BACS - HW13

109006241, helped by 109060082

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", na.strings = "?")
names(cars) = c("mpg", "cylinders", "displacement", "horsepower", "weight",
                "acceleration", "model_year", "origin", "car_name")
cars = na.omit(cars)
cars_log = with(cars, data.frame(log(mpg), log(cylinders), log(displacement), log(horsepower),
                                 log(weight), log(acceleration), model_year, origin))
head(cars_log, 5)
```

```
##   log.mpg. log.cylinders. log.displacement. log.horsepower. log.weight.
## 1 2.890372       2.079442          5.726848        4.867534    8.161660
## 2 2.708050       2.079442          5.857933        5.105945    8.214194
## 3 2.890372       2.079442          5.762051        5.010635    8.142063
## 4 2.772589       2.079442          5.717028        5.010635    8.141190
## 5 2.833213       2.079442          5.710427        4.941642    8.145840
##   log.acceleration. model_year origin
## 1          2.484907         70      1
## 2          2.442347         70      1
## 3          2.397895         70      1
## 4          2.484907         70      1
## 5          2.351375         70      1
```

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 (Give this smaller data frame an appropriate name – what might they jointly mean?)

```
var_4 = cars_log[2:5]
head(var_4, 5)
```

```
##   log.cylinders. log.displacement. log.horsepower. log.weight.
## 1       2.079442          5.726848        4.867534    8.161660
## 2       2.079442          5.857933        5.105945    8.214194
## 3       2.079442          5.762051        5.010635    8.142063
## 4       2.079442          5.717028        5.010635    8.141190
## 5       2.079442          5.710427        4.941642    8.145840
```

```
round(cor(var_4), 3)
```

```
##                   log.cylinders. log.displacement. log.horsepower. log.weight.
## log.cylinders.             1.000             0.947           0.827       0.883
## log.displacement.          0.947             1.000           0.872       0.943
## log.horsepower.            0.827             0.872           1.000       0.874
## log.weight.                0.883             0.943           0.874       1.000
```

ii. How much variance of the four variables is explained by their first principal component? (a summary of the prcomp() shows it, but try computing this from the eigenvalues alone)

```
# Using prcomp()
var_4_pca = prcomp(var_4, scale.=T)
summary(var_4_pca)
```

```
## Importance of components:
##                           PC1     PC2     PC3     PC4
## Standard deviation     1.9168 0.43316 0.32238 0.18489
## Proportion of Variance 0.9186 0.04691 0.02598 0.00855
## Cumulative Proportion  0.9186 0.96547 0.99145 1.00000
```

```
# Using eigenvalues
eig = eigen(cor(var_4))$values
proportion = eig[1] / sum(eig)
cat("Proportion of variance explained by the first principal component:", proportion)
```

```
## Proportion of variance explained by the first principal component: 0.9185647
```

iii. Looking at the values and valence (positiveness/negativeness) of the first principal component's eigenvector, what would you call the information captured by this component? (i.e., think what concept the first principal component captures or represents)

```
eigenvector = eigen(cor(var_4))$vectors[,1]
names(eigenvector) = colnames(var_4)
eigenvector
```

```
##    log.cylinders. log.displacement.   log.horsepower.       log.weight.
##        -0.4979145        -0.5122968        -0.4856159        -0.5037960
```

The information captured by the first principal component represents the direction of the maximum variance in the data. It captures the most important linear combination of the variables that explains the greatest amount of variability in the data.

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

```
cars_log$PC1_scores = var_4_pca$x[,"PC1"]
```

ii. Regress mpg over the column with PC1 scores (replacing cylinders, displacement, horsepower, and weight), as well as acceleration, model_year and origin

```
summary(lm(log.mpg. ~ PC1_scores + log.acceleration. + model_year + factor(origin), cars_log))
```

```
##
## Call:
## lm(formula = log.mpg. ~ PC1_scores + log.acceleration. + model_year +
##     factor(origin), data = cars_log)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51137 -0.06050 -0.00183  0.06322  0.46792
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.398114   0.166554   8.394 8.99e-16 ***
## PC1_scores         0.145663   0.005057  28.804  < 2e-16 ***
## log.acceleration. -0.191482   0.041722  -4.589 6.02e-06 ***
## model_year         0.029180   0.001810  16.122  < 2e-16 ***
## factor(origin)2    0.008272   0.019636   0.421    0.674
## factor(origin)3    0.019687   0.019395   1.015    0.311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1199 on 386 degrees of freedom
## Multiple R-squared:  0.8772, Adjusted R-squared:  0.8756
## F-statistic: 551.6 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 everything standardized. How important is this new column relative to other columns?

```
summary(lm(log.mpg. ~ PC1_scores + log.acceleration. + model_year + factor(origin), data.frame(scale(ca
```

```
##
## Call:
## lm(formula = log.mpg. ~ PC1_scores + log.acceleration. + model_year +
##     factor(origin), data = data.frame(scale(cars_log)))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.50385 -0.17791 -0.00538  0.18591  1.37608
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -0.01589    0.02563  -0.620    0.536
## PC1_scores                      0.82112    0.02851  28.804  < 2e-16 ***
## log.acceleration.              -0.10190    0.02220  -4.589 6.02e-06 ***
## model_year                      0.31611    0.01961  16.122  < 2e-16 ***
## factor(origin)0.525710525810929  0.02433    0.05775   0.421    0.674
## factor(origin)1.76714743013553   0.05790    0.05704   1.015    0.311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3526 on 386 degrees of freedom
## Multiple R-squared:  0.8772, Adjusted R-squared:  0.8756
## F-statistic: 551.6 on 5 and 386 DF,  p-value: < 2.2e-16
```

The beta coefficient of the PC1 score column went up from around 0.15 to 0.82. **In general**, a larger absolute beta value indicates a stronger association with the dependent variable

---

Question 2) Please download the Excel data file security_questions.xlsx from Canvas. In your analysis, you can either try to read the data sheet from the Excel file directly from R (there might be a package for that!) or you can try to export the data sheet to a CSV file before reading it into R.

A group of researchers is studying how customers who shopped on e-commerce websites over the winter holiday season perceived the security of their most recently used e-commerce site. Based on feedback from experts, the company has created eighteen questions (see 'questions' tab of excel file) regarding security considerations at e-commerce websites. Over 400 customers responded to these questions (see 'data' tab of Excel file). The researchers now wants to use the results of these eighteen questions to reveal if there are some underlying dimensions of people's perception of online security that effectively capture the variance of these eighteen questions. Let's analyze the principal components of the eighteen items.

```r
data2 = read_excel("security_questions.xlsx", sheet="data")
head(data2, 5)
```

```
## # A tibble: 5 x 18
##      Q1    Q2    Q3    Q4    Q5    Q6    Q7    Q8    Q9   Q10   Q11   Q12   Q13
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     7     5     5     7     7     4     4     7     5     7     5     7     5
## 2     5     5     6     6     6     5     5     7     5     6     6     6     6
## 3     6     6     6     6     7     6     6     6     5     7     6     6     5
## 4     5     5     5     5     5     5     5     5     5     5     5     5     4
## 5     7     7     7     7     7     4     5     7     6     7     6     7     6
## # ... with 5 more variables: Q14 <dbl>, Q15 <dbl>, Q16 <dbl>, Q17 <dbl>,
## #   Q18 <dbl>
```

a. How much variance did each extracted factor explain?

```r
data2_pca = prcomp(data2, scale.=F)
summary(data2_pca)
```

```
## Importance of components:
##                           PC1     PC2    PC3     PC4     PC5    PC6     PC7
## Standard deviation     4.5803 2.01574 1.6194 1.30124 1.25295 1.2341 1.07068
## Proportion of Variance 0.5097 0.09871 0.0637 0.04113 0.03814 0.0370 0.02785
## Cumulative Proportion  0.5097 0.60836 0.6721 0.71319 0.75133 0.7883 0.81618
##                            PC8    PC9    PC10    PC11    PC12   PC13    PC14
## Standard deviation     1.03349 0.9940 0.93530 0.88795 0.81779 0.8166 0.76556
## Proportion of Variance 0.02595 0.0240 0.02125 0.01915 0.01625 0.0162 0.01424
## Cumulative Proportion  0.84213 0.8661 0.88738 0.90653 0.92278 0.9390 0.95322
##                           PC15    PC16    PC17    PC18
## Standard deviation     0.74400 0.72833 0.65653 0.64084
## Proportion of Variance 0.01345 0.01289 0.01047 0.00998
## Cumulative Proportion  0.96667 0.97955 0.99002 1.00000
```

b. How many dimensions would you retain, according to the two criteria we discussed? (Eigenvalue >= 1 and Scree Plot – can you show the screeplot with eigenvalue=1 threshhold?)

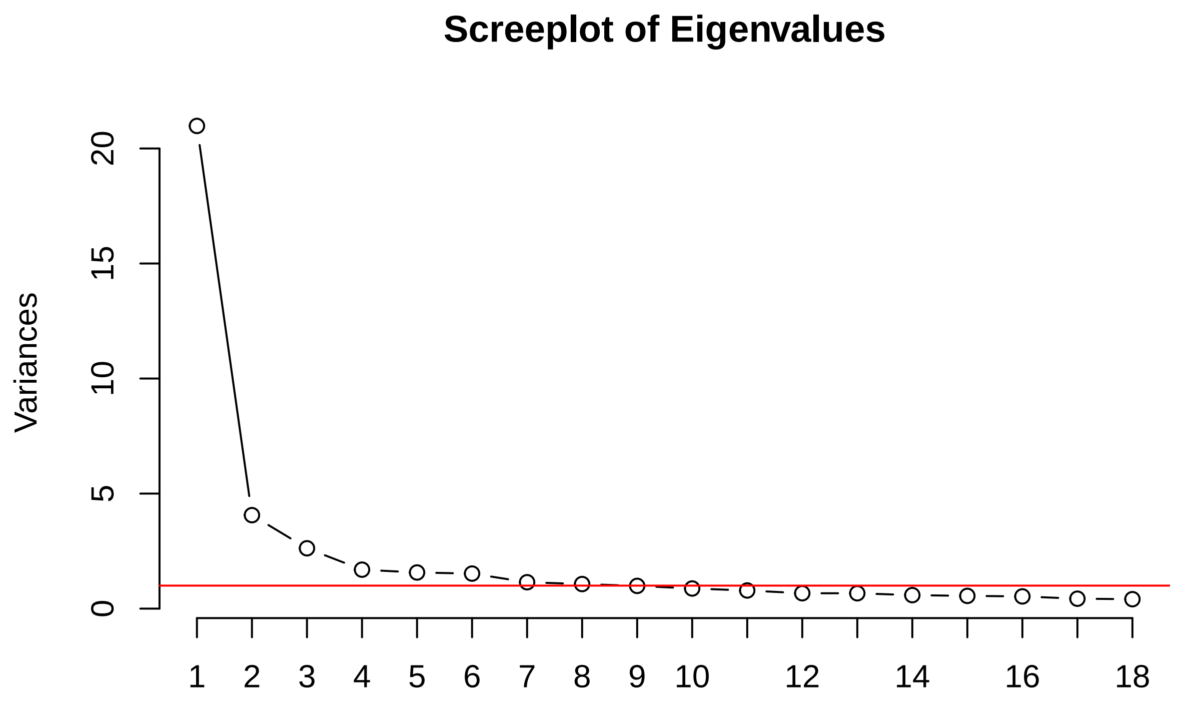
```
eigen(cov(data2))$values
```

```
##  [1] 20.9792250  4.0632189  2.6222946  1.6932329  1.5698748  1.5230757
##  [7]  1.1463561  1.0681022  0.9880189  0.8747800  0.7884507  0.6687853
## [13]  0.6668308  0.5860845  0.5535335  0.5304599  0.4310260  0.4106766
```

```
count = sum(eigen(cov(data2))$values >= 1)
cat("There are", count, "dimensions that we retain.")
```

```
## There are 8 dimensions that we retain.
```

```
screeplot(data2_pca, type="line", main="Screeplot of Eigenvalues", npcs=18)
abline(h=1, col="red")
```

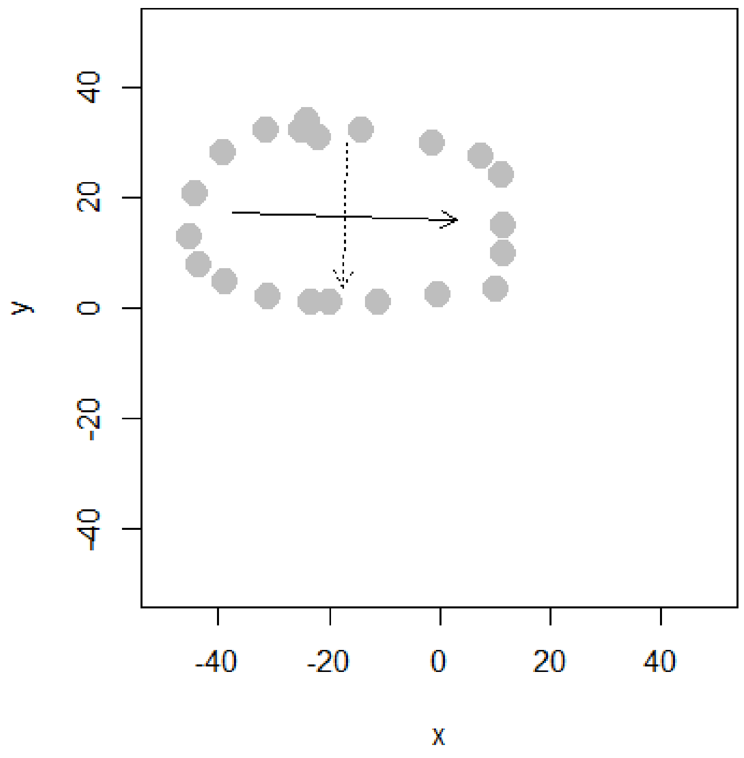


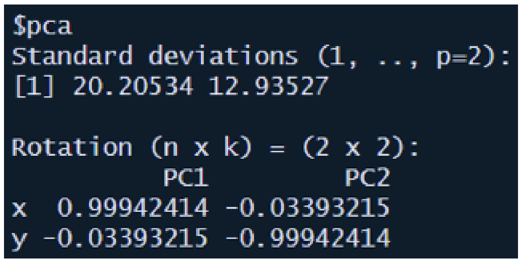
c. (ungraded) Can you interpret what any of the principal components mean? Try guessing the meaning of the first two or three PCs looking at the PC-vs-variable matrix

   The principal components are linear combinations of the original variables, which explains the amount of variance in the data. PC1 explains the greatest amount of variance in the data, followed by PC2, PC3, etc, subject to the constraint that it is orthogonal to the previous principal components.

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

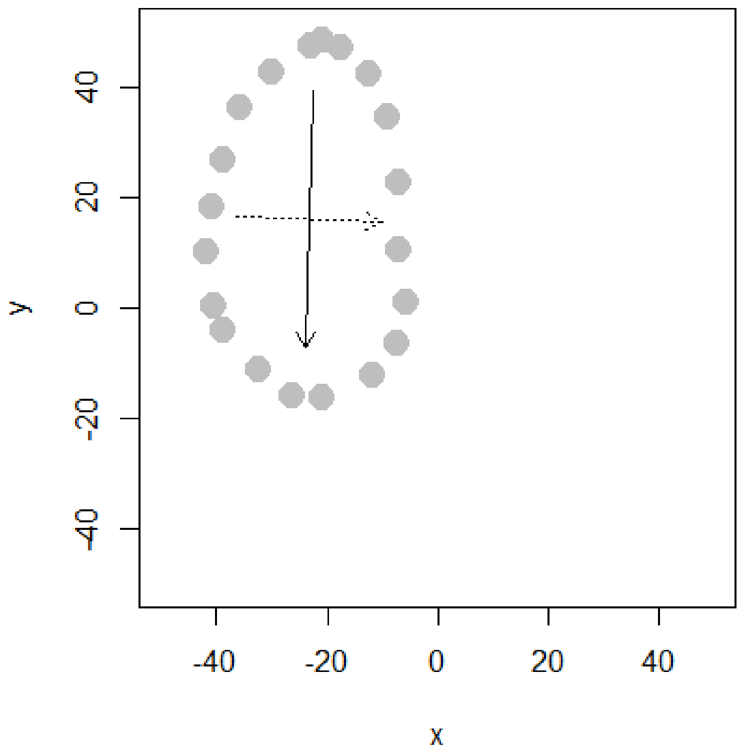
a. Create an oval shaped scatter plot of points that stretches in two directions – you should find that the principal component vectors point in the major and minor directions of variance (dispersion). Show this visualization.

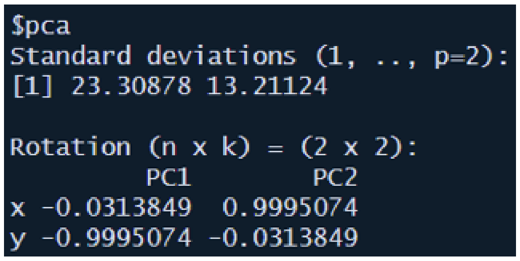




```
$pca
Standard deviations (1, .., p=2):
[1] 20.20534 12.93527

Rotation (n x k) = (2 x 2):
          PC1         PC2
x  0.99942414 -0.03393215
y -0.03393215 -0.99942414
```

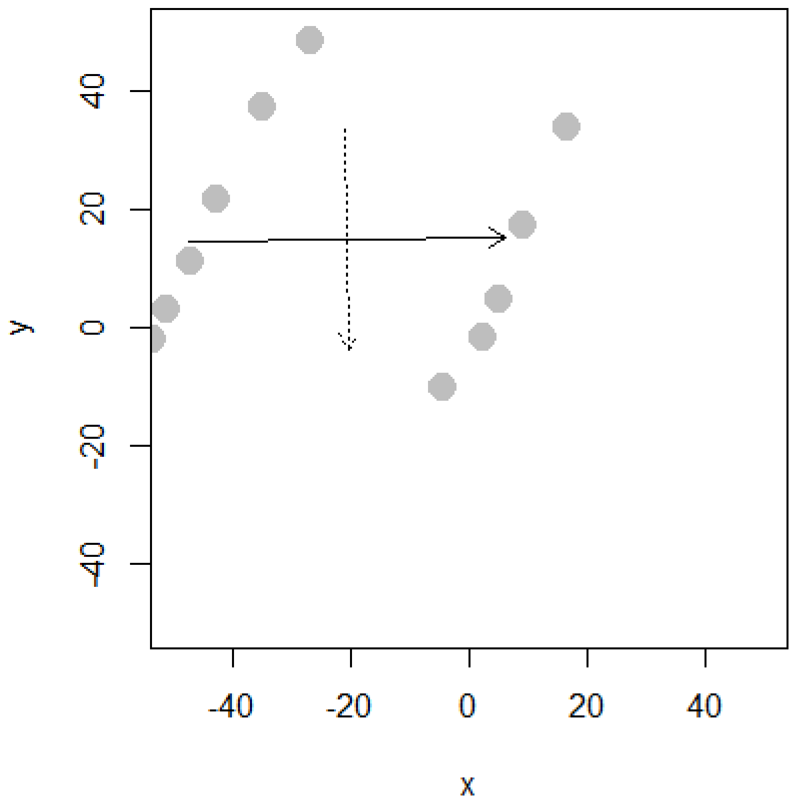


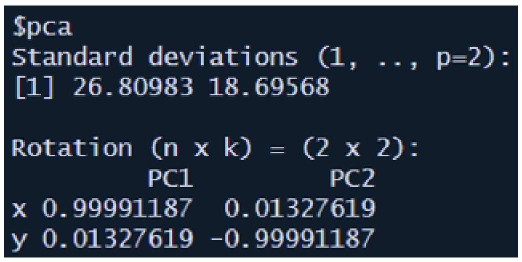


```
$pca
Standard deviations (1, .., p=2):
[1] 23.30878 13.21124

Rotation (n x k) = (2 x 2):
         PC1        PC2
x -0.0313849  0.9995074
y -0.9995074 -0.0313849
```

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





```
$pca
Standard deviations (1, .., p=2):
[1] 26.80983 18.69568

Rotation (n x k) = (2 x 2):
         PC1         PC2
x 0.99991187  0.01327619
y 0.01327619 -0.99991187
```