Problem Set 4

Di Tong

11/09/2019

```
knitr::opts_chunk$set(echo = TRUE)
```

Load and Prepare Data

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0       v purrr   0.3.0
## v tibble  2.0.1       v dplyr   0.8.0.1
## v tidyr   0.8.2       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

```
## -- Conflicts -------------------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
countries <- read_csv("/Users/ditong/Documents/uml/Problem-Set-4/countries.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   X1 = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
con_std <- as.data.frame(scale(countries[,2:22]))
con_names <- countries$X1
```

Factor Analysis

1. How do CFA and EFA differ?

The difference between CFA and EFA corresponds to the difference between CDA and EDA. CFA has pre-established theory and the hypothesis generated from theories for testing. It serves to examine if the number of factors and input feature loadings confirm to what is hypothesized or not. EFA, on the other hand, sets up without pre-established expectations and theoretical hypotheses. On the contrary, it atheoretical utilizes factor loadings to discover the factor structure of the data. Rather than guiding the analysis, the research question emerges from the EFA.

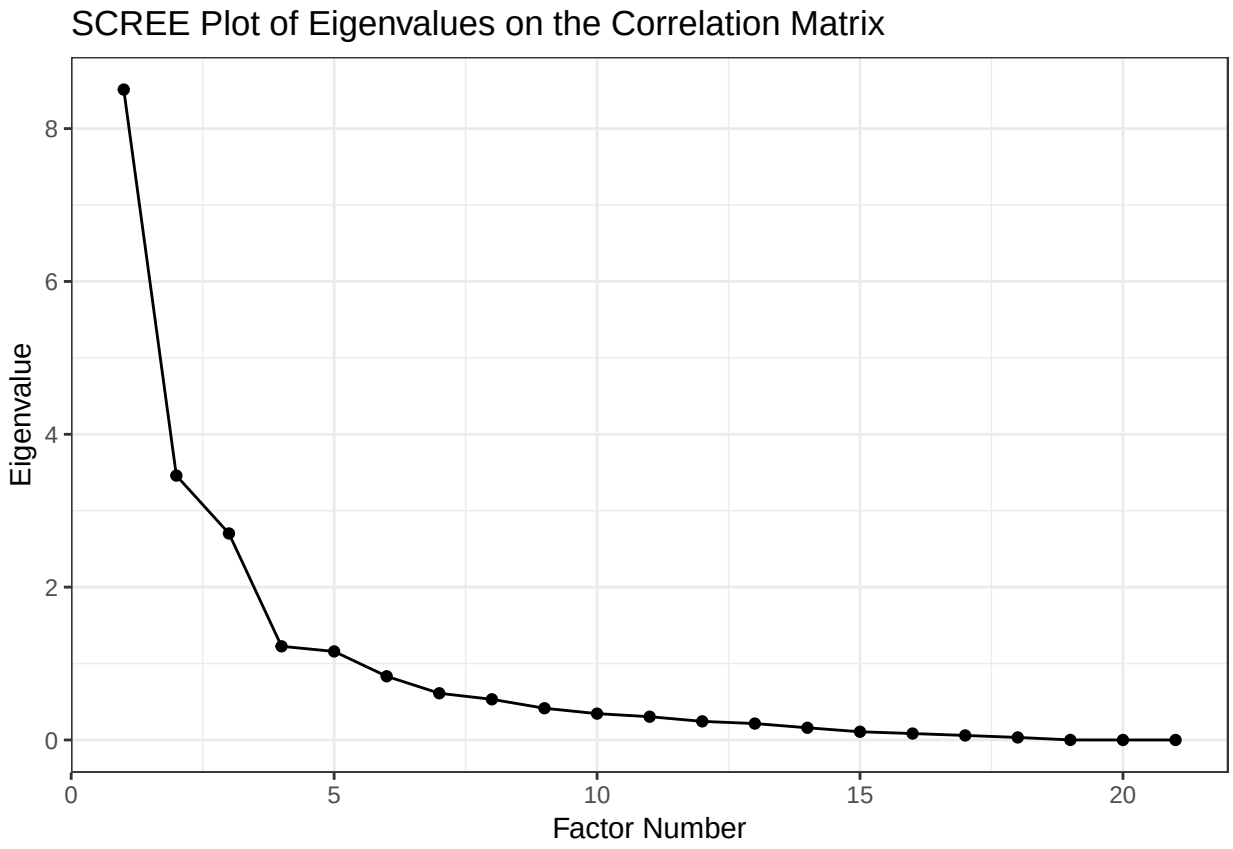
2. Fit three exploratory factor analysis models initialized at 2, 3, and 4 factors. Present the loadings from these solutions and discuss in substantive terms. How does each fit? What sense does this give you of the underlying dimensionality of the space? And so on.

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
#store the correlation matrix
con_cor <- cor(con_std)
#generate the eigenvalues
con_ev <- eigen(con_cor) # store EVs on the correlation matrix
#generate Scree plot
qplot(y = con_ev$values,
      main = 'SCREE Plot of Eigenvalues on the Correlation Matrix',
      xlab = 'Factor Number',
      ylab = 'Eigenvalue') +
    geom_line() +
    theme_bw()
```




```
library(psych)

#exploratory factor analysis models initialized at 2 factors
factan.2 <- fa(con_std,
               nfactors = 2)
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done
```

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

```
## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
## 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 extraction method.
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

```
## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
#present the loadings
factan.2$loadings
```

```
##
## Loadings:
##             MR1    MR2
## idealpoint   0.449  0.429
## polity       0.995
## polity2      0.995
## democ        0.931
## autoc       -0.969  0.159
## unreg        0.412 -0.131
## physint             0.782
## speech       0.631  0.154
## new_empinx   0.802  0.197
## wecon               0.509
## wopol        0.551
## wosoc        0.286  0.497
## elecsd       0.852
## gdp.pc.wdi          0.673
## gdp.pc.un           0.671
## pop.wdi      0.204 -0.476
## amnesty            -0.821
## statedept          -0.849
## milper       0.158 -0.468
## cinc         0.211 -0.366
## domestic9    0.288 -0.479
##
##                  MR1   MR2
## SS loadings    6.523 4.527
## Proportion Var 0.311 0.216
## Cumulative Var 0.311 0.526
```

```
library(psych)

#exploratory factor analysis models initialized at 3 factors
factan.3 <- fa(con_std,
               nfactors = 3)
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done

## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done

## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done

## 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 extraction method.

## In factor.scores, the correlation matrix is singular, an approximation is used

## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
#present the loadings
factan.3$loadings
```

```
##
## Loadings:
##             MR1    MR2    MR3
## idealpoint   0.432  0.468
## polity       0.992
## polity2      0.992
## democ        0.910  0.144
## autoc       -0.994  0.191
## unreg        0.413 -0.129
## physint             0.737 -0.136
## speech       0.646  0.128
## new_empinx   0.840  0.131 -0.125
## wecon               0.518
## wopol        0.552
## wosoc        0.263  0.547
## elecsd       0.858
## gdp.pc.wdi          0.856  0.158
## gdp.pc.un           0.853  0.157
## pop.wdi                    0.892
## amnesty            -0.715  0.243
## statedept          -0.803  0.144
## milper                     0.949
## cinc                       0.999
## domestic9    0.269 -0.443
##
##                  MR1   MR2   MR3
## SS loadings    6.466 4.275 2.881
## Proportion Var 0.308 0.204 0.137
## Cumulative Var 0.308 0.512 0.649
```

```
library(psych)

#exploratory factor analysis models initialized at 4 factors
factan.4 <- fa(con_std,
               nfactors = 4)
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done

## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done

## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done

## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done

## 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 extraction method.

## In factor.scores, the correlation matrix is singular, an approximation is used

## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
#present the loadings
factan.4$loadings
```

```
##
## Loadings:
##             MR1    MR3    MR4    MR2
## idealpoint   0.467         0.214 -0.294
## polity       0.995
## polity2      0.995
## democ        0.922         0.127
## autoc       -0.986         0.146
## unreg        0.405                 0.165
## physint      0.119                -0.761
## speech       0.658                -0.109
## new_empinx   0.855                -0.145
## wecon        0.105         0.390 -0.170
## wopol        0.555
## wosoc        0.300         0.350 -0.239
## elecsd       0.865
## gdp.pc.wdi                 0.986
## gdp.pc.un                  0.979
## pop.wdi             0.923
## amnesty             0.177 -0.197  0.602
## statedept  -0.137         -0.139  0.783
## milper              0.965
## cinc                0.981  0.111
## domestic9    0.247         0.204  0.757
##
##                  MR1   MR3   MR4   MR2
## SS loadings    6.605 2.811 2.426 2.370
## Proportion Var 0.315 0.134 0.116 0.113
## Cumulative Var 0.315 0.448 0.564 0.677
```

The 2-factor solution show that the first factor is about the regime type in terms of democratic level (captured by polity, polity2, democ, autoc, new_empinx, elecsd). The second factor is concerned with the economic and physical features (statedept, amnesty, physint, gdp, wecon). In the 3-factor solution, the first and

second factors are the same as that of the 2-factor solution. The third factor focuses on the military strength (milper, cinc) and population size (pop.wdi), which could also serve as an indicator of miliatry strength. As for the 4-factor solution, the first and the third factors are the same as the 3-factor solution (political regime and militray strength), the second and fourth factors divide the second factor in the 3-factor solution into physical intergreity/stability features and economic development level. Judging from this loading comparison of different solutions, the 4-factor model seems to most comprehensively capture the underlying dimensions. The SCREE plot also provide supportive evidence, as the first five eigenvalues are greater than 1, showing that 5 factors would be sufficent to capture all variances. Yet the fifth factor is already close to 1, so maybe the 4-factor model is good enough to represent the underlying dimensionality.

3. Rotate the 3-factor solution using any oblique method you would like and present a visual of the unrotated and rotated versions side-by-side. How do these differ and why does this matter (or not)?

```
library(psych)
library(lattice)
library(GPArotation)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## Initial (unrotated) factor solution
nonrotated.factors <- fa(con_cor,
        fm = "pa", # communalities along the diagonal (total variation across features)
        nfactors = 3,
        rotate = "none",
        residuals = TRUE)
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done
```

```
## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
## 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 extraction method.
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

```
## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
# Plot unrotated factor pattern
nonrot.pattern <- as.data.frame(nonrotated.factors$loadings[1:11,])

unrot_plot <- xyplot(PA2 ~ PA1, data = nonrot.pattern,
       aspect = 1,
       xlim = c(-.8, 1.3),
       ylim = c(-.6, .8),
       panel = function (x, y) {
         panel.segments(c(0, 0), c(0, 0),
             c(1, 0), c(0, 1), col = "gray")
         panel.text(1, 0, labels = "Initial\n(unrotated)\nfactor 1",
                    cex = .65, pos = 3, col = "gray")
```

```
          panel.text(0, .7, labels = "Initial\n(unrotated)\nfactor 2",
                     cex = .65, pos = 4, col = "gray")
          panel.segments(rep(0, 8), rep(0, 8), x, y,
             col = "black")
          panel.text(x[-7], y[-7], labels = rownames(nonrot.pattern)[-7],
             pos = 4, cex = .75)
          panel.text(x[7], y[7], labels = rownames(nonrot.pattern)[7],
                     pos = 1, cex = .75)
        },
        main = "Unrotated Factor Pattern",
        xlab = "",
        ylab = "",
        scales = list(x = list(at = c(0, 1)),
                      y = list(at = c(-.4, 0, .6)))
)

# Orthogonal (varimax) rotated factor solution
promax.factors <- fa(con_cor,
                     fm = "pa",
                     nfactors = 3,
                     rotate = "Promax")
```

```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate =
## rotate, : A loading greater than abs(1) was detected. Examine the loadings
## carefully.
```

```
## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
## 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 extraction method.
```

```
## In factor.scores, the correlation matrix is singular, an approximation is used
```

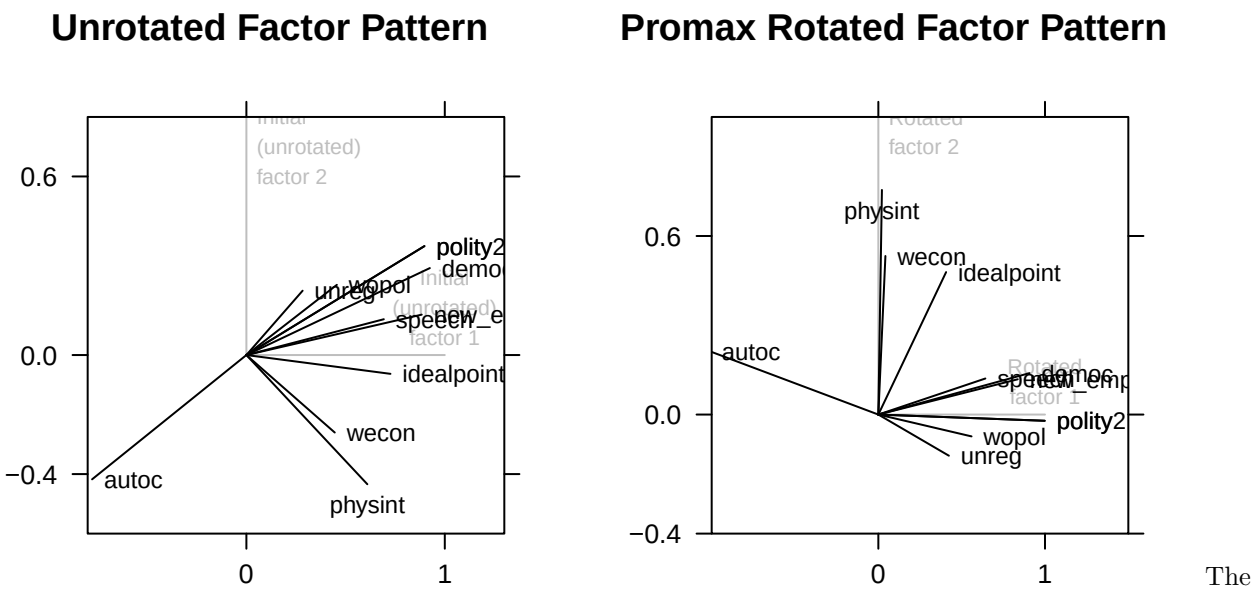
```
## Warning in cor.smooth(r): Matrix was not positive definite, smoothing was
## done
```

```
# Plot the factor pattern
promax.pattern <- as.data.frame(promax.factors$loadings[1:11,])

rot_plot <- xyplot(PA2 ~ PA1, data = promax.pattern,
        aspect = 1,
        xlim = c(-1, 1.5),
        ylim = c(-0.4, 1),
        panel = function (x, y) {
          panel.segments(c(0, 0), c(0, 0),
                         c(1, 0), c(0, 1), col = "gray")
          panel.text(1, 0, labels = "Rotated\nfactor 1",
                     cex = .65, pos = 3, col = "gray")
          panel.text(0, .95, labels = "Rotated\nfactor 2",
                     cex = .65, pos = 4, col = "gray")
          panel.segments(rep(0, 8), rep(0, 8), x, y,
                         col = "black")
```

```
          panel.text(x[-7], y[-7], labels = rownames(promax.pattern)[-7],
                     pos = 4, cex = .75)
          panel.text(x[7], y[7], labels = rownames(promax.pattern)[7],
                     pos = 1, cex = .75)
        },
        main = "Promax Rotated Factor Pattern",
        xlab = "",
        ylab = "",
        scales = list(x = list(at = c(0, 1)),
                      y = list(at = c(-.4, 0, .6)))
)

# view side by side
grid.arrange(unrot_plot, rot_plot,
             ncol = 2)
```



The unrotated and the rotated patterns do not differ much, showing that the results are pretty robust–we can see from both graphs that autoc, physint, wecon and idealpoint are associated with one factor, and the rest variables are associated with another factor. The rotated version makes it a bit clearer to visually recognize the identified pattern.

Principal Components Analysis

1. What is the statistical difference between PCA and FA? Describe the basic construction of each approach using equations and then point to differences that exist across these two widely used methods for reducing dimensionality.

There are two main differences between PCA and FA. The first difference can be reflected from the basic construction of each approach. PCA reduces dimensionality through identifying principal components/factors as combinations of all features: Comp1 =L1X1 +L2X2 + · · · +LkXk. FA, however, identify factors/components as the cause of the observed features: X1 = b1F + d1U1. Moreover, FA assumes the latent variables to be Gaussian, yet PCA do not assume about the distribution of the latent factors, thus allowing for true statistical independence.

2. Fit a PCA model. Present the proportion of explained variance across the first 10 components. What do these values tell you substantively (e.g., how many components likely characterize these data?)?

```r
rownames(con_std) <- con_names
# fit the pca model
pca<- prcomp(con_std,
             center = TRUE)
# present summary that includes info about the proportion of explained variance
summary(pca)
```

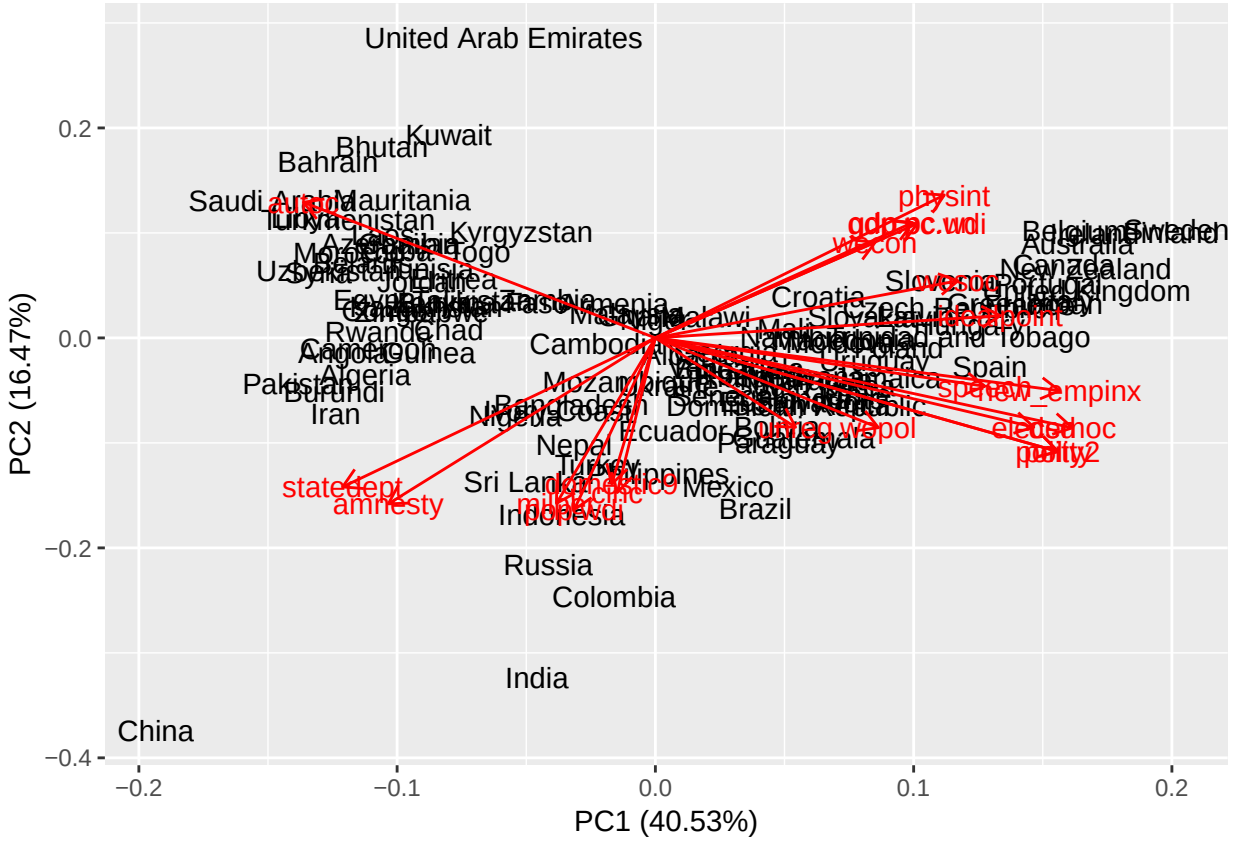
```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     2.9173 1.8600 1.6439 1.10713 1.07631 0.91289
## Proportion of Variance 0.4053 0.1648 0.1287 0.05837 0.05516 0.03968
## Cumulative Proportion  0.4053 0.5700 0.6987 0.75708 0.81225 0.85193
##                           PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     0.78181 0.72948 0.64421 0.58703 0.55164 0.49341
## Proportion of Variance 0.02911 0.02534 0.01976 0.01641 0.01449 0.01159
## Cumulative Proportion  0.88104 0.90638 0.92614 0.94255 0.95704 0.96864
##                           PC13   PC14    PC15    PC16    PC17    PC18
## Standard deviation     0.46337 0.3995 0.32765 0.29011 0.24347 0.18215
## Proportion of Variance 0.01022 0.0076 0.00511 0.00401 0.00282 0.00158
## Cumulative Proportion  0.97886 0.9865 0.99157 0.99558 0.99840 0.99998
##                           PC19      PC20      PC21
## Standard deviation     0.01990 7.605e-16 2.858e-16
## Proportion of Variance 0.00002 0.000e+00 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00 1.000e+00
```

Juding from the explained variance across the first 10 components, they accumulatively explain almost 95% of the variances, and the 10th component itself explains a little more than 1% of the variance. While it is very subjectively to determine how many components are sufficient to characterize the data, 10 components seem to be enough to capture the underlying dimensionalities. Within the 10 components, the gain in explained variances gradually becomes moderate and slow since the 6th component, so we can actually start from only looking at 5 components. To fully charaterzie data, 18 components would be enough as the cumulative explained variance is 1.

3. Present a biplot of the PCA fit from the previous question. Describe what you see (e.g., which countries are clustered together? Which input features are doing the bulk of the explaining? How do you know this?

```r
library(ggfortify)

autoplot(pca,
         shape = F, # labaels instead of points
         loadings.label = T)
```



We can see that on the left side, a group of west Asian and African countries cluster together. On the right side, a group of developed democracies (mainly western European countries with some from Oceania and East Asia) cluster on the edge, followed by a group of eastern European countries. In the middle and lower side of the figure, a group of Latin America countries, South Asia countries and Russia cluster together. Judging from the features and the distribution of countries, the first prinicpal component seems to capture the regime type (democratic vs authoritarian). The second principal components seem to capture the military strengths and stability level vs the economic devlopment level. The first principal component explains the bulk of variance (40.53%). From the graph we can also see that the divergence between the left and the right clusters is more pronounced than the devergence between the up and down sides.

Bonus Question (5 points): 1. Fit a sparse PCA model and a probabilistic PCA model. Compare these results substantively. What does each tell you and why do these distinctions matter in terms of inference (or not)?

```
library(sparsepca)

# fit sparse pca
pca_s <- spca(con_std,
              center = T)
```

```
## [1] "Iteration:    1, Objective: 6.79879e+00, Relative improvement Inf"
## [1] "Iteration:   11, Objective: 6.76709e+00, Relative improvement 3.84129e-04"
## [1] "Iteration:   21, Objective: 6.74437e+00, Relative improvement 3.04899e-04"
## [1] "Iteration:   31, Objective: 6.72565e+00, Relative improvement 2.61948e-04"
## [1] "Iteration:   41, Objective: 6.70896e+00, Relative improvement 2.38216e-04"
## [1] "Iteration:   51, Objective: 6.69373e+00, Relative improvement 2.21149e-04"
## [1] "Iteration:   61, Objective: 6.67954e+00, Relative improvement 2.06385e-04"
## [1] "Iteration:   71, Objective: 6.66627e+00, Relative improvement 1.94172e-04"
## [1] "Iteration:   81, Objective: 6.65358e+00, Relative improvement 1.85640e-04"
```

```
## [1] "Iteration:   91, Objective: 6.64148e+00, Relative improvement 1.79731e-04"
## [1] "Iteration:  101, Objective: 6.62968e+00, Relative improvement 1.76529e-04"
## [1] "Iteration:  111, Objective: 6.61809e+00, Relative improvement 1.72226e-04"
## [1] "Iteration:  121, Objective: 6.60686e+00, Relative improvement 1.68873e-04"
## [1] "Iteration:  131, Objective: 6.59577e+00, Relative improvement 1.67404e-04"
## [1] "Iteration:  141, Objective: 6.58482e+00, Relative improvement 1.64935e-04"
## [1] "Iteration:  151, Objective: 6.57406e+00, Relative improvement 1.62183e-04"
## [1] "Iteration:  161, Objective: 6.56350e+00, Relative improvement 1.60268e-04"
## [1] "Iteration:  171, Objective: 6.55301e+00, Relative improvement 1.59683e-04"
## [1] "Iteration:  181, Objective: 6.54257e+00, Relative improvement 1.59348e-04"
## [1] "Iteration:  191, Objective: 6.53221e+00, Relative improvement 1.56416e-04"
## [1] "Iteration:  201, Objective: 6.52207e+00, Relative improvement 1.54850e-04"
## [1] "Iteration:  211, Objective: 6.51197e+00, Relative improvement 1.54887e-04"
## [1] "Iteration:  221, Objective: 6.50194e+00, Relative improvement 1.53818e-04"
## [1] "Iteration:  231, Objective: 6.49199e+00, Relative improvement 1.52803e-04"
## [1] "Iteration:  241, Objective: 6.48212e+00, Relative improvement 1.50950e-04"
## [1] "Iteration:  251, Objective: 6.47234e+00, Relative improvement 1.51031e-04"
## [1] "Iteration:  261, Objective: 6.46257e+00, Relative improvement 1.51257e-04"
## [1] "Iteration:  271, Objective: 6.45282e+00, Relative improvement 1.50951e-04"
## [1] "Iteration:  281, Objective: 6.44317e+00, Relative improvement 1.49101e-04"
## [1] "Iteration:  291, Objective: 6.43359e+00, Relative improvement 1.47819e-04"
## [1] "Iteration:  301, Objective: 6.42413e+00, Relative improvement 1.46984e-04"
## [1] "Iteration:  311, Objective: 6.41470e+00, Relative improvement 1.47035e-04"
## [1] "Iteration:  321, Objective: 6.40526e+00, Relative improvement 1.47356e-04"
## [1] "Iteration:  331, Objective: 6.39588e+00, Relative improvement 1.46193e-04"
## [1] "Iteration:  341, Objective: 6.38661e+00, Relative improvement 1.44870e-04"
## [1] "Iteration:  351, Objective: 6.37736e+00, Relative improvement 1.44685e-04"
## [1] "Iteration:  361, Objective: 6.36813e+00, Relative improvement 1.45037e-04"
## [1] "Iteration:  371, Objective: 6.35888e+00, Relative improvement 1.45426e-04"
## [1] "Iteration:  381, Objective: 6.34969e+00, Relative improvement 1.44675e-04"
## [1] "Iteration:  391, Objective: 6.34050e+00, Relative improvement 1.45151e-04"
## [1] "Iteration:  401, Objective: 6.33128e+00, Relative improvement 1.45723e-04"
## [1] "Iteration:  411, Objective: 6.32208e+00, Relative improvement 1.45622e-04"
## [1] "Iteration:  421, Objective: 6.31286e+00, Relative improvement 1.46180e-04"
## [1] "Iteration:  431, Objective: 6.30362e+00, Relative improvement 1.46821e-04"
## [1] "Iteration:  441, Objective: 6.29435e+00, Relative improvement 1.47521e-04"
## [1] "Iteration:  451, Objective: 6.28509e+00, Relative improvement 1.45208e-04"
## [1] "Iteration:  461, Objective: 6.27604e+00, Relative improvement 1.44065e-04"
## [1] "Iteration:  471, Objective: 6.26700e+00, Relative improvement 1.44245e-04"
## [1] "Iteration:  481, Objective: 6.25796e+00, Relative improvement 1.44146e-04"
## [1] "Iteration:  491, Objective: 6.24893e+00, Relative improvement 1.44629e-04"
## [1] "Iteration:  501, Objective: 6.23989e+00, Relative improvement 1.44529e-04"
## [1] "Iteration:  511, Objective: 6.23088e+00, Relative improvement 1.44555e-04"
## [1] "Iteration:  521, Objective: 6.22187e+00, Relative improvement 1.45020e-04"
## [1] "Iteration:  531, Objective: 6.21283e+00, Relative improvement 1.45592e-04"
## [1] "Iteration:  541, Objective: 6.20379e+00, Relative improvement 1.45993e-04"
## [1] "Iteration:  551, Objective: 6.19472e+00, Relative improvement 1.46638e-04"
## [1] "Iteration:  561, Objective: 6.18562e+00, Relative improvement 1.47329e-04"
## [1] "Iteration:  571, Objective: 6.17652e+00, Relative improvement 1.46062e-04"
## [1] "Iteration:  581, Objective: 6.16752e+00, Relative improvement 1.45925e-04"
## [1] "Iteration:  591, Objective: 6.15851e+00, Relative improvement 1.46383e-04"
## [1] "Iteration:  601, Objective: 6.14948e+00, Relative improvement 1.46939e-04"
## [1] "Iteration:  611, Objective: 6.14047e+00, Relative improvement 1.45330e-04"
## [1] "Iteration:  621, Objective: 6.13161e+00, Relative improvement 1.43862e-04"
```

```
## [1] "Iteration:  631, Objective: 6.12279e+00, Relative improvement 1.44137e-04"
## [1] "Iteration:  641, Objective: 6.11395e+00, Relative improvement 1.44569e-04"
## [1] "Iteration:  651, Objective: 6.10510e+00, Relative improvement 1.45006e-04"
## [1] "Iteration:  661, Objective: 6.09624e+00, Relative improvement 1.45519e-04"
## [1] "Iteration:  671, Objective: 6.08736e+00, Relative improvement 1.46135e-04"
## [1] "Iteration:  681, Objective: 6.07845e+00, Relative improvement 1.46807e-04"
## [1] "Iteration:  691, Objective: 6.06951e+00, Relative improvement 1.47483e-04"
## [1] "Iteration:  701, Objective: 6.06054e+00, Relative improvement 1.48136e-04"
## [1] "Iteration:  711, Objective: 6.05155e+00, Relative improvement 1.48904e-04"
## [1] "Iteration:  721, Objective: 6.04252e+00, Relative improvement 1.49703e-04"
## [1] "Iteration:  731, Objective: 6.03345e+00, Relative improvement 1.50527e-04"
## [1] "Iteration:  741, Objective: 6.02439e+00, Relative improvement 1.50070e-04"
## [1] "Iteration:  751, Objective: 6.01534e+00, Relative improvement 1.50748e-04"
## [1] "Iteration:  761, Objective: 6.00625e+00, Relative improvement 1.51527e-04"
## [1] "Iteration:  771, Objective: 5.99713e+00, Relative improvement 1.52355e-04"
## [1] "Iteration:  781, Objective: 5.98800e+00, Relative improvement 1.52553e-04"
## [1] "Iteration:  791, Objective: 5.97886e+00, Relative improvement 1.52746e-04"
## [1] "Iteration:  801, Objective: 5.96973e+00, Relative improvement 1.52961e-04"
## [1] "Iteration:  811, Objective: 5.96059e+00, Relative improvement 1.53610e-04"
## [1] "Iteration:  821, Objective: 5.95145e+00, Relative improvement 1.53511e-04"
## [1] "Iteration:  831, Objective: 5.94230e+00, Relative improvement 1.54167e-04"
## [1] "Iteration:  841, Objective: 5.93312e+00, Relative improvement 1.54594e-04"
## [1] "Iteration:  851, Objective: 5.92394e+00, Relative improvement 1.54874e-04"
## [1] "Iteration:  861, Objective: 5.91481e+00, Relative improvement 1.53921e-04"
## [1] "Iteration:  871, Objective: 5.90569e+00, Relative improvement 1.54542e-04"
## [1] "Iteration:  881, Objective: 5.89655e+00, Relative improvement 1.55268e-04"
## [1] "Iteration:  891, Objective: 5.88754e+00, Relative improvement 1.52737e-04"
## [1] "Iteration:  901, Objective: 5.87853e+00, Relative improvement 1.53470e-04"
## [1] "Iteration:  911, Objective: 5.86949e+00, Relative improvement 1.54230e-04"
## [1] "Iteration:  921, Objective: 5.86043e+00, Relative improvement 1.54769e-04"
## [1] "Iteration:  931, Objective: 5.85138e+00, Relative improvement 1.54203e-04"
## [1] "Iteration:  941, Objective: 5.84235e+00, Relative improvement 1.54816e-04"
## [1] "Iteration:  951, Objective: 5.83330e+00, Relative improvement 1.54691e-04"
## [1] "Iteration:  961, Objective: 5.82426e+00, Relative improvement 1.55355e-04"
## [1] "Iteration:  971, Objective: 5.81526e+00, Relative improvement 1.54329e-04"
## [1] "Iteration:  981, Objective: 5.80628e+00, Relative improvement 1.54831e-04"
## [1] "Iteration:  991, Objective: 5.79732e+00, Relative improvement 1.54421e-04"
```

```
summary(pca_s)
```

```
##                           PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8
## Explained variance      8.507 3.456 2.699 1.222 1.155 0.830 0.608 0.529
## Standard deviations     2.917 1.859 1.643 1.106 1.075 0.911 0.780 0.727
## Proportion of variance 0.405 0.165 0.129 0.058 0.055 0.040 0.029 0.025
## Cumulative proportion  0.405 0.570 0.698 0.756 0.811 0.851 0.880 0.905
##                           PC9  PC10  PC11  PC12  PC13  PC14  PC15  PC16
## Explained variance      0.412 0.341 0.301 0.240 0.211 0.156 0.104 0.081
## Standard deviations     0.642 0.584 0.549 0.490 0.460 0.395 0.323 0.284
## Proportion of variance 0.020 0.016 0.014 0.011 0.010 0.007 0.005 0.004
## Cumulative proportion  0.925 0.941 0.955 0.967 0.977 0.984 0.989 0.993
##                          PC17  PC18  PC19  PC20  PC21
## Explained variance      0.056 0.030 0.000 0.000 0.000
## Standard deviations     0.237 0.174 0.018 0.000 0.000
## Proportion of variance 0.003 0.001 0.000 0.000 0.000
## Cumulative proportion  0.996 0.997 0.997 0.997 0.997
```

Judging from the explained variance, sparse PCA does not differ much from PCA. In theory, sparse PCA serves to address the limitations of PCA in terms of interpretability. While for PCA each component is a linear combination of all features, sparse PCA imposes the lasso (elastic net) constraint on the regression coefficients to produce sparse loadings.