

統計模擬 作業三

109354003 統碩一 吳書恆

109354027 統碩一 蔡海蓮

4/30/2021

1. Given the following data, use the orthogonalization methods such as Cholesky or QR to perform regression analysis, including the parameter estimates and their standard errors via the sweep operator. Note that, to solve the linear equation, you may use the Gauss elimination or use the function “lm” or “glm” in R.

x_1 Reactor temperature (°C)	x_2 Ratio of H ₂ to n-heptane (mole ratio)	x_3 Contact time (sec.)	y Conversion of n-heptane to acetylene(%)
1300	7.5	0.0120	49.0
1300	9.0	0.0120	50.2
1300	11.0	0.0115	50.5
1300	13.5	0.0130	48.5
1300	17.0	0.0135	47.5
1300	23.0	0.0120	44.5
1200	5.3	0.0400	28.0
1200	7.5	0.0380	31.5
1200	11.0	0.0320	34.5
1200	13.5	0.0260	35.0
1200	17.0	0.0340	38.0
1200	23.0	0.0410	38.5
1100	5.3	0.0840	15.0
1100	7.5	0.0980	17.0
1100	11.0	0.0920	20.5
1100	17.0	0.0860	29.5

It is anticipated that an equation of the following form would fit the data

$$E(Y) = \beta_0 + \sum \beta_i X_i + \sum \beta_{ii} X_i^2 + \sum_{i < j} \beta_{ij} X_i X_j, \text{ and } Var(Y) = \sigma^2$$

令 X 是資料矩陣，包含所有一次、二次與交互作用項，則透過 Cholesky 可以求得迴歸係數，

$$LL'\hat{\beta} = X'y, \text{ where } LL' \text{ is cholesky decomposition of } X'X.$$

而為了去估計係數 $\hat{\beta}$ 的標準誤，需透過 sweep operator 去估計，過程如下

$$SSCP = \begin{pmatrix} X'X & X'y \\ y'X & y'y \end{pmatrix} \xrightarrow{\text{sweep column 1 to p}} \begin{pmatrix} -(X'X)^{-1} & \hat{\beta} \\ \hat{\beta}' & RSS \end{pmatrix}$$

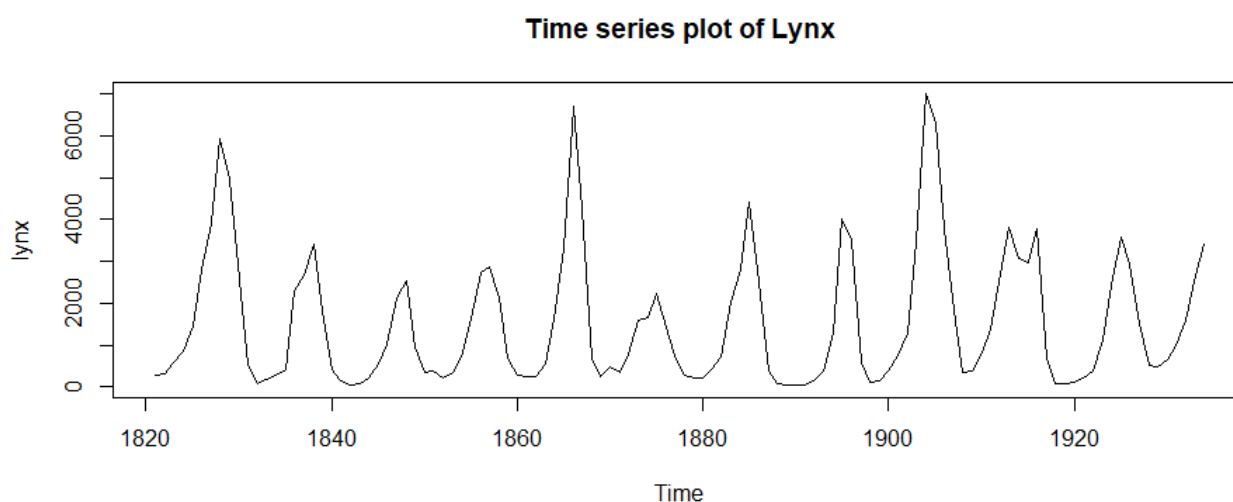
又 $Var(\hat{\beta}) = \sigma^2(X'X)^{-1}$, where $\sigma^2 = RSS/(n - p)$.

有了以上定義，計算出的係數估計列出如下表格，並一併與 lm() 函數進行比較。

	Cholesky	sweep	lm	lm
	$\hat{\beta}$	$se(\hat{\beta})$	$\hat{\beta}$	$se(\hat{\beta})$
Intercept	-3617.23	3136.04	-3617.23	3136.04
x1	5.32	4.88	5.32	4.88
x2	19.24	4.30	19.24	4.30
x3	13766.32	10448.32	13766.32	10448.32
x1^2	0.00	0.00	0.00	0.00
x2^2	-0.03	0.01	-0.03	0.01
x3^2	-11581.68	7698.61	-11581.68	7698.61
x1*x2	-0.01	0.00	-0.01	0.00
x1*x3	-10.58	8.24	-10.58	8.24
x2*x3	-21.03	9.24	-21.03	9.24

2. Figure a way to find the parameters of AR(1) and AR(2) models for the data “lynx” in R. Also, apply statistical software such as R, SAS, SPSS, and Minitab to get estimates for the AR(1) and AR(2) model and compare them to those from your program.

Lynx 資料時間序列圖如下，



AR(p)過程的定義如下：

$\{X_n: n = 0, \pm 1, \pm 2, \dots\}$ 為一時間序列，滿足對 $\forall n \in Z$

$$X_n = a_0 + a_1X_{n-1} + a_2X_{n-2} + \dots + a_pX_{n-p} + \epsilon_n,$$

$\{\epsilon_n: n = 0, \pm 1, \pm 2, \dots\}$ 為誤差序列且 $Var(\epsilon_n) = \sigma^2$ 。

有了以上定義，我們可以在 SAS 裡頭寫個線性估計式，與透過 R 的 `ar.ols()` 的結果差不多，但截距項有些差異。

```

#SAS#
proc model data=lynx;
  parms p0 p1;
  x = p0 + p1* lag1(x);
  fit x;
proc model data=lynx;
  parms p0 p1 p2;
  x = p0 + p1* lag1(x) + p2 * lag2(x);
  fit x;
run;
#R#
ar.ols(lynx, order = 1)
ar.ols(lynx, order = 2)

```

		SAS	R
AR(1)	a0	454.15	23.060
	a1	0.7197	0.7197
AR(2)	a0	710.10	12.140
	a1	1.1524	1.1524
	a2	-0.6062	-0.6062

3. Singular Value Decomposition (SVD) and Principal Component Analysis (PCA)

both can be used to reduce the data dimensionality. Please go to the webpage of Ministry of Interior and download Taiwan mortality data, 17 five-age groups for ages 0~4, 5~9, ..., 80~84 in 1991-2015, and use these data to demonstrate how these two methods work. The data of the years 1991-2010 are used as the “training” (in-sample) data and the years 2010-2015 are used as the “testing” (out-sample) data. Comments on your findings.

首先以下是子是用來估計死亡人數的模型

$$\ln(m_{xt}) = \alpha_x + \beta_x \kappa_t + \epsilon_{xt}$$

其中, α_x 表示每個年齡層的平均死亡人數; β_x 表示隨著年齡層增加的死亡人數增加率; κ_t 表示隨年增加的死亡人數增加率。為了估計 β_x 和 κ_t , 我們對 $\ln(m_{xt}) - \alpha_x$ 做 SVD 拆解, 意謂

$$\ln(m_{xt}) - \alpha_x = (UD)V' = (\widehat{\beta}_x) \widehat{\kappa}_t'$$

於是根據 1992~2010 的資料, 我們做了 SVD 後, 分別得到相對應的 $\widehat{\beta}_x$ 和 $\widehat{\kappa}_t$ 。為了預測往後 2011~2015 年的死亡人數, 我們透過 AR(1) 模型預測了 $\widehat{\kappa}_t'$ 模型如下:

$$\widehat{\kappa}_t' = 0.1561 + 1.0101 \widehat{\kappa}_{t-1}'$$

並透過以下模型預測,

$$\ln(m_{xt'}) = \alpha_x + \widehat{\beta}_x \widehat{\kappa}_t'$$

預測結果如下

```
> round(xnew.appr, 2)
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] ... [,17]
2011 -0.62 -0.60 -0.69 -0.74 -0.49 -0.29 -0.23 -0.09 0.09 ... 0.46
2012 -0.68 -0.67 -0.77 -0.82 -0.54 -0.32 -0.25 -0.10 0.10 ... 0.51
2013 -0.74 -0.73 -0.84 -0.90 -0.59 -0.35 -0.27 -0.11 0.11 ... 0.56
2014 -0.81 -0.79 -0.91 -0.98 -0.64 -0.39 -0.30 -0.12 0.12 ... 0.61
2015 -0.88 -0.86 -0.99 -1.06 -0.69 -0.42 -0.32 -0.13 0.13 ... 0.66
```

很可惜與真實資料的對數死亡人數差距有 20.84，說明降維拿來做預測估計沒有很好。

```
> sum((xnew - xnew.appr)^2)
[1] 20.84244
```

❖ PCA 降維待補 !!

4. (a) Write a small program to perform the “Permutation test” and test your result on the correlation of DDT vs. eggshell thickness in class, and the following data:

X	585	1002	472	493	408	690	291
Y	0.1	0.2	0.5	1.0	1.5	2.0	3.0

Check your answer with other correlation tests, such as regular Pearson and Spearman correlation coefficients.

- (b) Simulate a set of two correlated normal distribution variables, with zero mean and variance 1. Let the correlation coefficient be 0.2 and 0.8. (Use Cholesky!) Then convert the data back to Uniform(0,1) and record only the first decimal number. (亦即只取小數第一位，0至9的整數) Suppose the sample size is 10. Apply the permutation test, Pearson and Spearman correlation coefficients, and records the p-values of these three methods. (10,000 simulation runs)

(a) 此題有兩筆資料要進行相關性分析，首先是 DDT 與 egg 資料，此資料可以目測出呈現若相關，因此若此資料算出的 $\sum x_i y_i$ 小於 permutations 亂數算出的 $\sum x_i y_i$ 的個數夠少，才能說此負相關顯著的。執行結果告訴我們拒絕虛無假設。

```
> length(which(DDT2 %*% sort(egg) <= sum(DDT * egg)))/nrow(DDT2)
[1] 0.03333333
```

但 Pearson 和 Spearman 的相關係數估計結果都是不顯著的。

```
> cor.test(DDT, egg, method = "pearson")
...
t = -1.5343, df = 3, p-value = 0.2225
> cor.test(DDT, egg, method = "spearman")
...
S = 37.442, p-value = 0.05385
```

另外一筆資料是題目給的(X, Y), 結果如下:

```
> #Permutations test
> length(which(x2 %*% sort(y) <= sum(x * y))/nrow(x2))
[1] 0.07757937
> cor.test(x, y, method="pearson")
...
t = -1.5083, df = 5, p-value = 0.1919
> cor.test(x, y, method="spearman")
...
S = 86, p-value = 0.2357
```

(b)首先生成兩筆 $N(0, 1)$ 資料, 對其進行 Cholesky 職轉換後, 再將其轉為 uniform , 並取小數點後第一位, 最後在檢定其相關性是否仍顯著存在。函數撰寫如下:

```
choln2u <- function(rho = 0.2){
  A <- matrix(c(1, rho, rho, 1), ncol=2)
  B <- t(chol(A))
  t1 <- NULL
  t2 <- NULL
  t3 <- NULL
  for (i in 1:100) {
    x1 <- rnorm(10)
    y1 <- rnorm(10)
    xy1 <- rbind(x1, y1)
    xy <- B %*% xy1
    xy.unif <- floor(10*pnorm(xy))
    z1 <- cor.test(xy.unif[1, ], xy.unif[2, ], method =
"pearson")$p.value
    z2 <- cor.test(xy.unif[1, ], xy.unif[2, ], method =
"spearman")$p.value
    t1 <- c(t1, z1)
    t2 <- c(t2, z2)
    t <- NULL
    z0 <- sum(x1*y1)
    for (j in 1:1000) {
      x0 <- sample(x1, 10, F)
      y0 <- sample(y1, 10, F)
      t <- c(t, sum(x0*y0))
    }
    z3 <- sum(t >= z0)/1000
    t3 <- c(t3, z3)
  }
  print(paste('pearson p<0.05 個數:', length(t1[t1<0.05])))
  print(paste('spearman p<0.05 個數:', length(t2[t2<0.05])))
  print(paste('permutation test p<0.05 個數:', length(t2[t2<0.05])))
}
```

設定 rho=0.2 和 0.8, 分別執行 100 次的結果如下:

```
> choln2u(0.2)
[1] "pearson p<0.05 個數: 5"
[1] "spearman p<0.05 個數: 5"
[1] "permutation test p<0.05 個數: 5"
```

```
> choln2u(0.8)
[1] "pearson p<0.05 個數: 85"
[1] "spearman p<0.05 個數: 78"
[1] "permutation test p<0.05 個數: 78"
```

發現當相關性很高時，轉換過後的數值被認為是顯著相關的次數也會增加。

5. Using simulation to construct critical values of the Mann-Whitney-Wilcoxon test in the case that $2 \leq n_1, n_2 \leq 10$, where n_1 and n_2 are the number of observations in two populations. (Note: The number of replications shall be at least 10,000.)

❖ 待補 !!

6. Similar to what Efron did in the Law school data example, compute the bootstrap simulation for 50, 100, ..., 10,000 replications. But, instead of using the original 15 observations, we want to know if the number of observations plays an important role. Randomly select 10, 15, 20, and 25 observations and then see if the bootstrap variance converges as the number of replications increases. (Note: You also need to compare your results with that of population.)

school	LSAT	GPA	school	LSAT	GPA	school	LSAT	GPA	school	LSAT	GPA
1	622	3.23	22	614	3.19	43	573	2.85	63	572	3.08
2	542	2.83	23	628	3.03	44	644	3.38	64	610	3.13
3	579	3.24	24	575	3.01	(45)	545	2.76	65	562	3.01
(4)	653	3.12	25	662	3.39	46	645	3.27	66	635	3.30
5	606	3.09	26	627	3.41	(47)	651	3.36	67	614	3.15
(6)	576	3.39	27	608	3.04	48	562	3.19	68	546	2.82
7	620	3.10	28	632	3.29	49	609	3.17	69	598	3.20
8	615	3.40	29	587	3.16	(50)	555	3.00	(70)	666	3.44
9	553	2.97	30	581	3.17	51	586	3.11	71	570	3.01
10	607	2.91	(31)	605	3.13	(52)	580	3.07	72	570	2.92
11	558	3.11	32	704	3.36	(53)	594	2.96	73	605	3.45
12	596	3.24	33	477	2.57	54	594	3.05	74	565	3.15
(13)	635	3.30	34	591	3.02	55	560	2.93	75	686	3.50
14	581	3.22	(35)	578	3.03	56	641	3.28	76	608	3.16
(15)	661	3.43	(36)	572	2.88	57	512	3.01	77	595	3.19
16	547	2.91	37	615	3.37	58	631	3.21	78	590	3.15
17	599	3.23	38	606	3.20	59	597	3.32	(79)	558	2.81
18	646	3.47	39	603	3.23	60	621	3.24	80	611	3.16
19	622	3.15	40	535	2.98	61	617	3.03	81	564	3.02
20	611	3.33	41	595	3.11	62	637	3.33	(82)	575	2.74
21	546	2.99	42	575	2.92						

Sampled schools have bold-faced school numbers.

根據提議要求，我們寫個函數可以看執行 50, 100, ..., 10000 次的拔靴法，其變異數估計狀況折線圖，虛線是真實資料變異數值，因為有兩個變數，所以最後的圖形也會有兩個，函數如下：

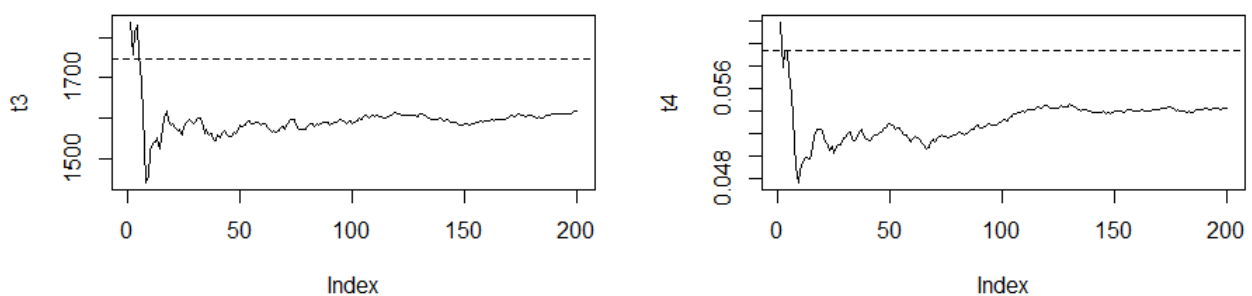
```
> library(bootstrap)
> law.bs <- function(x){
+   t1 <- NULL
+   t2 <- NULL
+   t3 <- NULL
```

```

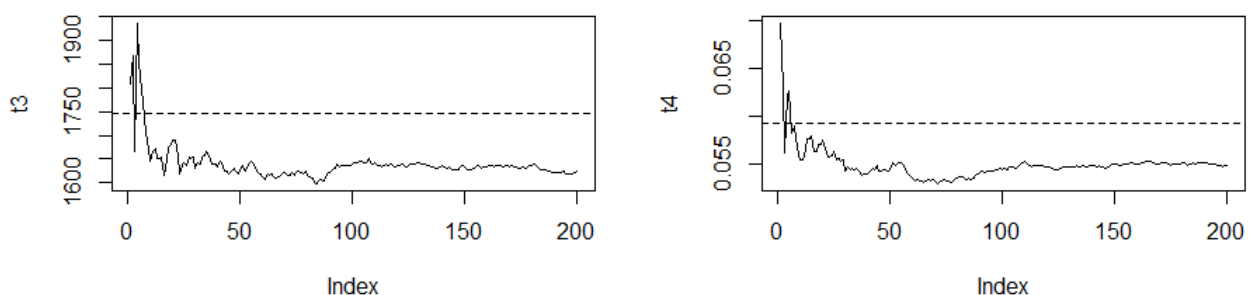
+   t4 <- NULL
+   for(j in 1:200){
+     k <- 50*j
+     for (i in k) {
+       x1 <- sample(1:15, x, T)
+       x2 <- law[x1,]
+       x3 <- array(apply(x2, 2, var))
+       t1 <- c(t1, x3[1])
+       t2 <- c(t2, x3[2])
+     }
+     t3 <- c(t3, mean(t1))
+     t4 <- c(t4, mean(t2))
+   }
+   par(mfrow=c(1,2))
+   p1 <- rbind(plot(t3, type = 'l'), abline(h = var(law$LSAT), lty =
+ 2))
+   p2 <- rbind(plot(t4, type = 'l'), abline(h = var(law$GPA), lty =
+ 2))
+   return(list(p1, p2))
+ }

```

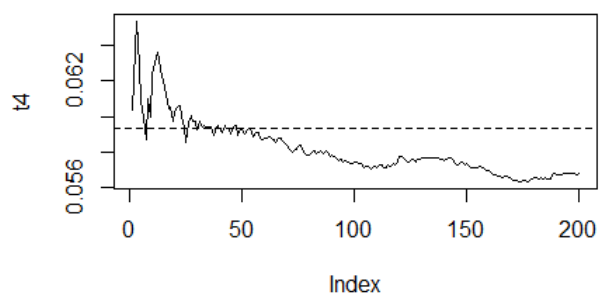
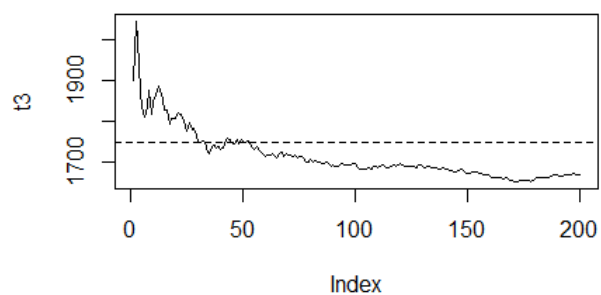
執行隨機抽取 15 個中抽取 10 個的 `law.bs()` 結果如下：



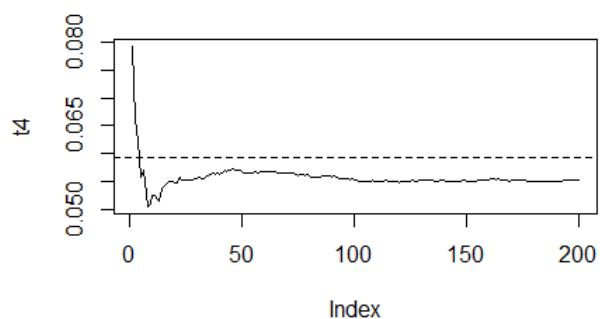
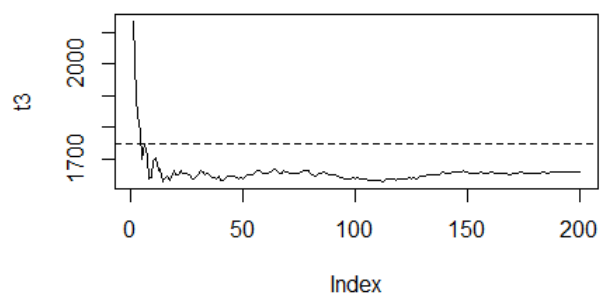
執行隨機抽取 15 個的 `law.bs()` 結果如下：



執行隨機抽取 20 個的 `law.bs()` 結果如下：



執行隨機抽取 25 個的 `law.bs()` 結果如下：



發現隨著執行 50, 100, ..., 10000 次的拔靴法重複數增加，會增加估計的精準度；而抽取的數量增加也會增加精準度。

Code:

```

#HW3
#1#####
library(ISR3)
x1 <- c(rep(1300,6),rep(1200,6), rep(1100,4))
x2 <- c(7.5, 9.0, 11.0, 13.5, 17.0,
        23.0, 5.3, 7.5, 11.0, 13.5,
        17.0, 23.0, 5.3, 7.5, 11.0,
        17.0)
x3 <- c(0.012, 0.012, 0.0115, 0.013, 0.0135,
        0.012, 0.040, 0.038, 0.032, 0.026,
        0.034, 0.041, 0.084, 0.098, 0.092,
        0.086)
y <- c(49.0, 50.2, 50.5, 48.5, 47.5,
        44.5, 28.0, 31.5, 34.5, 35.0,
        38.0, 38.5, 15.0, 17.0, 20.5,
        29.5)

x4 <- x1 * x1
x5 <- x2 * x2
x6 <- x3 * x3
x7 <- x1 * x2
x8 <- x1 * x3
x9 <- x2 * x3

X <- matrix(c(rep(1,16), x1, x2, x3, x4, x5, x6, x7, x8, x9), ncol =
10)

#Cholesky
A <- t(X) %*% X
L <- t(chol(A))
B <- solve(t(L), solve(L, t(X) %*% y))
round(B, 2)

#SWEEP
M <- matrix(rbind(cbind(A, t(X) %*% y), cbind(t(y) %*% X, t(y) %*%
y)), nrow = 11)
M2 <- SWP(M, 1:10)
M2

XX <- -M2[1:10,1:10]
round(sqrt(diag(XX * (M2[11,11]/6))) , 2)

g <- lm(y~x1+x2+x3+x4+x5+x6+x7+x8+x9)
summary(g)

#2#####
plot(lynx, type= 'l', main = 'Time series plot of Lynx')
ar.ols(lynx, order = 1)
ar.ols(lynx, order = 2)

#3#####

```

```

library(forecast)
mortality <- read.csv("StatisticalSimulation/malemortality.csv", h =
T)
x <- scale(log(mortality[mortality$year <= 99, -1]), scale = FALSE)
svd.x <- svd(x, 1, 1)
b <- svd.x$v
k <- svd.x$u * svd.x$d[1]
fit.k <- ar.ols(k, order = 1)
pred.k <- NULL
p.k <- k[19]
for(i in 1:5){
  p.k <- fit.k$x.intercept + fit.k$ar * p.k
  pred.k <- c(pred.k, p.k)
}
xnew.appr <- pred.k %*% t(b)
round(xnew.appr, 2)
xnew <- scale(log(mortality[mortality$year > 99, -1]), scale = FALSE)
sum((xnew - xnew.appr)^2)

pca.x <- princomp(x)

#4#####
#(a)
library(gtools)
DDT <- c(65, 98, 117, 122, 130)
egg <- c(0.52, 0.53, 0.49, 0.49, 0.37)
p <- permutations(5,5)
DDT2 <- matrix(DDT[t(p)], byrow = T ,ncol = 5)
length(which(DDT2 %*% sort(egg) <= sum(DDT * egg)))/nrow(DDT2)
cor.test(DDT, egg, method = "pearson")
cor.test(DDT, egg, method = "spearman")

x <- c(585, 1002, 472, 493, 408, 690, 291)
y <- c(0.1, 0.2, 0.5, 1, 1.5, 2, 3)
p2 <- permutations(7,7)
x2 <- matrix(x[t(p2)], byrow = T, ncol = 7)
length(which(x2 %*% sort(y) <= sum(x * y)))/nrow(x2)
cor.test(x, y, method="pearson")
cor.test(x, y, method="spearman")

#(b)
choln2u <- function(rho = 0.2){
  A <- matrix(c(1, rho, rho, 1), ncol=2)
  B <- t(chol(A))
  t1 <- NULL
  t2 <- NULL
  t3 <- NULL
  for (i in 1:100) {
    x1 <- rnorm(10)
    y1 <- rnorm(10)
    xy1 <- rbind(x1, y1)
  }
}

```

```

xy <- B %*% xy1
xy.unif <- floor(10*pnorm(xy))
z1 <- cor.test(xy.unif[1, ], xy.unif[2, ], method =
"pearson")$p.value
z2 <- cor.test(xy.unif[1, ], xy.unif[2, ], method =
"spearman")$p.value
t1 <- c(t1, z1)
t2 <- c(t2, z2)
t <- NULL
z0 <- sum(x1*y1)
for (j in 1:1000) {
  x0 <- sample(x1, 10, F)
  y0 <- sample(y1, 10, F)
  t <- c(t, sum(x0*y0))
}
z3 <- sum(t >= z0)/1000
t3 <- c(t3, z3)
}
print(paste('pearson p<0.05 個數:', length(t1[t1<0.05])))
print(paste('spearman p<0.05 個數:', length(t2[t2<0.05])))
print(paste('permutation test p<0.05 個數:', length(t2[t2<0.05])))
}

choln2u(0.2)
choln2u(0.8)

#5#####
MWWTest = function(seq1, seq2){
  seq = c(seq1, seq2)
  n1 = length(seq1)
  n2 = length(seq2)

  r1 = rank(seq)[1:n1]
  r2 = rank(seq)[-c(1:n1)]

  W1 = sum(r1)
  W2 = sum(r2)

  U1 = n1 * n2 + (n1 * (n1 + 1) / 2) - W1
  U2 = n1 * n2 + (n2 * (n2 + 1) / 2) - W2

  U = min(U1, U2)
  Z = (U - n1 * n2 / 2) / sqrt(n1 * n2 * (n1 + n2 + 1) / 12)
  df = data.frame(U1, U2)
  return (U1)
}

data = NULL
for (i in 2:10){
  for (j in 2:10){
    x = NULL
    for (k in 1:100){
      y = MWWTest(runif(i), runif(j))

```

```
x = c(x, y)
}
data = cbind(data, sort(as.numeric(x)))
}
}

df = data.frame(data, row.names = c(1:100))
critical = df[500, ]
cri_mat = matrix(critical, ncol = 9, byrow = TRUE)
cri_df = data.frame(cri_mat, row.names = c(2:10))
colnames(cri_df) = c(2:10)
print(cri_df)

#6#####
library(bootstrap)
law.bs <- function(x){
  t1 <- NULL
  t2 <- NULL
  t3 <- NULL
  t4 <- NULL
  for(j in 1:200){
    k <- 50*j
    for (i in k) {
      x1 <- sample(1:15, x, T)
      x2 <- law[x1,]
      x3 <- array(apply(x2, 2, var))
      t1 <- c(t1, x3[1])
      t2 <- c(t2, x3[2])
    }
    t3 <- c(t3, mean(t1))
    t4 <- c(t4, mean(t2))
  }
  par(mfrow=c(1,2))
  p1 <- rbind(plot(t3, type = 'l'), abline(h = var(law$LSAT), lty =
2))
  p2 <- rbind(plot(t4, type = 'l'), abline(h = var(law$GPA), lty = 2))
  return(list(p1, p2))
}

law.bs(10)
law.bs(15)
law.bs(20)
law.bs(25)
```