Sungwon Lee
9129780026

Exercise 1.

Control :	4.302	4.017	4.049	4.176		
P_1 :	2.021	3.19	3.25	3.276	3.292	3.267
P_2 :	3.397	3.552	3.63	3.578	3.612	

a) Refer to Appendix # 1.
p-value = 0.0001 (by permuting 10000 times)
Since our p-value is less than α, we reject the null hypothesis and conclude that the H_a is true. for $\alpha = 0.05$

b) The usual one way ANOVA F-test : p-value = 0.00114.
According to the results above, we can see that both methods give almost same p-value, which is nearly zero. However, we can also notify that since the p-value of permutation F-test is smaller than the one-way ANOVA test, the permutation F-test is more precise compare to the one-way ANOVA Test.

Exercise 2

(Refer to the appendix #2) By using r, when we apply the permutation F-test to the given data, we get the p-value of approximately 0.478. Also, by applying ANOVA F-test, we get the p-value of 0.47. Since the p-value of both methods are greater than the α, we can see that the data is normally distributed.

Exercise 3.

By doing kruskal-Wallis statistic test for data 1, we get the p-value of 0.002055 and X-squared of 12.375. Here, we get the similar result to p-value of permutation F-test and One way ANOVA test. Thus, by applying kruskal-wallis test, we reject the null hypothesis at significance level of 5%.

Exercise 6.

H_0 : The mean ranks of the groups are the same.
H_a : The mean ranks of the groups are different.

By testing for differences among the groups using the kruskal-wallis test, we can obtain $kw = 14.805$, with the degree of freedom as 6. and the p-value is 0.021853. Since the p-value (0.021853) is less than the significance level of 0.05, we can reject the null hypothesis and say that the alternate hypothesis is true. Thus, we can say that the mean ranks of the 7 groups in the data are different.

Exercise 8.

Since the total number of treatments = 7,

$k = 7-1 = 6.$

$df = 10 \cdot 7 - 7 = 63.$

$\therefore$ upper 10% of critical version : $f(0.1, 6, 63) = 3.781$

5% of critical version : $f(0.05, 6, 63) = 4.16.$

```
# Exercise 1

#a)
control = c(4.302, 4.017, 4.049, 4.176)
p1 = c(2.021, 3.19, 3.25, 3.276, 3.292, 3.267)
p2 = c(3.397, 3.552, 3.63, 3.578, 3.612)


treat = c(rep("control",length(control)), rep("p1", length(p1)), rep("p2", length(p2)))

outcome = c(control, p1, p2)

n1 = sum((treat=="control") + 0)
n2 = sum((treat=="p1") + 0)
n3 = sum((treat=="p2") + 0)

n = n1+n2+n3

nbar = n/3

ssxobs = n1*mean(outcome[treat=="control"])**2 + n2*mean(outcome[treat=="p1"])**2 + n3*mean(out
come[treat=="p2"])**2
sstobs = ssxobs - n*mean(outcome)**2
sseobs = (n-1)*var(outcome)-sstobs
fobs = (sstobs/2)/(sseobs/(n-3))
ranks = rank(outcome)
a = 12/(n*(n+1))
b=(n+1)/2

kwobs = a*((n1*mean(ranks[treat=="control"])-b)**2) + (n2*(mean(ranks[treat=="p1"]) - b)**2) +
 (n3*(mean(ranks[treat=="p2"])-b)**2)

tot = 10000

d=c()
perm1 = c()
perm2 = c()
perm3 = c()
f = c()
kw = c()
q = c()

for( i in 1:tot){
  permut = sample(outcome)
  ranks = rank(permut)
  d[i] = n1*(mean(permut[treat=="control"])**2) + n2*(mean(permut[treat=="p1"])**2) + n3*(mean
(permut[treat=="p2"])**2)

  ssx = n1*(mean(permut[treat=="control"])**2) + n2*(mean(permut[treat=="p1"])**2) + n3*(mean(p
ermut[treat=="p2"])**2)
  sst=ssx-n*mean(permut)**2
  sse = ssx-n*mean(permut)**2
  sse = (n-1)*var(permut)-sst
  mse = sse/(n-3)
  f[i] = (sst/2)/(sse/(n-3))
  kw[i] = a*((n1*(mean(ranks[treat=="control"])-b)**2) + (n2*(mean(ranks[treat=="p1"])-b)**2) +
(n3*(mean(ranks[treat=="p2"])-b)**2))
```

```
  perm1[i] = (d[i] >= ssxobs)+0
  perm2[i] = (f[i] >= fobs) +0
  perm3[i] = (kw[i] >= kwobs) +0
  maxmean = max(mean(permut[treat=="control"]), mean(permut[treat=="p1"]), mean(permut[treat==
"p2"]))
  minmean = min(mean(permut[treat=="control"]), mean(permut[treat=='p1']), mean(permut[treat==
"p2"]))
  q[i] = ((maxmean-minmean))/(((2/nbar)*mse)**0.5)
}

pvalue1 = sum(perm1)/tot
pvalue2=  sum(perm2)/tot
pvalue3 = sum(perm3)/tot

tkhsdq = quantile(q, .95)

ranks = rank(outcome)
rmean1 = mean(ranks[treat=="control"])
rmean2 = mean(ranks[treat=="p1"])
rmean3 = mean(ranks[treat=="p2"])

mseranks = n*(n+1)/12

pvalue1
```

```
## [1] 0
```

```
pvalue2
```

```
## [1] 0
```

```
pvalue3
```

```
## [1] 0
```

```
#b)
# One way analysis of variance
#.libPaths('C:/Users/pumad/STA104')


#install.packages("ImPerm")
library(ImPerm)
```

```
## Warning: package 'ImPerm' was built under R version 4.0.5
```

```
summary(aov(outcome~treat))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## treat        2   2.85  1.4251   12.57 0.00114 **
## Residuals   12   1.36  0.1134
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(aov(outcome~ treat))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## treat        2   2.85  1.4251   12.57 0.00114 **
## Residuals   12   1.36  0.1134
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(aovp(outcome~treat))
```

```
## [1] "Settings:  unique SS "
```

```
## Component 1 :
##             Df R Sum Sq R Mean Sq Iter  Pr(Prob)
## treat1       2   2.8503   1.42513 5000 < 2.2e-16 ***
## Residuals   12   1.3604   0.11337
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
kruskal.test(outcome~treat)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  outcome by treat
## Kruskal-Wallis chi-squared = 12.375, df = 2, p-value = 0.002055
```

```
# Exercise 2

# Permutation F test
weight1 = c(574, 926, 789, 805, 361, 529)
weight2 = c(791, 394, 667, 1385, 1021, 2073, 1263, 1016, 1101, 945, 139)
weight3 = c(865, 775, 729, 1721, 820, 1613, 1202, 1201, 205, 1380, 580, 1803)
weight4 = c(998, 1049, 736, 782, 742, 1219, 705, 1260, 611, 1350, 1657)
weight5 = c(406, 1529, 1132, 767, 1224, 314, 1728)

outcome = c(weight1, weight2, weight3, weight4, weight5)
treat = c(rep("code1",length(weight1)), rep("code2", length(weight2)), rep("code3", length(weig
ht3)),
          rep("code4", length(weight4)), rep("code5", length(weight5)))


n1 = sum((treat=="code1") + 0)
n2 = sum((treat=="code2") + 0)
n3 = sum((treat=="code3") + 0)
n4 = sum((treat=="code4") + 0)
n5 = sum((treat=="code5") + 0)

n = n1+n2+n3+n4+n5

nbar = n/5

ssxobs = n1*mean(outcome[treat=="code1"])**2 + n2*mean(outcome[treat=="code2"])**2 + n3*mean(ou
tcome[treat=="code3"])**2 +
  n4*mean(outcome[treat=="code4"])**2 + n5*mean(outcome[treat=="code5"])**2
sstobs = ssxobs - n*mean(outcome)**2
sseobs = (n-1)*var(outcome)-sstobs
fobs = (sstobs/4)/(sseobs/(n-5))
ranks = rank(outcome)
a = 12/(n*(n+1))
b=(n+1)/2

kwobs = a*((n1*mean(ranks[treat=="code1"])-b)**2) + (n2*(mean(ranks[treat=="code2"]) - b)**2) +
(n3*(mean(ranks[treat=="code3"])-b)**2) +
  (n4*(mean(ranks[treat=="code4"])-b)**2) + (n5*(mean(ranks[treat=="code5"])-b)**2)

tot = 10000

d=c()
perm1 = c()
perm2 = c()
perm3 = c()
f = c()
kw = c()
q = c()

for( i in 1:tot){
  permut = sample(outcome)
  ranks = rank(permut)
  d[i] = n1*(mean(permut[treat=="code1"])**2) + n2*(mean(permut[treat=="code2"])**2) + n3*(mean
(permut[treat=="code3"])**2) +
    n4*(mean(permut[treat=="code3"])**2) +  n5*(mean(permut[treat=="code3"])**2)

  ssx = n1*(mean(permut[treat=="code1"])**2) + n2*(mean(permut[treat=="code2"])**2) + n3*(mean
```

```
(permut[treat=="code3"])**2) +
    n4*(mean(permut[treat=="code4"])**2) + n5*(mean(permut[treat=="code5"])**2)
  sst=ssx-n*mean(permut)**2
  sse = ssx-n*mean(permut)**2
  sse = (n-1)*var(permut)-sst
  mse = sse/(n-5)
  f[i] = (sst/4)/(sse/(n-5))
  kw[i] = a*((n1*(mean(ranks[treat=="code1"])-b)**2) + (n2*(mean(ranks[treat=="code2"])-b)**2)
 + (n3*(mean(ranks[treat=="code3"])-b)**2) +
    (n4*(mean(ranks[treat=="code4"])-b)**2) + (n5*(mean(ranks[treat=="code5"])-b)**2))
  perm1[i] = (d[i] >= ssxobs)+0
  perm2[i] = (f[i] >= fobs) +0
  perm3[i] = (kw[i] >= kwobs) +0
  maxmean = max(mean(permut[treat=="code1"]), mean(permut[treat=="code2"]), mean(permut[treat==
"code3"]), mean(permut[treat=="code4"]), mean(permut[treat=="code5"]))
  minmean = min(mean(permut[treat=="code1"]), mean(permut[treat=='code2']), mean(permut[treat==
"code3"]), mean(permut[treat=="code4"]), mean(permut[treat=="code5"]))
  q[i] = ((maxmean-minmean))/(((2/nbar)*mse)**0.5)
}

pvalue1 = sum(perm1)/tot
pvalue2=  sum(perm2)/tot
pvalue3 = sum(perm3)/tot

tkhsdq = quantile(q, .95)

ranks = rank(outcome)
rmean1 = mean(ranks[treat=="code1"])
rmean2 = mean(ranks[treat=="code2"])
rmean3 = mean(ranks[treat=="code3"])
rmean4 = mean(ranks[treat=="code4"])
rmean5 = mean(ranks[treat=="code5"])


mseranks = n*(n+1)/12

pvalue1
```

```
## [1] 0.4711
```

```
pvalue2
```

```
## [1] 0.467
```

```
pvalue3
```

```
## [1] 0
```

```
# ANOVA F-test

summary(aov(outcome~treat))
```

```
##             Df  Sum Sq Mean Sq F value Pr(>F)
## treat        4  723714  180928   0.905   0.47
## Residuals   42 8393656  199849
```

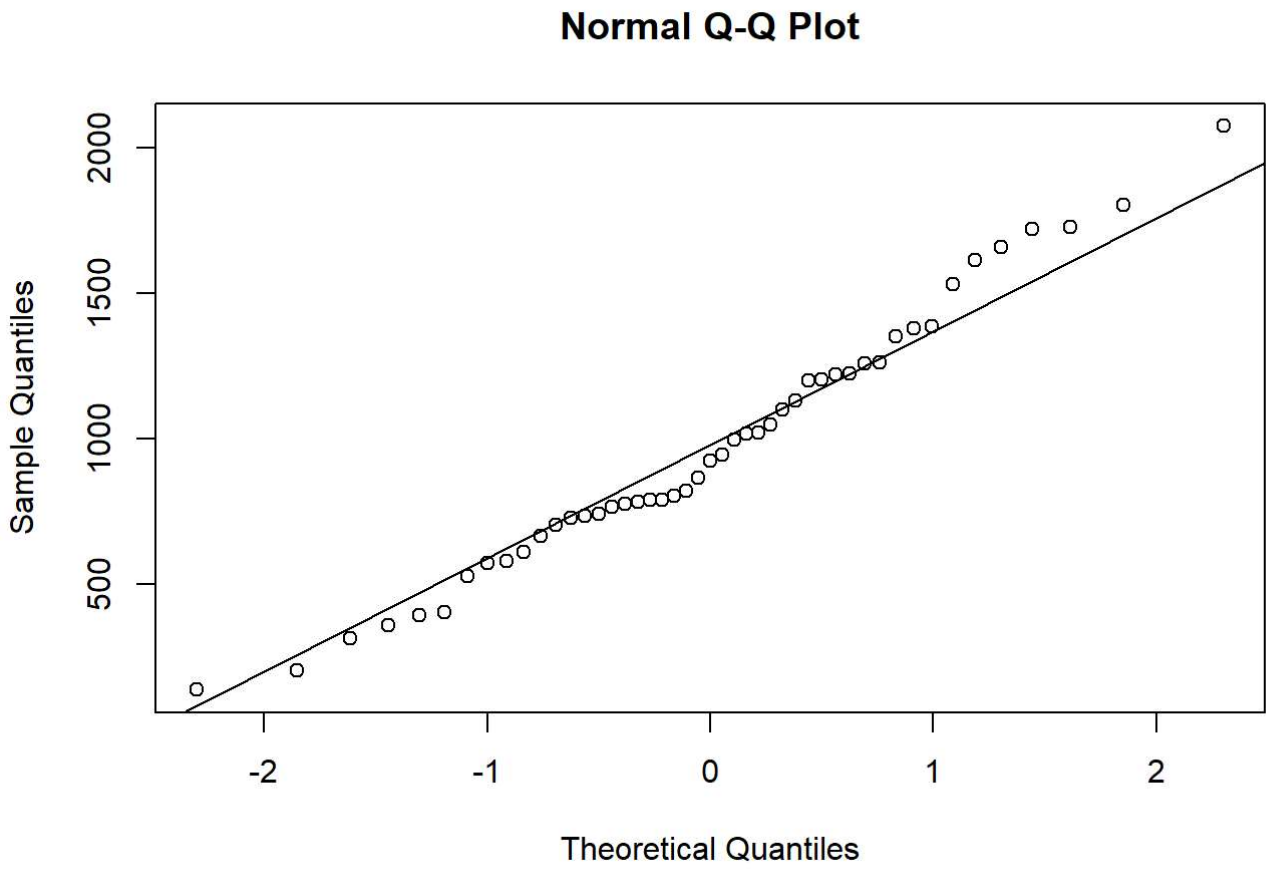
```
#install.packages("ggpubr")
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.0.5
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
qqnorm(outcome)
qqline(outcome)
```




```
# Exercise 3
kruskal.test(outcome~treat)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  outcome by treat
## Kruskal-Wallis chi-squared = 3.9549, df = 4, p-value = 0.4121
```

```
# Exercise 4

compact = c(791, 846, 1024, 1007, 1399, 1279, 1407, 656, 1036, 226)
heavy = c(423, 541, 517, 328, 1471, 533, 863, 786, 451, 1068)
light = c(551, 1068, 757, 1114, 920, 809, 1238, 1918, 1339, 1603)
medium = c(712, 435, 1298, 1733, 300, 701, 707, 790, 1800, 480)
mpv = c(1345, 1269, 1077, 1458, 996, 1306, 968, 943, 1026, 1564)
pickup = c(903, 949, 1183, 1051, 1342, 1184, 977, 1465, 892, 1074)
van = c(985, 1074, 742, 985, 805, 2613, 1387, 1320, 1434, 1603)

outcome = c(compact, heavy, light, medium, mpv, pickup, van)
treat = c(rep('compact',length(compact)), rep('heavy', length(heavy)), rep('light', length(ligh
t)),
          rep('medium', length(medium)), rep('mpv', length(mpv)),  rep('pickup', length(picku
p)),  rep('length', length(van)))

model = kruskal.test(outcome~treat)
ks = model[1][1]
ks
```

```
## $statistic
## Kruskal-Wallis chi-squared
##                   14.80486
```

```
fit = aov(outcome~treat)
fit
```

```
## Call:
##    aov(formula = outcome ~ treat)
##
## Terms:
##                   treat Residuals
## Sum of Squares  2443215   9789649
## Deg. of Freedom       6        63
##
## Residual standard error: 394.197
## Estimated effects may be unbalanced
```