Reasons for feature/variable selection

- Make the model easier to interpret.
- Removing variables that are redundant and do not add any information.
- Reduce the size of the problem to enable algorithms to work faster, making it possible to handle with high-dimensional data.
- Reduce overfitting.

Improving OLS

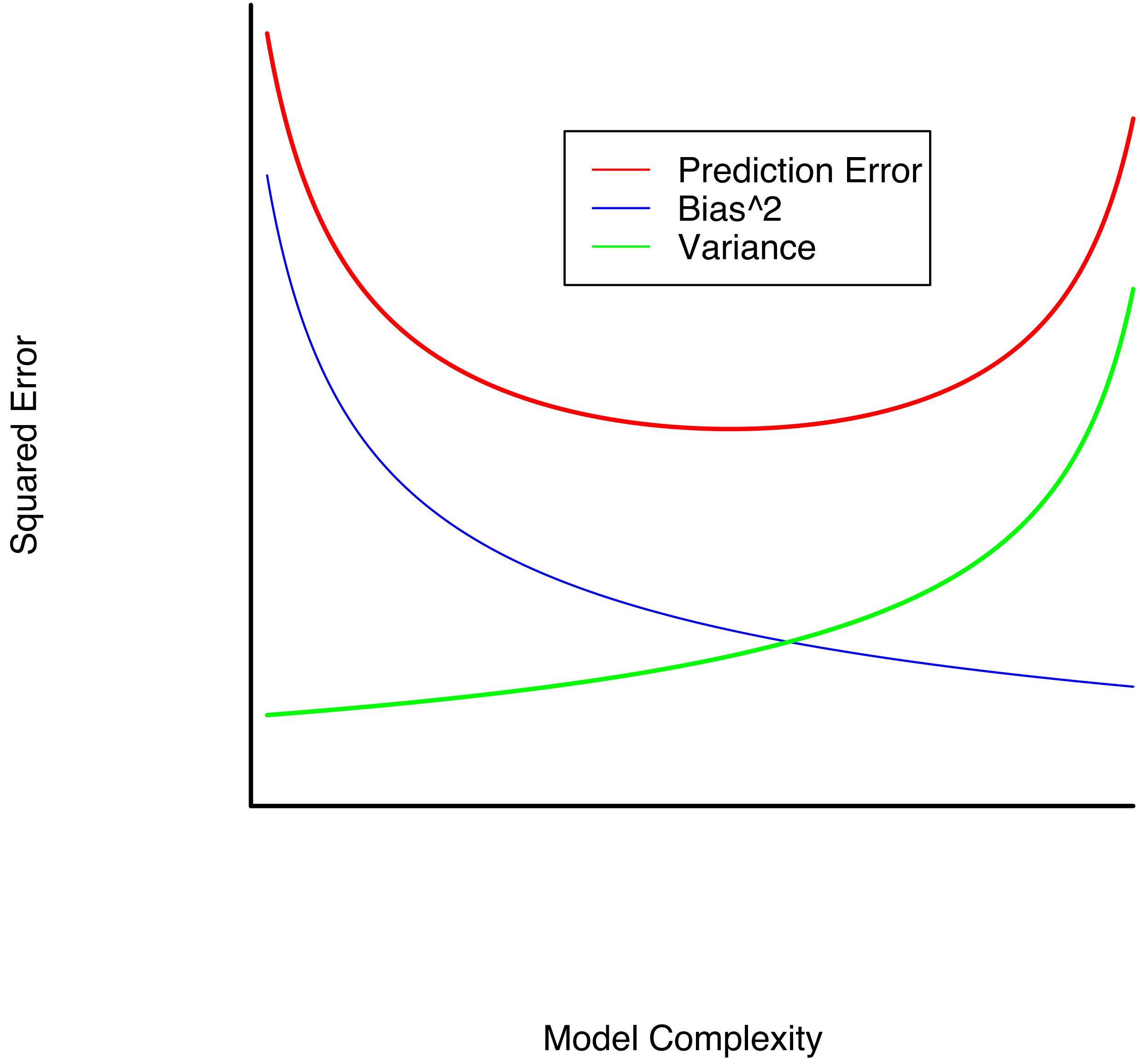
- We may want to improve the simple linear model by replacing OLS estimation with some alternative fitting procedure.
- Recall OLS estimators are those having the least variance for unbiased estimators.
- In that case, why use an alternative fitting procedure?
  - Prediction Accuracy
  - Model Interpretability

Bias/Variance Trade-Off

- There is always a trade-off between variance and bias.

- The more variance we introduce, the less bias. Similarly, the more bias, the less variance.

Depicting the bias-variance tradeoff



Classical Linear Regression Problem

- Given predictors X_{nxp} and response $y_{n\text{x}1}$.
- Linear model $y=X\beta+\varepsilon$, with $N(0,\sigma^2)$.
- Estimate β with $(X'X)^{-1}X'y$.
- Widely used in a huge amount of empirical statistical research.

Developing Trend

- Classical model requires $p < n$, but recent developments have pushed people beyond the classical model, to $p >> n$.

Prediction Accuracy

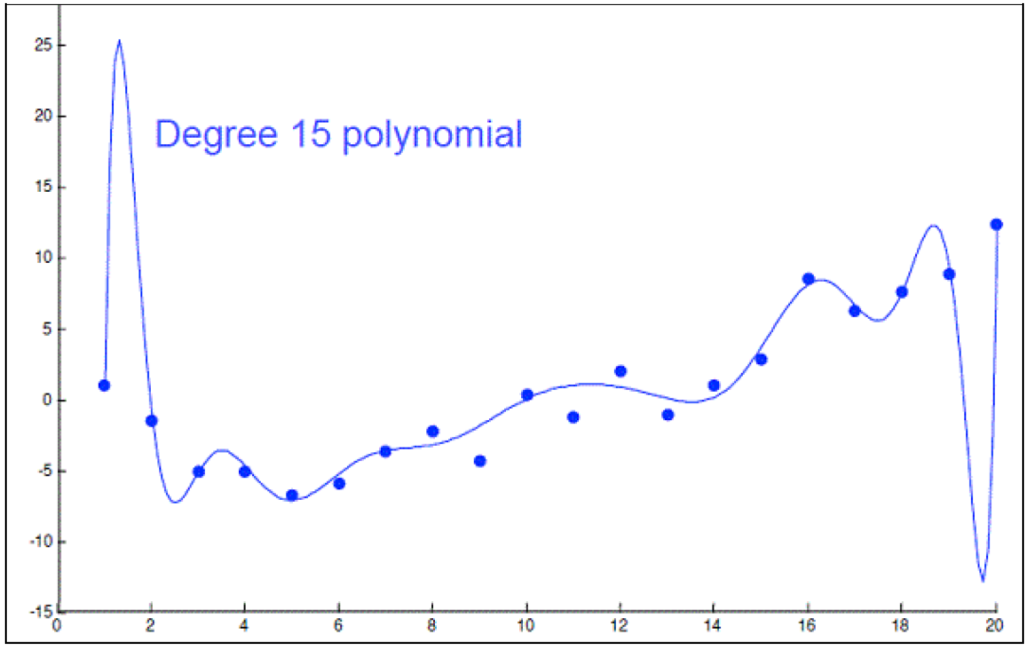
- The OLS estimates have relatively low bias and low variability especially when the relationship between the response and predictors is linear and *n* >> *p*.

- If *n* is not much larger than p, then the OLS fit can have high variance and may result in over fitting and poor estimates on unseen observations.

- If *p* > *n*, then the variability of the OLS fit increases dramatically, and the variance of these estimates in infinite.
  - Usually many have little or no effect on the response.

Feature/Variable Selection

- Carefully selected features can improve model accuracy, but adding too many can lead to overfitting.
  - **Overfitted models** describe **random error** or **noise** instead of any underlying relationship.
  - They generally have poor predictive performance on test data.



- For instance, we can use a 15-degree polynomial function to fit the following data so that the fitted curve goes nicely through the data points.
- However, a brand new dataset collected from the same population may not fit this particular curve well at all.

Feature/Variable Selection (cont.)

- **Subset Selection**
  - Identify a subset of the p predictors that we believe to be related to the response; then, fit a model using OLS on the reduced set.
  - Methods: best subset selection, stepwise selection
- **Shrinkage (Regularization)**
  - Involves shrinking the estimated coefficients toward zero relative to the OLS estimates; has the effect of reducing variance and performs variable selection.
  - Methods: ridge regression, lasso
- **Dimension Reduction**
  - Involves projecting the p predictors into a M-dimensional subspace, where M < p, and fit the linear regression model using the M projections as predictors.
  - Methods: principal components regression, partial least squares

Best subset selection algorithm

- We fit a separate OLS regression for each possible combination of the *p* predictors:

  1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.
  2. For $k = 1, 2, . . . , p$:
     1. Fit all models that contain exactly *k* predictors.
     2. Pick the best among these $\binom{n}{k}$ models, and call it M_k. Here best is defined as having the smallest RSS, or equivalently largest R^2.
  3. Select a single best model from among $M_0, . . . , M_p$ using cross-validated prediction error, C_p, AIC, or BIC, or adjusted R^2.

Best Subset Selection (cont.)

- While best subset selection is a simple and conceptually appealing approach, it suffers from computational limitations.

- The number of possible models that must be considered grows rapidly as *p* increases.

- Best subset selection becomes computationally *infeasible* for value of *p* greater than around 40.

Stepwise Selection

- For computational reasons, best subset selection cannot be applied with very large *p*.

- The larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data.

- An enormous search space can lead to overfitting and high variance of the coefficient estimates.

Stepwise Selection (cont.)

More attractive methods include:

- Forward Stepwise Selection
  - Begins with a null OLS model containing no predictors, and then adds one predictor at a time that improves the model the most until no further improvement is possible.

- Backward Stepwise Selection
  - Begins with a full OLS model containing all predictors, and then deletes one predictor at a time that improves the model the most until no further improvement is possible.

Forward Stepwise Selection

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = 0, 1, 2, \ldots, p$-1:
   a. Consider all $p - k$ models that augment the predictors in M_k with one additional predictor.
   b. Choose the best among these $p - k$ models, and call it M_{k+1}. Here best is defined as having smallest RSS or higher R^2.
3. Select a single best model from among $M_0, \ldots,$ M_p using cross-validated prediction error, C_p, AIC, or BIC.

Backward Stepwise Selection

1. Let M_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = p, p\text{-}1, 1$:
   a. Consider all k models that contain one but one of the predictors in M_k for a total of k-1 predictor.
   b. Choose the best among these k models, and call it $M_{k\text{-}1}$. Here best is defined as having smallest RSS or higher R^2.
3. Select a single best model from among M_0, . . . ,M_p using cross-validated prediction error, C_p, AIC, or BIC.

```r
require(readr)
require(broom); require(dplyr)
require(glmnet) # lars is where the LASSO functions live
pcancer <- read_delim(http://www-stat.stanford.edu/~hastie/ElemStatLearn/datasets/prostate.data, "\t", escape_double = FALSE, trim_ws = TRUE)
head(pcancer)

pr.train <- pcancer[which(pcancer$train),2:10]
pr.test <- pcancer[-which(pcancer$train),2:10]
pairs(pr.train[,1:9], pch=19, cex=.25)
round(cor(pr.train[,1:9]),3)

pr.ls <- lm(lpsa ~ ., data=pr.train)
summary(pr.ls)$coef
```

pcancer.R

###---- AIC model building ----#####

```
pr.fwd <- step(lm(lpsa ~ 1,
data=pr.train),~ lcavol + lweight +
age + lbph + svi + lcp + gleason +
pgg45,data=pr.train,
direction="forward")
```

```
pr.bck <- step(pr.ls,
direction=“backward”)
```

pcancer.R

Prediction Accuracy

- If we add bias, we can minimize or reduce some of the variance.

- How can we add bias?
  - One way to do this is to remove some of our terms.

- What are some methods for removing some of the terms?
  - P-values
  - AIC/BIC

Other Measures of Comparison

- To compare different models, we can use other approaches:
  - Adjusted R^2
  - AIC (Akaike information criterion)
  - BIC (Bayesian information criterion)
  - Mallow's C_p (equivalent to AIC for linear regression)
- These techniques adjust the training error for the model size, and can be used to select among a set of models with different numbers of variables.
- These methods add penalty to RSS for the number of predictors in the model.

Akaike Information Criteria (AIC)

$$AIC = -2\log Lik + 2p$$

- where ***Lik*** is the maximized value of the likelihood function for the estimated model.

Akaike, Hirotugu (1974). "A new look at the statistical model identification". *IEEE Transactions on Automatic Control* **19** (6): 716–723..

Bayesian Information Criteria

$$BIC = -2\log Lik + p\ln(n)$$

Schwarz, Gideon E. (1978). "Estimating the dimension of a model". *Annals of Statistics* **6** (2): 461–464.

AIC versus BIC

$$2p \quad vs. \quad p\ln(n)$$

- BIC and AIC are similar.
- Different penalty for number of parameters.
- The BIC penalizes free parameters more strongly than does the AIC.
- Implications: BIC tends to choose smaller models.
- **The larger the *n*, the more likely that AIC and BIC will disagree on model selection.**

Problems with p-values

- With p-values, we check to see if a β_n is non-0. If it is 0, we remove it from the model and create a new model.

- With large *n*, many *p*-values will be significant even if there is little to no effect.

- Increased power of large samples means you can detect smaller, subtler and more complex effects but this can cause overfitting/overcomplexity.

Problems with AIC/BIC

- AIC/BIC are measures of model fit with a penalty for the number of parameters.
- Would need to explore all models “by hand”.
- Too many models to do this even computationally.
- 2^p models to evaluate.

Adjusted R^2

- For an OLS model with *d* variables, the adjusted R^2 is calculated:

$$\text{Adjusted } R^2 = 1 - \frac{\text{RSS}/(n - p - 1)}{\text{TSS}/(n - 1)}$$

where RSS is the residual sum of squares on a training set of data; TSS is the total sum of squares.

- Unlike the other statistics, a large value of adjusted R^2 indicates a model with a small test error.

- The adjusted R^2 statistics *pays a price* for the inclusion of unnecessary variables in the model.

Mallow's C_p

- For a fitted OLS containing p predictors, the Cp estimate of test MSE:

$$Cp = \frac{1}{n}(RSS + 2p\hat{\sigma}^2)$$

where $\hat{\sigma}^2$ is an estimate of the variance of the error associated with each response measurement.

- Here, a penalty is added to the training RSS in order to adjust for the fact that the training error tends to underestimate the test error.

Prostate Cancer

```
library("ElemStatLearn")
data(prostate)
```

- lcavol: log-cancer volume
- lweight: log-prostate weight
- age: age of patient
- lbhp: log-amount of benign hyperplasia
- svi: seminal vesicle invasion
- lcp: log-capsular penetration
- gleason: Gleason Score,
- pgg45: percent of Gleason scores 4 or 5
- lpsa: the response variable, log-psa.

Prostate cancer

lasso.R

```
library("ElemStatLearn")

# load the data set "prostate"
data(prostate)
attach(prostate)
my_data<- prostate;
cor_my<-cor(my_data);

my_data$svi <- factor(my_data$svi)
my_data$gleason <- factor(my_data$gleason)

pairs(lpsa ~ lcavol + lweight + age + lbph + lcp + pgg45,
data = my_data)
cor(my_data[,c(1,2,3,4,6,8,9)])

boxplot(lpsa ~ svi, data = my_data)
fit <- lm(lpsa ~ ., data = my_data)
summary(fit)
```

lasso.R

```
my_data$lpsa <- log(my_data$lpsa)
my_data$psa <- NULL
pairs(lpsa ~ lcavol + lweight + age + lbph + lcp +
pgg45, data = my_data)

fit <- lm(lpsa ~ ., data = my_data)
summary(fit)

library(leaps)
fit <- regsubsets(lpsa ~ ., data = my_data,
intercept = TRUE, method = "exhaustive")
summary(fit)

summary(fit)$adjr2
fit <- lm(lpsa ~ lcavol + lweight + svi, data =
my_data)
plot(fit)
```

```
# load the package  "ElemStatLearn"
library("ElemStatLearn")
```

```
# load the data set "prostate"
data(prostate)
str( prostate )
# 'data.frame': 97 obs. of  10 variables:
# $ lcavol : num  -0.58 -0.994 -0.511 -1.204 0.751 ...
# $ lweight: num  2.77 3.32 2.69 3.28 3.43 ...
# $ age    : int  50 58 74 58 62 50 64 58 47 63 ...
# $ lbph   : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
# $ svi    : int  0 0 0 0 0 0 0 0 0 0 ...
# $ lcp    : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
# $ gleason: int  6 6 7 6 6 6 6 6 6 6 ...
# $ pgg45  : int  0 0 20 0 0 0 0 0 0 0 ...
# $ lpsa   : num  -0.431 -0.163 -0.163 -0.163 0.372 ...
# $ train  : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
table(prostate$train)
# FALSE  TRUE
# 30    67
# partition the original data into training and testing datasets
train <- subset( prostate, train==TRUE )[,1:9]
test  <- subset( prostate, train==FALSE)[,1:9]
# Note: There is a typo in "prostate" help file in "ElemStatLearn" package
# In "test  <- subset( prostate, train=FALSE )[,1:9]", "=" should be "==”
# check correlations
cor( prostate[,1:8] )
round(cor( train[,1:8] ),3)

# scatter plot,
pairs( prostate[,1:9], col="violet" )
```

```
# fit linear model on training dataset using LS method
trainst <- train
for(i in 1:8) {
  trainst[,i] <- trainst[,i] - mean(prostate[,i]);
  trainst[,i] <- trainst[,i]/sd(prostate[,i]);
}
fitls <- lm( lpsa ~
lcavol+lweight+age+lbph+svi+lcp+gleason+pgg45, data=trainst )
# reproduce Table 3.2 on page 50, as well as some numbers in Table
3.3 on page 63
summary(fitls)
#              Estimate Std. Error t value Pr(>|t|)
#   (Intercept)  2.46493    0.08931  27.598  < 2e-16 ***
#   lcavol       0.67953    0.12663   5.366 1.47e-06 ***
#   lweight      0.26305    0.09563   2.751  0.00792 **
#   age         -0.14146    0.10134  -1.396  0.16806
#   lbph         0.21015    0.10222   2.056  0.04431 *
#   svi          0.30520    0.12360   2.469  0.01651 *
#   lcp         -0.28849    0.15453  -1.867  0.06697 .
#   gleason     -0.02131    0.14525  -0.147  0.88389
#   pgg45        0.26696    0.15361   1.738  0.08755 .
# ---
#   Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
## Residual standard error: 0.7123 on 58 degrees
# Multiple R-squared:  0.6944,     Adjusted R-squa
# F-statistic: 16.47 on 8 and 58 DF,  p-value: 2.042
```

Conclusion: age, lcp, gleason, pgg45 are not significant at 5% significance level based on Z-test

- Question: Could we remove all the four predictors?
- Answer: Use partial F-test

```
## [1] Fit the full model and get RSSfull=0.7123^2*58=29.43
## [2] Fit the reduced model:
fitlsr <- lm( lpsa ~ lcavol+lweight+lbph+svi, data=trainst )
summary(fitlsr)
#              Estimate Std. Error t value Pr(>|t|)
#  (Intercept)  2.47142    0.08901  27.766  < 2e-16 ***
#  lcavol       0.59582    0.10910   5.461 8.85e-07 ***
#  lweight      0.23084    0.09456   2.441   0.0175 *
#  lbph         0.20313    0.10215   1.988   0.0512 .
#  svi          0.27814    0.11311   2.459   0.0167 *
#    ---
#  Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#
# Residual standard error: 0.7275 on 62 degrees of freedom
# Multiple R-squared:  0.6592,      Adjusted R-squared:  0.6372
# F-statistic: 29.98 on 4 and 62 DF,  p-value: 6.911e-14
##  then RSSreduced=0.7275^2*62=32.81
## [3] Test statistic: F=((32.81-29.43)/(62-58))/(29.43/58)=1.67
## [4] p-value: 1-pf(1.67, df1=4, df2=58)
## 0.1692794
```

Data types

- **MicroArray Data**:
  - p is number of genes, n is number of patients
- **Financial Data**:
  - *p* is number of stocks, prices, etc, n is number of time points
- **Data Mining**:
  - automated data collection can imply large numbers of variables
- **Texture Classification in Images** (eg. satellite):
  - p is number of pixels, n is number of images

Estimating the model

- Can we find an estimate for when p>>n?
- George Box (1986) Effect-Sparsity: the vast majority of factors have zero effect, only a small fraction actually affect the response.
- $y=X\beta+\varepsilon$ can still be modeled but β now must be sparse, containing a few nonzero elements, the remaining elements zero.

Commonly Used Strategies for Sparse Modeling

- All Subsets Regression
  - Fit all possible linear models for all levels of sparsity.
- Forward Stepwise Regression
  - Greedy approach that chooses each variable in the model sequentially by significance level.
- LASSO (Tibshirani1994), LARS (Efron, Hastie, Johnstone, Tibshirani 2002)
  - 'shrinks' some coefficient estimates to zero.

Model Interpretability

- When we have a large number of predictors in the model, there will generally be many that have little or no effect on the response.
- Including such irrelevant variable leads to unnecessary complexity.
- Leaving these variables in the model makes it harder to see the effect of the important variables.
- The model would be easier to interpret by removing (i.e. setting the coefficients to zero) the unimportant variables.

Shrinkage (Regularization) Methods

- The subset selection methods use OLS to fit a linear model that contains a subset of the predictors.

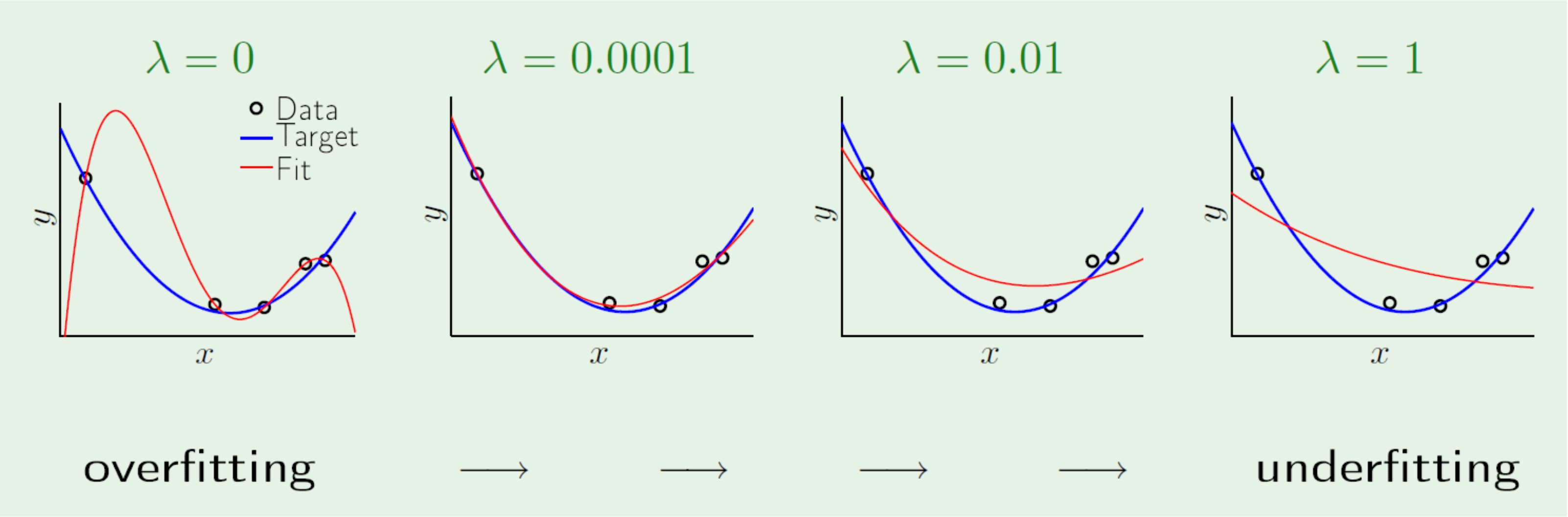
- As an alternative, we can fit a model containing all p predictors using a technique that constrains or regularizes the coefficient estimates (i.e. shrinks the coefficient estimates towards zero).

- Regularization is our first weapon to combat overfitting.

- It may not be immediately obvious why such a constraint should improve the fit, but it turns out that shrinking the coefficient estimates can significantly reduce their variance.

Shrinkage (Regularization) Methods (cont.)

- Regularization is our first weapon to combat overfitting.
- It constrains the machine learning algorithm to improve out-of-sample error, especially when noise is present.
- Look at what a little regularization can do:



Ridge Regression

- Recall that the least squares fitting procedure estimates $\beta_0, \beta_1, \ldots, \beta_p$ using the values that minimize

$$\mathrm{RSS} = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2$$

- Ridge regression is similar to OLS, except that the coefficient are estimated by minimizing a slightly different quantity

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathrm{RSS}+\lambda\sum_{j=1}^{p}\beta_j^2$$

where λ > =0 is a tuning parameter.

- Assume the data is normalized, the gradient of $\boldsymbol{\beta}_{Ridge}$ is:

$$\nabla_\beta = \frac{\partial}{\partial\beta}(\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\beta\|^2)$$

$$= \frac{\partial}{\partial\beta}\left((\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta\right)$$

$$= \frac{\partial}{\partial\beta}\left(\mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\beta - \beta^T\mathbf{X}^T\mathbf{y} + \beta^T\mathbf{X}^T\mathbf{X}\beta + \lambda\beta^T\beta\right)$$

$$= -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\beta + 2\lambda\beta = 2(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})\beta - 2\mathbf{X}^T\mathbf{y}$$

$$\nabla_\beta = 0 \quad \Rightarrow \beta_{\text{ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

Ridge Regression

- This has the effect of shrinking the estimated beta coefficients towards zero. It turns out that such a constraint should improve the fit, because shrinking the coefficients can significantly reduce their variance.

- Note that when λ = 0, the penalty term as no effect, and ridge regression will produce the OLS estimates. Thus, selecting a good value for λ is critical (can use cross-validation for this).

$$\hat{y} = \mathbf{X}\beta_{\text{ridge}} = \mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

- Consider the fitted response
- SVD of **X** $\quad \mathbf{X} = \mathbf{UWV}^T$

$$\begin{aligned}
\hat{y} &= \mathbf{UWV}^T\left((\mathbf{UWV}^T)^T(\mathbf{UWV}^T)+\lambda\mathbf{I}\right)^{-1}(\mathbf{UWV}^T)^T\mathbf{y} \\
&= \mathbf{UWV}^T\left(\mathbf{VW}^T\mathbf{U}^T\mathbf{UWV}^T+\lambda\mathbf{I}\right)^{-1}\mathbf{VW}^T\mathbf{U}^T\mathbf{y} \\
&= \mathbf{UWV}^T\left(\mathbf{VW}^2\mathbf{V}^T+\mathbf{V}\lambda\mathbf{IV}^T\right)^{-1}\mathbf{VW}^T\mathbf{U}^T\mathbf{y} \\
&= \mathbf{UWV}^T\left(\mathbf{V}(\mathbf{W}^2+\lambda\mathbf{I})\mathbf{V}^T\right)^{-1}\mathbf{VW}^T\mathbf{U}^T\mathbf{y} \\
&= \mathbf{UWV}^T(\mathbf{V}^T)^{-1}(\mathbf{W}^2+\lambda\mathbf{I})^{-1}(\mathbf{V})^{-1}\mathbf{VW}^T\mathbf{U}^T\mathbf{y} \\
&= \mathbf{UW}(\mathbf{W}^2+\lambda\mathbf{I})^{-1}\mathbf{W}^T\mathbf{U}^T\mathbf{y} \\
&= \sum_{i=1}^{M} u_i \frac{\sigma_i^2}{\sigma_i^2+\lambda} u_i^T\mathbf{y}, \quad \text{where} \sum_{i=1}^{M}\frac{\sigma_i^2}{\sigma_i^2+\lambda} \triangleq df(\lambda) \in \mathcal{R}
\end{aligned}$$

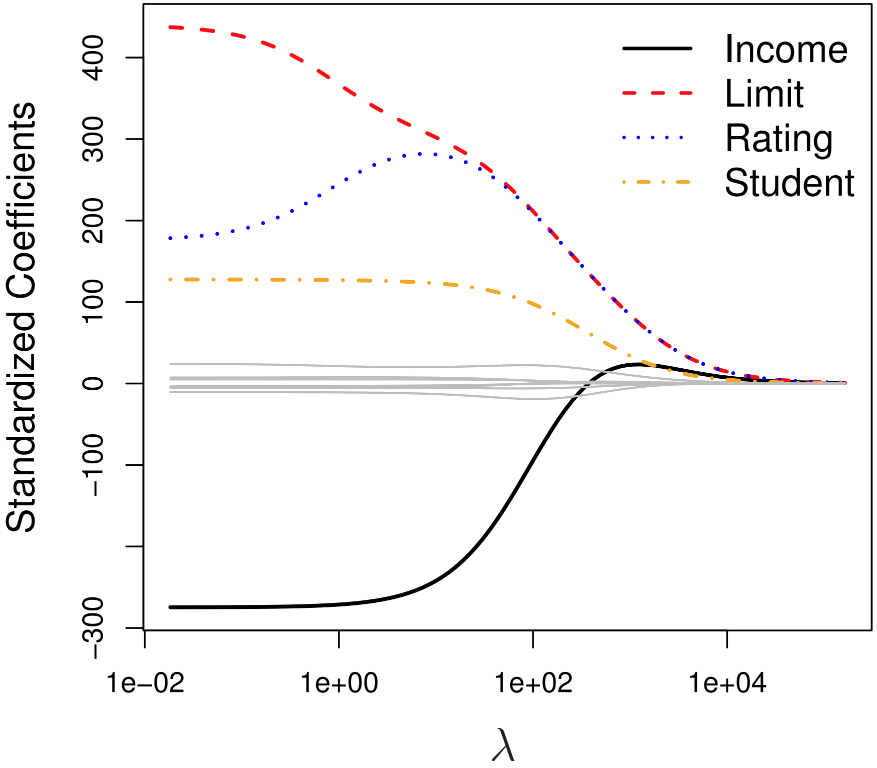
Ridge regression shrinks the coefficients with respect to the orthonormal basis formed by the principal components

where the u_j are the columns of U

Ridge Regression (cont.)

- As λ increases, the standardized ridge regression coefficients shrinks towards zero.

- Thus, when λ is extremely large, then all of the ridge coefficient estimates are basically zero; this corresponds to the *null model* that contains no predictors.



Ridge Regression (cont.)

- The standard OLS coefficient estimates are *scale equivariant*.
- However, the ridge regression coefficient estimates can change *substantially* when multiplying a given predictor by a constant, due to the sum of squared coefficients term in the penalty part of the ridge regression objective function.
- Thus, it is best to apply ridge regression after *standardizing the predictors*:

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}}$$

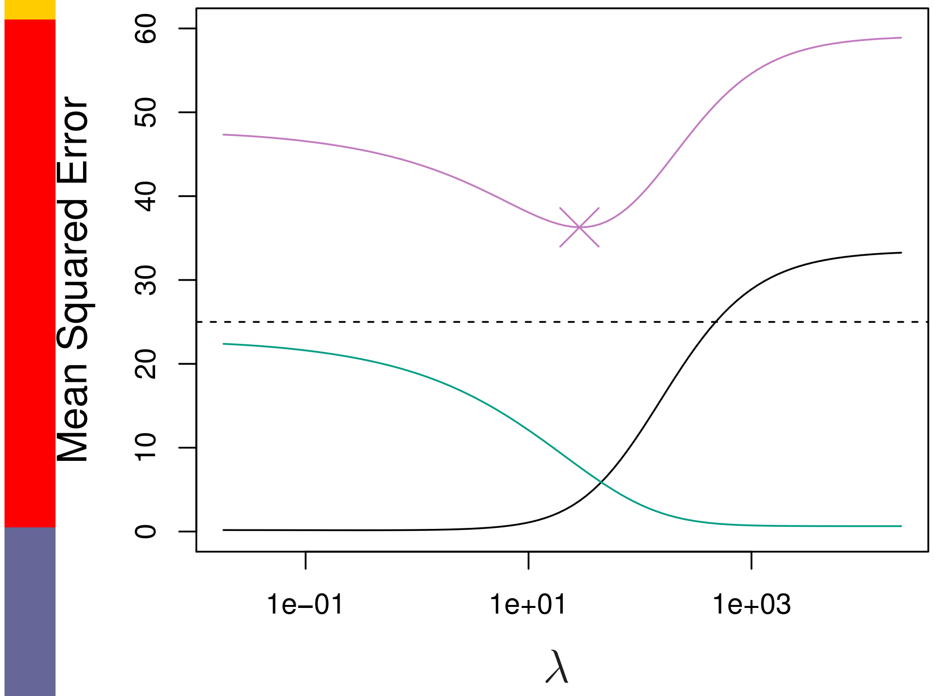
Ridge Regression (cont.)

- It turns out that the OLS estimates generally have low bias but can be highly variable. In particular when *n* and *p* are of similar size or when $n < p$, then the OLS estimates will be extremely variable

- The penalty term makes the ridge regression estimates *biased* but can also substantially reduce variance

- As a result, there is a bias/variance trade-off.

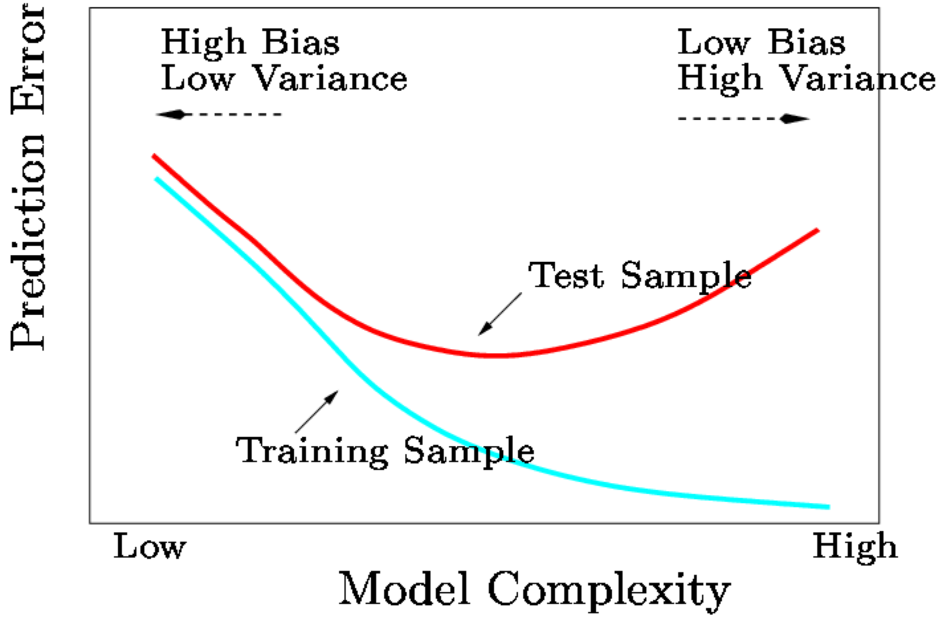
Ridge Regression (cont.)



- Black = Bias
- Green = Variance
- Purple = MSE

- Increased λ leads to increased bias but decreased variance

Ridge Regression (cont.)



- In general, the ridge regression estimates will be more biased than the OLS ones but have lower variance.
- Ridge regression will work best in situations where the OLS estimates have high variance.

Ridge Regression (cont.)

Computational Advantages of Ridge Regression

- If *p* is large, then using the best subset selection approach requires searching through enormous numbers of possible models.

- With ridge regression, for any given λ we only need to fit one model and the computations turn out to be very simple.

- Ridge regression can even be used when *p* > *n*, a situation where OLS fails completely (i.e. OLS estimates do not even have a unique solution).

Ridge Regression (cont.)

- In matrix form:

$$\mathrm{RSS}(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta,$$

the ridge regression solutions are easily seen to be

$$\hat{\beta}^{\text{ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y},$$

- The solution adds a positive constant to the diagonal of $\mathbf{X}^T\mathbf{X}$ before inversion (making the problem non-singular).

- The *singular value decomposition* (SVD) of the centered matrix **X** gives us some additional insight into the nature of ridge regression.

Ridge Regression (cont.)

- The SVD of the *N* x *p* matrix **X** has the form **X = UDV**^{T}
- Here, **U** and **V** are *N* x *p* and *p* x *p* orthogonal matrices, with the columns of **U** spanning the column space of **X**, and the columns of **V** spanning the row space.
- **D** is a *p* x *p* diagonal matrix, with diagonal entries $d_1 \geq d_2 \geq \cdots d_p \geq 0$ called singular values of **X**.
- If one or more values $d_j = 0$, **X** is singular.

Ridge Regression (cont.)

- Using SVD, we can write the OLS fitted vector as:

$$\begin{aligned}\mathbf{X}\hat{\beta}^{\text{ls}} &= \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\ &= \mathbf{U}\mathbf{U}^T\mathbf{y},\end{aligned}$$

- The ridge regression solutions are:

$$\begin{aligned}\mathbf{X}\hat{\beta}^{\text{ridge}} &= \mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} \\ &= \mathbf{U}\,\mathbf{D}(\mathbf{D}^2+\lambda\mathbf{I})^{-1}\mathbf{D}\,\mathbf{U}^T\mathbf{y} \\ &= \sum_{j=1}^{p}\mathbf{u}_j\frac{d_j^2}{d_j^2+\lambda}\mathbf{u}_j^T\mathbf{y},\end{aligned}$$

where $\mathbf{u}_j$ are the columns of **U**.

Ridge Regression (cont.)

- Like linear regression, ridge regression computes the coordinates of **y** with respect to the orthonormal basis **U**.

- It then *shrinks* these coordinates by the factor $\frac{d_j^2}{d_j^2 + \lambda}$.

- This means that a greater amount of shrinkage is applied to the coordinates of basis vectors with smaller $\frac{d_j^2}{d_j^2 + \lambda}$.

- The SVD of the centered matrix **X** is another way of expressing the *principal components* of the variables in **X**.

Ridge Regression (cont.)

- Thus, we have $\mathbf{X}^T\mathbf{X} = \mathbf{V}\mathbf{D}^2\mathbf{V}^T$, which is the *eigen decomposition* of $\mathbf{X}^T\mathbf{X}$.

- The eigenvectors v_j (columns of **V**) are also called the *principal components* directions of **X**.

- The first principal component direction v_1 has the property that $\mathbf{z}1 = \mathbf{X}v_1$ has the largest sample variance amongst all normalized linear combinations of the columns of **X**.

- The small singular values d_j correspond to directions in the column space of **X** having small variance, and ridge regression shrinks these directions the most.

LASSO

(Least Absolute Shrinkage and Selection Operator)

The Lasso

- One significant problem of ridge regression is that the penalty term will never force any of the coefficients to be exactly zero.
- Thus, the final model will include all *p* predictors, which creates a challenge in model interpretation
- A more modern machine learning alternative is the *lasso*.
- The lasso works in a similar way to ridge regression, except it uses a different penalty term that shrinks some of the coefficients exactly to zero.

LASSO and LARS: a quick tour

- LASSO solves

$$\min_{\beta} \| \mathbf{y} - \mathbf{X}\beta \|_2^2 \text{ s.t.} \| \beta \|_1 \leq t$$ for a given t

- LARS: a stepwise approximation to LASSO
  - Advantage: guaranteed to stop in n steps

LASSO Coefficient against Factor *s*

- The shrinkage factor s is defined as

$$s = t / \sum_1^p |\hat{\beta}_j|$$

- If $t = \sum_{i=1}^{P} | b_{\mathrm{LS}} |$, equivalently λ=0, there is no shrinkage at all.
- If s is small enough, some coefficients are 0 and the LASSO acts as subset selection method.

Prostate Cancer

```
library("ElemStatLearn")
data(prostate)
```

- lcavol : log-cancer volume
- lweight : log-prostate weight
- age : age of patient
- lbhp : log-amount of benign hyperplasia
- svi : seminal vesicle invasion
- lcp : log-capsular penetration
- gleason : Gleason Score,
- pgg45 : percent of Gleason scores 4 or 5
- lpsa is the response variable, log-psa.

The Lasso

- The lasso coefficients minimize the quantity:

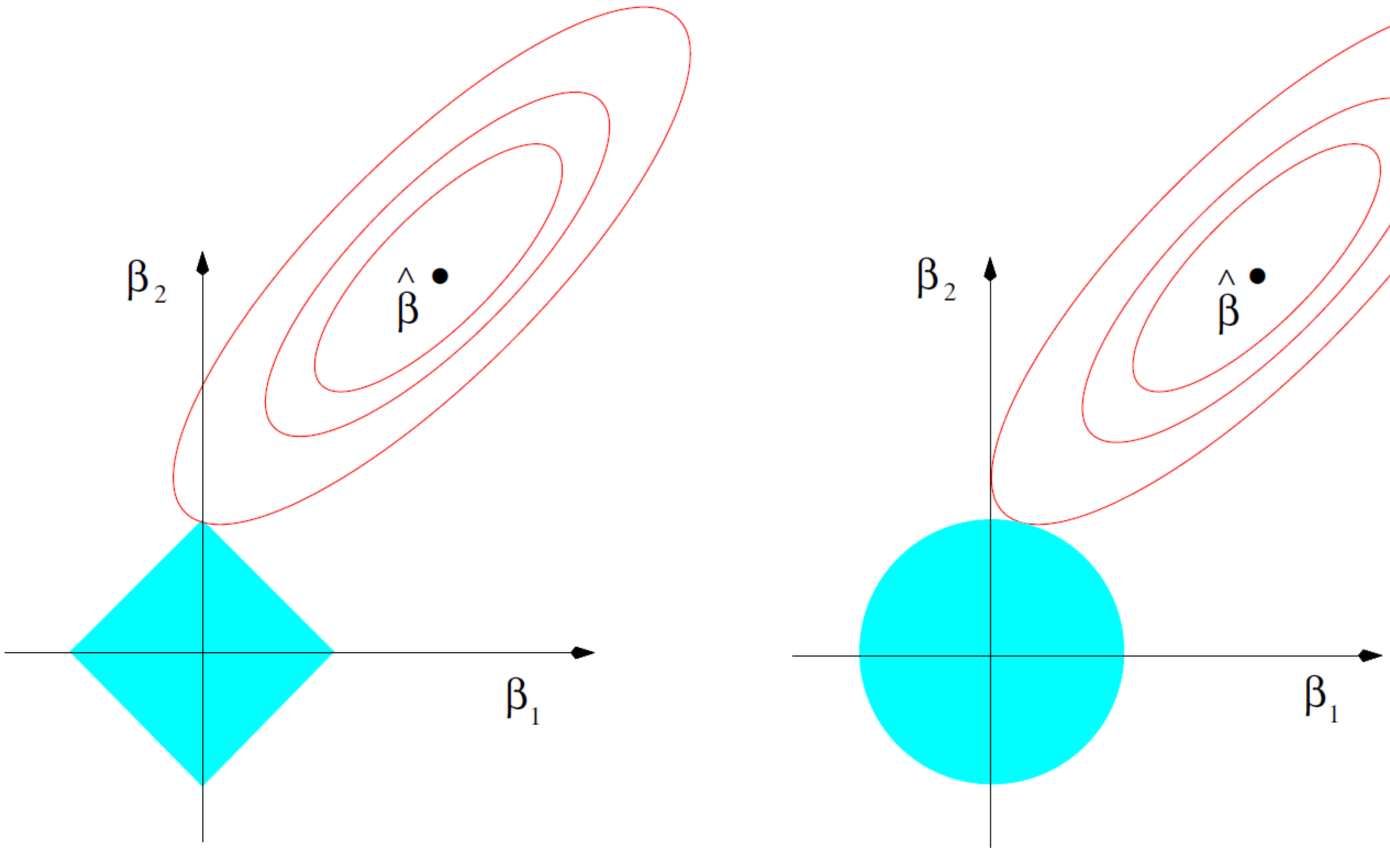
$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\left|\beta_j\right|=\mathrm{RSS}+\lambda\sum_{j=1}^{p}\left|\beta_j\right|$$

- The key difference from OLS and lasso is the lasso uses a penalty for including larger coefficients. This has the effect of forcing some of the coefficients to be exactly equal to zero when the tuning parameter λ is sufficiently large.

- Thus, the lasso performs variable/feature selection.

Why Sparse?

Lasso and Ridge regression



$$\min_{\beta} \| \mathbf{y} - \mathbf{X}\beta \|_2^2 + \lambda \| \beta \|_1$$

$$\min_{\beta} \| \mathbf{y} - \mathbf{X}\beta \|_2^2 + \lambda \| \beta \|^2$$

The Lasso (cont.)

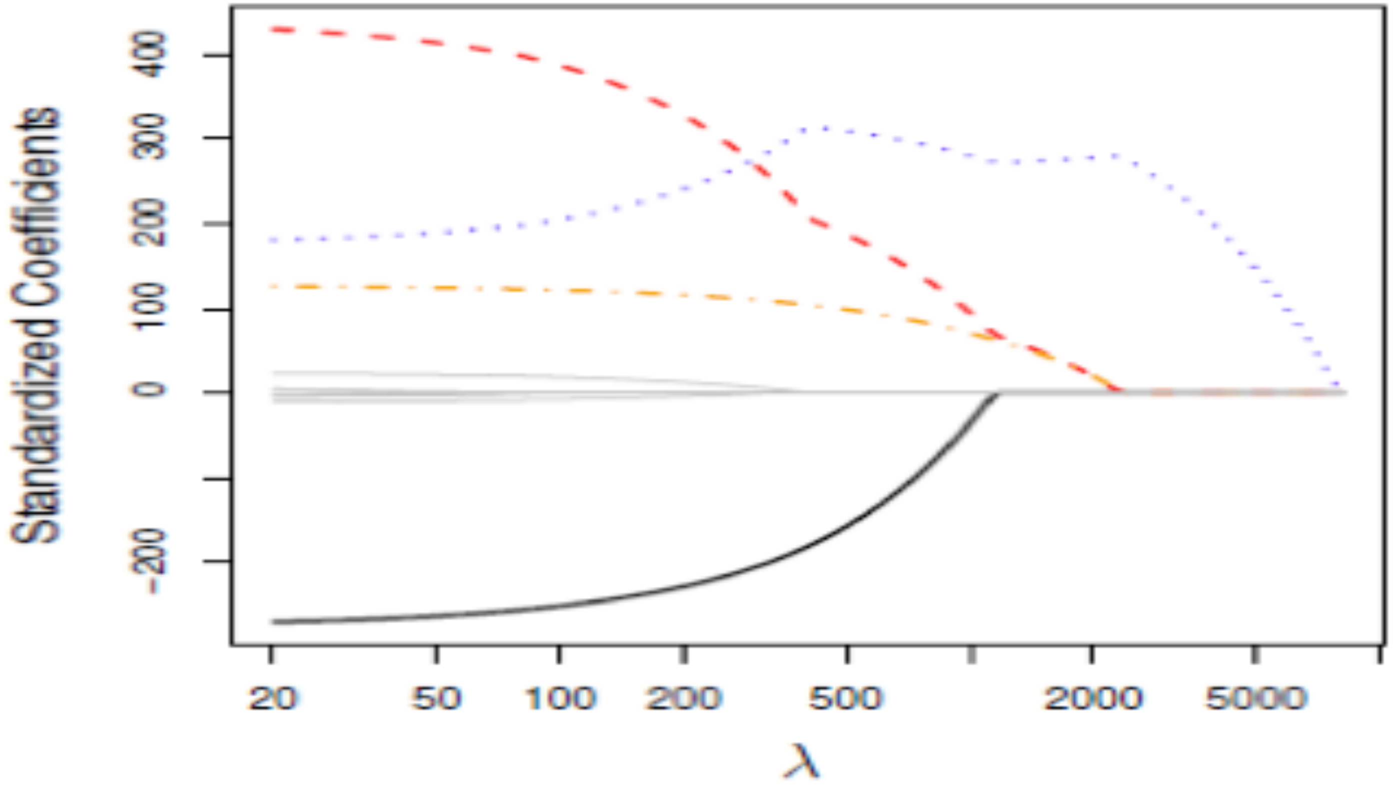
- The standard OLS coefficient estimates are *scale equivariant*.
- However, the lasso regression coefficient estimates can change *substantially* when multiplying a given predictor by a constant, due to the sum-of-absolute-values-of-the-coefficients term in the penalty part of the lasso regression objective function.
- Thus, it is best to apply lasso regression after *standardizing the predictors*:

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}}$$

The Lasso (cont.)

- Notice that the solution is indexed by the parameter λ.
  - So for each λ, we have a solution
  - The λ's trace out a path of solutions (see next slide)
- λ is the shrinkage parameter
  - λ controls the size of the coefficients
  - λ controls the amount of regularization
  - As λ ➔ 0, we obtain the OLS solution
  - As λ ➔ 1, we obtain $\beta^{lasso}=0$ (intercept only model)

The Lasso (cont.)



- When λ = 0, then the lasso simply gives the OLS fit.
- When λ becomes sufficiently large, the lasso give the null model in which all coefficient estimates equal zero.

Other Shrinkage Methods

- Ridge regression = $\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2=\text{RSS}+\lambda\sum_{j=1}^{p}\beta_j^2$
- Elastic Net = combination of ridge and lasso penalty (with 2 different lambdas)
- One significant problem of ridge regression is that the penalty term will never force any of the coefficients to be exactly zero.
- Thus, the final model will include all *p* predictors, which creates a challenge in model interpretation

Lasso vs. Ridge Regression

- The lasso has a major advantage over ridge regression, in that it produces simpler and more interpretable models that involved only a subset of predictors.

- The lasso leads to qualitatively similar behavior to ridge regression, in that as λ increases, the variance decreases and the bias increases.

- The lasso can generate more accurate predictions compared to ridge regression.

- Cross-validation can be used in order to determine which approach is better on a particular data set.

Selecting the Tuning Parameter λ

- As for subset selection, for ridge regression and lasso we require a method to determine which of the models under consideration in best; thus, we required a method selecting a value for the tuning parameter λ or equivalently, the value of the constraint *s*.

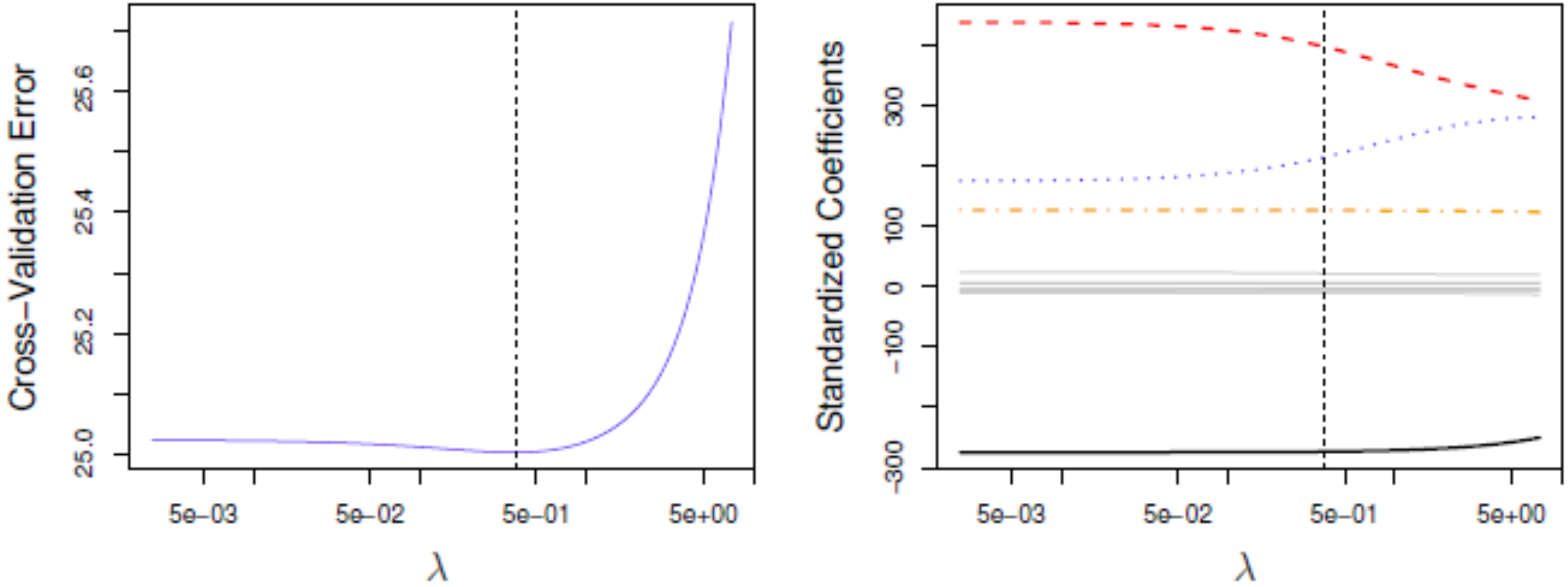
- Select a grid of potential values; use cross-validation to estimate the error rate on test data (for each value of λ) and select the value that gives the smallest error rate.

- Finally, the model is re-fit using all of the variable observations and the selected value of the tuning parameter λ.

Cross-Validation (k-fold)

- Original sample is partitioned into K subsamples
- 1 subsample is used for validation, K-1 is used for training
- Process (e.g. best fit, minimized loss function, etc.) conducted, then repeated K times
- Results are then averaged to produce final result
- 10 fold is most commonly used (also 20)

Selecting the Tuning Parameter λ: Credit Data Example



Left: Cross-validation errors that result from applying ridge regression to the `Credit` *data set with various values of λ.*
Right: The coefficient estimates as a function of λ. The vertical dashed lines indicates the value of λ selected by cross-validation.

LASSO R implementation

- **Glmnet** (Lasso and elastic-net regularized generalized linear models)
  - Fits linear models or generalized linear model penalizing the maximum likelihood with both the LASSO method and the Ridge Regression and also the mixture of the two penalties (the elastic net). To find the minimum the glmnet algorithm uses cyclical coordinate descent.
- **Lars** (Least Angle Regression)
  - A new model-selection method based on the traditional forward selection, i.e. given a collection of possible explanatory variables, we select the one having largest absolute correlation with the response y. In the Lars package there is an implementation of the LASSO method.

Generate random data

```
library(MASS)
# Package needed to generate  correlated precictors
library(glmnet)  # Package to fit
ridge/lasso/elastic net models

# Generate data
set.seed(19875)  # Set seed for
reproducibility
n <- 1000  # Number of observations
p <- 5000  # Number of predictors included
in model
real_p <- 15  # Number of true predictors
x <- matrix(rnorm(n*p), nrow=n, ncol=p)
y <- apply(x[,1:real_p], 1, sum) + rnorm(n)
```

lasso_ridge_elastic.R

Apply LASSO, Ridge and Elastic

```
# Split data into train (2/3) and test (1/3) sets
train_rows <- sample(1:n, .66*n)
x.train <- x[train_rows, ]
x.test <- x[-train_rows, ]

y.train <- y[train_rows]
y.test <- y[-train_rows]

# Fit models
# (For plots on left):
fit.lasso <- glmnet(x.train, y.train,
family="gaussian", alpha=1)
fit.ridge <- glmnet(x.train, y.train,
family="gaussian", alpha=0)
fit.elnet <- glmnet(x.train, y.train,
family="gaussian", alpha=.5)
```

```
# 10-fold Cross validation for each alpha = 0, 0.1, ... , 0.9,
1.0
# (For plots on Right)
# Assign a value to a name in an environs:
for (i in 0:10){
  assign(paste(“fit”, i, sep=“”), cv.glmnet(x.train, y.train,
type.measure=“mse”,  alpha=i/10,family="gaussian"))
}

# Plot solution path
par(mfrow=c(3,2))
# For plotting options, type '?plot.glmnet' in R console
plot(fit.lasso, xvar="lambda")
plot(fit10, main="LASSO")

plot(fit.ridge, xvar="lambda")
plot(fit0, main="Ridge")

plot(fit.elnet, xvar="lambda")
plot(fit5, main="Elastic Net")
```

Example: mtcars

- The dataset contains data extracted from the Motor Trend US magazine, and comprises fuel consumption and 10 aspects of automobile design and performance for 32 models of car.
- There are 32 observation and 10 features, the response variable we want to study is mpg, that is the miles per gallon (or fuel efficiency).
- The explanatory variables are:
  - cyl : Number of cylinders
  - disp : Displacement (volume of the engine)
  - hp : Gross horsepower
  - drat : Rear axle ratio
  - wt : Weight (1000 lbs)
  - qsec : 1/4 mile time
  - vs : V/S engine
  - am : Transmission (0 = automatic, 1 = manual)
  - gear : Number of forward gears
  - carb : Number of carburetors

Exercise

- Apply forward, backward stepwise regression, LASSO, Ridge and Elastic-net to mtcars data.

Least angle regression

LARS

Stepwise, Forward Stagewise, Least Angle

- **Stepwise regression**:
  - Pick predictor most correlated with y
  - Bring predictor completely into model (full LS   fit)
- **Forward stagewise**:
  - Pick predictor most correlated with *y*
  - Increment coefficient for predictor
- **Least Angle Regression**:
  - Pick predictor most correlated with *y*
  - Bring predictor into model only to extent it is better than others
  - Move in least-squares direction until another variable is as correlated

Least-angle regression

- In stepwise selection, the algorithm decides which variable has the highest F-score and then fully adds it to the model.
- The algorithm assumes

$$y = X\beta$$

- Variables are normalized to have mean 0 and variance 1.

$$\hat{y}_i = \frac{y_i - \overline{y}}{s_y}, \forall i$$

$$\hat{x}_{ij} = \frac{x_{ij} - \overline{x_j}}{s_{xj}}, \forall i, j$$

LARS

1. Start off with an empty model and calculate the correlation between *y* and every column (variable) of *X*.

2. Add the variable with the highest correlation. Start increasing the coefficient belonging to that variable until there is another variable with the same correlation.

3. Continue the procedure until all variables are added or in the case of n < p when n steps are taken.

A “more democratic” version of forward stepwise regression.

- Start with r = y, $\hat{\beta}_1, \hat{\beta}_2, \ldots \hat{\beta}_p = 0$. Assume x_j standardized.
- Find predictor x_j most correlated with r.
- Increase β_j in the direction of sign(r, x_j) until some other competitor x_k has as much correlation with current residual as does x_j.
- Move ($\hat{\beta}_j$, $\hat{\beta}_k$) in the joint least squares direction for (x_j, x_k) until some other competitor x_l has as much correlation with the current residual.
- Continue in this way until all predictors have been entered. Stop when (r, x_j) = 0 ∀ j, i.e. OLS solution.

LARS

- The model begins with a prediction vector $\mu_0 = 0$. The model is completely empty and all variables are normalized.
  - Calculating the correlations between x_i and y

$$c_j = \mathbf{corr}(\mathbf{x_j}, \mathbf{y}) = \frac{\mathbf{cov}(\mathbf{x_j}, \mathbf{y})}{\sigma_{x_j}\sigma_y} = \frac{\mathbf{E}[(x_j - \mathbf{E}(x_j))(y - \mathbf{E}(y))]}{\sigma_{x_j}\sigma_y}$$

  - Choosing the variable
    - Uses only those variables that have the highest correlations with the current residual.

$$C = max|c_j|, \mathcal{A} = \{j : c_j = C\}$$

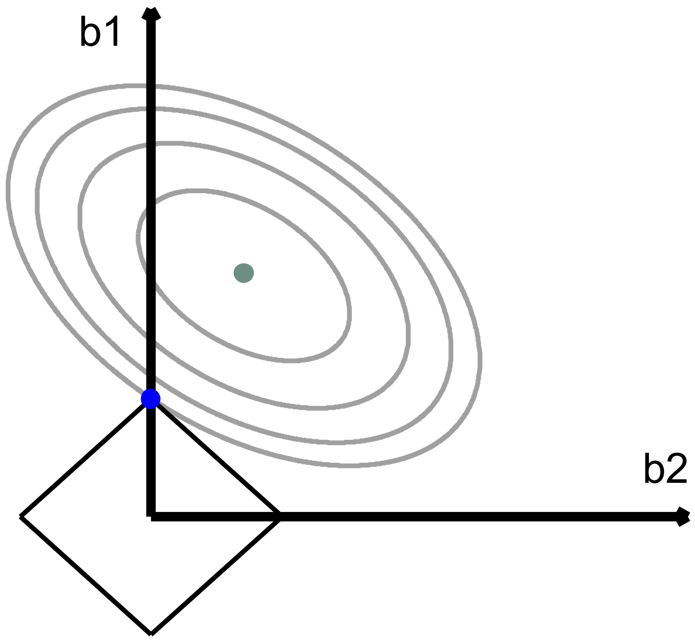
  - Choosing the right direction
  - Choosing the parameter

LARS performance

- Lars is expected to be useful on dependent datasets as it does not add variables completely to the active set. So when two variables are highly correlated, Lars is expected to add both of them to the model.

Lasso graphically

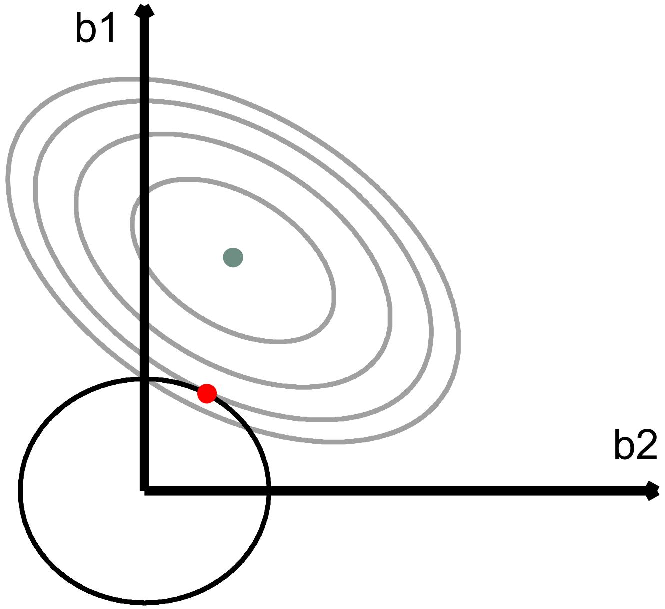
- The constraints can be seen below.



One property of this constraint is that there will be coefficients =0 for a subset of variables

Ridge Regression

- The constraints can be seen below.



The coefficients are shrunk but does not have the property of parsimony

Least Angle Regression

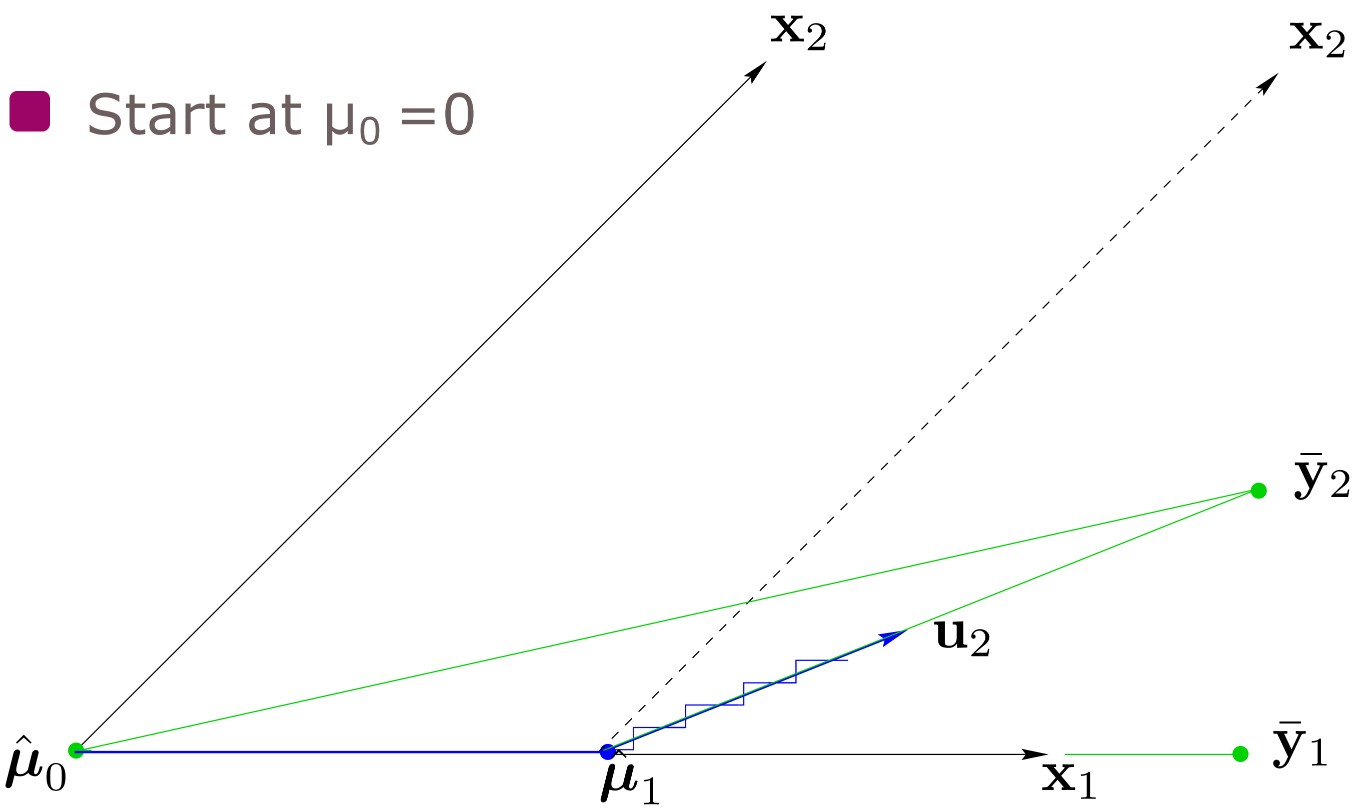
- The LAR**S** (S suggesting La**S**so and **S**tagewise)
- Starts like classic Forward Selection
  - Find predictor x_{j1} most correlated with the current residual
  - Make a step (epsilon) large enough until another predictor x_{j2} has as much correlation with the current residual
  - LARS – now step in the direction equiangular between two predictors until x_{j3} earns its way into the "correlated set"

Least Angle Regression Geometrically

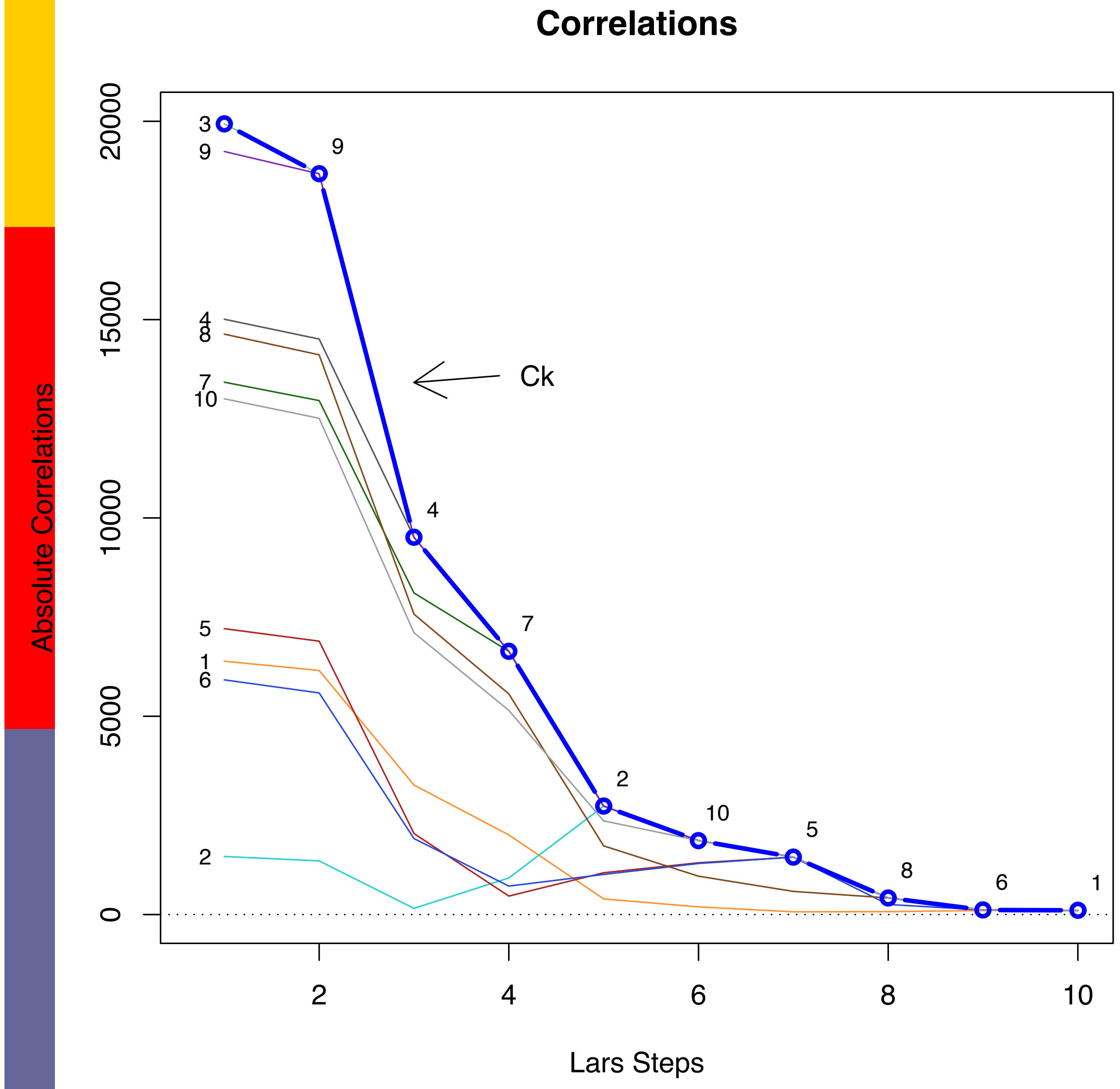
- Two covariates x_1 and x_2 and the space $L(x_1, x_2)$ that is spanned by them

- Start at $\mu_0 = 0$

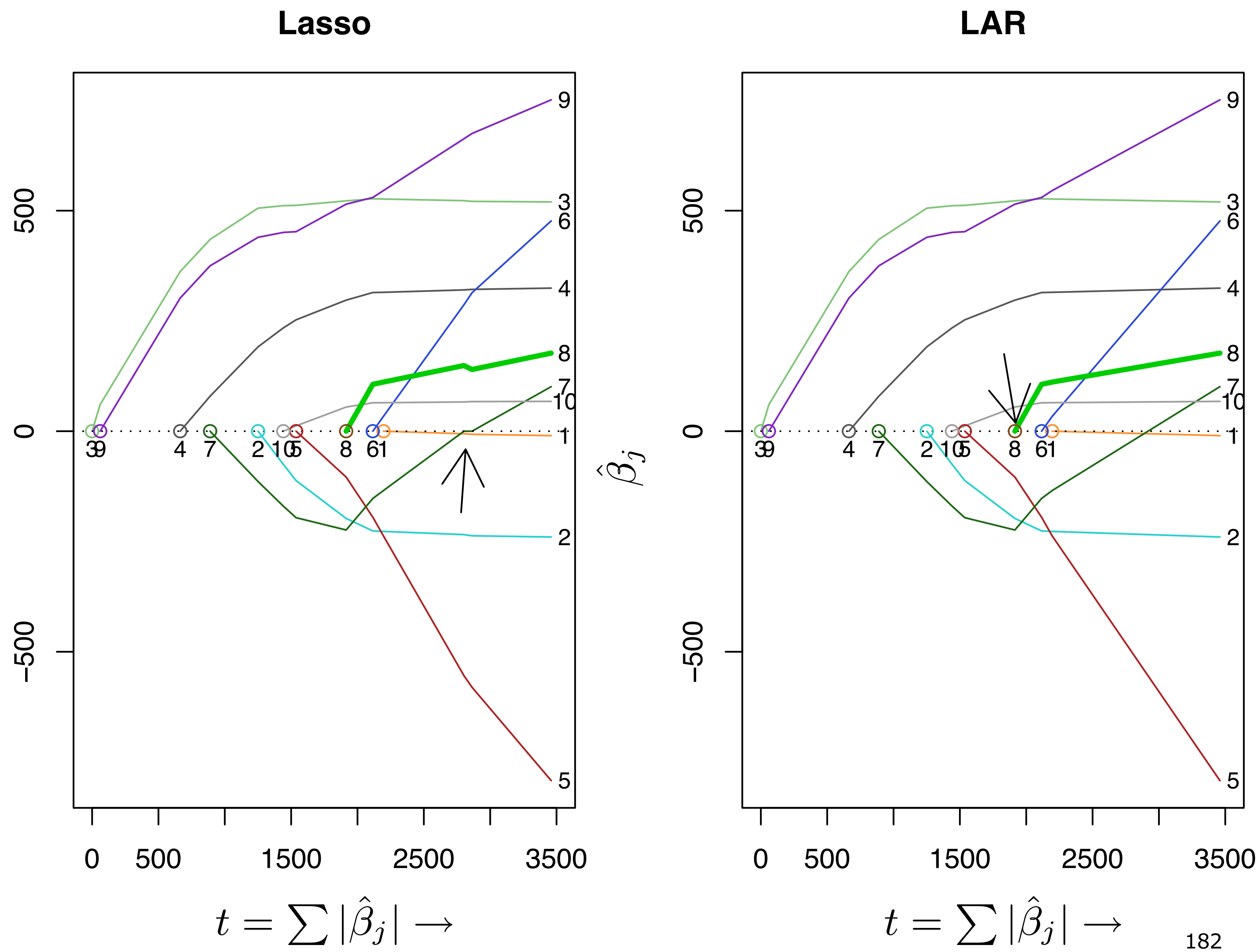
- y_2 is the projection of y onto $L(x_1, x_2)$



The LAR direction u_2 at step 2 makes an equal angle with x_1 and x_2.

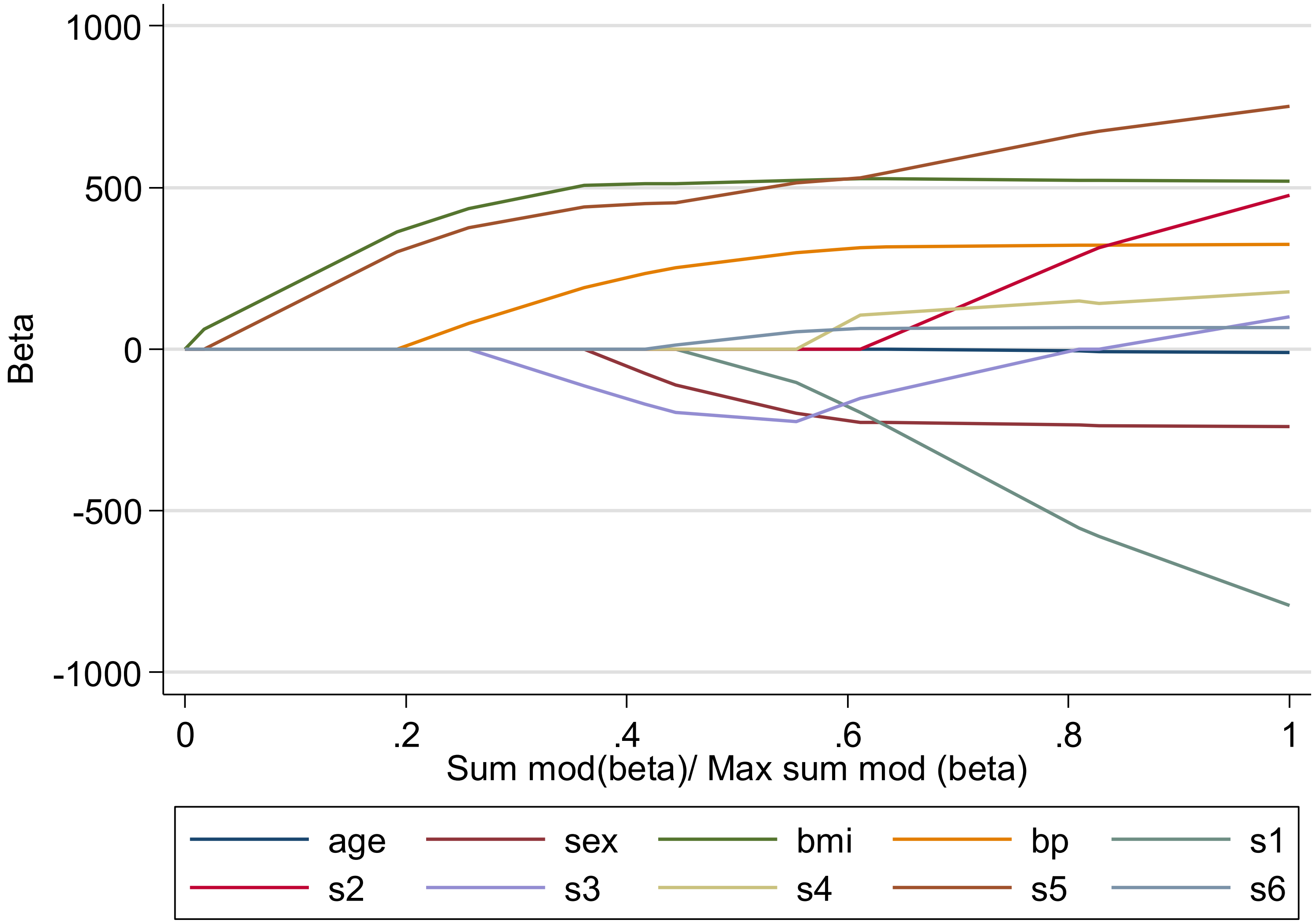


- Maximal correlations decrease with steps.
- Lars/lasso decreases RSS optimally with increase in $||\beta||_1$.
- *Dantzig Selector* decreases maximum $|\langle \mathbf{x}_j, \mathbf{r}\rangle|$ optimally with increase in $||\beta||_1$. (Candes and Tao, 2006)



Lasso
LAR
500
0
−500
$\hat{\beta}_j$
0
500
1500
2500
3500
$t = \sum |\hat{\beta}_j| \rightarrow$
9
3
6
4
8
7
10
1
2
5
3 9
4
7
2 10 5
8
6 1

Graph output



LAR, LASSO and forward stagewise

- Not always identical
- In orthogonal predictor case: yes
- In hard to verify case of monotone coefficient paths: yes
- In general, al most!
- LAR algorithm can be simply modified to give both Lasso and Forward Stagewise paths.

Example

- https://rdrr.io/cran/ElemStatLearn/man/prostate.html

```
library("ElemStatLearn")

# load the data set "prostate"
data(prostate)
```