Chapter 11:

Marjorie Blanco

Chapter 11 Page 402: #11.9 part (a)

1. You may use R to solve this part (NO built-in function). To get full credit, make sure you include the following steps, with code and output:

a) the sample correlation matrix

```
sons <- read_table("Software-Files/T3_8_SONS.DAT", col_names = c("y1", "y2", "x1", "x2"))
sons <- as.matrix(sons)
kable(sons) %>%
  kable_styling(bootstrap_options = "striped")
```

y1	y2	x1	x2
191	155	179	145
195	149	201	152
181	148	185	149
183	153	188	149
176	144	171	142
208	157	192	152
189	150	190	149
197	159	189	152
188	152	197	159
192	150	187	151
179	158	186	148
183	147	174	147
174	150	185	152
190	159	195	157
188	151	187	158
163	137	161	130
195	155	183	158
186	153	173	148
181	145	182	146
175	140	165	137
192	154	185	152
174	143	178	147
176	139	176	143
197	167	200	158
190	163	187	150

```
sons.std <-sweep(sons, 2, sqrt(apply(sons,2,var)), FUN="/")
first.son.meas <-sons.std[,1:2]
second.son.meas <-sons.std[,3:4]
```

first.son.meas =

```
kable(first.son.meas) %>%
  kable_styling(bootstrap_options = "striped")
```

y1	y2
19.56600	21.02287
19.97576	20.20908
18.54160	20.07345
18.74648	20.75161
18.02941	19.53092
21.30748	21.29413
19.36112	20.34471
20.18064	21.56539
19.25868	20.61597
19.66844	20.34471
18.33673	21.42976
18.74648	19.93782
17.82453	20.34471
19.46356	21.56539
19.25868	20.48034
16.69769	18.58150
19.97576	21.02287
19.05380	20.75161
18.54160	19.66655
17.92697	18.98840
19.66844	20.88724
17.82453	19.39529
18.02941	18.85277
20.18064	22.65045
19.46356	22.10792

second.son.meas =

```
kable(second.son.meas) %>%
  kable_styling(bootstrap_options = "striped")
```

x1	x2
17.82824	21.60972
20.01942	22.65295
18.42583	22.20585
18.72463	22.20585
17.03144	21.16262
19.12303	22.65295
18.92383	22.20585
18.82423	22.65295
19.62102	23.69618
18.62503	22.50392
18.52543	22.05682
17.33024	21.90779
18.42583	22.65295
19.42182	23.39811
18.62503	23.54715
16.03545	19.37423
18.22663	23.54715
17.23064	22.05682
18.12703	21.75875
16.43385	20.41746
18.42583	22.65295
17.72864	21.90779
17.52944	21.31166
19.91982	23.54715
18.62503	22.35488

```
R11 <-cor(first.son.meas)
R11
```

```
##           y1        y2
## y1 1.0000000 0.7345555
## y2 0.7345555 1.0000000
```

```
R22 <-cor(second.son.meas)
R22
```

```
##           x1        x2
## x1 1.0000000 0.8392519
## x2 0.8392519 1.0000000
```

```
R12 <-c(cor(first.son.meas[,1], second.son.meas[,1]),
        cor(first.son.meas[,1], second.son.meas[,2]),
        cor(first.son.meas[,2], second.son.meas[,1]),
        cor(first.son.meas[,2], second.son.meas[,2]))
```

R12 =

```
R12 <-matrix(R12, ncol=ncol(R22), byrow=T) # R12 has q2 columns, same as number of petal measurements
R12
```

```
##           [,1]      [,2]
## [1,] 0.7107518 0.7039807
## [2,] 0.6931573 0.7085504
```

```
R21 <-t(R12)  # R21=transpose of R12
```

```
S <- cbind(rbind(R22, R12), rbind(R21, R11))
```

S =

```
##           x1        x2        y1        y2
## x1 1.0000000 0.8392519 0.7107518 0.6931573
## x2 0.8392519 1.0000000 0.7039807 0.7085504
##    0.7107518 0.7039807 1.0000000 0.7345555
##    0.6931573 0.7085504 0.7345555 1.0000000
```

```
cor(sons)
```

```
##           y1        y2        x1        x2
## y1 1.0000000 0.7345555 0.7107518 0.7039807
## y2 0.7345555 1.0000000 0.6931573 0.7085504
## x1 0.7107518 0.6931573 1.0000000 0.8392519
## x2 0.7039807 0.7085504 0.8392519 1.0000000
```

b) the characteristic equation

```
# Finding the E1 and E2 matrices:
E1 <-solve(R11) %*% R12 %*% solve(R22) %*% R21
E2 <-solve(R22) %*% R21 %*% solve(R11) %*% R12
eigen(E1)
```

```
## eigen() decomposition
## $values
## [1] 0.621744734 0.002887956
##
## $vectors
##           [,1]       [,2]
## [1,] 0.7269968 -0.7040109
## [2,] 0.6866408  0.7101892
```

```
eigen(E2)
```

```
## eigen() decomposition
## $values
## [1] 0.621744734 0.002887956
##
## $vectors
##            [,1]       [,2]
## [1,] -0.6837994 -0.7091095
## [2,] -0.7296700  0.7050984
```

u1 = 0.7269968 * First Son Head Length + 0.6866408 * First Son Head Breadth

v1 = -0.6837994 * Second Son Head Length + -0.72967 * Second Son Head Breadth

u2 = -0.7040109 * First Son Head Length + 0.7101892 * First Son Head Breadth

v2 = -0.7091095 * Second Son Head Length + 0.7050984 * Second Son Head Breadth

c) eigenvalues

```
eigen(E1)$values
```

```
## [1] 0.621744734 0.002887956
```

```
eigen(E2)$values
```

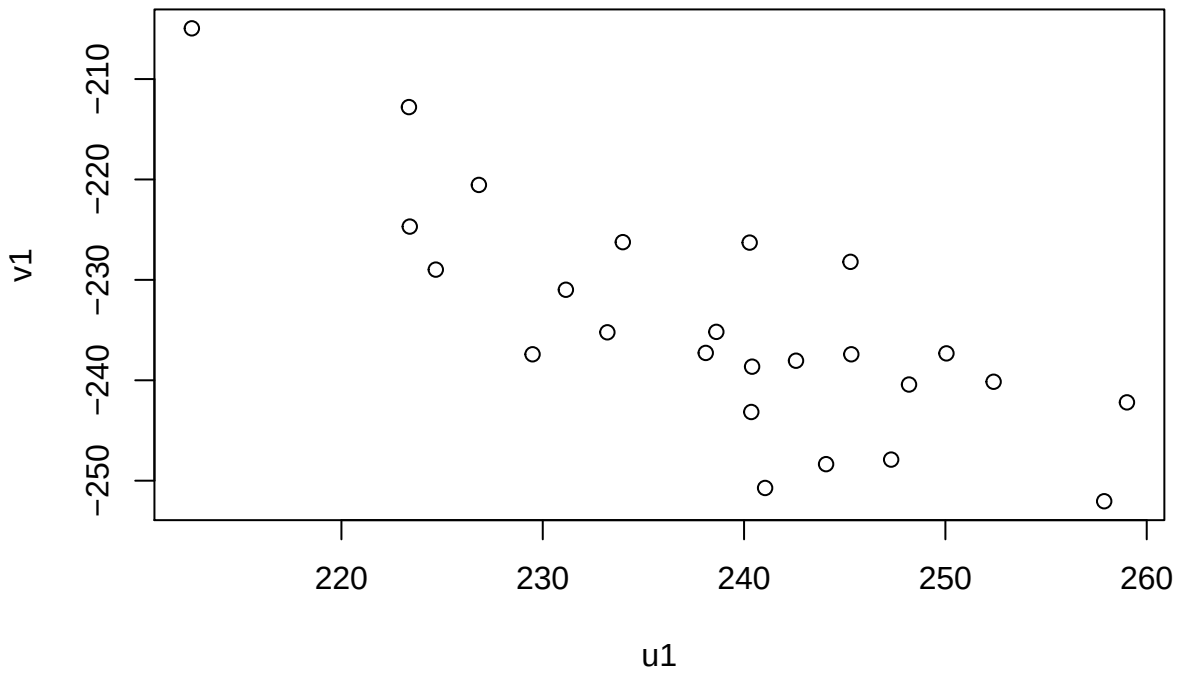
```
## [1] 0.621744734 0.002887956
```

(a) Find the canonical correlations between (y_1, y_2) and (x_1, x_2).

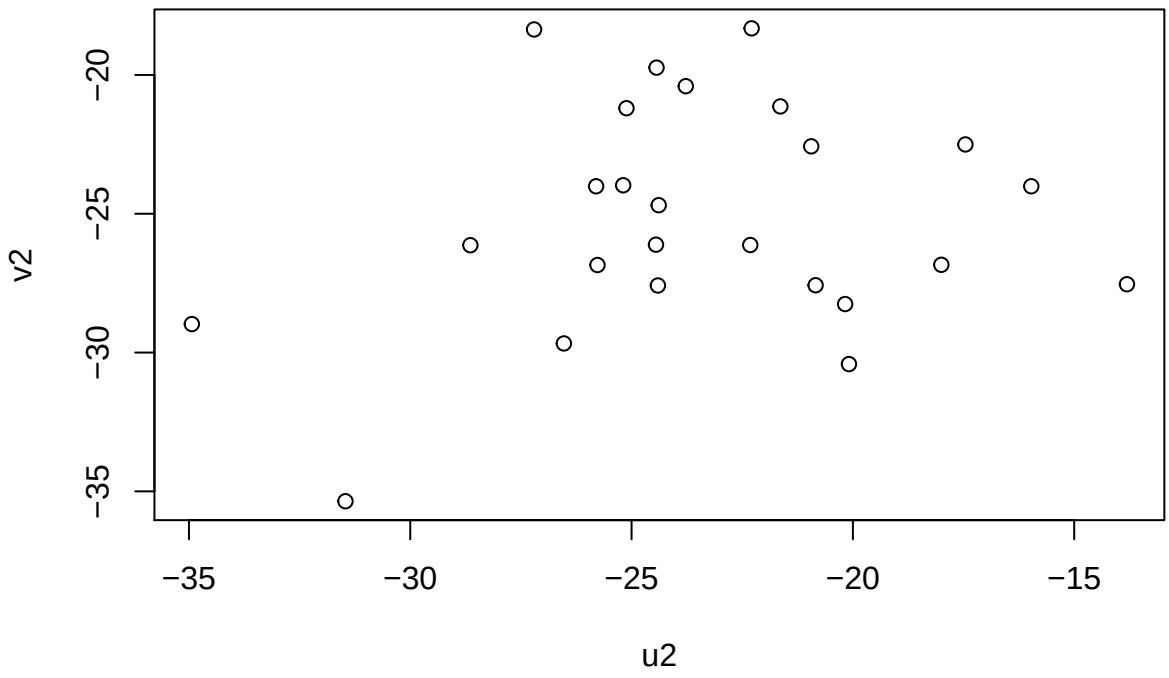
```
canon.corr <-sqrt(eigen(E1)$values)
canon.corr
```

```
## [1] 0.7885079 0.0537397
```

```
# Plotting the first set of canonical variables:
u1 <-as.matrix(sons[,1:2]) %*% as.matrix(eigen(E1)$vectors[,1])
v1 <-as.matrix(sons[,3:4]) %*% as.matrix(eigen(E2)$vectors[,1])
plot(u1,v1)
```




```
# Plotting the second set of canonical variables:
u2 <-as.matrix(sons[,1:2]) %*% as.matrix(eigen(E1)$vectors[,2])
v2 <-as.matrix(sons[,3:4]) %*% as.matrix(eigen(E2)$vectors[,2])
plot(u2,v2)
```



$r_1 = 0.7885079$

$r_2 = 0.0537397$

The first canonical variate captures the most explained variance, canonical r = 0.7885079.

```
cca1 <- cancor(first.son.meas, second.son.meas)
# The canonical correlations are the same as the ones we found,
# The canonical variates are a little different because the cancor
# function works with the centered data rather than the original data.
cca1
```

```
## $cor
## [1] 0.7885079 0.0537397
##
## $xcoef
##         [,1]       [,2]
## y1 0.1127152 -0.2789099
## y2 0.1064583  0.2813576
##
## $ycoef
##         [,1]       [,2]
## x1 0.1029701 -0.3610078
## x2 0.1098775  0.3589657
##
## $xcenter
##       y1       y2
```

```
## 19.02512 20.49662
##
## $ycenter
##       x1       x2
## 18.31030 22.24162
```

```
cc(first.son.meas, second.son.meas)
```

```
## $cor
## [1] 0.7885079 0.0537397
##
## $names
## $names$Xnames
## [1] "y1" "y2"
##
## $names$Ynames
## [1] "x1" "x2"
##
## $names$ind.names
## NULL
##
##
## $xcoef
##          [,1]      [,2]
## y1 -0.5521896 -1.366374
## y2 -0.5215372  1.378365
##
## $ycoef
##          [,1]      [,2]
## x1 -0.5044484 -1.768570
## x2 -0.5382877  1.758566
##
## $scores
## $scores$xscores
##              [,1]        [,2]
##  [1,] -0.57312842 -0.01368291
##  [2,] -0.37497221 -1.69526490
##  [3,]  0.48769136  0.07738080
##  [4,]  0.02087481  0.73218662
##  [5,]  1.05346966  0.02943785
##  [6,] -1.67622740 -2.01929228
##  [7,] -0.10631187 -0.66848874
##  [8,] -1.19547291 -0.10571105
##  [9,] -0.19121934 -0.15461844
## [10,] -0.27601046 -1.08840202
## [11,] -0.10654457  2.22681901
## [12,]  0.44529580 -0.38951099
## [13,]  0.74218106  1.43107765
## [14,] -0.79950955  0.87408659
## [15,] -0.12048251 -0.34156804
## [16,]  2.28398802  0.54041483
## [17,] -0.79939320 -0.57356728
## [18,] -0.14882378  0.31227335
## [19,]  0.69990185 -0.48346801
```

```
## [20,]  1.39298318 -0.57838947
## [21,] -0.55895778 -0.34060361
## [22,]  1.23733888  0.12243043
## [23,]  1.40715381 -0.90531017
## [24,] -1.76136757  1.38988577
## [25,] -1.08245687  1.62188500
##
## $scores$yscores
##               [,1]        [,2]
##  [1,]  0.583317147 -0.25867829
##  [2,] -1.083576808 -2.29934799
##  [3,] -0.039028038 -0.26723166
##  [4,] -0.189755833 -0.79567548
##  [5,]  1.225925130  0.36425453
##  [6,] -0.631393423 -0.71401655
##  [7,] -0.290241030 -1.14797136
##  [8,] -0.480665628 -0.18557273
##  [9,] -1.444163206  0.23982870
## [10,] -0.299958033 -0.09536041
## [11,] -0.009048238 -0.70546317
## [12,]  0.674085341  1.14622854
## [13,] -0.279695235  0.51901903
## [14,] -1.183233212  0.06795746
## [15,] -0.861514824  1.73922453
## [16,]  2.691019899 -1.01926884
## [17,] -0.660544431  2.44381628
## [18,]  0.644105541  1.58446004
## [19,]  0.352366953 -0.52503854
## [20,]  1.928492714  0.11072435
## [21,] -0.279695235  0.51901903
## [22,]  0.473114948  0.44163678
## [23,]  0.894489740 -0.25440160
## [24,] -1.514668603 -0.55069868
## [25,] -0.219735634 -0.35744398
##
## $scores$corr.X.xscores
##          [,1]       [,2]
## y1 -0.9352877 -0.3538884
## y2 -0.9271512  0.3746875
##
## $scores$corr.Y.xscores
##          [,1]        [,2]
## x1 -0.7539771 -0.01572908
## x2 -0.7582663  0.01474027
##
## $scores$corr.X.yscores
##          [,1]        [,2]
## y1 -0.7374817 -0.01901786
## y2 -0.7310660  0.02013559
##
## $scores$corr.Y.yscores
##          [,1]       [,2]
## x1 -0.9562074 -0.2926900
## x2 -0.9616470  0.2742901
```

Display the canonical correlations

```
first.son <- sons[,1:2]
second.son <- sons[,3:4]
cc1 <- cc(first.son, second.son)

# display the canonical correlations
cc1$cor
```

```
## [1] 0.7885079 0.0537397
```

```
xcoef <- cc1[1:2]

ycoef <- cc1[3:4]

cc1[3:4]
```

```
## $xcoef
##                               [,1]       [,2]
## First.Son.Head.Length  -0.05656620 -0.1399711
## First.Son.Head.Breadth -0.07073683  0.1869496
##
## $ycoef
##                               [,1]       [,2]
## Second.Son.Head.Length  -0.0502426 -0.1761479
## Second.Son.Head.Breadth -0.0802224  0.2620836
```

The raw canonical coefficients are interpreted similar to interpreting regression coefficients.

For Second.Son.Head.Length, a one unit increase in Second.Son.Head.Length leads to a 0.0502426 decrease in the first canonical variate, holding all other variables constant.

For Second.Son.Head.Breadth, a one unit increase in Second.Son.Head.Breadth leads to a 0.0802224 increase in the first canonical variate, holding all other variables constant.

For Second.Son.Head.Length, a one unit increase in Second.Son.Head.Length leads to a 0.1761479 decrease in the second canonical variate, holding all other variables constant.

For Second.Son.Head.Breadth, a one unit increase in Second.Son.Head.Breadth leads to a 0.2620836 increase in the second canonical variate, holding all other variables constant.

Compute canonical loadings

```
cc2 <- comput(first.son, second.son, cc1)
```

Display canonical loadings

```
cc2[3:6]
```

```
## $corr.X.xscores
##                              [,1]       [,2]
## First.Son.Head.Length  -0.9352877 -0.3538884
## First.Son.Head.Breadth -0.9271512  0.3746875
##
```

```
## $corr.Y.xscores
##                                [,1]        [,2]
## Second.Son.Head.Length  -0.7539771 -0.01572908
## Second.Son.Head.Breadth -0.7582663  0.01474027
##
## $corr.X.yscores
##                               [,1]        [,2]
## First.Son.Head.Length  -0.7374817 -0.01901786
## First.Son.Head.Breadth -0.7310660  0.02013559
##
## $corr.Y.yscores
##                                [,1]       [,2]
## Second.Son.Head.Length  -0.9562074 -0.2926900
## Second.Son.Head.Breadth -0.9616470  0.2742901
```

```
# tests of canonical dimensions
ev <- (1 - cc1$cor^2)

n <- dim(first.son)[1]
p <-dim(first.son)[2]
q <- dim(second.son)[2]
k <- min(p, q)
m <- n - 3/2 - (p + q)/2

w <- rev(cumprod(rev(ev)))

# initialize
d1 <- d2 <- f <- vector("numeric", k)

for (i in 1:k) {
    s <- sqrt((p^2 * q^2 - 4)/(p^2 + q^2 - 5))
    si <- 1/s
    d1[i] <- p * q
    d2[i] <- m * s - p * q/2 + 1
    r <- (1 - w[i]^si)/w[i]^si
    f[i] <- r * d2[i]/d1[i]
    p <- p - 1
    q <- q - 1
}

pv <- pf(f, d1, d2, lower.tail = FALSE)
(dmat <- cbind(WilksL = w, F = f, df1 = d1, df2 = d2, p = pv))
```

```
##         WilksL          F df1 df2            p
## [1,] 0.3771629 6.59719349   4  42 0.0003256458
## [2,] 0.9971120 0.06371905   1  22 0.8030550074
```

Standardized first.son canonical coefficients diagonal matrix of first.son sd's

```
s1 <- diag(sqrt(diag(cov(first.son))))
s1 %*% cc1$xcoef
```

```
##            [,1]      [,2]
```

```
## [1,] -0.5521896 -1.366374
## [2,] -0.5215372  1.378365
```

Standardized second.son canonical coefficients diagonal matrix of second.son sd's

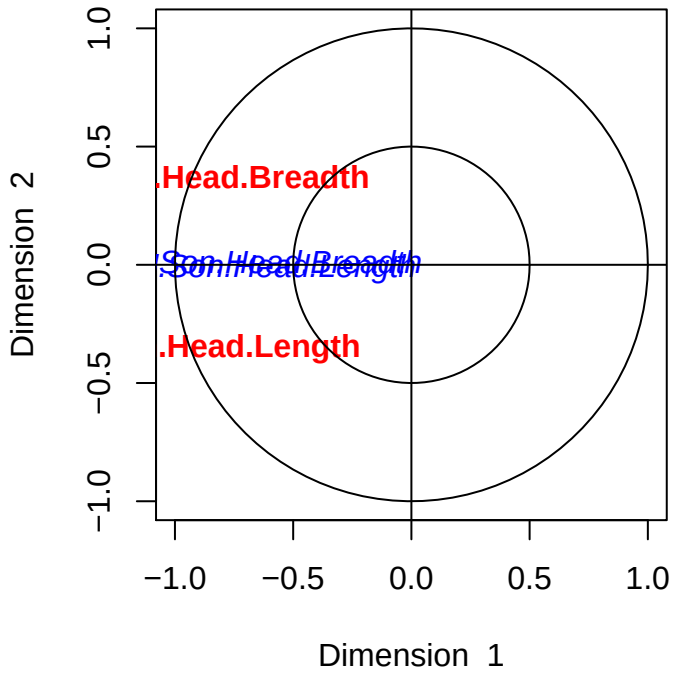
```
s2 <- diag(sqrt(diag(cov(second.son))))
s2 %*% cc1$ycoef
```

```
##            [,1]      [,2]
## [1,] -0.5044484 -1.768570
## [2,] -0.5382877  1.758566
```

```
matcor(first.son, second.son)
```

```
## $Xcor
##                         First.Son.Head.Length First.Son.Head.Breadth
## First.Son.Head.Length               1.0000000              0.7345555
## First.Son.Head.Breadth              0.7345555              1.0000000
##
## $Ycor
##                          Second.Son.Head.Length Second.Son.Head.Breadth
## Second.Son.Head.Length                1.0000000                0.8392519
## Second.Son.Head.Breadth               0.8392519                1.0000000
##
## $XYcor
##                          First.Son.Head.Length First.Son.Head.Breadth
## First.Son.Head.Length                1.0000000              0.7345555
## First.Son.Head.Breadth               0.7345555              1.0000000
## Second.Son.Head.Length               0.7107518              0.6931573
## Second.Son.Head.Breadth              0.7039807              0.7085504
##                          Second.Son.Head.Length Second.Son.Head.Breadth
## First.Son.Head.Length                 0.7107518                0.7039807
## First.Son.Head.Breadth                0.6931573                0.7085504
## Second.Son.Head.Length                1.0000000                0.8392519
## Second.Son.Head.Breadth               0.8392519                1.0000000
```

```
plt.cc(cc1, type="v", var.label = TRUE)
```




```
options(scipen=999) # Permits decimal values rather than scientific notation
cca2.fit = cca(first.son, second.son)
F.test.cca(cca2.fit)
```

```
##
##  F Test for Canonical Correlations (Rao's F Approximation)
##
##          Corr        F   Num df Den df    Pr(>F)
## CV 1 0.788508 6.597193 4.000000     42 0.0003256 ***
## CV 2 0.053740 0.063719 1.000000     22 0.8030550
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2. Chapter 11 Page 402: #11.9 part (c)

You CANNOT use R for this part.

(c) Test the significance of each canonical correlation.

H_0: there is no (linear) relationship between the y's and the x's all canonical correlations r_1, r_2 are non-significant.

H_1: there is (linear) relationship between the y's and the x's at least one canonical correlations r_1, r_2 is significant.

We reject the null hypothesis in favor of the alternative. This implies that at least r_1^2 is significantly different from zero.

We conclude that $r_1 = 0.7885079$ is significant since the p-value < 0.05

We conclude that $r_2 = 0.7885079$ is not significant since the p-value > 0.05