Breheny

November 1

Simple local models

- In the previous lecture, we examined a very simple local model: k-nearest neighbors
- Another simple local regression model is the local average:

$$\hat{f}(x_0) = \frac{\sum_i y_i I(|x_i - x_0| < h)}{\sum_i I(|x_i - x_0| < h)}$$

- However, both of these approaches have the disadvantage that they lead to discontinuous estimates — as an observation enters/leaves the neighborhood, the estimate changes abruptly

The Nadaraya-Watson kernel estimator

- As with kernel density estimators, we can eliminate this problem by introducing a continuous kernel which allows observations to enter and exit the model smoothly
- Generalizing the local average, we obtain the following estimator, known as the *Nadaraya-Watson kernel estimator*:

$$\hat{f}(x_0) = \frac{\sum_i y_i K_h(x_i, x_0)}{\sum_i K_h(x_i, x_0)},$$

  where $K_h(x_i, x_0) = K(\frac{x_i - x_0}{h})$ is the kernel, and if K is continuous, then so is $\hat{f}$
- As with kernel density estimates, we need to choose a bandwidth h, which controls the degree of smoothing

Expected loss and prediction error for regression

- Because it is customary to treat x as fixed in regression, instead of integrating over x to obtain the expected loss, we average over the observed values of x:

$$\mathbb{E}L(f, \hat{f}) = \frac{1}{n} \sum_i \mathbb{E}L(f(x_i), \hat{f}(x_i))$$

- The expected squared error loss is particularly convenient in regression, as it is directly related to the *expected prediction error*:

$$\text{EPE} = \mathbb{E} \left\{ \frac{1}{n} \sum_i (Y_i - \hat{f}(x_i))^2 \right\},$$

where Y_i and $\hat{f}$ are independent variables

Bias-variance decomposition of EPE

- **Theorem:** At a given point $\mathbf{x}_0$,

$$\text{EPE} = \sigma^2 + \text{Bias}^2(\hat{f}) + \text{Var}(\hat{f}),$$

  where σ^2 is the variance of $Y|\mathbf{x}_0$
- Thus, expected prediction error consists of three parts:
  - *Irreducible error*: this is beyond our control and would remain even if we were able to estimate f perfectly
  - *Bias* (squared): the difference between $\text{E}\{\hat{f}(\mathbf{x}_0)\}$ and the true value $f(\mathbf{x}_0)$
  - *Variance*: the variance of the estimate $\hat{f}(\mathbf{x}_0)$

Relationship between expected loss and EPE

- Furthermore,

$$EPE = \mathbb{E}L(f, \hat{f}) + \sigma^2$$

- Thus, the expected prediction error and the expected loss are equal up to a constant
- This is attractive because prediction error is easy to evaluate via cross-validation

Cross-validation

- Specifically, we can estimate the expected prediction error with

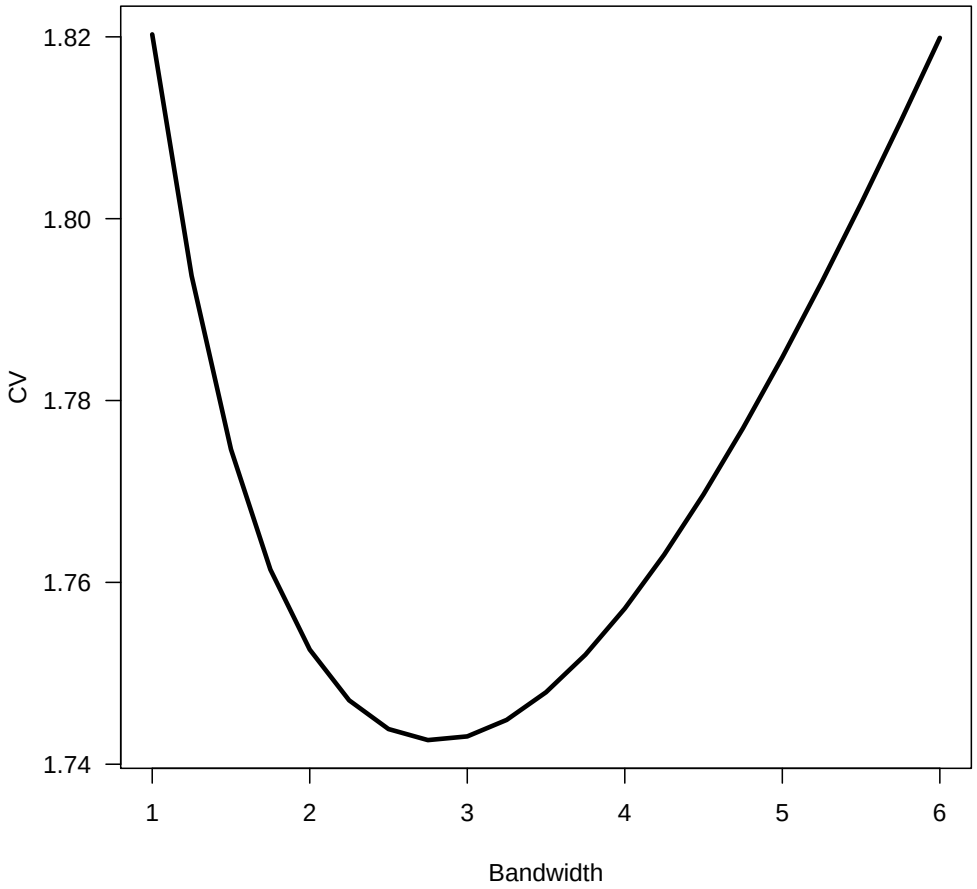
$$CV = \frac{1}{n} \sum_i \left\{ y_i - \hat{f}_{(-i)}(x_i) \right\}^2,$$

  where $\hat{f}_{(-i)}$ is the estimate of f obtained by omitting the pair $\{x_i, y_i\}$
- Furthermore, as we will see, one can obtain a closed form expression for the leave-one-out cross validation score above for any "linear smoother", without actually refitting the model n times

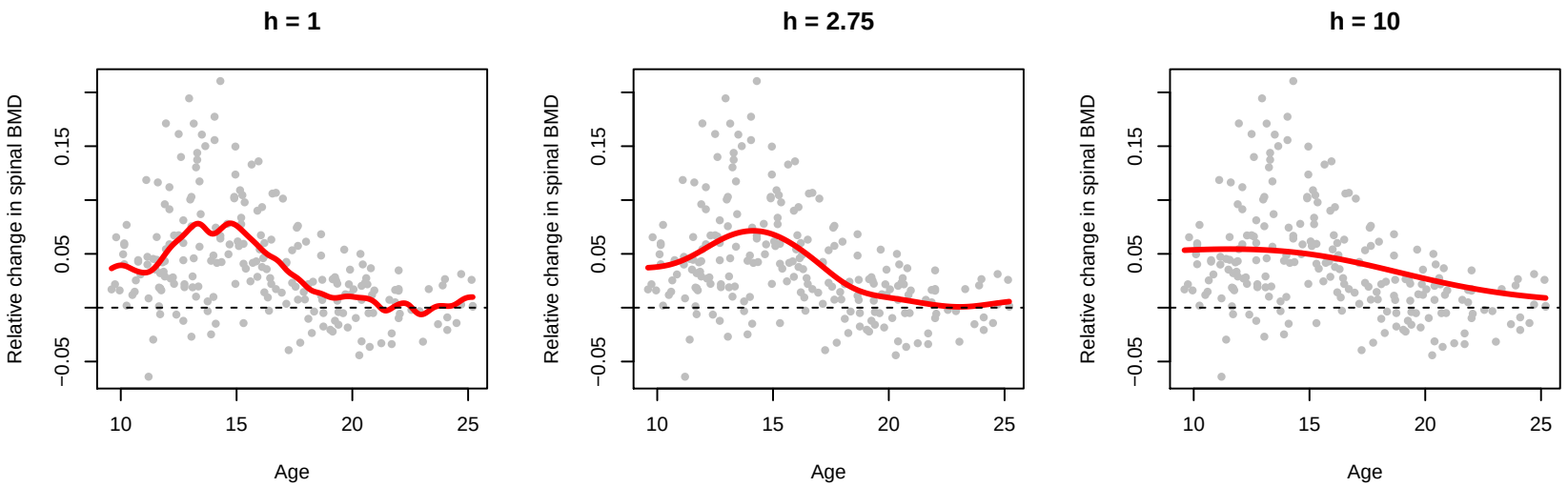
Bone mineral density data

- As an example of a real data set with an interesting change in $\mathbb{E}(y|x)$ as a function of x, we will look at a study of changes in bone mineral density in adolescents
- The outcome is the difference in spinal bone mineral density, taken on two consecutive visits, divided by the average of the two measurements
- Age is the average age over the two visits
- A person's bone mineral density generally increases until the individual is done growing, then remains relatively constant until old age

Cross-validation to choose bandwidth

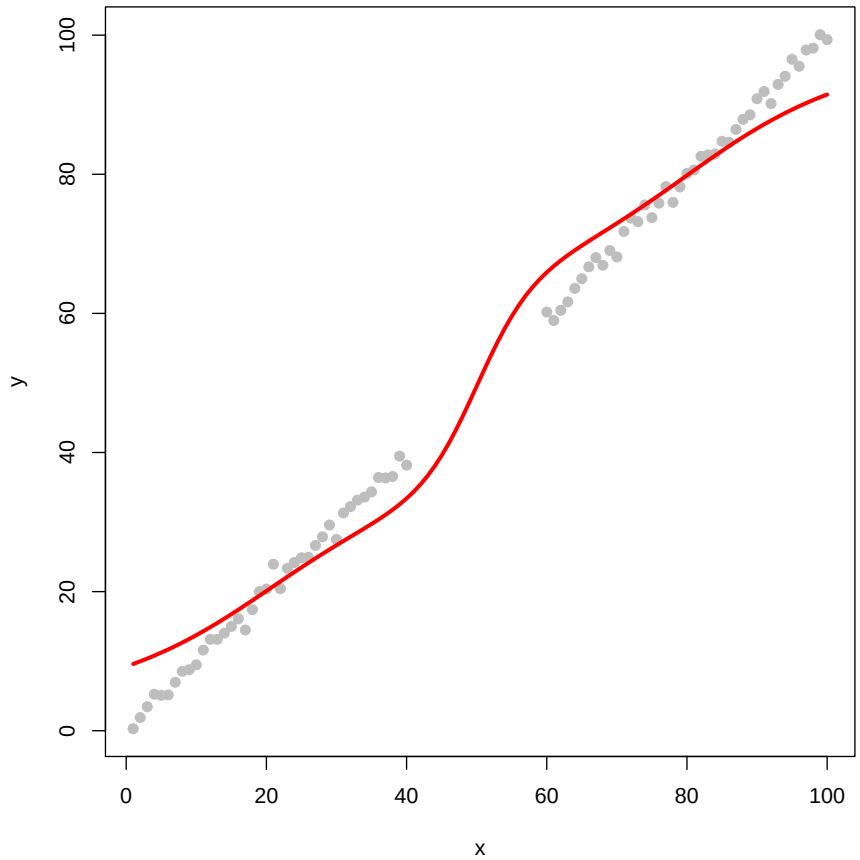


Estimates of the regression function



The problem with kernel weighted averages

Unfortunately, the Nadaraya-Watson kernel estimator suffers from bias, both at the boundaries and in the interior when the x_i's are not uniformly distributed:



Loess

- This arises due to the asymmetry effect of the kernel in these regions
- However, we can (up to first order) eliminate this problem by fitting straight lines locally, instead of constants
- In locally weighted regression, also known as *lowess* or *loess*, we solve a separate weighted least squares problem at each target point x_0:

$$(\hat{\alpha}, \hat{\beta}) = \text{arg min}_{\alpha,\beta} \sum_i K_h(x_0, x_i)(y_i - \alpha - x_i\beta)^2$$

- The estimate is then $\hat{f}(x_0) = \hat{\alpha} + x_0\hat{\beta}$

Loess is a linear smoother

- Let $\mathbf{X}$ denote the $n \times 2$ matrix with ith row $(1, x_i - x_0)$, and $\mathbf{W}$ denote the $n \times n$ diagonal matrix with ith diagonal element $w_i(x_0) = K_h(x_0, x_i)$
- Then,

$$\begin{aligned}\hat{f}(x_0) &= e_1'[\mathbf{X}'\mathbf{W}\mathbf{X}]^{-1}\mathbf{X}'\mathbf{W}\mathbf{y} \\ &= \sum_i l_i(x_0) y_i,\end{aligned}$$

  where $e_1 = (1, 0)' = (1, x_0 - x_0)'$ and it is important to keep in mind that both $\mathbf{X}$ and $\mathbf{W}$ depend implicitly on x_0
- Thus, our estimate is a linear combination of the y_i's; such estimates of f are called *linear smoothers*

Some facts about the linear weights

Homework: Show that the linear weights $\{l_i(x_0)\}$ defined on the previous slide satisfy

(a) $\sum_i l_i(x_0) = 1$ for all x_0

(b) $\sum_i l_i(x_0)(x_i - x_0) = 0$ for all x_0

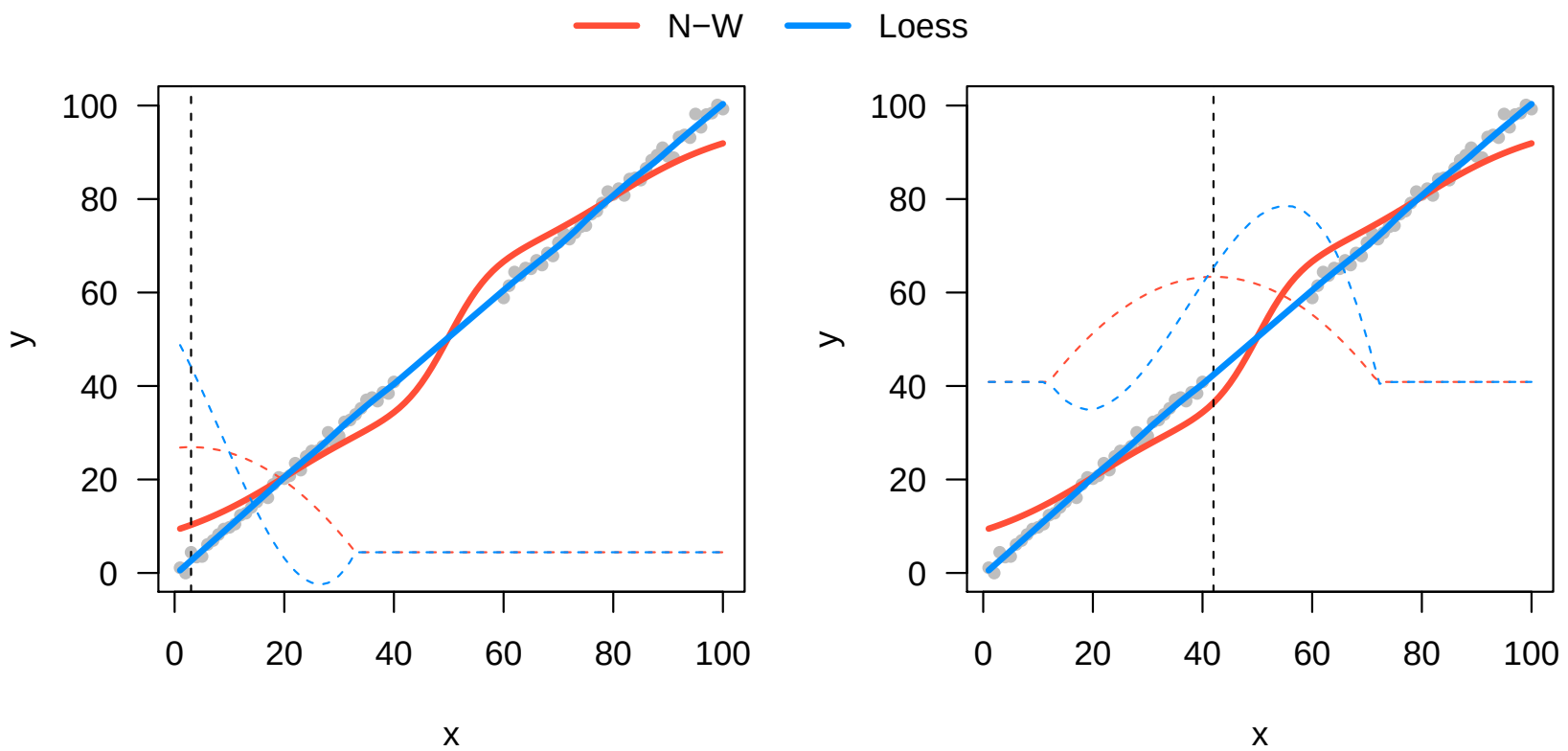
(c) If $K(x_i, x_0) = 0$, then $l_i(x_0) = 0$

(Note that property (a) ensures that the estimate preserves constant curves)

Effective kernels

- The loess approach is similar to the Nadaraya-Watson approach in that both are taking linear combinations of the responses $\{y_i\}$
- In loess, however, the weights $\{l_i(x_0)\}$ are constructed by combining both kernel weighting and least squares operations, forming what is sometimes called an *effective kernel* or *equivalent kernel*
- Before the development of loess, a fair amount of research focused on deriving adaptive modifications to kernels in order to alleviate the bias that we previously discussed
- However, local linear regression automatically modifies the kernel in such a way that this bias is largely eliminated, a phenomenon known as *automatic kernel carpentry*

Automatic kernel carpentry



Loess: Expectation and variance

- At any given target point x_0, $\hat{f}$ is a simple linear combination of random variables
- Thus,

$$\begin{aligned}\mathbb{E}\hat{f}(x_0) &= \sum_i l_i(x_0) f(x_i) \\ \mathbb{V}\hat{f}(x_0) &= \sigma^2 \sum_i l_i(x_0)^2 \\ &= \sigma^2 \left\| l(x_0) \right\|^2,\end{aligned}$$

where $\sigma^2 = \mathbb{V}(y)$

Bias: Loess vs. Nadaraya-Watson

- **Theorem:** Suppose that f is continuously differentiable up to second order and that $K(x, x_0) = 0$ if $|x - x_0| > h$. Then

$$\text{Loess: } \text{Bias}\{f(x_0)\} = O(h^2)$$

$$\text{N-W: } \text{Bias}\{f(x_0)\} = f'(x_0) \sum_i w_i(x_i - x_0) + O(h^2),$$

  where $w_i = K_h(x_i, x_0) / \sum_j K_h(x_j, x_0)$

- The leading term for the bias of the Nadaraya-Watson estimator is referred to as *design bias*; note that it is not present for loess estimators
- In other words, the automatic kernel carpentry that loess performs naturally eliminates design bias, and the resulting estimator is free of bias up to second order

The smoothing matrix

- Recall that loess is a linear smoother; thus,

$$\hat{\mathbf{f}} = \mathbf{L}\mathbf{y},$$

  where $\mathbf{L}$ is called the *smoothing matrix* whose elements consists of the linear weights $l_j(x_i)$
- Having our predictions take on this linear form greatly simplifies leave-one-out cross-validation

Closed form for leave-one-out cross-validation

- **Homework:** Show that

$$\frac{1}{n}\sum_i \left\{ y_i - \hat{f}_{(-i)}(x_i) \right\}^2 = \frac{1}{n}\sum_i \left(\frac{y_i - \hat{f}_i}{1 - l_{ii}} \right)^2$$

- Thus, we have a closed form solution for the leave-one-out cross-validation score that can be obtained from a single fit (*i.e.*, no need to refit anything)

Generalized cross-validation

- An alternative to cross-validation is to replace the individual l_{ii}'s by their average $n^{-1}\sum_i l_{ii} = \nu/n$, where $\nu = \mathrm{tr}(\mathbf{L})$
- This approach is called *generalized cross-validation*:

$$\mathrm{GCV} = \frac{1}{n}\sum_i \left(\frac{y_i - \hat{f}_i}{1-\nu}\right)^2$$

- GCV is equal to CV if all the l_{ii}'s are equal; otherwise, they will be different, although usually quite close

Generalized cross-validation (cont'd)

- Note that, for $x \approx 0$, $1/(1-x)^2 \approx 1 + 2x$; thus,

$$GCV \approx \frac{1}{n}\sum_i (y_i - \hat{f}_i)^2 + \frac{2\hat{\sigma}^2\nu}{n},$$

  where $\hat{\sigma}^2 = n^{-1}\sum_i (y_i - \hat{f}_i)^2$
- If we multiply by n and divide by $\hat{\sigma}^2$, we have that GCV is approximately proportional to $-2\text{loglik} + 2\nu$, the AIC of the fit (treating $\hat{\sigma}^2$ as a known constant)
- Note that, in this approximation, $\nu = \text{tr}(\mathbf{L})$ acts as the *effective degrees of freedom* in the fit

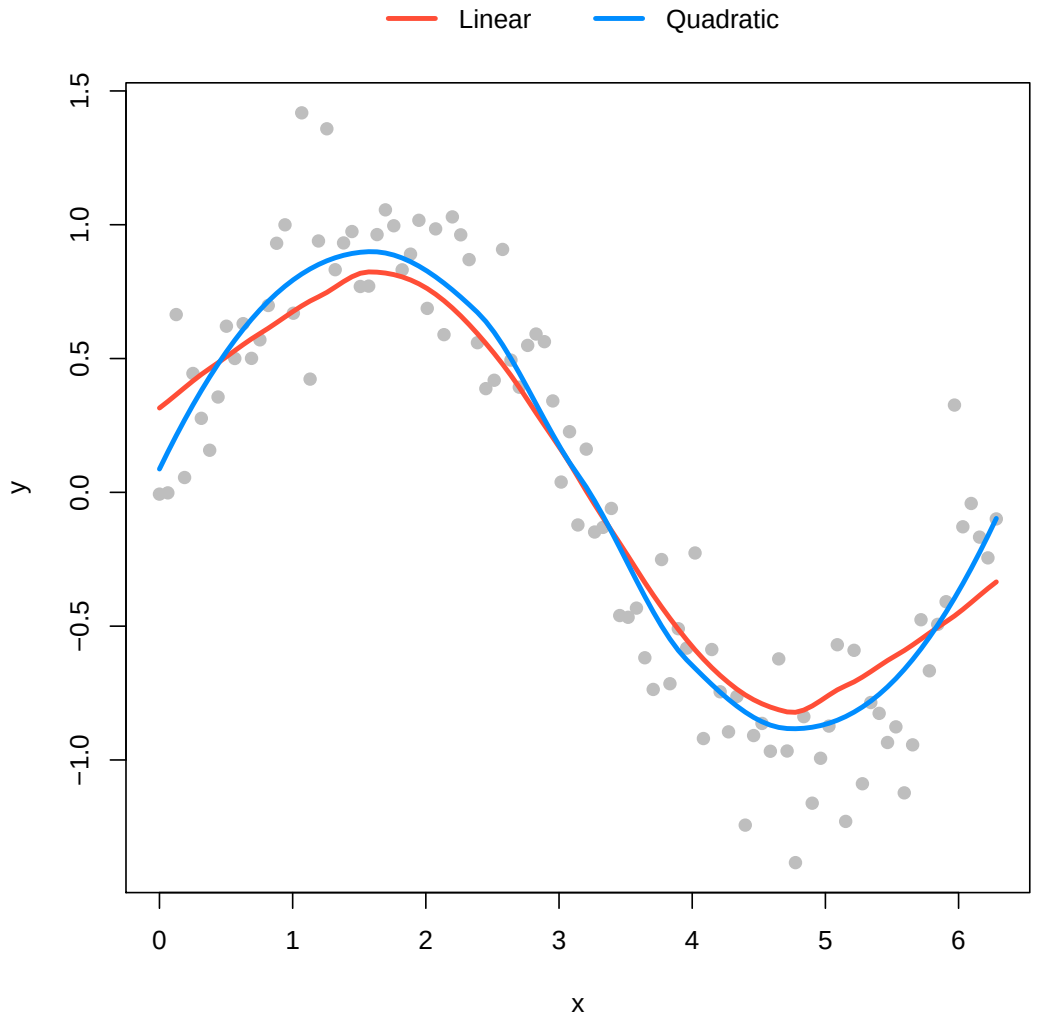
Effective degrees of freedom

- Note that the smoothing matrix is quite similar to the *projection matrix* or *hat matrix* from linear regression ($\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$), for which $\hat{\mathbf{f}} = \mathbf{H}\mathbf{y}$
- In linear regression, $\mathrm{tr}(\mathbf{H})$ is equal to the degrees of freedom; for linear smoothers, $\mathrm{tr}(\mathbf{L})$ defines the effective degrees of freedom (this analogy can be further justified in a number of ways)

Local polynomials

- Of course, one may ask: why stop at local linear regression? Why not add a quadratic term?
- It is not difficult to fit quadratic or even higher order terms: we simply let $\mathbf{X}$ have rows $\left[1, x_i - x_0, \ldots, (x_i - x_0)^d\right]$, where d is the degree of the polynomial
- The weight matrix $\mathbf{W}$, determined entirely by the kernel, remains the same, and we solve separate linear systems of equations for each target point $\mathbf{x}_0$
- By the same mechanism as our earlier theorem, it is straightforward to establish that the bias of a local polynomial fit is $O(h^{d+1})$

Bias due to local linear fitting



Local linear versus local quadratic fitting

- As the figure on the previous slide indicates, local linear models tend to be biased in regions of high curvature, a phenomenon referred to as "trimming the hills and filling in the valleys"
- Higher-order local polynomials correct for this bias, but at the expense of increased variability
- The conventional wisdom on the subject of local linear versus local quadratic fitting says that:
  - Local linear fits tend to be superior at the boundaries
  - Local quadratic fits tend to be superior in the interior
  - Local fitting to higher order polynomials is possible in principle, but rarely necessary in practice

Constant vs. adaptive h

- Our discussion of kernels has focused on keeping the half-width h constant
- An alternative approach is to use a nearest-neighbors type of kernel, in which h changes as a function of x_0 so that the number of points inside $(x_0 - h, x_0 + h)$ remains constant (as we will see, this is the default approach in `R`)
- The smoothing parameter in loess can therefore be made readily interpretable as the fraction of the sample size used in constructing the local fit at any point x_0