

S&DS 365 / 665  
Intermediate Machine Learning

# Sparse Regression

Friday, January 28

Yale

# Topics

- *Regression*
- High dimensional regression
- Sparsity and the lasso

# Regression

We observe pairs  $(X_1, Y_1), \dots, (X_n, Y_n)$ .

$\mathcal{D} = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$  is called the *training data*

$Y_i \in \mathbb{R}$  is the *response*;  $X_i \in \mathbb{R}^p$  is the *covariate* (or feature) vector

For example, suppose we have  $n$  subjects and  $Y_i$  is the blood pressure of subject  $i$  and  $X_i = (X_{i1}, \dots, X_{ip})$  is a vector of  $p = 5,000$  gene expression levels for subject  $i$

Remember:  $Y_i \in \mathbb{R}$  and  $X_i \in \mathbb{R}^p$

Given a new pair  $(X, Y)$ , we want to predict  $Y$  from  $X$ .

# Regression

Let  $\hat{Y}$  be a prediction of  $Y$ . The *prediction error* or *risk* is

$$R = \mathbb{E}(Y - \hat{Y})^2$$

where  $\mathbb{E}$  is the expected value (mean).

The best predictor is the *regression function*

$$m(x) = \mathbb{E}(Y|X = x) = \int y p(y|x) dy.$$

However, the true regression function  $m(x)$  is not known. We need to estimate  $m(x)$ .

# Regression

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

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

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

**Key fact:** Bias-variance decomposition:

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

where

$$\text{bias}(x) = \mathbb{E}(\hat{m}(x)) - m(x)$$

$$\text{var}(x) = \text{Variance}(\hat{m}(x))$$

$$\sigma^2 = \mathbb{E}(Y - m(X))^2$$

# Bias-Variance Tradeoff

Prediction Risk =  $\text{Bias}^2 + \text{Variance}$

Prediction methods with **low bias** tend to have **high variance**.

Prediction methods with **low variance** tend to have **high bias**.

For example, the predictor  $\hat{m}(x) \equiv 0$  has 0 variance but will be terribly biased.

To predict well, we need to balance the bias and the variance.

# Bias-Variance Tradeoff

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

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

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

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

# Linear Regression

Try to find the **best linear predictor**, that is, a predictor of the form:

$$m(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p.$$

**Important:** We do *not* assume the true regression function is linear.

Can always define  $x_1 = 1$ ; then the intercept is  $\beta_1$  and we can write

$$m(x) = \beta_1 x_1 + \cdots + \beta_p x_p = \beta^T x$$

where  $\beta = (\beta_1, \dots, \beta_p)$  and  $x = (x_1, \dots, x_p)$ .



# Low Dimensional Linear Regression

Assume for now that  $p$  (= length of each  $X_i$ ) is small. To find a good linear predictor we choose  $\beta$  to minimize the *training error*:

$$\text{training error} = \frac{1}{n} \sum_{i=1}^n (Y_i - \beta^T X_i)^2$$

The minimizer  $\hat{\beta} = (\hat{\beta}_1, \dots, \hat{\beta}_p)$  is called the *least squares estimator*.

# Low Dimensional Linear Regression

The least squares estimator is:

$$\hat{\beta} = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbb{Y}$$

where

$$\mathbb{X}_{n \times p} = \begin{pmatrix} X_{11} & X_{12} & \cdots & X_{1p} \\ X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{np} \end{pmatrix}$$

and

$$\mathbb{Y} = (Y_1, \dots, Y_n)^T.$$

Exercise: Show this

# Low Dimensional Linear Regression

Summary: the least squares estimator is  $m(x) = \hat{\beta}^T x = \sum_j \hat{\beta}_j x_j$  where

$$\hat{\beta} = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbb{Y}.$$

When we observe a new  $X$ , we predict  $Y$  to be

$$\hat{Y} = \hat{m}(X) = \hat{\beta}^T X.$$

We can try to improve this by:

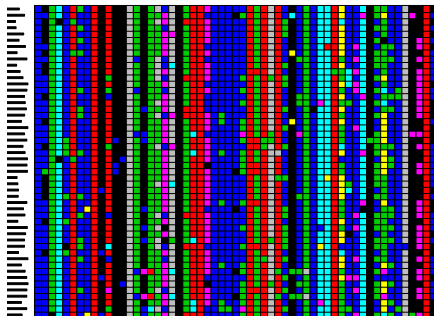
- (i) dealing with high dimensions
- (ii) using something more flexible than linear predictors.

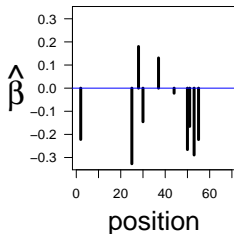
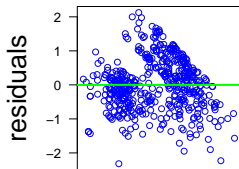
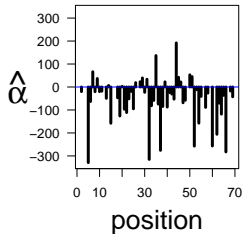
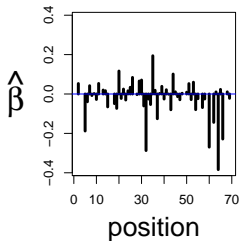
# Example

$Y$  = HIV resistance

$X_j$  = amino acid in position  $j$  of the virus.

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_{100} X_{100} + \epsilon$$





Top left:  $\hat{\beta}$  fitted values

Top right: marginal regression coefficients (one-at-a-time)

Bottom left:  $\hat{Y}_i - Y_i$  versus  $\hat{Y}_i$

Bottom right: a sparse regression (coming up soon)

# Topics

- Regression
- *High dimensional regression*
- Sparsity and the lasso

# High Dimensional Linear Regression

Now suppose  $p$  is large. We even might have  $p > n$  (more predictors than data points).

The least squares estimator is not defined since  $\mathbb{X}^T \mathbb{X}$  is not invertible. The variance of the least squares prediction is huge.

Recall the **bias-variance tradeoff**:

$$\text{Prediction Error} = \text{Bias}^2 + \text{Variance}$$

We need to increase the bias so that we can decrease the variance.

# Ridge Regression

Recall that the least squares estimator minimizes the training error  $\frac{1}{n} \sum_{i=1}^n (Y_i - \beta^T X_i)^2$ .

Instead, we can minimize the *penalized training error*:

$$\frac{1}{n} \sum_{i=1}^n (Y_i - \beta^T X_i)^2 + \lambda \|\beta\|_2^2$$

where  $\|\beta\|_2 = \sqrt{\sum_j \beta_j^2}$ .

The solution is (exercise!):

$$\hat{\beta} = (\mathbb{X}^T \mathbb{X} + \lambda I)^{-1} \mathbb{X}^T \mathbb{Y}$$



# Ridge Regression

The tuning parameter  $\lambda$  controls the bias-variance tradeoff:

$\lambda = 0 \implies$  least squares.

$\lambda = \infty \implies \hat{\beta} = 0.$

We choose  $\lambda$  to minimize  $\hat{R}(\lambda)$  where  $\hat{R}(\lambda)$  is an estimate of the prediction risk.

# Ridge Regression

To estimate the prediction risk, do *not* use training error:

$$R_{\text{training}} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2, \quad \hat{Y}_i = \mathbf{x}_i^T \hat{\beta}$$

because it is biased:  $\mathbb{E}(R_{\text{training}}) < R(\hat{\beta})$

Instead, we use *leave-one-out cross-validation*:

1. leave out  $(X_i, Y_i)$
2. find  $\hat{\beta}$
3. predict  $Y_i$ :  $\hat{Y}_{(-i)} = \hat{\beta}^T \mathbf{x}_i$
4. repeat for each  $i$

# Leave-one-out cross-validation

Can be shown that

$$\begin{aligned}\hat{R}(\lambda) &= \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_{(i)})^2 = \frac{1}{n} \sum_{i=1}^n \frac{(Y_i - \hat{Y}_i)^2}{(1 - H_{ii})^2} \\ &\approx \frac{R_{\text{training}}}{(1 - \frac{p}{n})^2} \\ &\approx R_{\text{training}} + \frac{2p\hat{\sigma}^2}{n}\end{aligned}$$

where

$$\begin{aligned}H &= \mathbb{X}(\mathbb{X}^T \mathbb{X} + \lambda I)^{-1} \mathbb{X}^T \\ p &= \text{trace}(H)\end{aligned}$$

## Example

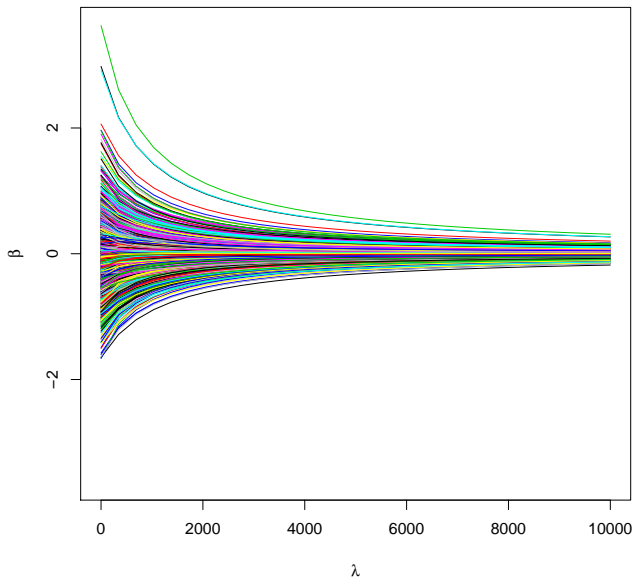
$$Y = 3X_1 + \cdots + 3X_5 + 0X_6 + \cdots + 0X_{1000} + \epsilon$$

$$n = 100, \quad p = 1,000.$$

So there are 1000 covariates but only 5 are relevant.

What does ridge regression do in this case?

# Ridge Regularization Paths



# 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?

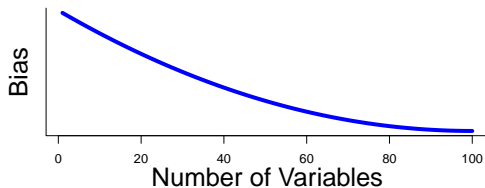
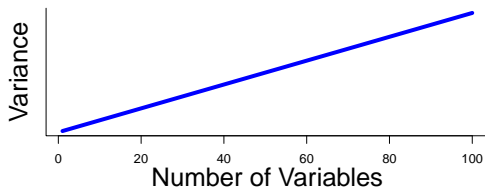
We could fit many submodels (with a small number of covariates) and choose the best one. This is called *model selection*.

Now the inaccuracy is

$$\text{prediction error} = \text{bias}^2 + \text{variance}$$

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

# The Bias-Variance Tradeoff



# The Bias-Variance Tradeoff

This is a Goldilocks problem: Can't use too few or too many variables.

Have to choose just the right variables.

Have to try all models with one variable, two variables,...

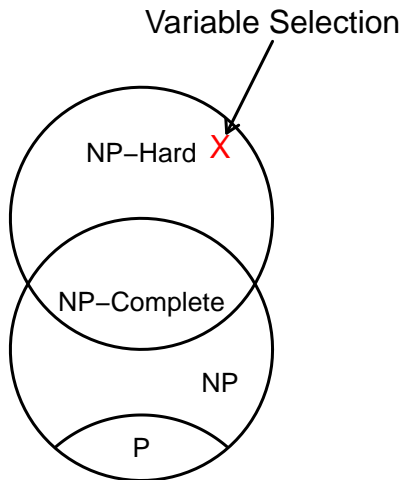
If there are  $p$  variables then there are  $2^p$  models.

Suppose we have 30,000 genes. We have to search through  $2^{30,000}$  models. But  $2^{30,000} >$  number of atoms in the universe.

This problem is NP-hard. This was a major bottleneck in statistics for many years.



# You are Here



# Two things that save us

Two key ideas to make this feasible are **sparsity** and **convex relaxation**.

**Sparsity**: probably only a few genes are needed to predict some disease  $Y$ . In other words, of  $\beta_1, \dots, \beta_{30,000}$  most  $\beta_j \approx 0$ .

But which ones? (Needle in a haystack.)

**Convex Relaxation**: Replace model search with something easier.

It is the marriage of these two concepts that makes it all work.

# Topics

- Regression
- High dimensional regression
- *Sparsity and the lasso*

# Sparsity

Consider:

$$\beta = (5, 5, 5, 0, 0, 0, \dots, 0).$$

This vector is high-dimensional but it is sparse.

Here is a less obvious example:

$$\beta = (50, 12, 6, 3, 2, 1.4, 1, 0.8, 0.6, 0.5, \dots)$$

It turns out that, if the  $\beta_j$ 's die off fairly quickly, then  $\beta$  behaves a like a sparse vector.

# Sparsity

We measure the (lack of) sparsity of  $\beta = (\beta_1, \dots, \beta_p)$  with the  $q$ -norm

$$\|\beta\|_q = \left( |\beta_1|^q + \dots + |\beta_p|^q \right)^{1/q} = \left( \sum_j |\beta_j|^q \right)^{1/q}.$$

Which values of  $q$  measure (lack of) sparsity?

sparse:  $a = \begin{matrix} 1 & 0 & 0 & \dots & 0 \end{matrix}$   
not sparse:  $b = \begin{matrix} 1/\sqrt{p} & 1/\sqrt{p} & 1/\sqrt{p} & \dots & 1/\sqrt{p} \end{matrix}$

	<div>✓ <math>q = 0</math></div>	<div>✓ <math>q = 1</math></div>	<div>✗ <math>q = 2</math></div>
$\ a\ _q$	1	1	1
$\ b\ _q$	p	$\sqrt{p}$	1

Lesson: Need to use  $q \leq 1$  to measure sparsity.

# Sparsity

So we estimate  $\beta = (\beta_1, \dots, \beta_p)$  by minimizing

$$\sum_{i=1}^n \left( Y_i - (\beta_0 + \beta_1 X_{i1} + \dots + \beta_p X_{ip}) \right)^2$$

subject to the constraint that  $\beta$  is sparse i.e.  $\|\beta\|_q \leq \text{small}$ .

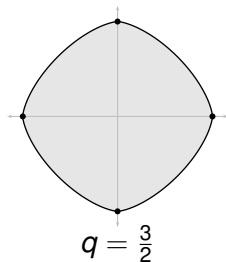
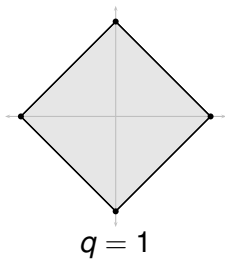
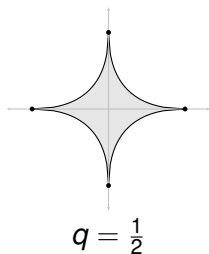
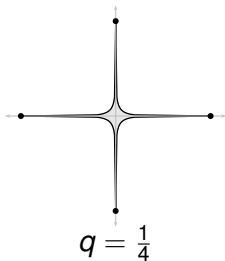
Can we do this minimization?

If we use  $q = 0$  this is same as searching through all  $2^p$  models.

What about other values of  $q$ ?

What does the set  $\{\beta : \|\beta\|_q \leq \text{small}\}$  look like?

**The set  $\|\beta\|_q \leq 1$  when  $p = 2$**



# Sparsity Meets Convexity

We need these sets to have a **nice** shape (convex). If so, the minimization is no longer NP-hard. In fact, it is easy.

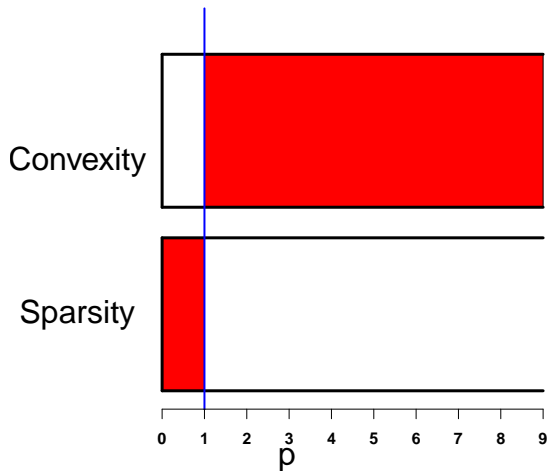
Sensitivity to sparsity:  $q \leq 1$  (actually,  $q < 2$  suffices)

Convexity (niceness):  $q \geq 1$

This means we should use  $q = 1$ .



# Where Sparsity and Convexity Meet



# Sparsity Meets Convexity

So we estimate  $\beta = (\beta_1, \dots, \beta_p)$  by minimizing

$$\sum_{i=1}^n \left( Y_i - (\beta_0 + \beta_1 X_{i1} + \dots + \beta_p X_{ip}) \right)^2$$

subject to the constraint that  $\beta$  is sparse i.e.  $\|\beta\|_1 = \sum_j |\beta_j| \leq \text{small}$ .

This is called the **lasso**. Invented by Rob Tibshirani in 1996. (Related work by Donoho and others around the same time).

# Lasso

The result is an estimated vector

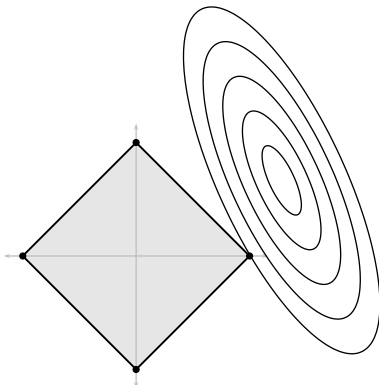
$$\hat{\beta}_1, \dots, \hat{\beta}_p$$

Most are 0!

Magically, we have done model selection without searching (thanks to sparsity plus convexity).

The next picture explains why some  $\hat{\beta}_j = 0$ .

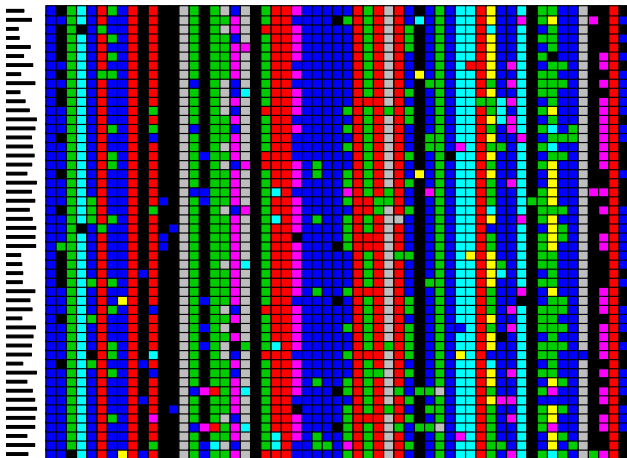
# Sparsity: How corners create sparse estimators



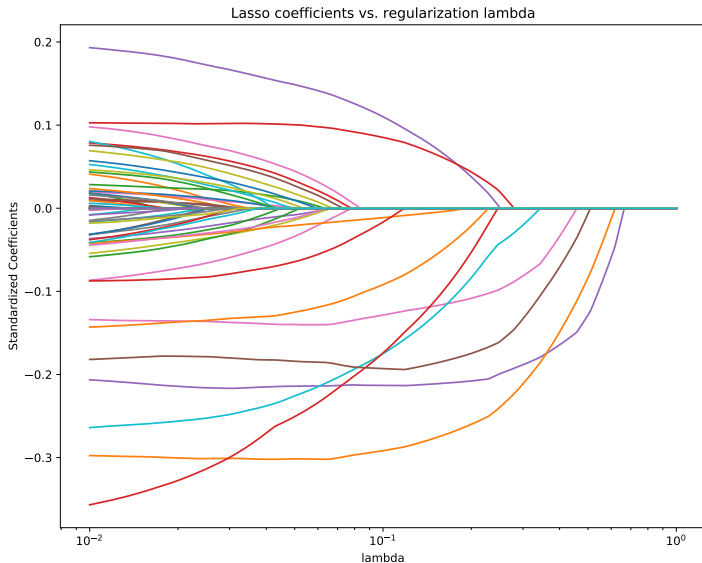
**Let's go to the notebook!**

# The lasso: HIV example again

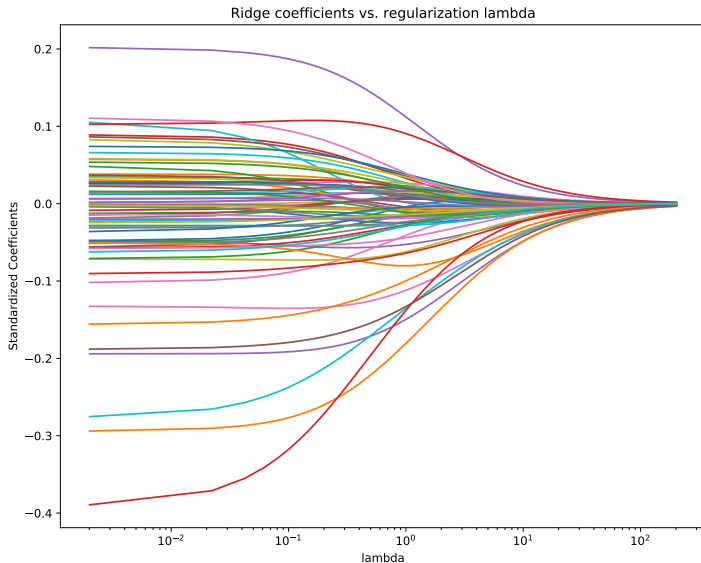
- $Y$  is resistance to HIV drug.
- $X_j$  = amino acid in position  $j$  of the virus.
- $p = 99$ ,  $n \approx 100$ .



# The lasso: HIV example



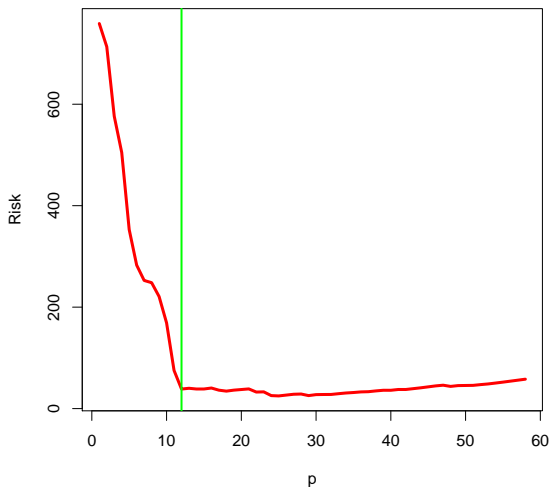
# Contrast with ridge regression



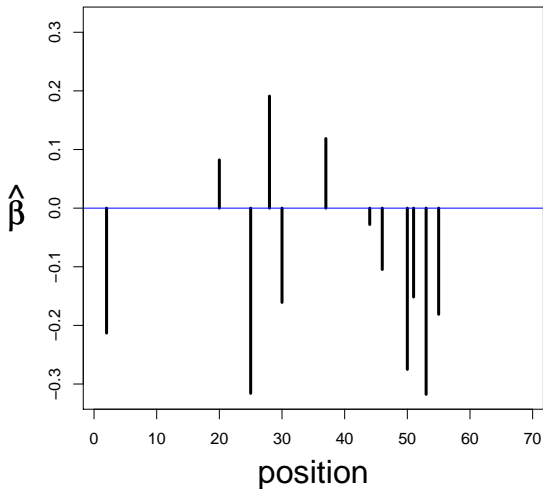


# Selecting $\lambda$

We choose the sparsity level by estimating prediction error.



# The lasso: An example



# Sparsity and convexity

To summarize: we penalize the sums of squares with

$$\|\beta\|_q = \left( \sum_j |\beta_j|^q \right)^{1/q}.$$

To get a sparse answer:  $q < 2$ .

To get a convex problem:  $q \geq 1$ .

So  $q = 1$  works.

The marriage of sparsity and convexity is one of the biggest developments in statistics and machine learning.

# The lasso

- $\hat{\beta}(\lambda)$  is called the **lasso** estimator. Then define

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

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

# The lasso

Choose  $\lambda$  by risk estimation.

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

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

where  $H$  is the hat matrix and  $s = \#\{j : \hat{\beta}_j \neq 0\}$ .

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

# The lasso

The complete steps are:

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



## Some convexity theory for the lasso

Consider a simpler model than regression: Suppose  $Y \sim N(\mu, 1)$ . Let  $\hat{\mu}$  minimize

$$A(\mu) = \frac{1}{2}(Y - \mu)^2 + \lambda|\mu|.$$

How do we minimize  $A(\mu)$ ?

- Since  $A$  is convex, we set the subderivative = 0. Recall that  $c$  is a *subderivative* of  $f(x)$  at  $x_0$  if

$$f(x) - f(x_0) \geq c(x - x_0).$$

- The *subdifferential*  $\partial f(x_0)$  is the set of subderivatives. Also,  $x_0$  minimizes  $f$  if and only if  $0 \in \partial f$ .

# $\ell_1$ and soft thresholding

- If  $f(\mu) = |\mu|$  then

$$\partial f = \begin{cases} \{-1\} & \mu < 0 \\ [-1, 1] & \mu = 0 \\ \{+1\} & \mu > 0. \end{cases}$$

- Hence,

$$\partial A = \begin{cases} \{\mu - Y - \lambda\} & \mu < 0 \\ \{\mu - Y + \lambda z : -1 \leq z \leq 1\} & \mu = 0 \\ \{\mu - Y + \lambda\} & \mu > 0. \end{cases}$$



# $\ell_1$ and soft thresholding

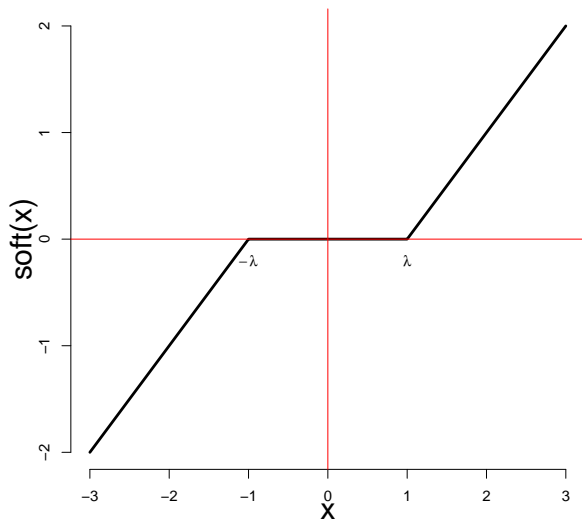
- $\hat{\mu}$  minimizes  $A(\mu)$  if and only if  $0 \in \partial A$ .
- So

$$\hat{\mu} = \begin{cases} Y + \lambda & Y < -\lambda \\ 0 & -\lambda \leq Y \leq \lambda \\ Y - \lambda & Y > \lambda. \end{cases}$$

- This can be written as

$$\hat{\mu} = \text{soft}(Y, \lambda) \equiv \text{sign}(Y) (|Y| - \lambda)_+.$$

# $\ell_1$ and soft thresholding



# The lasso: Computing $\hat{\beta}$

To minimize  $\sum_i (Y_i - \beta^T X_i)^2 + \lambda \|\beta\|_1$  by *coordinate descent*:

- Set  $\hat{\beta} = (0, \dots, 0)$  then iterate the following
- for  $j = 1, \dots, p$ :
  - ▶ set  $R_i = Y_i - \sum_{s \neq j} \hat{\beta}_s X_{si}$
  - ▶ Set  $\hat{\beta}_j$  to be least squares fit of  $R_i$ 's on  $X_j$ .
  - ▶  $\hat{\beta}_j \leftarrow \text{soft}(\hat{\beta}_j, \lambda / \sum_i X_{ij}^2)$
- Then use least squares  $\hat{\beta}$  on selected subset  $S$ .

# Variations on the lasso

- “Elastic net”: minimize

$$\sum_{i=1}^n (Y_i - \beta^T X_i)^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2$$

- Group lasso:

$$\beta = (\underbrace{\beta_1, \dots, \beta_k}_{v_1}, \dots, \underbrace{\beta_t, \dots, \beta_p}_{v_m})$$

minimize:

$$\sum_{i=1}^n (Y_i - \beta^T X_i)^2 + \lambda \sum_{j=1}^m \|v_j\|$$

# Summary

- 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* ( $\ell_1$ -regularized least squares) is a fast way to select variables.
- If there are good, sparse linear predictors, lasso will work well.