Exam 2021, R-solution

1 Canonical correlation analysis

We are working with the `"datavalg2020.csv"` data set.

```r
data <- read.csv('valg2020.csv')[-1]
data <- data[c("H10", "H20", "H30", "H35", "H40",
                 "H50", "H60", "H70", "H80", "H90",
                "A", "B", "C", "D", "F", "I", "O", "V", "Ø", "Å")]
```

We perform canonical correlation analysis with the education variables as VAR (Y) variables and party as WITH (X) variables:

```r
corr <- cor(data)

Eyy <- as.matrix(corr[1:10,1:10])
Exy <- as.matrix(corr[11:20,1:10])
Eyx <- as.matrix(corr[1:10,11:20])
Exx <- as.matrix(corr[11:20,11:20])

invExx <- solve(Exx)
invEyy <- solve(Eyy)

# Finding a
Cancorry <- geigen::geigen(Eyx %*% invExx %*% Exy,Eyy,symmetric = T)

# Finding b
Cancorrx <- geigen::geigen(Exy %*% invEyy %*% Eyx,Exx,symmetric = T)

# Squared canonical correlations
```

```
values = sort(Cancorry$values,decreasing = TRUE)
```

1.1 Variance explained by 3rd set of canonical variables.

The variance explained by the 3rd set of canonical variables is the squared canonical correlation.

```
values[3]

## [1] 0.7346896
```

The answer is **0.7346896**.

1.2 Eigenvalues significantly different from zero

. We compute the eigenvalues of $\boldsymbol{H}^{1-}\boldsymbol{E}$

```
H = Eyx%*%invExx%*%Exy
E = Eyy - Eyy%*%invExx%*%Exy
invE = solve(E)
Ev <- eigen(invE%*%H)
var = Ev$values
```

Next we compute the p-values using the `p.asym` function form the CCP library.

```
n = 98
p = length(Exx[,1])
q = length(Eyy[,1])

HypTest <- CCP::p.asym(sqrt(values),n,p,q,tstat="Wilks")

## Wilks' Lambda, using F-approximation (Rao's F):
##                     stat      approx df1      df2      p.value
## 1 to 10:   2.872876e-06 28.19092843 100 570.3155 0.000000e+00
## 2 to 10:   1.657652e-03 10.85893621  81 519.5091 0.000000e+00
## 3 to 10:   6.363783e-02  4.47557515  64 467.9205 0.000000e+00
## 4 to 10:   2.398618e-01  2.75464075  49 415.6457 2.477490e-08
## 5 to 10:   5.212080e-01  1.61225388  36 362.8481 1.673747e-02
## 6 to 10:   7.590637e-01  0.95466470  25 309.8332 5.289720e-01
## 7 to 10:   8.846663e-01  0.65806839  16 257.2619 8.336171e-01
## 8 to 10:   9.720505e-01  0.26948826   9 207.0183 9.821506e-01
## 9 to 10:   9.918963e-01  0.17529652   4 172.0000 9.508575e-01
## 10 to 10:  9.995962e-01  0.03514187   1  87.0000 8.517352e-01
```

```
sum(HypTest$p.value < 0.05)
```

```
## [1] 5
```

The answer is **5 eigenvalues**.

1.3 Fraction of variance in the WITH variable 'C' explained by V1 and V2

We will need to compute the correlation between WITH (X) variables and their canonical variables of the VAR (X) variables.

```
Corxy <- Exy %*% Cancorry$vectors
```

The output from the geigen function orders the eigenvectors from lowest to highest eigenvalue, so we are interested in the last two vectors.

```
VWCorrelations = data.frame("Variables" = names(data)[11:20],
                            "CorrV1" = -1*round(Corxy[,10],3),
                            "CorrV2" = -1*round(Corxy[,9],3))
```

```
## [1] "Standardized Canonical Coefficients for the VAR Variables"
## [1] "and Correlations Between the VAR Variables and Their Canonical Variables:'
##   Variables CorrV1 CorrV2
## A         A  0.919  0.290
## B         B  0.957 -0.128
## C         C  0.497 -0.097
## D         D  0.828  0.052
## F         F  0.947 -0.027
## I         I  0.952 -0.024
## O         O  0.830  0.324
## V         V  0.643  0.568
## Ø         Ø  0.941 -0.324
## Å         Å  0.936 -0.333
```

Finally, we find the answer by squaring and summing the correlations.

```
sum(VWCorrelations[3,c("CorrV1","CorrV2")]**2)
```

```
## [1] 0.256418
```

The answer is **0.256418**.

1.4 Interpretation of V4 and W4 based on correlations.

For V4, we have

```
Coryy <- Eyy %*% Cancorry$vectors
V4 = data.frame("CorrV4" = -1*round(Coryy[,7],3))
```

```
## [1] "Standardized Canonical Coefficients for V4 Variables"
## [1] "and Correlation with the VAR Variables:"
##     CorrV4
## H10  0.025
## H20 -0.078
## H30  0.024
## H35 -0.318
## H40  0.001
## H50 -0.010
## H60 -0.068
## H70  0.031
## H80  0.009
## H90  0.028
```

And for W4 we obtain

```
Corxx <- Exx %*% Cancorrx$vectors
W4 = data.frame("CorrW4" = -1*round(Corxx[,7],3))
```

```
## [1] "Standardized Canonical Coefficients for W4 Variables "
## [1] "and Correlation with the WITH Variables:"
##   CorrW4
## A -0.156
## B -0.027
## C  0.628
## D  0.054
## F -0.142
## I -0.015
## O  0.012
## V  0.049
## Ø -0.011
## Å -0.054
```

The answer is that **V4 is mainly negatively correlated with admittance education (H35), and W4 is a contrast between C vs. (A, F)**.

2 Linear regression

We read the data and fit the linear model using the `lm` function.

```
data <- haven::read_sas('exam2021.sas7bdat')[-1]
lmA <- lm(A~.,data=data)
```

2.1 Variable with largest tolerance

We compute the tolerances

```
olsrr::ols_vif_tol(lmA)
```

```
##    Variables    Tolerance        VIF
## 1        H10 6.220447e-04  1607.6015
## 2        H20 1.929100e-04  5183.7650
## 3        H30 1.358246e-03   736.2435
## 4        H35 6.562352e-04  1523.8439
## 5        H40 3.785097e-03   264.1941
## 6        H50 2.666141e-04  3750.7398
## 7        H60 3.826781e-04  2613.1622
## 8        H70 3.753306e-05 26643.1808
## 9        H80 4.310035e-05 23201.6663
## 10       H90 1.520617e-03   657.6276
```

The correct answer is **H40**.

2.2 4 variables that attain the highest R^2

We need to check all ways to combine 4 of the variables. The `combs` function can help us find all the combinations.

```
vars <- colnames(data)[-11]
combs <- utils::combn(vars,4)
```

We illustrate by printing the first couple of combinations:

```
##      [,1]  [,2]  [,3]
## [1,] "H10" "H10" "H10"
## [2,] "H20" "H20" "H20"
## [3,] "H30" "H30" "H30"
## [4,] "H35" "H40" "H50"
```

Now we loop though all the combinations and save the one with the largest R^2.

```
bestR = 0   # Initialize best R^2 value

for (i in 1:(dim(combs)[2])){
    # fit model
  lm4 <- lm(A~., data=data[c(combs[,i],"A")])
  sum <- summary(lm4)
  R <- sum$adj.r.squared
  if (R > bestR){   # update if this model is the best so far
    bestlm = lm4
    bestR = R
  }
}

bestlm

##
## Call:
## lm(formula = A ~ ., data = data[c(combs[, i], "A")])
##
## Coefficients:
## (Intercept)          H20          H40          H60          H70
##     809.359        5.517       -4.249      -21.543        4.115
```

The answer is **H20, H40, H60, H70**.

2.3 Observation with lowest leverage

We fit the new model and compute the leverage.

```
lmA <- lm(A~H20+H60+H80,data=data)
VGAM::hatvalues(lmA)
```

```
##          1          2          3          4          5          6          7
## 0.09436900 0.32991475 0.08288270 0.12347343 0.28220539 0.23161319 0.13632903
##          8          9         10         11         12         13         14
## 0.76661770 0.08862165 0.99880448 0.27290387 0.11258920 0.07545467 0.11838525
##         15         16         17
## 0.07748967 0.06751325 0.14083277
```

So the answer is observation **16**.

2.4 Observation with largest impact in the parameter estimate for H80

We are interested in the DFBETAS values for $H80$.

```
dfbetas(lmA)
```

```
##      (Intercept)           H20           H60          H80
## 1   0.1990933196 -0.1047265834  0.1224701759 -0.102249385
## 2  -0.1668667087  0.3053491310 -0.1381093222  0.025890641
## 3   0.2652844939 -0.1158536545  0.1473106080 -0.128653200
## 4  -0.1339429011  0.0973050314 -0.0378432099  0.002722790
## 5  -0.1935931920  0.3572778652 -0.1959588669  0.071691529
## 6   0.1520934325 -0.1254204565  0.0981590201 -0.062279879
## 7  -0.0114784576  0.0336990115 -0.0396479628  0.032094196
## 8   0.0883130970 -0.1528019769  1.1105686724 -1.262084310
## 9  -0.0771126015  0.0438126856 -0.0001335257 -0.018764586
## 10  0.0317438806 -3.6571574039  5.4092950618 -1.706691544
## 11  0.0009441777 -1.1870990410 -1.0428465676  1.801808552
## 12  0.1181256031  0.2535262158  0.1489561339 -0.301079177
## 13 -0.2041286694  0.0651754452  0.0550560970 -0.091499125
## 14 -0.1743476360  0.1222214592 -0.0656280824  0.025487848
## 15 -0.0558695698  0.0124331421 -0.0331640813  0.035277556
## 16  0.0074584760  0.0004479059  0.0019838075 -0.002793473
## 17  0.2144310867 -0.1550576602  0.1228658918 -0.079489529
```

The answer is observation **11**.

2.5 Number of observations with Cook's D lager than 0.235

We compute Cook's D and count.

```
cooksD <- cooks.distance(lmA)
sum(cooksD >= 0.235)

## [1] 3
```

The answer is **3** observations.

2.6 Parameter estimates with largest (absolute) variance

We find the covariance matrix of the parameter estimates.

```
vcov(lmA)

##               (Intercept)          H20         H60          H80
## (Intercept) 1180056.9412 -438.2618633 2773.303723 -4172.481194
## H20            -438.2619    0.2372283   -1.257122     1.430504
## H60            2773.3037   -1.2571220   18.916131   -45.942141
## H80           -4172.4812    1.4305040  -45.942141   129.055743
```

The answer is **Intercept and H80**

3 Discriminant analysis

We load the glass data.

```
colnames <- c("","RI", "Na", "Mg", "Al", "Si", "K", "Ca", "Ba",
                            "Fe", "Type")
glass <- read.table("glass.data",sep=",",
                           col.names = colnames)[,-1]
glass$Type <- as.factor(glass$Type)
```

3.1 Difference in substitution misclassifications when going from QDA to LDA

This problem can not be solved in R as it is not possible to perform Quadratic discriminant analysis on this data set in R. A problem like this will not occur in the exam.

If you are interested to see how to compute the misclassifications in the linear case it is computed below:

First, we perform Linear discriminant analysis with equal priors.

```
prior = c(rep(1/6,6))
LDA <- MASS::lda(Type~., data=glass,prior=prior)

Class_Level_Information = data.frame(
                                "Frequency" = LDA$counts,
                                "Proportion"=LDA$counts/LDA$N,
                                "Prior"=LDA$prior)
```

```
## [1] "Class Level Information:"
##   Frequency Proportion     Prior
## 1        70 0.32710280 0.1666667
## 2        76 0.35514019 0.1666667
## 3        17 0.07943925 0.1666667
## 5        13 0.06074766 0.1666667
## 6         9 0.04205607 0.1666667
## 7        29 0.13551402 0.1666667
```

We predict the data using the Linear Discriminant model.

```
LDpred <- predict(LDA)
```

We use the `table` function to create the confusion matrix:

```
LD_CM <- xtabs(~glass$Type+LDpred$class)
```

```
## [1] "Confusion Matrix for LDA:"
##             LDpred$class
## glass$Type   1  2  3  5  6  7
##          1 46 14 10  0  0  0
##          2 16 41 12  4  3  0
##          3  3  3 11  0  0  0
##          5  0  2  0 10  0  1
##          6  1  1  0  0  7  0
##          7  0  1  1  2  1 24
```

The resubstitution misclassifications in the Linear model is now

```
sum(LD_CM) - sum(diag(LD_CM))

## [1] 75
```

3.2 The most different classes measured by Mahalanobis distances (pooled model)

Since we are considering the pooled dispersion matrix, we are interested in the Linear Disrimination.

According to page 333, the empirical Mahalanobis' distance is

$$D^2 = \left(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_2\right)^T \hat{\Sigma}^{-1} \left(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_2\right)$$

First we need to compute estimates of the mean and pooled covariance matrix.

Then we can compute Mahalanobis' distrance between each of the classes.

```
pcov = Rfast::pooled.cov(as.matrix(glass[,-10]),glass$Type)
invCov = solve(pcov)
Means = LDA$means

maha = matrix(NA,ncol=6,nrow=6)
for (i in c(1:6)){
  for (j in c(1:6)){
    mu = Means[i,] - Means[j,]
    maha[i,j] = t(mu)%*%invCov%*%mu
  }
}
maha
```

```
##           [,1]      [,2]      [,3]     [,4]     [,5]     [,6]
## [1,]  0.000000  1.239070  1.401595 17.49199 15.72381 37.11137
## [2,]  1.239070  0.000000  2.014705 11.35021 13.57536 32.09556
## [3,]  1.401595  2.014705  0.000000 18.88398 16.68870 41.34160
## [4,] 17.491986 11.350209 18.883983  0.00000 16.08992 22.73775
## [5,] 15.723807 13.575356 16.688700 16.08992  0.00000 13.94608
## [6,] 37.111366 32.095556 41.341602 22.73775 13.94608  0.00000
```

We see that the biggest difference measured in Mahalanobis' distance is between **group 3 and 7**.

3.3 Test if type 1 and type 3 have different mean values when using 9 variables.

We need to use theorem 5.12.

To do this, we perform a Linear Discriminant Analysis only on the data belonging to Type 1 and 3 and find Mahalanobis' distance between the two groups. We subset the data and repeat the process from before

```
glass13 <- glass[glass$Type %in% c(1,3),]
glass13$Type <- droplevels(glass13$Type)

prior13 <- c(1/2,1/2)
LDA13 <- MASS::lda(Type~., data=glass13,prior=prior13)
```

Resulting in

```
## [1] "Mahalanobis' distance between Type 1 and 3"
##          [,1]     [,2]
## [1,] 0.000000 2.163477
## [2,] 2.163477 0.000000
```

Finally, we can compute the test-statistic and the p-value

```
n1 <- 70
n2 <- 17
p <- 9
d2 <- maha13[1,2]

Ftest <- (n1+n2-p-1)/(p*(n1+n2-2))*n1*n2/(n1+n2)*d2

1-pf(Ftest,p,n1+n2-p-1)     #p value

## [1] 0.004285748
```

The answer is **0.0043**.

3.4 Test whether Ca, Fe and Fi construbute to discrimination between type 1 and 3.

We will use theorem 5.21.

First, we need to perform a Linear Discriminant analysis on the subsetted glass data, but without including Ca, Fe and Fi.

```
LDA_small <- MASS::lda(Type~RI+Na+Mg+Al+K+Ba, data=glass13,
                       prior=prior13)
```

Next, we compute Mahalanobis' distance as in the previous two problems.

```
##          [,1]     [,2]
## [1,] 0.000000 1.113519
## [2,] 1.113519 0.000000
```

Comparing with theorem 5.21, we have obtained

- $n_1 = 70$
- $n_2 = 17$
- $p = 9$
- $q = 6$
- $d^2 = 2.163477$
- $d_1^2 = 1.113519$

The correct answer is

$$\frac{70+17-9-1}{9-6}\frac{2.163477-1.113519}{(70+17)(70+17-2)/(70\cdot 17)+1.113519}$$

3.5 Incease in resubstituion specificity when going from LDA to QDA

Again, we can not solve the problem.

If you are interested to see how to compute the resubstitution specificity for the linear model, see below:

We once again consider the confusion matrices of the model.

```
##            LDpred$class
## glass$Type  1  2  3  5  6  7
##          1 46 14 10  0  0  0
##          2 16 41 12  4  3  0
##          3  3  3 11  0  0  0
##          5  0  2  0 10  0  1
##          6  1  1  0  0  7  0
##          7  0  1  1  2  1 24
```

We are interested in the specificity or the true negative rate of the models. This is defined on page 351 as

$$SPC = TNR = \frac{TN}{NN}$$

So we need to find the number of times the model did not predict an observation was 3 and was correct. A way to find this is to use that $NN = FP + FN + TP + TN$.

They can be counted manually or computed in R:

```
NN <- 214            # Total number of observations
LD_CM

##            LDpred$class
## glass$Type  1  2  3  5  6  7
##          1 46 14 10  0  0  0
##          2 16 41 12  4  3  0
##           3   3   3 11   0   0   0
##           5   0   2   0 10   0   1
##           6   1   1   0   0   7   0
##           7   0   1   1   2   1  24
```

```
FP <- sum(LD_CM[,3])-LD_CM[3,3] # Where guess=3 but Type!=3
FN <- sum(LD_CM[3,])-LD_CM[3,3] # Where guess!=3 but Type=3 '
TP <- LD_CM[3,3]                # Where guess=Type=3
TN1 <- NN - FP - FN - TP

SCP1 <- TN1/NN

SCP1
```

```
## [1] 0.8130841
```

4 PCA and Factor analysis

Data is the same as in section 3, except we remove types 4 and 6.

4.1 Correlation test

First, we compute the correlations (see page 34).

```
C <- cor(glass[c("Mg","Ba","Al")])
C
```

```
##              Mg         Ba         Al
## Mg  1.0000000 -0.5250485 -0.5302028
## Ba -0.5250485  1.0000000  0.4907969
## Al -0.5302028  0.4907969  1.0000000
```

```
R <- (C[1,2]-C[1,3]*C[2,3])/sqrt((1-C[1,3]**2)*(1-C[2,3]**2))
R
```

```
## [1] -0.35849
```

Next, we compute the test-statistic from theorem 1.37:

```
# theorem 1.37
dim(glass) #find number of variables:

## [1] 205   9

n <- 205  # number of observations
p <- 10   # number of variables
m <- 9    # p-m = number of variables we condition on

testT <-  R/sqrt(1-R^2)*sqrt(n-2-(p-m))
testT

## [1] -5.457865
```

The answer is **-5.4580** .

4.2 PCA - components needed to account for 90% variance

We peform a PCA analysis on the data. Note that we start be removing the "Type" variable.

```
glass <- glass[c("Na", "Mg", "Al", "Si", "K", "Ca", "Ba", "Fe")]

Eigs <- eigen(cor(glass))

var <- Eigs$values
varProp <- var/sum(var)
cumu = rep(NA,length(var))
for (i in 1:length(var)){
  cumu[i] = sum(varProp[1:i])
}
results <- data.frame("eigenvalues"=var,"proportion"=varProp,
                      "cumulative" = cumu)
```

Printing the dataframe, we obtain:

```
## [1] "Eigenvalues of the Correlation Matrix:"
##   eigenvalues   proportion cumulative
## 1 2.270318488 0.2837898110  0.2837898
## 2 1.588162562 0.1985203203  0.4823101
## 3 1.303578212 0.1629472765  0.6452574
## 4 1.164256682 0.1455320852  0.7907895
## 5 0.904243767 0.1130304709  0.9038200
## 6 0.459771844 0.0574714805  0.9612914
## 7 0.307945145 0.0384931432  0.9997846
## 8 0.001723299 0.0002154124  1.0000000
```

The answer is **5** components.

4.3 Test equality of last two eigenvalues

We use theorem 6.8 – we are using the correlation matrix, so we need to compute Z_2. $n = 205$ as before.

```
k <- 8 # total number of eigenvalues
m <- 6 # k-m = number of eigenvalues we test

lambdas <- Eigs$values[7:8]
lhat <- sum(lambdas)/(k-m)

Z <- -n*log(prod(lambdas)/lhat**(k-m))
Z

## [1] 781.1623
```

The answer is **781.16**.

4.4 Variance in Al explained by unrotated factor 1 and 2

First, we perform factor analysis using the function `principal` from the psych package.

```
#### Factor analysis 4 Factors ####
#unrotated factors
fa <- psych::principal(cor(glass),nfactors = 4,rotate = "none")
fal <- fa$loadings[,1:4]
```

```
#rotated factors:
rfa <-psych::principal(cor(glass),nfactors = 4, rotate = "varimax", scores = T)
rfal <- rfa$loadings[,1:4]

Factors <- data.frame("FA" = fal,"Rot_FA" = rfal)
```

We print the loadings of unrotated factor one and two:

	FA.PC1	FA.PC2
Al	0.8093665	0.2007839

The variance in Al explained by these two factors is computed as

$$0.8093665^2 + 0.2007839^2$$

```
## [1] 0.6953883
```

The answer is **0.6954**.

4.5 Interpretation of the rotated factors

We print the four rotated factors:

```
Factors[,5:8]

##     Rot_FA.RC1 Rot_FA.RC2  Rot_FA.RC3  Rot_FA.RC4
## Na  0.32611921  0.7234953 -0.26010357  0.37152243
## Mg -0.92752970  0.2058985  0.12294188  0.12417329
## Al  0.72413864  0.2119368  0.49683103 -0.07092262
## Si  0.01911454  0.1135380 -0.07184815 -0.96377396
## K   0.04524753 -0.1360537  0.91186022  0.09700902
## Ca  0.22179202 -0.7443236 -0.55398937  0.10443620
## Ba  0.75451029  0.2977496 -0.01502855  0.17830848
## Fe -0.07148905 -0.5224218  0.06759816  0.18468256
```

- Factor 1 is mainly a contrast between Al and Ba vs. Mg
- Factor 2 is mainly a contrast between Ca and Fe vs. Na
- Factor 3 is mainly a contrast between K and Ca

- Factor 4 is mainly Si

5 MANOVA

We are working with the "StudenrsPerformance.csv" data set.

```
grades <- read.csv("StudentsPerformance.csv")
print("Dimensions:")

## [1] "Dimensions:"

dim(grades)

## [1] 100   6

head(grades)

##   X         lunch parental.level.of.education math.score reading.score
## 1 1      standard           bachelor's degree         72            72
## 2 2      standard                some college         69            90
## 3 3      standard             master's degree         90            95
## 4 4 free/reduced          associate's degree         47            57
## 5 5      standard                some college         76            78
## 6 6      standard          associate's degree         71            83
##   writing.score
## 1            74
## 2            88
## 3            93
## 4            44
## 5            75
## 6            78
```

5.1 Usual test for parental effect.

We fit the linear model and use the `Ano.fit` function from the `car` library.

```
dep_vars <- cbind(grades$math.score,grades$reading.score,grades$writing.score)
lmfit <- manova(dep_vars ~ lunch+parental.level.of.education ,data=grades)

Ano.fit <- car::Anova(lmfit,test = 'Wilks')
Ano.fit
```

```
##
## Type II MANOVA Tests: Wilks test statistic
##                              Df test stat approx F num Df den Df   Pr(>F)
## lunch                         1   0.85464   5.2157      3   92.0 0.002268 **
## parental.level.of.education  4   0.85460   1.2426     12  243.7 0.254445
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value for testing parental effect is 0.254445. The answer is $]\mathbf{0.1, 0.5}]$.

5.2 Q1

We find on page 305-306, that Q1 is the residual variation, i.e. the error SSCP matrix.

```
Q1 <- Ano.fit$SSPE
Q1
```

```
##          [,1]     [,2]     [,3]
## [1,] 20363.54 17783.05 18667.91
## [2,] 17783.05 20911.69 20941.91
## [3,] 18667.91 20941.91 22742.52
```

The correct answer is

$$\begin{pmatrix} 20363.54 & 17783.05 & 18667.91 \\ 17783.05 & 20911.69 & 20941.91 \\ 18667.91 & 20941.91 & 22742.52 \end{pmatrix}$$

5.3 The usual test statistic for lunch effect when considering only *math* and *reading*.

This is a case of two-way manova and we will use theorem 4.26. In order to do so, we need thee SSCP matrix corresponding to lunch.

```
print("Type III SSCP matrix for lunch")
```

```
## [1] "Type III SSCP matrix for lunch"
```

```
Q3 <- Ano.fit$SSP$lunch
Q3

##           [,1]     [,2]     [,3]
## [1,] 3074.723 2903.689 2722.253
## [2,] 2903.689 2742.169 2570.825
## [3,] 2722.253 2570.825 2410.188
```

Finally, we may compute the test statistic, considering only the effect of *math* and *reading*.

```
det(Q1[1:2,1:2])/(det(Q1[1:2,1:2]+Q3[1:2,1:2]))

## [1] 0.8666419
```

The answer is **0.8666**.

6 Conditional mean and expectation of multivariate random variables

6.1 Squared correlation between Y_1 and Y_2

We use the results from section 1.1.3.

$$\text{Corr}(Y_1, Y_2)^2 = \frac{\text{Cov}(Y_1, Y_2)}{\text{Var}(Y_1)\text{Var}(Y_2)} = \frac{1}{2 \cdot 4} = \frac{1}{8}$$

6.2 Conditional mean $E\left(Y_1 | X_2 = x_2\right)$

Consider the submatrix

$$\Sigma_{y_1 x_2} = \begin{pmatrix} 2 & 0 \\ 0 & 4 \end{pmatrix}$$

and use theorem 1.27.

$$E\left(Y_1 | X_2 = x_2\right) = 0 + 0 \cdot 2^{-1}(x_2 - 1) = 0$$

6.3 Conditional mean $E\left(Y_1|X_1 = x_1\right)$

Consider the submatrix

$$\Sigma_{y_1x_1} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

and use theorem 1.27.

$$E\left(Y_1|X_1 = x_1\right) = 0 + 1 \cdot 2^{-1}(x_1 - 1) = \frac{1}{2}x_1 - \frac{1}{2}$$

6.4 Conditional mean $E\left(Y_1|X_1 = x_1, X_2 = x_2\right)$

Consider the submatrix

$$\Sigma_{y_1x_1x_2} = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 4 \end{pmatrix}$$

and use theorem 1.27.

$$\begin{aligned} E\left(Y_1|X_1 = x_2, X_2 = x_2\right) &= 0 + \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 1 & 4 \end{pmatrix}^{-1} \begin{pmatrix} x_1 - 1 \\ x_2 - 1 \end{pmatrix} \\ &= \frac{1}{7} \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 4 & -1 \\ -1 & 2 \end{pmatrix} \begin{pmatrix} x_1 - 1 \\ x_2 - 1 \end{pmatrix} \\ &= \frac{1}{7} \begin{pmatrix} 4 & -1 \end{pmatrix} \begin{pmatrix} x_1 - 1 \\ x_2 - 1 \end{pmatrix} \\ &= \frac{4}{7}x_1 - \frac{1}{7}x_2 - \frac{3}{7} \end{aligned}$$

6.5 Squared partial correlation between Y_1 and Y_2 given X_1

We use theorem 1.27 to find the conditional dispersion matrix:

$$D\left(\boldsymbol{Y}|X_1 = x_1\right) = \begin{pmatrix} 2 & 1 \\ 1 & 4 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} 2^{-1} \begin{pmatrix} 1 & 0 \end{pmatrix}$$

Solution with R:

```
Sigma <- matrix(c(2,1,1,0,1,4,0,1,1,0,2,1,0,1,1,4),ncol=4)

Sigma11 <- Sigma[1:2,1:2]
Sigma22 <- Sigma[3,3]
Sigma12 <- Sigma[1:2,3]

D <- Sigma11- (Sigma12 %*% t(Sigma12))/Sigma22
```

Then we can compute the squared partial correlation:

```
rho12_sq <- D[1,2]^2/(D[1,1]*D[2,2]) #

rho12_sq

## [1] 0.1666667

1/6

## [1] 0.1666667
```

The answer is $\mathbf{\frac{1}{6}}$.

6.6 Squared partial correlation between Y_1 and Y_2 given X_1 and X_2

We use theorem 1.27 to find the conditional dispersion matrix:

$$D\left(\boldsymbol{Y}|\boldsymbol{X}=\boldsymbol{x}\right)=\begin{pmatrix}2&1\\1&4\end{pmatrix}-\begin{pmatrix}1&0\\0&1\end{pmatrix}\begin{pmatrix}2&1\\1&4\end{pmatrix}^{-1}\begin{pmatrix}1&0\\0&1\end{pmatrix}$$

Solution with R:

```
Sigma11 <- Sigma[1:2,1:2]
Sigma22 <- Sigma[3:4,3:4]
Sigma12 <- Sigma[1:2,3:4]

D <- Sigma11- (Sigma12 %*% solve(Sigma22) %*% t(Sigma12))
```

Then we can compute the squared partial correlation:

```
rho12_sq <- D[1,2]^2/(D[1,1]*D[2,2]) #

rho12_sq
```

```
## [1] 0.2461538
```

```
64/260
```

```
## [1] 0.2461538
```

The answer is $\mathbf{\frac{64}{260}}$.

6.7 Squared multiple correlation between Y_1 and $(X_1, X_2)^T$

We use theorem 1.42 with

$$1 - \frac{\det \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 4 \end{pmatrix}}{2 \cdot \det \begin{pmatrix} 2 & 1 \\ 1 & 4 \end{pmatrix}}$$

```
Sigmai <- Sigma[c(1,3,4),c(1,3,4)]
sigmaii <- Sigma[1,1]
Sigmaxx <- Sigma[3:4,3:4]

rho_sq <- 1-det(Sigmai)/(sigmaii*det(Sigmaxx))
rho_sq
```

```
## [1] 0.2857143
```

```
2/7
```

```
## [1] 0.2857143
```

The answer is $\mathbf{\frac{2}{7}}$.