高维情形下线性模型的泛化误差研究

第二周 (12.11-12.18) 工作

Yukun Dong

目录

1 任务:

- 继续上一周，写清楚各种变量如何生成，并且补画 β 的 MSE 随其维数的变化。
- 固定模型参数，选择一部分参数进行训练，考察均方损失下的预报的风险。

1.1 固定样本数量 $n = 200$，模拟 β 的训练平均 MSE 与维数 p 的关系

固定样本数量 $n = 200$。对于给定维数 p,随机生成 $\beta^\star \in \mathbb{R}^{p+1}$, $\beta_1, \cdots, \beta_{p+1}$ iid $\sim N(0,1)$。随机生成设计阵 $X \in \mathbb{R}^{n\times(p+1)}$, x_{ij} iid $\sim N(0,1)$。再生成 $y = X\beta^\star + \epsilon$, $\epsilon \sim N(0, I_n)$。

对于训练出来的 $\hat{\beta} \in \mathbb{R}^{p+1}$，其与真实值的平均 MSE 为

$$MSE_{average} = \frac{1}{p+1}\sum_{i=1}^{p+1}(\hat{\beta}_i - \beta_i^\star)^2$$

对于 bootstrap，$\hat{\beta}$ 是对次抽样训练出来的 $\hat{\beta}_i$ 取平均；对于 cross validation，$\hat{\beta}$ 是对每个子模型训练出来的 $\hat{\beta}_i$ 取平均。

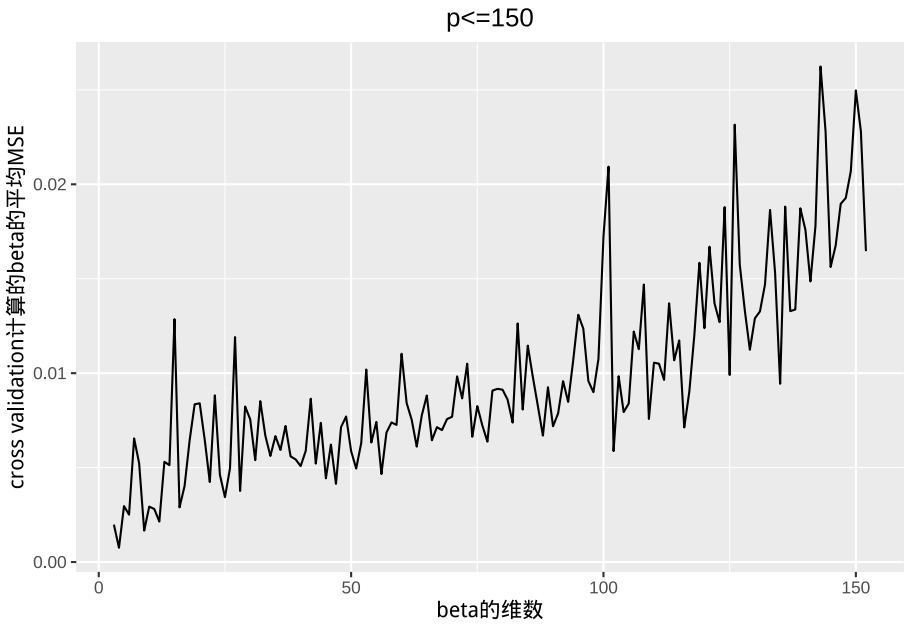
1.1.1 cross validation

```
library(ggplot2)
library(MASS)
library(showtext)
## 载入需要的程辑包: sysfonts
## 载入需要的程辑包: showtextdb
set.seed(1234)
r <- 10
K <- 5
n <- 200
d <- 150
ns <- 1:(d+1)
average.mse <- length(ns)
#ns <- floor(seq(1,n-1,length=d+1))
i <- 1

for (p in ns) {
  X <- matrix(rnorm(n * p), n, p)
```

```
  X <- cbind(rep(1, n), X)
  beta <- rnorm(p + 1)
  y <- X %*% beta + rnorm(n)
  beta_hat <- numeric(p + 1)
  for (j in 1:r) {
    indices <- sample(n)
    break.points <- round(seq(1, n, length = K + 1))
    for (k in 1:K) {
      chunk <- indices[break.points[k]:(break.points[k + 1] - 1)]
      X.train <- X[-chunk,]
      y.train <- y[-chunk]
      X.test <- X[chunk,]
      y.test <- y[chunk]
      beta.train <- ginv(X.train) %*% y.train # M-P 广义逆
      beta_hat <- beta_hat + beta.train
    }
  }
  beta_hat <- beta_hat / (r * K)
  average.mse[i] <- sum((beta_hat - beta)^2)/length(beta_hat)
  i <- i + 1
}

p_values <- ns[-1]
plot_df <- data.frame(p = p_values+1, average.mse = average.mse[-1])
showtext::showtext_begin()
ggplot(plot_df, aes(x = p)) +
  geom_line(aes(y = average.mse)) +
  xlab("beta 的维数") +
  ylab("cross validation 计算的 beta 的平均 MSE") +
  ggtitle("p<=150") +
  theme(plot.title = element_text(hjust = 0.5))
```



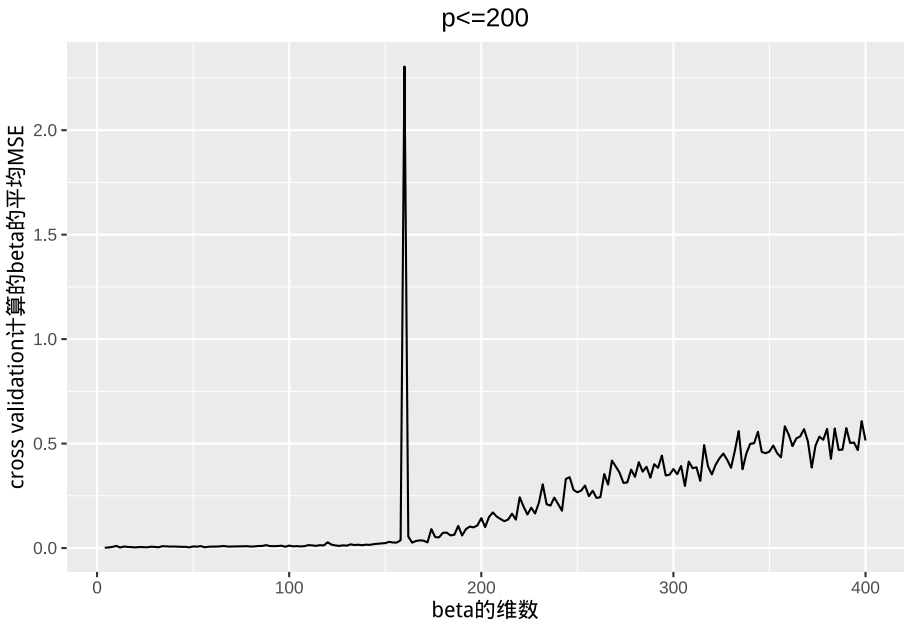

```
showtext::showtext_end()
```

```
library(ggplot2)
library(MASS)
set.seed(1234)
r <- 10
K <- 5
n <- 200
d <- 200
ns <- seq(1, 2*d, 2)
average.mse <- length(ns)
#ns <- floor(seq(1,n-1,length=d+1))
i <- 1

for (p in ns) {
  X <- matrix(rnorm(n * p), n, p)
  X <- cbind(rep(1, n), X)
  beta <- rnorm(p + 1)
```

```
  y <- X %*% beta + rnorm(n)
  beta_hat <- numeric(p + 1)
  for (j in 1:r) {
    indices <- sample(n)
    break.points <- round(seq(1, n, length = K + 1))
    for (k in 1:K) {
      chunk <- indices[break.points[k]:(break.points[k + 1] - 1)]
      X.train <- X[-chunk,]
      y.train <- y[-chunk]
      X.test <- X[chunk,]
      y.test <- y[chunk]
      beta.train <- ginv(X.train) %*% y.train # M-P 广义逆
      beta_hat <- beta_hat + beta.train
    }
  }
  beta_hat <- beta_hat / (r * K)
  average.mse[i] <- sum((beta_hat - beta)^2)/length(beta_hat)
  i <- i + 1
}

p_values <- ns[-1]
plot_df <- data.frame(p = p_values+1, average.mse = average.mse[-1])
showtext::showtext_begin()
ggplot(plot_df, aes(x = p)) +
  geom_line(aes(y = average.mse)) +
  xlab("beta 的维数") +
  ylab("cross validation 计算的 beta 的平均 MSE") +
  ggtitle("p<=200") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
showtext::showtext_end()
```

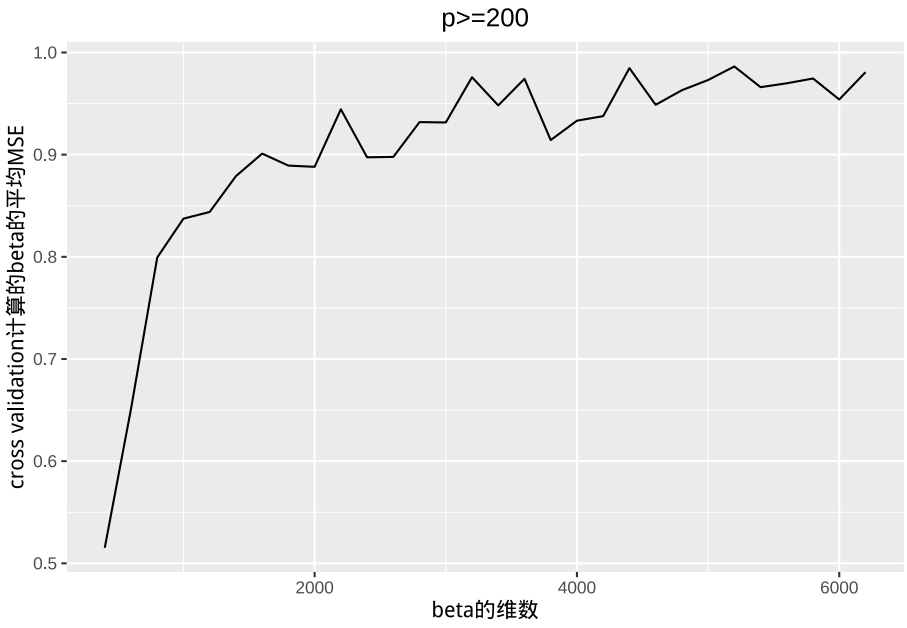
高维情形:

```
library(ggplot2)
library(MASS)
set.seed(1234)
r <- 5
K <- 5
n <- 200
d <- 30
ns <- floor(seq(n - 1, n - 1 + d * n, n))
average.mse <- length(ns)
i <- 1

for (p in ns) {
  X <- matrix(rnorm(n * p), n, p)
  X <- cbind(rep(1, n), X)
```

```
  beta <- rnorm(p + 1)
  y <- X %*% beta + rnorm(n)
  beta_hat <- numeric(p + 1)
  for (j in 1:r) {
    indices <- sample(n)
    break.points <- round(seq(1, n, length = K + 1))
    for (k in 1:K) {
      chunk <- indices[break.points[k]:(break.points[k + 1] - 1)]
      X.train <- X[-chunk,]
      y.train <- y[-chunk]
      X.test <- X[chunk,]
      y.test <- y[chunk]
      beta.train <- ginv(X.train) %*% y.train # M-P 广义逆
      beta_hat <- beta_hat + beta.train
    }
  }
  beta_hat <- beta_hat / (r * K)
  average.mse[i] <- sum((beta_hat - beta)^2)/length(beta_hat)
  i <- i + 1
}

p_values <- ns[-1]
plot_df <- data.frame(p = p_values+1, average.mse = average.mse[-1])
showtext::showtext_begin()
ggplot(plot_df, aes(x = p)) +
  geom_line(aes(y = average.mse)) +
  xlab("beta 的维数") +
  ylab("cross validation 计算的 beta 的平均 MSE") +
  ggtitle("p>=200") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
showtext::showtext_end()
```

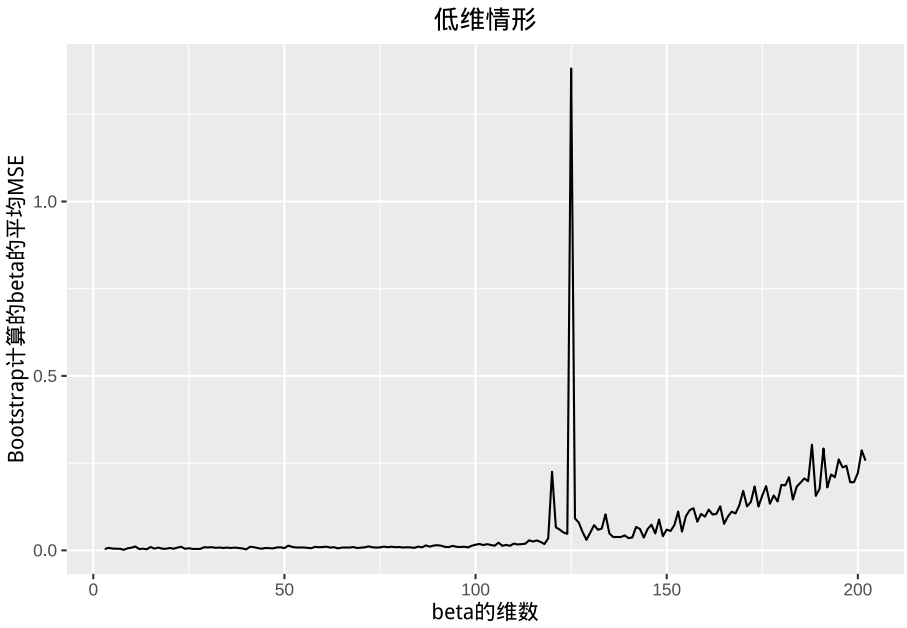
1.1.2 bootstrap

低维情形:

```
library(ggplot2)
library(MASS)
library(showtext)
set.seed(1234)
B <- 10
n <- 200
d <- 200
errors <- numeric(d + 1)
ns <- 1:(d+1)
average.mse <- numeric(length(ns))
i <- 1
```

```r
for (p in ns) {
  X <- matrix(rnorm(n * p), n, p)
  X <- cbind(rep(1, n), X)
  beta <- rnorm(p + 1)
  y <- X %*% beta + rnorm(n)
  beta_hat <- numeric(p + 1)
  for (j in 1:B) {
    indices <- sample(1:n, n, replace = TRUE)
    left.indices <- setdiff(1:n, unique(indices))
    X.b <- X[indices, ]
    y.b <- y[indices]
    X.left <- X[left.indices, ]
    y.left <- y[left.indices]
    beta.train <- ginv(X.b) %*% y.b # M-P 广义逆
    beta_hat <- beta_hat + beta.train
  }
  beta_hat <- beta_hat / (B)
  average.mse[i] <- sum((beta_hat - beta)^2)/length(beta_hat)
  i <- i + 1
}

p_values <- ns[-1]
plot_df <- data.frame(p = p_values+1, average.mse = average.mse[-1])
showtext_begin()
ggplot(plot_df, aes(x = p)) +
  geom_line(aes(y = average.mse)) +
  xlab("beta 的维数") +
  ylab("Bootstrap 计算的 beta 的平均 MSE") +
  ggtitle(" 低维情形") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
showtext_end()
```

维数较高情形:

```
library(ggplot2)
library(MASS)
library(showtext)
set.seed(1234)
B <- 10
n <- 200
d <- 30
errors <- numeric(d + 1)
ns <- floor(seq(n - 1, n - 1 + d * n, n))
average.mse <- numeric(length(ns))
i <- 1

for (p in ns) {
  X <- matrix(rnorm(n * p), n, p)
```

```
  X <- cbind(rep(1, n), X)
  beta <- rnorm(p + 1)
  y <- X %*% beta + rnorm(n)
  beta_hat <- numeric(p + 1)
  for (j in 1:B) {
    indices <- sample(1:n, n, replace = TRUE)
    left.indices <- setdiff(1:n, unique(indices))
    X.b <- X[indices, ]
    y.b <- y[indices]
    X.left <- X[left.indices, ]
    y.left <- y[left.indices]
    beta.train <- ginv(X.b) %*% y.b # M-P 广义逆
    beta_hat <- beta_hat + beta.train
  }
  beta_hat <- beta_hat / (B)
  average.mse[i] <- sum((beta_hat - beta)^2)/length(beta_hat)
  i <- i + 1
}

p_values <- ns[-1]
plot_df <- data.frame(p = p_values+1, average.mse = average.mse[-1])
showtext_begin()
ggplot(plot_df, aes(x = p)) +
  geom_line(aes(y = average.mse)) +
  xlab("beta 的维数") +
  ylab("Bootstrap 计算的 beta 的平均 MSE") +
  ggtitle(" 高维情形") +
  theme(plot.title = element_text(hjust = 0.5))
```


```
showtext_end()
```

1.2 固定模型参数，选择一部分参数进行训练，考察均方损失下的预报的风险

这一次的模拟理论基础来自 Mikhail Belkin 等在 2020 年的文章 Two models of double descent for weak features。其模型设定为：给定模型参数 $\beta^\star = (\beta_0, \cdots, \beta_D)^T \in \mathbb{R}^{D+1}$，对于 $p < D$，随机选取大小为 p 的子集 T 作为训练集，即 $\hat{\beta}_T = (X_T' X_T)^+ X_T y; \hat{\beta}_{T^c} = 0$。在这个缩减的模型之下进行计算 $\mathbb{E}[(y - x\hat{\beta})^2]$。

1.2.1 实验模拟的设定

生成 $\beta \in \mathbb{R}^{1000}, \beta_i \text{iid} \sim \text{Unif}(0,1)$，再将 β 标准化得到 $\beta^\star, \|\beta^\star\| = 1$。随机生成 $n \times d$ 标准正态设计阵 X，生成样本 $y = X\beta^\star + \epsilon,\ \epsilon \sim N(0, 0.1^2 I_n)$。样本个数 n 取在 200 到 10000 不等，尤其在 $n \approx p$ 的时候增加样本点。不论 n 多大，计算泛化误差的时候，固定测试集大小为 10000。

1.2.2 实验模拟

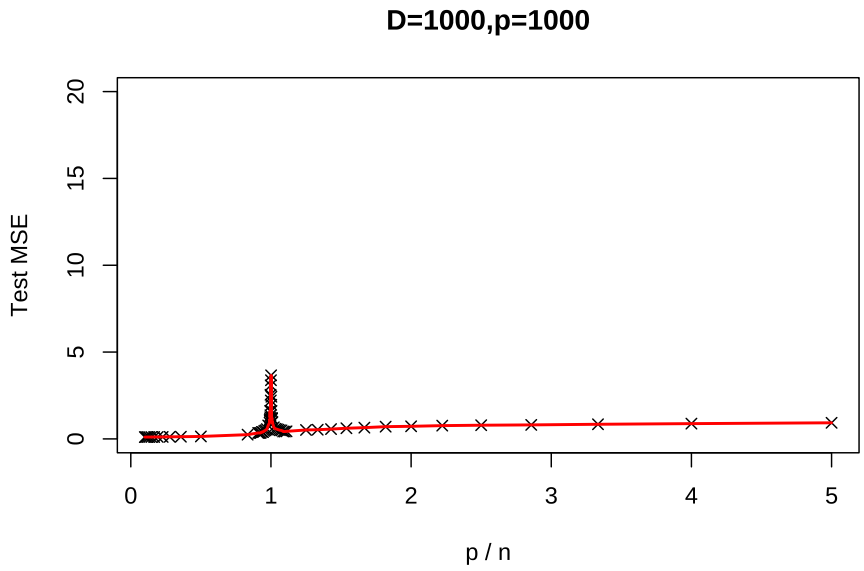
```r
library(MASS)

beta = runif(1000) # real coefficients
beta = beta/sqrt(sum(beta^2)) # convert to a unit vector
M = 3 # number of simulations
N = c(seq(200, 800, 50), seq(900, 990, 10), seq(991,1000,1),
      seq(1001, 1009, 1), seq(1010, 1100, 10), seq(1200, 10000, 800)) # number of sampl
test_MSE_1000 = matrix(nrow = length(N), ncol = M)

for (i in 1:length(N)){
  for (m in 1:M){
    X = replicate(1000, rnorm(N[i]))
    e = rnorm(N[i], sd = 0.1)
    y = X %*% beta + e
    if (N[i] < 1000){
      beta_hat = ginv(X) %*% y
    } else {
      dat = as.data.frame(cbind(y, X))
      names(dat)[1] = "y"
      lm_model = lm(y ~ .-1, data = dat)
      beta_hat = matrix(lm_model$coefficients, ncol = 1)
    }
    X_test = replicate(1000, rnorm(N[i]))
    e_test = rnorm(N[i], sd = 0.1)
    y_test = X_test %*% beta + e_test
    preds_test = X_test %*% beta_hat
    test_MSE_1000[i, m] = sqrt(mean((y_test - preds_test)^2))
  }
}

plot(1000/N,apply(test_MSE_1000, 1, mean), ylab = "Test MSE", xlab = "p / n",
```

```
     pch = 4,
     ylim = c(0,20), main = "D=1000,p=1000")
lines(1000/N, apply(test_MSE_1000, 1, mean), col = "red", lwd = 2)
```



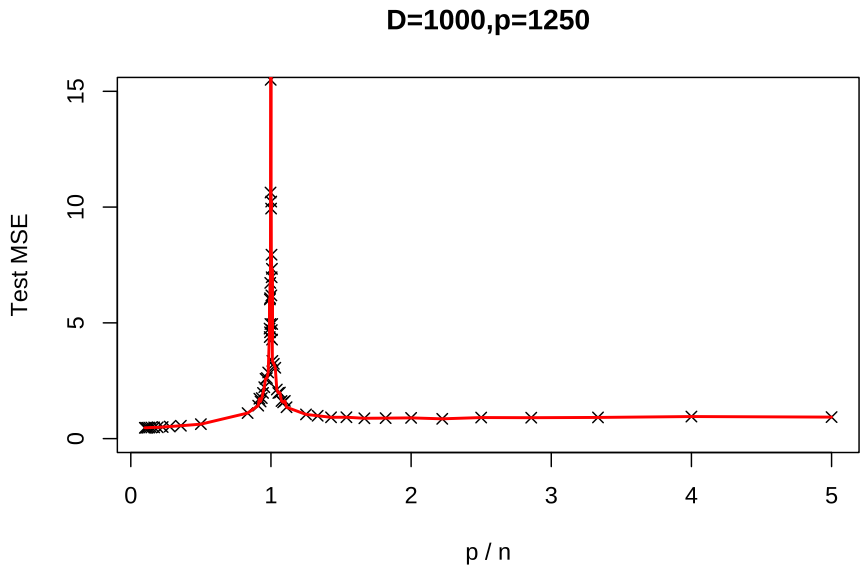

```
library(MASS)

beta = runif(1250) # real coefficients
beta = beta/sqrt(sum(beta^2)) # convert to a unit vector
M = 3 # number of simulations
N = c(seq(200, 800, 50), seq(900, 990, 10), seq(991,1000,1),
      seq(1001, 1009, 1), seq(1010, 1100, 10), seq(1200, 10000, 800)) # number of sampl
test_MSE_1250 = matrix(nrow = length(N), ncol = M)

for (i in 1:length(N)){
  for (m in 1:M){
    X = replicate(1250, rnorm(N[i]))
    X_T <- X[, 1:1000]
    e = rnorm(N[i], sd = 0.1)
```

```
    y = X %*% beta + e
    if (N[i] < 1000){
      beta_hat = ginv(X_T) %*% y
      beta_hat <- c(beta_hat, rep(0,250))
    } else {
      dat = as.data.frame(cbind(y, X_T))
      names(dat)[1] = "y"
      lm_model = lm(y ~ .-1, data = dat)
      beta_hat = matrix(lm_model$coefficients, ncol = 1)
      beta_hat <- c(beta_hat, rep(0,250))
    }
    X_test = replicate(1250, rnorm(N[i]))
    e_test = rnorm(N[i], sd = 0.1)
    y_test = X_test %*% beta + e_test
    preds_test = X_test %*% beta_hat
    test_MSE_1250[i, m] = sqrt(mean((y_test - preds_test)^2))
  }
}

plot(1000/N,apply(test_MSE_1250, 1, mean), ylab = "Test MSE", xlab = "p / n",
     pch = 4,
     ylim = c(0,15), main = "D=1000,p=1250")
lines(1000/N, apply(test_MSE_1250, 1, mean), col = "red", lwd = 2)
```



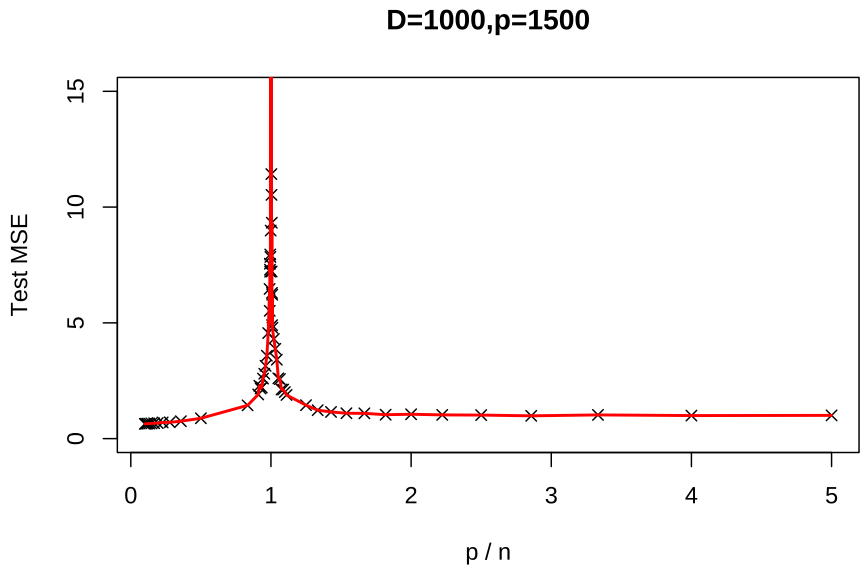

```
library(MASS)

beta = runif(1500) # real coefficients
beta = beta/sqrt(sum(beta^2)) # convert to a unit vector
M = 3 # number of simulations
N = c(seq(200, 800, 50), seq(900, 990, 10), seq(991,1000,1),
      seq(1001, 1009, 1), seq(1010, 1100, 10), seq(1200, 10000, 800)) # number of sampl
test_MSE_1500 = matrix(nrow = length(N), ncol = M)

for (i in 1:length(N)){
  for (m in 1:M){
    X = replicate(1500, rnorm(N[i]))
    X_T <- X[, 1:1000]
    e = rnorm(N[i], sd = 0.1)
    y = X %*% beta + e
    if (N[i] < 1000){
      beta_hat = ginv(X_T) %*% y
      beta_hat <- c(beta_hat, rep(0,500))
```

```
    } else {
      dat = as.data.frame(cbind(y, X_T))
      names(dat)[1] = "y"
      lm_model = lm(y ~ .-1, data = dat)
      beta_hat = matrix(lm_model$coefficients, ncol = 1)
      beta_hat <- c(beta_hat, rep(0,500))
    }
    X_test = replicate(1500, rnorm(N[i]))
    e_test = rnorm(N[i], sd = 0.1)
    y_test = X_test %*% beta + e_test
    preds_test = X_test %*% beta_hat
    test_MSE_1500[i, m] = sqrt(mean((y_test - preds_test)^2))
  }
}

plot(1000/N,apply(test_MSE_1500, 1, mean), ylab = "Test MSE", xlab = "p / n",
     pch = 4,
     ylim = c(0,15), main = "D=1000,p=1500")
lines(1000/N, apply(test_MSE_1500, 1, mean), col = "red", lwd = 2)
```

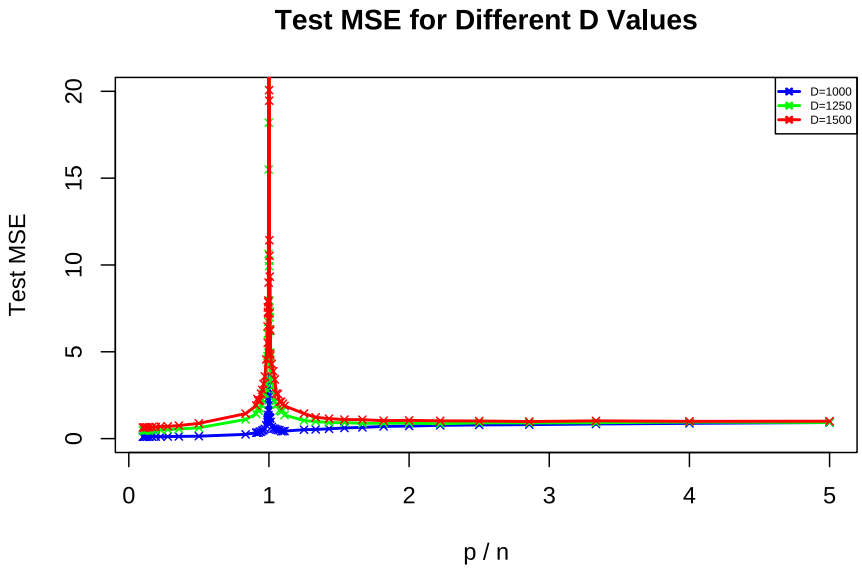


不同 D 的结果画在一张图上：

```
plot(1000/N, rep(0, length(N)), type="n", ylim=c(0, 20), xlab="p / n", ylab="Test MSE",

lines(1000/N, apply(test_MSE_1000, 1, mean), col="blue", lwd=2)
points(1000/N, apply(test_MSE_1000, 1, mean), col="blue", pch=4, cex=0.7)
lines(1000/N, apply(test_MSE_1250, 1, mean), col="green", lwd=2)
points(1000/N, apply(test_MSE_1250, 1, mean), col="green", pch=4, cex=0.7)
lines(1000/N, apply(test_MSE_1500, 1, mean), col="red", lwd=2)
points(1000/N, apply(test_MSE_1500, 1, mean), col="red", pch=4, cex=0.7)

legend("topright", legend=c("D=1000", "D=1250", "D=1500"), col=c("blue", "green", "red"
```



根据文章，有如下关系：

$$\mathbb{E}[(y-X'\beta)^2] = \begin{cases} [(1-\frac{p}{D})\,\|\beta\|^2+\sigma^2]\cdot\left(1+\frac{p}{n-p-1}\right) & \text{if } p \le n-2; \\ +\infty & \text{if } n-1 \le p \le n+1; \\ \|\beta\|^2\cdot\left[1-\frac{n}{D}\left(2-\frac{D-n-1}{p-n-1}\right)\right]+\sigma^2\cdot\left(1+\frac{n}{p-n-1}\right) & \text{if } p \ge n+2. \end{cases}$$

即在子集大小接近样本个数的时候，泛化误差会增大，而这之后 p 增大泛化误差降低。上图的结果与理论是较为符合的。

1.3 复现 Hasite 的部分结果

还在阅读中。

1.4 Double Descent

在统计学和机器学习中，**双降现象 (double descent)** 是一种现象，即拥有少量参数的统计模型和拥有极大数量参数的模型都会有较小的错误，但是

当一个模型的参数数量大约与用来训练该模型的数据点数量相同时，该模型将会有较大的错误。

Belkin et al.(2018) observed a phenomenon that test error decreased again after the expected U-shaped curve when the model size increased. They called this phenomenon "double descent".

Nakkrian et al.(2019) conducted simulations on a wide range of neutral network models. And found not only the model size but also the training epochs will bring double descent.

Hastie et al.(2020) found double descent occurring in fundamental models like least squares regression. And they believed when $p > n$, the variance decreases as p grows.