HW5

賴冠維

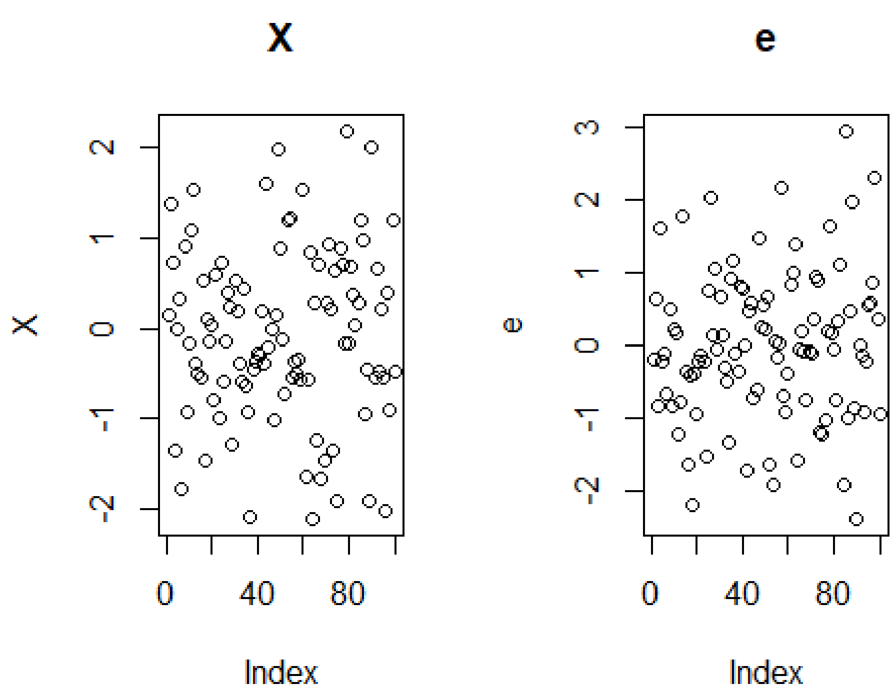
2020/12/4

```
## -- Attaching packages --------------------------------------- tidyve
rse 1.3.0 --

## √ ggplot2 3.3.2     √ purrr   0.3.4
## √ tibble  3.0.4     √ dplyr   1.0.2
## √ tidyr   1.1.2     √ stringr 1.4.0
## √ readr   1.3.1     √ forcats 0.5.0

## Warning: package 'tibble' was built under R version 4.0.3

## -- Conflicts ------------------------------------------ tidyverse_co
nflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

(8)

(a)

設定 set.seed(12345),以 rnom()取出 x,e 各 100 個 observations



(b)

製造$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon$
設定β_0 到 β_3為(1,10,5,20)

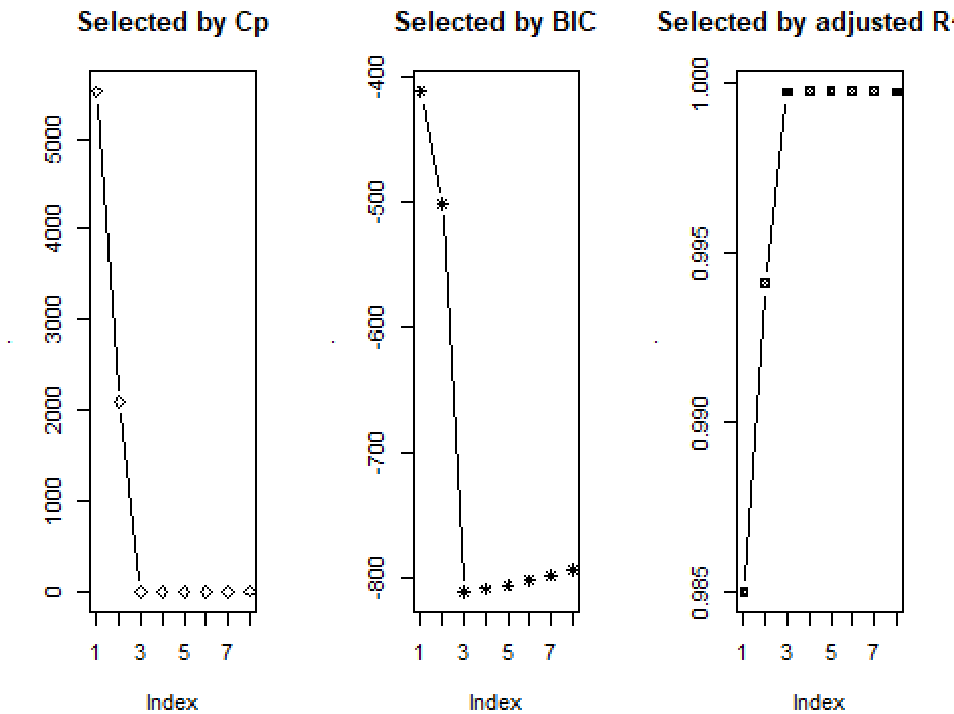
(c)

首先使用默認方法，即為 Exhaustive Search (窮舉所有方法)，所得如下：
可以看到默認為到擷取 8 個變數，並列出每個變數下表現最好的變數組合。

```
## Subset selection object
## 10 Variables  (and intercept)
##     Forced in Forced out
## V1      FALSE      FALSE
## V2      FALSE      FALSE
## V3      FALSE      FALSE
## V4      FALSE      FALSE
## V5      FALSE      FALSE
## V6      FALSE      FALSE
## V7      FALSE      FALSE
## V8      FALSE      FALSE
## V9      FALSE      FALSE
## V10     FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          V1  V2  V3  V4  V5  V6  V7  V8  V9  V10
## 1  ( 1 ) " " " " "*" " " " " " " " " " " " " " "
## 2  ( 1 ) " " "*" "*" " " " " " " " " " " " " " "
## 3  ( 1 ) "*" "*" "*" " " " " " " " " " " " " " "
## 4  ( 1 ) "*" "*" "*" "*" " " " " " " " " " " " "
## 5  ( 1 ) "*" "*" "*" " " " " "*" " " " " " " "*"
## 6  ( 1 ) "*" "*" "*" " " "*" " " " " "*" " " "*"
## 7  ( 1 ) "*" "*" "*" "*" "*" "*" " " "*" " " " "
## 8  ( 1 ) "*" "*" "*" "*" "*" "*" " " "*" " " "*"
```

接著以Cp、BIC、$Adj\ R\ square$三種不同標準來挑選變數，
以 Cp(複雜度)最低、BIC 值最小、Adj R^2 最大為標準進行變數篩選，
可以看到這三種方法(Cp、BIC、$Adj\ R\ square$)分別挑選了 4、3、5 個變數，
並且變數組成也不盡相同，代表不同方法所在意的地方都各有差異。

```
## [1] "Cp Select: 4 Variables"

## (Intercept)          V1          V2          V3          V4
##   0.9478419   9.5931787   5.2919897  20.1234912  -0.1000258

## [1] "BIC Select: 3 Variables"

## (Intercept)          V1          V2          V3
##    1.072448    9.610514    4.907571   20.117291
```



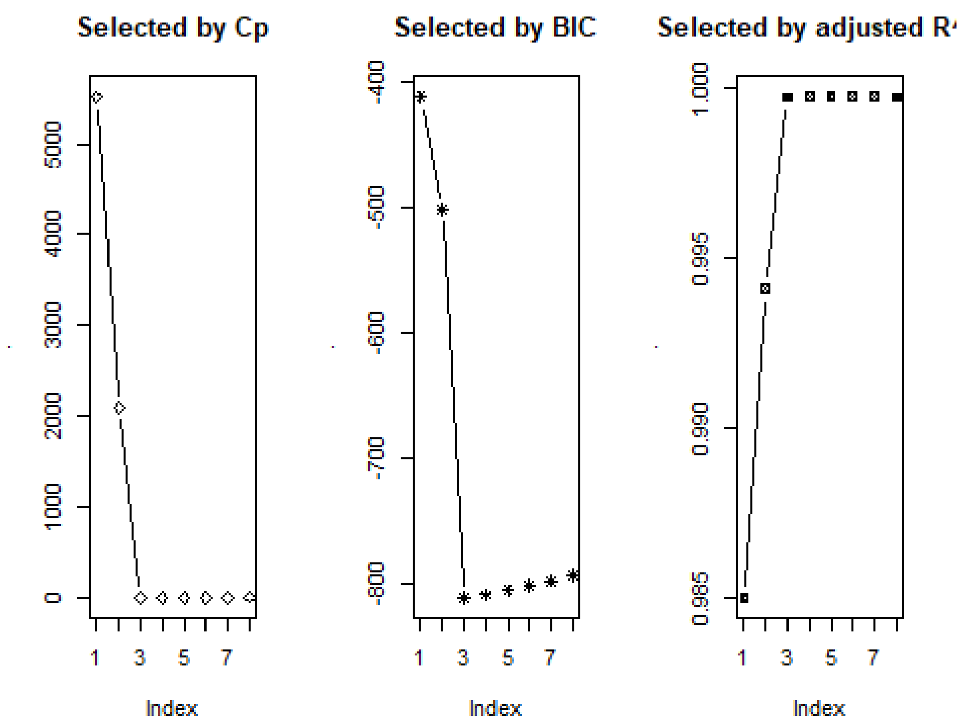

```
## [1] "Adj R^2 Select: 5 Variables"

##   (Intercept)            V1            V2            V3            V6
     V10
##   0.905780200   9.615345281   5.407379418  20.099445433  -0.077575224   0.
002591922
```

(d)

首先使用 Forward stepwise Selection，Forward Stepwise 的作法：
* 在一個空的迴歸中逐一添加變數，直到任何一個變數的額外貢獻度(AIC、BIC、Cp 等)無統計意義就停止。

可以看到這三種方法(Cp、BIC、$Adj\ R\ square$)分別挑選了 4、3、5 個變數，與前述結果相同。

```
## [1] "Cp Select: 4 Variables"

## (Intercept)          V1          V2          V3          V4
##   0.9478419   9.5931787   5.2919897  20.1234912  -0.1000258

## [1] "BIC Select: 3 Variables"

## (Intercept)          V1          V2          V3
##    1.072448    9.610514    4.907571   20.117291
```

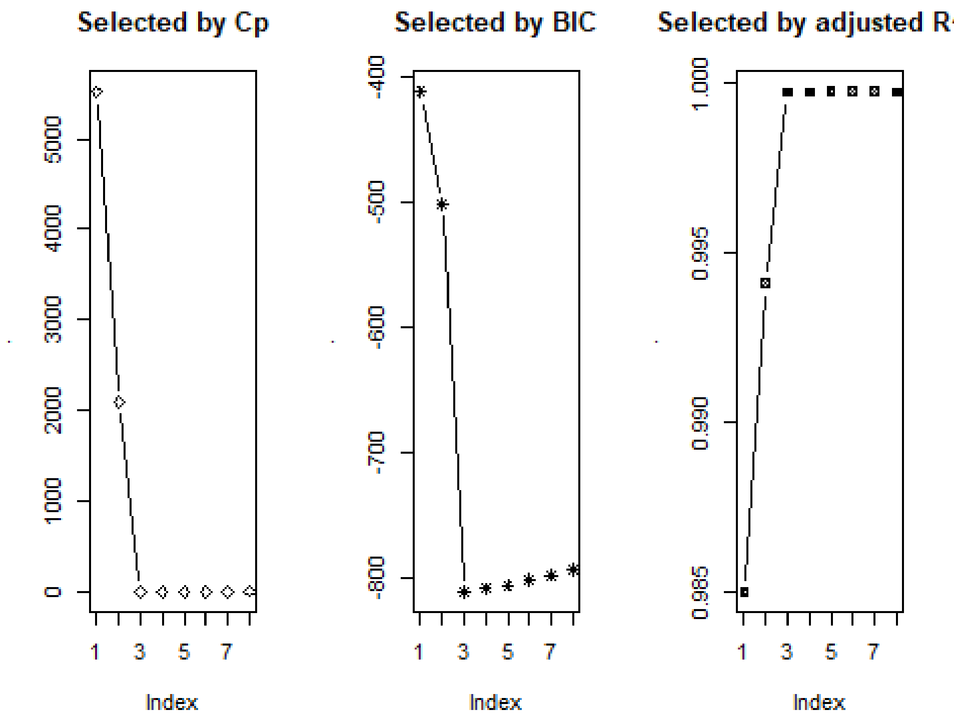



```
## [1] "Adj R^2: 5 Variables"

##   (Intercept)            V1            V2            V3            V6
      V10
##  0.905780200  9.615345281  5.407379418 20.099445433 -0.077575224  0.
002591922
```

接下來採用 Backwards stepwise Selection，Backward Stepwise：
* 在一個完整的迴歸中，逐一移除變數，直到移除任何一個變數時，模型都會損失過多的解釋力，那就停止。

可以看到這三種方法(Cp、BIC、$Adj\ R\ square$)分別挑選了 3、3、5 個變數，
僅 Cp 挑選結果改變，其餘相同。

```
## [1] "Cp Select: 3 Variables"

## (Intercept)          V1          V2          V3
##    1.072448    9.610514    4.907571   20.117291

## [1] "BIC Select: 3 Variables"

## (Intercept)          V1          V2          V3
##    1.072448    9.610514    4.907571   20.117291
```

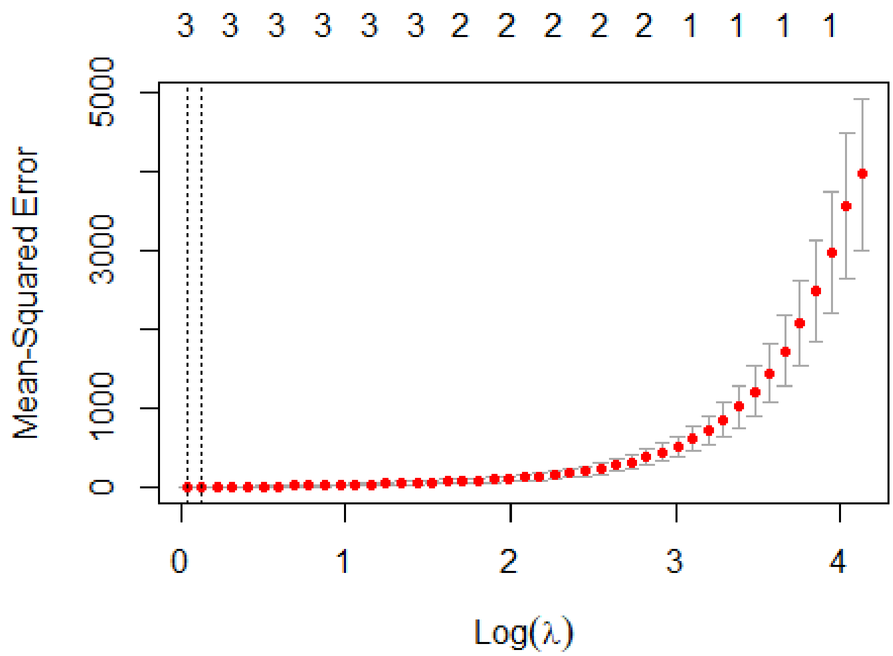



```
## [1] "Adj R^2: 5 Variables"

##   (Intercept)            V1            V2            V3            V6 
     V10 
##  0.905780200  9.615345281  5.407379418 20.099445433 -0.077575224  0.
002591922
```

(e)

使用 Lasso Regression，並且使用 Cross Validation 來挑選最佳的λ，可由下圖所見：不論是λ_{min}或是λ_{lse}皆選取 3 個變數。

```
## Loading required package: glmnet

## Warning: package 'glmnet' was built under R version 4.0.3

## Loading required package: Matrix

## 
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
## 
##     expand, pack, unpack

## Loaded glmnet 4.0-2
```




```
##
## Call:  cv.glmnet(x = d[, -11], y = d[, 11], nfolds = 10, family = "g
aussian",      alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Measure    SE Nonzero
## min  1.036   4.714 1.364       3
## 1se  1.137   5.387 1.662       3
```

λ_{min} 選到X_1, X_2, X_3，其參數為 9.148515,3.866065,19,799337

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##             s0
## V1   9.148515
## V2   3.866065
## V3  19.799337
## V4   .
## V5   .
## V6   .
## V7   .
## V8   .
## V9   .
## V10  .
```

λ_{lse} 選到X_1, X_2, X_3，其參數為 9.105123,3.764425,19,767835

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##             s0
## V1   9.105123
## V2   3.764425
## V3  19.767835
## V4    .
## V5    .
## V6    .
## V7    .
## V8    .
## V9    .
## V10   .
```

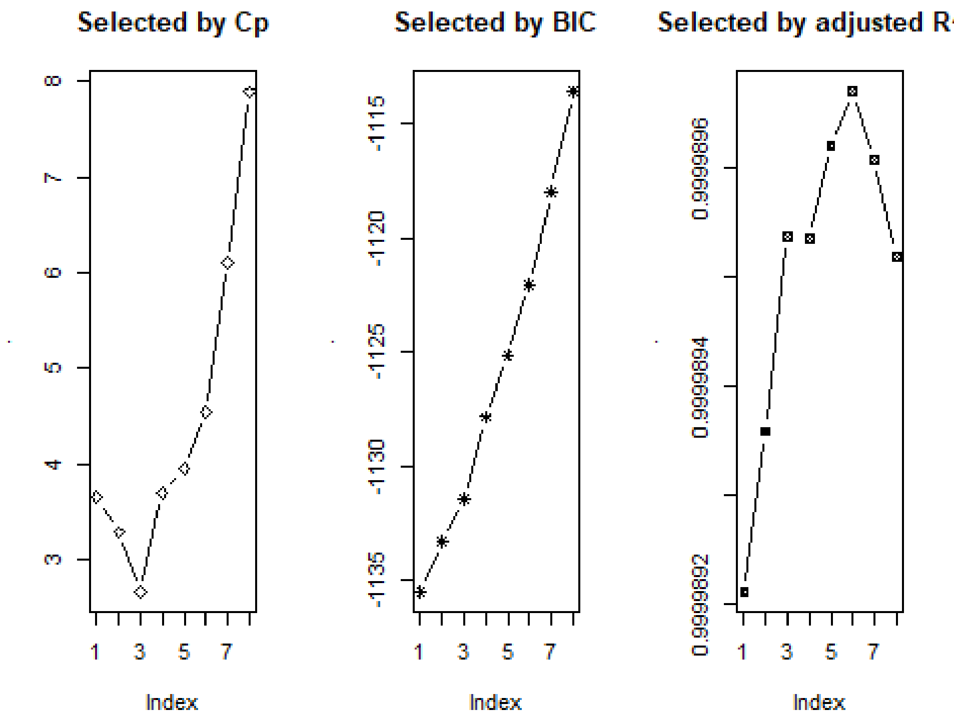
可以發現不論是λ_{min}或是λ_{lse}其所選取之變數以及所配飾參數的值 皆與使用 Forward、Backward Selection 時採用 Cp、BIC 標準時
所選取之變數相同，配飾參數的值也相近。

(f)

製造新的$Y_1 = 5 + 7X^7 + \epsilon$

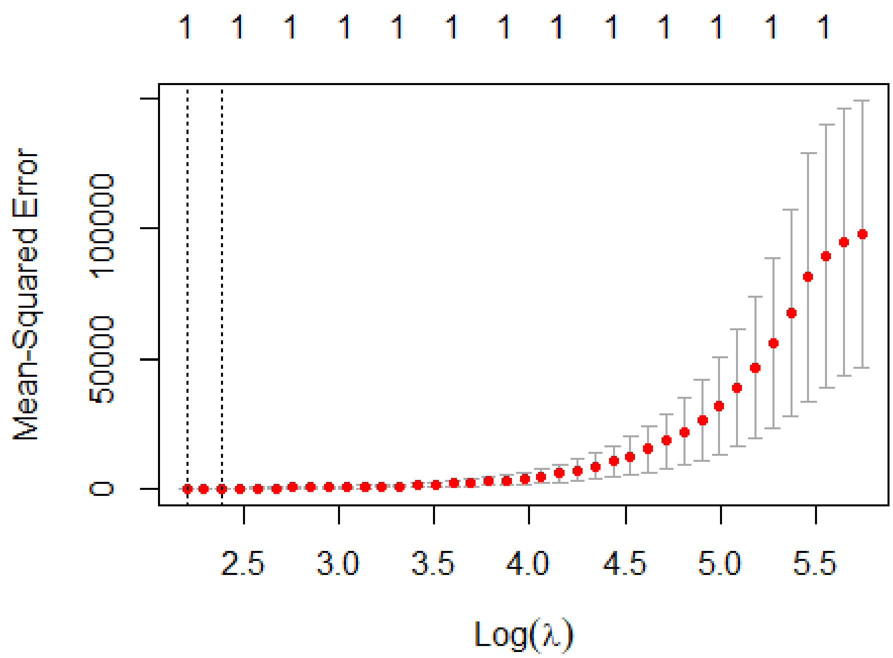
可以發現三種不同方法所選取的變數皆不同，相同的是皆選取了X_7並且參數十分接近當初所模擬的值，
可能是因為X_7為 7 次方項，整個Y_1幾乎由X_7這個變數決定，造成其餘變數估計較不準確， 但是 BIC 所選取變數與當初設定相同，而且配適參數相當接近，是三個當中表現最佳者。

```
## [1] "Cp Select: 3 Variables"

##  (Intercept)           V1            V6            V7
##  5.048981502 -0.232936749 -0.008446317  7.004208175

## [1] "BIC Select: 3 Variables"

## (Intercept)          V7
##    4.994720    7.001095
```




```
## [1] "Adj R^2: 5 Variables"

## (Intercept)          V1          V3          V4          V6
 V7
##   4.94075112 -0.53434753  0.24072919  0.60752611 -0.36064537  6.99332
263
##           V8
##   0.04919629
```

由 Lasso Regression 所篩選之變數，皆僅選取X_7出來
可以發現 Lasso Regression 可能為更保守的變數選取方法




```
##
## Call:  cv.glmnet(x = d[, -11], y = d[, 11], nfolds = 10, family = "g
aussian",      alpha = 1)
##
## Measure: Mean-Squared Error
##
##     Lambda Measure     SE Nonzero
## min  9.052   122.5  70.84       1
## 1se 10.903   176.5 102.83       1

## 10 x 1 sparse Matrix of class "dgCMatrix"
##                s0
## V1   .
## V2   .
## V3   .
## V4   .
## V5   .
## V6   .
## V7   6.791399488
## V8   .
## V9   0.000998944
## V10 .

## 10 x 1 sparse Matrix of class "dgCMatrix"
##                s0
```

```
## V1  .
## V2  .
## V3  .
## V4  .
## V5  .
## V6  .
## V7  6.749653974
## V8  .
## V9  0.000940358
## V10 .
```

(10)

(a)

建立 X、β、Y、ϵ

```
##  num [1:1000, 1:20] 0.1567 1.37381 0.73067 -1.3508 -0.00851 ...
## X

##  num [1:20, 1] 0.001 0.0001 0.001 13 0.004 0.007 15 16 0.005 17 ...
## beta

##  num [1:1000, 1] 27.3 -67 36.8 -51.8 -48.3 ...
## Y
```

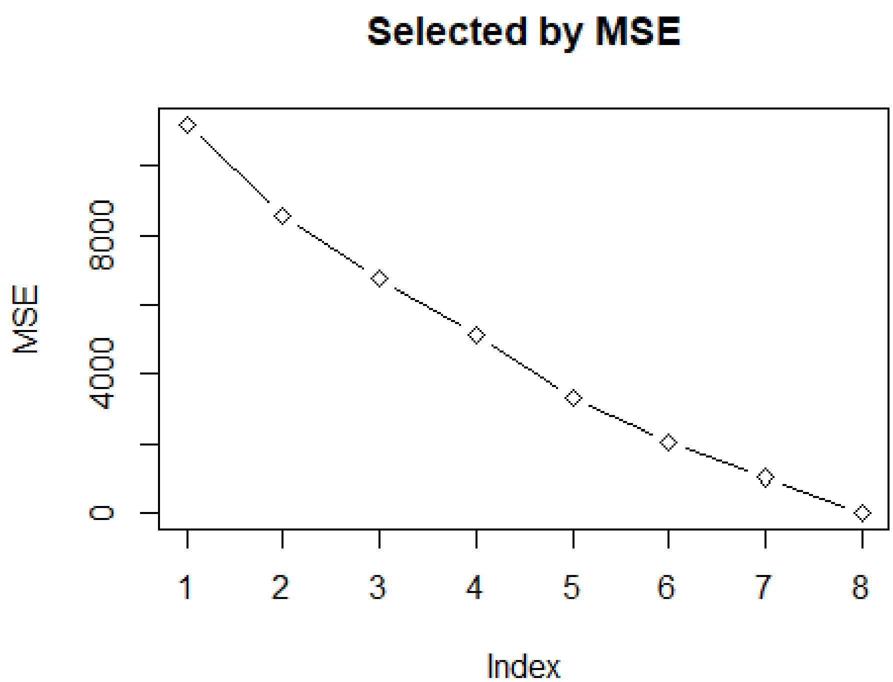
(b)

把 DATA 拆成 Train、Test，Train 有 100 筆、Test 有 900 筆

```
## Train: 100 21

## Test: 900 21
```

(c)

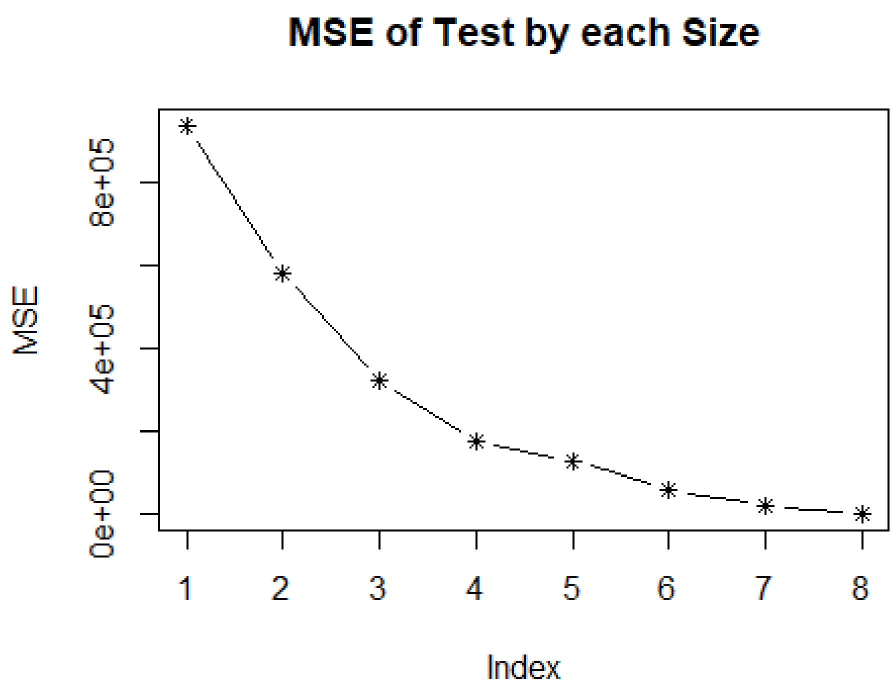
列出以 Train Data 配飾，在變數選取 1 個至 8 個時,表現最佳的模型，
列出這 8 個模型的 MSE，可以發現在增加變數個數後，MSE 顯著遞減。



(d)

用上述根據 Train Data 所選的所含變數 1 個至所含變數 8 個的最佳模型預測 Test Data，

一樣可以發現當變數數量增加，Test Data 的 MSE 隨變數數量顯著遞減。



(e)

由上圖可以看到在變數數量為 8 時有最小的 MSE，因此認為是最佳的 Model，下面列出所選的 8 個變數以及對 Test Data 預測時的 MSE。

```
##              [,1]
## (Intercept)  TRUE
## X1          FALSE
## X2          FALSE
## X3          FALSE
## X4           TRUE
## X5          FALSE
## X6          FALSE
## X7           TRUE
## X8           TRUE
## X9          FALSE
## X10          TRUE
## X11         FALSE
## X12         FALSE
## X13          TRUE
## X14          TRUE
## X15         FALSE
## X16         FALSE
## X17          TRUE
## X18         FALSE
```

```
## X19           TRUE
## X20          FALSE

## [1] "MSE of the Best Model in All Model Size(8 Variables):  113.0871
94280063"
```

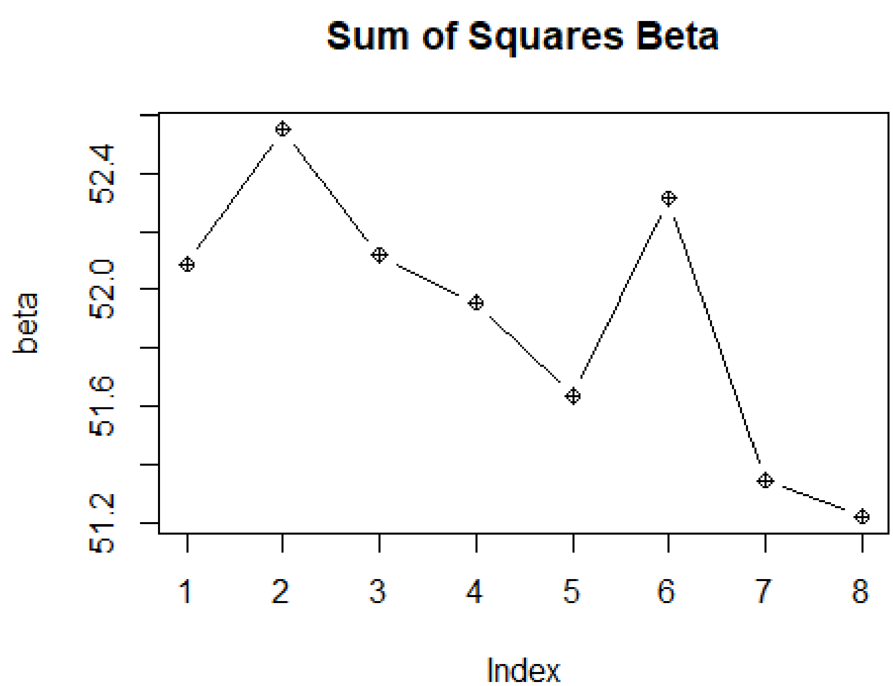
(f)

我們可以發現我們設定的所有變數都被 Train Data 所配飾的 Model 篩選出來。

```
## The significant variables that we set :  4 7 8 10 13 14 17 19

## The Variabls select in the best model using train data:  (Intercept)
 X4 X7 X8 X10 X13 X14 X17 X19
```

(g)

此題計算$\sqrt{\Sigma_{j=1}^{p}(\beta_j - \beta_j^r)^2}$，
計算在不同 Model Size 下β的距離，由圖可知當變數個數為 8 時，有最小的 Distance，
故為表現最佳的模型，但此方法並不如其他方法來的穩健， 可以看到 Distance 先隨著變數個數下降，但中間又陡升，最後變數個數為 8 時，才降至最低。



附錄：(Code)

```
library(tidyverse)
library(ISLR)
library(leaps)
### (8)
#### (a)
set.seed(1234567)
X = rnorm(100)
e = rnorm(100)
par(mfrow=c(1,2))
plot(X,main="X")
plot(e,main="e")
#### (b)
#### (c)
predictors = sapply(1:10, function(a){
  X^a
})
data = cbind(predictors,Y) %>% as.data.frame()
model = regsubsets(x=data[,1:10],y=data[,11],data=data)
summary(model)
modelsum = summary(model)
par(mfrow=c(1,3))
modelsum$cp %>% plot(lwd =1.7, cex = .8,pch= 5,type="b",main= "Selected by Cp")
paste("Cp Select:",which.min(modelsum$cp),"Variables")
coef(model,which.min(modelsum$cp))
modelsum$bic %>% plot(lwd =1.7, cex = .8,pch= 8,type="b",main= "Selected by BIC")
paste("BIC Select:",which.min(modelsum$bic),"Variables")
coef(model,which.min(modelsum$bic))
```

```
# Adj R^2

modelsum$adjr2 %>% plot(lwd =1.7, cex = .8,pch= 7,type="b",main= "Selected by adjusted R^2")

paste("Adj R^2 Select:",which.max(modelsum$adjr2),"Variables")

coef(model,which.max(modelsum$adjr2))

#### (d)

model_f = regsubsets(x=data[,1:10],y=data[,11],data=data,method = "forward")

modelf_sum = summary(model_f)

par(mfrow=c(1,3))

modelf_sum$cp %>% plot(lwd =1.7, cex = .8,pch= 5,type="b",main= "Selected by Cp")

paste("Cp Select:",which.min(modelf_sum$cp),"Variables")

coef(model,which.min(modelf_sum$cp))

modelf_sum$bic %>% plot(lwd =1.7, cex = .8,pch= 8,type="b",main= "Selected by BIC")

paste("BIC Select:",which.min(modelf_sum$bic),"Variables")

coef(model,which.min(modelf_sum$bic))

modelf_sum$adjr2 %>% plot(lwd =1.7, cex = .8,pch= 7,type="b",main= "Selected by adjusted R^2")

paste("Adj R^2:",which.max(modelf_sum$adjr2),"Variables")

coef(model,which.max(modelf_sum$adjr2))

model_f = regsubsets(x=data[,1:10],y=data[,11],data=data,method = "backward")

modelf_sum = summary(model_f)

par(mfrow=c(1,3))

# Cp

modelf_sum$cp %>% plot(lwd =1.7, cex = .8,pch= 5,type="b",main= "Selected by Cp")

paste("Cp Select:",which.min(modelf_sum$cp),"Variables")

coef(model,which.min(modelf_sum$cp))

# BIC

modelf_sum$bic %>% plot(lwd =1.7, cex = .8,pch= 8,type="b",main= "Selected by BIC")
```

```
paste("BIC Select:",which.min(modelf_sum$bic),"Variables")

coef(model,which.min(modelf_sum$bic))

# Adj R^2

modelf_sum$adjr2 %>% plot(lwd =1.7, cex = .8,pch= 7,type="b",main= "Selected by adjusted R^2")

paste("Adj R^2:",which.max(modelf_sum$adjr2),"Variables")

coef(model,which.max(modelf_sum$adjr2))

#### (e)

require(glmnet)

d = as.matrix(data)

CV_lasso = cv.glmnet(x = d[,-11],y = d[,11],

          family = "gaussian",nfold = 10,alpha = 1)

plot(CV_lasso)

CV_lasso

lasso_min = glmnet(x = d[,-11],y = d[,11],

         family = "gaussian",alpha = 1,lambda = CV_lasso$lambda.min)

lasso_lse = glmnet(x = d[,-11],y = d[,11],

         family = "gaussian",alpha = 1,lambda = CV_lasso$lambda.1se)

print(lasso_min$beta)

print(lasso_lse$beta)

#### (f)

Y_1 = 5+7*X^7+e

data = cbind(predictors,Y_1) %>% as.data.frame()

model = regsubsets(Y_1~.,data=data)

modelsum = summary(model)

par(mfrow=c(1,3))

# Cp

modelsum$cp %>% plot(lwd =1.7, cex = .8,pch= 5,type="b",main= "Selected by Cp")
```

```
paste("Cp Select:",which.min(modelf_sum$cp),"Variables")

coef(model,which.min(modelsum$cp))

# BIC

modelsum$bic %>% plot(lwd =1.7, cex = .8,pch= 8,type="b",main= "Selected by BIC")

paste("BIC Select:",which.min(modelf_sum$bic),"Variables")

coef(model,which.min(modelsum$bic))

# Adj R^2

modelsum$adjr2 %>% plot(lwd =1.7, cex = .8,pch= 7,type="b",main= "Selected by
adjusted R^2")

paste("Adj R^2:",which.max(modelf_sum$adjr2),"Variables")

coef(model,which.max(modelsum$adjr2))

require(glmnet)

d = as.matrix(data)

CV_lasso = cv.glmnet(x = d[,-11],y = d[,11],

          family = "gaussian",nfold = 10,alpha = 1)

plot(CV_lasso)

CV_lasso

lasso_min = glmnet(x = d[,-11],y = d[,11],

         family = "gaussian",alpha = 1,lambda = CV_lasso$lambda.min)

lasso_lse = glmnet(x = d[,-11],y = d[,11],

         family = "gaussian",alpha = 1,lambda = CV_lasso$lambda.1se)

print(lasso_min$beta)

print(lasso_lse$beta)

### (10)

#### (a)

set.seed(1234567)

X = matrix(rnorm(20000),ncol=20)

b = c(0.001,0.0001,0.001,13,0.004,
```

```
    0.007,15,16,0.005,17,
    0.008,-0.002,19,20,-0.001,
    0.005,22,0.003,23,0.003) %>% as.matrix(ncol=1)
e = rnorm(1000)
Y = X%*%b+e
cat("X",str(X))
cat("beta",str(b))
cat("Y",str(Y))
data = cbind(X,Y)
#### (b)
set.seed(1122)
index = sample(1:1000,100,replace = F)
data = as.data.frame(data)
names(data) = c(paste0("X",seq(1:20)),"Y")
train = data[index,]
test = data[-index,]
cat("Train:",dim(train)); cat("Test:",dim(test))
#### (c)
model = regsubsets(x = train[,1:20],y = train[,21],data=train)
model_sum = summary(model)
MSE = model_sum$rss/20
plot(MSE,lwd =1.7, cex = .8,pch= 5,type="b",main= "Selected by MSE")
#### (d)
ind = model_sum$which %>% .[1,-1] #去掉截距項
sub = train[,ind] %>% cbind(Y = train$Y) %>% as.data.frame()
names(sub) = c("X19","Y")
model = lm(Y~.,data=sub)
Y_hat = predict(model,newdata = test[,-21])
```

```r
M_1=sum((Y_hat-test$Y)^2)/dim(sub)[2]-1  #sub 扣掉 Y 其他為解釋變數
MSE = sapply(2:8, function(a){
 ind = model_sum$which %>% .[a,-1] #去掉截距項
 sub = train[,ind] %>% cbind(Y = train$Y)
 model = lm(Y~.,data=sub)
 Y_hat = predict(model,newdata = test[,-21])
 sum((Y_hat-test$Y)^2)/dim(sub)[2]-1  #sub 扣掉 Y 其他為解釋變數
})
MSE = c(M_1,MSE)
plot(MSE,lwd =1.7, cex = .8,pch= 8,type="b",main= "MSE of Test by each Size")
#### (e)
ind = model_sum$which %>% .[8,-1] #去掉截距項
sub = train[,ind] %>% cbind(Y = train$Y) %>% as.data.frame()
model_sum$which %>% .[which.min(MSE),] %>% as.matrix()
model = lm(Y~.,data=sub)
Y_hat = predict(model,newdata = test[,-21])
M_1=sum((Y_hat-test$Y)^2)/dim(sub)[2]-1  #sub 扣掉 Y 其他為解釋變數
paste("MSE of the Best Model in All Model Size(8 Variables): ",M_1)
#### (f)
```

我們可以發現我們設定的所有變數都被 Train Data 所配飾的 Model 篩選出來。

```r
cat("The significant variables that we set : ",which(b>1))
cat("The Variabls select in the best model using train data: ",model_sum$which %>% .[which.min(MSE),] %>% which(TRUE) %>% names())
model_full = lm(Y~.,data=data)
model_sub = regsubsets(Y~.,data=data)
model_sub_sum = summary(model_sub)
coef = matrix(0,ncol = 20,nrow = 8) %>% as.data.frame()
names(coef) = c(paste0("X",seq(1:20)))
```

```
for (a in 2:8) {
  ind = model_sub_sum$which %>% .[a,-1] #去掉截距項
  sub = data[,ind] %>% cbind(Y = train$Y) %>% as.data.frame()
  lm_sub = lm(Y~.,data = sub)
  coef[a,which(colnames(data) %in% names(lm_sub$coefficients)[-1])] =
lm_sub$coefficients[-1]
}
beta = sapply(1:8, function(a){
  sqrt(sum((model_full$coefficients[-1] - coef[a,])^2))
})
plot(beta,type = "b",pch= 10,main = "Sum of Squares Beta")
```