Introduction to Statistical Computing

Seyoung Park

October 30, 2018

Objective

- Linear Regression Analysis with R
- Learn "Estimation", "Inference" and "Shrinkage methods (Lasso regression, Ridge regression)" with linear regression model

Install “faraway” package

All datasets used in this topic are from “faraway” package.

Reference book : “Linear models with R” by Julian J. Faraway

```r
install.packages("faraway")
library("faraway")
```

Regression Analysis

- Regression analysis: used for modeling the relationship between explanatory variables (X) and a dependent variable (Y)
- Regression model: $Y = f(X)$. Here $f(\cdot)$ explains the regression relationship
- Example 1: X: height, Y: weight
- Example 2: X: Math score, Y: total score

Simple Linear Regression

- Simple linear regression model is based on the following linear model: $Y = \beta_0 + \beta_1 X$
- Simple linear regression analysis: linear regression model with a single explanatory variable (X), i.e. $Y = \beta_0 + \beta_1 X + \epsilon$. Here ϵ represents a random error, e.g. $\epsilon \sim N(0, \sigma^2)$
- Given n data pairs $\{(x_i, y_i), i = 1, \cdots, n\}$, we can write

  $$y_i = \beta_0 + \beta_1 x_i + \epsilon_i,$$

  where ϵ_is are indepedent random errors.
- β_0 and β_1 are unknown parameters
- How to interpret β_0 and β_1? β_1 is the effect of X on Y.
- Is the obtained β_1 statistically significant?

“stat500” Example

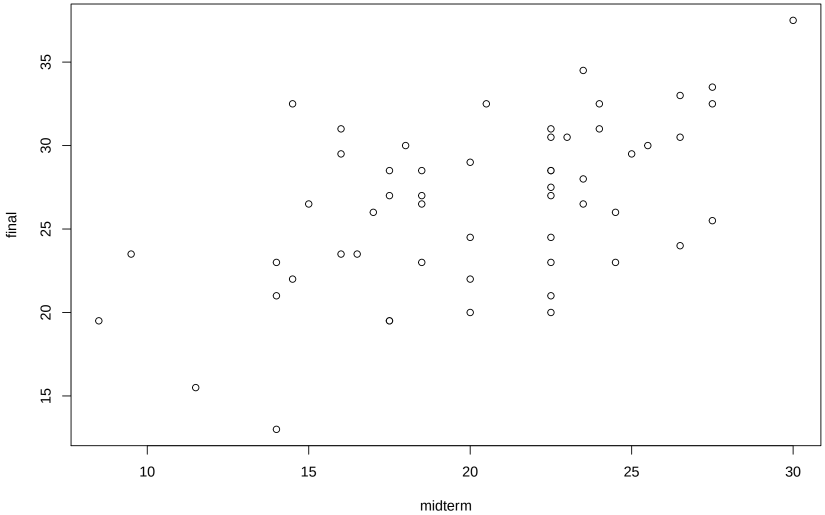
```r
# use "stat500" data
# stat500 is 55 by 4 dimensional data
library("faraway")
data(stat500)
head(stat500)
```

```
##   midterm final   hw total
## 1    24.5  26.0 28.5  79.0
## 2    22.5  24.5 28.2  75.2
## 3    23.5  26.5 28.3  78.3
## 4    23.5  34.5 29.2  87.2
## 5    22.5  30.5 27.3  80.3
## 6    16.0  31.0 27.5  74.5
```

```r
# We can specify column/row names using
# "colnames" and "rownames"
```

"stat500" Example (cont.)

```r
# scatter plot
plot(final ~ midterm, data = stat500)
```



“stat500” Example (cont.)

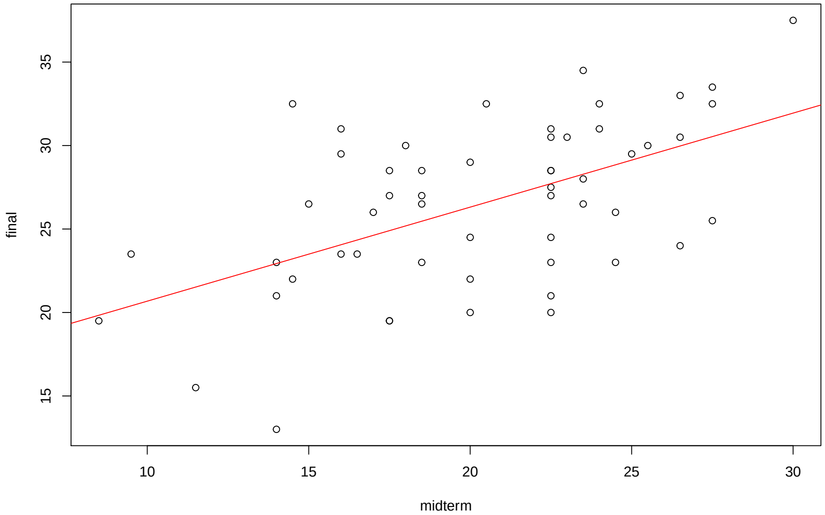
Fitted linear regression model is $\textit{final} = 15.05 + 0.56\ \textit{midterm}$

```r
# apply linear regression model
lm(final ~ midterm, data = stat500)
```

```
##
## Call:
## lm(formula = final ~ midterm, data = stat500)
##
## Coefficients:
## (Intercept)      midterm
##     15.0462       0.5633
```

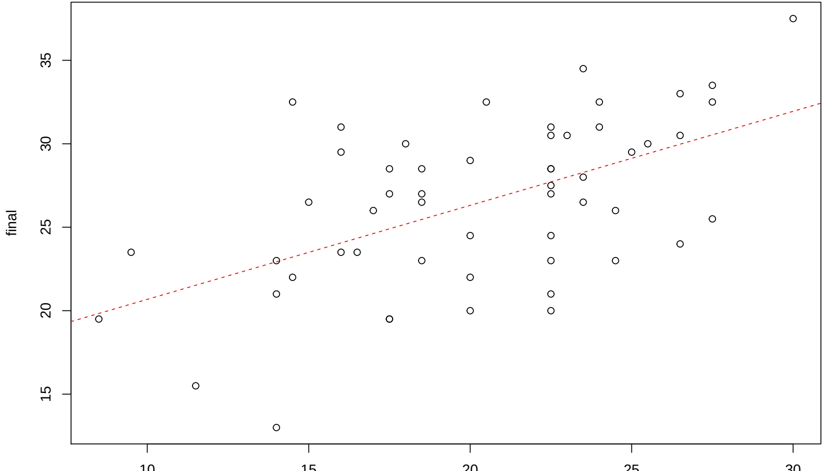
“stat500” Example (cont.)

```
plot(final ~ midterm, data = stat500) # scatter plot
abline(lm(final ~ midterm, data = stat500), col ="red")
```



"stat500" Example (cont.)

```
# scatter plot and fitted regression line (version2)
plot(final ~ midterm, data = stat500) # scatter plot
fit = lm(final ~ midterm, data = stat500)
abline(coef(fit), col ="red", lty = 2)
```



“stat500” Example (cont.)

```
fit = lm(final ~ midterm, data = stat500)
# Check quantities in the "fit"
names(fit)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "ra
##  [5] "fitted.values" "assign"        "qr"            "df
##  [9] "xlevels"       "call"          "terms"         "mo
```

```
# detailed results
summary(fit)
```

```
##
## Call:
## lm(formula = final ~ midterm, data = stat500)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.932 -2.657  0.527  2.984  9.286
```

Multiple Linear Regression

- Multiple linear regression analysis is based on multiple explanatory variables $X_1, X_2, \cdots, X_p$ and $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots \beta_p X_p + \epsilon$
- Given n data pairs $\{(x_{i1}, \cdots, x_{ip}, y_i), i = 1, \cdots, n\}$, we can write
$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \beta_p x_{ip} + \epsilon_i,$$
where ϵ_is are indepedent random errors.
- How to interpret $\beta_1, \cdots, \beta_p$? β_j is the effect of X_j on Y when the other $p - 1$ explanatory variables are fixed.
- Is the obtained linear regression model statistically significant? using F-test
- Are the obtained β_js statistically significant? using t-test or F-test
- How to analyze when there are too many explanatory variables (i.e. p is too large) ?

“stat500” Example

Fitted linear regression model is
$final = 16.81 + 0.57\ midterm - 0.08\ hw$

```r
# use "stat500" data
# stat500 is 55 by 4 dimensional data
data(stat500)
lm(final ~ midterm + hw, data = stat500)
```

```
##
## Call:
## lm(formula = final ~ midterm + hw, data = stat500)
##
## Coefficients:
## (Intercept)      midterm           hw
##    16.81061      0.58179     -0.08157
```

“stat500” Example (cont.)

```r
summary(lm(final ~ midterm + hw, data = stat500))
```

```
##
## Call:
## lm(formula = final ~ midterm + hw, data = stat500)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.0388  -2.5964   0.3714   3.0063   9.3497
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 16.81061    4.08112   4.119 0.000137 ***
## midterm      0.58179    0.12445   4.675 2.12e-05 ***
## hw          -0.08157    0.14916  -0.547 0.586836
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
```

How to estimate coefficients? Least squares

- Data: $\{(x_{i1}, \cdots, x_{ip}, y_i), i = 1, \cdots, n\}$
- Find $\beta_0, \beta_1, \cdots, \beta_p$ that minimizes the sum of squared residuals (SSR):

$$\text{minimize} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}))^2$$

  In vector form

$$\text{minimize} \sum_{i=1}^{n} \|y_i - x_i' \beta\|^2,$$

  where $\beta = (\beta_0, \beta_1, \cdots, \beta_p)$ and $x_i = (1, x_{i1}, \cdots, x_{ip})'$
- The obtained minimizer $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_p)$ is called the "Ordinary Least Squares" estimator (OLS estimator) for β.

OLS estimator

Let X is an n by $p+1$ matrix and y is an n-dimensional vector such that

$$X = \begin{bmatrix} -x_1- \\ -x_2- \\ \vdots \\ -x_n- \end{bmatrix} \quad \text{and} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Note that the first column of X are all ones!

- Minimize $\|y - X\beta\|^2$ is equivalent to

$$X^T X\hat{\beta} = X^T y \Rightarrow \hat{\beta} = (X^T X)^{-1} X^T y$$

- Fitted (predicted) values are $\hat{y} = X\hat{\beta} = X(X^T X)^{-1} X^T y$
- $H = X(X^T X)^{-1} X^T$ are called the hat-matrix

OLS estimator ("stat500" Example)

```
# still consider stat500
data(stat500)

# select "midterm" and "hw" for X
# select "final" for y
X = stat500[,c(1,3)]; y = stat500[,2]

# the first column of X must be all ones!
X = cbind(rep(1, nrow(X)), X)

# X and y must be matrix/vector for computation
X = as.matrix(X); y = as.matrix(y)

# OLS estimator
OLS = solve(t(X)%*%X, t(X)%*%y)
```

OLS estimator ("stat500" Example)

```r
# confirm that two results are the same!
OLS
```

```
##                          [,1]
## rep(1, nrow(X)) 16.81060740
## midterm          0.58178957
## hw              -0.08156661
```

```r
lm(final ~ midterm + hw, data = stat500)$coefficients
```

```
## (Intercept)     midterm          hw
## 16.81060740  0.58178957 -0.08156661
```

Goodness of fit

- It is essential to measure how well the linear regression model fits the data
- R^2 ("R-squared") is one popular measure. Sometimes called the "coefficient of determination" or "percentage of variance explained"
- Total sum of squares:

$$SST = \sum_{i=1}^{n} (y_i - \bar{y})^2, \quad \bar{y} = \sum_{i=1}^{n} y_i / n$$

- Regression sum of squares, or called the explained sum of squares:

$$SSR = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2, \quad \hat{y}_i = x_i' \hat{\beta}$$

- Sum of squares of residuals (related to unexplained variance):

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Goodness of fit

- It holds that $SSR + SSE = SST$. Why?
- R^2 (R-squared):

$$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

- R^2 has ranges from 0 to 1. 0 indicates that $\hat{y}_i = \bar{y}$. On the other hand, 1 indicates $\hat{y}_i = y_i$, i.e. linear regression predictions perfectly fit the data (or perfectly explains the observed variation)
- Larger R^2 indicates a better fit to the data
- For simple linear regression (i.e. $p = 1$), R^2 is equal to r^2 which is a square of the sample correlation between X and Y

Adjusted R^2

- Note that R^2, i.e.,
$$R^2 = 1 - \frac{SSE}{SST}$$
is based on biased estimates of the variances of the dependent variable and of the errors. Why?

- Adjusted R^2 is unbiased estimator:
$$Adjusted\ R^2 = 1 - \frac{SSE/(n-p-1)}{SST/(n-1)} = 1 - (1 - R^2)\frac{n-1}{n-p-1},$$
where $n-1$ and $n-p-1$ represent degree of freedom of the estimate of the variance of the dependent variable and of the estimate of the error variance, respectively

“Galapagos Islands” Example

```r
# use "Galapagos" data
# "gala" is 30 by 7 dimensional data
library("faraway")
data(gala)
head(gala)
```

```
##              Species Endemics  Area Elevation Nearest Sc
## Baltra            58       23 25.09       346     0.6
## Bartolome         31       21  1.24       109     0.6  2
## Caldwell           3        3  0.21       114     2.8  5
## Champion          25        9  0.10        46     1.9  4
## Coamano            2        1  0.05        77     1.9
## Daphne.Major      18       11  0.34       119     8.0
```

"Galapagos Islands" Example (cont.)

Fitted linear regression model is

$$Species = 7.08 - 0.02 Area + 0.32 Elevation - 0.23 Scruz - 0.07 Adjacent$$

```r
# Fit a linear model
fit = lm(Species ~ Area+Elevation+Scruz+Adjacent, gala)
fit
```

```
##
## Call:
## lm(formula = Species ~ Area + Elevation + Scruz + Adjace
##
## Coefficients:
## (Intercept)         Area    Elevation        Scruz
##     7.07538     -0.02398      0.31957     -0.23936
```

"Galapagos Islands" Example (cont.)

```
# compute R-squared
y = gala$Species
R2 = 1 - deviance(fit)/ sum((y-mean(y))^2)
R2
```

```
## [1] 0.7658462
```

```
# compute Adjusted R-squared
n=30; p=4
R2_adjusted = 1 - (1-R2)*(n-1)/(n-p-1)
R2_adjusted
```

```
## [1] 0.7283816
```

```
# "Multiple R-squared" is the R-squared value
summary(fit)
```

```
##
```

"Galapagos Islands" Example (cont.)

```r
# Fit a linear model by adding one more variable
fit2 = lm(Species ~ Area+Elevation+Scruz+Adjacent+
            Nearest, gala)
fit2
```

```
##
## Call:
## lm(formula = Species ~ Area + Elevation + Scruz + Adjace
##     Nearest, data = gala)
##
## Coefficients:
## (Intercept)         Area    Elevation        Scruz      A
##    7.068221    -0.023938     0.319465    -0.240524     -(
##     Nearest
##    0.009144
```

"Galapagos Islands" Example (cont.)

```r
# compute R-squared
y = gala$Species
R2 = 1 - deviance(fit2)/ sum((y-mean(y))^2)
R2
```

```
## [1] 0.7658469
```

```r
# compute Adjusted R-squared
n=30; p=5
R2_adjusted = 1 - (1-R2)*(n-1)/(n-p-1)
R2_adjusted
```

```
## [1] 0.7170651
```

```r
# "Multiple R-squared" is the R-squared value
summary(fit2)
```

```
##
```

Inference of model

- Are any of the p predictors $X_1, \cdots, X_p$ useful when predicting the dependent variable Y?
- Consider the following hypothesis test:

$$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0 \quad \text{versus} \quad H_a : \beta_j \neq 0 \quad \text{for some } j$$

- Corresponding F-statistics is

$$F = \frac{SSR/p}{SSE/(n-p-1)} = \frac{MSR}{MSE},$$

where MSR and MSE are "regression mean square" and "mean square error", respectively

F-statistic (Analysis of Variance)

Table 1: Analysis of variance table

Source of variation	df	Sum of squares	Mean of squares	F-statistic
Regression	p	$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$	$MSR = \frac{SSR}{p}$	$F = \frac{MSR}{MSE}$
Residual	n - p -1	$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$	$MSE = \frac{SSE}{n-p-1}$	
Total	n - 1	$SST = \sum_{i=1}^{n}(y_i - \bar{y})^2$		

- To compute p-value, we refer to $F_{p,n-p-1}$ which represent the F-distribution with degress of freedom $(p, n-p-1)$
- Null hypothesis H_0 is rejected if the F value computed from the data is greater than the critical value. More specifically, given the significance level α such as 0.01, 0.05, 0.1, check whether $F > F^{-1}_{1-\alpha}(p, n-p-1)$ or not, where $F^{-1}_{1-\alpha}(p, n-p-1)$ is the $1-\alpha$ quantile of the $F_{p,n-p-1}$ distribution

F-statistic (Analysis of Variance)

- Or Null hypothesis H_0 is rejected if the p-value computed from the data is less than the significance level α. More specifically, check whether $P(F(p, n-p-1) > F) < \alpha$ or not
- Larger F would means rejection of the null hypothesis H_0
- F value is related to R^2:

$$F = \frac{R^2}{1-R^2}\frac{n-p-1}{p}$$

- What if we get a very small F statistic? We can try nonlinear transformation variables or apply other models: e.g. $x_j \leftarrow \log(x_j + 1)$

"Galapagos Islands" Example

```
fit = lm(Species ~ Area+Elevation+Scruz+Adjacent, gala)
# Check quantities in the "fit"
names(fit)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "ra
##  [5] "fitted.values" "assign"        "qr"            "df
##  [9] "xlevels"       "call"          "terms"         "mo
```

```
# Compute F-Statistic
n=30; p=4
SST = sum((gala$Species - mean(gala$Species))^2)
SSE = deviance(fit)
MSE = SSE/fit$df.residual   #fit$df.residual = n-p-1
SSR = SST - SSE
MSR = SSR/p
Fstat = MSR/MSE
```

“Galapagos Islands” Example (cont.)

```
# Compare with critical value when alpha = 0.05
crit_value = qf(0.95, p, n-p-1)
Fstat > crit_value
```

```
## [1] TRUE
```

```
# Fstat > crit_value! This means we reject H0

# Compute p-value
pvalue = 1-pf(Fstat, p, n-p-1)
pvalue < 0.05
```

```
## [1] TRUE
```

```
# pvalue < significance level!
# This means we reject H0
```

Testing single predictor

- Suppose that the previous hypotheis test indicates the rejection of H_0, i.e., some predictors are useful when predicting Y under the linear regression model
- Now we are interested in whether one particular explanatory variable (say β_j) can be dropped from the linear regression model, i.e. consider

$$H_0 : \beta_j = 0 \quad \text{versus} \quad H_a : \beta_j \neq 0$$

- This can be rewritten as

$$H_0 : M_1 \quad \text{versus} \quad H_a : M_2,$$

where $M_1 = \{x_1, \cdots, x_{j-1}, x_{j+1}, \cdots, x_p\}$ and $M_2 = \{x_1, \cdots, x_p\}$

Comparing two nested models (general version)

- Consider two linear regression models M_1 and M_2 satisfying $M_1 \subset M_2$, and $|M_1| = p_1$ and $|M_2| = p_2$
- Let SSE_1 and SSE_2 be the sum of squares of residuals of the models M_1 and M_2, respectively:
- Then F-Statistic is

$$F = \frac{(SSE_1 - SSE_2)/(p_2 - p_1)}{SSE_2/(n - p_2 - 1)}$$

- Referred distribution is $F_{p_2 - p_1, n - p_2 - 1}$

Comparing two nested models (testing single variable version)

- Now revisit the following "testing single variable" problem:

$$H_0 : \beta_j = 0 \quad \text{versus} \quad H_a : \beta_j \neq 0$$

- $|M_1| := p_1 = p - 1$ and $|M_2| := p_2 = p$
- Recall that SSE_1 and SSE_2 are the sum of squares of residuals of the models M_1 and M_2, respectively:
- Then F-Statistic is

$$F = \frac{(SSE_1 - SSE_2)}{SSE_2/(n-p-1)}$$

- Referred distribution is $F_{1,n-p-1} =^d [t(n-p-1)]^2$, where $t(n-p-1)$ represents Student's t-distribution with a degree of freedom $n-p-1$

“savings” Example

```
# use "savings" data
# savings is an old economic dataset on 50
# different countries (50 by 5 dimensional data)
library("faraway")
data(savings)
head(savings)
```

```
##                sr pop15 pop75     dpi ddpi
## Australia 11.43 29.35  2.87 2329.68 2.87
## Austria   12.07 23.32  4.41 1507.99 3.93
## Belgium   13.17 23.80  4.43 2108.47 3.82
## Bolivia    5.75 41.89  1.67  189.13 0.22
## Brazil    12.88 42.19  0.83  728.47 4.56
## Canada     8.79 31.72  2.85 2982.88 2.43
```

```
# We can specify column/row names using
# "colnames" and "rownames"
```

“savings” Example (cont.)

savings	*Savings rates in 50 countries*

Description

The `savings` data frame has 50 rows and 5 columns. The data is averaged over the period 1960-1970.

Usage

```
data(savings)
```

Format

This data frame contains the following columns:

`sr` savings rate - personal saving divided by disposable income

`pop15` percent population under age of 15

`pop75` percent population over age of 75

`dpi` per-capita disposable income in dollars

`ddpi` percent growth rate of dpi

Details

Now also appears as `LifeCycleSavings` in the `datasets` package

Source

Belsley, D., Kuh. E. and Welsch, R. (1980) "Regression Diagnostics" Wiley.

"savings" Example (cont.)

Fitted linear regression model is
$sr = 28.57 - 0.46pop15 - 1.69pop75 - 0.0003dpi + 0.41ddpi$

```
# apply linear regression model
fit2 = lm(sr ~ ., data = savings)

# Compute F-Statistic
n=nrow(savings); p=ncol(savings)-1
SST2 = sum((savings$sr - mean(savings$sr))^2)
SSE2 = deviance(fit2)
MSE2 = SSE2/fit2$df.residual    #fit$df.residual = n-p-1
SSR2 = SST2 - SSE2
MSR2 = SSR2/p
Fstat2 = MSR2/MSE2
```

“savings” Example (cont.)

Is pop75 significant in the full model?

```
fit2 = lm(sr ~ ., data = savings)
fit1 = lm(sr ~ pop15 + dpi + ddpi, savings)
SSE1 = deviance(fit1)
Fstat = (SSE1-SSE2)/(SSE2/(n-p-1))
1-pf(Fstat,1,n-p-1)    # this is the p-value
```

```
## [1] 0.1255298
```

```
# compute p-value using t-distribution
2*pt(-1.561,n-p-1)
```

```
## [1] 0.1255297
```

“savings” Example (cont.)

We can perform the hypothesis testing using “anova” function

```r
# compare two tested model
anova(fit1, fit2)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ pop15 + dpi + ddpi
## Model 2: sr ~ pop15 + pop75 + dpi + ddpi
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     46 685.95
## 2     45 650.71  1    35.236 2.4367 0.1255
```

Questions

1. Perform the following hypothesis test:

$$H_0 : \text{both dpi and ddpi are not significant}$$
$$\text{versus} \quad H_a : \text{All explanatory variables are significant}$$

2. Perform the following hypothesis test:

$$H_0 : \text{both dpi and ddpi are not significant}$$
$$\text{versus} \quad H_a : \text{pop15, pop75, ddpi are all significant}$$

Questions (cont.)

```r
# [1.] compare two tested model
fit2 = lm(sr ~ ., data = savings)
fit1 = lm(sr ~ pop15 + pop75, savings)
anova(fit1, fit2)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ pop15 + pop75
## Model 2: sr ~ pop15 + pop75 + dpi + ddpi
##   Res.Df    RSS Df Sum of Sq     F  Pr(>F)
## 1     47 726.17
## 2     45 650.71  2    75.455 2.609 0.08471 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
```

Questions (cont.)

```r
# [2.] compare two tested model
fit2 = lm(sr ~ pop15 + pop75 + ddpi, data = savings)
fit1 = lm(sr ~ pop15 + pop75, savings)
anova(fit1, fit2)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ pop15 + pop75
## Model 2: sr ~ pop15 + pop75 + ddpi
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1     47 726.17
## 2     46 652.61  1    73.562 5.1851 0.02748 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.
```

Testing a subspace

One can consider the following hypothesis test:

$$H_0 : \beta_{pop15} = \beta_{pop75} \quad \text{versus} \quad H_a : \beta_{pop15} \neq \beta_{pop75}$$

```
fit1 = lm(sr ~ I(pop15 + pop75)  + dpi + ddpi, savings)
fit2 = lm(sr ~ pop15 + pop75 + dpi + ddpi, data = savings)
anova(fit1,fit2)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ I(pop15 + pop75) + dpi + ddpi
## Model 2: sr ~ pop15 + pop75 + dpi + ddpi
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     46 673.63
## 2     45 650.71  1    22.915 1.5847 0.2146
```

Testing a subspace

One can consider the following hypothesis test:

$$H_0 : \beta_{pop75} = 4\beta_{pop15} \quad \text{versus} \quad H_a : \beta_{pop15} \neq 2\beta_{pop75}$$

```
fit1 = lm(sr ~ I(1*pop15 + 4*pop75)  + dpi + ddpi, savings)
anova(fit1,fit2)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ I(1 * pop15 + 4 * pop75) + dpi + ddpi
## Model 2: sr ~ pop15 + pop75 + dpi + ddpi
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     46 651.33
## 2     45 650.71  1   0.61849 0.0428 0.8371
```

Testing a subspace

One can consider the following hypothesis test:

$$H_0 : \beta_{ddpi} = 0.5 \quad \text{versus} \quad H_a : \beta_{ddpi} \neq 0.5$$

```
fit1 = lm(sr ~ pop15+pop75+dpi+offset(0.5*ddpi), savings)
anova(fit1,fit2)
```

```
## Analysis of Variance Table
##
## Model 1: sr ~ pop15 + pop75 + dpi + offset(0.5 * ddpi)
## Model 2: sr ~ pop15 + pop75 + dpi + ddpi
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     46 653.78
## 2     45 650.71  1    3.0635 0.2119 0.6475
```

Questions

Q1 . Perform the following hypothesis test:

$$H_0 : \beta_{pop75} = 4\beta_{pop15} \text{ and } ddpi = 0.5 \quad \text{versus} \quad H_a : \text{full model}$$

Caution when using multiple constraints in the "lm" function

Consider the following hypothesis test:

$$H_0 : \beta_{pop75} = 4\beta_{pop15} \text{ and } \beta_{dpi} = \beta_{pop15} \quad \text{versus} \quad H_a : \text{full model}$$

```r
# which linear model is correct between the following two?
fit1 = lm(sr~I(pop15+4*pop75)+I(dpi+pop15)+ddpi,savings)
fit11 = lm(sr~I(pop15+4*pop75+dpi)+ddpi,savings)
```

Caution when using multiple constraints (cont.)

The model "fit1" is equivalent to the following linear model:

$$\begin{aligned} y &= (X_{pop15} + 4X_{pop75})\beta_1 + (X_{pop15} + X_{dpi})\beta_2 + X_{ddpi}\beta_3 + \epsilon \\ &= X_{pop15}(\beta_1 + \beta_2) + X_{pop75}(4\beta_1) + X_{dpi}\beta_2 + X_{ddpi}\beta_3, \end{aligned}$$

which implies

$$\beta_{pop15} = \beta_1 + \beta_2,\ \beta_{pop75} = 4\beta_1,\ \beta_{dpi} = \beta_2,\ \beta_{ddpi} = \beta_3.$$

This can be rewritten as the following compact form:

$$\beta_{pop15} = \beta_{pop75}/4 + \beta_{dpi},$$

which is not eqivalent to the null hypothesis
$H_0 : \beta_{pop75} = 4\beta_{pop15} = 4\beta_{dpi}$

Penalization methods (Shrinkage methods)

- Recall that linear regression is based on minimizing residual sum of squares:

$$\text{minimize}_{\beta} \sum_{i=1}^{n} (y_i - x_i'\beta)^2$$

- The obtained minimizer $\hat{\beta}$ (OLS estimator) is generally a good estimator of β.
- However, (1). when the number of explanatory variables (p) is much larger than sample size (n); (2). when columns of the design matrix X are highly correlated, obtained $\hat{\beta}$ can be unstable and often less interpretable

Penalization methods (Shrinkage methods)

- Shrnkage methods give a penalty on the coefficient (β) in the optimization problem such that the obtained coefficient ($\hat{\beta}$) can't be too large!
- Shrnkage methods generally solve

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + Penalty(\beta),$$

where $Penalty(\cdot)$ is a function that penalizes β

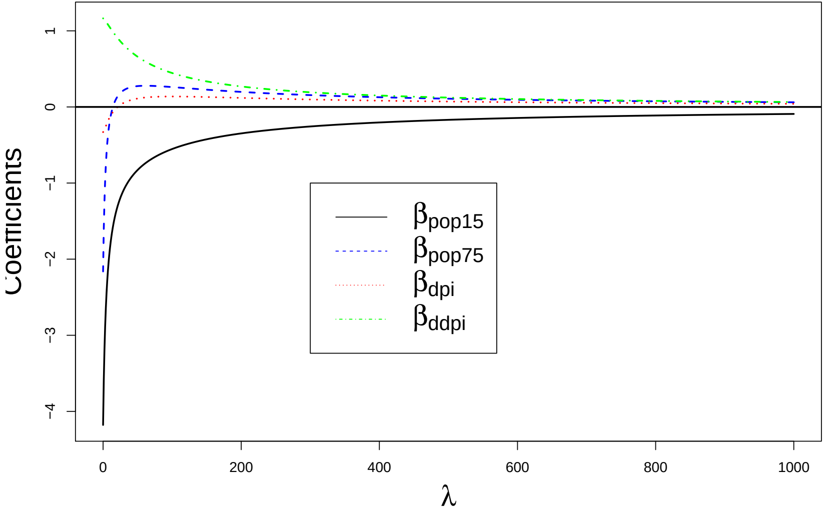
- In this class, we will learn "Ridge penalty" and "Lasso penalty"

Ridge regression

- Ridge regression is based on limiting $\sum_{j=1}^{p} \beta_j^2$
- Suppose that $X \in \mathbb{R}^{n \times p}$ is columnwise centered. Ridge regression solves
$$\text{minimize}_\beta \sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2,$$
where $\lambda > 0$ is a user-determined tuning parameter that controls the tradeoff between fit and penalty
- Ridge regression has a closed form solution:
$$\hat{\beta}_{ridge}(\lambda) = (X'X + \lambda I_{p\times p})^{-1}X'y,$$
where $I_{p\times p}$ is a p by p identity matrix

```
library("MASS")
# ridge regression
lm.ridge(sr ~ ., data= savings, lambda = 1)
```

Ridge regression



Apendix

```
# Ridge regression
lam_set = seq(0, 1000, 1)
result = lm.ridge(sr ~ .,data=savings,lambda=lam_set)
plot(lam_set, result$coef["pop15",],type = "l",
  xlim=range(lam_set),ylim=range(result$coef),lwd=2,
 xlab=expression(lambda),ylab="Coefficients",cex.lab=2)
lines(lam_set,result$coef["pop75",],col="blue",lty=2,lwd=2
lines(lam_set,result$coef["dpi",],col="red",lty=3,lwd=2)
lines(lam_set,result$coef["ddpi",],col="green",lty=4,lwd=2
abline(h = 0, lwd = 2)

# Add legend
legend(300,-1,legend=expression(beta[pop15],beta[pop75],
  beta[dpi],beta[ddpi]),
col=c("black", "blue", "red", "green"), lty=1:4, cex=1)
```

Ridge regression with orthonormal design matrix

- In the case of an orthonormal design matrix $X \in \mathbb{R}^{n \times p}$, i.e, $X'X = I_{p \times p}$,

$$\hat{\beta}^{OLS} = X'y, \quad \hat{\beta}^{ridge} = X'y/(1+\lambda),$$

  which clearly illustrates the shrinkage effect of Ridge regression

- Ridge regression produce the effect of shrinking the estimates of β toward zero that cause a bias but reduce a variance of the estimator. Think about $MSE(\hat{\beta}) = Bias(\hat{\beta})^2 + Variance(\hat{\beta})$!

Lasso regression

- Lasso regression is based on limiting $\sum_{j=1}^{p} |\beta_j|$
- Lasso regression solves

$$\text{minimize}_{\beta} \ \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|,$$

  where $\lambda > 0$ is a user-determined tuning parameter that controls the tradeoff between fit and penalty

- Lasso regression doesn't have a closed form solution
- Compared to Ridge regression, Lasso regression provides a more sparse solution

Lasso regression with orthonormal design matrix

- In the case of an orthonormal design matrix $X \in \mathbb{R}^{n \times p}$, i.e, $X'X = I_{p \times p}$,

$$\begin{aligned} \hat{\beta}_j^{Lasso} &= \hat{\beta}_j^{OLS} - \lambda \quad \text{if } \hat{\beta}_j^{OLS} > \lambda \\ &= 0 \quad \text{if } \lambda \leq \hat{\beta}_j^{OLS} \leq \lambda \\ &= \hat{\beta}_j^{OLS} + \lambda \quad \text{if } \hat{\beta}_j^{OLS} < -\lambda \end{aligned}$$

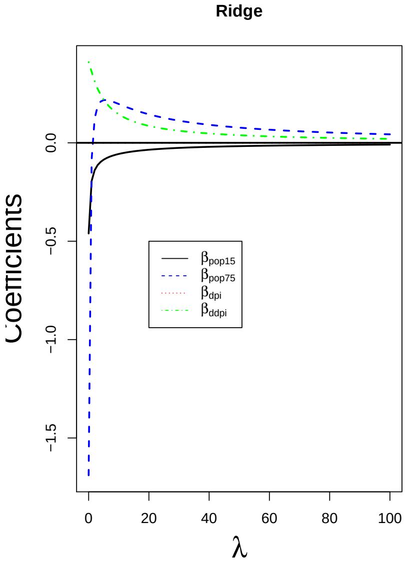
  which clearly illustrates the shrinkage effect of Lasso regression

- Lasso regression has the effect of making the estimates of some of β_js exactly zero that cause a bias but reduce a variance of the estimator.

Lasso and Ridge regression

```r
library(glmnet)
# Lasso regression
X = as.matrix(savings[,2:5])
y = as.matrix(savings[,1])
lam_set1 = seq(0, 100, 1)
lam_set2 = seq(0, 10, 0.1)

# Lasso and Ridge regression
ridge=glmnet(X,y,family="gaussian",alpha=0,lambda=lam_set1
lasso=glmnet(X,y,family="gaussian",alpha=1,lambda=lam_set2
```

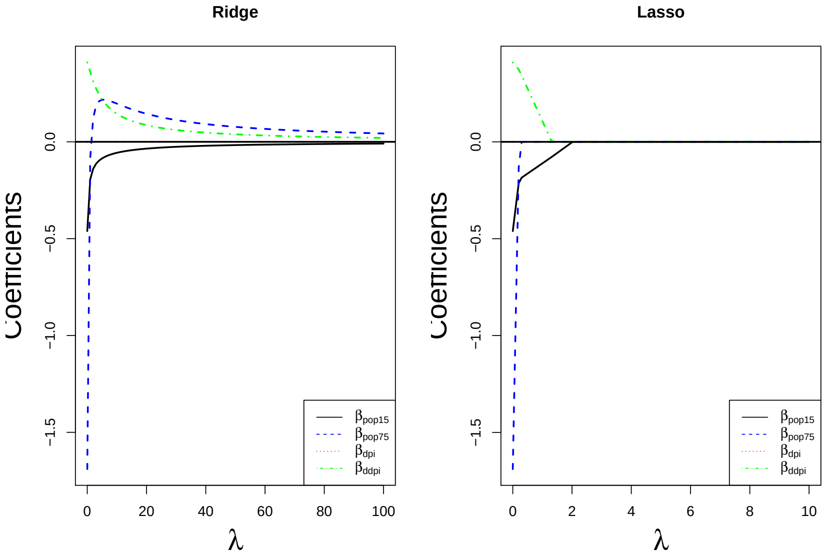
Ridge regression (not efficient)



Appendix (not efficient)

```r
par(mfrow = c(1,2))
# ridge regression
plot(lam_set1, rev(ridge$beta[1,]),type = "l",
  xlim=range(lam_set1),ylim=range(ridge$beta),lwd=2,
main = "Ridge",
xlab=expression(lambda),ylab="Coefficients",cex.lab=2)
lines(lam_set1,rev(ridge$beta[2,]),col="blue",lty=2,lwd=2)
lines(lam_set1,rev(ridge$beta[3,]),col="red",lty=3,lwd=2)
lines(lam_set1,rev(ridge$beta[4,]),col="green",lty=4,lwd=2)
abline(h = 0, lwd = 2)
# Add legend
legend(20,-0.5,legend=expression(beta[pop15],beta[pop75],
  beta[dpi],beta[ddpi]),
col=c("black", "blue", "red", "green"), lty=1:4, cex=1)
```

Ridge/Lasso regression (efficient)



Ridge/Lasso regression (efficient)

```
par(mfrow = c(1,2))
names = c("Ridge", "Lasso");result = list(ridge, lasso)
lam = list(lam_set1,lam_set2)

for (i in 1:2){
plot(lam[[i]], rev(result[[i]]$beta[1,]),type = "l",
  xlim=range(lamd[[i]]),ylim=range(result[[i]]$beta),
main = names[i],
xlab=expression(lambda),ylab="Coefficients",cex.lab=2)
lines(lam[[i]],rev(result[[i]]$beta[2,]),col="blue",lty=2)
lines(lam[[i]],rev(result[[i]]$beta[3,]),col="red",lty=3)
lines(lam[[i]],rev(result[[i]]$beta[4,]),col="green",lty=4)
abline(h = 0, lwd = 2)
# Add legend
legend("bottomright",legend=expression(beta[pop15],
    beta[pop75], beta[dpi],beta[ddpi]),
col=c("black", "blue", "red", "green"), lty=1:4, cex=1)}
```

Sparsity assumption on the coefficient when p>n

- When $p > n$, i.e. high-dimensional case, β is not uniquely defined, which cause an identifiability issue. Why?
- With a sparsity condition $\|\beta\|_0 \leq s$ for some $s < n$, we could estimate β
- "Sparsity assumption" is essential in the high-dimensional model

Comparisons of regression methods via simulation models

```
# Generate X and y using normal distribution
set.seed(2000)
n = 100; p = 50
X = matrix(rnorm(n*p), ncol = p)
X = cbind(rep(1,n), X)

# column normalizing
norm_vec = sqrt(apply(X^2, 2, mean))
X = X / matrix(rep(norm_vec, each = n), nrow = n)

# Check the norm of columns
#sqrt(apply(X^2, 2, mean))

# Generate a dependent variables y
beta = runif(p+1)
#beta[6:p+1] = 0
y = X %*% beta + 0.1*rnorm(n)
```

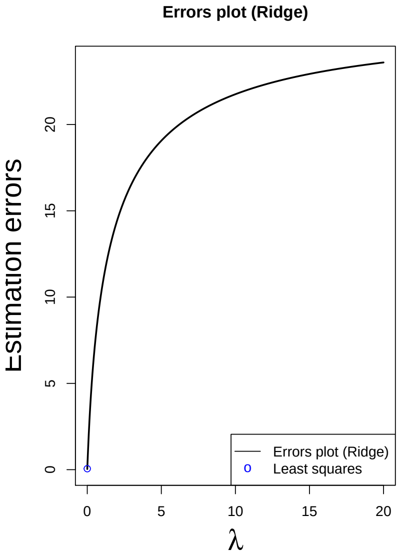
Comparisons of regression methods via simulation models

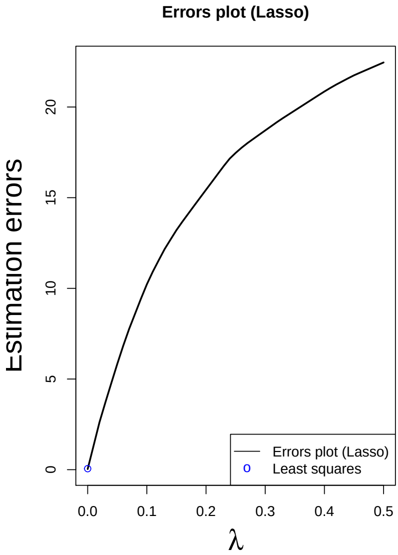
```
# Apply Least squares
ls_beta = lm(y ~ X[,2:(p+1)])$coefficients

# Apply Ridge regression
lam_Ridge = seq(0, 20, 0.05)
ridge=glmnet(X[,2:(p+1)],y,family="gaussian", alpha=0,
             lambda=lam_Ridge)
ridge_beta = rbind(ridge$a0,ridge$beta)

# Apply Lasso regression
lam_Lasso = seq(0, 0.5, 0.01)

lasso=glmnet(X[,2:(p+1)],y,family="gaussian", alpha=1,
             lambda=lam_Lasso)
lasso_beta = rbind(lasso$a0,lasso$beta)
```

Comparisons of regression methods via simulation models

```
# Analyzing estimation errors
err_ls = sqrt(sum(ls_beta - beta)^2)

err_ridge = NULL
for (i in 1:length(lam_Ridge)){
  err_ridge=c(err_ridge,sqrt(sum(ridge_beta[,i]-beta)^2))}

err_ridge = rev(err_ridge)

err_lasso = NULL
for (i in 1:length(lam_Lasso)){
  err_lasso=c(err_lasso,sqrt(sum(lasso_beta[,i]-beta)^2))}

err_lasso = rev(err_lasso)
```

Comparisons of regression methods via simulation models





Appendix

```
# Drawing error plots

par(mfrow = c(1,2))
names = c("Errors plot (Ridge)", "Errors plot (Lasso)");
result = list(err_ridge, err_lasso);
lam = list(lam_Ridge,lam_Lasso)
for (i in 1:2){
# ridge regression
plot(lam[[i]], result[[i]],type = "l",
xlim=range(lam[[i]]), ylim=range(result[[i]]),lwd=2,
main = names[i],
xlab=expression(lambda),ylab="Estimation errors")
points(0,err_ls, col = "blue")

# Add legend
legend("bottomright",legend=c(names[[i]],"Least squares"),
col=c("black","blue"),lty=c(1,0),pch = c("","o"),cex=1)}
```

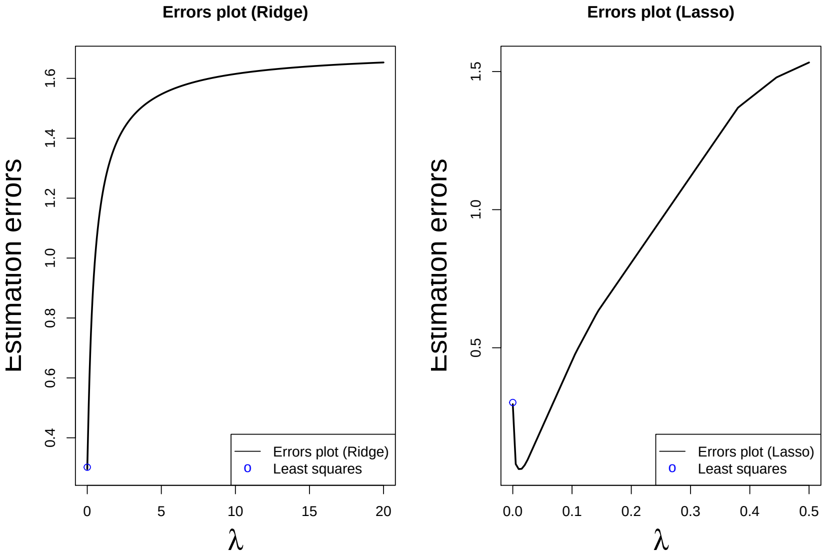
Make a function

```
# load the uploaded "generating_plot" function instead

generating_plot = function(X, y, beta, lam_Ridge, lam_Lasso
# Apply Least squares
ls_beta = lm(y ~ X[,2:(p+1)])$coefficients

# Apply Ridge regression
ridge=glmnet(X[,2:(p+1)],y,family="gaussian", alpha=0, lamb
ridge_beta = rbind(ridge$a0,ridge$beta)

# Apply Lasso regression

lasso=glmnet(X[,2:(p+1)],y,family="gaussian", alpha=1, lamb
lasso_beta = rbind(lasso$a0,lasso$beta)

# Analyzing estimation errors
err_ls = sqrt(sum(ls_beta - beta)^2)
```

Comparisons of regression methods (Sparse model case)

```
# Generate a dependent variables y with a sparse beta!
beta[6:p+1] = 0
y = X %*% beta + 0.1*rnorm(n)

lam_Ridge = seq(0, 20, 0.05)
lam_Lasso = seq(0, 0.5, 0.005)

results = generating_plot(X,y,beta,lam_Ridge,lam_Lasso)

# We can observe that Lasso gives more accurate solutions
# with some penalty parameter lambda when underlying beta
# is sparse!

# results$lasso_beta[,3] gives a sparse solution and
# provides the most accurate solution!
```

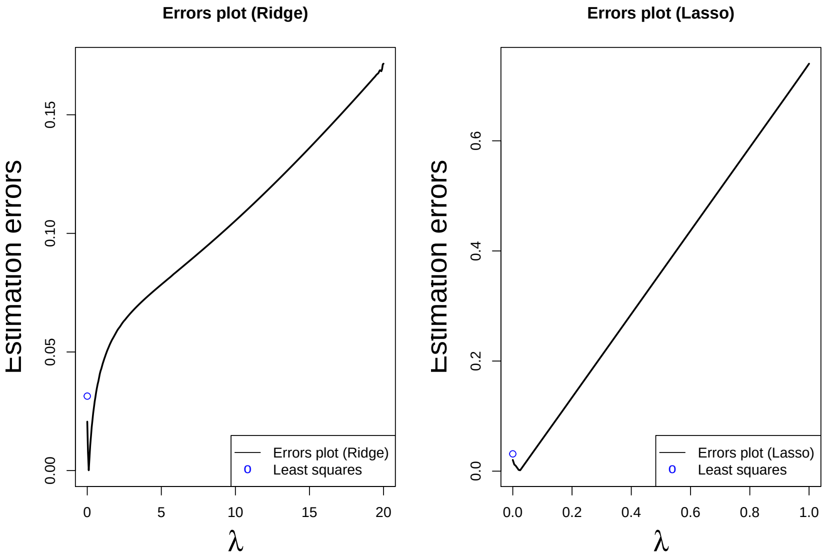
Comparisons of regression methods (Sparse model case)



Comparisons of regression methods (highly correlated matrix X)

```
## Now we consider correlated design matrix case
set.seed(3000)
n = 100; p = 70
# Generating covariance (correlation) matrix
sigma = 0.99
A = array(0,c(p,p))
for (i in 1:p){for (j in 1:p){A[i,j] = sigma^(abs(i-j))}}
Z = matrix(rnorm(n*p), ncol = p)
# The generating X has independent rows but dependent
# columns whose population covariance matrix is A
# library("expm")
X = Z %*% sqrtm(A); X = cbind(rep(1,n), X)
beta = c(rep(1,5), rep(0, p-4))
y = X %*% beta + 0.1*rnorm(n)
lam_Ridge = seq(0, 20, 0.05); lam_Lasso = seq(0, 1, 0.005)
results = generating_plot(X, y, beta, lam_Ridge, lam_Lasso)
```

Comparisons of regression methods (highly correlated matrix X)



Model selection criterion

- Among many obtained linear models (by using different λ values), we could choose the best one based on some criterion
- "R-squared" and "Adjusted R-squared" are one of such criteria, but not often used in the high-dimensional model
- More popular criterion are "Akaike information criterion" (AIC) and "Bayesian information criterion" (BIC)

AIC and BIC

- AIC/BIC consider trade-off between goodness of fit and simplicity of the model.
- AIC/BIC only provide a relative quality of the model, i.e. they do not provide a statistical inference (i.e. test) of a model
- Lower AIC/BIC indicates a better model!
- For the model M, let L be the maximum value of the log-likelihood function for the model M. Then

$$AIC(M) = -2\log L + 2|M| = n\log\left(\sum_{i=1}^{n}\frac{(y_i - x_i'\hat{\beta})^2}{n}\right) + 2|M|$$

$$BIC(M) = -2\log L + |M|\log n = n\log\left(\frac{\sum_{i=1}^{n}(y_i - x_i'\hat{\beta})^2}{n}\right) + |M|\log n$$

- BIC penalizes larger models more aggressively, i.e. BIC prefers smaller models compared to AIC

AIC and BIC (cont.)

- The likelihood function is

$$L = (2\pi\sigma^2)^{-n/2} \exp\left(\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i'\beta)^2\right)$$

- The log-likelihood function is

$$\log L = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i'\beta)^2$$

- Since $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i'\beta)^2$, the maximum value of the log-likelihood function is

$$-\frac{n}{2}\log(\hat{\sigma}^2) + \text{constant} = -\frac{n}{2}\log\left(\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i'\hat{\beta})^2\right) + \text{constant},$$

which gives a AIC/BIC formula

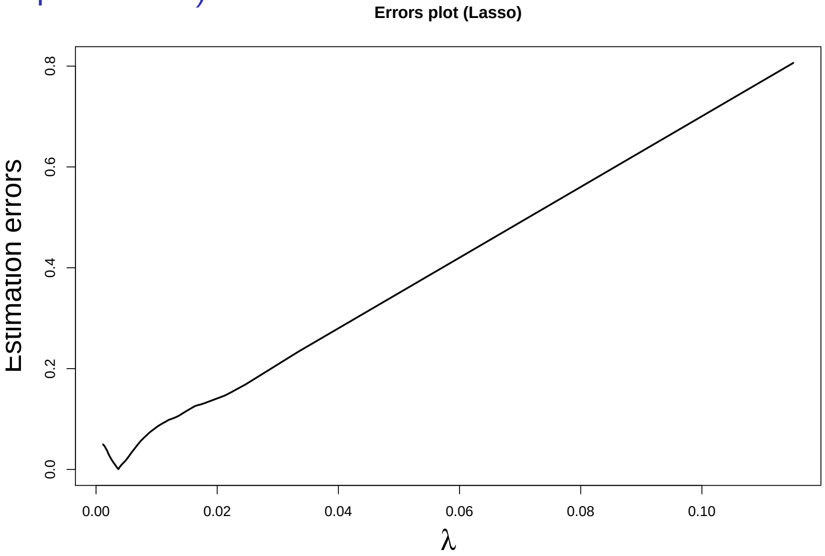
Selecting model using AIC/BIC via simulation model (p>n and sparse case)

```
## Apply AIC/BIC to the simulation model

# Generate X and y using normal distribution
set.seed(1000)
n = 100; p = 200
X = matrix(rnorm(n*p), ncol = p)
X = cbind(rep(1,n), X)

# column normalizing
norm_vec = sqrt(apply(X^2, 2, mean))
X = X / matrix(rep(norm_vec, each = n), nrow = n)

# Generate a dependent variables y
beta = runif(p+1)
beta[8:p+1] = 0
y = X %*% beta + 0.1*rnorm(n)
```

Selecting model using AIC/BIC via simulation model (p>n and sparse case)

```
# Apply Lasso regression
lam_Lasso = seq(0.05, 5, 0.01)*sqrt(log(p)/n)*0.1

lasso=glmnet(X[,2:(p+1)],y,family="gaussian", alpha=1,
             lambda=lam_Lasso)
lasso_beta = rbind(lasso$a0,lasso$beta)

# Analyzing estimation errors
err_lasso = NULL
for (i in 1:length(lam_Lasso)){
err_lasso=c(err_lasso,sqrt(sum(lasso_beta[,i]-beta)^2))}

err_lasso = rev(err_lasso)
```

Selecting model using AIC/BIC via simulation model (p>n and sparse case)

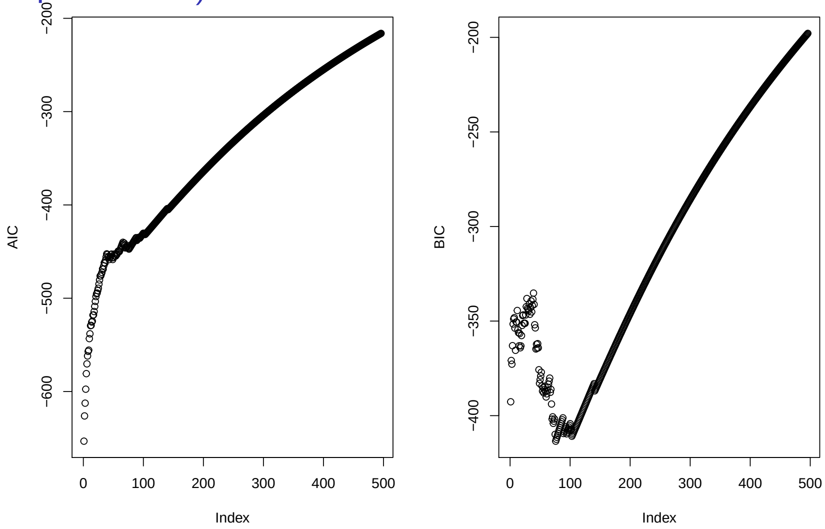



```
##                   V1       V2       V3       V4       V5       V6       V7
```

Selecting model using AIC/BIC via simulation model (p>n and sparse case)

```r
# Drawing error plots
names = "Errors plot (Lasso)" ; result = err_lasso;
lam = lam_Lasso

plot(lam, result,type = "l",
xlim=range(lam), ylim=range(result),lwd=2,
main = names,
xlab=expression(lambda),ylab="Estimation errors")

ind = which(err_lasso==min(err_lasso))
round(lasso_beta[,ncol(lasso_beta)-ind+1],3)
```

Selecting model using AIC/BIC via simulation model (p>n and sparse case)

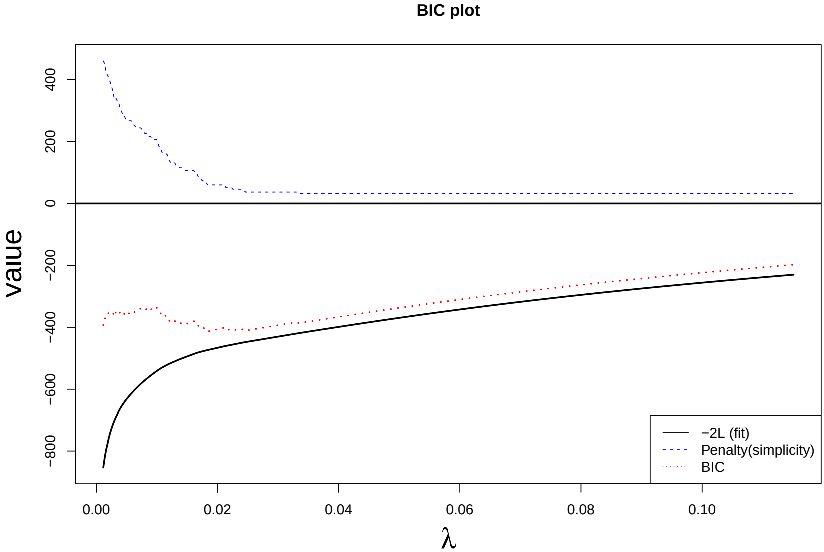



```
##                    V1         V2         V3         V4         V5
##   0.60696    0.96263    0.58212    0.63603    0.21744    0.68105    0
```

Appendix

```r
# Computing AIC/BIC
AIC = NULL; BIC = NULL; BIC_fit = NULL; BIC_pen = NULL
for (i in 1:length(lam_Lasso)){
AIC=c(AIC, n*log(sum((y - X%*% lasso_beta[,i])^2)/n)+
        2*sum(abs(lasso_beta[2:(p+1),i])>0.00001))
BIC=c(BIC, n*log(sum((y - X%*% lasso_beta[,i])^2)/n)+
          log(n)*sum(abs(lasso_beta[2:(p+1),i])>0.00001))
BIC_fit=c(BIC_fit,n*log(sum((y-X%*%lasso_beta[,i])^2)/n))
BIC_pen=c(BIC_pen, log(n)*
            sum(abs(lasso_beta[2:(p+1),i])>0.00001))}
AIC = rev(AIC); BIC = rev(BIC)
BIC_fit = rev(BIC_fit); BIC_pen = rev(BIC_pen)
par(mfrow = c(1,2))
plot(AIC); plot(BIC)
ind_B = which(BIC==min(BIC))
round(lasso_beta[,ncol(lasso_beta)-ind_B+1],5)
```

Selecting model using AIC/BIC via simulation model (p>n and sparse case)



Appendix

```r
# Drawing BIC plot
names = "BIC plot" ;
lam = lam_Lasso

par(mfrow=c(1,1))
plot(lam, BIC_fit,type = "l",
xlim=range(lam),lwd=2,main=names,ylim=
  c(min(BIC_fit), max(BIC_pen)),
xlab=expression(lambda),ylab="Value",cex.lab=2)

lines(lam,BIC_pen, col = "blue", lty = 2)
lines(lam,BIC,col="red",lty=3, lwd=2)
# Add legend
legend("bottomright",legend=c("-2L (fit)",
                  "Penalty(simplicity)", "BIC"),
col=c("black", "blue", "red"), lty=1:3, cex=1)
```