BACS HW (Week 13)

108020024

due on 05/14 (Sun) Helped by 108020033

Question 1) Let's revisit the issue of multicollinearity of main effects (between cylinders, displacement, horsepower, and weight) we saw in the cars dataset, and try to apply principal components to it. Start by recreating the cars_log dataset, which log-transforms all variables except model year and origin. Important: remove any rows that have missing values.

```
cars <- read.table("auto-data.txt", header=FALSE, na.strings = "?")
names(cars) <- c("mpg", "cylinders", "displacement", "horsepower", "weight",
                 "acceleration", "model_year", "origin", "car_name")

cars_log <- with(cars, data.frame(log(mpg), log(cylinders), log(displacement),
log(horsepower),log(weight),log(acceleration),model_year, origin))

cars_log <- na.omit(cars_log)
```

a) Let's analyze the principal components of the four collinear variables

```
i) Create a new data.frame of the four log-transformed variables with high multicollinearity
```

```
mc <- subset(cars_log, select=c("log.weight." , "log.cylinders.", "log.displacement." , "log.horsepower
```

```
ii) How much variance of the four variables is explained by their first principal component?
```

```
mc_eigen <- eigen(cor(mc))
mc_eigen$values[1]/sum(mc_eigen$values)
```

```
## [1] 0.9185647
```

variance of the four variables is explained by their first principal component is 0.9185647.

```
iii)Looking at the values and valence (positiveness/negativeness) of the first principal component's
what would you call the information captured by this component?
```

```
mc_eigen$vectors
```

```
##             [,1]        [,2]        [,3]       [,4]
## [1,] -0.5037960  0.01530917  0.77500928  0.3812031
## [2,] -0.4979145 -0.53580374 -0.52633608  0.4335503
## [3,] -0.5122968 -0.25665246  0.07354139 -0.8162556
## [4,] -0.4856159  0.80424467 -0.34193949  0.0210980
```

We see pc1 equally captures cylinders, displacement, horsepower, and weight, and pc2 captures mostly horsepower, pc3 mostly caputure weight, pc4 captures displacement.

b) Let's revisit our regression analysis on cars_log

```
    i)Store the scores of the first principal component as a new column of cars_log
    Give this new column a name suitable for what it captures (see 1.a.i.)
```

```
cars_log$pc_score <- prcomp(mc)$x
```

```
    ii) Regress mpg over the column with PC1 scores (replacing cylinders, displacement, horsepower, and
```

```
md<-lm( log.mpg.~ pc_score[,"PC1"] +log.acceleration. +model_year+factor(origin), data=cars_log)
summary(md)
```

```
##
## Call:
## lm(formula = log.mpg. ~ pc_score[, "PC1"] + log.acceleration. +
##     model_year + factor(origin), data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.53593 -0.06148  0.00149  0.06293  0.50928
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.395518   0.172873   8.073 8.84e-15 ***
## pc_score[, "PC1"]  0.387073   0.014110  27.433  < 2e-16 ***
## log.acceleration. -0.189830   0.043246  -4.390 1.47e-05 ***
## model_year         0.029244   0.001871  15.628  < 2e-16 ***
## factor(origin)2   -0.010840   0.020738  -0.523    0.601
## factor(origin)3    0.002243   0.020517   0.109    0.913
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1239 on 386 degrees of freedom
## Multiple R-squared:  0.8689, Adjusted R-squared:  0.8672
## F-statistic: 511.7 on 5 and 386 DF,  p-value: < 2.2e-16
```

```
    iii) Try running the regression again over the same independent variables, but this time with every
```

```
temp <- as.data.frame(cbind(scale(cars_log[,c(-8)]),origin = cars_log[,8]))
md2<-lm( log.mpg.~ pc_score.PC1 +log.acceleration. +model_year+factor(origin), data=temp)
summary(md2)
```

```
##
## Call:
## lm(formula = log.mpg. ~ pc_score.PC1 + log.acceleration. + model_year +
##     factor(origin), data = temp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.57609 -0.18081  0.00438  0.18506  1.49772
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.004201   0.026912   0.156    0.876
## pc_score.PC1       0.832369   0.030342  27.433  < 2e-16 ***
## log.acceleration. -0.101021   0.023014  -4.390 1.47e-05 ***
## model_year         0.316814   0.020272  15.628  < 2e-16 ***
## factor(origin)2   -0.031878   0.060987  -0.523    0.601
## factor(origin)3    0.006595   0.060336   0.109    0.913
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3644 on 386 degrees of freedom
## Multiple R-squared:  0.8689, Adjusted R-squared:  0.8672
## F-statistic: 511.7 on 5 and 386 DF,  p-value: < 2.2e-16
```

pc_score.PC1 is very important in this model, since it's estimated beta value is 0.832369, is relatively much higher than other variables, so it has a bigger influence on log.mpg.

Question 2)

```
sq <- read.csv("security_questions.csv")
```

a) How much variance did each extracted factor explain?

```
summary(prcomp(sq, scale. = TRUE))
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.0514 1.26346 1.07217 0.87291 0.82167 0.78209 0.70921
## Proportion of Variance 0.5173 0.08869 0.06386 0.04233 0.03751 0.03398 0.02794
## Cumulative Proportion  0.5173 0.60596 0.66982 0.71216 0.74966 0.78365 0.81159
##                            PC8     PC9   PC10    PC11    PC12    PC13    PC14
## Standard deviation     0.68431 0.67229 0.6206 0.59572 0.54891 0.54063 0.51200
## Proportion of Variance 0.02602 0.02511 0.0214 0.01972 0.01674 0.01624 0.01456
## Cumulative Proportion  0.83760 0.86271 0.8841 0.90383 0.92057 0.93681 0.95137
##                           PC15   PC16   PC17   PC18
## Standard deviation     0.48433 0.4801 0.4569 0.4489
## Proportion of Variance 0.01303 0.0128 0.0116 0.0112
## Cumulative Proportion  0.96440 0.9772 0.9888 1.0000
```

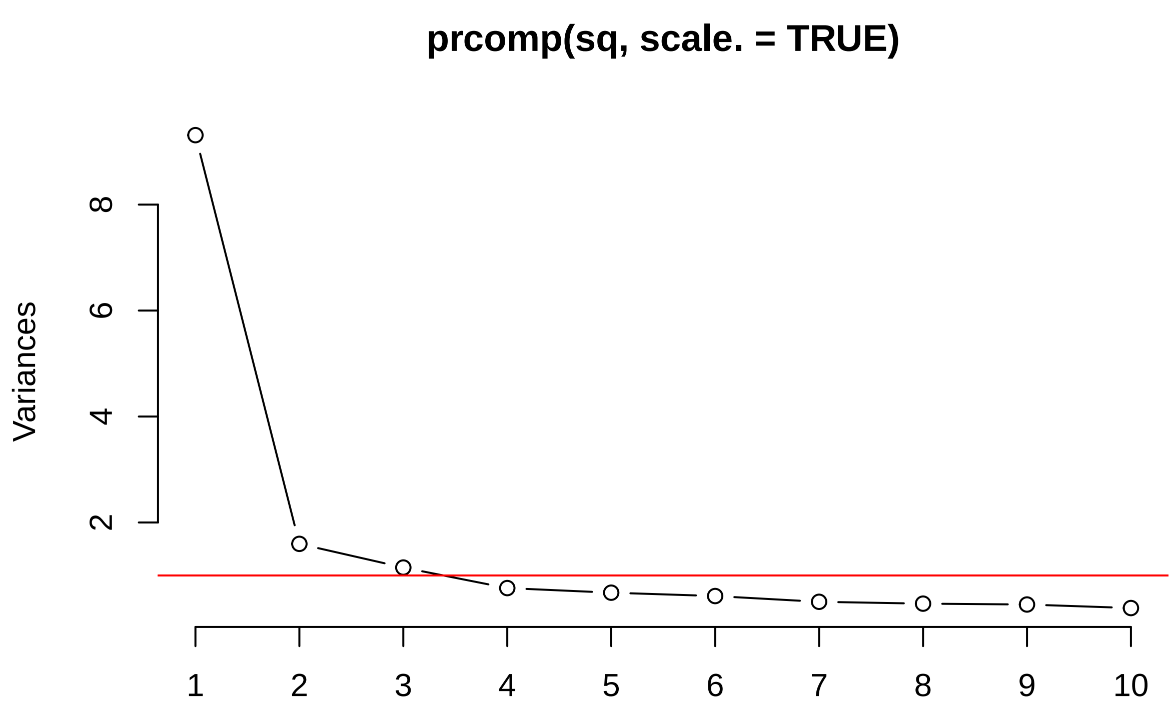
The proportion of Variance each extracted factor explain are (0.5173 0.08869 0.06386 0.04233 0.03751 0.03398 0.02794 0.02602 0.02511 0.0214 0.01972 0.01674 0.01624 0.01456 0.01303 0.0128 0.0116 0.0112)

b) How many dimensions would you retain, according to the two criteria we discussed?

```
eigen(cor(sq))$values
```

```
##  [1] 9.3109533 1.5963320 1.1495582 0.7619759 0.6751412 0.6116636 0.5029855
##  [8] 0.4682788 0.4519711 0.3851964 0.3548816 0.3013071 0.2922773 0.2621437
## [15] 0.2345788 0.2304642 0.2087471 0.2015441
```

```
screeplot(prcomp(sq, scale. = TRUE), type="lines")
abline(h = 1, col = "red")
```



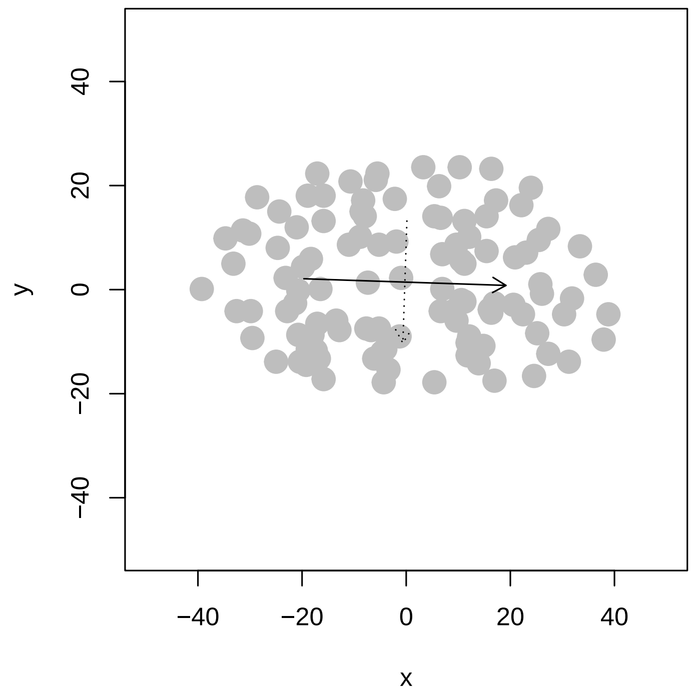
I would retain 3 dimensions.

c) (ungraded) Can you interpret what any of the principal components mean? Try guessing the meaning of the first two or three PCs looking at the PC-vs-variable matrix

Question 3) Let's simulate how principal components behave interactively: run the interactive_pca() function from the compstatslib package we have used earlier:

```
library(compstatslib)
```

a) Create an oval shaped scatter plot of points that stretches in two directions – you should find that the principal component vectors point in the major and minor directions of variance (dispersion). Show this visualization.



b) Can you create a scatterplot whose principal component vectors do NOT seem to match the major directions of variance? Show this visualization.