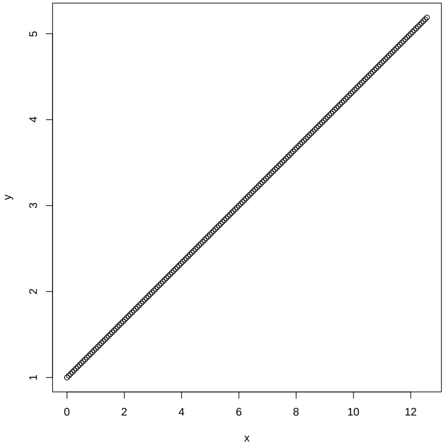
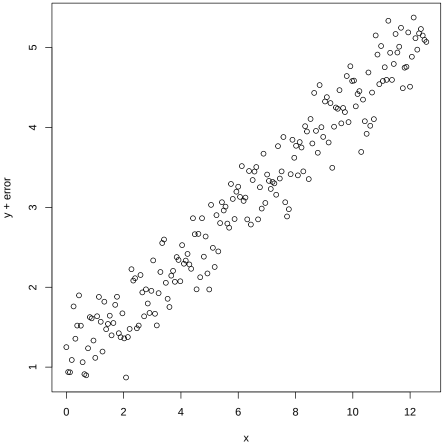
Computer Intensive Statistics and Applications
Chapter 6: Nonparametric Regression

Shaobo Jin

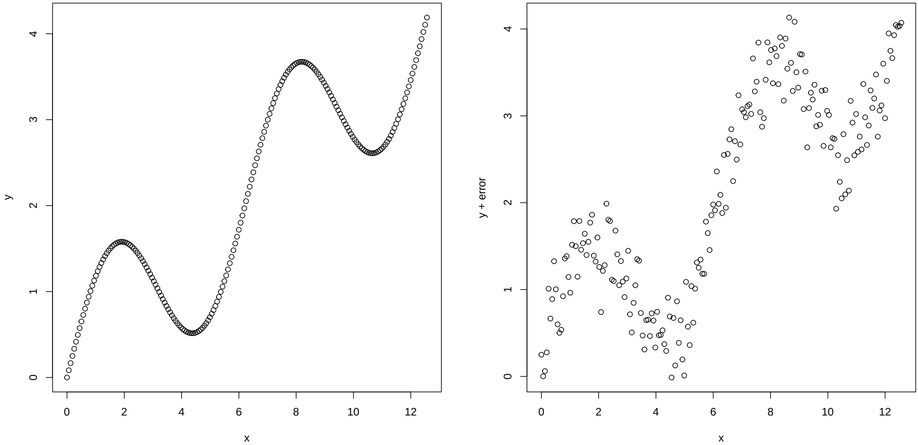
Department of Mathematics

Draw A Line/Curve





Draw A Line/Curve



Conditional Expectation

Suppose that we have data on (Y, X), where Y is the response and X is the covariate/feature. We often want to find a function $g(X)$ such that the mean squared error

$$\min_g \mathrm{E}_{(Y,X)}\left[(Y - g(X))^2\right]$$

is minimized.

The minimizer is

$$m(x) = \mathrm{E}[Y \mid X = x],$$

since

$$\mathrm{E}\left[(Y - g(X))^2\right] = \mathrm{E}\left[(Y - m(X))^2\right] + \mathrm{E}\left[(m(X) - g(X))^2\right].$$

Regression Model

Suppose that we have observed $\{(Y_i, X_i), \ i = 1, ..., n\}$. We often formulate our model as

$$Y_i \quad = \quad m(X_i) + \epsilon_i, \ i = 1, ..., n,$$

where the error terms ϵ_1, ..., ϵ_n are iid with zero expectation and finite variance σ^2.

- We also assume no omitted variables, i.e., X and ϵ are independent.

(Semi-)Parametric Regression

Suppose that we have assumed some parametric form for $m(\cdot)$ that are linear in the parameters, e.g.,

$$\begin{aligned} m(x) &= \beta_0 + \beta_1 x, \\ m(x) &= \beta_0 + \beta_1 x + \beta_2 x^2, \\ m(x) &= \beta_0 + \beta_1 x + \beta_2 \exp(x), \\ m(x) &= \beta_0 + \sum_{j=1}^{p} \beta_j b_j(x). \end{aligned}$$

Knowledge from regression analysis shows that it is often the case that

$$\hat{m}(x) = \sum_{i=1}^{n} W_i(x, X_1, ..., X_n) y_i,$$

as a weighted average of the response variables.

Nonparametric Regression

Consider

$$
\begin{aligned}
m \in \mathcal{S}^k \quad = \quad & \{m : \mathbb{R} \to \mathbb{R},\ m \text{ is continuously differentiable up to order } k, \\
& \text{and } \int \left[m^{(k)}(x)\right]^2 dx < \infty \right\},
\end{aligned}
$$

where we will not completely specify the parametric form of m.

Many estimators turn out to be also of the form

$$
\hat{m}(x) \quad = \quad \sum_{i=1}^{n} W_i(x, X_1, ..., X_n)\, y_i.
$$

Kernel Regression: Basic Idea

Suppose that we want to model $m(x) = \mathrm{E}(Y \mid X = x)$.

- The observed $\{x_i\}$ that are close to x should carry more information about Y than $\{x_i\}$ that are far away.
- More informative $\{x_i\}$ should be given higher weights in

$$\hat{m}(x) = \sum_{i=1}^{n} W_i(x, X_1, ..., X_n) y_i.$$

- To construct weights, we can use

$$\mathrm{E}(Y \mid X = x) = \int y \frac{f_{(X,Y)}(x, y)}{f_X(x)} dy.$$

Nadaraya-Watson Estimator

Suppose that we use kernel density estimation to estimate $f_{(X,Y)}(x,y)$ and $f_X(x)$ as

$$\begin{aligned}
\hat{f}_{(X,Y)}(x,y) &= \frac{1}{nh_xh_y}\sum_{i=1}^{n} K_x\left(\frac{x-X_i}{h_x}\right) K_y\left(\frac{y-Y_i}{h_y}\right), \\
\hat{f}_X(x) &= \frac{1}{nh_x}\sum_{i=1}^{n} K_x\left(\frac{x-X_i}{h_x}\right).
\end{aligned}$$

Then,

$$\hat{m}(x) = \int y \frac{\hat{f}_{(X,Y)}(x,y)}{\hat{f}_X(x)} dy = \frac{\sum_{i=1}^{n} K_x\left(\frac{x-X_i}{h_x}\right) Y_i}{\sum_{i=1}^{n} K_x\left(\frac{x-X_i}{h_x}\right)},$$

which is known as the Nadaraya-Watson estimator.

- A large h typically means a large bias and a small variance.
- A small h typically means a small bias and a high variance.

Nadaraya-Watson Estimator: MSE

If the density of X and $m(x)$ are smooth enough, then

$$\begin{aligned} \mathrm{E}\left[\hat{m}(x)\right] - m(x) &= \frac{1}{2} h^2 \mu_2(K) \left[2m'(x)\frac{f'(x)}{f(x)} + m''(x)\right] + o\left(h^2\right), \\ \mathrm{Var}\left[\hat{m}(x)\right] &= \frac{\sigma^2 \|K\|_2^2}{nhf(x)} + o\left(\frac{1}{nh}\right), \\ \mathrm{E}\left(\left[\hat{m}(x) - m(x)\right]^2\right) &= \frac{\sigma^2 \|K\|_2^2}{nhf(x)} + \frac{h^4}{4}\mu_2^2(K)\left[2m'(x)\frac{f'(x)}{f(x)} + m''(x)\right]^2 \\ &\quad + o\left(\frac{1}{nh}\right) + o\left(h^4\right), \end{aligned}$$

when $h \to 0$ and $nh \to \infty$.

- It also suggests that $\hat{m}(x)$ is a consistent estimator of $m(x)$.

Nadaraya-Watson Estimator: Another Perspective

Suppose that, for a fixed x, we use $c(x)$ to predict the value of y.

1. The value c is chosen to minimize
$$L = \sum_{i=1}^{n} (c - Y_i)^2 .$$
Then, $\hat{c} = \bar{Y}$.
2. The value c is chosen to minimize
$$L = \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)(c - Y_i)^2 .$$
Then, $\hat{c}$ is the Nadaraya-Watson estimator.

Confidence Interval

For a fixed x, if the bandwidth satisfies $h = cn^{-1/5}$ for a constant c, we would expect

$$n^{2/5}\left\{\hat{m}_h(x) - m(x)\right\} \overset{d}{\to} N\left(b(x),\, v^2(x)\right),$$

where

$$\begin{aligned} b(x) &= \frac{1}{2}c^2\mu_2(K)\left[2m'(x)\frac{f'(x)}{f(x)} + m''(x)\right], \\ v^2(x) &= \frac{\sigma^2\|K\|_2^2}{cf(x)}. \end{aligned}$$

However, the bias depends on unknown quantities. Hence, we can only obtain a confidence interval for $\mathrm{E}\left[\hat{m}_h(x)\right]$, not $m(x)$.

Pointwise Confidence Band

In the spirit of central limit theorem, the distribution of $\sqrt{n}\left\{\hat{m}_h(x) - \mathrm{E}\left[\hat{m}_h(x)\right]\right\}$ can be approximated by

$$N\left(0,\ \frac{\sigma^2 \|K\|_2^2}{nhf(x)}\right).$$

Hence, an asymptotic interval for $\mathrm{E}\left[\hat{m}_h(x)\right]$ is

$$\hat{m}_h(x) \pm \lambda_{1-\alpha/2}\sqrt{\frac{\hat{\sigma}^2(x)}{nh\hat{f}_h(x)}\|K\|_2^2},$$

where

$$\hat{\sigma}^2(x) = \frac{\sum_{i=1}^n K_x\left(\frac{x-X_i}{h_x}\right)\left[Y_i - \hat{m}_h(x)\right]^2}{\sum_{i=1}^n K_x\left(\frac{x-X_i}{h_x}\right)}.$$

Constant Local Approximation

The Nadaraya-Watson estimator minimizes

$$L = \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)(Y_i - c)^2,$$

where $\{X_i\}$ that are close to x receive more weights.

The constant c can be interpreted as the function is a constant in a neighborhood of x, i.e., for all u that is close to x,

$$m(u) \approx m(x).$$

We can easily generalize the idea from local constant to local polynomials.

Local Polynomial Approximation

1. Assume that for all u that is close to x,

$$m(u) \approx m(x) + \beta_1(x)(u-x).$$

The local linear estimator minimizes

$$L = \sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right)\left[Y_i - m(x) - \beta_1(x)(X_i - x)\right]^2.$$

2. If $m(u) \approx m(x) + \beta_1(x)(u-x) + \beta_2(x)(u-x)^2$, the local quadratic estimator minimizes

$$L = \sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right)\left[Y_i - m(x) - \beta_1(x)(X_i - x) - \beta_2(x)(X_i - x)^2\right]^2.$$

A Useful Lemma

Without proof we state the following lemma.

Lemma

Suppose that y is an $n \times 1$ vector, Z is an $n \times p$ matrix, γ is a $p \times 1$ vector, and W is an $n \times n$ symmetric matrix. The gradient vector of

$$L = (y - Z\gamma)^T W (y - Z\gamma)$$

is given by

$$\frac{\partial L}{\partial \gamma} = -2Z^T W (y - Z\gamma),$$

and the unique minimizer of L is given by

$$\gamma = \left(Z^T W Z\right)^{-1} Z^T W y,$$

provided that the inverse of $Z^T W Z$ exists.

Local Linear Estimator

Let

$$y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}, Z = \begin{pmatrix} 1 & X_1 - x \\ \vdots & \vdots \\ 1 & X_n - x \end{pmatrix}, \gamma = \begin{pmatrix} m(x) \\ \beta_1(x) \end{pmatrix}, W = \operatorname{diag}\left\{ K\left(\frac{x - X_i}{h} \right) \right\}.$$

The minimizer of

$$L = \sum_{i=1}^{n} K\left(\frac{x - X_i}{h} \right) \left[Y_i - m(x) - \beta_1(x)(X_i - x) \right]^2$$

is given by

$$\hat{\gamma} = \begin{bmatrix} \hat{m}(x) \\ \hat{\beta}_1(x) \end{bmatrix} = \left(Z^T W Z \right)^{-1} Z^T W y.$$

The local linear estimator is given by $\hat{m}(x)$.

Local Linear Estimator: MSE

The derivation of bias and variance of the local linear estimator becomes demanding. The results are

$$\begin{aligned}
\mathrm{E}\left[\hat{m}(x)\right] - m(x) &= \frac{1}{2} h^2 \mu_2(K)\, m''(x) + o\left(h^2\right), \\
\mathrm{Var}\left[\hat{m}(x)\right] &= \frac{\sigma^2 \|K\|_2^2}{nhf(x)} + o\left(\frac{1}{nh}\right), \\
\mathrm{E}\left(\left[\hat{m}(x) - m(x)\right]^2\right) &= \frac{\sigma^2 \|K\|_2^2}{nhf(x)} + \frac{h^4}{4} \mu_2^2(K) \left[m''(x)\right]^2 \\
&\quad + o\left(\frac{1}{nh}\right) + o\left(h^4\right),
\end{aligned}$$

when $h \to 0$ and $nh \to \infty$.

- It also suggests that $\hat{m}(x)$ is a consistent estimator of $m(x)$.

Boundary Bias

The MSE of Nadaraya-Watson estimator is

$$\begin{aligned} \mathrm{E}\left(\left[\hat{m}(x) - m(x)\right]^2\right) &= \frac{\sigma^2 \|K\|_2^2}{nhf(x)} + \frac{h^4}{4}\mu_2^2(K)\left[2m'(x)\frac{f'(x)}{f(x)} + m''(x)\right]^2 \\ &\quad + o\left(\frac{1}{nh}\right) + o\left(h^4\right). \end{aligned}$$

- The Nadaraya-Watson estimator is prone to boundary bias at the boundary points, typically of order h.
- The local linear estimator is much less prone to boundary bias, typically of order h^2.
- The local linear estimator also depends less on $f(x)$ since $f'(x)/f(x)$ is not included in the bias term.

Residual Bootstrap

Algorithm 1: Residual bootstrap for regression

Fit the regression model using observed data ;
Obtain the residuals $\hat{\epsilon} = Y - \hat{m}(X)$;
Normalize the residuals, if needed, and obtain $\tilde{\epsilon}$ such that $n^{-1}\sum_{i=1}^{n}\tilde{\epsilon}_i = 0$;
for *each integer j from 1 to B* **do**
  Draw a random sample ϵ_j^* of size n from the empirical distribution of $\tilde{\epsilon}$, i.e., sample with replacement from $\tilde{\epsilon}$;
  Calculate the bootstrap response $Y_j^* = \hat{m}(X) + \epsilon_j^*$;
  Obtain the bootstrap estimator $\hat{m}_j^*$;
end

(Semi-)Parametric Regression

Suppose that we want to approximate $m(x)$ by a linear function (in β) as

$$g(x) = \sum_{k=1}^{K} \beta_k g_k(x),$$

where the function forms of $\{g_k(x)\}$ are pre-determined. For example

$$\begin{aligned}
m(x) &= \beta_0 + \beta_1 x, \\
m(x) &= \beta_0 + \beta_1 x + \beta_2 x^2, \\
m(x) &= \beta_0 + \beta_1 x + \beta_2 \exp(x), \\
m(x) &= \beta_0 + \sum_{j=1}^{p} \beta_j b_j(x).
\end{aligned}$$

Linear Regression

It is often the case that we want to minimize

$$L = \sum_{i=1}^{n} \left[Y_i - \sum_{k=1}^{K} \beta_k g_k\left(x\right) \right]^2,$$

where the model $\sum_{k=1}^{K} \beta_k g_k\left(x\right)$ is linear in β. The minimizer is

$$\hat{\beta} = \left(G^T G\right)^{-1} G^T Y,$$

where $Y^T = \begin{bmatrix} Y_1 & \cdots & Y_n \end{bmatrix}$, and

$$\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_K \end{bmatrix}, \ G = \begin{bmatrix} g_1\left(X_1\right) & \cdots & g_M\left(X_1\right) \\ \vdots & \ddots & \vdots \\ g_1\left(X_n\right) & \cdots & g_M\left(X_n\right) \end{bmatrix}$$

Penalization

It is also often the case that we want to minimize

$$L = \sum_{i=1}^{n} \left[Y_i - \sum_{k=1}^{K} \beta_k g_k \left(x_i \right) \right]^2 + \lambda \rho \left(\beta \right),$$

where $\rho\left(\beta\right)$ is a penalization term, and $\lambda > 0$ is a tuning parameter.

- Penalization can introduce some bias but greatly reduce the variance, so the MSE becomes smaller.
- Penalization works even when $n < K$.

Ridge Regression

The ridge regression minimizes

$$L = \sum_{i=1}^{n} \left[Y_i - \beta_0 - \sum_{k=1}^{K} \beta_k g_k\left(X_i\right) \right]^2 + \lambda \sum_{k=1}^{K} \beta_k^2,$$

where the intercept β_0 is unpenalized.

- The intercept can be estimated by

$$\hat{\beta}_0 = \frac{1}{n} \sum_{i=1}^{n} Y_i - \sum_{k=1}^{K} \hat{\beta}_k \left[\frac{1}{n} \sum_{i=1}^{n} g_k\left(X_i\right) \right].$$

- If we center all covariates/features such that $n^{-1} \sum_{i=1}^{n} g_k\left(X_i\right) = 0$ for all k, then $\hat{\beta}_0 = \bar{Y}$.
- Without loss of generality, we often center both Y and G, and don't include the intercept.

Ridge Regression: No Intercept

Suppose that all variables have been centered, and we consider

$$\begin{aligned} L &= \sum_{i=1}^{n}\left[Y_i - \sum_{k=1}^{K}\beta_k g_k\left(X_i\right)\right]^2 + \lambda\sum_{k=1}^{K}\beta_k^2 \\ &= \left(Y - G\beta\right)^T\left(Y - G\beta\right) + \lambda\beta^T\beta. \end{aligned}$$

By the lemma above, we get

$$\frac{\partial L}{\partial \beta} = -2G^T\left(y - G\beta\right) + 2\lambda\beta,$$

and the ridge estimator is

$$\hat{\beta}^{\text{ridge}} = \left(G^T G + \lambda I\right)^{-1} G^T Y.$$

Ridge Estimator

$$\hat{\beta}^{\text{ridge}} = \left(G^T G + \lambda I\right)^{-1} G^T Y.$$

- If $\lambda = 0$ and the inverse of $G^T G$ exists, the ridge estimator reduces to the ordinary least squares estimator.
- For any $\lambda > 0$, $G^T G + \lambda I > 0$ (positive definite) and the inverse exists, whereas the inverse of $G^T G$ may not exist.
- The ridge estimator is simply the least squares estimator if we augment our data to

$$\tilde{Y} = \begin{bmatrix} Y_{n\times 1} \\ 0_{K\times 1} \end{bmatrix}, \qquad \tilde{G} = \begin{bmatrix} G_{n\times K} \\ \sqrt{\lambda} I_{K\times K} \end{bmatrix}.$$

Bias and Variance of Ridge Estimator

Suppose that the eigendecomposition of G^TG is $G^TG = UDU^T$, where D is a diagonal matrix with diagonal entries $\{d_k\}$.

1. The bias is
$$\begin{aligned} \mathrm{E}\left[\hat{\beta}^{\text{ridge}} \mid X\right] - \beta &= \lambda \left(G^TG + \lambda I\right)^{-1} \beta \\ &= \lambda U \left(D + \lambda I\right)^{-1} U^T \beta. \end{aligned}$$

2. The variance satisfies
$$\operatorname{tr}\left\{\operatorname{Var}\left[\hat{\beta}^{\text{ridge}} \mid X\right]\right\} = \sigma^2 \sum_{k=1}^{K} \frac{d_k}{\left(d_k + \lambda\right)^2},$$
if we assume $\operatorname{Var}\left(Y \mid X\right) = \sigma^2 I$.

A general trend is that the bias increases and the variance decreases as λ increases.

MSE of Ridge Estimator

The general bias-variance decomposition still holds for a random vector:

$$\begin{aligned}\text{MSE} &= \mathrm{E}\left[\left(\hat{\beta}^{\text{ridge}}-\beta\right)^T\left(\hat{\beta}^{\text{ridge}}-\beta\right) \mid X\right] \\ &= \operatorname{tr}\left\{\operatorname{Var}\left[\hat{\beta}^{\text{ridge}} \mid X\right]\right\} \\ &\quad + \left(\mathrm{E}\left[\hat{\beta}^{\text{ridge}} \mid X\right]-\beta\right)^T\left(\mathrm{E}\left[\hat{\beta}^{\text{ridge}} \mid X\right]-\beta\right).\end{aligned}$$

Using the above bias and variance, the MSE becomes

$$\text{MSE}_\lambda\left(\hat{\beta}^{\text{ridge}}\right) = \sigma^2\sum_{k=1}^{K}\frac{d_k}{(d_k+\lambda)^2}+\sum_{k=1}^{K}\left(\frac{\lambda\left[U^T\beta\right]_k}{d_k+\lambda}\right)^2,$$

where $\left[U^T\beta\right]_k$ is the kth entry of the vector $U^T\beta$.

Lasso

The least absolute shrinkage and selection operator (lasso) minimizes

$$L = \sum_{i=1}^{n} \left[Y_i - \beta_0 - \sum_{k=1}^{K} \beta_k g_k \left(X_i \right) \right]^2 + \lambda \sum_{k=1}^{K} \left| \beta_k \right|.$$

Similar to the ridge estimator, we can consider the model without the intercept for demeaned data as

$$L = \sum_{i=1}^{n} \left[Y_i - \sum_{k=1}^{K} \beta_k g_k \left(X_i \right) \right]^2 + \lambda \sum_{k=1}^{K} \left| \beta_k \right|.$$

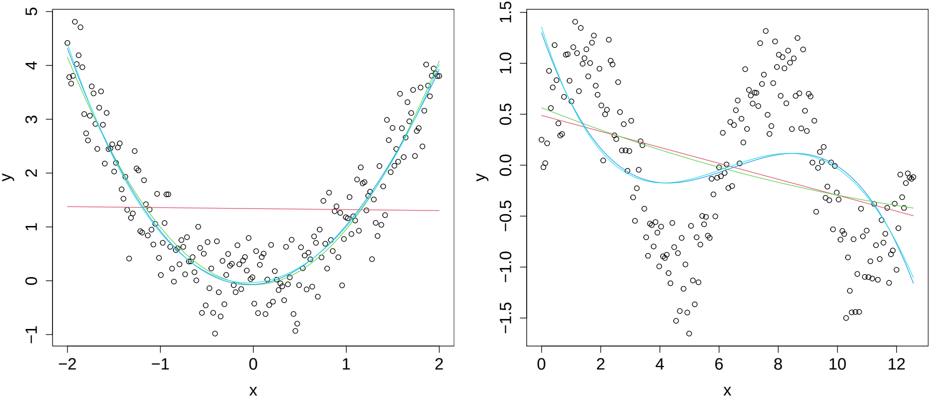
- We often write $\sum_{k=1}^{K} \left| \beta_k \right| = \left\| \beta \right\|_1$.

Ridge and Lasso

- Bias-variance trade-off : If $\lambda = 0$, the usual estimator is obtained. If $\lambda > 0$, the bias-variance trade-off occurs.
- Shrinkage:
  - $\hat{\beta}$ obtained for a positive λ is shrunk towards zero, so $\hat{\beta}$ is actually a function of λ.
  - The ridge never produces exact zero estimates, but the lasso can produce exact zero estimates (variable selection).
- High dimensional: They work when $n < K$.

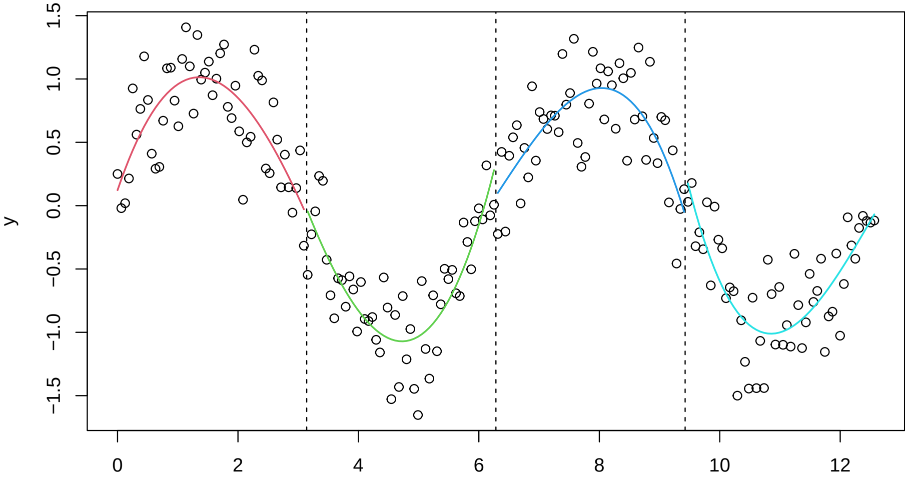
Specify Mean Function

It is not always easy to specify the closed form expression of the mean function $m(x)$.



Alternative: Piecewise Polynomial

We partition the data into several parts and fit polynomials to each part separately.



Piecewise polynomial

A piecewise polynomial is obtaind by

1. partitioning the range of x into contiguous intervals using the knots,
2. Between every two consecutive knots, fitting a polynomial model (in x) to the data points in the interval.

In practice, it is common to use the cubic polynomials (with degree 3 and order 4).

Example: Piecewise Cubic Polynomial

Consider two knots ξ_1 and ξ_2. We fit three cubic polynomials

$$\begin{aligned}
\text{for } x < \xi_1: \quad & m_1(x) = \beta_0^{(1)} + \beta_1^{(1)} x + \beta_2^{(1)} x^2 + \beta_3^{(1)} x^3, \\
\text{for } \xi_1 \le x < \xi_2: \quad & m_2(x) = \beta_0^{(2)} + \beta_1^{(2)} x + \beta_2^{(2)} x^2 + \beta_3^{(2)} x^3, \\
\text{for } x \ge \xi_2: \quad & m_3(x) = \beta_0^{(3)} + \beta_1^{(3)} x + \beta_2^{(3)} x^2 + \beta_3^{(3)} x^3.
\end{aligned}$$

It is the same as

$$\begin{aligned}
\mathrm{E}(Y \mid X = x) = \; & 1(x < \xi_1)\left(\beta_0^{(1)} + \beta_1^{(1)} x + \beta_2^{(1)} x^2 + \beta_3^{(1)} x^3\right) \\
& + 1(\xi_1 \le x < \xi_2)\left(\beta_0^{(2)} + \beta_1^{(2)} x + \beta_2^{(2)} x^2 + \beta_3^{(2)} x^3\right) \\
& + 1(x \ge \xi_2)\left(\beta_0^{(3)} + \beta_1^{(3)} x + \beta_2^{(3)} x^2 + \beta_3^{(3)} x^3\right).
\end{aligned}$$

However, we still cannot guarantee continuity. We want our fitted model to be continuous and smooth (sufficiently many continuous derivatives).

Cubic Spline

In order to produce a continuous and smooth fitted curve, we will impose the following constraints.

1. The fitted curve must be continuous everywhere, including the knots.
2. The fitted curve has continuous first and second order derivatives.

If piecewise cubic polynomials are used, then we have a cubic spline.

Cubic Spline: Restrictions

In order to achieve continuity, we need

$$m_1(\xi_1) = m_2(\xi_1) \quad \text{and} \quad m_2(\xi_2) = m_3(\xi_2).$$

In order to achieve smoothness, we need

$$\frac{dm_1(\xi_1)}{dx} = \frac{dm_2(\xi_1)}{dx} \quad \text{and} \quad \frac{dm_2(\xi_2)}{dx} = \frac{dm_3(\xi_2)}{dx},$$
$$\frac{d^2m_1(\xi_1)}{dx^2} = \frac{d^2m_2(\xi_1)}{dx^2} \quad \text{and} \quad \frac{d^2m_2(\xi_2)}{dx^2} = \frac{d^2m_3(\xi_2)}{dx^2},$$

Example: Cubic Spline

Consider two knots ξ_1 and ξ_2. With the continuity and smoothness requirements, we get

$$\begin{aligned} m(x) &= \beta_0^{(3)} + \left(\beta_1^{(2)} - \beta_1^{(1)}\right)\xi_1^3 + \left(\beta_1^{(3)} - \beta_1^{(2)}\right)\xi_2^3 \\ &\quad +\beta_1^{(1)}x + \beta_2^{(1)}x^2 + \beta_3^{(1)}x^3 \\ &\quad + \left(\beta_3^{(2)} - \beta_3^{(1)}\right)\left[\max\left(0, x-\xi_1\right)\right]^3 \\ &\quad + \left(\beta_3^{(3)} - \beta_3^{(2)}\right)\left[\max\left(0, x-\xi_2\right)\right]^3. \end{aligned}$$

It is the same as regress Y on the intercept, x, x^2, x^3, $\left[\max\left(0, x-\xi_1\right)\right]^3$, and $\left[\max\left(0, x-\xi_2\right)\right]^3$.

Cubic Spline

Suppose that we have K knots (excluding the lower and upper limits of the range). Then, there are

$$4(K+1) - K - K - K = K + 4$$

free parameters to be estimated in cubic spline. That is, a cubic spline with K knots has $K+4$ degrees of freedom. That is, the cubic spline is equivalent to

$$m(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \sum_{k=1}^{K} \beta_{k+3} \left[\max(0, x - \xi_k)\right]^3.$$

In general, spline is a function defined by piecewise polynomials with continuity and smoothness conditions.

Still Not Necessarily Enough

- The fit of a cubic spline is often poor for very small or very large x values, due to the lack of information and large variation.
- We need to impose additional boundary constraints, i.e. the curve is linear in the region where X is smaller than or larger than the observed values. Then we have a natural spline.
- If we impose the boundary constraints to a cubic spline, we have a natural cubic spline.

Natural Cubic Spline

If we approximate $m(x)$ by a natural cubic spline, then

$$m(x) \approx \beta_0 + \beta_1 x + \sum_{k=1}^{K-2} \beta_{k+1} \left[d_k(x) - d_{K-1}(x)\right],$$

where

$$d_k(x) = \frac{\left[\max(0, x-\xi_k)\right]^3 - \left[\max(0, x-\xi_K)\right]^3}{\xi_K - \xi_k}.$$

More General View: Basis Expansion

From the above examples, we have

$$m(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \sum_{k=1}^{K} \beta_{k+3} \left[\max(0, x - \xi_k)\right]^3,$$

$$m(x) = \beta_0 + \beta_1 x + \sum_{k=1}^{K-2} \beta_{k+1} \left[d_k(x) - d_{K-1}(x)\right].$$

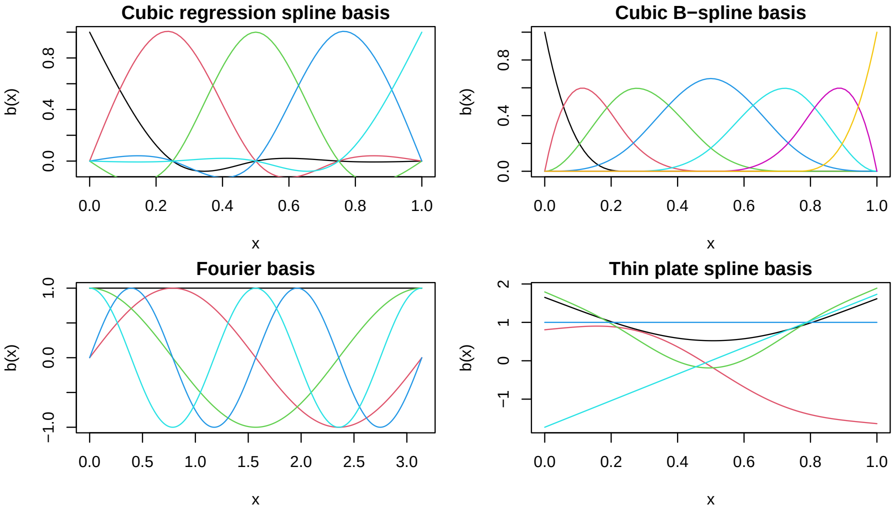
It is equivalent to using some basis functions and performing a "global" regression.

- We choose a series of functions $\{b_k(x)\}$ and use global data to fit

$$m(x) = \sum_k \beta_k b_k(x).$$

- $b_k(x)$'s are the basis functions and $\sum_k \beta_k b_k(x)$ is the basis expansion.

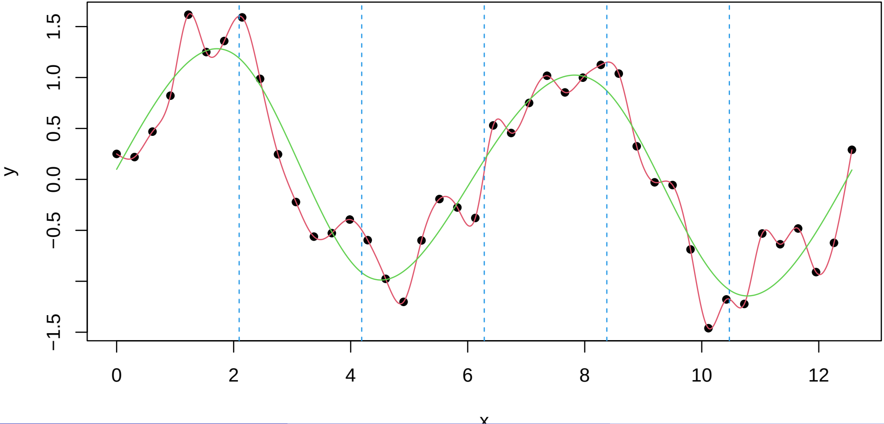
Choice of Basis Functions



Overfitting

In practice, we choose $K \ll n$ interior knots and find a piecewise polynomial $f(x)$ that minimizes

$$\text{loss function} + \text{penalty for wiggliness.}$$



Smoothing Spline

When we fit a model to the data, we want a good fit but also smooth. One way is to minimize

$$\text{loss function} + \text{penalty for wiggliness}$$

to avoid overfitting.

- For example, we can minimize

$$\sum_{i=1}^{n} [y_i - m(x_i)]^2 + \lambda \int \left[m''(u)\right]^2 du.$$

- The minimizer is call a smoothing spline.
- In fact, the minimizer that satisfies

$$m(x_i) = y_i, \text{ and } m''(a) = m''(b) = 0,$$

  is a natural cubic spline with interior knots at the observed x_1, .., x_n values, where a and b are known finite boundary points.

Ridge Regression Perspective

Suppose that

$$m\left(x\right)=\sum_{k}\beta_{k}b_{k}\left(x\right)=B^{T}\left(x\right)\beta.$$

Then

$$\begin{aligned}&\sum_{i=1}^{n}\left[y_{i}-m\left(x_{i}\right)\right]^{2}+\lambda\int\left[m''\left(u\right)\right]^{2}du\\=&\sum_{i=1}^{n}\left[y_{i}-B^{T}\left(x_{i}\right)\beta\right]^{2}+\lambda\beta^{T}\left[\int\frac{d^{2}B\left(u\right)}{du^{2}}\left[\frac{d^{2}B\left(u\right)}{du^{2}}\right]^{T}du\right]\beta,\end{aligned}$$

which is simply a ridge regression with regression coefficients β.

Additive Model

- To account for non-linearity, we can assume

$$m(x) = m(x_1, x_2, \cdots, x_p),$$

  where the function form $m()$ is estimated from the data.
- However, this formulation suffers from curse of dimensionality.
- In practice, we often consider the generalized additive model (GAM), such as

$$m(x_1, x_2, \cdots, x_p) = m_1(x_1) + m_2(x_2) + m_3(x_3) + \cdots + m_p(x_p),$$
$$m(x_1, x_2, \cdots, x_p) = m_1(x_1) + m_{2,3}(x_2, x_3) + \cdots + m_p(x_p).$$

- Roughly speaking, GAM uses basis expansions to approximate unknown functions forms, and uses some penalty terms to control the wiggliness.

Tuning Parameter

Many procedures in our course includes a tuning parameter.

1. bandwidth h in kernel density estimation,
2. bandwidth h in Nadaraya-Watson estimator,
3. bandwidth h in local polynomial regression,
4. shrinkage parameter λ in ridge and lasso.
5. smoothing parameter λ in spline estimator.

General Selection Methods

A general principle is to specify some criterion function and choose the tuning parameter that optimizes such criterion function. For example,

- find the AMISE and choose the tuning parameter that minimizes such AMISE,
- find the tuning parameter value using cross validation.

Cross Validation (CV) Algorithm

Algorithm 2: One version of cross validation

```
Specify a grid of candidate tuning parameter values ;
Specify a criterion function ;
Randomly split the data set into K nonoverlapping groups (K-fold CV) or
  split the data set into n groups (leave-one-out CV, aka jackknife) ;
for k = 1 in 1 : K do
    Take the kth group as test set and the remaining groups as training
      set ;
    while for each tuning parameter value do
        Fit it on the training set and evaluate it on the test set ;
        Retain the performance of tuning parameter (e.g., MSE, AMISE,
          misclassification error, log-likelihood) ;
    end
end
Summarize the performance (e.g., average across K groups) ;
Choose the tuning parameter that performs the best ;
Refit to the entire data set using the chosen tuning parameter value ;
```

Recall: Kernel Density Using AMISE

Let $\hat{f}_h(x) = \hat{f}_h(x; X_1, ..., X_n)$ be the kernel density estimator of $f(x)$, where the bandwidth h needs to be specified.

- The AMISE is

$$\text{AMISE}\left(\hat{f}_h\right) = \frac{1}{nh}\|K\|_2^2 + \frac{1}{4}h^4\mu_2^2(K)\left\|f''(x)\right\|_2^2.$$

- Minimizing AMISE $\left(\hat{f}\right)$ as a function in h yields

$$h_0 = \left[\frac{\|K\|_2^2}{n\mu_2^2(K)\|f''(x)\|_2^2}\right]^{1/5},$$

which depends on the known quantity $\|f''(x)\|_2^2$.

Recall: Kernel Density Using Cross Validation

The integrated squared error (ISE) is

$$\int \left[\hat{f}(x) - f(x)\right]^2 dx = \int \hat{f}_h^2(x)\, dx - 2\int \hat{f}_h(x) f(x)\, dx + \int f^2(x)\, dx.$$

We can estimate the second integral by $n^{-1}\sum_{i=1}^{n} \hat{f}_{h,-i}(X_i)$, where

$$\hat{f}_{h,-i}(x) = \frac{1}{(n-1)h}\sum_{j\neq i} K\left(\frac{x - X_j}{h}\right).$$

For each h in a pre-specified grid of candidate bandwidths, we compute

$$\mathrm{CV}(h) = \int \hat{f}_h^2(x)\, dx - \frac{2}{n}\sum_{i=1}^{n} \hat{f}_{h,-i}(X_i).$$

The CV bandwidth minimizes $\mathrm{CV}(h)$.

Nadaraya-Watson Estimator: Minimum MSE

The MSE of the Nadaraya-Watson estimator is

$$\begin{aligned} \mathrm{E}\left(\left[\hat{m}(x) - m(x)\right]^2\right) &= \frac{\sigma^2 \|K\|_2^2}{nhf(x)} + \frac{h^4}{4}\mu_2^2(K)\left[2m'(x)\frac{f'(x)}{f(x)} + m''(x)\right]^2 \\ &\quad + o\left(\frac{1}{nh}\right) + o\left(h^4\right). \end{aligned}$$

The minimizer of the leading term is

$$h_0 = \frac{\sigma^{2/5}\|K\|_2^{2/5}}{n^{1/5} f^{1/5}(x)} \mu_2^{-2/5}(K)\left[2m'(x)\frac{f'(x)}{f(x)} + m''(x)\right]^{-2/5},$$

the same order as $n^{-1/5}$. However, it still depends on unknown quantities.

Nadaraya-Watson Estimator: Cross Validation

Suppose that we split the data set into K nonoverlapping folds

$$\{1, 2, ..., n\} = V_1 \cup V_2 \cup \cdots \cup V_K.$$

For each h in a pre-specified grid of candidate bandwidths,

1. For each $k \in \{1, ..., K\}$, compute the cross validation error

   $$\mathrm{CV}_k(h) = \sum_{i \in V_k} \left[Y_i - \hat{m}_{h,-k}(X_i)\right]^2,$$

   where $\hat{m}_{h,-k}(x)$ is the estimator excluding the fold V_k.
2. Summarize the performance $\mathrm{CV}(h) = n^{-1} \sum_{k=1}^{K} \mathrm{CV}_k(h)$.
3. Choose h in the grid that minimizes $\mathrm{CV}(h)$
4. Refit to the entire data set using the chosen h.

Linear Smoother

An obvious drawback of leave-one-out cross validation is its computational burden. In some special cases, we can develop a short cut.

Definition

An estimator $\hat{m}(x)$ is a linear smoother if, for each x, there is a vector

$$\ell(x) = \begin{bmatrix} \ell_1(x) & \ell_2(x) & \cdots & \ell_n(x) \end{bmatrix}^T$$

such that $\hat{m}(x) = \ell^T(x) Y = \sum_{i=1}^n \ell_i(x) Y_i$, where Y is the vector of observed responses.

1. The Nadaraya-Watson estimator is a linear smoother.
2. The spline is a linear smoother.
3. Beyond our course, the Gaussian process regression and the RKHS estimator are also linear smoothers.

Fitted Value of Linear Smoother

The fitted value of a linear smoother is of the form

$$\begin{bmatrix} \hat{m}(X_1) \\ \vdots \\ \hat{m}(X_n) \end{bmatrix} = \begin{bmatrix} \ell^T(X_1) Y \\ \vdots \\ \ell^T(X_n) Y \end{bmatrix} = LY,$$

for some $n \times n$ smoothing matrix L. We call $\mathrm{tr}(L)$ the effective degrees of freedom, mimicking the number of parameters in a parametric model.

- The estimator is linear in Y but don't confuse it with linear regression.
- Linear regression is a special case of linear smoothing with $\ell(x) = x^T \left(X^T X\right)^{-1} X^T$ and $L = X \left(X^T X\right)^{-1} X^T$.

Smoothing Matrix

A linear smoother is a linear combination of the responses with coefficients $\{\ell_i(x)\}$. The coefficient vector $\ell(x)$ often satisfy

$$\sum_{i=1}^{n} \ell_i(x) = 1, \quad \text{for all } x.$$

Example

For the Nadaraya-Watson estimator,

$$\sum_{i=1}^{n} \ell_i(x) = \sum_{i=1}^{n} \frac{K_x\left(\frac{x-X_i}{h_x}\right)}{\sum_{j=1}^{n} K_x\left(\frac{x-X_j}{h_x}\right)} = 1.$$

Short Cut: Leave-One-Out CV

The leave-one-out CV error is

$$\mathrm{CV}(h) = \frac{1}{n}\sum_{i=1}^{n}\left[Y_i - \hat{m}_{h,-i}(X_i)\right]^2.$$

If $\hat{m}$ is a linear smoother, the error can be expressed as

$$\mathrm{CV}(h) = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{Y_i - \hat{m}_h(X_i)}{1 - L_{ii}}\right]^2,$$

where L_{ii} is the (i, i)th entry of L. Thus, the cross validation error can be obtained from the full data model.

Generalized Cross Validation

The generalized cross validation replaces L_{ii} by the average of all diagonal elements as

$$\mathrm{GCV}(h) = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{Y_i - \hat{m}_h(X_i)}{1 - n^{-1}\sum_{j=1}^{n} L_{jj}}\right]^2 = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{Y_i - \hat{m}_h(X_i)}{1 - \mathrm{tr}(L)/n}\right]^2,$$

where $\mathrm{tr}(L)$ is our effective degrees of freedom.

Multivariate Kernel Density

We have only introduced the kernel regression for $x \in \mathbb{R}$. It can be easily extended to $x \in \mathbb{R}^d$. Still consider the relation

$$\mathrm{E}(Y \mid X = x) = \int y \frac{f_{(X,Y)}(x, y)}{f_X(x)} dy = \frac{\int y f_{(X,Y)}(x, y)\, dy}{f_X(x)}.$$

- We can estimate $f_X(x)$ by

$$\hat{f}_X(x) = \frac{1}{n \det(H)} \sum_{i=1}^{n} K_x \left[H^{-1} (x - X_i) \right].$$

- We can estimate $f_{(X,Y)}(x, y)$ by

$$\hat{f}_{(X,Y)}(x, y) = \frac{1}{nh \det(H)} \sum_{i=1}^{n} K_y \left(\frac{y - Y_i}{h} \right) K_x \left[H^{-1} (x - X_i) \right].$$

Multivariate Kernel Regression

Hence, we estimate $m(x) = \mathrm{E}(Y \mid X = x)$ by

$$\hat{m}_H(x) = \int y \frac{\hat{f}_{(X,Y)}(x,y)}{\hat{f}_X(x)} dy = \frac{\sum_{i=1}^n K\left[H^{-1}(x - X_i)\right] Y_i}{\sum_{i=1}^n K\left[H^{-1}(x - X_i)\right]}.$$

The bias and variance are

$$\mathrm{E}[\hat{m}(x)] - m(x) = \frac{1}{2}\mu_2(K)\left\{2\frac{[m'(x)]^T H^2 f'(x)}{f(x)} + \mathrm{tr}\left[H m''(x) H\right]\right\},$$

$$\mathrm{Var}[\hat{m}(x)] = \frac{\sigma^2(x)\|K\|_2^2}{n\det(H) f(x)}.$$

Local Polynomial Approximation

Assume that for all u that is close to x,

$$m(u) \approx m(x) + \beta_1^T(x)(u - x).$$

The local linear estimator minimizes

$$L = \sum_{i=1}^{n} K\left[H^{-1}(x - X_i)\right]\left[Y_i - m(x) - \beta_1^T(x)(X_i - x)\right]^2.$$

Let

$$y = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}, Z = \begin{pmatrix} 1 & (X_1 - x)^T \\ \vdots & \vdots \\ 1 & (X_n - x)^T \end{pmatrix}, \gamma = \begin{pmatrix} m(x) \\ \beta_1(x) \end{pmatrix}$$

$$W = \operatorname{diag}\left\{K\left[H^{-1}(x - X_i)\right]\right\}.$$

Local Linear Estimator

The minimizer of L is given by

$$\hat{\gamma} = \begin{bmatrix} \hat{m}(x) \\ \hat{\beta}_1(x) \end{bmatrix} = \left(Z^T W Z\right)^{-1} Z^T W y.$$

The local linear estimator is $\hat{m}(x)$.

The bias and variance are

$$\begin{aligned} \mathrm{E}\left[\hat{m}(x)\right] - m(x) &\approx \frac{1}{2}\mu_2(K)\,\mathrm{tr}\left[H m''(x) H\right], \\ \mathrm{Var}\left[\hat{m}(x)\right] &\approx \frac{\sigma^2(x)\,\|K\|_2^2}{n\det(H)\, f(x)}. \end{aligned}$$