HW13

108048110

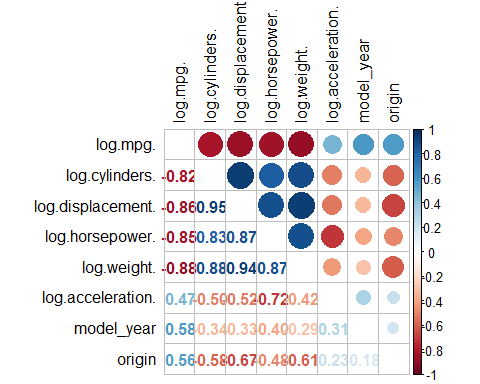
5/10/2022

# BACS HW - Week 13

## Prerequisite

library(corrplot)  
library(ggplot2)  
library(ggbiplot)  
library(grid)  
library(factoextra)  
library(tidyverse)  
library(magrittr)  
library(FactoMineR)

cor\_plt <- function(data){  
 cor\_data <- round(cor(data[, 1:length(data)], use='pairwise.complete.obs'), 3)  
 corrplot.mixed(cor\_data, tl.col='black', tl.pos='lt')  
}  
  
auto = read.table('data/auto-data.txt', header=FALSE, na.strings = '?')  
names(auto) <- c("mpg", "cylinders", "displacement", "horsepower", "weight",   
 "acceleration", "model\_year", "origin", "car\_name")  
auto = as.data.frame(auto[complete.cases(auto),])  
  
car\_log = with(auto, data.frame(log(mpg),  
 log(cylinders),  
 log(displacement),  
 log(horsepower),  
 log(weight),  
 log(acceleration),  
 model\_year,  
 origin))  
car\_log = as.data.frame(car\_log[complete.cases(car\_log),])  
cor\_plt(car\_log)



## PCA

* **Note.** PCA, principal component analysis is a dimensionality reduction method often used to reduce the dimensionalty of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.
* Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimentionality reduction is to trade a little accuracy for simplicity!
* Smaller data sets are easier to explore, compute and visualize that makes analyzing data much faster and easier without extraneous variables to process.
* To conclude in short, the idea of PCA is to reduce the number of variables of a data set, while preserving as much information as possible.

## Question 1) Principal Component Analysis

### a. Analyze the principal components of the four collinear variables.

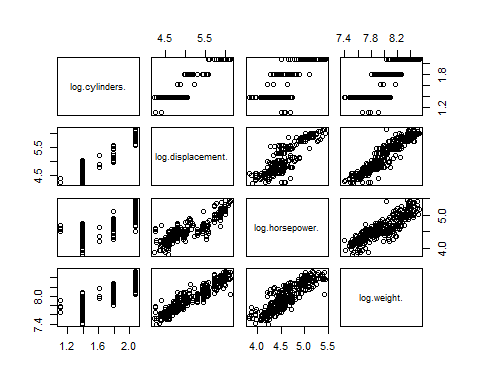
**(cylinders, displacement, horsepower, and weight)**

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

high\_corr\_variables = with(car\_log, data.frame(log.cylinders.,  
 log.displacement.,  
 log.horsepower.,  
 log.weight.)  
)  
knitr::kable(head(high\_corr\_variables))

|  |  |  |  |
| --- | --- | --- | --- |
| log.cylinders. | log.displacement. | log.horsepower. | log.weight. |
| 2.079442 | 5.726848 | 4.867534 | 8.161660 |
| 2.079442 | 5.857933 | 5.105945 | 8.214194 |
| 2.079442 | 5.762051 | 5.010635 | 8.142063 |
| 2.079442 | 5.717028 | 5.010635 | 8.141190 |
| 2.079442 | 5.710427 | 4.941642 | 8.145840 |
| 2.079442 | 6.061457 | 5.288267 | 8.375860 |

plot(high\_corr\_variables)



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

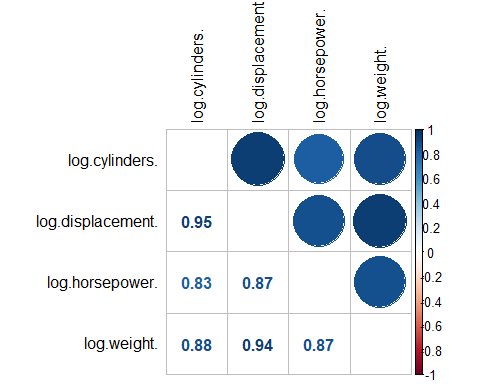
# Principal component of this "high\_corr\_variables" are the eigenvectors of its covariance matrix  
head(cov(high\_corr\_variables))

## log.cylinders. log.displacement. log.horsepower. log.weight.  
## log.cylinders. 0.09135350 0.1524578 0.08578724 0.07508073  
## log.displacement. 0.15245781 0.2837631 0.15953003 0.14123145  
## log.horsepower. 0.08578724 0.1595300 0.11790905 0.08438670  
## log.weight. 0.07508073 0.1412315 0.08438670 0.07907185

head(cor(high\_corr\_variables))

## log.cylinders. log.displacement. log.horsepower. log.weight.  
## log.cylinders. 1.0000000 0.9469109 0.8265831 0.8833950  
## log.displacement. 0.9469109 1.0000000 0.8721494 0.9428497  
## log.horsepower. 0.8265831 0.8721494 1.0000000 0.8739558  
## log.weight. 0.8833950 0.9428497 0.8739558 1.0000000

cor\_plt(high\_corr\_variables)



* **Concept:** Recall that covariance matrix calculates the similarities between the variables using dot product, while correlation matrix uses the standardized dot product, in other words, correlation matrix can be interpreted as a standardized version of covariance matrix.

eigen\_vectors =eigen(cov(high\_corr\_variables))$vectors # eigen vectors of covariance of high\_corr\_variables  
colnames(eigen\_vectors) = c('PC1', 'PC2', 'PC3', 'PC4')  
row.names(eigen\_vectors) = names(high\_corr\_variables)  
knitr::kable(head(eigen\_vectors))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | PC1 | PC2 | PC3 | PC4 |
| log.cylinders. | -0.3944484 | 0.3261534 | 0.6895416 | 0.5124126 |
| log.displacement. | -0.7221160 | 0.3613485 | -0.1626248 | -0.5670353 |
| log.horsepower. | -0.4322835 | -0.8728969 | 0.2158783 | -0.0676648 |
| log.weight. | -0.3689037 | -0.0331992 | -0.6719242 | 0.6413469 |

eigen\_values = eigen(cov(high\_corr\_variables))$values # eigen values of covariance of high\_corr\_variables  
eigen\_values

## [1] 0.534692011 0.023024805 0.009092508 0.005288216

# confirm with principle components analysis  
high\_corr\_var\_pca = prcomp(high\_corr\_variables)  
summary(high\_corr\_var\_pca)

## Importance of components:  
## PC1 PC2 PC3 PC4  
## Standard deviation 0.7312 0.15174 0.09535 0.07272  
## Proportion of Variance 0.9346 0.04025 0.01589 0.00924  
## Cumulative Proportion 0.9346 0.97486 0.99076 1.00000

* ***iii.*** What would you call the information captured by the first principal component?

high\_corr\_var\_pca$center

## log.cylinders. log.displacement. log.horsepower. log.weight.   
## 1.653046 5.127891 4.587931 7.959180

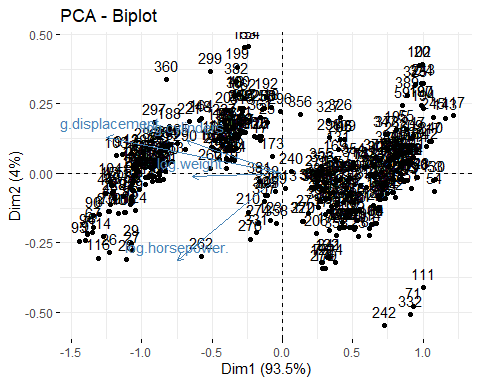
# square roots of the eigenvalues of the covariace matrix  
high\_corr\_var\_pca$sdev

## [1] 0.73122637 0.15173927 0.09535464 0.07272012

# verify with the eigenvalues we calculated  
sqrt(eigen\_values)

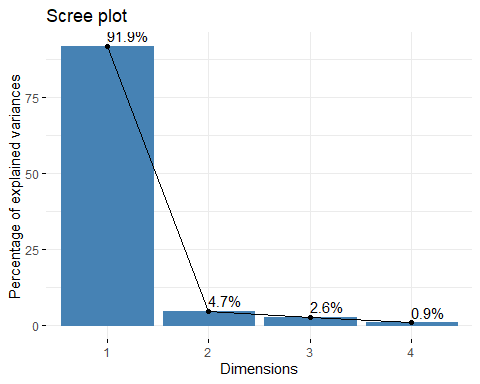
## [1] 0.73122637 0.15173927 0.09535464 0.07272012

# x returns the centered data multiply by the rotation matrix  
Scores = high\_corr\_var\_pca$x  
fviz\_pca\_biplot(high\_corr\_var\_pca)

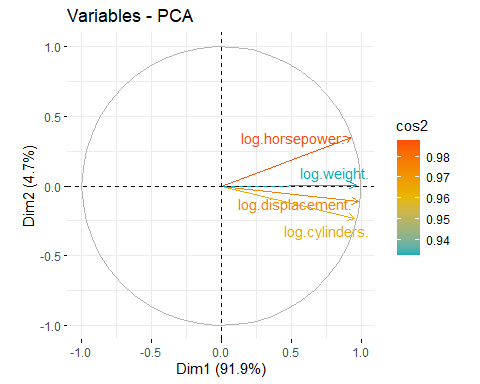


* **Note.** The idea of principal component analysis is that it tries to put maximum possible information in the first components, then the maximum remaining information in the second and so on, until having something like shown in the scree plot below.

fit <- high\_corr\_variables %>% scale()  
res.pca = PCA(fit, graph=FALSE)  
fviz\_eig(res.pca, addlabels=TRUE)



fviz\_pca\_var(res.pca, col.var = "cos2",  
 gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),   
 repel = TRUE)



* + **Note.** Organizing information in PC this way will allow you to reduce dimensionality without losing much information by discarding the components with low information and considering the remaining components as your new variables.
  + Geometrically speaking, PC represents the directions of the data that explains a maximal amount of variance, in other words, the lines that capture most information of the data.

### b. Revisit our regression analysis on car\_log.

* ***i.*** Store the scores of the first principal component as a new column of cars\_log.

car\_log$scores <- Scores[,'PC1']

* ***ii.*** Regress mpgover the column with PC1 scores as well as acceleration, model\_year and origin.

summary(  
 lm(log.mpg.~  
 log.acceleration.+  
 model\_year+  
 factor(origin)+  
 scores,  
 data=car\_log)  
)

##   
## Call:  
## lm(formula = log.mpg. ~ log.acceleration. + model\_year + factor(origin) +   
## scores, data = car\_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 \*\*\*  
## 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   
## scores 0.387073 0.014110 27.433 < 2e-16 \*\*\*  
## ---  
## 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.*** Run the regression over the same independent variables with everything standardized.
* How important is this new column relative to other columns?

sapply(high\_corr\_variables, function(x) {max(x)-min(x)})

## log.cylinders. log.displacement. log.horsepower. log.weight.   
## 0.9808293 1.9007897 1.6094379 1.1589573

* **Note.** These four scales have different ranges. Since PCA is quite sensitive regarding the variances of the initial variables. Variables with larger ranges will dominate over those with small ranges, which will lead to biased results.

high\_corr\_var\_pca = prcomp(high\_corr\_variables, scale. = TRUE)  
Scores = high\_corr\_var\_pca$x  
car\_log$scores <- Scores[,'PC1']  
  
summary(  
 lm(scale(log.mpg.)~  
 scale(log.acceleration.)+  
 scale(model\_year)+  
 factor(origin)+  
 scores,  
 data=car\_log  
 )  
)

##   
## Call:  
## lm(formula = scale(log.mpg.) ~ scale(log.acceleration.) + scale(model\_year) +   
## factor(origin) + scores, data = car\_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   
## scale(log.acceleration.) -0.10190 0.02220 -4.589 6.02e-06 \*\*\*  
## scale(model\_year) 0.31611 0.01961 16.122 < 2e-16 \*\*\*  
## factor(origin)2 0.02433 0.05775 0.421 0.674   
## factor(origin)3 0.05790 0.05704 1.015 0.311   
## scores 0.42837 0.01487 28.804 < 2e-16 \*\*\*  
## ---  
## 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

* **Ans.** Variables are now transformed into same scale, and column scores is very significant relative to the other columns.

## Question 2) Analyze the principal components of the eighteen items from the excel data file security\_questions.xlsx.

### a. How much variance did each extracted factor explain?

questions <- readxl::read\_excel('data/security\_questions.xlsx',  
 sheet=1,  
 col\_names = c('Question', 'Description'))  
responds <- readxl::read\_excel('data/security\_questions.xlsx',  
 sheet=2,  
 col\_names = TRUE)  
  
cov(responds)[1:5, 1:5]

## Q1 Q2 Q3 Q4 Q5  
## Q1 1.9871043 1.368567 1.180663 0.9153832 1.0734629  
## Q2 1.3685674 2.735179 1.185338 1.0127429 1.0465102  
## Q3 1.1806625 1.185338 2.148600 1.0839567 1.0165811  
## Q4 0.9153832 1.012743 1.083957 2.5370737 0.8373304  
## Q5 1.0734629 1.046510 1.016581 0.8373304 1.9944750

sapply(responds, function(x){max(x)-min(x)})

## Q1 Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q9 Q10 Q11 Q12 Q13 Q14 Q15 Q16 Q17 Q18   
## 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

# they are in the same scale -> no need of scaling  
respond\_pca <- prcomp(responds)  
summary(respond\_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?

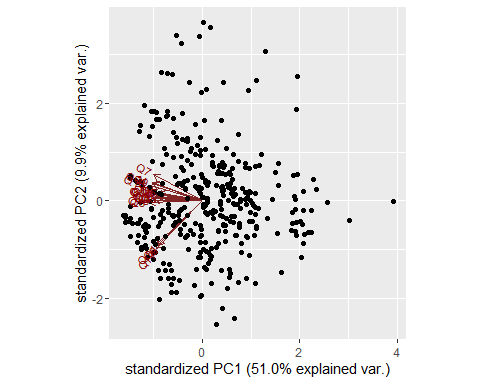
Criteria

|  |  |
| --- | --- |
| **Criteria I** | Eigenvalue ≥ 1 |
| **Criteria II** | **Factors before the** |

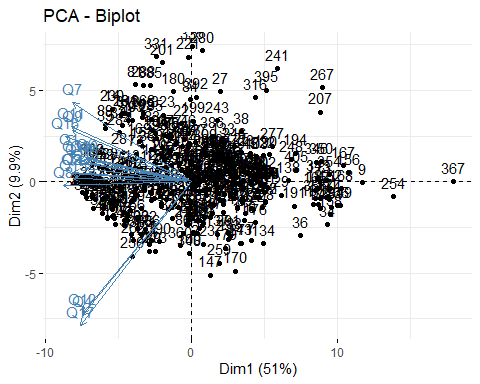
respond\_eigen <- eigen(cor(responds))  
knitr::kable(head(respond\_eigen$values))

|  |
| --- |
| x |
| 9.3109533 |
| 1.5963320 |
| 1.1495582 |
| 0.7619759 |
| 0.6751412 |
| 0.6116636 |

ggbiplot(respond\_pca, labels=rownames(respond\_pca))

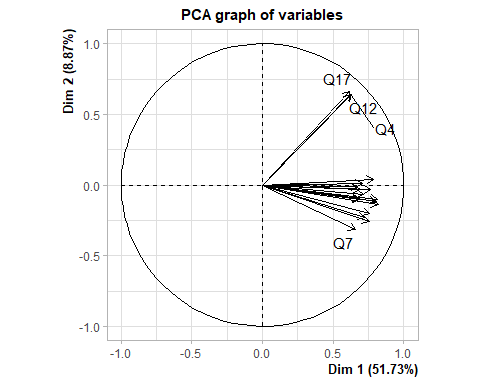
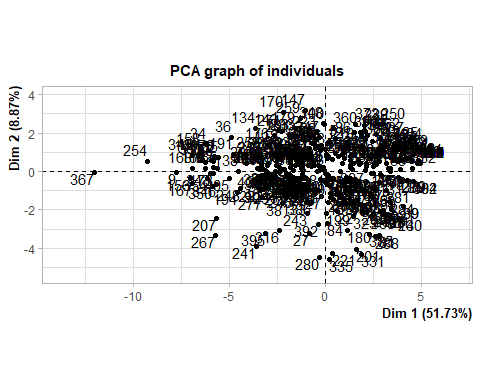


fviz\_pca\_biplot(respond\_pca)

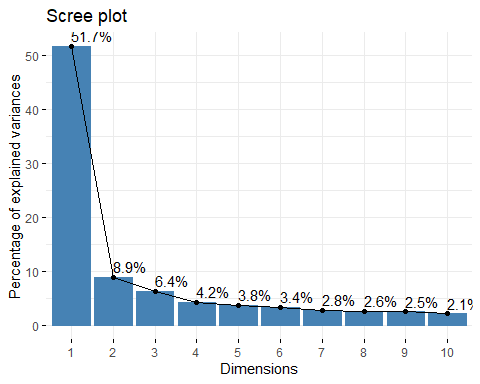


**Ans.** According to the scree plot, the third PC does not lie before the *elbow*, despite the fact that it has an eigenvalue bigger than 1. Hence, I will choose only the first 2 PC’s to retain in the model.

res.pca <- PCA(responds)



fviz\_eig(res.pca, addlabels=TRUE)



## c. Can you interpret what any of the principal components mean?

## Guess the meaning of the first two or three PCs.

respond\_eigen$values %>% subset(respond\_eigen$values>=1)

## [1] 9.310953 1.596332 1.149558

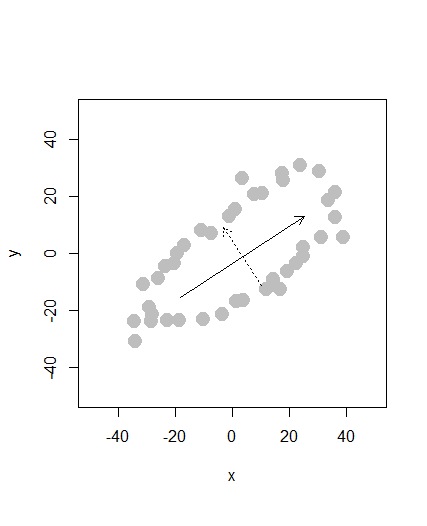
summary(respond\_pca)

## 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

* **Ans.** The first PC seems to give weights equally to every factor, while the second PC gives a larger weight to Q17 Q12 and Q4.
* So, PC1 and PC2 not only capture more variance than the original data on average, they also offer significantly more variance than the remaining PCs.

## Question 3) Simulate how principal components behave interactively.

### a. Create an oval shaped scatter plot of points that stretches in two directions. Show this visualization.



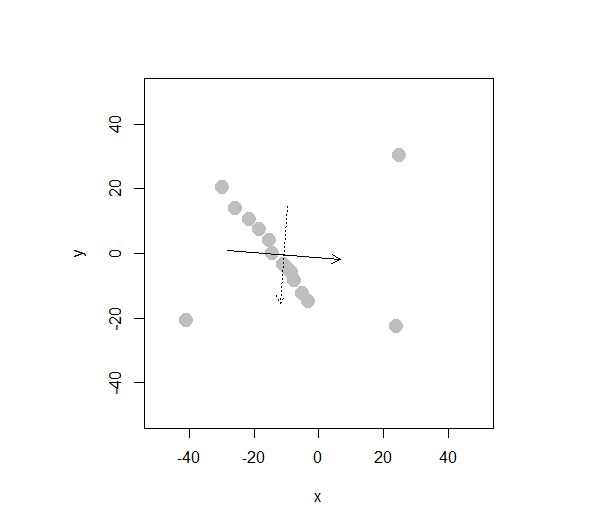
Standard Deviations (1, .., p=2):

|  |  |
| --- | --- |
| 36.96495 | 18.43857 |

Rotation (n x k) = (2 x 2):

|  |  |  |
| --- | --- | --- |
| - |  |  |
|  | 0.8330720 | -0.5474046 |
|  | 0.5474046 | 0.8330720 |

### b. Create a scatterplot whose principal component vectors do NOT seem to match the major directions of variance. Show this visualization.



Standard Deviations (1, .., p=2):

|  |  |
| --- | --- |
| 17.51977 | 14.99578 |

Rotation (n x k) = (2 x 2):

|  |  |  |
| --- | --- | --- |
| - |  |  |
|  | 0.9969089 | -0.0785661 |
|  | -0.0785661 | -0.9969089 |