Shrinkage Methods

Lecture 17

Alexandra Chronopoulou

ILLINOIS

COLLEGE OF LIBERAL ARTS & SCIENCES

Department of Statistics
101 Illini Hall, MC-374
725 S. Wright St.
Champaign, IL 61820-5710

Shrinkage Methods

What to do if we have too many predictors?

- We have already discussed that too many predictors can create collinearity problems.
- Increasing the number of predictors will increase the prediction error.
- More predictors do not necessarily mean a better model, but more predictors would mean more in formation.

We will study three different methods we can use to shrink the number of predictors in order to find a trade-off between *model bias* and *prediction error*.

1. Principal Components Regression
2. Ridge Regression
3. Lasso Regression

Principal Components Regression

What to do when we have too many predictors?:

- Dimensionality reduction in the predictors space.
- Predictors might be highly correlated.
- Take matrix $\mathbf{X}$ of predictors and center the columns of $\mathbf{X}$ to have zero mean. Consider $\mathbf{X}$ with no intercept column.
- Find directions of greater variation in the data.

Principal Component Analysis

The steps to find directions of greater variation in matrix $\mathbf{X}$:

- Find $\mathbf{u}_1$ to maximize variance of $\mathbf{u}_1^\top \mathbf{X}$ subject to $\mathbf{u}_1^\top \mathbf{u}_1 = 1$.
- Find $\mathbf{u}_2$ to maximize variance of $\mathbf{u}_2^\top \mathbf{X}$ subject to $\mathbf{u}_1^\top \mathbf{u}_2 = 0$ and $\mathbf{u}_2^\top \mathbf{u}_2 = 1$.
- Continue looking for directions of greatest variation in the data which are orthogonal to the previous ones.
- Continue until the total number of dimensions is exhausted.

The principal components are given by the columns of matrix $\mathbf{Z}$, where

$$\mathbf{Z} = \mathbf{XU}$$

$\mathbf{z}_i$ and $\mathbf{u}_i$ are the columns of $\mathbf{Z}$ and $\mathbf{U}$ respectively. U is called the rotation matrix. $\mathbf{Z}$ is a version of the data rotated in such a way that the resulting principal components are orthogonal.

Principal Component Analysis

- Each Principal Component is a linear combination of the original variables $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ with weights given by each column $\mathbf{u}_i$ of matrix $\mathbf{U}$:

$$\mathbf{z}_i = u_{1i}\mathbf{x}_1 + u_{2i}\mathbf{x}_2 + \ldots + u_{mi}\mathbf{x}_m$$

- Principal components are very *sensitive* to outliers.
- The Mahalanobis distance can be used to measure the distance of a point to the data mean, after adjusting for correlation in the data.
- Under the multivariate normality assumption in m dimensions, the Mahalanobis distance $d_i = \sqrt{(\mathbf{x}_i - \mu)^\top \Sigma^{-1} (\mathbf{x}_i - \mu)}$ can be estimated using the sample estimators for μ and Σ and the quantity d_i^2 follows a χ^2_m distribution. This can be used to detect outliers in higher dimensions.

Principal Components Regression (PCR)

- Replace model $Y \sim X$ by the model $Y \sim Z$
- Only need to use the first few columns of Z as predictors
- Interpretation of the PCAs as predictors might be challenging. We need to use the values of $\mathbf{u}_i$ in the rotation matrix (also called the loadings) for interpretation.
- Sometimes we can make better predictions with a small number of PCs in $\mathbf{Z}$ than with a large number of predictors in $\mathbf{X}$

How many Principal Components?

- The trace of the sample variance-covariance S of $\mathbf{X}$ (total sample variance) is equal to the sum if its eigenvalues:

$$trace(S) = s_1^2 + s_2^2 + \ldots + s_m^2 = \lambda_1 + \lambda_2 + \ldots + \lambda_m$$

- Most of the total variance of a data set is concentrated in the first principal components.
- Make a plot of the PCs standard deviations ($\sqrt{\lambda_i}$) vs. the PC index i. This is called the scree plot.
- Look for the PC index i where there is a big change in slope (the elbow) in the scree plot.
- Another way is to calculate the cumulative variance explained by the first PCs, and retain the number of PCs explaining between 70% to 90% of the total variation.
- An alternative way is to discard PCs such that $\lambda_i < \bar{\lambda}$.

PCR Example: `meatspec` data set

- `meatspec` data set from the *faraway* library.
- The goal is to predict fat content (response) using 100-channel spectrum of absorbances (predictors) in 215 samples of finely chopped meat.
- Partition the data into *training sample* and *testing sample* to test model performance.
- Fit model with all predictors using the training data set and make a prediction on the testing set.
- Select few PCs representing the 100 predictors and repeat the process (*shrinkage effect)*
- What about if we select the number of PCs to minimize the prediction error instead?. We can use *cross-validation*.

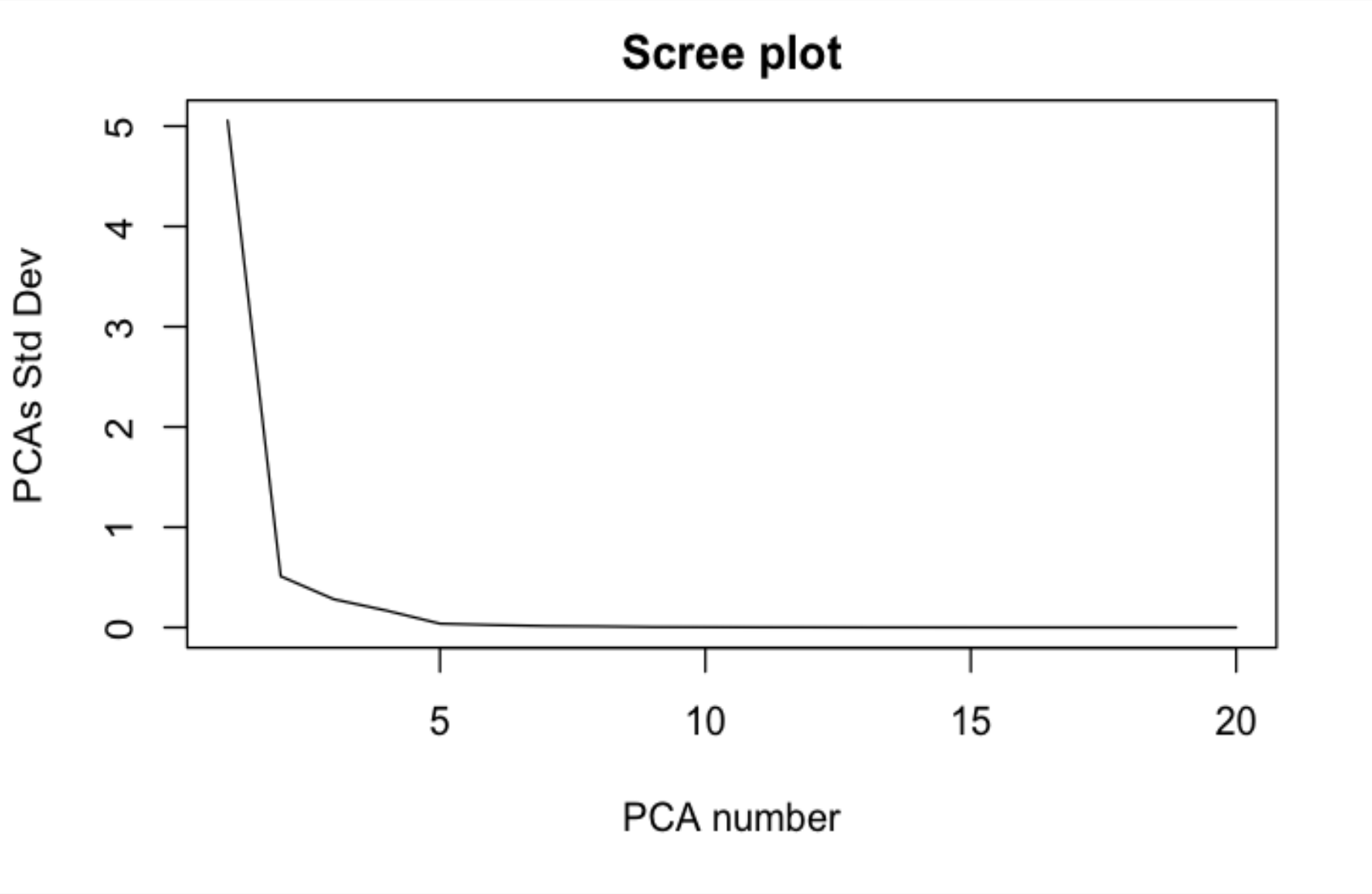
PCR Example: `meatspec` data set

Function **prcomp** can be used to calculate the PCs and extract the λ's squared-roots (sdev) and *eigenvectors* (rotation) of the variance-covariance matrix:

```
data(meatspec,package="faraway")
trainmeat<-meatspec[1:172,]
testmeat<-meatspec[173:215,]
mod1<-lm(fat~.,trainmeat)
meatpca<-prcomp(trainmeat[,-101])
round(meatpca$sdev,3)[1:50]
```

```
##  [1] 5.055 0.511 0.282 0.168 0.038 0.025 0.014 0.011 0.005 0.003 0.002 0.002
## [13] 0.001 0.001 0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [25] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [37] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [49] 0.000 0.000
```

The first two PC's explains around 90% of the total variation of meat fat content.



According to the scree plot, 4 PC's (elbow at 5th PC) seem adequate to represent the data.

The **pcr** function (principal component regression) from the *pls package* has useful features for prediction and cross-validation. We can easily calculate the RMSE for the training set and the testing set.

```
modpcr<-pcr(fat ~ ., data=trainmeat,ncomp=50)
#summary(modpcr)
#RMSE with 4 PCAs
rmse(predict(modpcr,ncomp=4),trainmeat$fat)
```

```
## [1] 4.064745
```

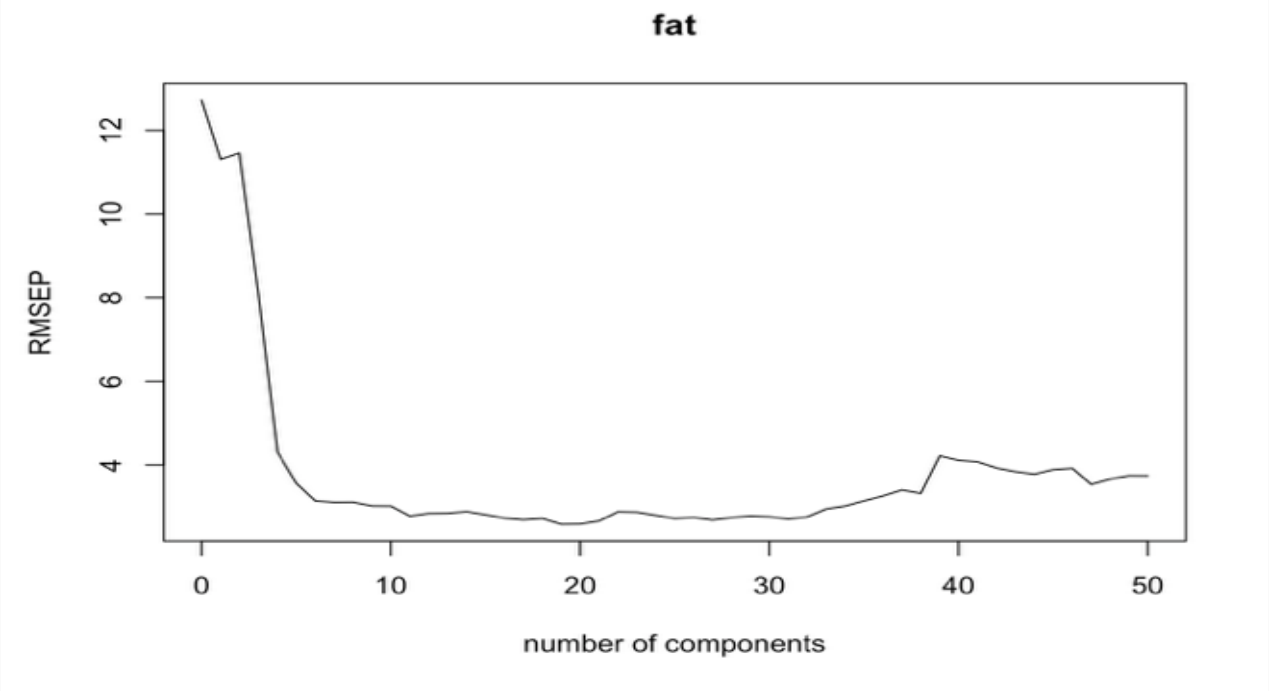
```
rmse(predict(modpcr,testmeat,ncomp=4),testmeat$fat)
```

```
## [1] 4.533982
```

As expected the testing error is larger than the training error. We can do better by selecting the number of PC's that minimize the CV error.

You can use the function **RMSEP** instead, to select the number of PC's that minimize the 10-fold Cross-Validation error. The resulting Cross-Validation error is < 2.5

```
#Use 10-fold Cross-Validation
set.seed(123)
modpcrcv<-pcr(fat~.,data=trainmeat,validation="CV",ncomp=50)
pcrCV<-RMSEP(modpcrcv,estimate="CV")
plot(pcrCV)
```



Ridge Regression

Ridge Regression

- Although the aim of PCR is to reduce dimensionality in the number of predictors, you still have to measure all the predictors since each PC is a linear combination of all predictors.
- Ridge regression assumes that after normalization, some of the regression coefficients should not be very large.
- Ridge regression is very useful when you have collinearity and the LS regression coefficients are unstable.
- The method uses a penalized regression since the LS minimization problem has a penalty term:

$$\text{minimize}(y - X\beta)^\top (y - X\beta) + \lambda \sum_j \beta_j^2$$

for some $\lambda \geq 0$. The penalty term is $\sum_j \beta_j^2$

Ridge Regression

- Usually predictors are standardized first (centered by their means and scaled by their standard deviations) and the response y is centered.
- The ridge regression estimates are:

$$\hat{\boldsymbol{\beta}} = (X^\top X + \lambda I)^{-1} X^\top y$$

(Note the extra term λI or *ridge* in the $X^\top X$ matrix)

- The difference with standard LS is that the problem solution $\boldsymbol{\beta}$ minimizes:

$$(y - X\boldsymbol{\beta})^\top (y - X\boldsymbol{\beta}) \quad \text{subject to} \quad \sum_{j=1}^{p} \boldsymbol{\beta}_j^2 \leq t^2$$

- The parameter λ (or t) should be chosen to have stable estimates of β. Can use function *lm.ridge* from the *MASS* library.

Ridge Regression

- Note that when $\lambda = 0$ the ridge regression estimation problem reduces to the standard LS problem, while when $\lambda \to \infty$, $\hat{\boldsymbol{\beta}} \to \mathbf{0}$.
- It is useful to plot the values of $\hat{\beta}_j$ as a function of λ.
- The value of λ can be also chosen using automated methods as Generalized Cross-Validation (GCV) (similar to Cross-Validation).
- Ridge regression coefficient estimates are biased.

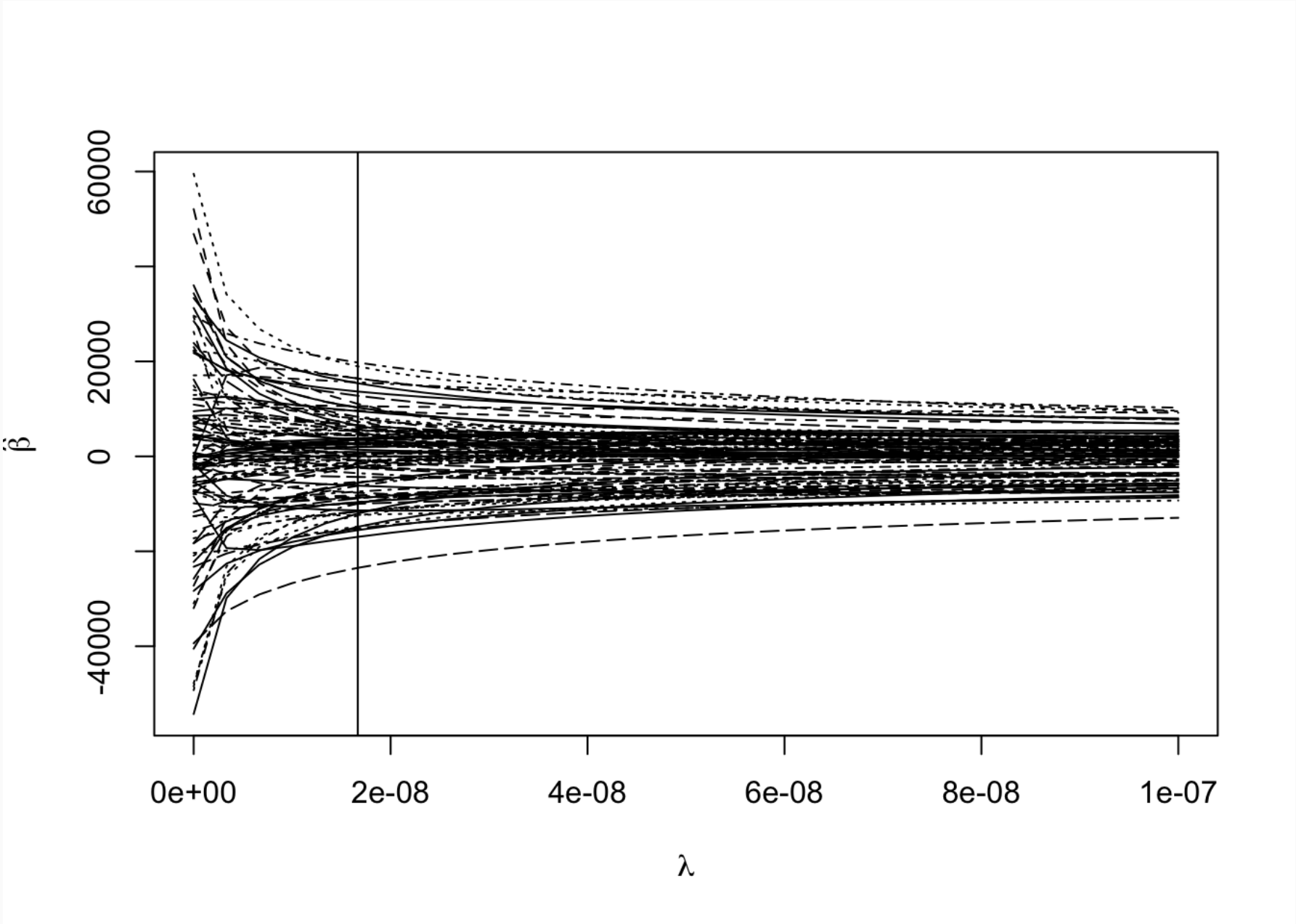
Example: meatspec data set

The value of λ is selected by minimizing the GCV. Some experimentation is required to get the initial λ range.

```
library(MASS)
library(faraway)
data(fat)
trainmeat<-meatspec[1:172,]
testmeat<-meatspec[173:215,]
ridgemod<-lm.ridge(fat~.,trainmeat,lambda=seq(0,0.0000001,len=31))
# Predictors are centered and scaled.
# The response is centered
matplot(ridgemod$lambda,coef(ridgemod),type="l",xlab=expression(lambda),ylab=exp
ression(hat(beta)),col=1)
# Look for the value of lambda that minimizes GCV
which.min(ridgemod$GCV)
```

```
## 1.666667e-08
##            6
```

```
abline(v=1.666667e-08)
```



Trace plot for the *meatspec* data. The vertical line is the value of λ that minimizes the GCV.

Lasso Regression

Lasso Regression

- In this case the estimated $\hat{\boldsymbol{\beta}}$ minimizes:

$$\text{minimize}(y - X\boldsymbol{\beta})^\top (y - X\boldsymbol{\beta}) + \lambda \sum_j |\beta_j|$$

  for some $\lambda \geq 0$. The penalty term is $\sum_j |\beta_j|$ (L_1 constraint).

- In two-dimensions the constraint defines a square. In higher dimensions it defines a polytope.

- Lasso is useful when the response can be explained by few predictors with zero effect on the remaining predictors (Lasso is similar to a variable selection method).

- When $\beta_j = 0$ the corresponding predictor is eliminated. This is not the case for ridge regression.

Lasso regression

- Use Lasso when the effect of predictors is *sparse*. This means that only few predictors will have an effect on the response (e.g. gene expression data) or when number of predictors is large ($p > n$)
- Use the *lars R package* for Lasso
- Select t in the constraint $\sum_{j=1}^{p} |\beta|_j \leq t$ by using Cross-Validation (CV).
- As t increases, the number of predictors increases.

Example: state data set

We have data from 50 states:

- Population: population estimate as of July 1, 1975
- Income: per capita income (1974)
- Illiteracy: illiteracy (1970, percent of population)
- Life Exp: life expectancy in years (1969–71)
- Murder: murder and non-negligent manslaughter rate per 100,000 population (1976)
- HS Grad: percent high-school graduates (1970)
- Frost: mean number of days with minimum temperature below freezing (1931–1960) in capital or large city
- Area: land area in square miles

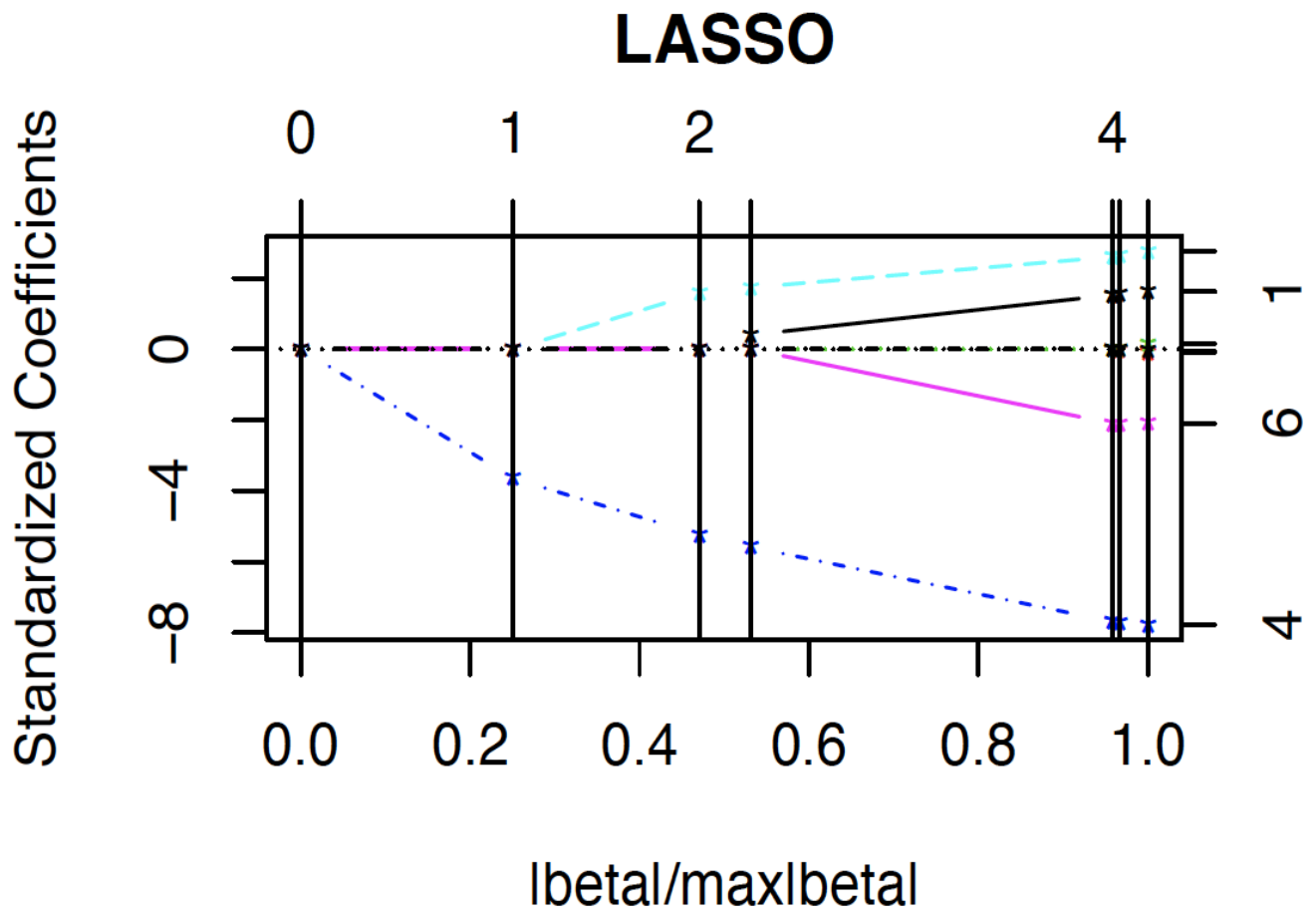
We want to predict *Life Expectancy*

Example: state data set

Use function *lars* from library *lars*

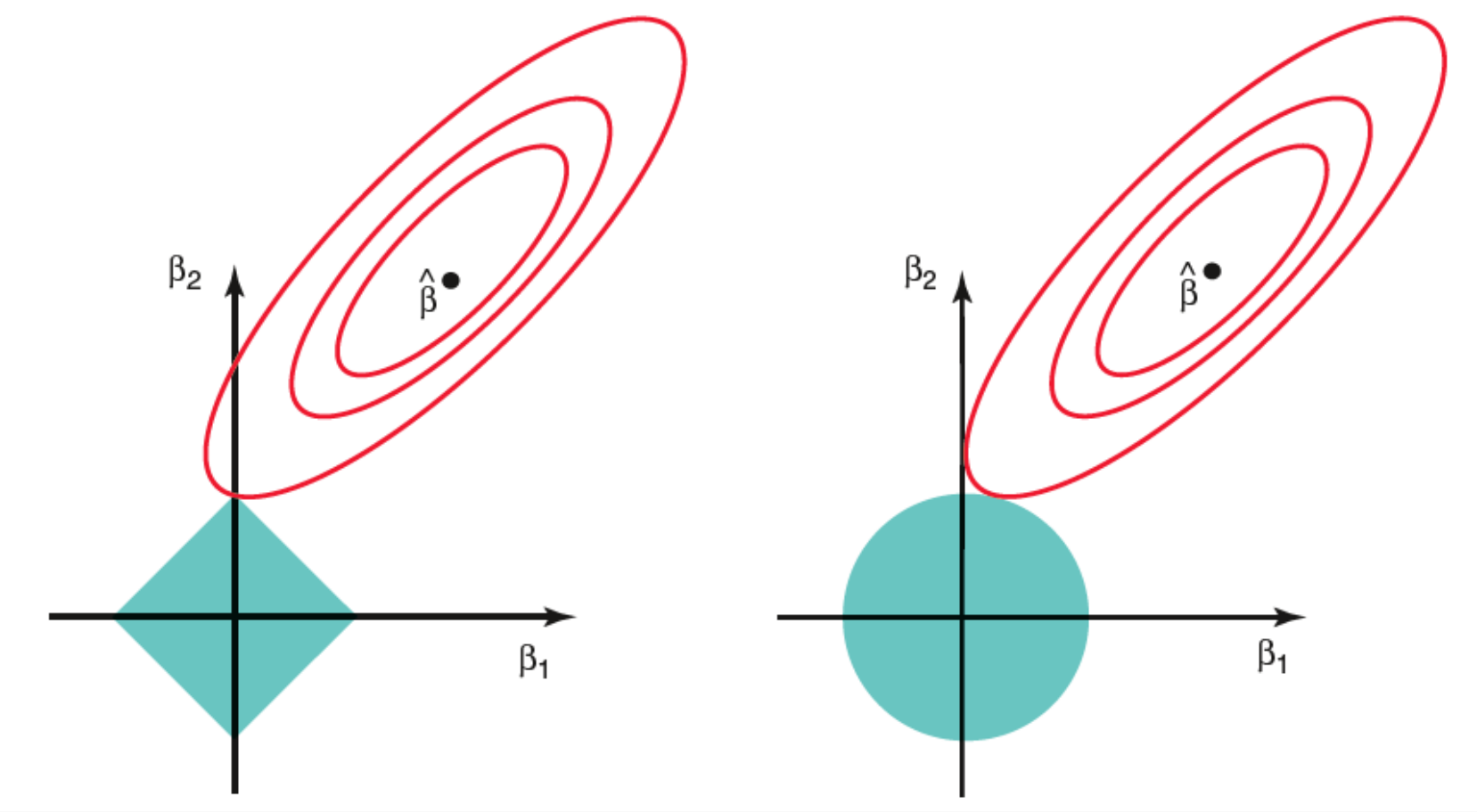
```
library(lars)

## Loaded lars 1.2

data(state)
statedata<-data.frame(state.x77,row.names = state.abb)
names(statedata)[1:4]

## [1] "Population" "Income"     "Illiteracy" "Life.Exp"

names(statedata)[5:8]

## [1] "Murder"  "HS.Grad" "Frost"   "Area"

#No model equation in function lars
modlasso<-lars(as.matrix(statedata[,-4]),
                         statedata$Life.Exp)
```

```
plot(modlasso)
```



- The x-axis in the graph is the scaled value of $t = \sum_{j=1}^{p} |\beta_j|$
- This value has been scaled to its maximum value which is the least-square solution.
- As t increases more predictors enter into the model.
- Variable 4 (murder rate) enters first, followed by HS graduation, population and days of frost (*Frost*). The remaining variables enter when t is large (close to the LS solutions).
- t can be selected by cross-validation (lars uses 10-fold CV by default) using function *cvlars*

Comparing Ridge Regression and Lasso



Comparing Ridge Regression and Lasso

- Lasso selects a sub-set of predictors (some coefficients equal to zero).
- Ridge regression performs better when the response is a function of many predictors with coefficients around the same size.
- Lasso will perform better when a relatively small number of predictors have large coefficients and the rest are very small or equal to zero.
- Since the number of predictors is never known *a priori*, cross-validation can be used to decide which approach is better for a particular data set.