

Homework - 2

Group 4

5/19/2020

Problem 1

Perform principal component analysis on NHL.xlsx, which contains statistics of 30 teams in the National Hockey League. The description of the variables is provided in the 'Description' sheet of the file. Focus only on the variables 12 through 25, and create a new data frame.

- Input the new data frame to `fa.parallel()` function to determine the number of components to extract
- Input the new data frame to `principal()` function to extract the components. If raw data is input, the correlation matrix is automatically calculated by `principal()` function.
- Rotate the components
- Compute component scores
- Graph an orthogonal solution using `factor.plot()`
- Interpret the results

First, import all the required libraries

```
library(dplyr)
library(readxl)
library(psych)
```

```
# Import the NHL excel file as a dataframe
```

```
NHL <- data.frame(read_xlsx("./data/NHL.xlsx", sheet = "Data"))
```

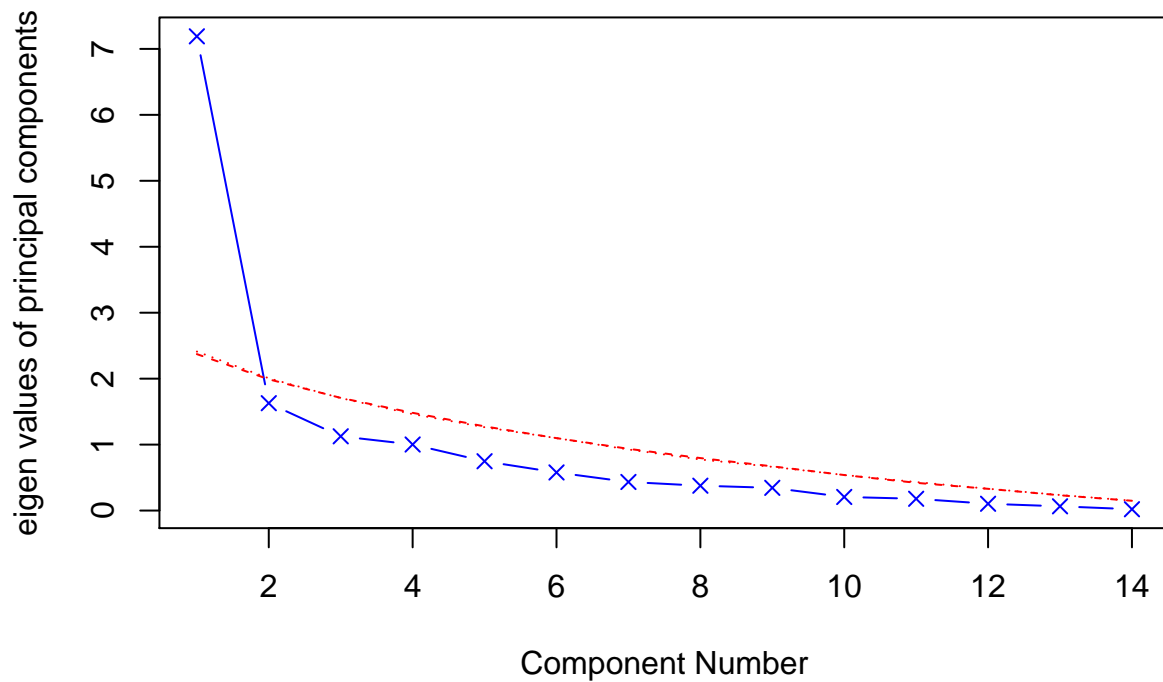
```
# Select the columns 13-26, the 1st column is the index column from excel
```

```
df <- NHL[, 13:26]
```

```
# Use Parallel Analysis Scree Plots to figure out the number of factors to extract
```

```
fa.parallel(df, fa = "pc", n.iter = 100, show.legend = FALSE)
```

Parallel Analysis Scree Plots



Parallel analysis suggests that the number of factors = NA and the number of components = 1

From the above plot it will be appropriate to use 1 factor

Perform PCA with varimax orthogonal rotation

```
pc <- principal(df, nfactors = 1, rotate = "none", scores = TRUE)
```

pc

Principal Components Analysis

Call: principal(r = df, nfactors = 1, rotate = "none", scores = TRUE)

Standardized loadings (pattern matrix) based upon correlation matrix

	PC1	h2	u2	com
## gg	0.83	0.682	0.32	1
## gag	-0.82	0.671	0.33	1
## five	0.90	0.818	0.18	1
## PPP	0.16	0.026	0.97	1
## PKP	0.70	0.490	0.51	1
## shots	0.61	0.367	0.63	1
## sag	-0.63	0.400	0.60	1
## sc1	0.82	0.666	0.33	1
## tr1	0.74	0.552	0.45	1
## lead1	0.82	0.667	0.33	1
## lead2	0.75	0.564	0.44	1
## wop	0.71	0.500	0.50	1
## wosp	0.84	0.710	0.29	1
## face	0.28	0.078	0.92	1

```
##
##          PC1
## SS loadings    7.19
## Proportion Var 0.51
##
## Mean item complexity = 1
## Test of the hypothesis that 1 component is sufficient.
##
## The root mean square of the residuals (RMSR) is 0.11
## with the empirical chi square 70.28 with prob < 0.69
##
## Fit based upon off diagonal values = 0.95
```

Now rotate the components

```
# PCA with varimax rotation
```

```
pc <- principal(df, nfactors = 1, rotate = "varimax", scores = TRUE)
```

```
pc
```

```
## Principal Components Analysis
## Call: principal(r = df, nfactors = 1, rotate = "varimax", scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##      PC1    h2    u2 com
## gg      0.83 0.682 0.32  1
## gag     -0.82 0.671 0.33  1
## five    0.90 0.818 0.18  1
## PPP     0.16 0.026 0.97  1
## PKP     0.70 0.490 0.51  1
## shots   0.61 0.367 0.63  1
## sag     -0.63 0.400 0.60  1
## sc1     0.82 0.666 0.33  1
## tr1     0.74 0.552 0.45  1
## lead1   0.82 0.667 0.33  1
## lead2   0.75 0.564 0.44  1
## wop     0.71 0.500 0.50  1
## wosp    0.84 0.710 0.29  1
## face    0.28 0.078 0.92  1
##
##          PC1
## SS loadings    7.19
## Proportion Var 0.51
##
## Mean item complexity = 1
## Test of the hypothesis that 1 component is sufficient.
##
## The root mean square of the residuals (RMSR) is 0.11
## with the empirical chi square 70.28 with prob < 0.69
##
## Fit based upon off diagonal values = 0.95
```

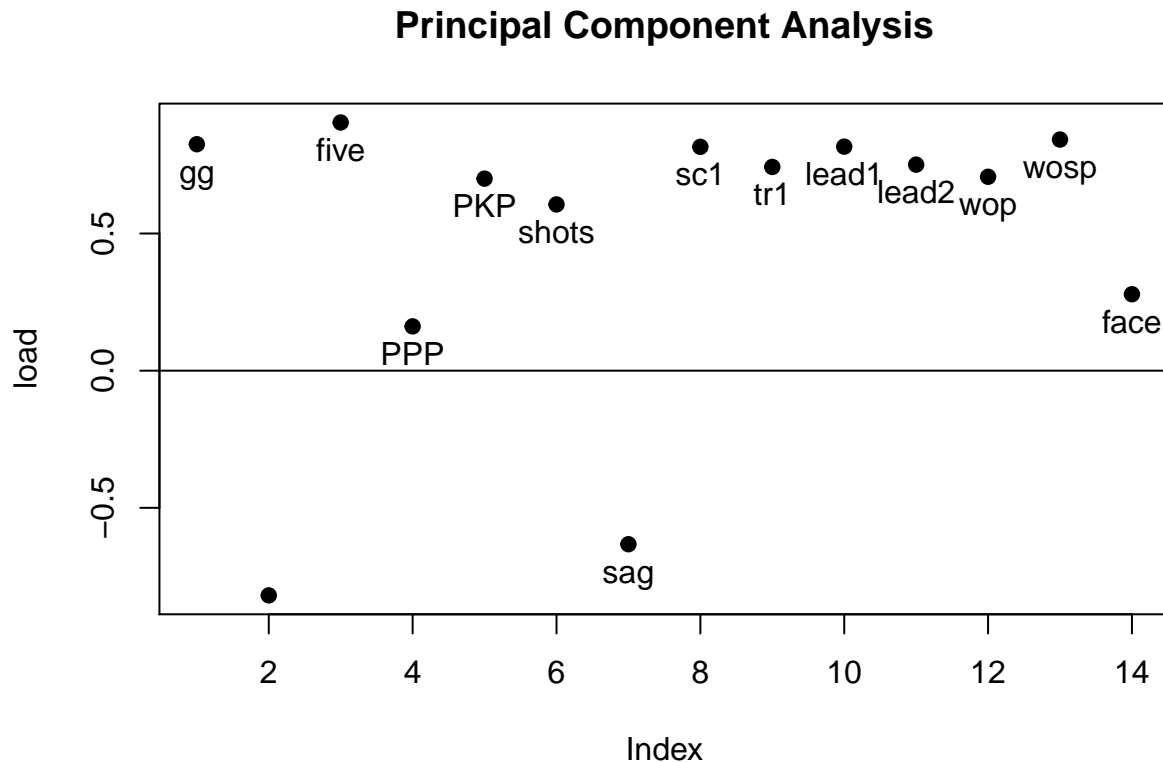
Let's see the component scores

```
head(pc$scores)
```

```
##          PC1
```

```
## [1,] 1.3503592
## [2,] 1.1106603
## [3,] 0.7062801
## [4,] 0.7251884
## [5,] 1.1956402
## [6,] 1.1813631

# Plot the components and analyze them
factor.plot(pc, labels = colnames(df))
```



```
rm(list = ls())
```

Observations made from the plots - - The Principal Component loads all the components except PPP and face.

- There isn't any clear indication as to what the component signifies. Further analysis may be required.

Problem 2

Perform principal component analysis on Glass Identification Data.xlsx

- Input the raw data matrix to `fa.parallel()` function to determine the number of components to extract
- Input the raw data matrix to `principal()` function to extract the components. If raw data is input, the correlation matrix is automatically calculated by `principal()` function.
- Rotate the components

- Compute component scores
- Graph an orthogonal solution using factor.plot()
- Interpret the results

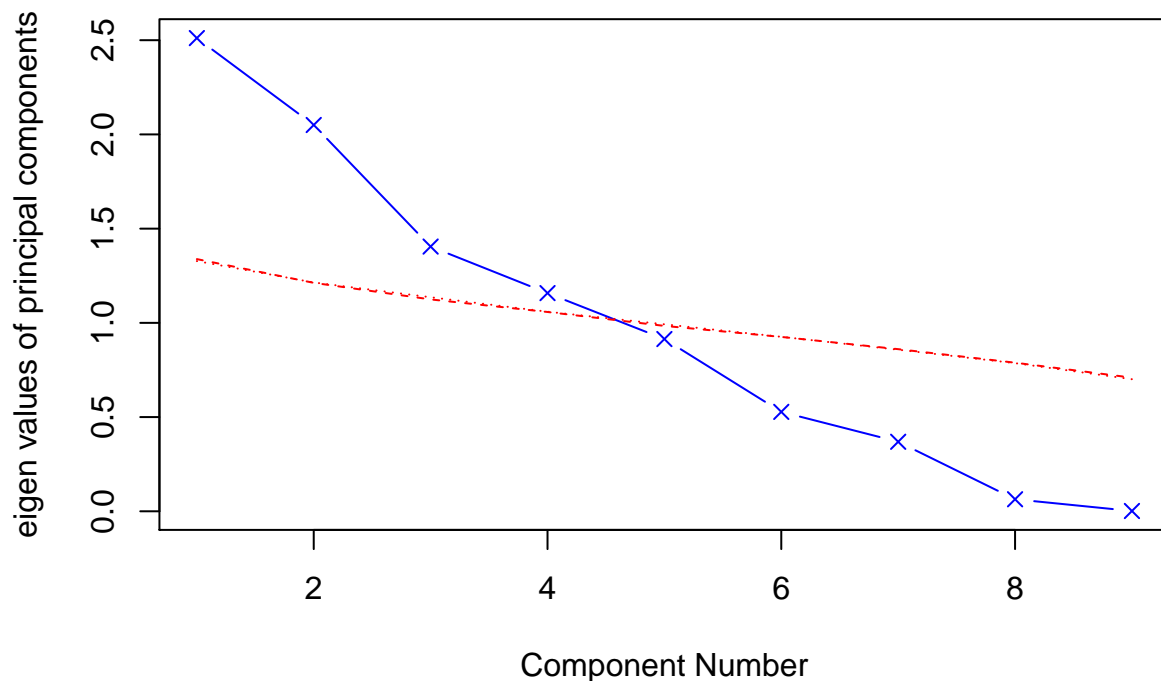
```
glass_identification_data <- data.frame(read_xlsx("./data/Glass Identification Data.xlsx",
  sheet = "Glass Data"
))

fa.parallel(glass_identification_data[, 2:10], fa = "pc", n.iter = 100, show.legend = FALSE)

## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect. Try a
## different factor score estimation method.

## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected. Examine the results carefully
```

Parallel Analysis Scree Plots



Parallel analysis suggests that the number of factors = NA and the number of components = 4
 The Skree Plot and the fa.parallel() function suggests nfactors = 4.

```
# Perform PCA without rotation
pc <- principal(glass_identification_data[, 2:10], nfactors = 4, rotate = "none", scores = TRUE)
pc
```

```
## Principal Components Analysis
## Call: principal(r = glass_identification_data[, 2:10], nfactors = 4,
```

```
## rotate = "none", scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
## PC1 PC2 PC3 PC4 h2 u2 com
## RI -0.86 0.41 0.10 -0.16 0.95 0.051 1.5
## Na 0.41 0.39 -0.46 -0.53 0.80 0.195 3.8
## Mg -0.18 -0.85 0.01 -0.41 0.92 0.081 1.5
## Al 0.68 0.42 0.39 0.15 0.81 0.186 2.5
## Si 0.36 -0.22 -0.54 0.70 0.97 0.031 2.7
## K 0.35 -0.22 0.79 0.04 0.79 0.212 1.6
## CA -0.78 0.49 0.00 0.30 0.94 0.058 2.0
## Ba 0.40 0.69 0.09 -0.14 0.67 0.333 1.7
## Fe -0.29 -0.09 0.34 0.25 0.27 0.730 3.0
##
## PC1 PC2 PC3 PC4
## SS loadings 2.51 2.05 1.40 1.16
## Proportion Var 0.28 0.23 0.16 0.13
## Cumulative Var 0.28 0.51 0.66 0.79
## Proportion Explained 0.35 0.29 0.20 0.16
## Cumulative Proportion 0.35 0.64 0.84 1.00
##
## Mean item complexity = 2.3
## Test of the hypothesis that 4 components are sufficient.
##
## The root mean square of the residuals (RMSR) is 0.08
## with the empirical chi square 102.53 with prob < 7.4e-20
##
## Fit based upon off diagonal values = 0.92
```

Rotate the components

```
# Perform PCA with rotation
pc <- principal(glass_identification_data[, 2:10], nfactors = 4, rotate = "varimax", scores = TRUE)
pc
```

```
## Principal Components Analysis
## Call: principal(r = glass_identification_data[, 2:10], nfactors = 4,
## rotate = "varimax", scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
## RC1 RC2 RC3 RC4 h2 u2 com
## RI 0.84 -0.07 0.15 0.47 0.95 0.051 1.7
## Na -0.06 0.22 -0.86 0.09 0.80 0.195 1.2
## Mg -0.35 -0.86 0.04 0.21 0.92 0.081 1.5
## Al -0.42 0.80 0.03 0.01 0.81 0.186 1.5
## Si -0.13 0.00 -0.02 -0.98 0.97 0.031 1.0
## K -0.62 0.22 0.51 0.30 0.79 0.212 2.7
## CA 0.91 0.12 0.30 0.06 0.94 0.058 1.3
## Ba -0.01 0.72 -0.33 0.17 0.67 0.333 1.5
## Fe 0.12 -0.04 0.50 0.07 0.27 0.730 1.2
##
## RC1 RC2 RC3 RC4
## SS loadings 2.26 2.03 1.48 1.36
## Proportion Var 0.25 0.23 0.16 0.15
## Cumulative Var 0.25 0.48 0.64 0.79
## Proportion Explained 0.32 0.28 0.21 0.19
```

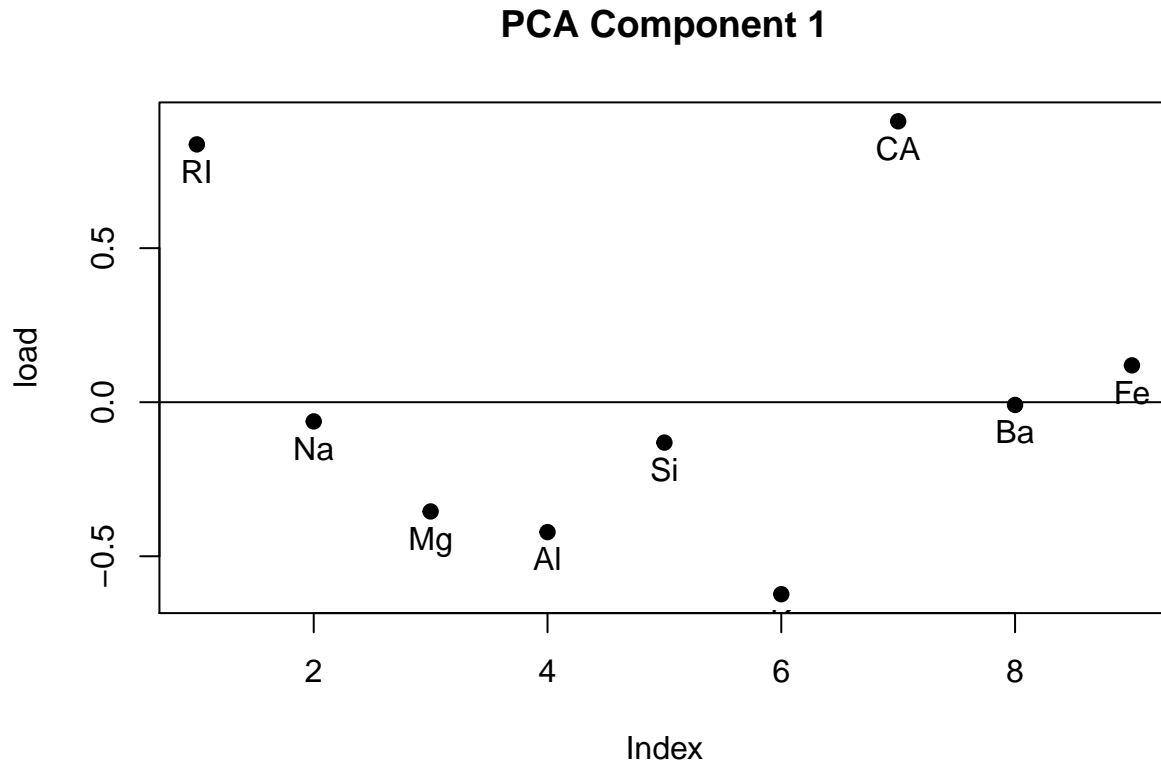
```
## Cumulative Proportion 0.32 0.60 0.81 1.00
##
## Mean item complexity = 1.5
## Test of the hypothesis that 4 components are sufficient.
##
## The root mean square of the residuals (RMSR) is 0.08
## with the empirical chi square 102.53 with prob < 7.4e-20
##
## Fit based upon off diagonal values = 0.92
```

```
head(pc$scores)
```

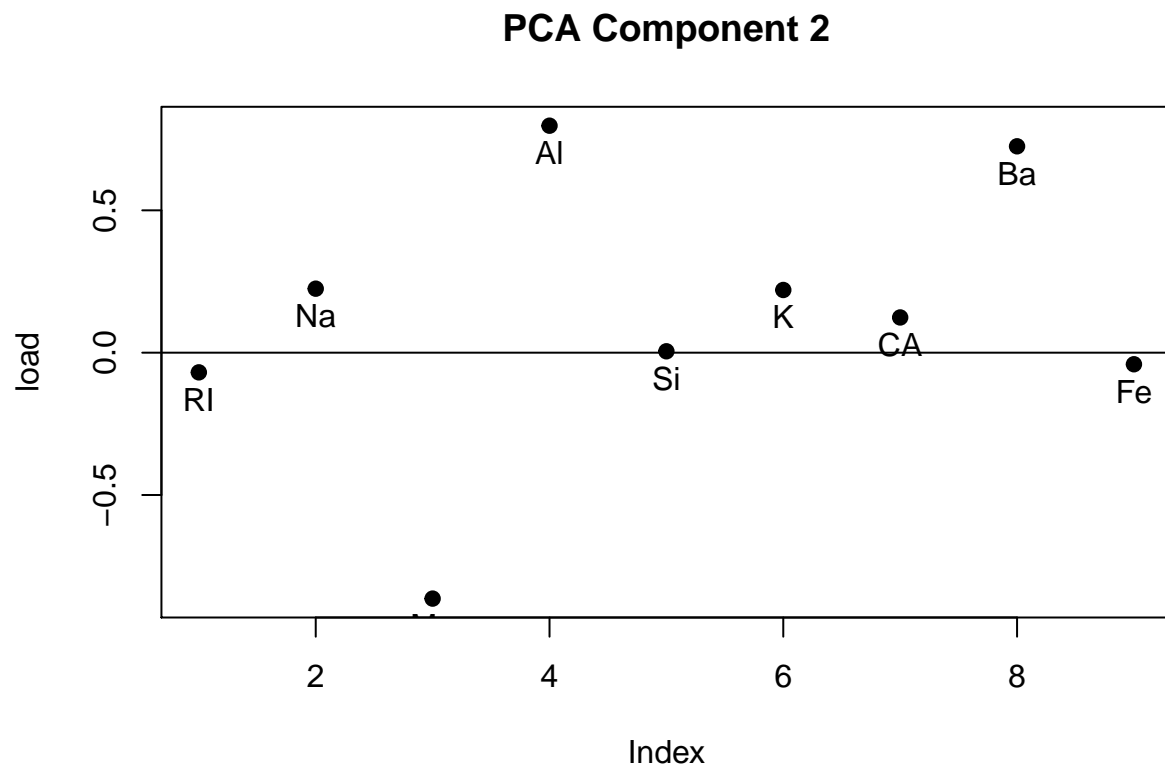
```
##           RC1          RC2          RC3          RC4
## [1,] 0.2516834 -1.1257154 -0.8331376  1.14203433
## [2,] -0.5120556 -0.5823124 -0.7217195  0.07184681
## [3,] -0.6811108 -0.4417522 -0.4610237 -0.39146231
## [4,] -0.4363986 -0.6266048 -0.1520952  0.09532063
## [5,] -0.4446499 -0.6485935 -0.1947898 -0.37616223
## [6,] -0.7149524 -0.2237372  1.1926990 -0.41874608
```

```
# Plot the components and analyze them
```

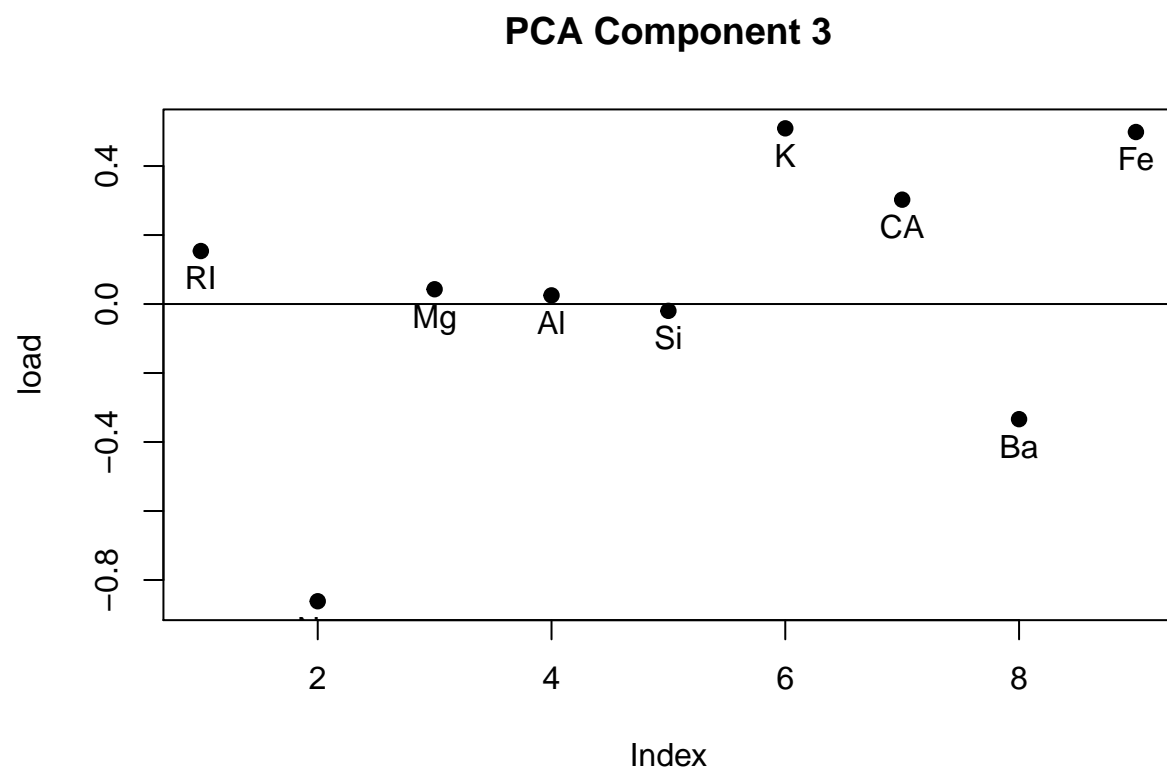
```
factor.plot(pc,
  choose = c(1),
  labels = colnames(glass_identification_data[, 2:10]),
  title = "PCA Component 1"
)
```



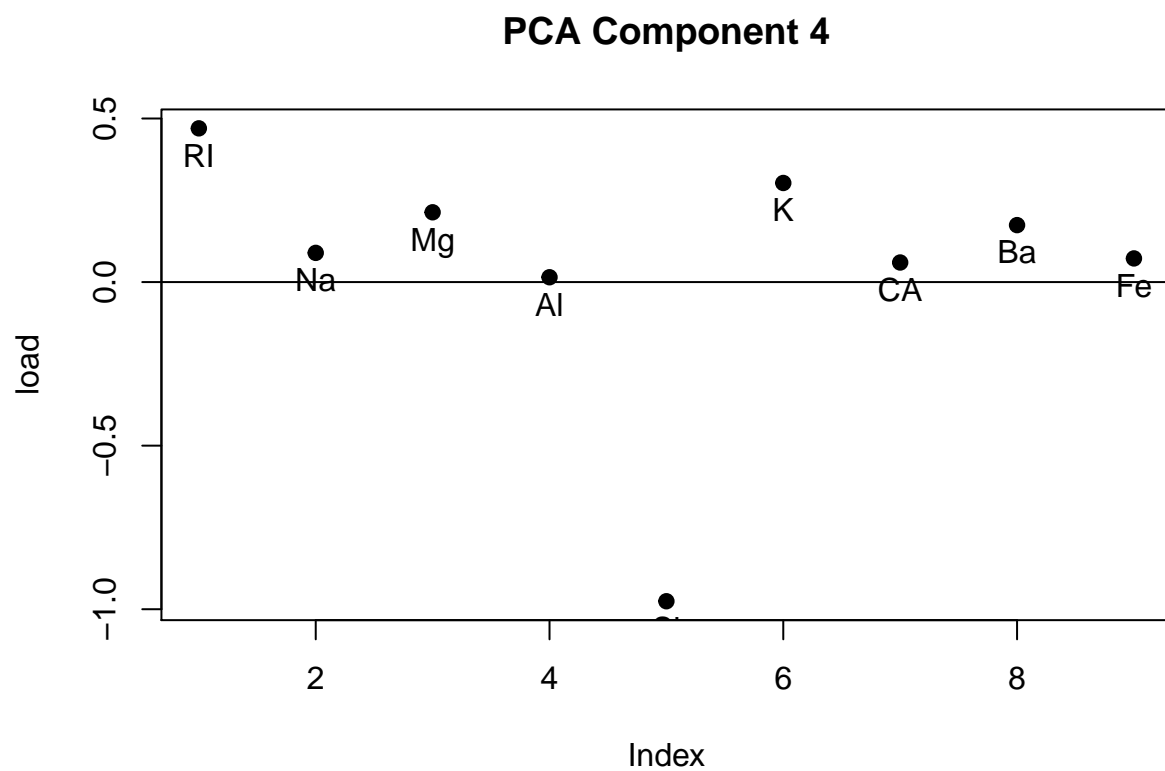
```
factor.plot(pc,
  choose = c(2),
  labels = colnames(glass_identification_data[, 2:10]),
  title = "PCA Component 2"
)
```



```
factor.plot(pc,
  choose = c(3),
  labels = colnames(glass_identification_data[, 2:10]),
  title = "PCA Component 3"
)
```

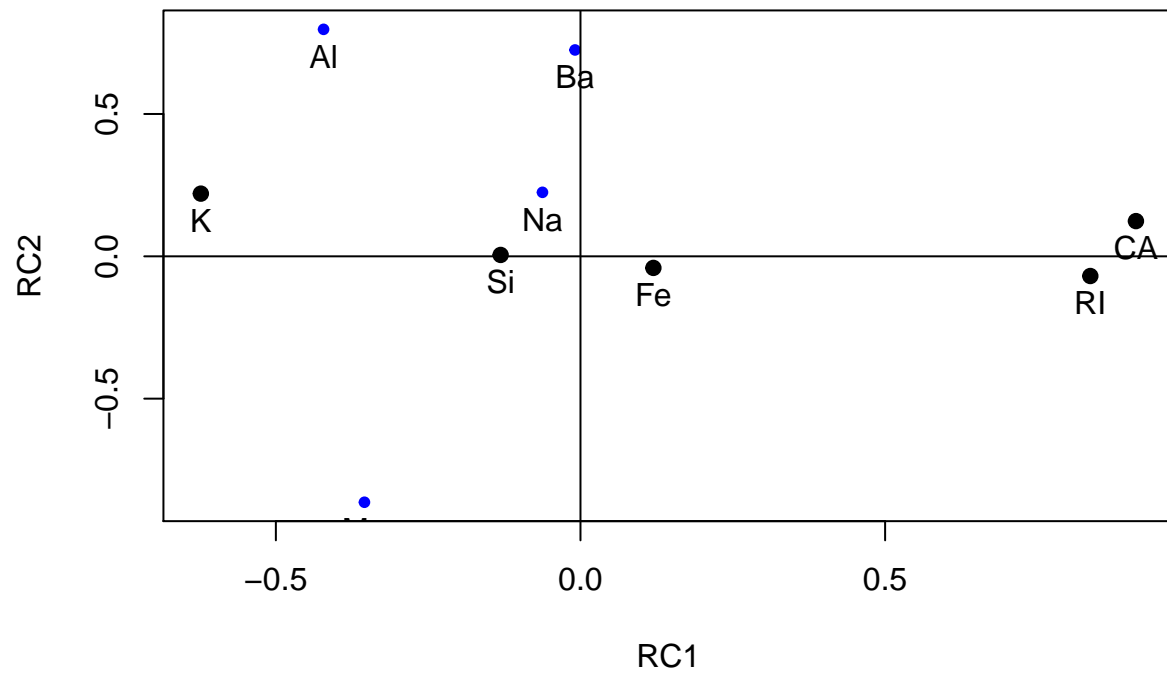



```
factor.plot(pc,  
  choose = c(4),  
  labels = colnames(glass_identification_data[, 2:10]),  
  title = "PCA Component 4"  
)
```



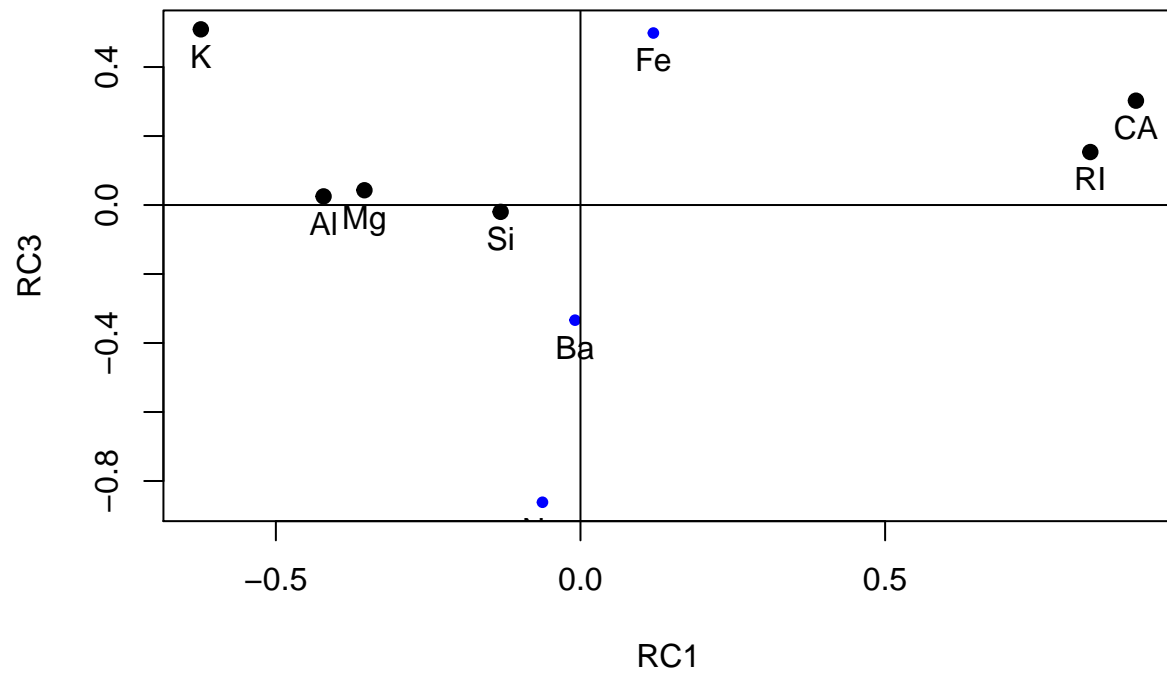
```
factor.plot(pc, choose = c(1, 2), labels = colnames(glass_identification_data[, 2:10]))
```

Principal Component Analysis



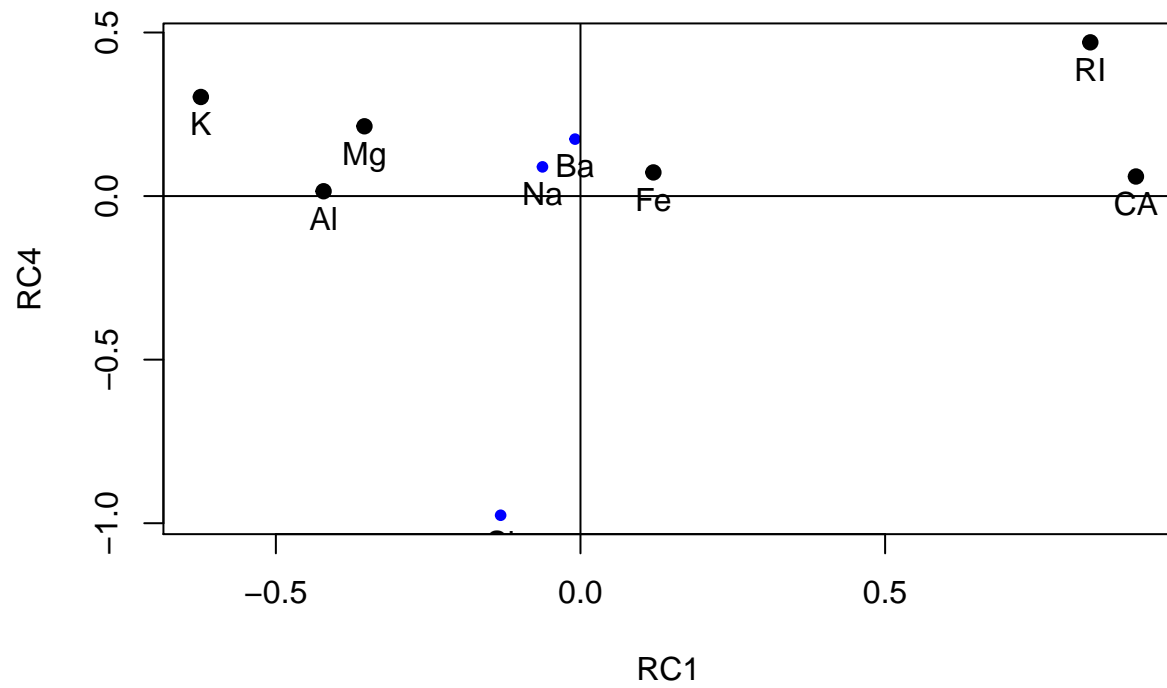
```
factor.plot(pc, choose = c(1, 3), labels = colnames(glass_identification_data[, 2:10]))
```

Principal Component Analysis



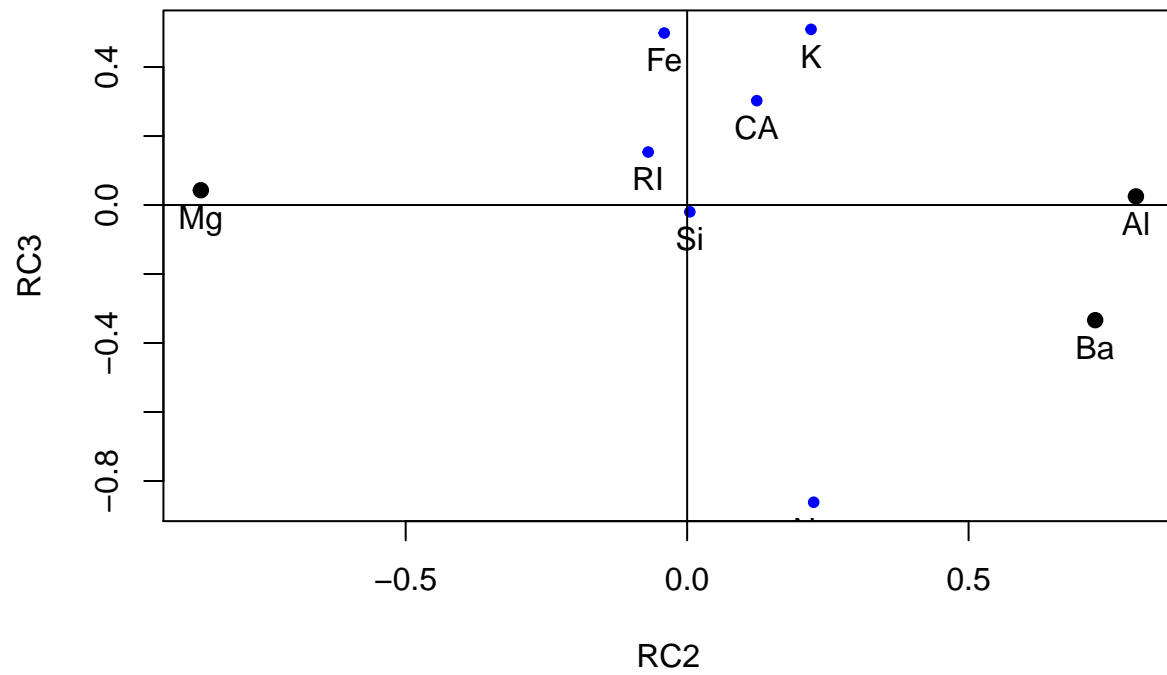
```
factor.plot(pc, choose = c(1, 4), labels = colnames(glass_identification_data[, 2:10]))
```

Principal Component Analysis



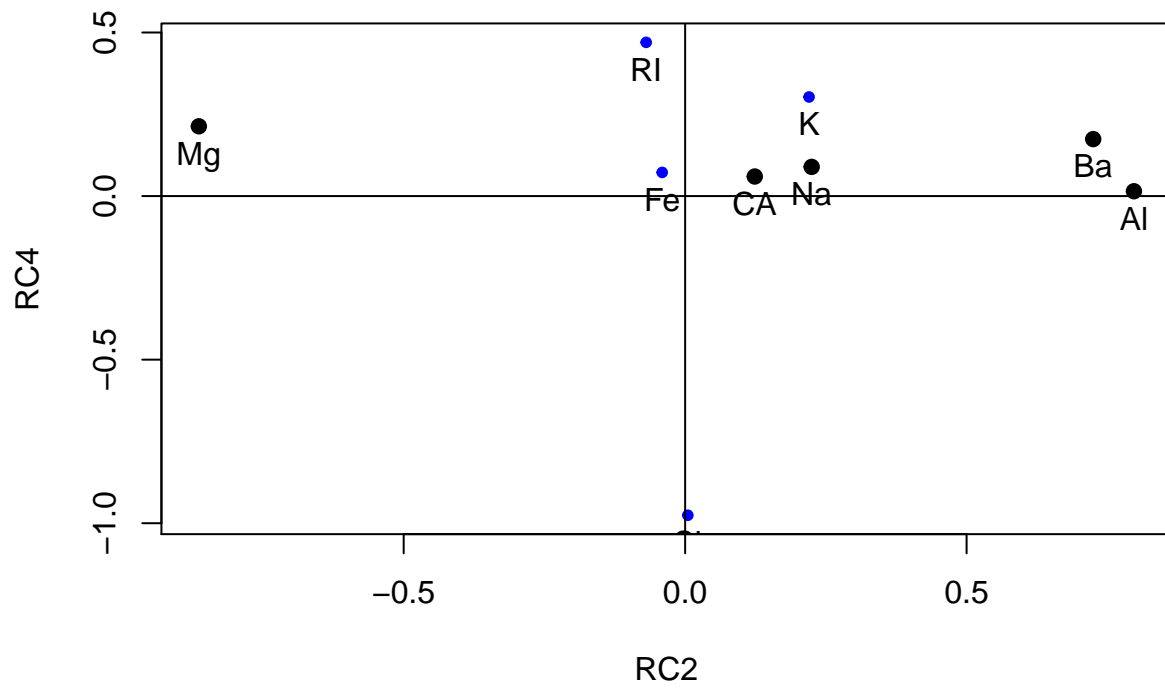
```
factor.plot(pc, choose = c(2, 3), labels = colnames(glass_identification_data[, 2:10]))
```

Principal Component Analysis



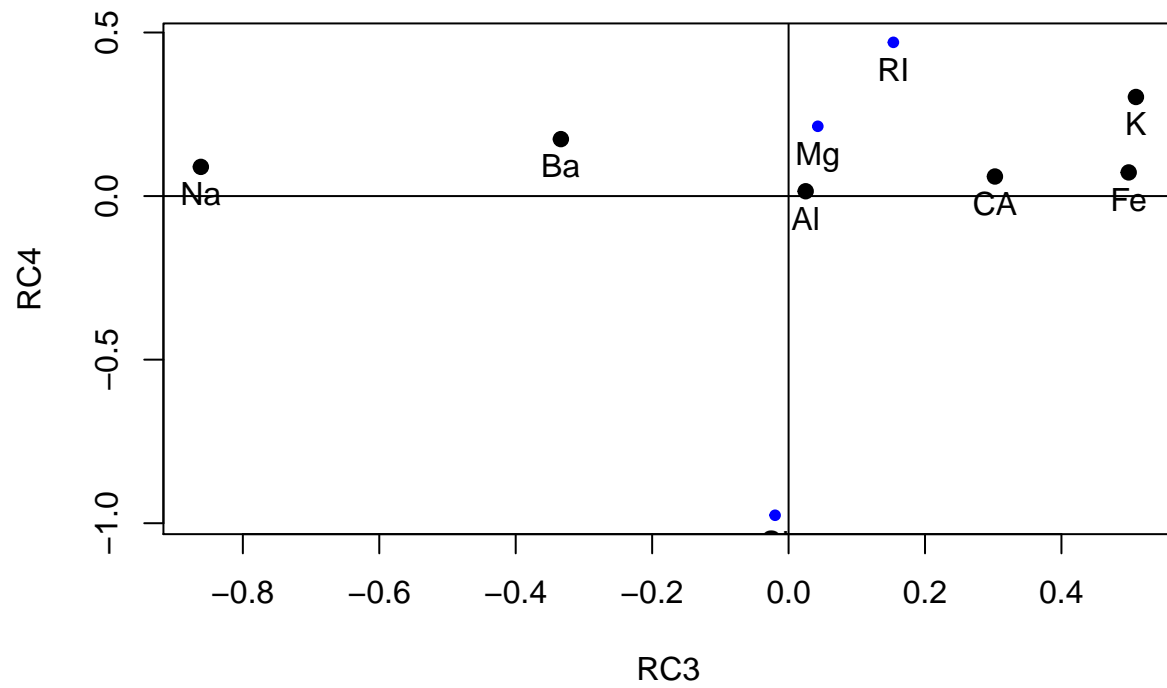
```
factor.plot(pc, choose = c(2, 4), labels = colnames(glass_identification_data[, 2:10]))
```

Principal Component Analysis



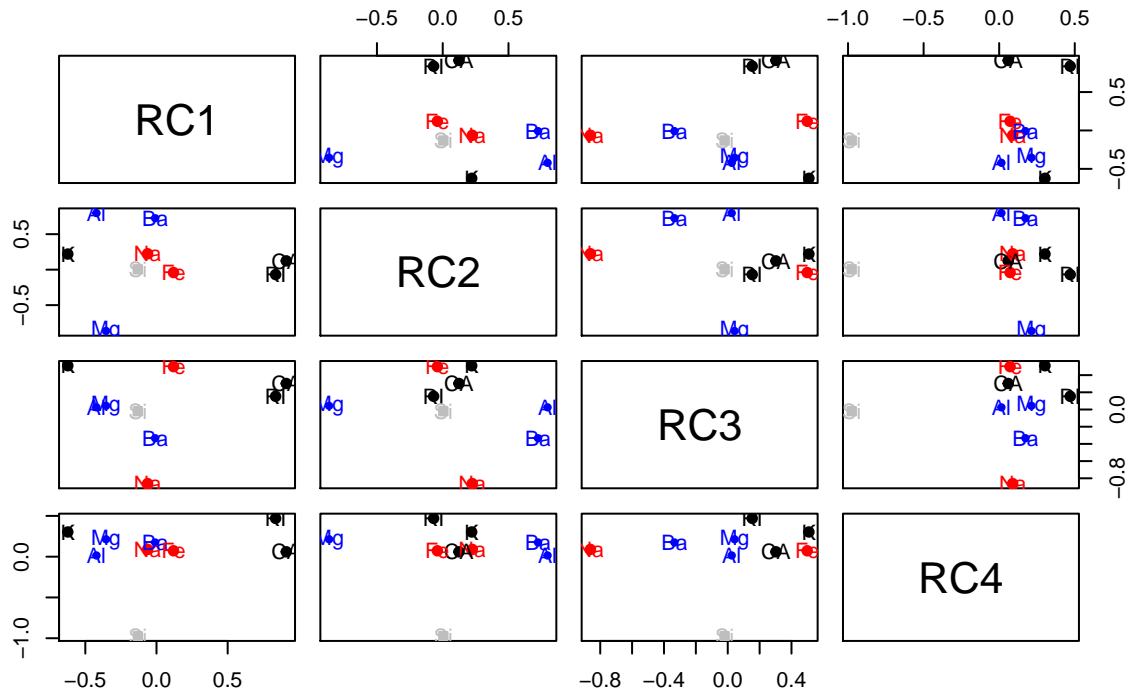
```
factor.plot(pc, choose = c(3, 4), labels = colnames(glass_identification_data[, 2:10]))
```

Principal Component Analysis



```
factor.plot(pc, labels = colnames(glass_identification_data[, 2:10]))
```


Principal Component Analysis



```
rm(list = ls())
```

Observations made - - PC1: loads RI, CA, Mg, Al and K.

- PC2: loads Mg, Al and Ba.
- PC3: loads Na, K, Ba and Fe.
- PC4: loads RI and Si.

Hence - 1. PC1 signifies Calcium, Potassium, Magnesium and Aluminum heavy glass. It also signifies glass with high refractive index.

2. PC2 signifies glass with high Magnesium, Aluminum and Barium concentration.
3. PC3 signifies glass with high Sodium, Potassium, Barium and Iron concentration.
4. PC4 signifies glass with high Refractive Index and Silicon concentration.

Problem 3

Perform factor analysis on Herman74.cor, which is a data structure available in the base installation (A correlation matrix of 24 psychological tests given to 145 seventh and eight-grade children in a Chicago suburb by Holzinger and Swineford).

- Input the correlation matrix to `fa.parallel()` function to determine the number of components to extract
- Input the correlation matrix to `fa()` function to extract the components. If raw data is input, the correlation matrix is automatically calculated by `fa()` function.

- Rotate the factors
- Compute factor scores
- Graph an orthogonal solution using factor.plot()
- Graph an oblique solutions using fa.diagram()
- Interpret the results

```
"Harman75.cor" <-
structure(list(cov = structure(c(
  1, 0.318, 0.403, 0.468, 0.321,
  0.335, 0.304, 0.332, 0.326, 0.116, 0.308, 0.314, 0.489, 0.125,
  0.238, 0.414, 0.176, 0.368, 0.27, 0.365, 0.369, 0.413, 0.474,
  0.282, 0.318, 1, 0.317, 0.23, 0.285, 0.234, 0.157, 0.157, 0.195,
  0.057, 0.15, 0.145, 0.239, 0.103, 0.131, 0.272, 0.005, 0.255,
  0.112, 0.292, 0.306, 0.232, 0.348, 0.211, 0.403, 0.317, 1, 0.305,
  0.247, 0.268, 0.223, 0.382, 0.184, -0.075, 0.091, 0.14, 0.321,
  0.177, 0.065, 0.263, 0.177, 0.211, 0.312, 0.297, 0.165, 0.25,
  0.383, 0.203, 0.468, 0.23, 0.305, 1, 0.227, 0.327, 0.335, 0.391,
  0.325, 0.099, 0.11, 0.16, 0.327, 0.066, 0.127, 0.322, 0.187,
  0.251, 0.137, 0.339, 0.349, 0.38, 0.335, 0.248, 0.321, 0.285,
  0.247, 0.227, 1, 0.622, 0.656, 0.578, 0.723, 0.311, 0.344, 0.215,
  0.344, 0.28, 0.229, 0.187, 0.208, 0.263, 0.19, 0.398, 0.318,
  0.441, 0.435, 0.42, 0.335, 0.234, 0.268, 0.327, 0.622, 1, 0.722,
  0.527, 0.714, 0.203, 0.353, 0.095, 0.309, 0.292, 0.251, 0.291,
  0.273, 0.167, 0.251, 0.435, 0.263, 0.386, 0.431, 0.433, 0.304,
  0.157, 0.223, 0.335, 0.656, 0.722, 1, 0.619, 0.685, 0.246, 0.232,
  0.181, 0.345, 0.236, 0.172, 0.18, 0.228, 0.159, 0.226, 0.451,
  0.314, 0.396, 0.405, 0.437, 0.332, 0.157, 0.382, 0.391, 0.578,
  0.527, 0.619, 1, 0.532, 0.285, 0.3, 0.271, 0.395, 0.252, 0.175,
  0.296, 0.255, 0.25, 0.274, 0.427, 0.362, 0.357, 0.501, 0.388,
  0.326, 0.195, 0.184, 0.325, 0.723, 0.714, 0.685, 0.532, 1, 0.17,
  0.28, 0.113, 0.28, 0.26, 0.248, 0.242, 0.274, 0.208, 0.274, 0.446,
  0.266, 0.483, 0.504, 0.424, 0.116, 0.057, -0.075, 0.099, 0.311,
  0.203, 0.246, 0.285, 0.17, 1, 0.484, 0.585, 0.408, 0.172, 0.154,
  0.124, 0.289, 0.317, 0.19, 0.173, 0.405, 0.16, 0.262, 0.531,
  0.308, 0.15, 0.091, 0.11, 0.344, 0.353, 0.232, 0.3, 0.28, 0.484,
  1, 0.428, 0.535, 0.35, 0.24, 0.314, 0.362, 0.35, 0.29, 0.202,
  0.399, 0.304, 0.251, 0.412, 0.314, 0.145, 0.14, 0.16, 0.215,
  0.095, 0.181, 0.271, 0.113, 0.585, 0.428, 1, 0.512, 0.131, 0.173,
  0.119, 0.278, 0.349, 0.11, 0.246, 0.355, 0.193, 0.35, 0.414,
  0.489, 0.239, 0.321, 0.327, 0.344, 0.309, 0.345, 0.395, 0.28,
  0.408, 0.535, 0.512, 1, 0.195, 0.139, 0.281, 0.194, 0.323, 0.263,
  0.241, 0.425, 0.279, 0.382, 0.358, 0.125, 0.103, 0.177, 0.066,
  0.28, 0.292, 0.236, 0.252, 0.26, 0.172, 0.35, 0.131, 0.195, 1,
  0.37, 0.412, 0.341, 0.201, 0.206, 0.302, 0.183, 0.243, 0.242,
  0.304, 0.238, 0.131, 0.065, 0.127, 0.229, 0.251, 0.172, 0.175,
  0.248, 0.154, 0.24, 0.173, 0.139, 0.37, 1, 0.325, 0.345, 0.334,
  0.192, 0.272, 0.232, 0.246, 0.256, 0.165, 0.414, 0.272, 0.263,
  0.322, 0.187, 0.291, 0.18, 0.296, 0.242, 0.124, 0.314, 0.119,
  0.281, 0.412, 0.325, 1, 0.324, 0.344, 0.258, 0.388, 0.348, 0.283,
  0.36, 0.262, 0.176, 0.005, 0.177, 0.187, 0.208, 0.273, 0.228,
  0.255, 0.274, 0.289, 0.362, 0.278, 0.194, 0.341, 0.345, 0.324,
  1, 0.448, 0.324, 0.262, 0.173, 0.273, 0.287, 0.326, 0.368, 0.255,
```

```

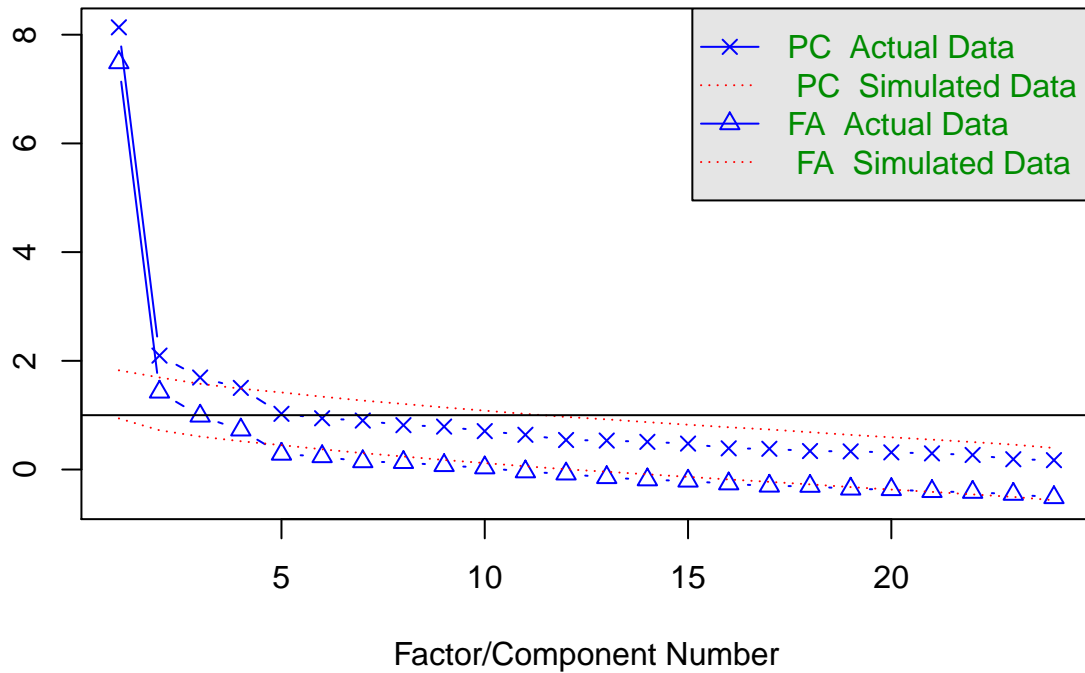
0.211, 0.251, 0.263, 0.167, 0.159, 0.25, 0.208, 0.317, 0.35,
0.349, 0.323, 0.201, 0.334, 0.344, 0.448, 1, 0.358, 0.301, 0.357,
0.317, 0.272, 0.405, 0.27, 0.112, 0.312, 0.137, 0.19, 0.251,
0.226, 0.274, 0.274, 0.19, 0.29, 0.11, 0.263, 0.206, 0.192, 0.258,
0.324, 0.358, 1, 0.167, 0.331, 0.342, 0.303, 0.374, 0.365, 0.292,
0.297, 0.339, 0.398, 0.435, 0.451, 0.427, 0.446, 0.173, 0.202,
0.246, 0.241, 0.302, 0.272, 0.388, 0.262, 0.301, 0.167, 1, 0.413,
0.463, 0.509, 0.366, 0.369, 0.306, 0.165, 0.349, 0.318, 0.263,
0.314, 0.362, 0.266, 0.405, 0.399, 0.355, 0.425, 0.183, 0.232,
0.348, 0.173, 0.357, 0.331, 0.413, 1, 0.374, 0.451, 0.448, 0.413,
0.232, 0.25, 0.38, 0.441, 0.386, 0.396, 0.357, 0.483, 0.16, 0.304,
0.193, 0.279, 0.243, 0.246, 0.283, 0.273, 0.317, 0.342, 0.463,
0.374, 1, 0.503, 0.375, 0.474, 0.348, 0.383, 0.335, 0.435, 0.431,
0.405, 0.501, 0.504, 0.262, 0.251, 0.35, 0.382, 0.242, 0.256,
0.36, 0.287, 0.272, 0.303, 0.509, 0.451, 0.503, 1, 0.434, 0.282,
0.211, 0.203, 0.248, 0.42, 0.433, 0.437, 0.388, 0.424, 0.531,
0.412, 0.414, 0.358, 0.304, 0.165, 0.262, 0.326, 0.405, 0.374,
0.366, 0.448, 0.375, 0.434, 1
), .Dim = c(24, 24), .Dimnames = list(
  c(
    "VisualPerception", "Cubes", "PaperFormBoard", "Flags",
    "GeneralInformation", "ParagraphComprehension", "SentenceCompletion",
    "WordClassification", "WordMeaning", "Addition", "Code",
    "CountingDots", "StraightCurvedCapitals", "WordRecognition",
    "NumberRecognition", "FigureRecognition", "ObjectNumber",
    "NumberFigure", "FigureWord", "Deduction", "NumericalPuzzles",
    "ProblemReasoning", "SeriesCompletion", "ArithmeticProblems"
  ), c(
    "VisualPerception", "Cubes", "PaperFormBoard", "Flags",
    "GeneralInformation", "ParagraphComprehension", "SentenceCompletion",
    "WordClassification", "WordMeaning", "Addition", "Code",
    "CountingDots", "StraightCurvedCapitals", "WordRecognition",
    "NumberRecognition", "FigureRecognition", "ObjectNumber",
    "NumberFigure", "FigureWord", "Deduction", "NumericalPuzzles",
    "ProblemReasoning", "SeriesCompletion", "ArithmeticProblems"
  )
), center = c(
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
), n.obs = 145), .Names = c(
  "cov",
  "center", "n.obs"
))

fa.parallel(Harman74.cor$cov,
  n.obs = Harman74.cor$n.obs,
  n.iter = 100
)

```

eigenvalues of principal components and factor analysis

Parallel Analysis Scree Plots



Parallel analysis suggests that the number of factors = 4 and the number of components = 3

The Screeplot suggests 4 components.

Perform Factor Analysis

```
correlations <- cov2cor(Harman74.cor$cov)
```

```
fa <- fa(correlations, nfactors = 4, rotate = "none", scores = TRUE)
```

```
fa
```

```
## Factor Analysis using method = minres
```

```
## Call: fa(r = correlations, nfactors = 4, rotate = "none", scores = TRUE)
```

```
## Standardized loadings (pattern matrix) based upon correlation matrix
```

```
##
```

	MR1	MR2	MR3	MR4	h2	u2	com
## VisualPerception	0.60	0.03	0.38	-0.22	0.55	0.45	2.0
## Cubes	0.37	-0.03	0.26	-0.15	0.23	0.77	2.2
## PaperFormBoard	0.42	-0.12	0.36	-0.13	0.34	0.66	2.3
## Flags	0.48	-0.11	0.26	-0.19	0.35	0.65	2.0
## GeneralInformation	0.69	-0.30	-0.27	-0.04	0.64	0.36	1.7
## ParagraphComprehension	0.69	-0.40	-0.20	0.08	0.68	0.32	1.8
## SentenceCompletion	0.68	-0.41	-0.30	-0.08	0.73	0.27	2.1
## WordClassification	0.67	-0.19	-0.09	-0.11	0.51	0.49	1.3
## WordMeaning	0.70	-0.45	-0.23	0.08	0.74	0.26	2.0
## Addition	0.47	0.53	-0.48	-0.10	0.74	0.26	3.1
## Code	0.56	0.36	-0.16	0.09	0.47	0.53	2.0
## CountingDots	0.47	0.50	-0.14	-0.24	0.55	0.45	2.6
## StraightCurvedCapitals	0.60	0.26	0.01	-0.29	0.51	0.49	1.9
## WordRecognition	0.43	0.06	0.01	0.42	0.36	0.64	2.0

```

## NumberRecognition      0.39  0.10  0.09  0.37  0.31  0.69  2.2
## FigureRecognition      0.51  0.09  0.35  0.25  0.45  0.55  2.3
## ObjectNumber           0.47  0.21 -0.01  0.39  0.41  0.59  2.4
## NumberFigure           0.52  0.32  0.16  0.14  0.41  0.59  2.1
## FigureWord             0.44  0.10  0.10  0.13  0.23  0.77  1.4
## Deduction              0.62 -0.13  0.14  0.04  0.42  0.58  1.2
## NumericalPuzzles       0.59  0.21  0.07 -0.14  0.42  0.58  1.4
## ProblemReasoning       0.61 -0.10  0.12  0.03  0.40  0.60  1.1
## SeriesCompletion        0.69 -0.06  0.15 -0.10  0.51  0.49  1.2
## ArithmeticProblems     0.65  0.17 -0.19  0.00  0.49  0.51  1.3
##
##
##              MR1  MR2  MR3  MR4
## SS loadings      7.65 1.69 1.22 0.92
## Proportion Var    0.32 0.07 0.05 0.04
## Cumulative Var    0.32 0.39 0.44 0.48
## Proportion Explained 0.67 0.15 0.11 0.08
## Cumulative Proportion 0.67 0.81 0.92 1.00
##
## Mean item complexity = 1.9
## Test of the hypothesis that 4 factors are sufficient.
##
## The degrees of freedom for the null model are 276 and the objective function was 11.44
## The degrees of freedom for the model are 186 and the objective function was 1.72
##
## The root mean square of the residuals (RMSR) is 0.04
## The df corrected root mean square of the residuals is 0.05
##
## Fit based upon off diagonal values = 0.98
## Measures of factor score adequacy
##
##              MR1  MR2  MR3  MR4
## Correlation of (regression) scores with factors 0.97 0.91 0.87 0.79
## Multiple R square of scores with factors        0.94 0.82 0.75 0.62
## Minimum correlation of possible factor scores    0.89 0.65 0.50 0.24

```

Rotate the factors

```

fa_orthogonal <- fa(correlations, nfactors = 4, rotate = "varimax", scores = TRUE)

fa_orthogonal

```

```

## Factor Analysis using method = minres
## Call: fa(r = correlations, nfactors = 4, rotate = "varimax", scores = TRUE)
## Standardized loadings (pattern matrix) based upon correlation matrix
##
##              MR1  MR3  MR2  MR4  h2  u2  com
## VisualPerception 0.15 0.68 0.20 0.15 0.55 0.45 1.4
## Cubes            0.11 0.45 0.08 0.08 0.23 0.77 1.3
## PaperFormBoard   0.15 0.55 -0.01 0.11 0.34 0.66 1.2
## Flags            0.23 0.53 0.09 0.07 0.35 0.65 1.5
## GeneralInformation 0.73 0.19 0.22 0.14 0.64 0.36 1.4
## ParagraphComprehension 0.76 0.21 0.07 0.23 0.68 0.32 1.4
## SentenceCompletion 0.81 0.19 0.15 0.07 0.73 0.27 1.2
## WordClassification 0.57 0.34 0.23 0.14 0.51 0.49 2.2
## WordMeaning       0.81 0.20 0.05 0.22 0.74 0.26 1.3
## Addition          0.17 -0.11 0.82 0.16 0.74 0.26 1.2
## Code              0.18 0.11 0.54 0.37 0.47 0.53 2.1

```

```
## CountingDots          0.02  0.20  0.71 0.09 0.55 0.45 1.2
## StraightCurvedCapitals 0.18  0.42  0.54 0.08 0.51 0.49 2.2
## WordRecognition       0.21  0.05  0.08 0.56 0.36 0.64 1.3
## NumberRecognition     0.12  0.12  0.08 0.52 0.31 0.69 1.3
## FigureRecognition     0.07  0.42  0.06 0.52 0.45 0.55 2.0
## ObjectNumber          0.14  0.06  0.22 0.58 0.41 0.59 1.4
## NumberFigure          0.02  0.31  0.34 0.45 0.41 0.59 2.7
## FigureWord            0.15  0.25  0.18 0.35 0.23 0.77 2.8
## Deduction             0.38  0.42  0.10 0.29 0.42 0.58 2.9
## NumericalPuzzles      0.18  0.40  0.43 0.21 0.42 0.58 2.8
## ProblemReasoning      0.37  0.41  0.13 0.29 0.40 0.60 3.0
## SeriesCompletion       0.37  0.52  0.23 0.22 0.51 0.49 2.7
## ArithmeticProblems    0.36  0.19  0.49 0.29 0.49 0.51 2.9
##
##                      MR1  MR3  MR2  MR4
## SS loadings          3.64 2.93 2.67 2.23
## Proportion Var       0.15 0.12 0.11 0.09
## Cumulative Var       0.15 0.27 0.38 0.48
## Proportion Explained 0.32 0.26 0.23 0.19
## Cumulative Proportion 0.32 0.57 0.81 1.00
##
## Mean item complexity = 1.9
## Test of the hypothesis that 4 factors are sufficient.
##
## The degrees of freedom for the null model are 276 and the objective function was 11.44
## The degrees of freedom for the model are 186 and the objective function was 1.72
##
## The root mean square of the residuals (RMSR) is 0.04
## The df corrected root mean square of the residuals is 0.05
##
## Fit based upon off diagonal values = 0.98
## Measures of factor score adequacy
##
##                      MR1  MR3  MR2  MR4
## Correlation of (regression) scores with factors 0.93 0.87 0.91 0.82
## Multiple R square of scores with factors        0.87 0.76 0.83 0.68
## Minimum correlation of possible factor scores    0.74 0.52 0.65 0.36
```

Let's see the factor scores

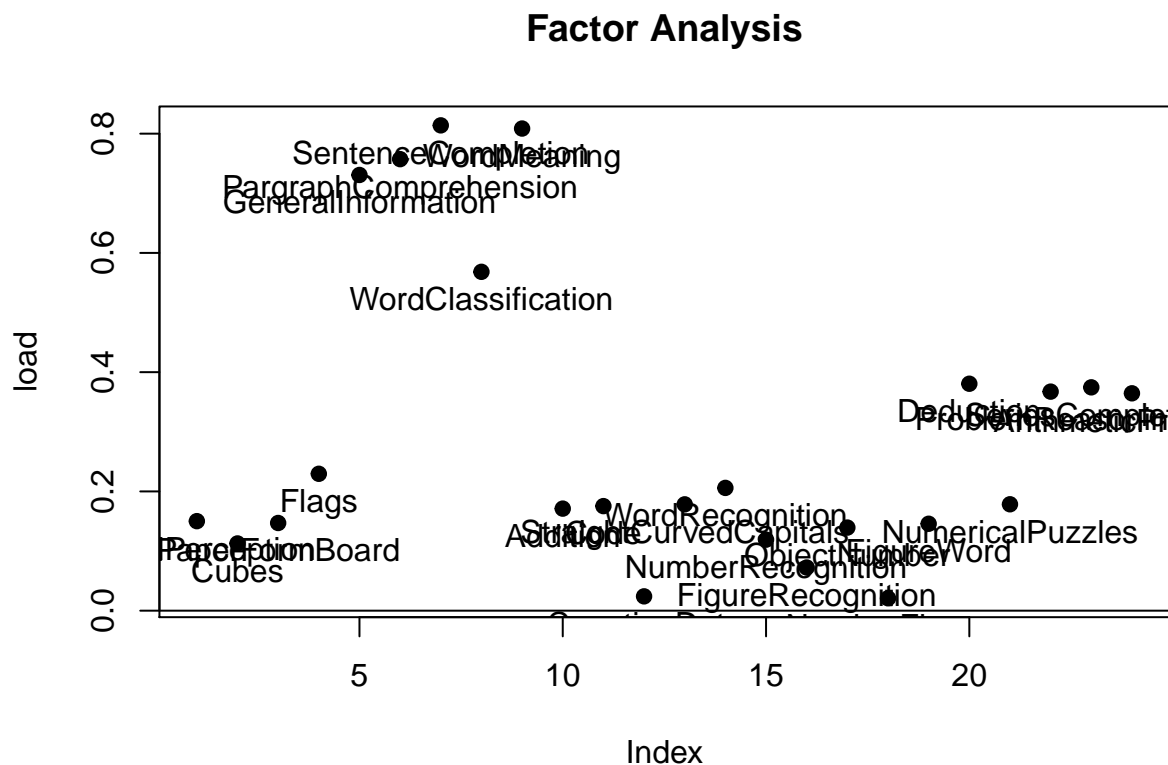
```
fa_orthogonal$weights
```

```
##                      MR1          MR3          MR2          MR4
## VisualPerception    -0.0912528883  0.30864122  0.019076056 -0.059475135
## Cubes                -0.0273878717  0.12425214 -0.003998151 -0.036046103
## PaperFormBoard      -0.0002664514  0.14435978 -0.021455124 -0.013910017
## Flags                -0.0158992046  0.16513630 -0.018187085 -0.067188757
## GeneralInformation   0.1791509833 -0.01576399  0.006192041 -0.064803732
## ParagraphComprehension 0.2087548894 -0.02967329 -0.089257772  0.053969292
## SentenceCompletion   0.3560671670 -0.07802378  0.009878325 -0.152657827
## WordClassification   0.0744517660  0.07646981  0.004918710 -0.042449858
## WordMeaning          0.3541762245 -0.12345571 -0.083274087  0.047708409
## Addition             0.0370184288 -0.28588788  0.530044962 -0.033006173
## Code                 -0.0152781994 -0.07727463  0.134310508  0.115144252
## CountingDots         -0.0568764212  0.04221931  0.241300880 -0.083059603
## StraightCurvedCapitals -0.0599618479  0.14994774  0.162459388 -0.110602297
```

```
## WordRecognition      0.0022332713 -0.07875443 -0.049514025  0.253344226
## NumberRecognition    -0.0305344160 -0.04164026 -0.040528450  0.215786525
## FigureRecognition    -0.0678653395  0.09799311 -0.077886130  0.223301180
## ObjectNumber         -0.0435663449 -0.07170630 -0.015111696  0.278035153
## NumberFigure         -0.0730633557  0.05023790  0.036233555  0.166892502
## FigureWord           -0.0300328554  0.02668857  0.002475198  0.083460720
## Deduction            0.0074291922  0.09386230 -0.040441038  0.055443652
## NumericalPuzzles     -0.0250346901  0.11130425  0.071040209 -0.001990422
## ProblemReasoning     0.0145135443  0.07473804 -0.014585388  0.057911235
## SeriesCompletion     0.0124897301  0.17157853 -0.007344283 -0.017896058
## ArithmeticProblems   0.0062265043  0.00390309  0.085098413  0.047726702
```

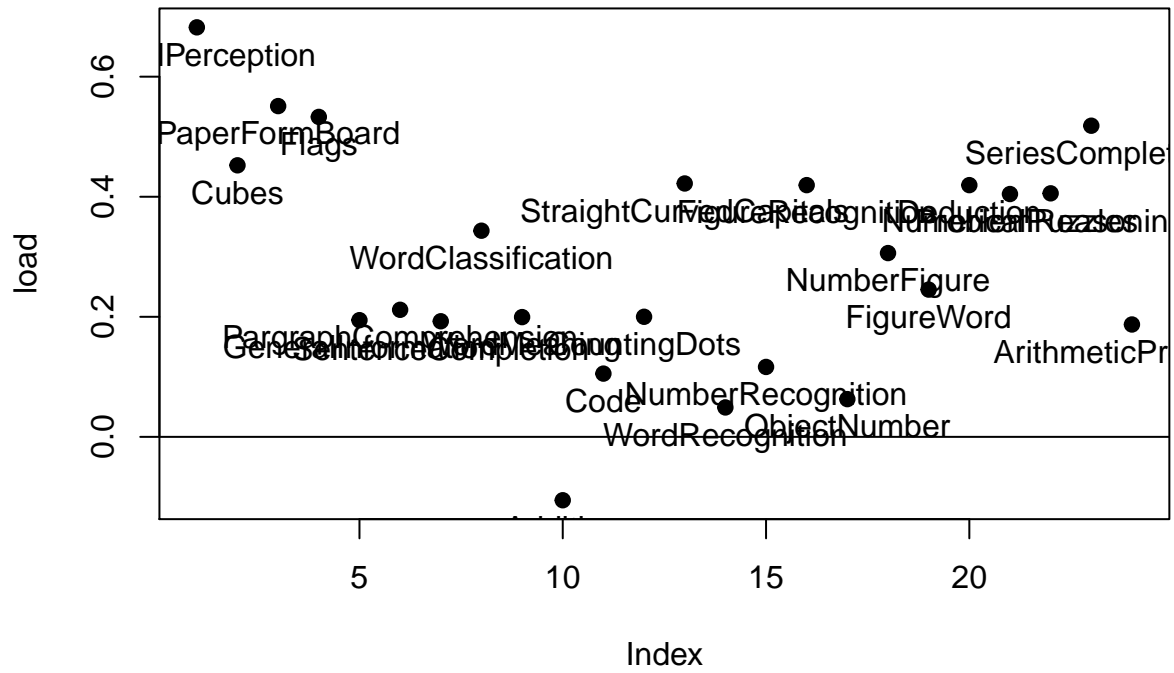
Plotting an orthogonal solution

```
factor.plot(fa_orthogonal, choose = c(1), labels = rownames(fa$loadings))
```



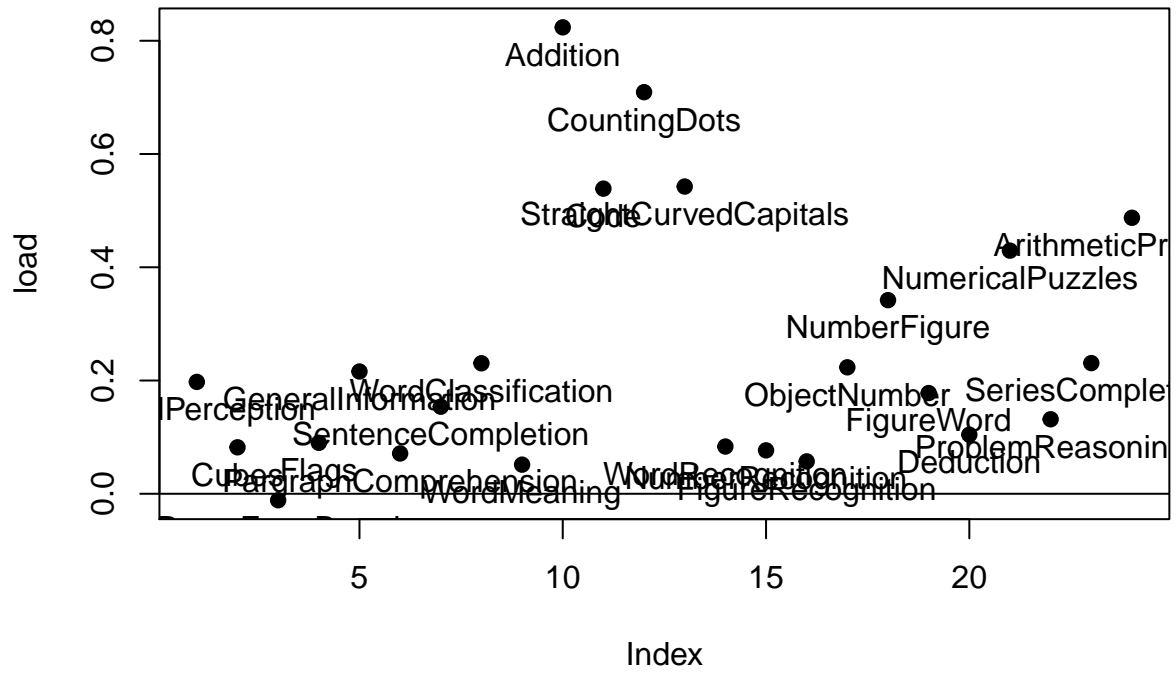
```
factor.plot(fa_orthogonal, choose = c(2), labels = rownames(fa$loadings))
```

Factor Analysis



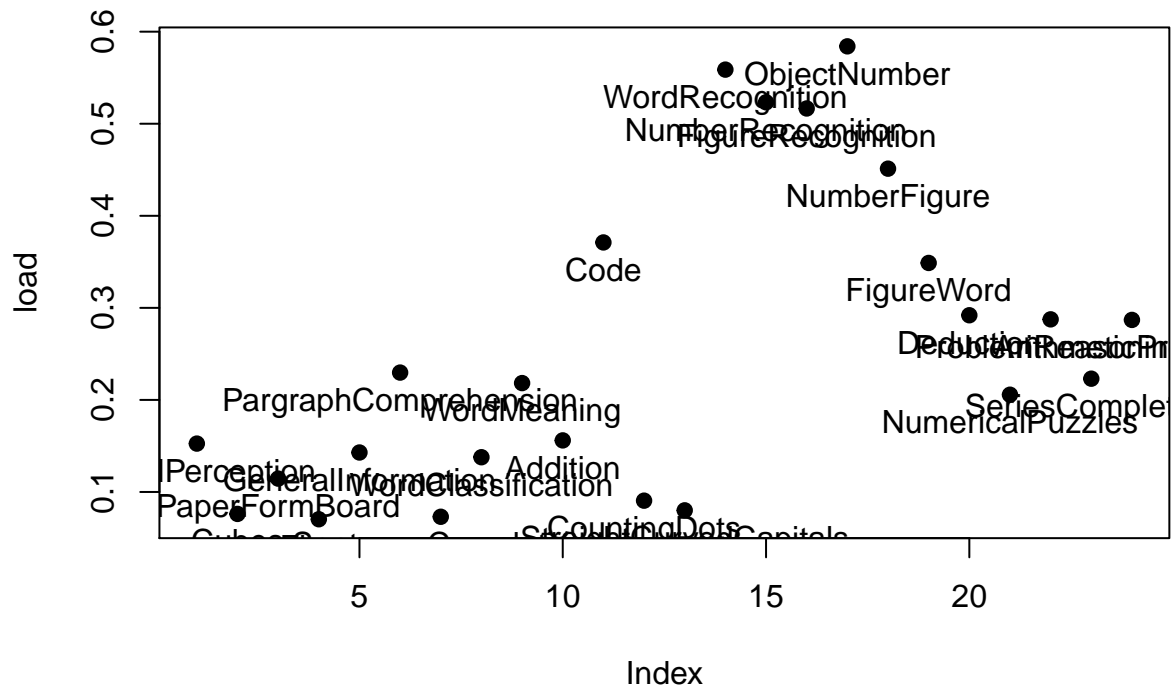
```
factor.plot(fa_orthogonal, choose = c(3), labels = rownames(fa$loadings))
```


Factor Analysis



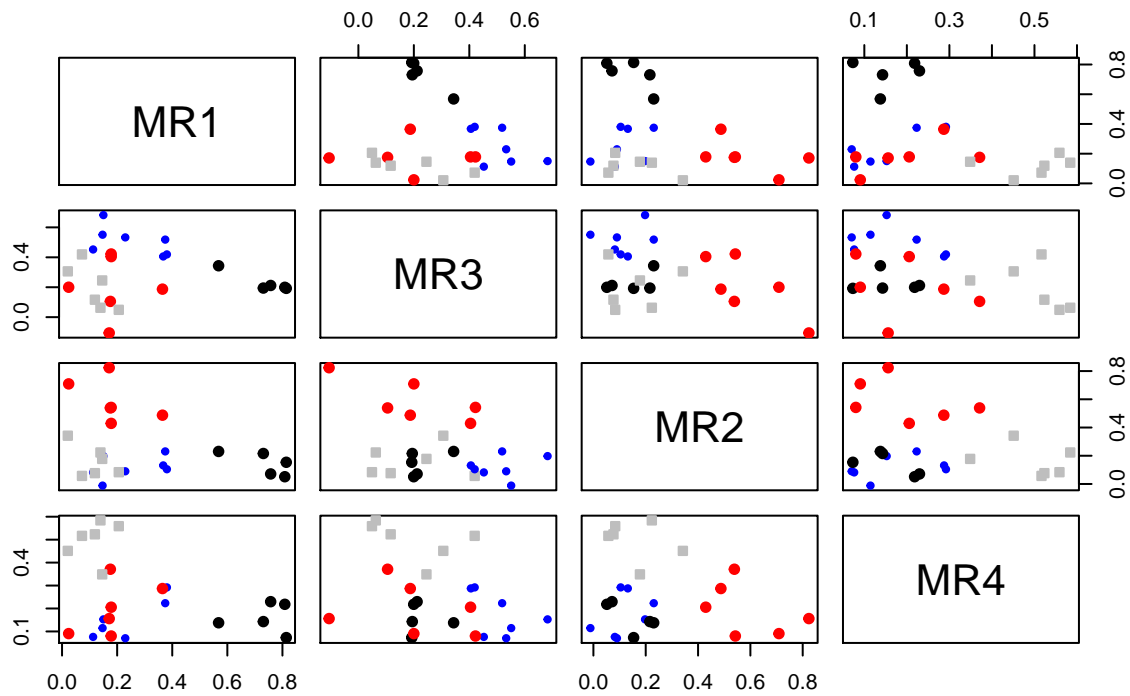
```
factor.plot(fa_orthogonal, choose = c(4), labels = rownames(fa$loadings))
```

Factor Analysis



```
factor.plot(fa_orthogonal)
```

Factor Analysis



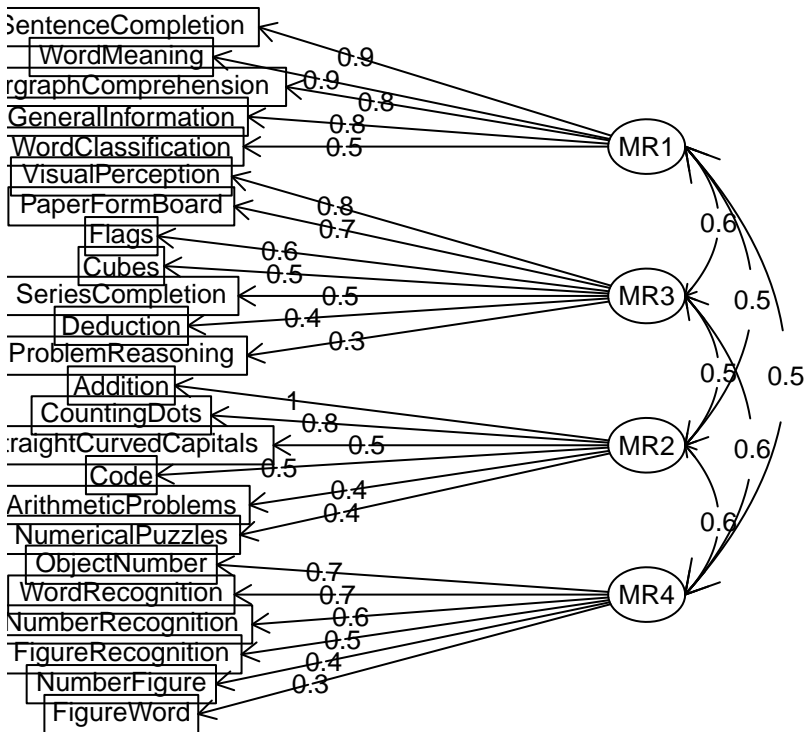
Now let's plot an oblique solution

```
fa_oblique <- fa(Harman74.cor$cov, nfactors = 4, rotate = "promax", scores = TRUE)
```

```
## Loading required namespace: GPArotation
```

```
fa.diagram(fa_oblique)
```

Factor Analysis



```
rm(list = ls())
```

We can make the following observations -

- Factor 1 seems to be related to language related attributes like Sentence Completion, WordMeaning, Word Classification etc.
- Factor 2 seems to be related to more general problem solving attributes like Deduction, Problem Reasoning, Cubes, Flage etc.
- Factor 3 seems to be related to mathematical attributes like Code, Counting Data, Addition, Number Recognition etc.
- Factor 4 seems to be related to Word and Number recognition.

Problem 5

Perform multidimensional scaling on Vertebral Column Data.xlsx

- Input the raw data matrix to `fa.parallel()` function to determine the number of components to extract
- Input the raw data matrix to `cmdscale()` function to perform multidimensional scaling. `cmdscale()` function which is available in the base installation performs a classical multidimensional scaling.
- Graph an orthogonal solution using `factor.plot()`
- Interpret the results

```
vertebral_column_data <- data.frame(read_xlsx("./data/Vertebral Column Data.xlsx",
  sheet = "column_3C"
))
```

```
fa.parallel(vertebral_column_data[, 1:6], n.iter = 100)
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect. Try a
## different factor score estimation method.
```

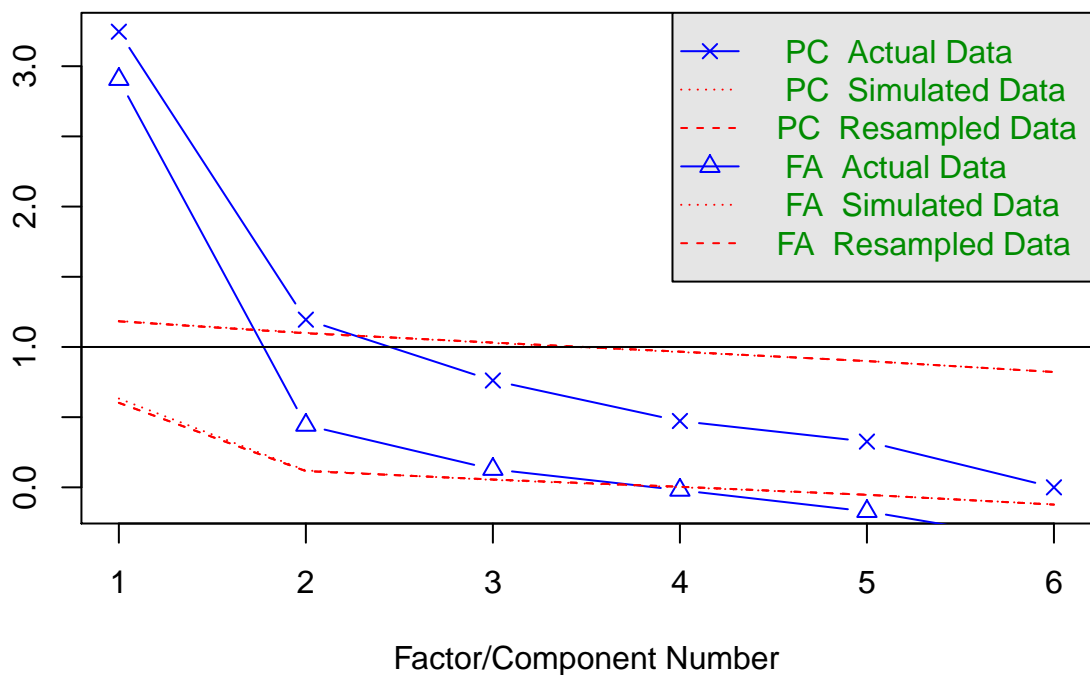
```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected. Examine the results carefully
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect. Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected. Examine the results carefully
```

eigenvalues of principal components and factor analysis

Parallel Analysis Scree Plots



```
## Parallel analysis suggests that the number of factors = 3 and the number of components = 2
```

Scree Plots suggests number of factors 3

```
# Calculate the distance matrix
d <- dist(vertebral_column_data[, 1:6])
```

```
# Perform Multidimensional Scaling
fit <- cmdscale(d, k = 3, eig = TRUE)
```

The MDS has reduced the data to 3 dimensions

```
head(fit$points)
```

```
##           [,1]      [,2]      [,3]
## [1,] -25.21264  13.204206 -15.891671
## [2,] -37.55028 -18.951621 -11.839171
## [3,] -21.95087  23.063614  -6.318516
## [4,] -10.84709  13.917984 -12.971068
## [5,] -27.73305  -7.589005 -18.435332
## [6,] -39.74800 -22.959841   2.545529
```

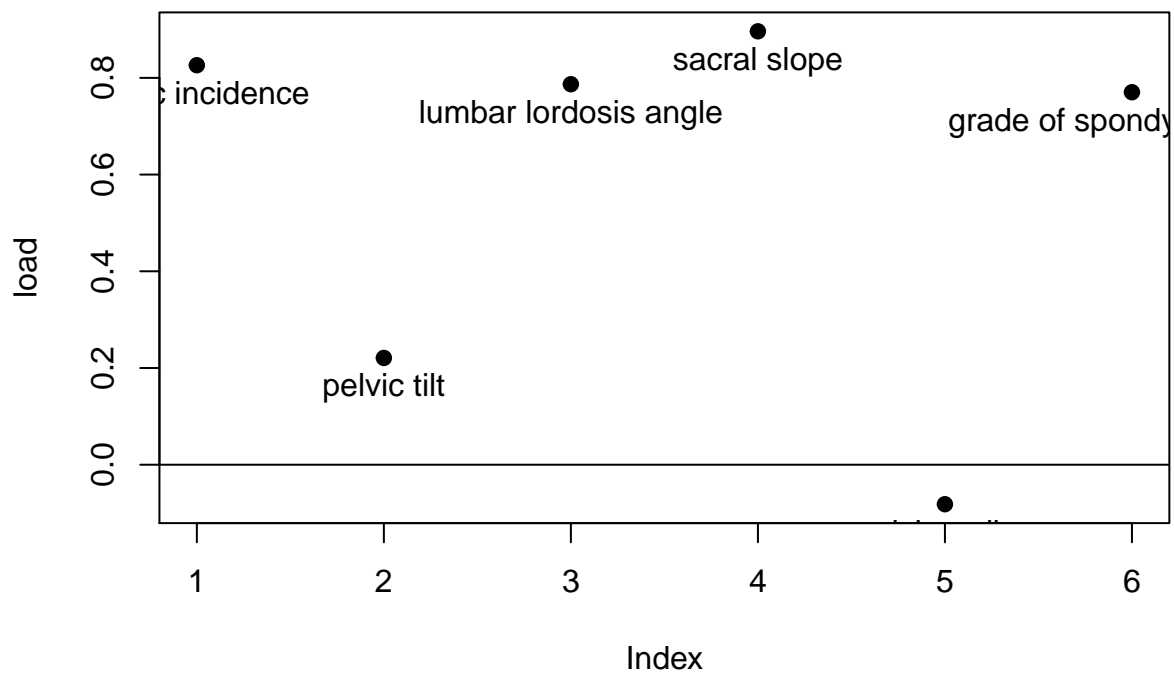
This is the same as calculating 3 factors.

Let's graph the orthogonal solution now

```
pc <- principal(vertebral_column_data[, 1:6], nfactors = 3, rotate = "varimax")
```

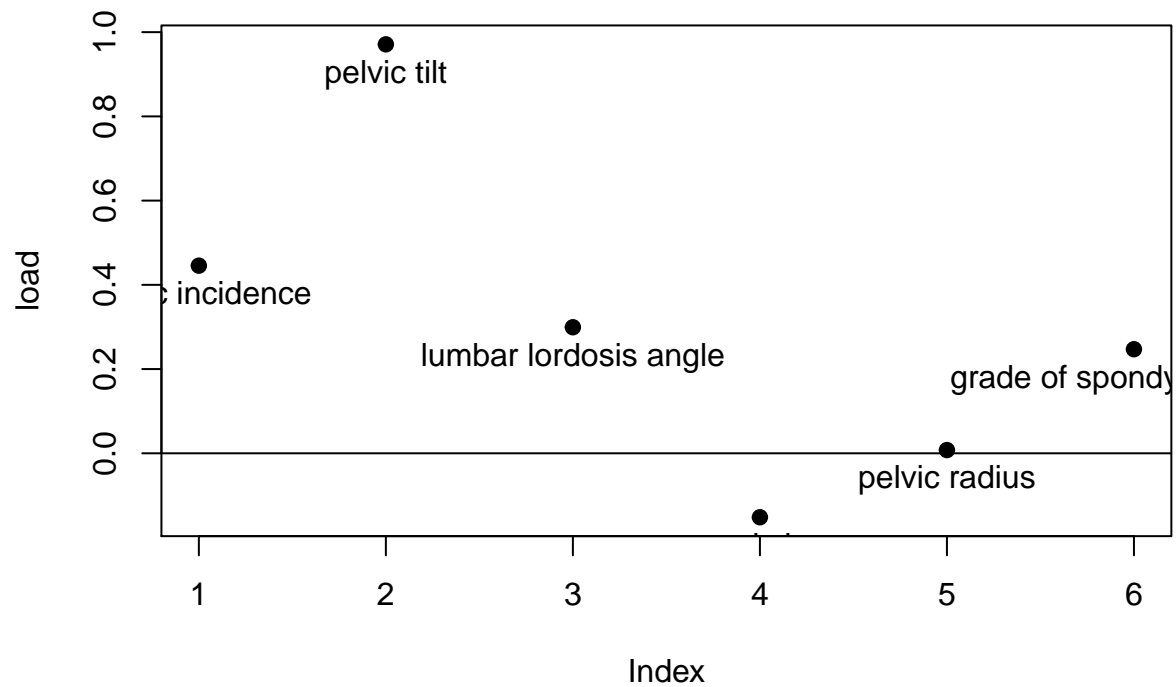
```
factor.plot(pc, choose = c(1), labels = c(
  "pelvic incidence", "pelvic tilt",
  "lumbar lordosis angle", "sacral slope",
  "pelvic radius", "grade of spondylolisthesis."
))
```

Principal Component Analysis



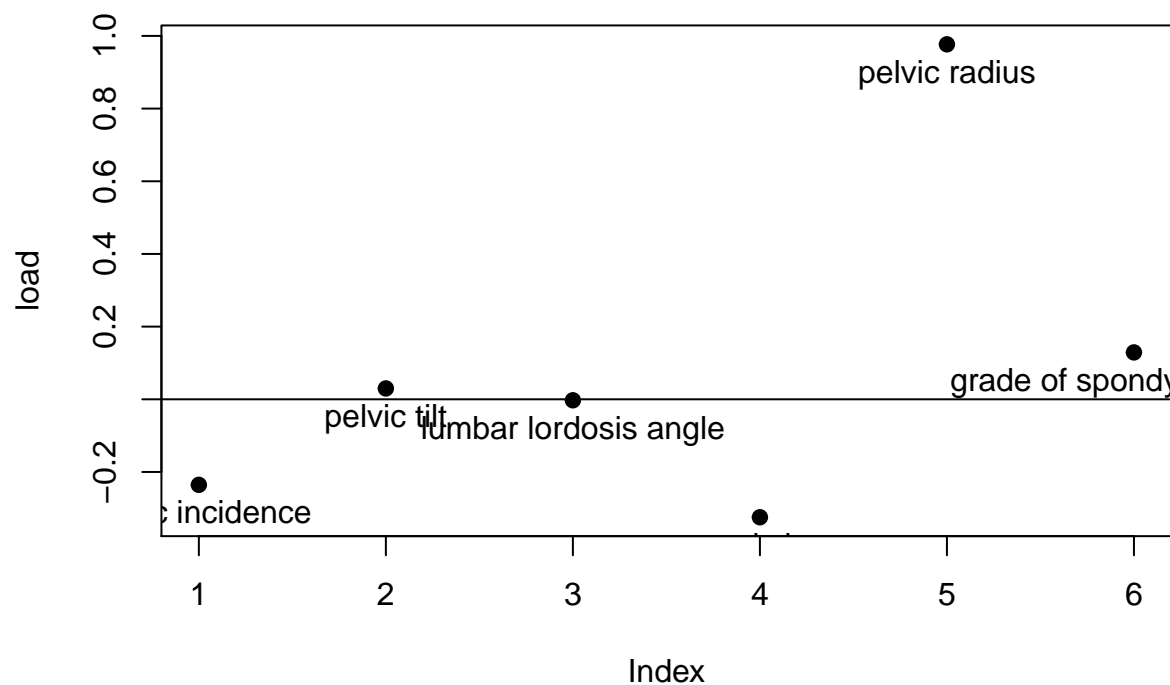
```
factor.plot(pc, choose = c(2), labels = c(
  "pelvic incidence", "pelvic tilt",
  "lumbar lordosis angle", "sacral slope",
  "pelvic radius", "grade of spondylolisthesis."
))
```

Principal Component Analysis



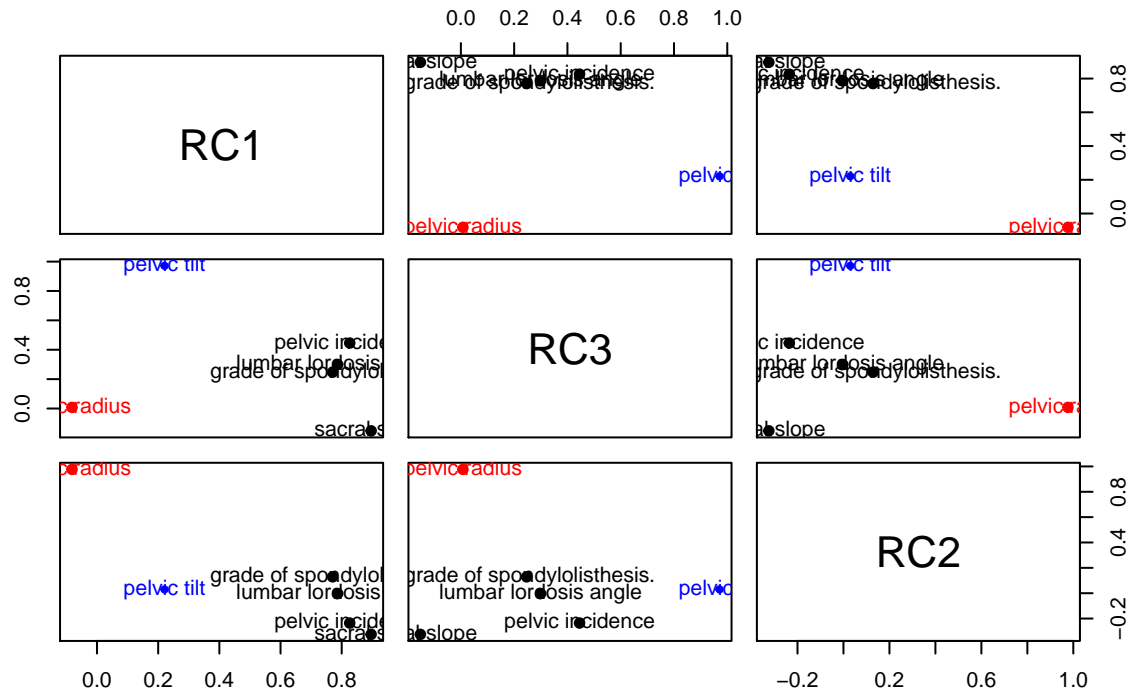
```
factor.plot(pc, choose = c(3), labels = c(
  "pelvic incidence", "pelvic tilt",
  "lumbar lordosis angle", "sacral slope",
  "pelvic radius", " grade of spondylolisthesis."
))
```

Principal Component Analysis



```
factor.plot(pc, labels = c(
  "pelvic incidence", "pelvic tilt",
  "lumbar lordosis angle", "sacral slope",
  "pelvic radius", "grade of spondylolisthesis."
))
```


Principal Component Analysis



We can make the following observations -

- Component 1 loads pelvic incidence, lumbar lordosis angle, sacral slope and grade of spondylolisthesis.
- Component 2 loads pelvic incidence and pelvic tilt.
- Component 3 loads sacral slope and pelvic radius.