Assignment2

Michael Nefiodovas(22969312) Carmen Leong(22789943)
Nicholas Choong(21980614)

Question 1

(a) Why should you not automatically scale the data prior to a PCA or FA? Restrict your answer to one or two concise sentences.

Scaling can possibly cause information loss especially for variables with high magnitude. If the variables with high magnitude are important, scaling is not advised since scaling centres all the variables and transforms their variability and range to more comparable ranges. Moreover, if the variables in the data set have same units of measurement, scaling is not necessarily required.

(b) The dataset ass2pop.csv is available in the LMS folder 'Data sets'. For a description of the data see Assignment 1. Here we work with a part of the dataset only. Let Σ be the covariance matrix consisting of rows 1:11, and columns 3:13. Read the data into R. The value for $\Sigma[1, 1]$ should be 0.8266. In your answer show the R commands you use to calculate the following and show the results stating clearly what each part is.

```
ass2pop <- read.csv("ass2pop.csv", header = FALSE)
S0 <- as.matrix(ass2pop[1:11, 3:13])
S0
```

```
##           V3        V4        V5        V6        V7       V8       V9      V10
## 1   0.826600  0.085098  0.813860  0.941140 -0.094576 0.157220 0.413020 0.544060
## 2   0.085098  0.772170  0.087477  0.095301 -0.113380 0.029254 0.055516 0.013586
## 3   0.813860  0.087477  0.808930  0.928340 -0.042251 0.235750 0.469730 0.581750
## 4   0.941140  0.095301  0.928340  1.093100 -0.082746 0.181890 0.485200 0.627510
## 5  -0.094576 -0.113380 -0.042251 -0.082746  0.803680 0.619900 0.535770 0.435710
## 6   0.157220  0.029254  0.235750  0.181890  0.619900 1.045000 0.815470 0.658660
## 7   0.413020  0.055516  0.469730  0.485200  0.535770 0.815470 0.885550 0.756200
## 8   0.544060  0.013586  0.581750  0.627510  0.435710 0.658660 0.756200 0.784750
## 9  -0.070067 -0.079740 -0.012993 -0.085199  0.522590 0.719410 0.510570 0.396520
## 10 -0.379050 -0.058024 -0.302510 -0.413400  0.722830 0.829310 0.471130 0.274940
## 11  0.723150  0.042291  0.731960  0.892100  0.142650 0.321660 0.544810 0.615920
##          V11       V12       V13
## 1  -0.070067 -0.379050  0.723150
## 2  -0.079740 -0.058024  0.042291
## 3  -0.012993 -0.302510  0.731960
## 4  -0.085199 -0.413400  0.892100
## 5   0.522590  0.722830  0.142650
```

```
## 6   0.719410  0.829310  0.321660
## 7   0.510570  0.471130  0.544810
## 8   0.396520  0.274940  0.615920
## 9   1.016400  0.691920  0.125210
## 10  0.691920  1.150600 -0.113900
## 11  0.125210 -0.113900  1.548100
```

The covariance matrix is a 11x11 square matrix giving the covariance between each pair of the first 11 variables from the first population.

i. the eigenvalues of Σ;

```
ev <- eigen(S0)
vals <- ev$values
vals
```

```
##  [1] 4.8414428891 3.5168110273 0.8049944895 0.6695186459 0.4372377465
##  [6] 0.2438244571 0.1239747301 0.0465762465 0.0421426530 0.0078666975
## [11] 0.0004904174
```

The eigenvalues of the covariance matrix encode the variability of the data in an orthogonal basis that captures as much of the data's variability as possible.

ii. the matrix $\Sigma^{2/3}$;

```
V1 <- ev$vectors
vals2 <- diag(vals^(2 / 3))
S1 <- V1 %*% vals2 %*% t(V1)
S1
```

```
##              [,1]         [,2]        [,3]        [,4]        [,5]        [,6]
##  [1,]  0.55097423  0.043824977  0.53202284  0.59708083 -0.08208317 0.07478956
##  [2,]  0.04382498  0.835844641  0.04746383  0.04894243 -0.08508635 0.03004535
##  [3,]  0.53202284  0.047463830  0.53086756  0.58531690 -0.05511226 0.13214522
##  [4,]  0.59708083  0.048942432  0.58531690  0.72776400 -0.06855236 0.07585321
##  [5,] -0.08208317 -0.085086348 -0.05511226 -0.06855236  0.70145652 0.31976753
##  [6,]  0.07478956  0.030045346  0.13214522  0.07585321  0.31976753 0.77108082
##  [7,]  0.21924834  0.044517479  0.25893374  0.26069436  0.32016690 0.49253446
##  [8,]  0.31900542 -0.001977267  0.34065592  0.35716789  0.27873243 0.37373762
##  [9,] -0.05286721 -0.054618851 -0.02149901 -0.07249309  0.27348623 0.41329665
## [10,] -0.24791854 -0.025569342 -0.19959764 -0.26035371  0.43838671 0.52914125
## [11,]  0.36579932  0.014650738  0.37040230  0.47900278  0.07636199 0.13912266
##             [,7]         [,8]        [,9]       [,10]       [,11]
##  [1,] 0.21924834  0.319005420 -0.05286721 -0.24791854  0.36579932
##  [2,] 0.04451748 -0.001977267 -0.05461885 -0.02556934  0.01465074
##  [3,] 0.25893374  0.340655923 -0.02149901 -0.19959764  0.37040230
##  [4,] 0.26069436  0.357167887 -0.07249309 -0.26035371  0.47900278
##  [5,] 0.32016690  0.278732427  0.27348623  0.43838671  0.07636199
##  [6,] 0.49253446  0.373737622  0.41329665  0.52914125  0.13912266
```

```
##  [7,] 0.65414626  0.453457265  0.27138848  0.26158871  0.26347006
##  [8,] 0.45345726  0.565724190  0.21461427  0.14675864  0.29956013
##  [9,] 0.27138848  0.214614273  0.88768892  0.38079110  0.05894989
## [10,] 0.26158871  0.146758645  0.38079110  0.87810393 -0.06151348
## [11,] 0.26347006  0.299560127  0.05894989 -0.06151348  1.21280678
```

iii. the matrix $2\Sigma^{-1/4}\Sigma\Sigma^{-1/4}$ and its eigenvalues

```
vals3 <- diag(vals^(-1 / 4))
S2 <- V1 %*% vals3 %*% t(V1)
mat <- 2 * S2 %*% S0 %*% S2
mat
```

```
##              [,1]        [,2]        [,3]        [,4]       [,5]       [,6]
##  [1,]  0.95373655  0.05761053  0.88179563  0.94316103 -0.1413759 0.09789139
##  [2,]  0.05761053  1.74377277  0.06508920  0.06457208 -0.1381399 0.05401604
##  [3,]  0.88179563  0.06508920  0.90866337  0.92222858 -0.1081073 0.20097407
##  [4,]  0.94316103  0.06457208  0.92222858  1.26786709 -0.1115446 0.08286691
##  [5,] -0.14137589 -0.13813987 -0.10810731 -0.11154459  1.4128098 0.41380042
##  [6,]  0.09789139  0.05401604  0.20097407  0.08286691  0.4138004 1.43627830
##  [7,]  0.29775815  0.07633015  0.36715267  0.36335471  0.4753268 0.74261589
##  [8,]  0.47496012 -0.01210738  0.50692054  0.51509129  0.4388716 0.53754681
##  [9,] -0.08184375 -0.08471910 -0.03929485 -0.12183553  0.3642407 0.59189457
## [10,] -0.38155096 -0.02937148 -0.30798393 -0.38557858  0.6491559 0.81751402
## [11,]  0.47984888  0.01407658  0.48733943  0.66371355  0.1078890 0.16435516
##             [,7]        [,8]        [,9]       [,10]       [,11]
##  [1,] 0.29775815  0.47496012 -0.08184375 -0.38155096  0.47984888
##  [2,] 0.07633015 -0.01210738 -0.08471910 -0.02937148  0.01407658
##  [3,] 0.36715267  0.50692054 -0.03929485 -0.30798393  0.48733943
##  [4,] 0.36335471  0.51509129 -0.12183553 -0.38557858  0.66371355
##  [5,] 0.47532679  0.43887159  0.36424069  0.64915594  0.10788904
##  [6,] 0.74261589  0.53754681  0.59189457  0.81751402  0.16435516
##  [7,] 1.24740659  0.68401995  0.36766788  0.35964420  0.33917846
##  [8,] 0.68401995  1.07939051  0.29850212  0.19877858  0.38448716
##  [9,] 0.36766788  0.29850212  1.74592985  0.52326068  0.07726484
## [10,] 0.35964420  0.19877858  0.52326068  1.63512469 -0.08004537
## [11,] 0.33917846  0.38448716  0.07726484 -0.08004537  2.22935626
```

```
vals4 <- eigen(mat)$values
vals4
```

```
##  [1] 4.40065581 3.75063249 1.79442970 1.63648238 1.32247911 0.98757168
##  [7] 0.70420091 0.43163061 0.41057352 0.17738881 0.04429074
```

Question 2

Consider the abalone data. We want to compare the performance of linear regression and PCR for the raw abalone data following the description given in Q3 of Lab 3. In the analysis we use the predictor variables Length, Height, Whole Weight, Shucked Weight, Viscera Weight and Dried-Shell Weight and we consider Rings as the response variable. Hint. Note the change of predictor variables used in Q2 compared to the variables in the Lab.

```
coln <- c(
  "Sex", #      nominal         M, F, and I (infant)
  "Length", #       continuous  mm  Longest shell measurement
  "Diameter", #     continuous  mm  perpendicular to length
  "Height", #       continuous  mm  with meat in shell
  "Whole_weight", #     continuous  grams   whole abalone
  "Shucked_weight", #   continuous  grams   weight of meat
  "Viscera_weight", #   continuous  grams   gut weight (after bleeding)
  "Shell_weight", #     continuous  grams   after being dried
  "Rings" #         integer         +1.5 gives the age in years
)
abalone <- read_csv(file = "./abalone.csv", col_names = coln)
```

```
## Rows: 4177 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Sex
## dbl (8): Length, Diameter, Height, Whole_weight, Shucked_weight, Viscera_wei...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
abalone$Sex <- as.factor(abalone$Sex)
summary(abalone)
```

```
##  Sex          Length         Diameter          Height        Whole_weight
##  F:1307   Min.   :0.075   Min.   :0.0550   Min.   :0.0000   Min.   :0.0020
##  I:1342   1st Qu.:0.450   1st Qu.:0.3500   1st Qu.:0.1150   1st Qu.:0.4415
##  M:1528   Median :0.545   Median :0.4250   Median :0.1400   Median :0.7995
##           Mean   :0.524   Mean   :0.4079   Mean   :0.1395   Mean   :0.8287
##           3rd Qu.:0.615   3rd Qu.:0.4800   3rd Qu.:0.1650   3rd Qu.:1.1530
##           Max.   :0.815   Max.   :0.6500   Max.   :1.1300   Max.   :2.8255
##  Shucked_weight   Viscera_weight    Shell_weight        Rings
##  Min.   :0.0010   Min.   :0.0005   Min.   :0.0015   Min.   : 1.000
##  1st Qu.:0.1860   1st Qu.:0.0935   1st Qu.:0.1300   1st Qu.: 8.000
##  Median :0.3360   Median :0.1710   Median :0.2340   Median : 9.000
##  Mean   :0.3594   Mean   :0.1806   Mean   :0.2388   Mean   : 9.934
##  3rd Qu.:0.5020   3rd Qu.:0.2530   3rd Qu.:0.3290   3rd Qu.:11.000
##  Max.   :1.4880   Max.   :0.7600   Max.   :1.0050   Max.   :29.000
```

```
summary
```

```
## function (object, ...)
## UseMethod("summary")
## <bytecode: 0x119b2f660>
## <environment: namespace:base>
```

```
# Regression
big.lm <- lm(Rings ~ Length + Height + Whole_weight + Shucked_weight + Viscera_weight + Shell_weight, da
summary(big.lm)
```

```
##
## Call:
## lm(formula = Rings ~ Length + Height + Whole_weight + Shucked_weight +
##     Viscera_weight + Shell_weight, data = abalone)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.1753  -1.3558  -0.4047   0.9204  14.1281
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.9788     0.2702  11.023  < 2e-16 ***
## Length           8.1964     0.8125  10.088  < 2e-16 ***
## Height          12.8998     1.5440   8.355  < 2e-16 ***
## Whole_weight     9.3558     0.7355  12.721  < 2e-16 ***
## Shucked_weight -20.2996     0.8266 -24.558  < 2e-16 ***
## Viscera_weight -10.1086     1.3086  -7.725 1.39e-14 ***
## Shell_weight     9.3257     1.1345   8.220 2.68e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.227 on 4170 degrees of freedom
## Multiple R-squared:  0.5236, Adjusted R-squared:  0.5229
## F-statistic: 763.8 on 6 and 4170 DF,  p-value: < 2.2e-16
```

(a) For the regular linear regression use forward selection and state the order in which the variables are chosen. Calculate the residual standard deviation for each number of predictors. Hint. you may make use of the code in Lab 3.

```
glancerows <- data.frame()
fm.fwd <- fm.null <- lm(Rings ~ 1, abalone)
summary(lm(fm.fwd))
```

```
##
## Call:
## lm(formula = fm.fwd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -8.9337 -1.9337 -0.9337  1.0663 19.0663
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.93368    0.04989   199.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.224 on 4176 degrees of freedom
```

```
row1 <- data.frame(modelno = 0, variable = "Intercept", sigma = glance(lm(fm.fwd))$sigma)
glancerows <- rbind(glancerows, row1)
add1(fm.fwd, big.lm, test = "F")
```

```
## Single term additions
##
## Model:
## Rings ~ 1
##                Df Sum of Sq   RSS    AIC F value    Pr(>F)
## <none>                      43411 9780.8
## Length          1   13454.5 29956 8233.3 1875.17 < 2.2e-16 ***
## Height          1   13490.7 29920 8228.2 1882.48 < 2.2e-16 ***
## Whole_weight    1   12676.8 30734 8340.3 1722.07 < 2.2e-16 ***
## Shucked_weight  1    7689.9 35721 8968.4  898.79 < 2.2e-16 ***
## Viscera_weight  1   11019.1 32392 8559.8 1420.27 < 2.2e-16 ***
## Shell_weight    1   17097.2 26313 7691.7 2712.72 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fm.fwd <- update(fm.fwd, . ~ . + Shell_weight)
row1 <- data.frame(modelno = 1, variable = "Shell_weight", sigma = glance(lm(fm.fwd))$sigma)
glancerows <- rbind(glancerows, row1)
add1(fm.fwd, big.lm, test = "F")
```

```
## Single term additions
##
## Model:
## Rings ~ Shell_weight
##                Df Sum of Sq   RSS    AIC  F value    Pr(>F)
## <none>                      26313 7691.7
## Length          1       9.9 26304 7692.1   1.5726    0.2099
## Height          1     259.3 26054 7652.3  41.5374 1.288e-10 ***
## Whole_weight    1    1740.8 24573 7407.8 295.7039 < 2.2e-16 ***
## Shucked_weight  1    3476.1 22837 7101.9 635.3248 < 2.2e-16 ***
## Viscera_weight  1    1067.0 25246 7520.8 176.4105 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fm.fwd <- update(fm.fwd, . ~ . + Shucked_weight)
row1 <- data.frame(modelno = 2, variable = "Shucked_weight", sigma = glance(lm(fm.fwd))$sigma)
glancerows <- rbind(glancerows, row1)
add1(fm.fwd, big.lm, test = "F")
```

```
## Single term additions
##
## Model:
## Rings ~ Shell_weight + Shucked_weight
##                Df Sum of Sq   RSS    AIC F value    Pr(>F)
## <none>                      22837 7101.9
## Length          1    978.17 21859 6921.0  186.74 < 2.2e-16 ***
## Height          1    802.26 22035 6954.5  151.93 < 2.2e-16 ***
## Whole_weight    1    803.77 22034 6954.2  152.23 < 2.2e-16 ***
## Viscera_weight  1     73.91 22763 7090.4   13.55 0.0002353 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fm.fwd <- update(fm.fwd, . ~ . + Length)
row1 <- data.frame(modelno = 3, variable = "Length", sigma = glance(lm(fm.fwd))$sigma)
glancerows <- rbind(glancerows, row1)
add1(fm.fwd, big.lm, test = "F")
```

```
## Single term additions
##
## Model:
## Rings ~ Shell_weight + Shucked_weight + Length
##                Df Sum of Sq   RSS    AIC  F value Pr(>F)
## <none>                      21859 6921.0
## Height          1    375.05 21484 6850.8  72.8318 <2e-16 ***
## Whole_weight    1    563.47 21296 6814.0 110.3882 <2e-16 ***
## Viscera_weight  1      3.59 21856 6922.4   0.6844 0.4081
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fm.fwd <- update(fm.fwd, . ~ . + Whole_weight)
row1 <- data.frame(modelno = 4, variable = "Whole_weight", sigma = glance(lm(fm.fwd))$sigma)
glancerows <- rbind(glancerows, row1)
add1(fm.fwd, big.lm, test = "F")
```

```
## Single term additions
##
## Model:
## Rings ~ Shell_weight + Shucked_weight + Length + Whole_weight
##                Df Sum of Sq   RSS  AIC F value    Pr(>F)
## <none>                      21296 6814
## Height          1    318.39 20977 6753  63.307 2.261e-15 ***
## Viscera_weight  1    268.17 21028 6763  53.194 3.597e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fm.fwd <- update(fm.fwd, . ~ . + Height)
row1 <- data.frame(modelno = 5, variable = "Height", sigma = glance(lm(fm.fwd))$sigma)
glancerows <- rbind(glancerows, row1)
add1(fm.fwd, big.lm, test = "F")
```

```
## Single term additions
```

```
##
## Model:
## Rings ~ Shell_weight + Shucked_weight + Length + Whole_weight +
##     Height
##                Df Sum of Sq   RSS    AIC F value    Pr(>F)
## <none>                      20977 6753.0
## Viscera_weight  1    295.96 20681 6695.7  59.674 1.393e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
fm.fwd <- update(fm.fwd, . ~ . + Viscera_weight)
row1 <- data.frame(modelno = 6, variable = "Viscera_weight", sigma = glance(lm(fm.fwd))$sigma)
glancerows <- rbind(glancerows, row1)
glancerows
```

```
##   modelno       variable    sigma
## 1       0      Intercept 3.224169
## 2       1   Shell_weight 2.510500
## 3       2 Shucked_weight 2.339087
## 4       3         Length 2.288719
## 5       4   Whole_weight 2.259299
## 6       5         Height 2.242614
## 7       6 Viscera_weight 2.227005
```

```
glancerows1 <- data.frame(modelno = glancerows$modelno, sigma = glancerows$sigma)
rownames(glancerows1) <- glancerows$variable
glancerows1
```

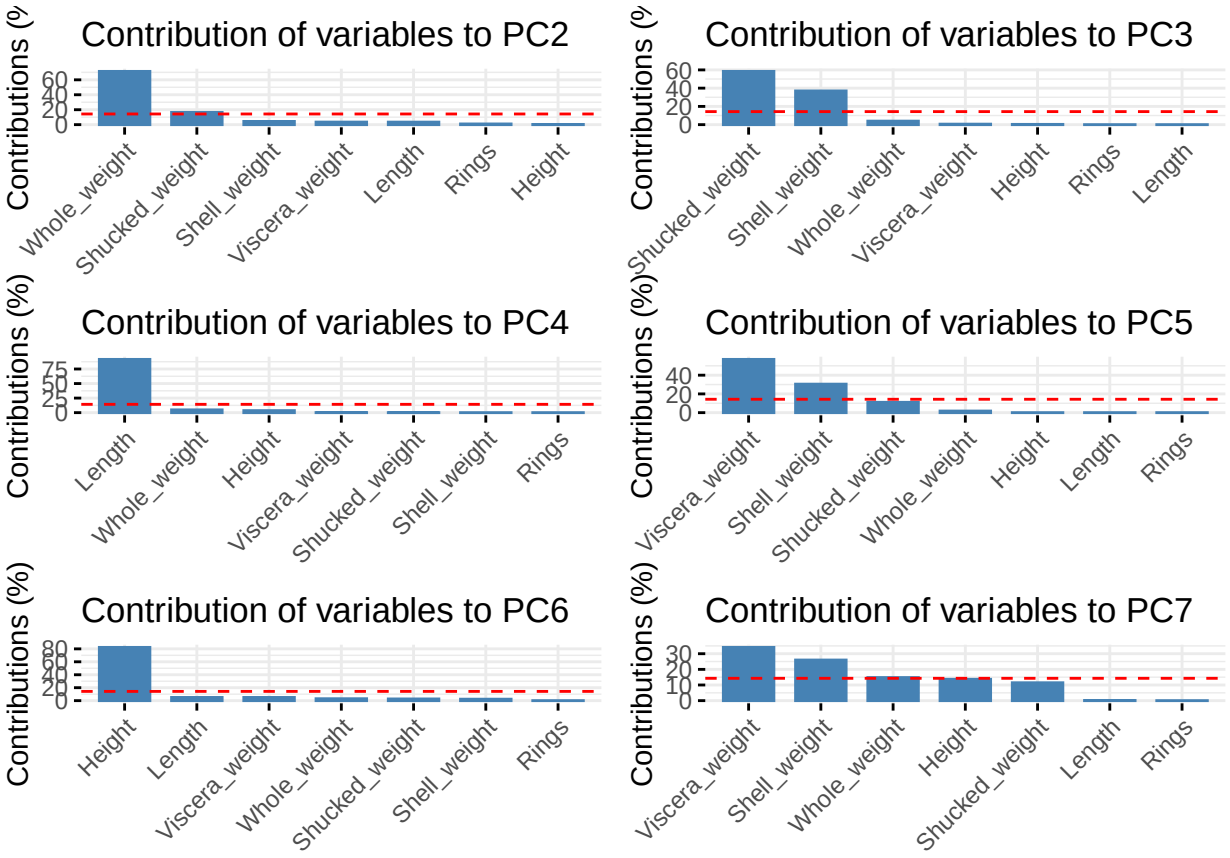
```
##                modelno    sigma
## Intercept            0 3.224169
## Shell_weight         1 2.510500
## Shucked_weight       2 2.339087
## Length               3 2.288719
## Whole_weight         4 2.259299
## Height               5 2.242614
## Viscera_weight       6 2.227005
```

Shell_weight, Shucked_weight, Length, Whole_weight, Height, Viscera_weight

(b) Carry out PCR on the raw data using the same variables and response as in part (a). For each additional principal component you add to the regression model as predictor, calculate the residual standard deviation and list which of the variables has the heighest absolute weight in the respective principal component.

```
abalone1 <- dplyr::select(
  abalone,
  Rings,
  Shell_weight,
  Shucked_weight,
  Length,
```

```
  Whole_weight,
  Height,
  Viscera_weight
)
abalone_pr <- prcomp(abalone1, scale = F)

pc2 <- fviz_contrib(abalone_pr, choice = "var", axes = 2) +
  ggtitle("Contribution of variables to PC2")

pc3 <- fviz_contrib(abalone_pr, choice = "var", axes = 3) +
  ggtitle("Contribution of variables to PC3")

pc4 <- fviz_contrib(abalone_pr, choice = "var", axes = 4) +
  ggtitle("Contribution of variables to PC4")

pc5 <- fviz_contrib(abalone_pr, choice = "var", axes = 5) +
  ggtitle("Contribution of variables to PC5")

pc6 <- fviz_contrib(abalone_pr, choice = "var", axes = 6) +
  ggtitle("Contribution of variables to PC6")

pc7 <- fviz_contrib(abalone_pr, choice = "var", axes = 7) +
  ggtitle("Contribution of variables to PC7")

figure <- ggarrange(pc2, pc3, pc4, pc5, pc6, pc7,
  ncol = 2, nrow = 3
)
figure
```




```
set.seed(1000)
pcr_model <- pcr(Rings ~ ., data = abalone1, scale = F, validation = "CV")
summary(pcr_model)
```

```
## Data:    X dimension: 4177 6
##  Y dimension: 4177 1
## Fit method: svdpc
## Number of components considered: 6
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           3.225    2.728    2.264    2.256    2.253    2.246    2.253
## adjCV        3.225    2.728    2.264    2.256    2.253    2.248    2.251
##
## TRAINING: % variance explained
##        1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## X        97.71    98.82    99.42    99.73    99.87   100.00
## Rings    28.46    50.72    51.11    51.29    51.38    52.36
```

```
var_contribution <- c("", "Whole_weight", "Shucked_weight", "Length", "Viscera_weight", "Height", "Visc
pcr_df <- data.frame(var_contribution, RMSEP(pcr_model)$val[1, , ])
colnames(pcr_df) <- c("variable", "pcr")
pcr_df
```

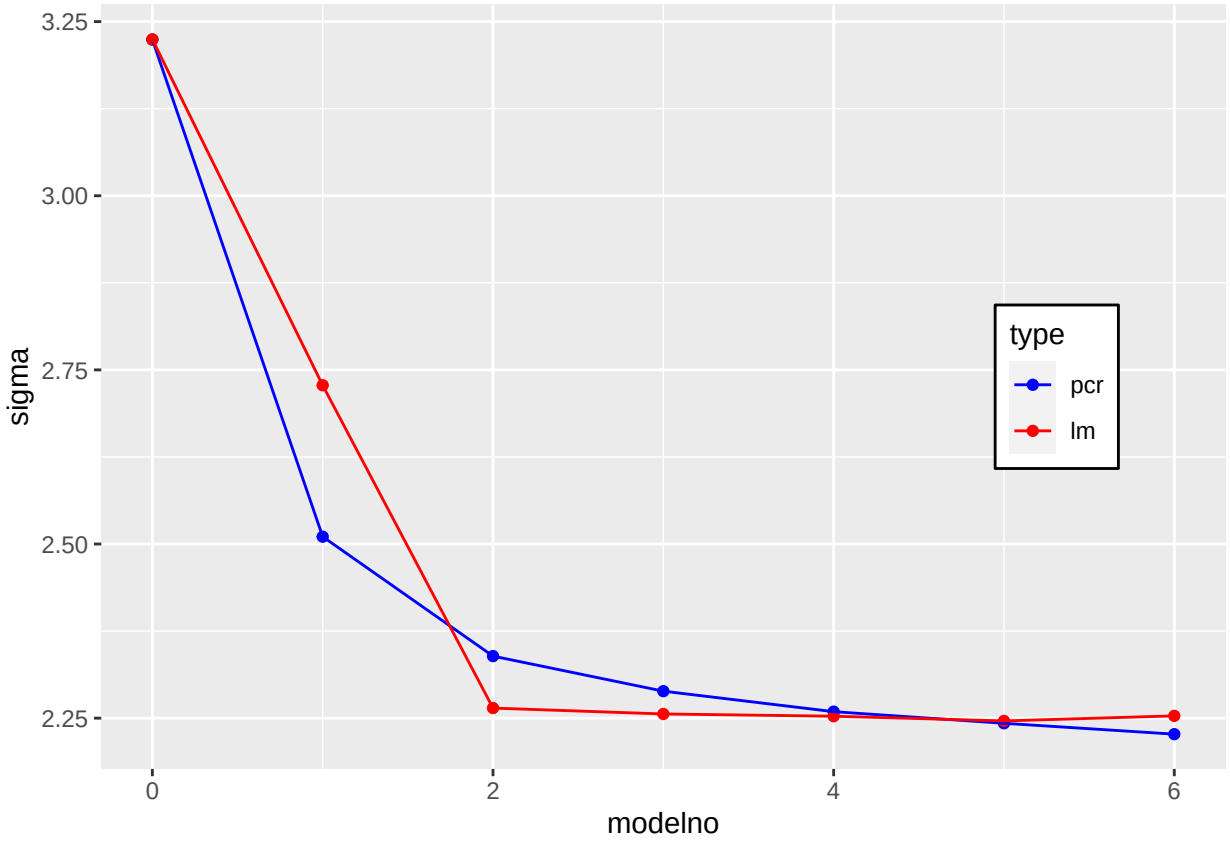
```
##                   variable      pcr
## (Intercept)                 3.224555
## 1 comps       Whole_weight 2.727867
## 2 comps     Shucked_weight 2.264461
## 3 comps             Length 2.255978
## 4 comps     Viscera_weight 2.252774
## 5 comps             Height 2.246019
## 6 comps     Viscera_weight 2.253278
```

(c) In a single graph show plots of residual standard deviation resulting from your models on the y -axis against the number of variables/PC components on the x-axis.

```
rsd_df <- data.frame(modelno = glancerows1$modelno, lm = glancerows1$sigma, pcr = pcr_df$pcr)
rsd_df
```

```
##   modelno       lm      pcr
## 1       0 3.224169 3.224555
## 2       1 2.510500 2.727867
## 3       2 2.339087 2.264461
## 4       3 2.288719 2.255978
## 5       4 2.259299 2.252774
## 6       5 2.242614 2.246019
## 7       6 2.227005 2.253278
```

```
rsd_df2 <- tidyr::pivot_longer(rsd_df, -modelno, names_to = "type", values_to = "value")
ggplot(rsd_df2, aes(x = modelno, y = value, color = type, group = type)) +
  geom_point() +
  geom_line() +
  xlab("modelno") +
  ylab("sigma") +
  scale_color_manual(
    labels = c("pcr", "lm"),
    values = c("blue", "red")
  ) +
  theme(
    legend.position = c(0.85, 0.5),
    legend.background = element_rect(fill = "white", color = "black")
  )
```



(d) Explain why you do not require to a variable selection method when selecting the predictors in PCR.

(e) Comment on your findings and in particular on what approaches work better for these data and why.

Question 3

We consider the 13-dimensional wine recognition data of Example 4.6 and Lab 4. The data are available in the Data Sets folder. Here we want to compare a factor analysis of all observations with those obtained from cultivar 1 and cultivar 2. The cultivar membership of the observations is given in column 1 of the data set. For part of this analysis you may report the relevant results obtained in the lab. You may find it useful to create two data frames: one for the complete data and a separate one for the first two cultivars of the data. We refer to the latter as the *cultivar12* data. Hint. use the R command factanal from the stats library.

```
cultivar <- read.table(file = "wine.tsv", sep = ",")

cultivar12 <- cultivar[cultivar$V1 != 3, 2:14]
cultivar12
```

```
cultivar <- cultivar[, 2:14]
cultivar
```

(a) Scale the data and work with the scaled data. How many observations are in the cultivar12 data?

```
cultivar_scaled <- scale(cultivar, center = TRUE)
cultivar_scaled

cultivar12_scaled <- scale(cultivar12, center = TRUE)
cultivar12_scaled
```

There are 130 observations in the cultivar12 data.

(b) Separately for the complete and for the cultivar12 data, carry out, display and report the results of the following:

i. Calculate the sample covariance matrix of the scaled data and the eigenvalues of this matrix. What is the value of $\hat{\sigma}^2$ for k=2? How different are the values of $\hat{\sigma}^2$ for the complete and the two cultivar12 datasets? Hint. You may use the information Box 6.7 in your calculations.

```
S1 <- cov(cultivar_scaled)
val1 <- eigen(S1)$values
val1
```

```
##  [1] 4.7058503 2.4969737 1.4460720 0.9189739 0.8532282 0.6416570 0.5510283
##  [8] 0.3484974 0.2888799 0.2509025 0.2257886 0.1687702 0.1033779
```

```
S2 <- cov(cultivar12_scaled)
val2 <- eigen(S2)$values
val2
```

```
##  [1] 4.76329126 1.80266171 1.51098222 1.17760632 0.86441690 0.72558092
##  [7] 0.60146583 0.42569389 0.36925807 0.29315154 0.19816224 0.19072623
## [13] 0.07700288
```

```
sigma_hat_sq1 <- (1 / (13 - 2)) * (sum(val1[3:13]))
sigma_hat_sq2 <- (1 / (13 - 2)) * (sum(val2[3:13]))

sigma_hat_sq1
```

```
## [1] 0.527016
```

```
sigma_hat_sq2
```

```
## [1] 0.5849134
```

The value of $\hat{\sigma}^2$ for the complete dataset is 0.57897 lower that the one for cultivar12 dataset.

ii. Calculate and list the factor loadings for the 2-factor principal axis factoring using the value of $\hat{\sigma}^2$ calculated in the previous part.

```
# Factor loading for complete cultivar dataset
Om1 <- diag(rep(sigma_hat_sq1, 13))
S_A1 <- S1 - Om1

eig_A1 <- eigen(S_A1)

Gamma_hat_1 <- eig_A1$vectors[, 1:2]
Lambda_hat_1 <- diag(eig_A1$values[1:2]^(1 / 2))
Ahat1 <- Gamma_hat_1 %*% Lambda_hat_1
Ahat1
```

```
##              [,1]        [,2]
##  [1,] -0.29504100 -0.67883001
##  [2,]  0.50121729 -0.31570222
##  [3,]  0.00419282 -0.44361896
##  [4,]  0.48922349  0.01486432
##  [5,] -0.29026293 -0.42055185
##  [6,] -0.80677348 -0.09128633
##  [7,] -0.86457063  0.00471567
##  [8,]  0.61026726 -0.04039350
##  [9,] -0.64071874 -0.05516200
## [10,]  0.18115202 -0.74387639
## [11,] -0.60654976  0.39192100
## [12,] -0.76896884  0.23087893
## [13,] -0.58618456 -0.51216004
```

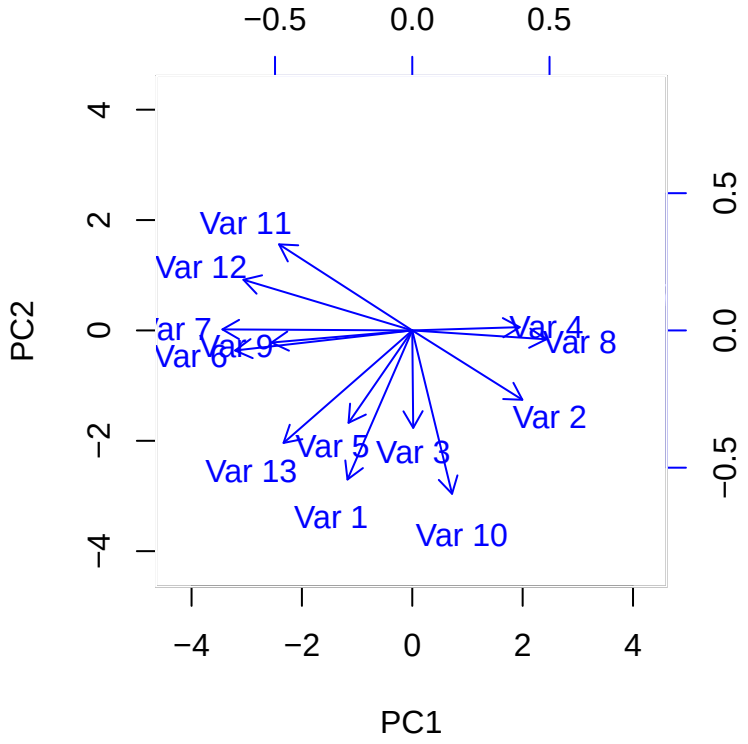
```
# Factor loading for cultivar12 datasets
Om2 <- diag(rep(sigma_hat_sq2, 13))
S_A2 <- S2 - Om2

eig_A2 <- eigen(S_A2)

Gamma_hat_2 <- eig_A2$vectors[, 1:2]
Lambda_hat_2 <- diag(eig_A2$values[1:2]^(1 / 2))
Ahat2 <- Gamma_hat_2 %*% Lambda_hat_2
Ahat2
```

```
##               [,1]         [,2]
##  [1,] -0.731306139  0.191368509
##  [2,] -0.001599091 -0.527327970
##  [3,] -0.297809477 -0.391927572
##  [4,]  0.395377590 -0.525165136
##  [5,] -0.459622750  0.002654609
##  [6,] -0.766785883 -0.147231579
##  [7,] -0.807245450 -0.202261191
##  [8,]  0.505090569 -0.059404908
##  [9,] -0.497678257 -0.236450111
## [10,] -0.782286987  0.183898642
## [11,] -0.021329329  0.362116482
```

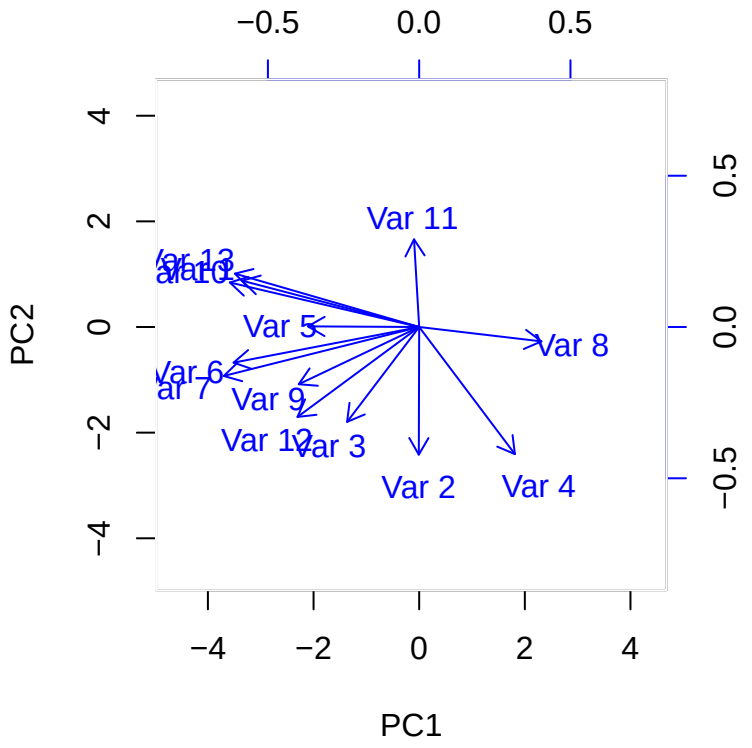
```
## [12,] -0.502888280 -0.371551565
## [13,] -0.761294450  0.220492655
```

iii. Show biplots of the factor loadings.

```
wine_pr1 <- prcomp(cultivar_scaled, scale = F)
wine_pr2 <- prcomp(cultivar12_scaled, scale = F)

biplot(wine_pr1$x, Ahat1, col = c("white", "blue"))
```




```
biplot(wine_pr2$x, Ahat2, col = c("white", "blue"))
```



iv. Compare the results obtained from the complete data and the cultivar12 data and comment on the main differences, similarities etc.

The eigenvalues of complete data and cultivar12 data are very similar. Hence, the values of $\hat{\sigma}^2$ for both data are very similar as $\hat{\sigma}^2$ is calculated based on the eigenvalues. The eigenvectors and the factor loadings for the complete data and cultivar12 data are quite different. The factor loadings for both dataset differ in terms of absolute value, relative order by size and the sign. This difference can also be seen in the biplots where the variables are grouped differently and have different angles.

(c) We next turn to ML factor loadings and testing. In your calculations use the option "none" for rotation. If you use other commands, you may not achieve full marks for this question. Separately for the complete and for the cultivar12 data, carry out, display and report the results of the following:

i. Calculate the factor loadings for the 2-factor ML without rotation. List your factor loadings and show biplots of the factor loadings.

```
fa1 <- factanal(cultivar_scaled, factors = 2, rotation = "none")$loadings
fa2 <- factanal(cultivar12_scaled, factors = 2, rotation = "none")$loadings

fa1
```

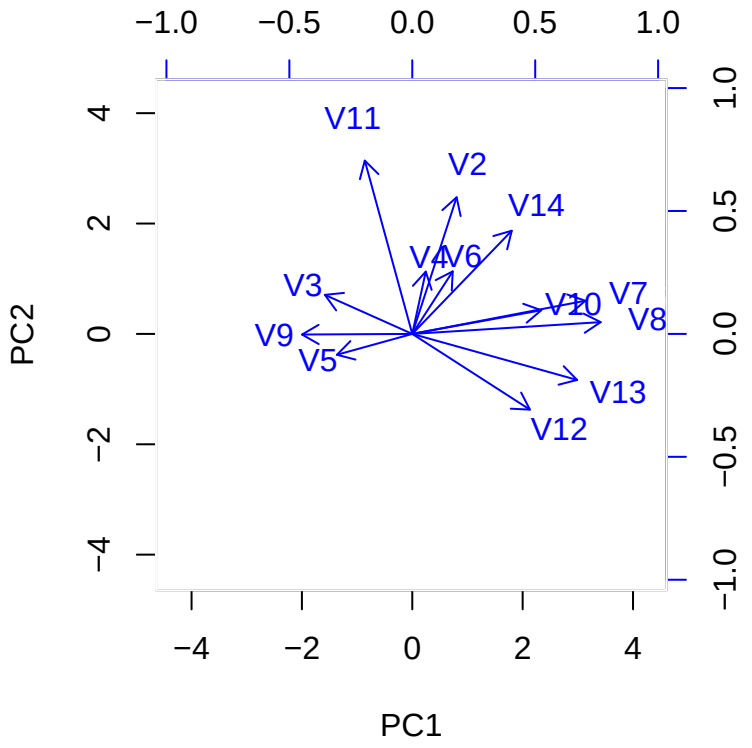
```
##
```

```
## Loadings:
##     Factor1 Factor2
## V2   0.225   0.695
## V3  -0.444   0.198
## V4           0.317
## V5  -0.383  -0.106
## V6   0.206   0.318
## V7   0.880   0.168
## V8   0.958
## V9  -0.561
## V10  0.656   0.120
## V11 -0.242   0.881
## V12  0.598  -0.385
## V13  0.838  -0.234
## V14  0.506   0.525
##
##                Factor1 Factor2
## SS loadings      4.253   2.036
## Proportion Var   0.327   0.157
## Cumulative Var   0.327   0.484
```

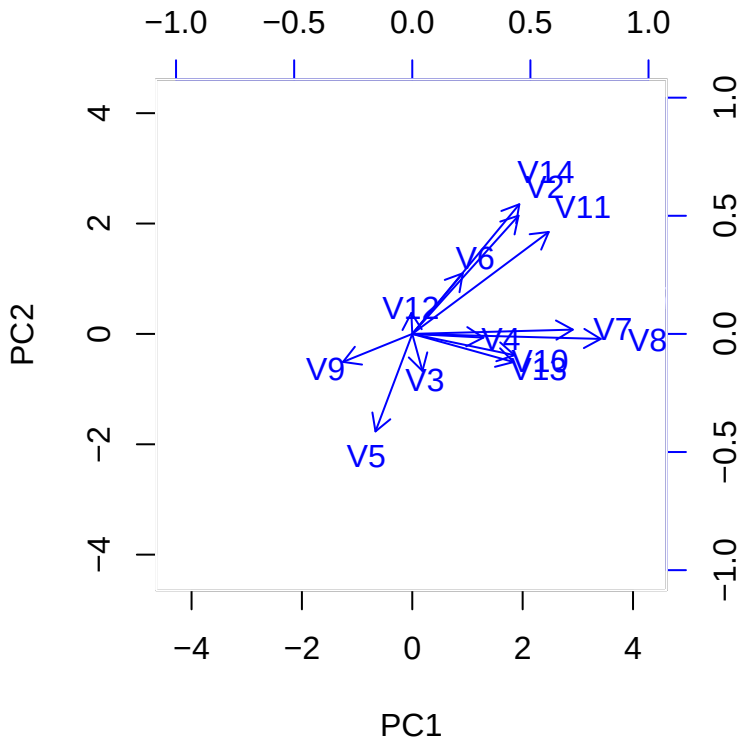
```
fa2
```

```
##
## Loadings:
##     Factor1 Factor2
## V2   0.562   0.626
## V3          -0.196
## V4   0.376
## V5  -0.195  -0.516
## V6   0.267   0.321
## V7   0.849
## V8   0.997
## V9  -0.366  -0.148
## V10  0.541  -0.113
## V11  0.722   0.539
## V12          0.111
## V13  0.536  -0.149
## V14  0.566   0.686
##
##                Factor1 Factor2
## SS loadings      3.842   1.631
## Proportion Var   0.296   0.125
## Cumulative Var   0.296   0.421
```

```
biplot(wine_pr1$x, fa1, col = c("white", "blue"))
```




```
biplot(wine_pr1$x, fa2, col = c("white", "blue"))
```



ii. Carry out a sequence of hypothesis tests starting with the one-factor model.

A. What is the largest number $\hat{\sigma}^2$ of factors you can test with these data? Why can we not exceed this number?

B. For each $k \leq k_{max}$, state the number of degrees of freedom of the χ^2 distribution, the limiting distribution of the test statistic $-2logLR_k$, and report the p-value for each set of tests.

C. What is the appropriate k-factor model for the complete and cultivar12 data?

iii. Compare the results of parts (b) and (c).

Question 4

Consider the Boston Housing data which are available from

library (MASS) Boston attach (Boston) In Lab 5 we used these data with the 11 variables shown in Table 7.3 of Chapter 7.

```
attach(Boston)
Boston
```

(a) Use the split of the 11 variables as in Q3 of Lab 5. Calculate canonical correlation scores. List the strength of the four correlations and show the four CC score plots corresponding to ($U_{\bullet j}$, $V_{\bullet j}$) for j = 1, . . . , 4.

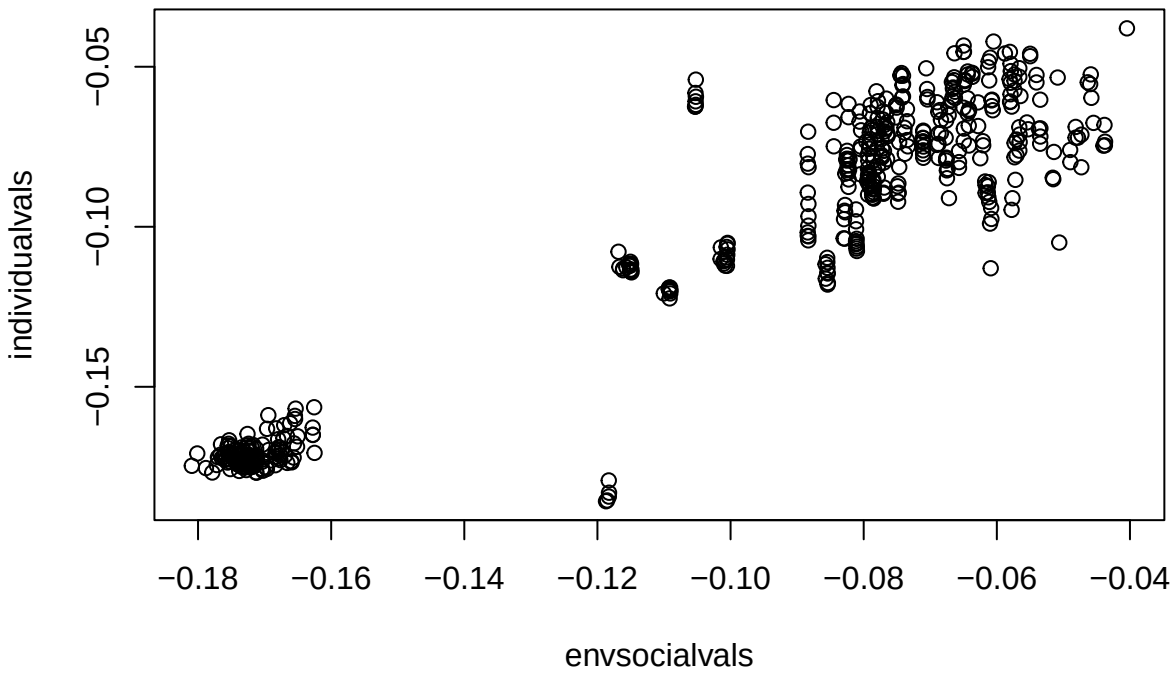
```
Boston.rearranged <- Boston %>% dplyr::select(
  "crim",
  "indus",
  "nox",
  "dis",
  "rad",
  "ptratio",
  "black",
  "rm",
  "age",
  "tax",
  "medv"
)
envsocial <- Boston.rearranged[, 1:7]
individual <- Boston.rearranged[, 8:11]

boston.CC <- cancor(envsocial, individual)

print("The strength of the four correlations: ")
```

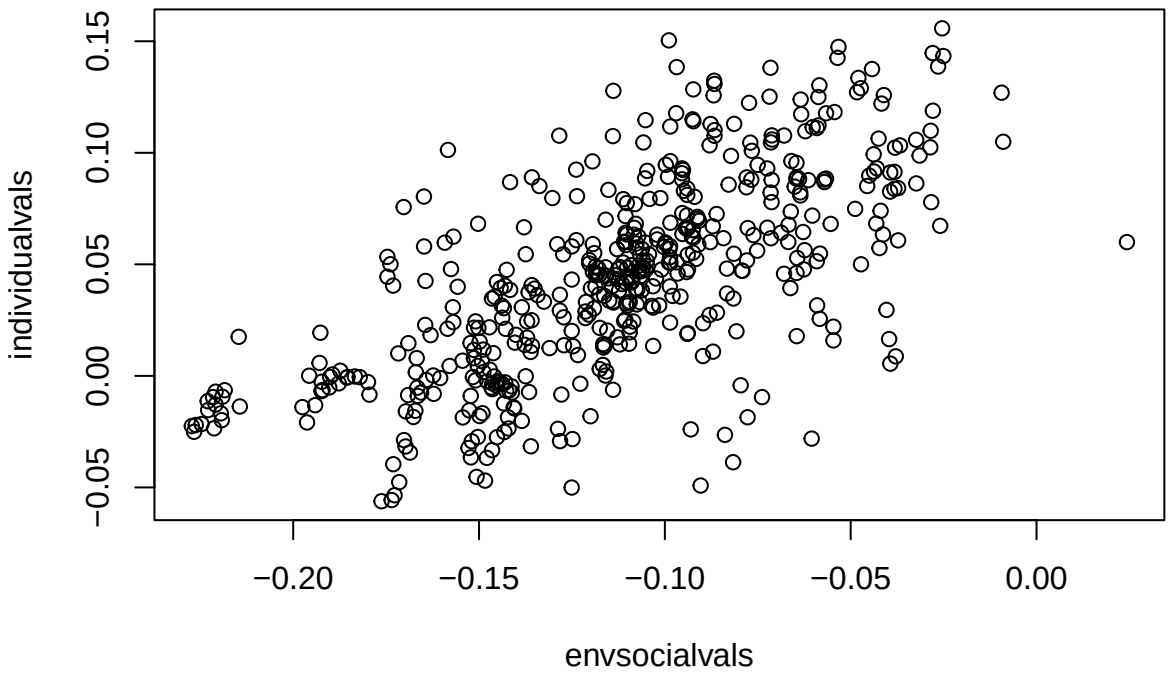
```
## [1] "The strength of the four correlations: "
```

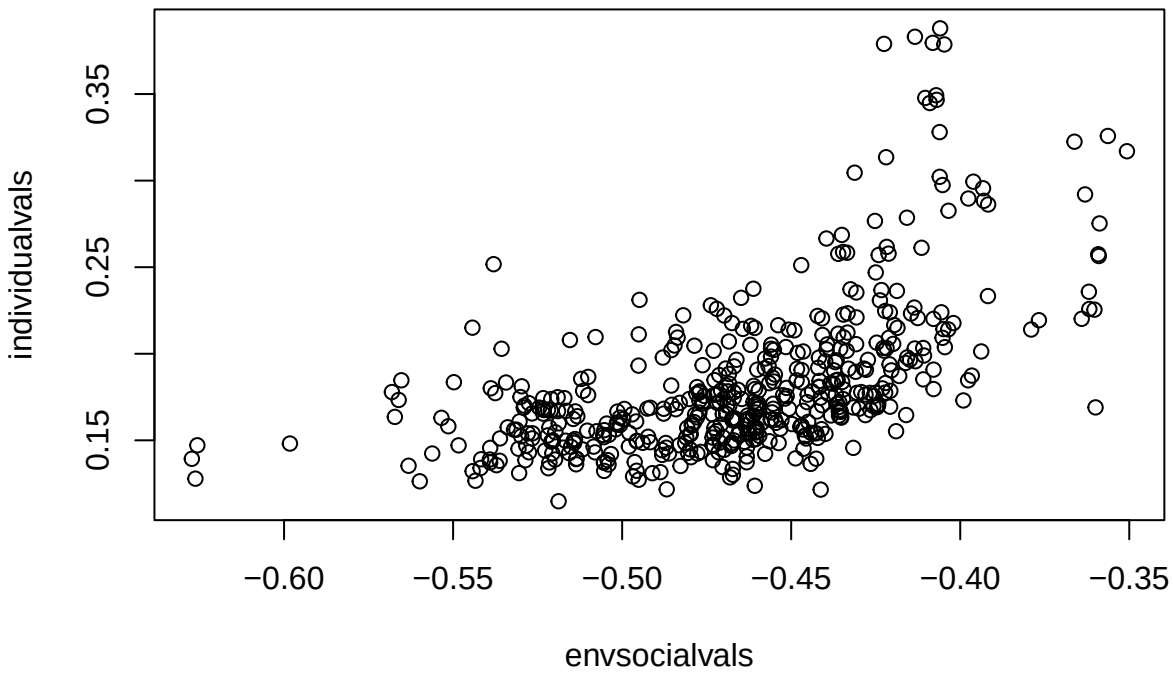
```
print(boston.CC$cor)
```

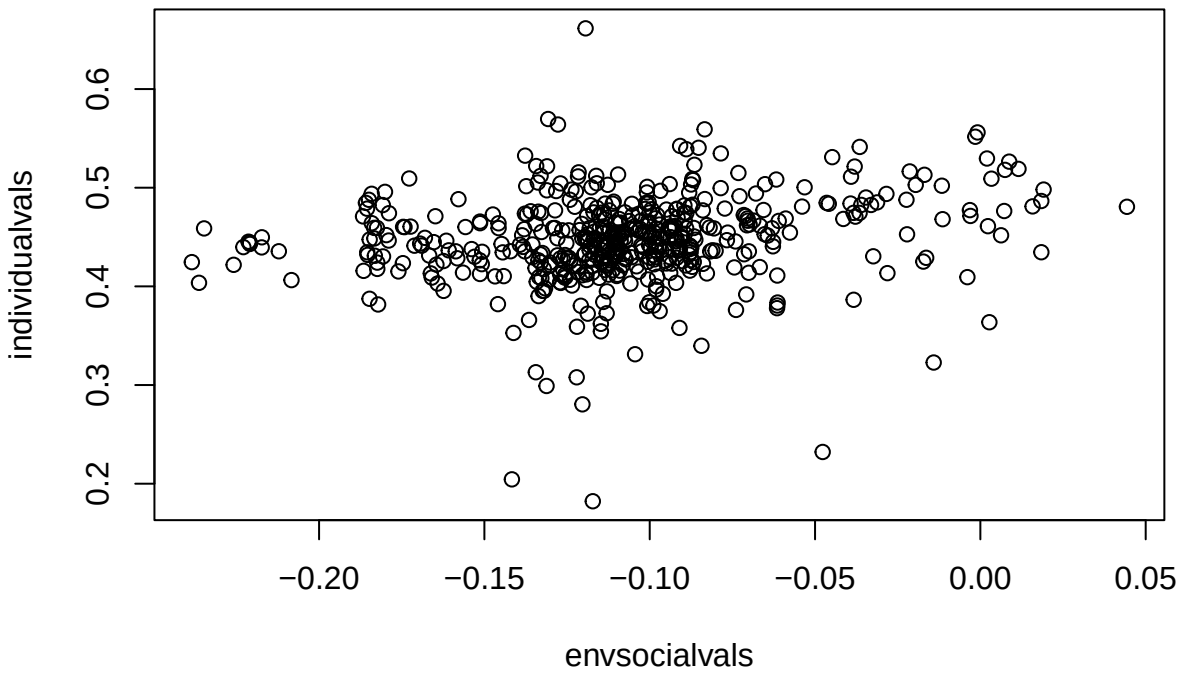
```
## [1] 0.9451239 0.6786623 0.5714338 0.2009740
```

```
for (i in 1:4) {
  envsocialvals <- as.matrix(envsocial) %*% as.matrix(boston.CC$xcoef[, i])

  individualvals <- as.matrix(individual) %*% as.matrix(boston.CC$ycoef[, i])

  plot(envsocialvals, individualvals)
}
```


individualvals
−0.05
−0.10
−0.15
−0.18
−0.16
−0.14
−0.12
−0.10
−0.08
−0.06
−0.04
envsocialvals


individualvals
0.15
0.10
0.05
0.00
−0.05
−0.20
−0.15
−0.10
−0.05
0.00
envsocialvals


individualvals
0.15
0.25
0.35
−0.60
−0.55
−0.50
−0.45
−0.40
−0.35
envsocialvals



(b) Comment on the plots and anything unusual you notice.

The first CC score plot has the unusual property that the data splits into two separate clusters.
The second and third CC score plots don't exhibit any interesting behavior.
The fourth CC score plot shows very little correlation.s

(c) Use all variables of the Boston Housing data other than chas and add the extra variables zn and lstat to the previous $X^{[2]}$ data to increase these to 6-dimensional data. Use the $X^{[1]}$ data of part (a). Repeat the calculations and graphics of part (a) for these data.

```
Boston.rearranged <- Boston %>% dplyr::select(
  "crim",
  "indus",
  "nox",
  "dis",
  "rad",
  "ptratio",
  "black",
  "rm",
  "age",
  "tax",
  "medv",
  "zn",
  "lstat"
)
```

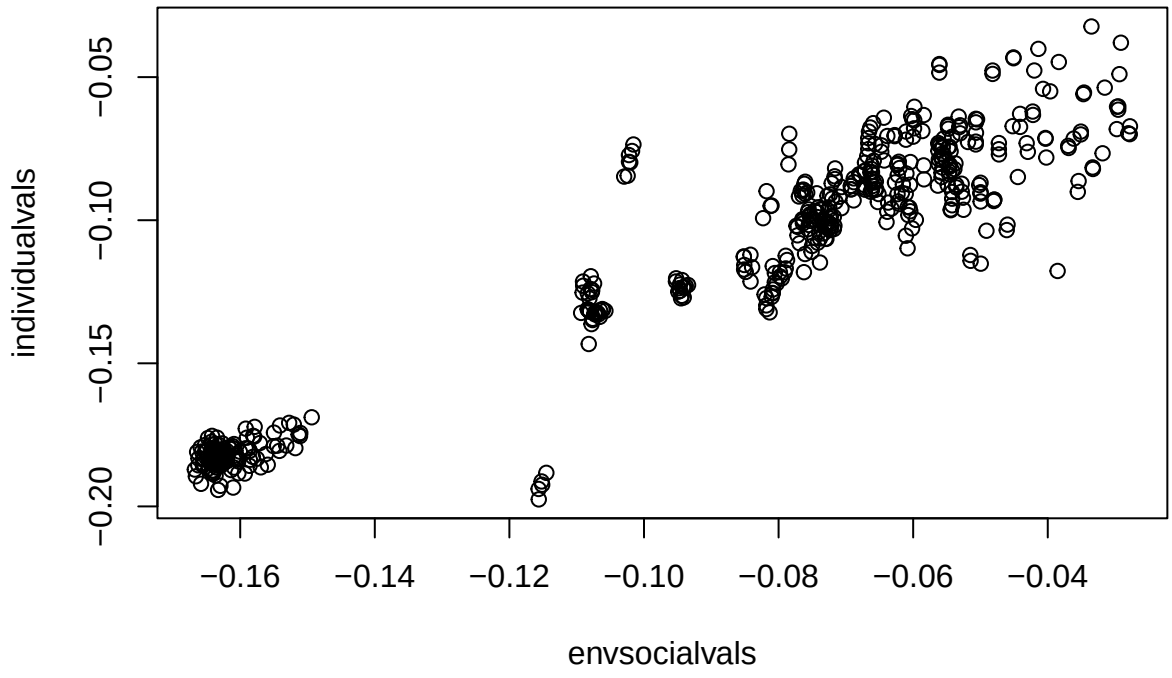
```
envsocial <- Boston.rearranged[, 1:7]
individual <- Boston.rearranged[, 8:13]

boston.CC <- cancor(envsocial, individual)

print("The strength of the four correlations: ")
```

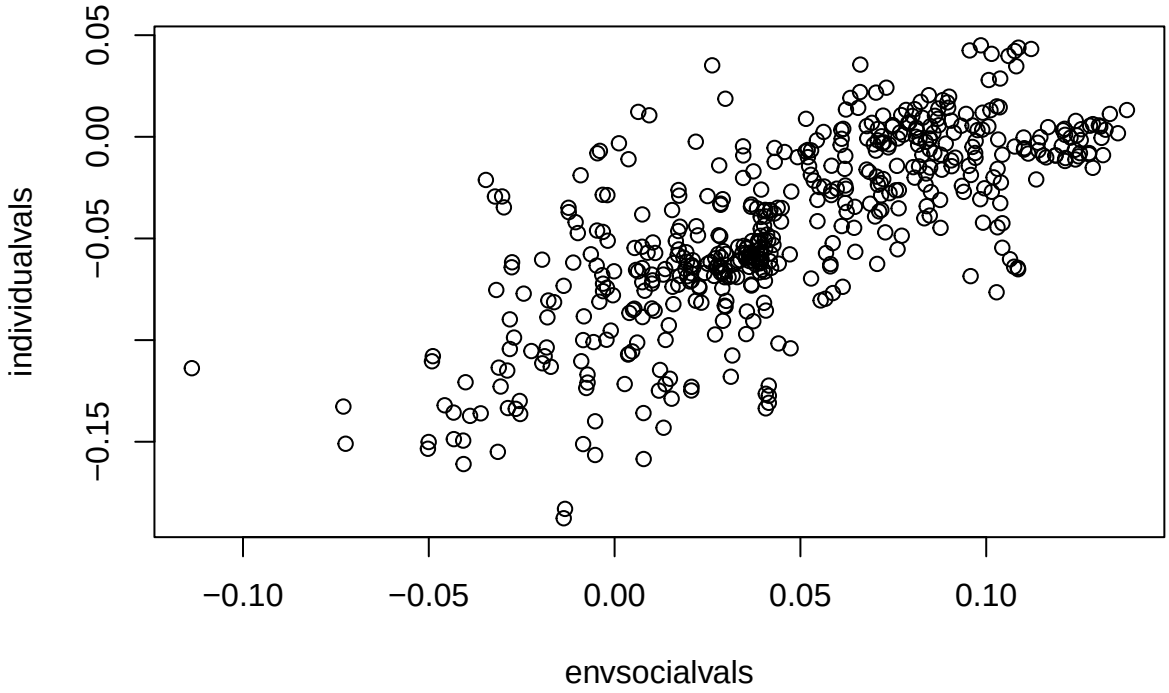
```
## [1] "The strength of the four correlations: "
```

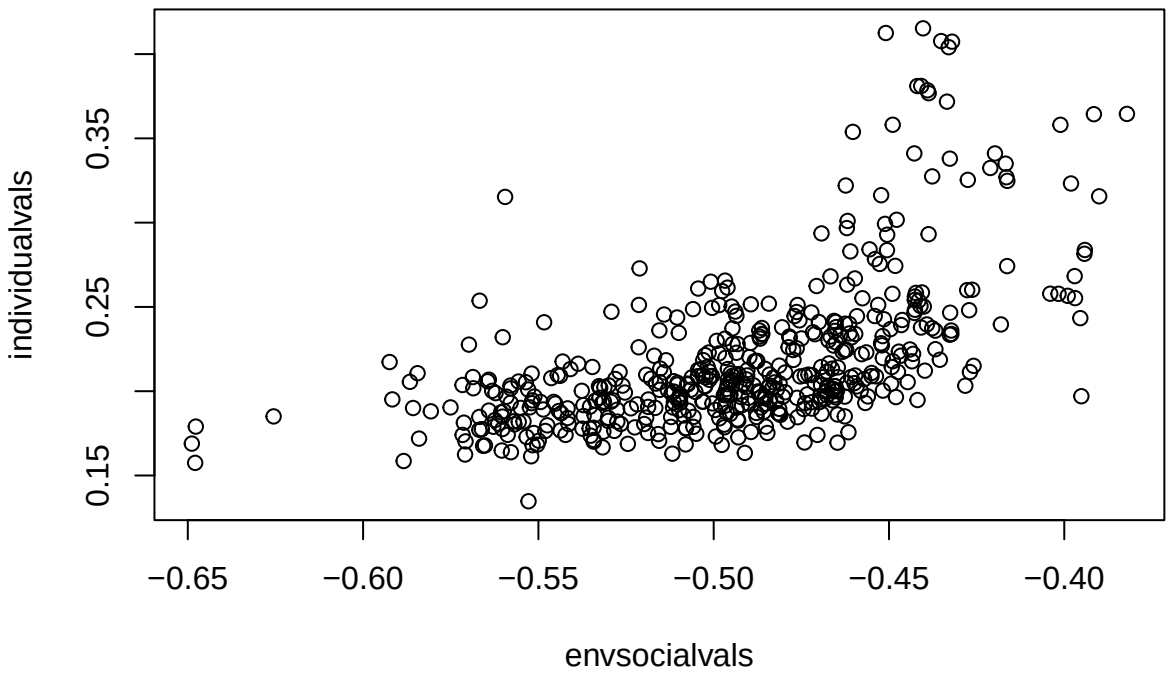
```
print(boston.CC$cor)
```

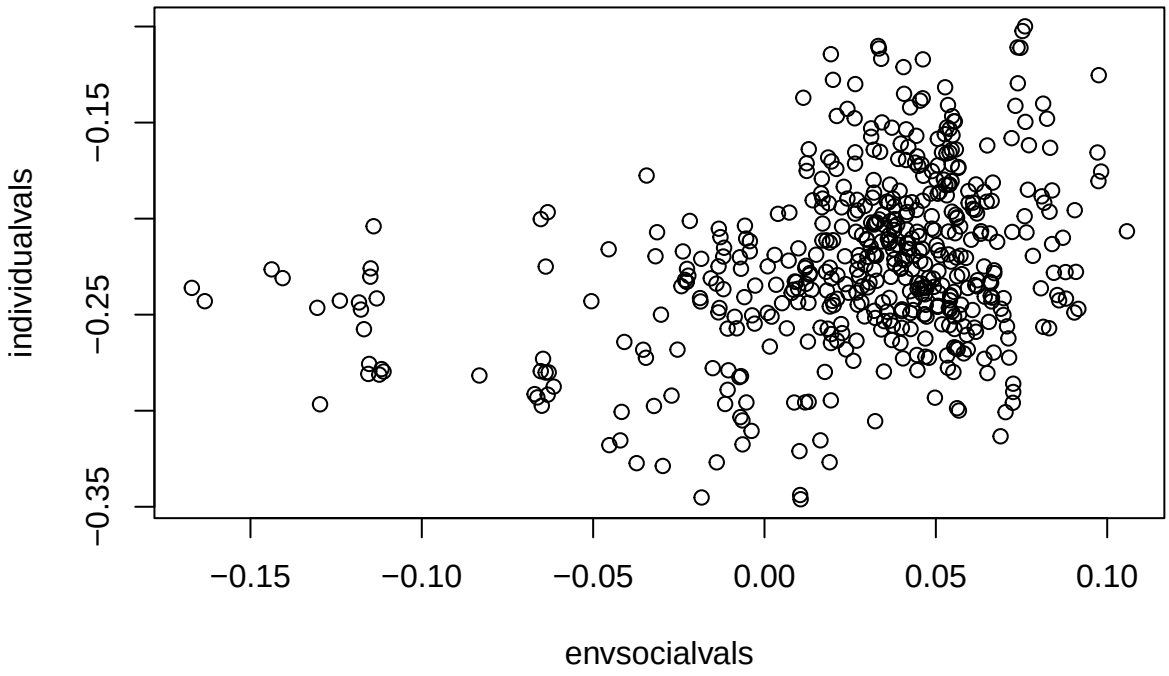
```
## [1] 0.9537171 0.7344798 0.5778651 0.3259343 0.2076161 0.1257320
```

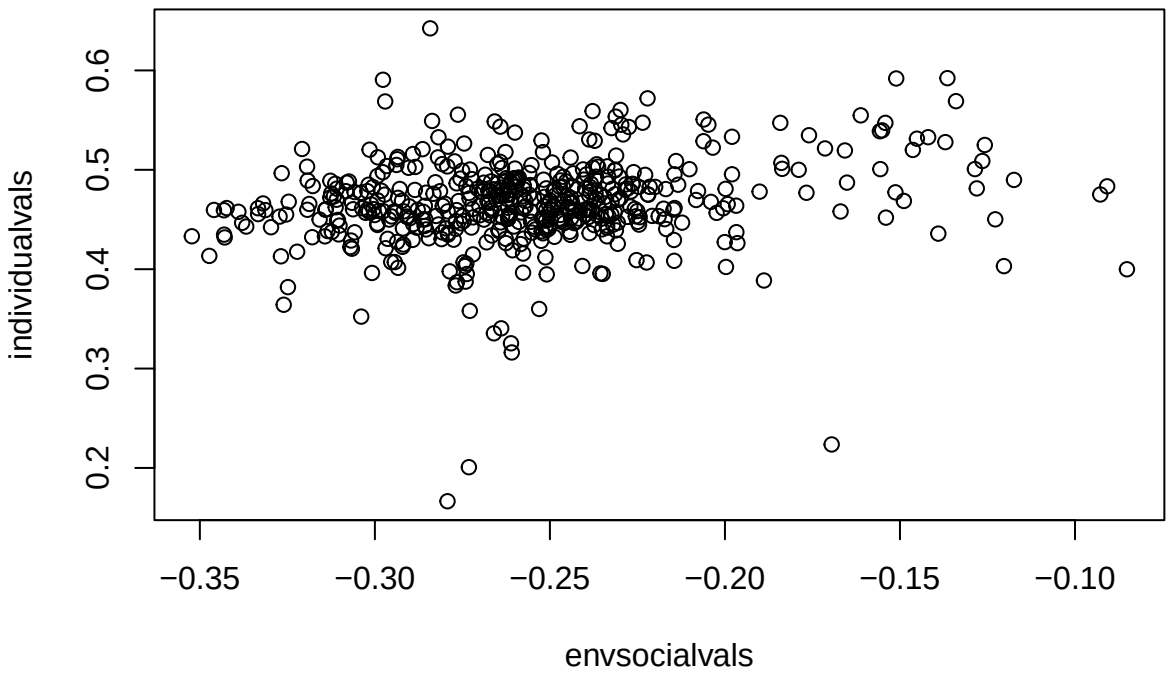
```
for (i in 1:6) {
  envsocialvals <- as.matrix(envsocial) %*% as.matrix(boston.CC$xcoef[, i])

  individualvals <- as.matrix(individual) %*% as.matrix(boston.CC$ycoef[, i])

  plot(envsocialvals, individualvals)
}
```

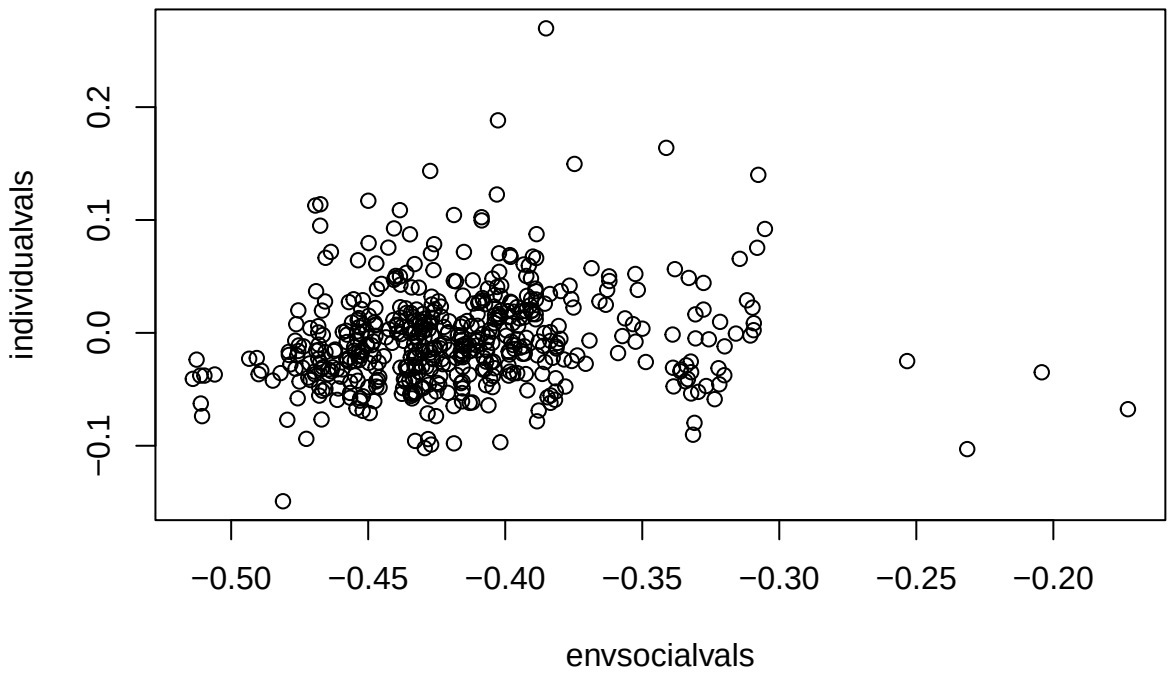



individualvals
0.05
0.00
−0.05
−0.15
−0.10
−0.05
0.00
0.05
0.10
envsocialvals


individualvals
0.15
0.25
0.35
−0.65
−0.60
−0.55
−0.50
−0.45
−0.40
envsocialvals


individualvals
−0.15
−0.25
−0.35
−0.15
−0.10
−0.05
0.00
0.05
0.10
envsocialvals


individualvals
0.2
0.3
0.4
0.5
0.6
−0.35
−0.30
−0.25
−0.20
−0.15
−0.10
envsocialvals



(d) Compare the results of parts (a) and (c) and comment on the differences and why they could occur.

Firstly, the correlation scores are uniformly higher in part (c). This is likely because the variance of the additional two variables can be used to find correlations in the X1 data.

There are also now 6 total pairs of CC scores. This is possible because the rank of the data matrix has increased to 6.

(e) Carry out a hypothesis test for the data described in part (c) using the statistic T_k of Lecture 5 and the values of the correlation strengths obtained in part (c). Calculate the p-values for each statistic and report these p-values.

```
Tk <- function(n, d1, d2, cor, k) {
  constant <- n - 0.5 * (d1 + d2 + 3)
  terms <- na.omit(1 - cor[k + 1:length(cor)]^2)
  logterm <- log(prod(terms))
  result <- -constant * logterm
  return(result)
}

n <- 506
d1 <- 7
d2 <- 6
for (k in 1:5) {
  tk <- Tk(n, d1, d2, c(boston.CC$cor), k)
```

```
  df <- (d1 - k) * (d2 - k)
  print(paste0("k=", k, " p-value = ", pchisq(tk, df, lower.tail = FALSE)))
}
```

```
## [1] "k=1 p-value = 1.10813747800098e-122"
## [1] "k=2 p-value = 2.08614517693533e-49"
## [1] "k=3 p-value = 3.18145252539475e-13"
## [1] "k=4 p-value = 4.14651696567399e-05"
## [1] "k=5 p-value = 0.0189154942592372"
```

(f) Using a 1% significance level, make a decision regarding the number of nonzero correlation coefficients of the population model based on your results in part (e).

Looking at the results. At a 1% significance value, we would only retain the null hypothesis for k=5.

The hypothesis test at k=5 is $H_0^5 : v_1 \neq 0, \ldots, v_5 \neq 0, v_6 = 0$ vs $H_1^5 : v_1 \neq 0, \ldots, v_6 \neq 0$. Since we fail to reject the null hypothesis, we assume that the 6th correlation coefficient is zero.

Therefore we conclude that 1,. . . ,5 are nonzero correlation coefficients (thus there are 5).

(g) Does the decision change if you replace the 1% significance level by a 5% significance level? If yes, how? Comment.

Yes, it does change. No decisions would be rejected and therefore we would assume that there are 6 nonzero correlation coefficients.