S&DS 365 / 665

Intermediate Machine Learning

Smoothing and Kernels

Monday, January 31

Yale

Reminders

- Assignment 1 next week
- OH posted to Canvas / EdD
- Any questions?

Topics for today

- Recap of lasso
- A simple algorithm for the lasso
- Nonparametric regression
- Smoothing methods
- Bias, variance, and curse of dimensionality

Recall from last time

- For low dimensional (linear) prediction, we can use least squares.
- For high dimensional linear regression, we face a bias-variance tradeoff: omitting too many variables causes bias while including too many variables causes high variance.
- The key is to select a good subset of variables.
- The *lasso* (ℓ_1-regularized least squares) is a fast way to select variables.
- If there are good, sparse linear predictors, lasso will work well.

Regression

Given the training data $\mathcal{D} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ we want to construct $\widehat{m}$ to make

$$\text{prediction risk} = R(\widehat{m}) = \mathbb{E}(Y - \widehat{m}(X))^2$$

small. Here, (X, Y) are a new pair.

Key fact: Bias-variance decomposition:

$$R(\widehat{m}) = \int \text{bias}^2(x)p(x)dx + \int \text{var}(x)p(x) + \sigma^2$$

where

$$\begin{aligned} \text{bias}(x) &= \mathbb{E}(\widehat{m}(x)) - m(x) \\ \text{var}(x) &= \text{Variance}(\widehat{m}(x)) \\ \sigma^2 &= \mathbb{E}(Y - m(X))^2 \end{aligned}$$

Bias-Variance Tradeoff

More generally, we need to tradeoff approximation error against estimation error:

$$R(\widehat{f}) - R^* = \underbrace{R(\widehat{f}) - \inf_{f \in \mathcal{F}} R(f)}_{\text{estimation error}} + \underbrace{\inf_{f \in \mathcal{F}} R(f) - R^*}_{\text{approximation error}}$$

where R^* is the smallest possible risk and $\inf_{f \in \mathcal{F}} R(f)$ is smallest possible risk using class of estimators $\mathcal{F}$.

- Approximation error is a generalization of squared bias
- Estimation error is a generalization of variance
- Decomposition holds more generally, even for classification

Sparse Linear Regression

Ridge regression does not take advantage of sparsity.

Maybe only a small number of covariates are important predictors. How do we find them?

We could fit many submodels (with a small number of covariates) and choose the best one. This is called *model selection*.

Now the inaccuracy is

prediction error = bias^2 + variance

The bias is the errors due to omitting important variables. The variance is the error due to having to estimate many parameters.

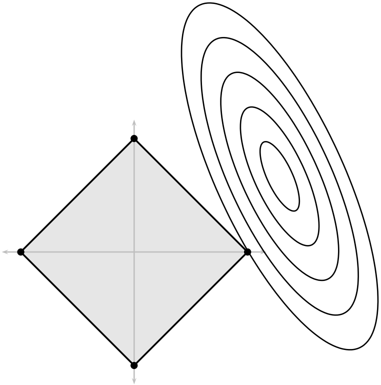
Sparsity Meets Convexity

Lasso regression

$$\widehat{\beta} = \arg\min_{\beta} \frac{1}{2n} \sum_{i=1}^{n} (Y_i - \beta^T X_i)^2 + \lambda \|\beta\|_1$$

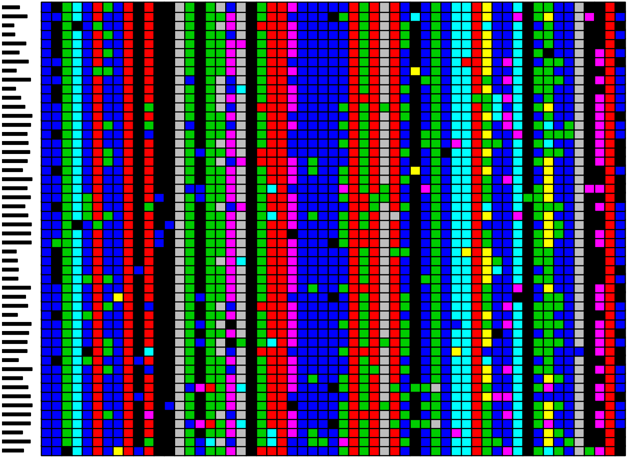
where $\|\beta\|_1 = \sum_j |\beta_j|$.

Sparsity: How corners create sparse estimators

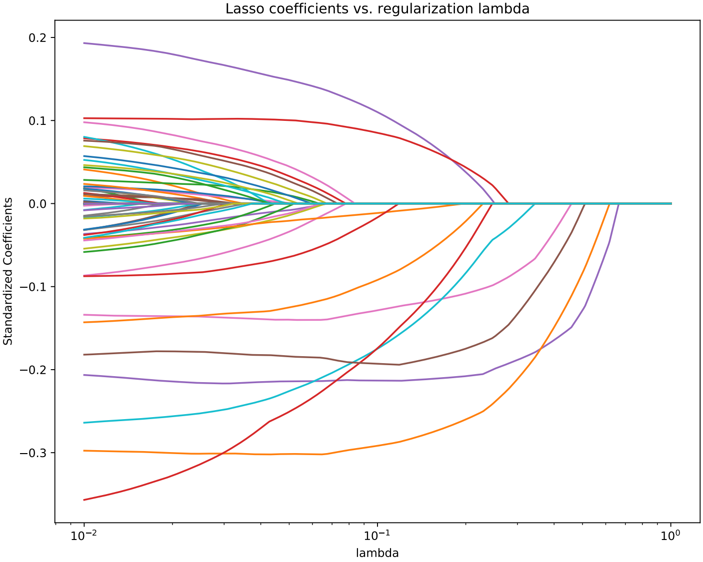


The lasso: HIV example

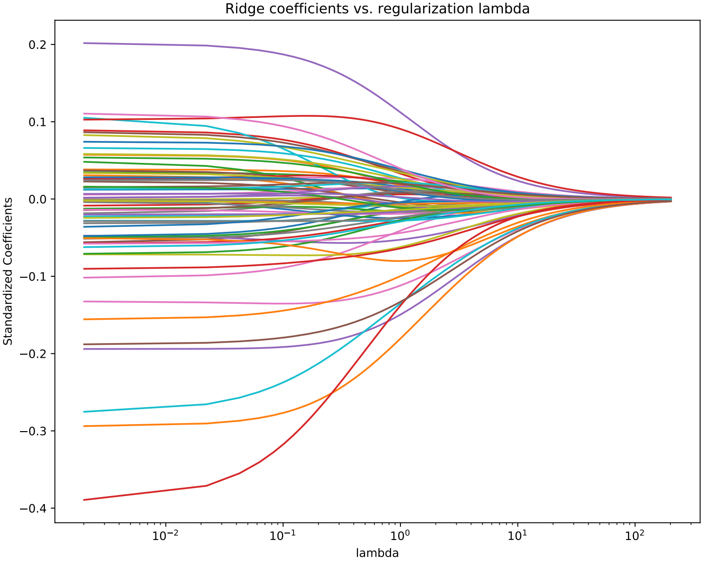
- Y is resistance to HIV drug.
- X_j = amino acid in position j of the virus.
- $p = 99$, $n \approx 100$.



The lasso: HIV example



Contrast with ridge regression



The lasso

- $\widehat{\beta}(\lambda)$ is called the lasso estimator. Selected set of variables is

$$\widehat{S}(\lambda) = \left\{ j : \ \widehat{\beta}_j(\lambda) \neq 0 \right\}.$$

- After you find $\widehat{S}(\lambda)$, you should re-fit the model by doing least squares on the sub-model $\widehat{S}(\lambda)$.

Selecting λ

To choose λ by risk estimation:

Re-fit the model with the non-zero coefficients. Then apply leave-one-out cross-validation:

$$\widehat{R}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{(Y_i - \widehat{Y}_i)^2}{(1 - H_{ii})^2} \approx \frac{1}{n}\frac{RSS}{\left(1 - \frac{s}{n}\right)^2}$$

where RSS is residual sum of squares and H is the hat matrix and $s = \#\{j:\ \widehat{\beta}_j \neq 0\}$.

Choose $\widehat{\lambda}$ to minimize $\widehat{R}(\lambda)$.

The lasso

The complete steps are:

1. Find $\widehat{\beta}(\lambda)$ and $\widehat{S}(\lambda)$ for each λ.
2. Choose $\widehat{\lambda}$ to minimize estimated risk.
3. Let $\widehat{S}$ be the selected variables.
4. Let $\widehat{\beta}$ be the least squares estimator using only $\widehat{S}$.
5. Prediction: $\widehat{Y} = X^T \widehat{\beta}$.

An algorithm for the lasso: Derived in steps

We'll derive a simple algorithm for computing the lasso solution in steps.

I'll do the first step in detail. The next steps only require simple calculations that I'll leave to you.

An algorithm for the lasso: Step 1

First consider minimizing

$$\frac{1}{2}(y - \beta)^2 + \lambda|\beta|$$

where y is a single number.

Taking the derivative and setting to zero, we get

$$\beta - y + \lambda v = 0$$

where

$$v \quad \begin{cases} = \text{sign}(\beta) & \text{if } |\beta| > 0 \\ \in [-1, 1] & \text{if } \beta = 0. \end{cases}$$

An algorithm for the lasso: Step 1

Solution can be written as

$$\widehat{\beta} = \begin{cases} y - \lambda & \text{if } \beta > 0 \\ y + \lambda & \text{if } \beta < 0 \\ y - \lambda\left(\frac{y}{\lambda}\right) & \text{if } \beta = 0. \end{cases}$$

Equivalently:

$$\widehat{\beta} = \begin{cases} y - \lambda & \text{if } y > \lambda \\ y + \lambda & \text{if } y < \lambda 0 \\ 0 & \text{if } |y| \leq \lambda. \end{cases}$$

An algorithm for the lasso: Step 1

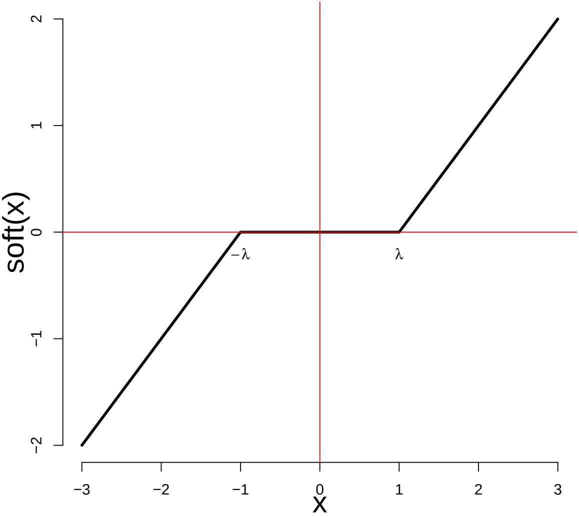
Equivalently:

$$\widehat{\beta} = \begin{cases} y - \lambda & \text{if } y > \lambda \\ y + \lambda & \text{if } y < \lambda 0 \\ 0 & \text{if } |y| \leq \lambda. \end{cases}$$

Soft thresholding

$$\begin{aligned} \widehat{\beta} &= \mathrm{Soft}_\lambda(y) \\ &\equiv \mathrm{sign(y)}\,(|y| - \lambda)_+ = \left(1 - \frac{\lambda}{|y|}\right)_+ y \end{aligned}$$

Soft thresholding



An algorithm for the lasso: Step 2

Next consider minimizing

$$\frac{1}{2}(y - x\beta)^2 + \lambda|\beta|$$

where y and x are a single numbers.

Show that

$$\widehat{\beta} = \text{Soft}_{\frac{\lambda}{x^2}}\left(\frac{xy}{x^2}\right)$$

An algorithm for the lasso: Step 3

Now consider minimizing

$$\frac{1}{2n}\sum_{i=1}^{n}(y_i - x_i\beta)^2 + \lambda|\beta|$$

for data $(x_1, y_1), \ldots, (x_n, y_n)$ with $p = 1$.

Show that

$$\widehat{\beta} = \text{Soft}_{\widetilde{\lambda}}\left(\widehat{\beta}_{OLS}\right)$$

where β_{OLS} is the least squares estimator and $\widetilde{\lambda} = \frac{\lambda}{\frac{1}{n}\sum_{i=1}^{n} x_i^2}$.

An algorithm for the lasso: Step 4

Finally, consider minimizing

$$\frac{1}{2n}\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda|\beta|$$

for data $(x_1, y_1), \ldots, (x_n, y_n)$ with $p \geq 1$. Apply previous algorithm to estimate β_1, holding other coefficients fixed.

Show that

$$\widehat{\beta}_1 = \text{Soft}_{\widetilde{\lambda}}\left(\widehat{\beta}_{OLS}\right)$$

where β_{OLS} is the least squares estimator for data (x_{i1}, r_i) with $r_i = y_i - \sum_{j\neq 1} x_{ij}\beta_j$. and $\widetilde{\lambda} = \frac{\lambda}{\frac{1}{n}\sum_{i=1}^{n} x_{i1}^2}$.

The lasso: Computing $\widehat{\beta}$

To minimize $\frac{1}{2n}\sum_i (y_i - \beta^T x_i)^2 + \lambda\|\beta\|_1$:

Lasso by coordinate descent

- Set $\widehat{\beta} = (0, \ldots, 0)$, then iterate until convergence:
- for $j = 1, \ldots, p$:
  - set $r_i = y_i - \sum_{s \neq j} \widehat{\beta}_s x_{si}$
  - Set $\widehat{\beta}_j$ to be least squares fit of r_i's on x_j.
  - $\widehat{\beta}_j \leftarrow \text{Soft}_{\lambda_j}(\widehat{\beta}_j)$ where $\lambda_j = \frac{\lambda}{\frac{1}{n}\sum_i x_{ij}^2}$.
- Then use least squares $\widehat{\beta}$ on selected subset $\widehat{S}(\lambda)$.

Next up

Nonparameteric regression by smoothing

Nonparametric Regression

Given $(X_1, Y_1), \ldots, (X_n, Y_n)$ predict Y from X.

Assume only that $Y_i = m(X_i) + \epsilon_i$ where where $m(x)$ is a smooth function of x.

The most popular methods are *kernel methods*. However, there are two types of kernels:

1. Smoothing kernels
2. Mercer kernels

Smoothing kernels involve local averaging.
Mercer kernels involve regularization.

Smoothing Kernels

- Smoothing kernel estimator:

$$\widehat{m}_h(x) = \frac{\sum_{i=1}^n Y_i \, K_h(X_i, x)}{\sum_{i=1}^n K_h(X_i, x)}$$

where $K_h(x, z)$ is a *kernel* such as

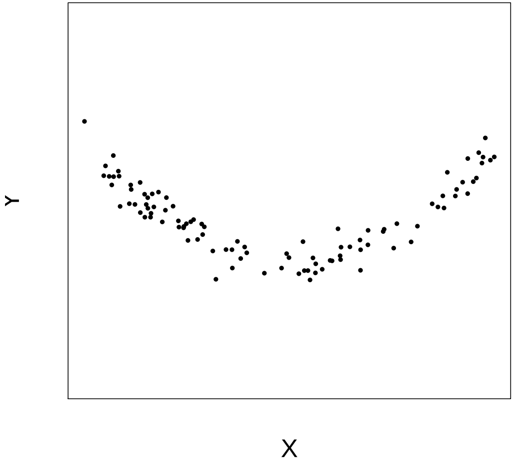
$$K_h(x, z) = \exp\left(-\frac{\|x - z\|^2}{2h^2}\right)$$

and $h > 0$ is called the *bandwidth*.

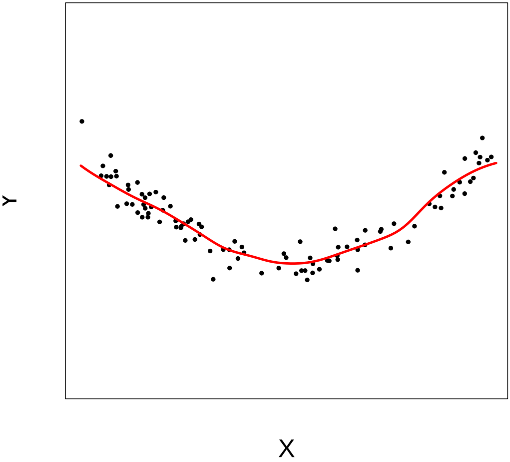
- $\widehat{m}_h(x)$ is just a local average of the Y_i's near x.

- The bandwidth h controls the bias-variance tradeoff:
Small h = large variance while *large h = large bias*.

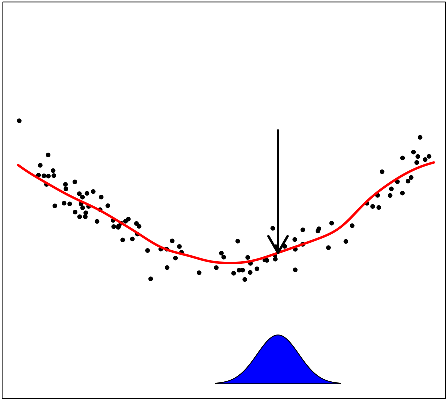
Example: Some Data – Plot of Y_i versus X_i



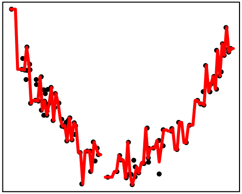
Example: $\widehat{m}(x)$



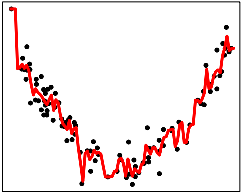
$\widehat{m}(x)$ is a local average



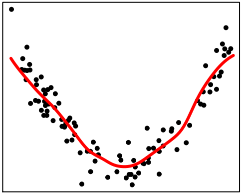
Effect of the bandwidth h



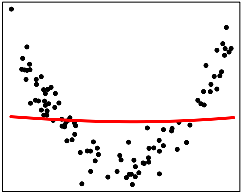
very small bandwidth



small bandwidth



medium bandwidth



large bandwidth

Smoothing Kernels

$$\text{Risk} = \mathbb{E}(Y - \widehat{m}_h(X))^2 = \text{bias}^2 + \text{variance} + \sigma^2.$$

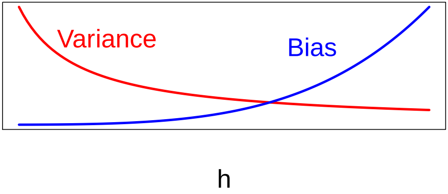
$$\text{bias}^2 \approx h^4,$$

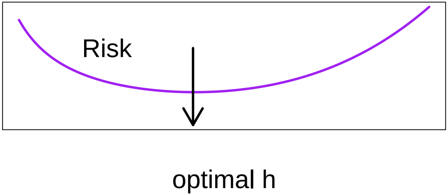
$\text{variance} \approx \frac{1}{nh^p}$ where p = dimension of X.

$\sigma^2 = \mathbb{E}(Y - m(X))^2$ is the unavoidable prediction error.

small h: low bias, high variance (undersmoothing)
large h: high bias, low variance (oversmoothing)

Risk Versus Bandwidth





Estimating the Risk: Cross-Validation

To choose h we need to estimate the risk $R(h)$. We can estimate the risk by using *cross-validation*.

1. Omit (X_i, Y_i) to get $\widehat{m}_{h,(i)}$, then predict: $\widehat{Y}_{(i)} = \widehat{m}_{h,(i)}(X_i)$.
2. Repeat this for all observations.
3. The cross-validation estimate of risk is:

$$\widehat{R}(h) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_{(i)})^2.$$

Shortcut formula:

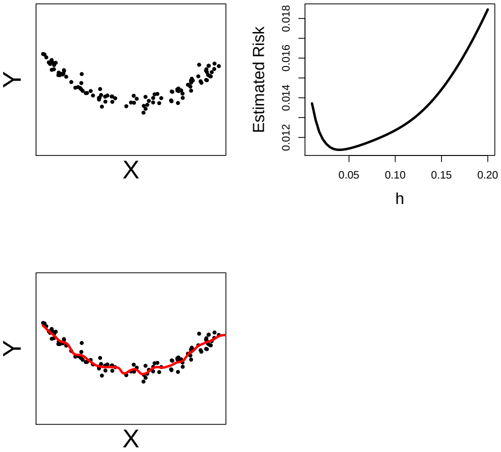
$$\widehat{R}(h) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \widehat{Y}_i}{1 - L_{ii}}\right)^2$$

where $L_{ii} = K_h(X_i, X_i)/\sum_t K_h(X_i, X_t)$.

Summary so far

1. Compute $\widehat{m}_h$ for each h.
2. Estimate the risk $\widehat{R}(h)$.
3. Choose bandwidth $\widehat{h}$ to minimize $\widehat{R}(h)$.
4. Let $\widehat{m}(x) = \widehat{m}_{\widehat{h}}(x)$.

Example



Multiple Regression

Kernel smoothing extends easily to the case where X has dimension $p > 1$. For example, just use

$$K(x, y) = e^{-\|x-y\|^2/2}.$$

However, this is hard to interpret and is subject to the curse of dimensionality. This means that the *statistical performance* and the *computational complexity* degrade as dimension p increases.

An alternative is to use something less nonparametric such as additive model where we restrict $m(x_1, \ldots, x_p)$ to be of the form:

$$m(x_1, \ldots, x_p) = \beta_0 + \sum_j m_j(x_j).$$

Additive Models

Model: $m(x) = \beta_0 + \sum_{j=1}^{p} m_j(x_j)$.

We can take $\widehat{\beta}_0 = \overline{Y}$ and we will ignore β_0 from now on.

We want to minimize

$$\sum_{i=1}^{n} \left(Y_i - (m_1(X_{i1}) + \cdots + m_p(X_{ip})) \right)^2$$

subject to m_j smooth.

Additive Models

The backfitting algorithm:

- Set $\widehat{m}_j = 0$
- Iterate until convergence:
  - Iterate over j:
    - $R_i = Y_i - \sum_{k \neq j} \widehat{m}_k(X_{ik})$
    - $\widehat{m}_j \longleftarrow \text{smooth}(X_j, R)$

Here, $\text{smooth}(X_j, R)$ is any one-dimensional nonparametric regression function.

R: glm

But what if p is large?

Sparse Additive Models

Ravikumar, Lafferty, Liu and Wasserman, JRSS B (2009)

Additive Model: $Y_i = \sum_{j=1}^{p} m_j(X_{ij}) + \varepsilon_i, \quad i = 1, \ldots, n$

High dimensional: $n \ll p$, with most $m_j = 0$.

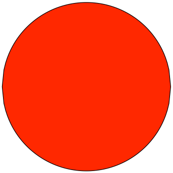
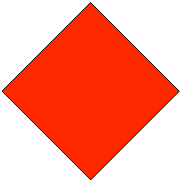
Optimization:

$$
\begin{aligned}
\text{minimize} \quad & \mathbb{E}\Big(Y - \textstyle\sum_j m_j(X_j)\Big)^2 \\
\text{subject to} \quad & \sum_{j=1}^{p} \sqrt{\mathbb{E}(m_j^2)} \leq L_n \\
& \mathbb{E}(m_j) = 0
\end{aligned}
$$

Related work by Bühlmann and van de Geer (2009), Koltchinskii and Yuan (2010), Raskutti, Wainwright and Yu (2011)

Sparse Additive Models

$$C = \left\{ m \in \mathbb{R}^4 \ : \ \sqrt{m_1(x_1)^2 + m_1(x_2)^2} + \sqrt{m_2(x_1)^2 + m_2(x_2)^2} \leq L \right\}$$

$\pi_{12} C =$  $\pi_{13} C =$ 

Stationary Conditions

Lagrangian

$$\mathcal{L}(f,\lambda) = \frac{1}{2}\mathbb{E}\left(Y - \sum_{j=1}^{p} m_j(X_j)\right)^2 + \lambda \sum_{j=1}^{p} \sqrt{\mathbb{E}(m_j^2(X_j))}$$

Let $R_j = Y - \sum_{k\neq j} m_k(X_k)$ be jth residual. Stationary condition

$$m_j - \mathbb{E}(R_j \mid X_j) + \lambda v_j = 0 \quad a.e.$$

where $v_j \in \partial\sqrt{\mathbb{E}(m_j^2)}$ satisfies

$$\begin{aligned} v_j &= \frac{m_j}{\sqrt{\mathbb{E}(m_j^2)}} \quad \text{if } \mathbb{E}(m_j^2) \neq 0 \\ \sqrt{\mathbb{E}v_j^2} &\leq 1 \quad \text{otherwise} \end{aligned}$$

Stationary Conditions

Rewriting,

$$\begin{aligned} m_j + \lambda v_j &= \mathbb{E}(R_j \mid X_j) \equiv P_j \\ \left(1 + \frac{\lambda}{\sqrt{\mathbb{E}(m_j^2)}}\right) m_j &= P_j \text{ if } \mathbb{E}(P_j^2) > \lambda \\ m_j &= 0 \text{ otherwise} \end{aligned}$$

This implies

$$m_j = \left[1 - \frac{\lambda}{\sqrt{\mathbb{E}(P_j^2)}}\right]_+ P_j$$

SpAM Backfitting Algorithm

Input: Data (X_i, Y_i), regularization parameter λ.

Iterate until convergence:

For each $j = 1, \ldots, p$:

Compute residual: $R_j = Y - \sum_{k \neq j} \widehat{m}_k(X_k)$

Estimate projection $P_j = \mathbb{E}(R_j \mid X_j)$, smooth: $\widehat{P}_j = \mathcal{S}_j R_j$

Estimate norm: $s_j = \sqrt{\mathbb{E}[P_j]^2}$

Soft-threshold: $\widehat{m}_j \leftarrow \left[1 - \frac{\lambda}{\widehat{s}_j}\right]_+ \widehat{P}_j$

Output: Estimator $\widehat{m}(X_i) = \sum_j \widehat{m}_j(X_{ij})$.

Example: Boston Housing Data

Predict house value Y from 10 covariates.

We added 20 irrelevant (random) covariates to test the method.

$Y =$ house value; $n = 506$, $p = 30$.

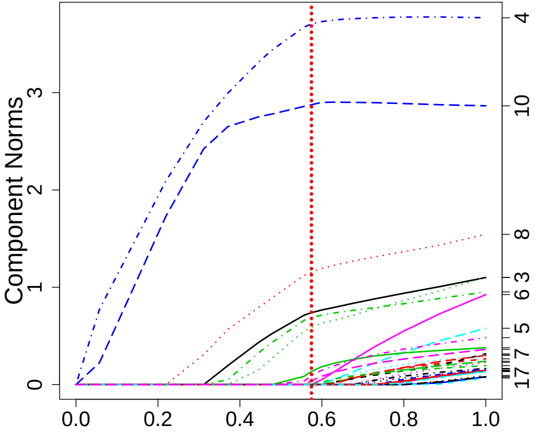
$$Y = \beta_0 + m_1(\mathrm{crime}) + m_2(\mathrm{tax}) + \cdots + \cdots m_{30}(X_{30}) + \epsilon.$$

Note that $m_{11} = \cdots = m_{30} = 0$.

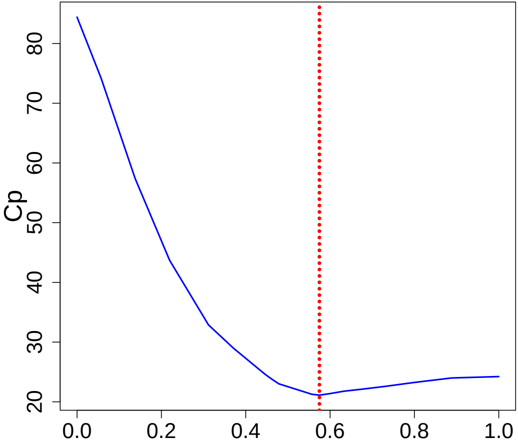
We choose λ by minimizing the estimated risk.

SpAM yields 6 nonzero functions. It correctly reports that $\widehat{m}_{11} = \cdots = \widehat{m}_{30} = 0$.

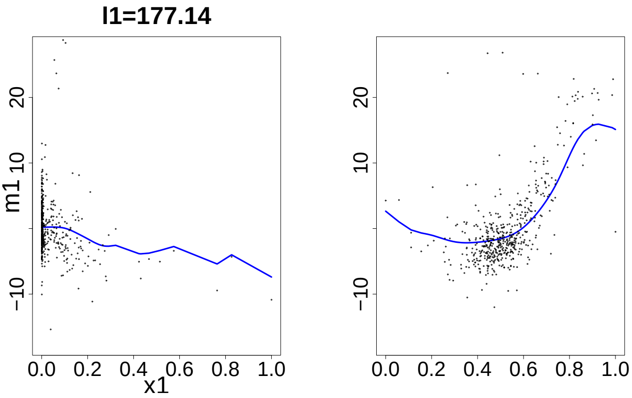
L_2 norms of fitted functions versus $1/\lambda$



Estimated Risk Versus λ



Example Fits



Example Fits