Practical Problem Set 2

Preet Paul

2024-08-25

Problem 1

Loading the dataset

```
nutrition <- read.csv("C:\\Users\\PREET PAUL\\Desktop\\Presidency University M.Sc. Notes\\3rd Se
mester\\nutrient_data.csv")
head(nutrition)
```

```
##   id calcium   iron protein      a       c
## 1  1  522.29 10.188  42.561 349.13  54.141
## 2  2  343.32  4.113  67.793 266.99  24.839
## 3  3  858.26 13.741  59.933 667.90 155.455
## 4  4  575.98 13.245  42.215 792.23 224.688
## 5  5 1927.50 18.919 111.316 740.27  80.961
## 6  6  607.58  6.800  45.785 165.68  13.050
```

```
View(nutrition)
```

```
# Eliminating the index column from our dataset

nutrition <- nutrition[,-c(1)]
```

```
# Calculating the mean of each columns in the dataset

avg = apply(nutrition, MARGIN = 2, FUN = mean)
avg
```

```
##   calcium      iron   protein         a         c
## 624.04925  11.12990  65.80344 839.63535  78.92845
```

```
S = var(nutrition)  ## Sample Var-Cov matrix
S
```

```
##             calcium       iron   protein           a          c
## calcium 157829.4439  940.08944 6075.8163  102411.127  6701.6160
## iron       940.0894   35.81054  114.0580    2383.153   137.6720
## protein   6075.8163  114.05803  934.8769    7330.052   477.1998
## a       102411.1266 2383.15341 7330.0515 2668452.371 22063.2486
## c         6701.6160  137.67199  477.1998   22063.249  5416.2641
```

```
mu = c(1000,15,60,800,75)  ## Specified value of mean  under null
```

Loading the package “ICSNP”

```
library(ICSNP)
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: ICS
```

- Thus, we see that as the p-value for the test is very small (< 2.2e-16), H0 is rejected at 1% level of significance.

Finding the Simultaneous confidence intervals for the mean intake

```
n = nrow(nutrition)
p = ncol(nutrition)
attach(nutrition)

# Simultaneous confidence interval for calcium
# Here qf() stands for the quantile function of F-distribution

ci.calcium = c(mean(calcium)-(sqrt(p*(n-1)*var(calcium))*qf(0.99, p, (n-p))/sqrt(n*(n-p))),
               mean(calcium)+(sqrt(p*(n-1)*var(calcium))*qf(0.99, p, (n-p))/sqrt(n*(n-p))))

# Simultaneous confidence interval for iron
ci.iron = c(mean(iron)-(sqrt(p*(n-1)*var(iron))*qf(0.99, p, (n-p))/sqrt(n*(n-p))),
            mean(iron)+(sqrt(p*(n-1)*var(iron))*qf(0.99, p, (n-p))/sqrt(n*(n-p))))

# Simultaneous confidence interval for protein
ci.protein = c(mean(protein)-(sqrt(p*(n-1)*var(protein))*qf(0.99, p, (n-p))/sqrt(n*(n-p))),
               mean(protein)+(sqrt(p*(n-1)*var(protein))*qf(0.99, p, (n-p))/sqrt(n*(n-p))))

# Simultaneous confidence interval for Vitamin A
ci.a = c(mean(nutrition$a)-(sqrt(p*(n-1)*var(nutrition$a))*qf(0.99, p, (n-p))/sqrt(n*(n-p))),
         mean(nutrition$a)+(sqrt(p*(n-1)*var(nutrition$a))*qf(0.99, p, (n-p))/sqrt(n*(n-p))))

# Simultaneous confidence interval for Vitamin C
ci.c = c(mean(c)-(sqrt(p*(n-1)*var(c))*qf(0.99, p, (n-p))/sqrt(n*(n-p))),
         mean(c)+(sqrt(p*(n-1)*var(c))*qf(0.99, p, (n-p))/sqrt(n*(n-p))))

# Showing the all the outputs as a list

confidence <- list(ci.calcium,ci.iron,ci.protein,ci.a,ci.c)
names(confidence) <- c("ci.calcium","ci.iron","ci.protein","ci.a","ci.c")
confidence
```

```
## $ci.calcium
## [1] 524.2269 723.8716
##
## $ci.iron
## [1]  9.626276 12.633523
##
## $ci.protein
## [1] 58.12079 73.48609
##
## $ci.a
## [1]  429.1824 1250.0883
##
## $ci.c
## [1] 60.43646 97.42043
```

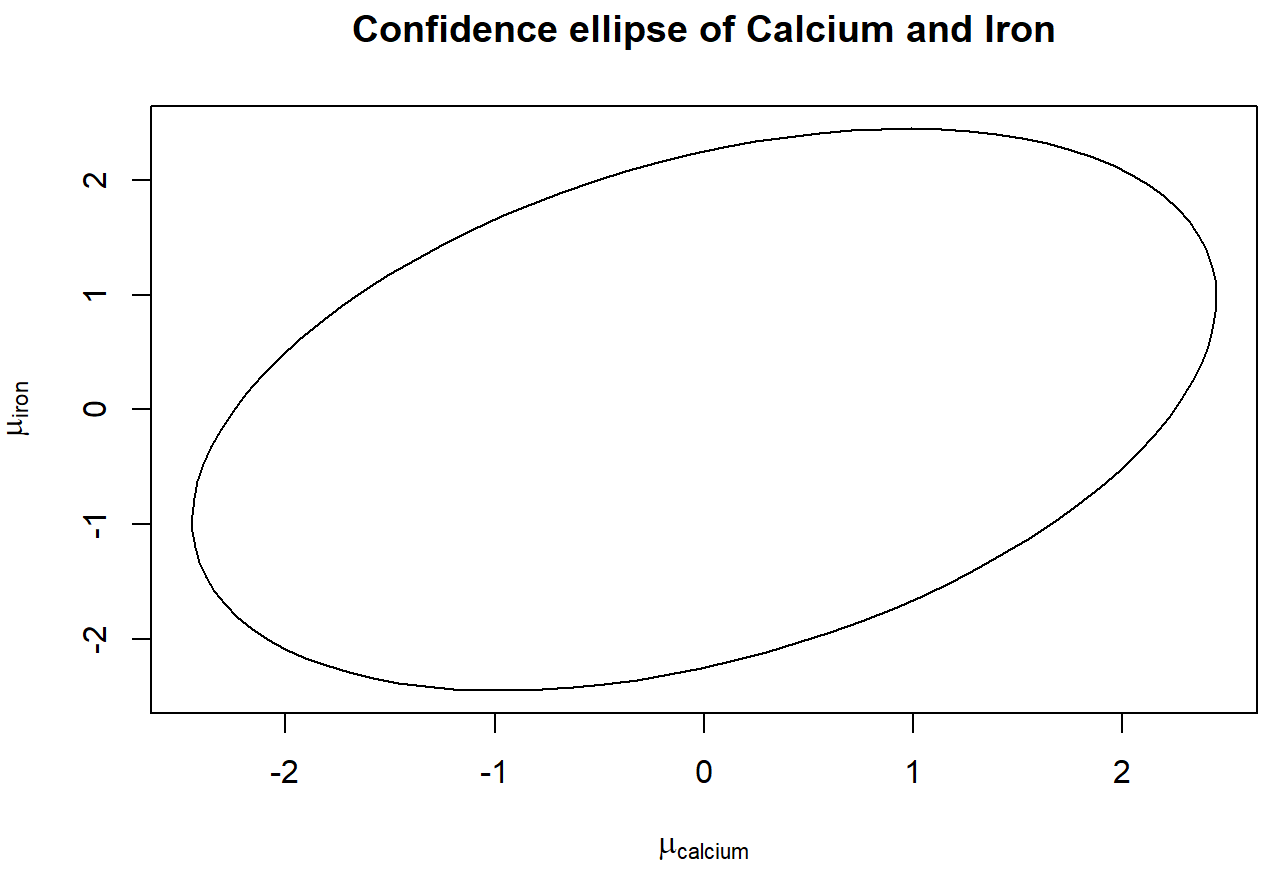
Plot the confidence interval and confidence ellipsoid

```
library(ellipse)
```

```
##
## Attaching package: 'ellipse'
```

```
## The following object is masked from 'package:graphics':
##
##     pairs
```

```
plot(ellipse(cor(calcium,iron)),type="l",xlab=expression(mu[calcium]),
     ylab=expression(mu[iron]),main="Confidence ellipse of Calcium and Iron")
```



Finding the Bonferroni confidence intervals for the mean intake

```
a = 0.01

# Bonferroni confidence interval for calcium
bon.calcium = c(mean(calcium)-(sqrt(var(calcium))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
                mean(calcium)+(sqrt(var(calcium))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

# Bonferroni confidence interval for iron
bon.iron = c(mean(iron)-(sqrt(var(iron))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
             mean(iron)+(sqrt(var(iron))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

# Bonferroni confidence interval for protein
bon.protein = c(mean(protein)-(sqrt(var(protein))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
                mean(protein)+(sqrt(var(protein))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

# Bonferroni confidence interval for Vitamin A
bon.a = c(mean(nutrition$a)-(sqrt(var(nutrition$a))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
          mean(nutrition$a)+(sqrt(var(nutrition$a))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

# Bonferroni confidence interval for Vitamin C
bon.c = c(mean(c)-(sqrt(var(c))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
          mean(c)+(sqrt(var(c))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

# Showing the all the outputs as a list

bonferroni.confidence <- list(bon.calcium,bon.iron,bon.protein,bon.a,bon.c)
names(bonferroni.confidence) <- c("bon.calcium","bon.iron","bon.protein","bon.a","bon.c")
bonferroni.confidence
```

```
## $bon.calcium
## [1] 578.6645 669.4340
##
## $bon.iron
## [1] 10.44627 11.81353
##
## $bon.protein
## [1] 62.31048 69.29640
##
## $bon.a
## [1]  653.0207 1026.2500
##
## $bon.c
## [1] 70.52097 87.33593
```

Generating Profile Plots

Loading the package “plotrix”

```
library(plotrix)
```

```
# Standardizing each of the observations by dividing them by their hypothesized means

Z <- nutrition/mu
View(Z)

# Finding the means of the standardized variables
avg_new <- apply(Z,2, FUN = mean)
avg_new
```

```
##    calcium       iron    protein          a          c
## 12.2944444  0.2224489  1.2900526 15.1272213  1.5227934
```

Generating Profile Plots for Simultaneous confidence intervals

```
sim_ci <- data.frame(c(ci.calcium[1],ci.calcium[2]),c(ci.iron[1],ci.iron[2]),c(ci.protein[1],ci.
protein[2]),
             c(ci.a[1],ci.a[2]),c(ci.c[1],ci.c[2]))
colnames(sim_ci) <- c("calcium","iron","protein","a","c")
rownames(sim_ci) <- c("lower bound","upper bound")
sim_ci
```

```
##              calcium      iron  protein         a        c
## lower bound 524.2269  9.626276 58.12079  429.1824 60.43646
## upper bound 723.8716 12.633523 73.48609 1250.0883 97.42043
```

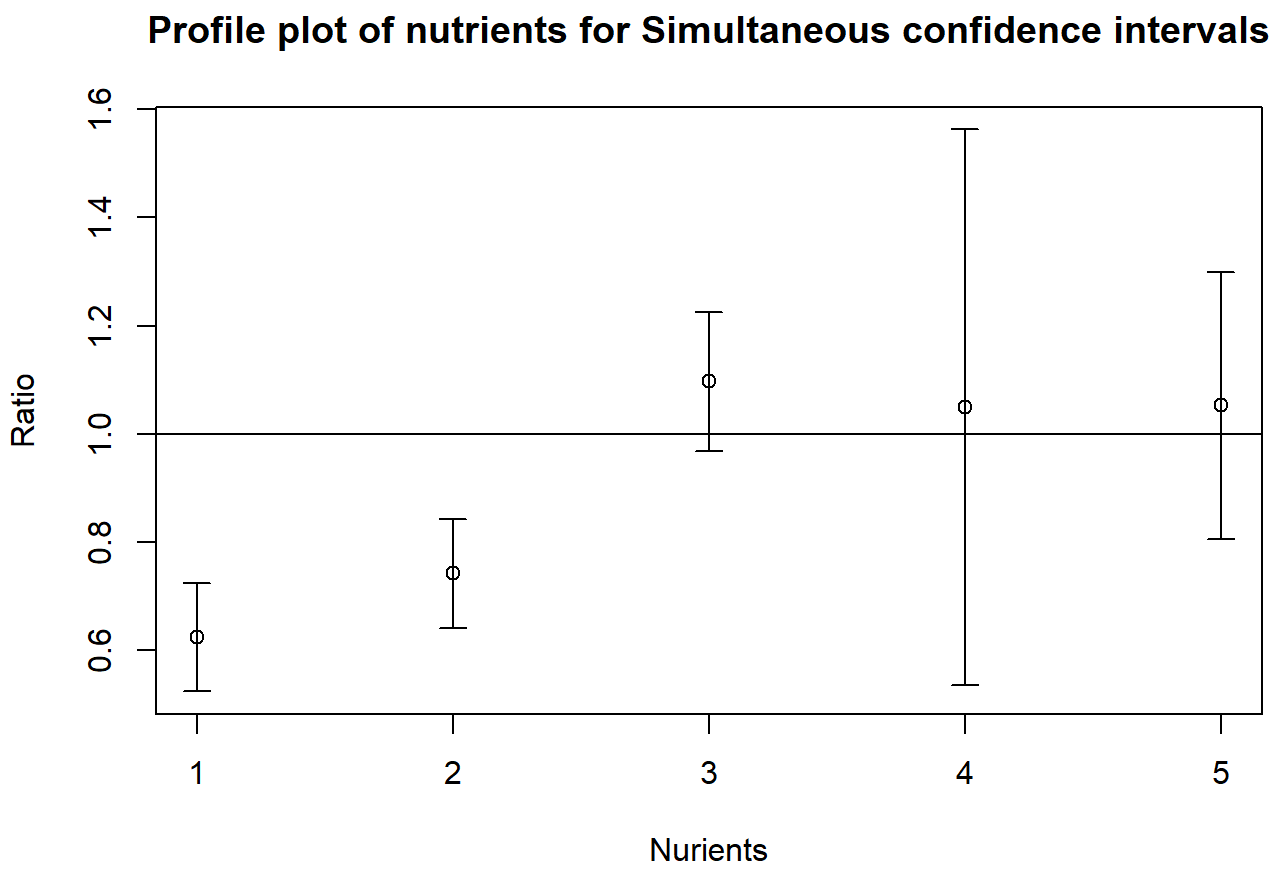
```
# Standardizing the simultaneous confidence intervals

sim_std <- data.frame(sim_ci$calcium/1000,sim_ci$iron/15,sim_ci$protein/60,sim_ci$a/800,sim_ci
$c/75)
colnames(sim_std) <- c("calcium","iron","protein","a","c")
sim_std
```

```
##     calcium      iron   protein        a         c
## 1 0.5242269 0.6417518 0.9686799 0.536478 0.8058195
## 2 0.7238716 0.8422349 1.2247681 1.562610 1.2989391
```

```
Result <- apply(sim_std,2, FUN = mean)
Result
```

```
##   calcium      iron   protein         a         c
## 0.6240493 0.7419933 1.0967240 1.0495442 1.0523793
```

```
#Profile Plot
plotCI(x=c(1:5), y=Result, li=sim_std[1,], ui=sim_std[2,],xlab="Nurients",
       ylab="Ratio",main="Profile plot of nutrients for Simultaneous confidence intervals")
abline(h=1)
```



Generating Profile Plots for Bonferroni confidence intervals

```
bon_ci <- data.frame(c(bon.calcium[1],bon.calcium[2]),c(bon.iron[1],bon.iron[2]),c(bon.protein
[1],bon.protein[2]),
                     c(bon.a[1],bon.a[2]),c(bon.c[1],bon.c[2]))
colnames(bon_ci) <- c("calcium","iron","protein","a","c")
rownames(bon_ci) <- c("lower bound","upper bound")
bon_ci
```

```
##              calcium     iron  protein         a        c
## lower bound 578.6645 10.44627 62.31048  653.0207 70.52097
## upper bound 669.4340 11.81353 69.29640 1026.2500 87.33593
```

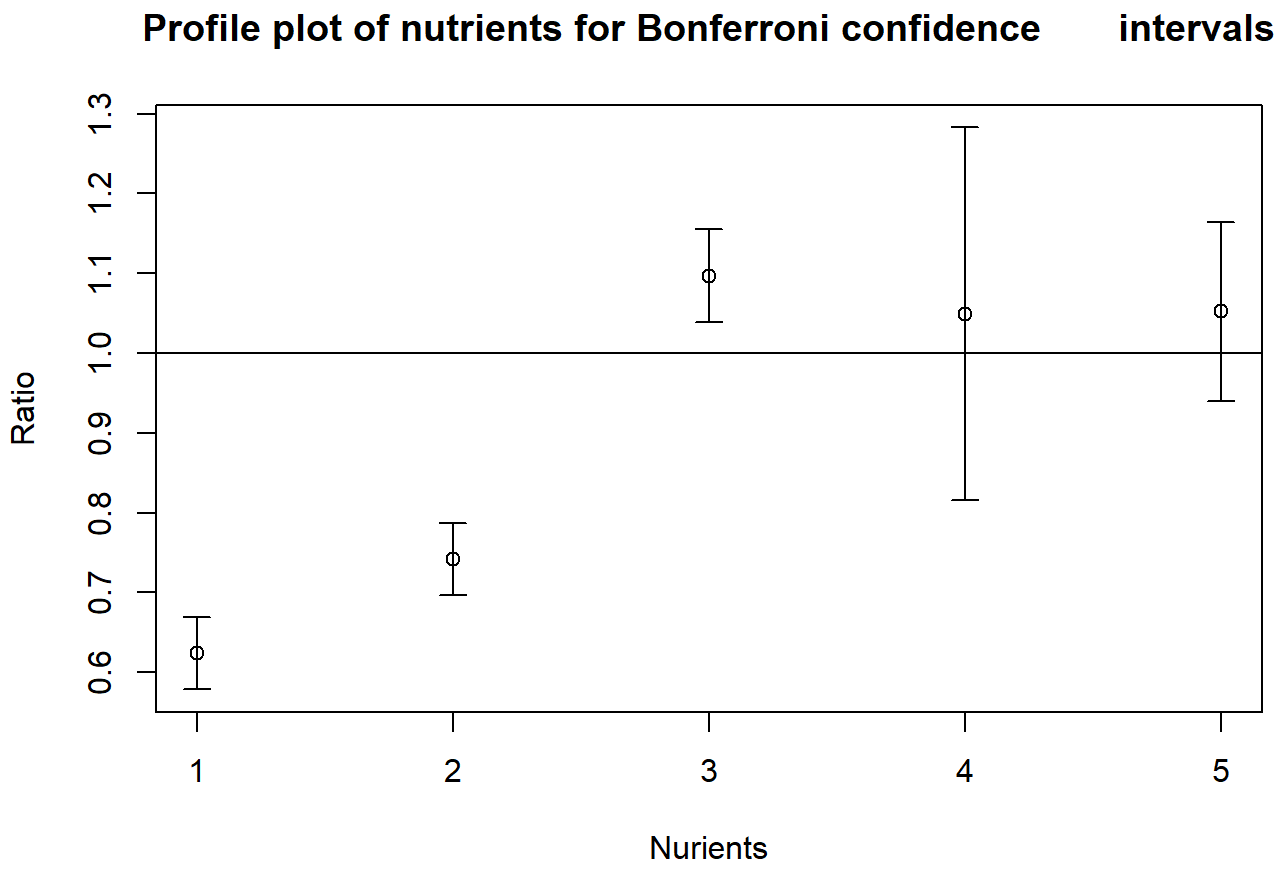
```
# Standardizing the simultaneous confidence intervals

bon_std <- data.frame(bon_ci$calcium/1000,bon_ci$iron/15,bon_ci$protein/60,bon_ci$a/800,bon_ci
$c/75)
colnames(bon_std) <- c("calcium","iron","protein","a","c")
bon_std
```

```
##     calcium      iron  protein         a         c
## 1 0.5786645 0.6964179 1.038508 0.8162759 0.9402795
## 2 0.6694340 0.7875687 1.154940 1.2828125 1.1644790
```

```
Result1 <- apply(bon_std,2, FUN = mean)
Result1
```

```
##   calcium      iron   protein         a         c
## 0.6240493 0.7419933 1.0967240 1.0495442 1.0523793
```

```
#Profile Plot
plotCI(x=c(1:5), y=Result1, li=bon_std[1,], ui=bon_std[2,],xlab="Nurients",
       ylab="Ratio",main="Profile plot of nutrients for Bonferroni confidence       intervals")
abline(h=1)
```



Problem 2

Loading the dataset

```
shoe <- read.csv("C:\\Users\\PREET PAUL\\Desktop\\Presidency University M.Sc. Notes\\3rd Semeste
r\\shoe.csv")
head(shoe)
```

```
##         X Model.1     X.1       X.2     X.3        X.4 X.5 Model.2     X.6
## 1                                                        NA
## 2 Subject   Style Comfort Stability Cushion Durability  NA   Style Comfort
## 3       1       6       8         3       5         19  NA       8       6
## 4       2       6       7         3       4          9  NA       8       6
## 5       3       5       7         1       4         16  NA       7       5
## 6       4      10       9         8       4          4  NA       9       8
##         X.7     X.8        X.9
## 1
## 2 Stability Cushion Durability
## 3         5       6         10
## 4         3       6          4
## 5         6       4         17
## 6         6       3          4
```

```
View(shoe)
which(is.na(shoe))  #Checking for missing values
```

```
##  [1] 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181
## [20] 182 183 184 185 186 187 188 189
```

```
shoe <- shoe[-c(1),]
colnames(shoe) <- shoe[c(1),]  # Giving column names to the dataset
shoe <- shoe[-c(1),-c(1)]
View(shoe)
```

Subsetting Model1 and Model2 from the dataset

```
Model1 <- data.matrix(shoe[,c(1:5)])
View(Model1)
Model2 <- data.matrix(shoe[,c(7:11)])
View(Model2)

which(is.na(Model1))
```

```
## integer(0)
```

```
which(is.na(Model2))
```

```
## integer(0)
```

Part (a)

```
Y <- Model1 - Model2

# Calculating the means of the variables of Y
Y_avg <- apply(Y,2, FUN = mean)
Y_avg
```

```
##      Style    Comfort  Stability    Cushion Durability
##       1.52       0.76       1.16       1.28       2.04
```

Performing Hotelling's T^2 test

```
HotellingsT2(Y, test="f")
```

```
##
##  Hotelling's one sample T2-test
##
## data:  Y
## T.2 = 3.737, df1 = 5, df2 = 20, p-value = 0.01497
## alternative hypothesis: true location is not equal to c(0,0,0,0,0)
```

- Since the p-value is 0.01497, we reject our null hypothesis H0 at 5% level of significance. Thus, we can conclude that there is a significant difference between two show models at 5% level of significance.

Calculating Simultaneous confidence intervals

```
Y <- as.data.frame(Y)
n = nrow(Y)
p = ncol(Y)
f.value <- qf(0.95, p, (n-p))
a = 0.05

ci.style <- c(mean(Y$Style)-(sqrt(p*(n-1)*var(Y$Style))*qf((1-a), p, (n-p))/sqrt(n*(n-p))),
              mean(Y$Style)+(sqrt(p*(n-1)*var(Y$Style))*qf((1-a), p, (n-p))/sqrt(n*(n-p))))

ci.comfort <- c(mean(Y$Comfort)-(sqrt(p*(n-1)*var(Y$Comfort))*qf((1-a), p, (n-p))/sqrt(n*(n-
p))),
                mean(Y$Comfort)+(sqrt(p*(n-1)*var(Y$Comfort))*qf((1-a), p, (n-p))/sqrt(n*(n-
p))))

ci.stability <- c(mean(Y$Stability)-(sqrt(p*(n-1)*var(Y$Stability))*qf((1-a), p, (n-p))/sqrt(n*
(n-p))),
                  mean(Y$Stability)+(sqrt(p*(n-1)*var(Y$Stability))*qf((1-a), p, (n-p))/sqrt(n*
(n-p))))

ci.cushion <- c(mean(Y$Cushion)-(sqrt(p*(n-1)*var(Y$Cushion))*qf((1-a), p, (n-p))/sqrt(n*(n-
p))),
                mean(Y$Cushion)+(sqrt(p*(n-1)*var(Y$Cushion))*qf((1-a), p, (n-p))/sqrt(n*(n-
p))))

ci.durability <- c(mean(Y$Durability)-(sqrt(p*(n-1)*var(Y$Durability))*qf((1-a), p, (n-p))/sqrt
(n*(n-p))),
                   mean(Y$Durability)+(sqrt(p*(n-1)*var(Y$Durability))*qf((1-a), p, (n-p))/sqrt
(n*(n-p))))


# Showing the all the outputs as a list

confidence1 <- list(ci.style,ci.comfort,ci.stability,ci.cushion,ci.durability)
names(confidence1) <- c("ci.style","ci.comfort","ci.stability","ci.cushion","ci.durability")
confidence1
```

```
## $ci.style
## [1] -2.058375  5.098375
##
## $ci.comfort
## [1] -2.289721  3.809721
##
## $ci.stability
## [1] -2.818271  5.138271
##
## $ci.cushion
## [1] -1.893478  4.453478
##
## $ci.durability
## [1] -6.519546 10.599546
```

Calculating Bonferroni confidence intervals

```
bon.style <- c(mean(Y$Style)-(sqrt(var(Y$Style))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
               mean(Y$Style)+(sqrt(var(Y$Style))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

bon.comfort <- c(mean(Y$Comfort)-(sqrt(var(Y$Comfort))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
                 mean(Y$Comfort)+(sqrt(var(Y$Comfort))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

bon.stability <- c(mean(Y$Stability)-(sqrt(var(Y$Stability))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
                   mean(Y$Stability)+(sqrt(var(Y$Stability))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

bon.cushion <- c(mean(Y$Cushion)-(sqrt(var(Y$Cushion))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)),
                 mean(Y$Cushion)+(sqrt(var(Y$Cushion))*qt(1-(a/(2*p)), df=n-1)/sqrt(n)))

bon.durability <- c(mean(Y$Durability)-(sqrt(var(Y$Durability))*qt(1-(a/(2*p)), df=n-1)/sqrt
(n)),
                    mean(Y$Durability)+(sqrt(var(Y$Durability))*qt(1-(a/(2*p)), df=n-1)/sqrt
(n)))

# Showing the all the outputs as a list

bonferroni.confidence1 <- list(bon.style,bon.comfort,bon.stability,bon.cushion,bon.durability)
names(bonferroni.confidence1) <- c("bon.style","bon.comfort","bon.stability","bon.cushion","bon.
durability")
bonferroni.confidence1
```

```
## $bon.style
## [1] 0.01276358 3.02723642
##
## $bon.comfort
## [1] -0.5245638  2.0445638
##
## $bon.stability
## [1] -0.5156757  2.8356757
##
## $bon.cushion
## [1] -0.0566913  2.6166913
##
## $bon.durability
## [1] -1.565341  5.645341
```

Generating Profile Plots for Bonferroni confidence intervals

Loading the package

```
library(plotrix)
```

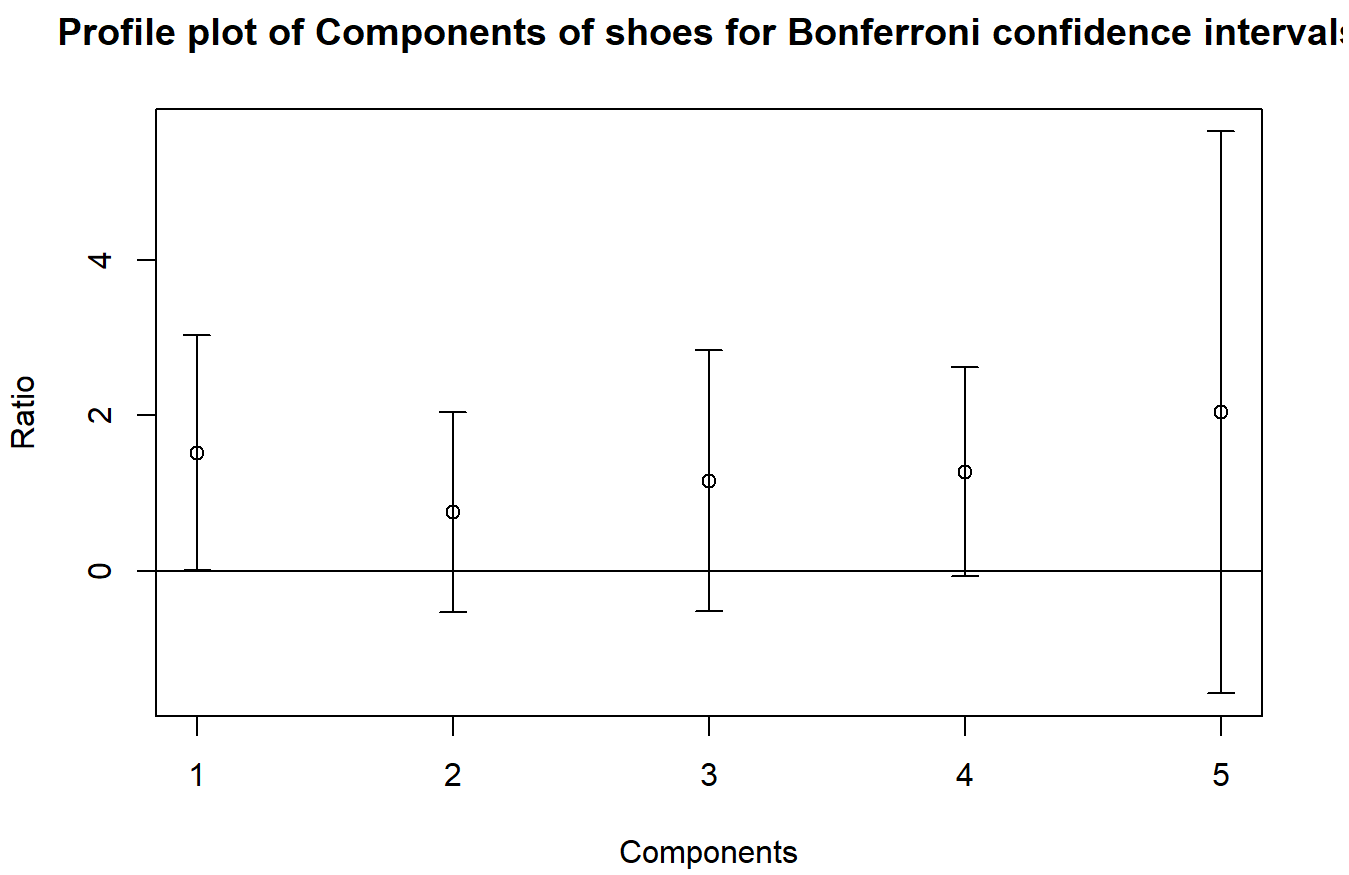
```
bon_ci1 <- data.frame(c(bon.style[1],bon.style[2]),c(bon.comfort[1],bon.comfort[2]),c(bon.stabil
ity[1],bon.stability[2]),
                      c(bon.cushion[1],bon.cushion[2]),c(bon.durability[1],bon.durability[2]))
colnames(bon_ci1) <- c("style","comfort","stability","cushion","durability")
rownames(bon_ci1) <- c("lower bound","upper bound")
bon_ci1
```

```
##                  style    comfort  stability    cushion durability
## lower bound 0.01276358 -0.5245638 -0.5156757 -0.0566913  -1.565341
## upper bound 3.02723642  2.0445638  2.8356757  2.6166913   5.645341
```

```
Result2 <- apply(bon_ci1,2, FUN = mean)
Result2
```

```
##      style    comfort  stability    cushion durability
##       1.52       0.76       1.16       1.28       2.04
```

```
#Profile Plot
plotCI(x=c(1:5), y=Result2, li=bon_ci1[1,], ui=bon_ci1[2,],xlab="Components",
       ylab="Ratio",main="Profile plot of Components of shoes for Bonferroni confidence interval
s")
abline(h=0)
```



Generating Profile Plots for simultaneous confidence intervals

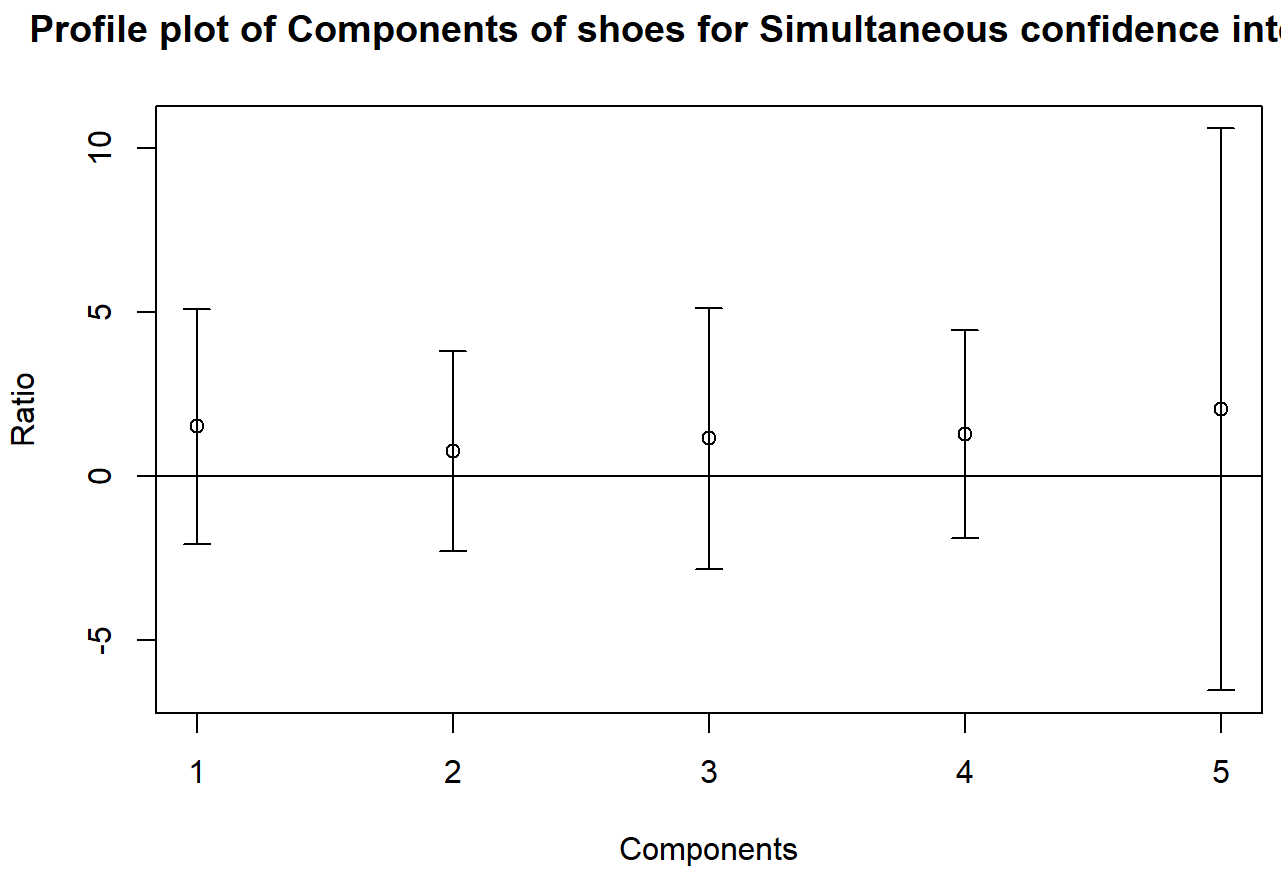
```
sim_ci1 <- data.frame(c(ci.style[1],ci.style[2]),c(ci.comfort[1],ci.comfort[2]),c(ci.stability
[1],ci.stability[2]),
                      c(ci.cushion[1],ci.cushion[2]),c(ci.durability[1],ci.durability[2]))
colnames(sim_ci1) <- c("style","comfort","stability","cushion","durability")
rownames(sim_ci1) <- c("lower bound","upper bound")
sim_ci1
```

```
##                style   comfort stability   cushion durability
## lower bound -2.058375 -2.289721 -2.818271 -1.893478  -6.519546
## upper bound  5.098375  3.809721  5.138271  4.453478  10.599546
```

```
Result3 <- apply(sim_ci1,2, FUN = mean)
Result3
```

```
##      style    comfort  stability    cushion durability
##       1.52       0.76       1.16       1.28       2.04
```

```
#Profile Plot
plotCI(x=c(1:5), y=Result3, li=sim_ci1[1,], ui=sim_ci1[2,],xlab="Components",
       ylab="Ratio",main="Profile plot of Components of shoes for Simultaneous confidence interv
als")
abline(h=0)
```



- So, from the profile plot of bonferroni confidence interval, we see that style has a significant difference.

Problem 3

Loading the dataset

```
data1 <- read.csv("C:\\Users\\PREET PAUL\\Desktop\\Presidency University M.Sc. Notes\\3rd Semest
er\\drug.csv")
head(data1)
```

```
##    Drug        X   X.1 X.2 Placebo      X.3   X.4
## 1                      NA
## 2 Fever Pressure Aches  NA   Fever Pressure Aches
## 3  36.5       72    18  NA    40.9       54    14
## 4  36.6       84    16  NA    39.5       75    18
## 5  38.2       60    29  NA    39.4       57    24
## 6  37.6       82    13  NA    38.2       71    24
```

```
View(data1)

which(is.na(data1))  # Checking for missing values
```

```
##  [1] 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88
```

```
data1 <- data1[-c(1),]  # Eliminating the empty row

colnames(data1) <- data1[c(1),]  # Renaming the columns of the dataset
data1 <- data1[-c(1),]
View(data1)
```

Subsetting the dataset into Drug and Placebo

```
drug <- data1[,c(1:3)]
placebo <- data1[,c(5:7)]
drug
```

```
##    Fever Pressure Aches
## 3   36.5       72    18
## 4   36.6       84    16
## 5   38.2       60    29
## 6   37.6       82    13
## 7     37       68    25
## 8   37.9       54    27
## 9   37.4       80    25
## 10  35.2       99     8
## 11  38.2       65    21
## 12  37.5       55    11
## 13  35.8       70    16
## 14  37.4       76    13
## 15  37.2       49    29
## 16  36.5       59    24
## 17  38.3       77    12
## 18  37.5       66    19
## 19    36       79    14
## 20  36.9       67    12
## 21  39.3       53     7
## 22  38.8       67    13
```

```
View(placebo)
placebo <- placebo[-c(19,20),]
placebo
```

```
##    Fever Pressure Aches
## 3   40.9       54    14
## 4   39.5       75    18
## 5   39.4       57    24
## 6   38.2       71    24
## 7   39.7       65    22
## 8   38.9       49    30
## 9   38.6       58    25
## 10  39.9       52    17
## 11  41.3       62    18
## 12  38.1       57    20
## 13  39.6       78    19
## 14  37.1       92    15
## 15  39.5       63    13
## 16  40.3       52    25
## 17  41.5       46    27
## 18  39.3       56    14
## 19  37.6       86    16
## 20  40.6       48    21
```

```
drug$Fever <- as.numeric(drug$Fever)
drug$Pressure <- as.numeric(drug$Pressure)
drug$Aches <- as.numeric(drug$Aches)
drug
```

```
##    Fever Pressure Aches
## 3   36.5       72    18
## 4   36.6       84    16
## 5   38.2       60    29
## 6   37.6       82    13
## 7   37.0       68    25
## 8   37.9       54    27
## 9   37.4       80    25
## 10  35.2       99     8
## 11  38.2       65    21
## 12  37.5       55    11
## 13  35.8       70    16
## 14  37.4       76    13
## 15  37.2       49    29
## 16  36.5       59    24
## 17  38.3       77    12
## 18  37.5       66    19
## 19  36.0       79    14
## 20  36.9       67    12
## 21  39.3       53     7
## 22  38.8       67    13
```

```
placebo$Fever <- as.numeric(placebo$Fever)
placebo$Pressure <- as.numeric(placebo$Pressure)
placebo$Aches <- as.numeric(placebo$Aches)
placebo
```

```
##    Fever Pressure Aches
## 3   40.9       54    14
## 4   39.5       75    18
## 5   39.4       57    24
## 6   38.2       71    24
## 7   39.7       65    22
## 8   38.9       49    30
## 9   38.6       58    25
## 10  39.9       52    17
## 11  41.3       62    18
## 12  38.1       57    20
## 13  39.6       78    19
## 14  37.1       92    15
## 15  39.5       63    13
## 16  40.3       52    25
## 17  41.5       46    27
## 18  39.3       56    14
## 19  37.6       86    16
## 20  40.6       48    21
```

Part (a) - Population covariance matrices are equal

Performing Hotelling T^2 test

```
test <- HotellingsT2(drug, placebo)
test
```

```
##
##  Hotelling's two sample T2-test
##
## data:  drug and placebo
## T.2 = 14.115, df1 = 3, df2 = 34, p-value = 3.857e-06
## alternative hypothesis: true location difference is not equal to c(0,0,0)
```

- Since the p-value is very small, we reject the null hypothesis H0 at 5% level of significance.

Part (b) - Population covariance matrices are unequal

```
a = 0.05
p = ncol(drug)
n1 = nrow(drug)
n2 = ncol(placebo)

# mean of drug and placebo
x1.bar = apply(drug,2, FUN = mean)
x2.bar = apply(placebo,2, FUN = mean)

# covariance matrices of drug and placebo
S1 = cov(drug)
S2 = cov(placebo)

# pooled covariance matrix
Sp = ((n1 - 1)*S1 + (n2 - 1)*S2)/(n1+n2-2)
```

Obtaining Hotelling T^2 test statistic

```
T2 = t(x1.bar - x2.bar) %*% solve(Sp*((1/n1)+(1/n2))) %*% (x1.bar - x2.bar)
T2
```

```
##          [,1]
## [1,] 13.44912
```

```
F.statistic = T2*(n1+n2-p-1)/(p*(n1+n2-2))
F.statistic  # Obtaining the F-statistic
```

```
##          [,1]
## [1,] 4.056085
```

```
#critical point at 5% level of significance
critical = qf((1-a), p, n1+n2-p-1)
critical
```

```
## [1] 3.12735
```

```
# Rejection of Null hypothesis
F.statistic > critical
```

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

- Hence, we reject the null hypothesis H0 at 5% level of significance
- Therefore, we see that the drug is effective reducing at reducing these three symptoms.

Problem 4

Loading the dataset

```
soil <- read.csv("C:\\Users\\PREET PAUL\\Desktop\\Presidency University M.Sc. Notes\\3rd Semeste
r\\soil.csv")
head(soil)
```

```
##      X yield water herbicide
## 1 loam  76.7  29.5       7.5
## 2 loam  60.5  32.1       6.3
## 3 loam  96.1  40.7       4.2
## 4 loam  88.1  45.1       4.9
## 5 loam  50.2  34.1      11.7
## 6 loam  55.0  31.1       6.9
```

```
View(soil)
which(is.na(soil))  # Checking for missing values
```

```
## integer(0)
```

```
names(soil)[1] <- "Soil.Type"
```

Performing MANOVA on the dataset

```
model <- manova(cbind(yield, water, herbicide)~Soil.Type, data = soil)
model
```

```
## Call:
##    manova(cbind(yield, water, herbicide) ~ Soil.Type, data = soil)
##
## Terms:
##                 Soil.Type Residuals
## yield             911.416  4057.451
## water             121.906  2833.986
## herbicide          32.348   112.436
## Deg. of Freedom         3        28
##
## Residual standard errors: 12.03781 10.06051 2.003891
## Estimated effects may be unbalanced
```

SUmmary of the model

```
summary(model, test="Wilks")   # Wilk's lambda test
```

```
##           Df   Wilks approx F num Df den Df  Pr(>F)
## Soil.Type  3 0.48941    2.405      9 63.428 0.02047 *
## Residuals 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model, test="Pillai")  # The Pillai test
```

```
##           Df Pillai approx F num Df den Df  Pr(>F)
## Soil.Type  3 0.5345   2.0234      9     84 0.04641 *
## Residuals 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model, test="Roy")     # Roy's Union-Intersection test
```

```
##           Df     Roy approx F num Df den Df    Pr(>F)
## Soil.Type  3 0.94364   8.8073      3     28 0.0002844 ***
## Residuals 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model, test="Hotelling-Lawley")  # Lawley-Hotelling test
```

```
##           Df Hotelling-Lawley approx F num Df den Df   Pr(>F)
## Soil.Type  3          0.99464    2.726      9     74 0.008399 **
## Residuals 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- From the above 4 test, we reject the null hypothesis H0 at 5% level of significance.
- Thus, there is significant difference of 4 levels of soil type.

Performing tests on clay and salty groups

```
data2 <- soil[which(soil$Soil.Type == "clay" | soil$Soil.Type == "salty"), ]
data2
```

```
##    Soil.Type yield water herbicide
## 17     salty  62.8  25.9       2.9
## 18     salty  45.0  15.9       1.2
## 19     salty  47.8  36.1       4.1
## 20     salty  75.6  27.7       6.3
## 21     salty  46.6  46.9       3.6
## 22     salty  50.6  29.7       4.7
## 23     salty  45.7  27.6       6.2
## 24     salty  68.4  35.3       1.9
## 25      clay  52.5  39.0       3.1
## 26      clay  80.0  54.2       4.0
## 27      clay  54.7  32.1       5.7
## 28      clay  63.5  25.6       3.0
## 29      clay  46.3  31.8       7.4
## 30      clay  61.5  16.8       1.9
## 31      clay  62.9  25.8       2.4
## 32      clay  49.3  39.4       5.2
```

```
model2 <- manova(cbind(yield,water,herbicide) ~ Soil.Type, data = data2)
model2
```

```
## Call:
##    manova(cbind(yield, water, herbicide) ~ Soil.Type, data = data2)
##
## Terms:
##                 Soil.Type Residuals
## yield             49.7025 1790.8475
## water              24.010  1472.047
## herbicide          0.2025   48.7075
## Deg. of Freedom         1        14
##
## Residual standard errors: 11.31007 10.25408 1.865236
## Estimated effects may be unbalanced
```

Performing the tests

```
summary(model2, test="Wilks")   # Wilk's lambda test
```

```
##           Df   Wilks approx F num Df den Df Pr(>F)
## Soil.Type  1 0.95813  0.17478      3     12 0.9114
## Residuals 14
```

```
summary(model2, test="Pillai")  # The Pillai test
```

```
##           Df   Pillai approx F num Df den Df Pr(>F)
## Soil.Type  1 0.041866  0.17478      3     12 0.9114
## Residuals 14
```

```
summary(model2, test="Roy")     # Roy's Union-Intersection test
```

```
##           Df      Roy approx F num Df den Df Pr(>F)
## Soil.Type  1 0.043696  0.17478      3     12 0.9114
## Residuals 14
```

```
summary(model2, test="Hotelling-Lawley")  # Lawley-Hotelling test
```

```
##           Df Hotelling-Lawley approx F num Df den Df Pr(>F)
## Soil.Type  1         0.043696  0.17478      3     12 0.9114
## Residuals 14
```

- From the above, we can clearly see that there is no significant difference between clay and salty groups at 5% level of significance.

Performing tests on loam and sandy groups

```
data3 <- soil[which(soil$Soil.Type == "loam" | soil$Soil.Type == "sandy"), ]
data3
```

```
##    Soil.Type yield water herbicide
## 1       loam  76.7  29.5       7.5
## 2       loam  60.5  32.1       6.3
## 3       loam  96.1  40.7       4.2
## 4       loam  88.1  45.1       4.9
## 5       loam  50.2  34.1      11.7
## 6       loam  55.0  31.1       6.9
## 7       loam  65.4  21.6       4.3
## 8       loam  65.7  27.7       5.3
## 9      sandy  67.3  48.3       5.5
## 10     sandy  61.3  28.9       6.9
## 11     sandy  58.2  42.5       4.8
## 12     sandy  76.9  20.4       3.0
## 13     sandy  66.9  23.9       1.1
## 14     sandy  55.4  29.1       5.0
## 15     sandy  50.5  18.0       4.8
## 16     sandy  64.1  14.5       3.7
```

```
model3 <- manova(cbind(yield,water,herbicide) ~ Soil.Type, data = data3)
model3
```

```
## Call:
##    manova(cbind(yield, water, herbicide) ~ Soil.Type, data = data3)
##
## Terms:
##                 Soil.Type Residuals
## yield            203.7756 2266.6038
## water             82.3556 1361.9387
## herbicide         16.6056   63.7287
## Deg. of Freedom         1        14
##
## Residual standard errors: 12.724 9.86313 2.133554
## Estimated effects may be unbalanced
```

Performing the tests

```
summary(model3, test="Wilks")   # Wilk's lambda test
```

```
##           Df   Wilks approx F num Df den Df  Pr(>F)
## Soil.Type  1 0.54421   3.3501      3     12 0.05554 .
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model3, test="Pillai")  # The Pillai test
```

```
##           Df  Pillai approx F num Df den Df  Pr(>F)
## Soil.Type  1 0.45579   3.3501      3     12 0.05554 .
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model3, test="Roy")     # Roy's Union-Intersection test
```

```
##           Df     Roy approx F num Df den Df  Pr(>F)
## Soil.Type  1 0.83753   3.3501      3     12 0.05554 .
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model3, test="Hotelling-Lawley")  # Lawley-Hotelling test
```

```
##           Df Hotelling-Lawley approx F num Df den Df  Pr(>F)
## Soil.Type  1          0.83753   3.3501      3     12 0.05554 .
## Residuals 14
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Hence, at 5% level of significance, we accept the null hyppthesis H0.
- Therefore, the difference between loam and sandy groups is insignificant.

Performing tests for clay and salty vs. loam and sandy

```
data4 <- soil
data4[which(soil$Soil.Type == "loam" | soil$Soil.Type == "sandy"),1] <- "Group1"
data4[which(soil$Soil.Type == "clay" | soil$Soil.Type == "salty"),1] <- "Group2"


model4 <- manova(cbind(yield,water,herbicide) ~ Soil.Type, data = data4)
model4
```

```
## Call:
##    manova(cbind(yield, water, herbicide) ~ Soil.Type, data = data4)
##
## Terms:
##                 Soil.Type Residuals
## yield             657.938  4310.929
## water              15.540  2940.352
## herbicide          15.540   129.244
## Deg. of Freedom         1        30
##
## Residual standard errors: 11.9874 9.900087 2.075607
## Estimated effects may be unbalanced
```

Performing the tests

```
summary(model4, test="Wilks")   # Wilk's lambda test
```

```
##           Df   Wilks approx F num Df den Df   Pr(>F)
## Soil.Type  1 0.65944   4.8201      3     28 0.007893 **
## Residuals 30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model4, test="Pillai")  # The Pillai test
```

```
##           Df  Pillai approx F num Df den Df   Pr(>F)
## Soil.Type  1 0.34056   4.8201      3     28 0.007893 **
## Residuals 30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model4, test="Roy")     # Roy's Union-Intersection test
```

```
##           Df     Roy approx F num Df den Df   Pr(>F)
## Soil.Type  1 0.51644   4.8201      3     28 0.007893 **
## Residuals 30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model4, test="Hotelling-Lawley")  # Lawley-Hotelling test
```

```
##           Df Hotelling-Lawley approx F num Df den Df   Pr(>F)
## Soil.Type  1          0.51644   4.8201      3     28 0.007893 **
## Residuals 30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- From the above, we can clearly see that there is significant difference between clay and salty groups vs. loam and sandy groups

Finding 95% Simultaneous & Bonferroni CI for the individual means

```
Group1 <- data.matrix(data4[1:16,-c(1)])
Group2 <- data.matrix(data4[17:32,-c(1)])

Y = Group1 - Group2
Y.bar = apply(Y,2, FUN = mean)
Y.bar
```

```
##     yield     water herbicide
##   9.06875  -1.39375   1.39375
```

```
S <- cov(Y)
p <- ncol(Y)
n <- nrow(Y)
alpha <- 0.05
```

Calculating Simultaneous CI

```
MargSC <- sqrt(p * (n-1) * qf((1-alpha),p,(n-p))/(n-p))*sqrt(diag(S)/n)
MargSC <- as.matrix(MargSC)
MargSC
```

```
##                [,1]
## yield     13.361451
## water     11.274499
## herbicide  2.942108
```

```
SCI <- data.frame("lower bound" = as.vector(Y.bar - MargSC),
                  "upper bound" = as.vector(Y.bar - MargSC))
rownames(SCI) <- c("yield","water","herbicide")
SCI
```

```
##           lower.bound upper.bound
## yield       -4.292701   -4.292701
## water      -12.668249  -12.668249
## herbicide   -1.548358   -1.548358
```

Calculating Bonferroni Interval

```
MargBI <- qt(1 - (alpha /(2*p)), (n-1))*sqrt(diag(S)/n)
MargBI <- as.matrix(MargBI)
MargBI
```

```
##               [,1]
## yield     10.47523
## water      8.83908
## herbicide  2.30658
```

```
BI <- data.frame("lower bound" = as.vector(Y.bar - MargBI),
                  "upper bound" = as.vector(Y.bar + MargBI))
rownames(BI) <- c("yield","water","herbicide")
BI
```

```
##           lower.bound upper.bound
## yield      -1.4064768    19.54398
## water     -10.2328301     7.44533
## herbicide  -0.9128298     3.70033
```

Problem 5

Loading the dataset

```
study <- read.csv("C:\\Users\\PREET PAUL\\Desktop\\Presidency University M.Sc. Notes\\3rd Semest
er\\study.csv")
head(study)
```

```
##   gender economic kindness optimism
## 1   male  wealthy        5        3
## 2   male  wealthy        4        6
## 3   male  wealthy        3        4
## 4   male  wealthy        2        4
## 5   male   middle        4        6
## 6   male   middle        3        6
```

```
View(study)
```

Performing two-way MANOVA

```
model5 <- manova(cbind(kindness,optimism) ~ gender + economic + gender*economic, data=study)
model5
```

```
## Call:
##    manova(cbind(kindness, optimism) ~ gender + economic + gender *
##     economic, data = study)
##
## Terms:
##                   gender economic gender:economic Residuals
## kindness        12.04167 28.58333        11.08333  53.25000
## optimism        22.04167 23.08333        36.08333  33.75000
## Deg. of Freedom        1        2               2        18
##
## Residual standard errors: 1.719981 1.369306
## Estimated effects may be unbalanced
```

Performing the tests

```
summary(model5, test="Wilks")   # Wilk's lambda test
```

```
##                 Df   Wilks approx F num Df den Df   Pr(>F)
## gender           1 0.58825   5.9496      2     17 0.010997 *
## economic         2 0.50412   3.4716      4     34 0.017562 *
## gender:economic  2 0.38703   5.1630      4     34 0.002325 **
## Residuals       18
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model5, test="Pillai")  # The Pillai test
```

```
##                 Df  Pillai approx F num Df den Df   Pr(>F)
## gender           1 0.41175   5.9496      2     17 0.010997 *
## economic         2 0.51728   3.1399      4     36 0.025881 *
## gender:economic  2 0.70379   4.8866      4     36 0.002985 **
## Residuals       18
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model5, test="Roy")     # Roy's Union-Intersection test
```

```
##                 Df     Roy approx F num Df den Df   Pr(>F)
## gender           1 0.69995   5.9496      2     17 0.010997 *
## economic         2 0.89368   8.0431      2     18 0.003193 **
## gender:economic  2 1.14396  10.2957      2     18 0.001045 **
## Residuals       18
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(model5, test="Hotelling-Lawley")  # Lawley-Hotelling test
```

```
##                 Df Hotelling-Lawley approx F num Df den Df   Pr(>F)
## gender           1          0.69995   5.9496      2     17 0.010997 *
## economic         2          0.94119   3.7648      4     32 0.012790 *
## gender:economic  2          1.34909   5.3964      4     32 0.001948 **
## Residuals       18
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Hence, we can clearly see that the mean effects as well as the interaction effects differs significantly at 5% level of significance.