Assignment 2

Henrik Olaussen

2023-08-29

a)

(i)

In a mixture of factors analyzers-model, the distribution of the i'th observation Y_i is modeled by:

$$Y_j = \mu_i + B_i U_{ij} + e_{ij}, \quad i = 1, 2, ..., g, \quad j = 1, 2, ..., n$$

The final model has the form:

$$f(\mathbf{y}, \Psi) = \sum_{i=1}^{g} \pi_i \phi(\mathbf{y}; \mu_i, \Sigma_i)$$

where:

$$\Sigma_i = B_i B_i^T + D_i$$

In our case, we are going to fit models with g = 3 components and q = 1,2,3,4,5 and 6 factors. In the output from the 6 different mfa models, mu_i holds the values of the i'th component mean. The pi_i-values are the mixture proportions π_i, and B holds the factor loading contained in the matrix $B \in \mathbb{R}^{p \times q}$ from the model above. Lastly, "diag D" holds the values on the diagonal matrix D, where D_i give the variance of the different e_{ij} $(i = 1, 2, .., g)$, respectively. $U_{ij} \in \mathbb{R}^q$ holds the factors.

(ii)

On the other hand, we have the MCFA model. The MCFA approach provides a greater reduction in the number of parameters compared to MFA. In this case, we fit the usual model $f(\mathbf{y}, \Psi)$, but with the following restrictions:

$$\mu_i = A\xi_i$$

and

$$\Sigma_i = A\Omega_i A^T + D$$

where $A \in \mathbb{R}^{p \times q}$, $\xi \in \mathbb{R}^q$ (xi_i in the mcfa output), $\Omega_i \in \mathbb{R}^{q \times q}$ and $D \in \mathbb{R}^{p \times p}$. Ω_i is a positive definite symmetric matrix, and D a diagonal matrix. A holds the loadings on the q unobservable factors. Moreover, the i'th component distribution of Y_j is modeled by:

$$Y_j = AU_{ij} + e_{ij}$$

with probability pi_i for $i = 1, 2, .., g$, $j = 1, ..., n$. The factors $U_{i1}, .., U_{in}$ are distributed independently $N(\xi_i, \Omega_i)$, independently of e_{ij} which again are distributed independently $N(0, D)$.

b)

Before we can fit the models, we have to load the data:

```
wine <- read.table('wine.data',sep=',')
wine_2 <- subset(wine, select = -c(1)) #exclude labels
```

The dataset has n = 178 observations.

(i) Furthermore, we can now fit the 12 different models. This is done through a for-loop below:

```
mfa.models = list()
mcfa.models = list()

for (q in 1:6) {
  mfa.models[[q]] = mfa(wine_2, g = 3, q)
  mcfa.models[[q]] = mcfa(wine_2, g = 3, q, itmax = 200, init_method = "eigen-A")
}
```

Firstly, the output from the different mfa models can be seen below. What each output parameter mean is briefly explained in a(i) and (ii).

q = 1:

```
mfa.models[[1]]
```

```
## Call:
## mfa(Y = wine_2, g = 3, q = q)
##
## Coefficients:
## pi_i :  0.276 0.312 0.412
##
## mu_1:
##  [1]  13.1491351   3.3358144   2.4327910  21.4165983  99.0185542   1.6794202
##  [7]   0.7999989   0.4510685   1.1604736   7.3354120   0.6871079   1.6900653
## [13] 626.9228569
## mu_2:
##  [1]   13.7843545    1.9350973    2.4333359   16.8222804  106.0994906
##  [6]    2.8540376    3.0012231    0.2860384    1.8989611    5.6079262
## [11]    1.0665040    3.1504739 1137.9627307
## mu_3:
##  [1]  12.307456   1.969405   2.271438  20.229102  95.412293   2.285087
##  [7]   2.118281   0.359389   1.646498   3.113305   1.056307   2.822232
## [13] 531.273884
##
##  B:
##              [,1]
## V2   -0.08089372
## V3   -0.04358400
## V4   -0.07405956
## V5   -0.57326748
## V6   -2.21853426
## V7   -0.33817489
```

```
## V8   -0.46800457
## V9    0.02432866
## V10  -0.27778736
## V11  -0.60248989
## V12   0.01342511
## V13  -0.15360296
## V14 -16.97715494
## diag D:
##  [1] 2.533278e-01 8.573730e-01 6.303116e-02 7.291591e+00 1.774563e+02
##  [6] 7.296444e-02 5.772352e-02 1.081541e-02 1.665606e-01 1.896367e+00
## [11] 2.383959e-02 1.341477e-01 2.750866e+04
```

q = 2:

```
mfa.models[[2]]
```

```
## Call:
## mfa(Y = wine_2, g = 3, q = q)
##
## Coefficients:
## pi_i :  0.33 0.394 0.276
##
## mu_1:
##  [1]   13.7498530    1.9652768    2.4575312   17.0254777  106.4623528
##  [6]    2.8379847    2.9620832    0.2936622    1.8797204    5.5123826
## [11]    1.0657768    3.1389696 1117.6002675
## mu_2:
##  [1]  12.2702538   1.9471351   2.2438681  20.2146361  94.6128976   2.2725206
##  [7]   2.1107820   0.3566524   1.6510254   3.0826744   1.0563618   2.8163831
## [13] 520.8511657
## mu_3:
##  [1]  13.1492489   3.3354397   2.4330359  21.4157330  99.0423322   1.6791255
##  [7]   0.7989322   0.4507138   1.1601194   7.3374547   0.6867980   1.6896335
## [13] 627.1191681
##
##  B:
##             [,1]         [,2]
## V2   0.10958194 -0.043266297
## V3  -0.01598599  0.097183838
## V4  -0.02680262  0.248612591
## V5  -0.20450317  1.897581582
## V6   1.08988288  2.753371694
## V7   0.32329411  0.099191217
## V8   0.44448739  0.167864225
## V9  -0.03901990  0.021611634
## V10  0.28710753  0.046954185
## V11  0.63986461  0.112710795
## V12 -0.01598577  0.005425077
## V13  0.14012120  0.064004213
## V14 25.47872185 -0.544854666
## diag D:
##  [1] 2.399939e-01 8.506307e-01 2.384615e-03 4.211792e+00 1.686734e+02
##  [6] 7.303960e-02 5.905540e-02 9.688298e-03 1.609702e-01 1.881148e+00
## [11] 2.371949e-02 1.346658e-01 2.855282e+04
```

q = 3:

```
mfa.models[[3]]
```

```
## Call:
## mfa(Y = wine_2, g = 3, q = q)
##
## Coefficients:
## pi_i :  0.332 0.271 0.396
##
## mu_1:
##  [1]   13.7401473    1.9855817    2.4592279   17.2145028  106.8335227
##  [6]    2.8484449    2.9842589    0.2885035    1.9025514    5.5093196
## [11]    1.0682862    3.1636878 1118.6083052
## mu_2:
##  [1]  13.1550386   3.3850285   2.4340567  21.4721004  98.4393464   1.6813022
##  [7]   0.7915674   0.4554784   1.1651409   7.3840317   0.6844869   1.7009855
## [13] 625.8605512
## mu_3:
##  [1]  12.2750232   1.9120717   2.2425524  20.0520650  94.6894636   2.2517571
##  [7]   2.0765899   0.3592059   1.6213032   3.0868002   1.0515118   2.7727923
## [13] 518.2428647
##
##  B:
##               [,1]          [,2]        [,3]
## V2   -0.16320957  -0.031044342  0.07881484
## V3    0.19107671  -0.007175472 -0.22851351
## V4    0.14454907  -0.202279550  0.03686336
## V5    1.32594188  -1.389377542 -0.21726995
## V6   -0.22498645  -2.985870306  2.55413788
## V7   -0.17122651  -0.224317757 -0.17769235
## V8   -0.21569406  -0.331404752 -0.24366342
## V9    0.03151433  -0.006158487  0.02428965
## V10  -0.18972163  -0.167103932 -0.13211267
## V11  -0.88047379  -0.597386277  0.45668161
## V12   0.01748441   0.005881609  0.00569385
## V13   0.02147367  -0.069490117 -0.23921071
## V14 -47.76179858 -17.472208797 56.87730816
## diag D:
##  [1] 2.247243e-01 7.414614e-01 1.432529e-03 4.439644e+00 1.600632e+02
##  [6] 7.354249e-02 5.667200e-02 9.610304e-03 1.625424e-01 9.271402e-01
## [11] 2.376455e-02 1.020871e-01 2.219069e+04
```

q = 4:

```
mfa.models[[4]]
```

```
## Call:
## mfa(Y = wine_2, g = 3, q = q)
##
## Coefficients:
## pi_i :  0.394 0.33 0.276
##
```

```
## mu_1:
##  [1]  12.2874051   1.9338169   2.2377734  20.2046692  93.9835683   2.2710285
##  [7]   2.1007091   0.3665044   1.6328638   3.1003107   1.0572392   2.7958979
## [13] 518.2887771
## mu_2:
##  [1]   13.746427    2.005396    2.456310   17.045218  106.378556    2.840604
##  [7]    2.981778    0.290011    1.898775    5.528682    1.063081    3.157100
## [13] 1116.878029
## mu_3:
##  [1]  13.1250189   3.3068716   2.4426578  21.4158114 100.0085151   1.6762899
##  [7]   0.7868103   0.4412454   1.1623786   7.2874920   0.6886082   1.6958076
## [13] 629.9870685
##
##  B:
##              [,1]        [,2]         [,3]         [,4]
## V2   -0.03639042 -0.14228843  -0.12075438  -0.04460325
## V3    0.25518333 -0.18442293   0.19654213   0.37984197
## V4    0.01001252 -0.06085414   0.22101481  -0.10040740
## V5    0.25133635 -0.63859916   1.78345775  -0.35887217
## V6   -3.17609909  0.69261574   1.86036769  -2.75307117
## V7   -0.28834433 -0.13061107   0.07989169   0.05065707
## V8   -0.38565679 -0.19835111   0.14258253   0.06966626
## V9    0.04672446 -0.01254441   0.01272784  -0.02932361
## V10  -0.24231782 -0.13285846   0.01753969   0.05764091
## V11  -0.39893090 -1.20484622  -0.55525344  -0.56868573
## V12  -0.02731094  0.06916791   0.00985351  -0.05227659
## V13  -0.10672695 -0.01678159   0.12188256   0.13679059
## V14 -39.89250813 -0.16547150 -30.32244612 -61.84719765
## diag D:
##  [1] 2.289020e-01 5.990651e-01 1.265269e-03 4.107118e+00 1.536048e+02
##  [6] 7.611116e-02 5.107328e-02 8.585220e-03 1.630622e-01 1.470396e-01
## [11] 1.604027e-02 1.128935e-01 2.269928e+04
```

q = 5:

```r
mfa.models[[5]]
```

```
## Call:
## mfa(Y = wine_2, g = 3, q = q)
##
## Coefficients:
## pi_i :  0.369 0.329 0.302
##
## mu_1:
##  [1]  12.2450109   1.9463634   2.2641404  20.5799356  94.6580495   2.2859157
##  [7]   2.1639333   0.3515999   1.7146329   3.0445955   1.0620643   2.8695315
## [13] 518.5551518
## mu_2:
##  [1]    13.7491520    2.0039338    2.4553102   17.0220794  106.3123173
##  [6]     2.8408187    2.9841416    0.2895807    1.8998424    5.5366455
## [11]     1.0631304    3.1580343 1118.3217413
## mu_3:
##  [1]  13.1088189   3.1750755   2.3949281  20.8623501  98.7977640   1.7119534
```

```
## [7]   0.8246050   0.4531089   1.1031570   6.9977877   0.7144751   1.7014387
## [13] 621.4042896
##
##  B:
##             [,1]          [,2]          [,3]        [,4]        [,5]
## V2   0.06723991  -0.153112875   0.016862332  0.01386645 -0.07130186
## V3  -0.21172403   0.059268751   0.103985599 -0.46041198 -0.14999618
## V4  -0.12798891  -0.072353738  -0.186716247 -0.08555570  0.03004808
## V5  -1.22451786  -0.579523085  -1.194839296 -0.98364293 -0.02562357
## V6   1.87451024  -5.245556474   0.138652152 -3.56953002 10.68249804
## V7   0.30255735  -0.054274304  -0.204471901 -0.10961231 -0.06338747
## V8   0.28740682  -0.110587496  -0.209995667 -0.12304898 -0.08310620
## V9  -0.03660765   0.006094174  -0.025203278  0.01556717 -0.01180103
## V10  0.16958952  -0.173482039  -0.013589941 -0.10651108 -0.02531396
## V11  0.21327799  -1.463787965   0.005757728  0.05747893 -0.51993053
## V12  0.01734899   0.036084506  -0.048435920  0.08628857  0.05111227
## V13  0.08632959   0.068187527  -0.054350452 -0.11271488 -0.04607022
## V14 22.57040285 -56.582846511 -10.362871338 37.00835419 27.34573585
## diag D:
##  [1] 2.232571e-01 6.424493e-01 3.408148e-03 3.846742e+00 2.085426e+01
##  [6] 3.647627e-02 8.628230e-02 8.735123e-03 1.458263e-01 1.735471e-01
## [11] 1.240243e-02 9.843997e-02 2.329450e+04
```

q = 6:

```
mfa.models[[6]]
```

```
## Call:
## mfa(Y = wine_2, g = 3, q = q)
##
## Coefficients:
## pi_i :  0.395 0.329 0.276
##
## mu_1:
##  [1]  12.2893289   1.9348061   2.2385160  20.2095958  94.0663582   2.2715057
##  [7]   2.1014817   0.3663386   1.6338999   3.1031097   1.0569721   2.7961236
## [13] 519.4722522
## mu_2:
##  [1]   13.7484821    2.0044392    2.4562683   17.0321426  106.3365050
##  [6]    2.8403164    2.9822689    0.2899429    1.8984387    5.5325558
## [11]    1.0632573    3.1568480 1117.7095790
## mu_3:
##  [1]  13.1263465   3.3088790   2.4427400  21.4146653  99.9940615   1.6771471
##  [7]   0.7863877   0.4413962   1.1615697   7.2938481   0.6882592   1.6954804
## [13] 629.6388342
##
##  B:
##              [,1]          [,2]         [,3]         [,4]          [,5]
## V2   -0.03163291  -0.048736425  0.007931012  0.148957816   0.083293375
## V3    0.17464195  -0.020431872 -0.225892482 -0.289492880   0.434824755
## V4    0.11606256  -0.191033133  0.020329072 -0.081069734  -0.061399561
## V5    0.99824978  -1.435025783 -0.083697935 -0.737681245  -0.128863569
## V6   -4.70108095  -4.379946520  4.984378775 -2.646709407   0.062469477
```

```
## V7   -0.16586158  -0.223264205 -0.197025860  0.029487931  -0.044516826
## V8   -0.19514171  -0.323720834 -0.239262210  0.058011812  -0.044686305
## V9    0.06894901  -0.001158169  0.005878341 -0.018258009   0.003793035
## V10  -0.20747396  -0.195495855 -0.094919003  0.002255569   0.081254462
## V11  -0.08277682  -0.758786758  0.127346459  1.171574343   0.418619354
## V12  -0.01505683   0.008827763  0.031037087 -0.018325210  -0.081020613
## V13  -0.06134587  -0.074191235 -0.160551407 -0.065979635  -0.013354582
## V14 -42.62955883 -28.263740926 52.296789065 23.969411348 -14.304887095
##             [,6]
## V2   0.06941776
## V3  -0.07364245
## V4  -0.03428647
## V5  -0.46625515
## V6  -0.65971050
## V7   0.06127513
## V8  -0.01332749
## V9   0.07560626
## V10  0.06481352
## V11  0.38968304
## V12  0.02154330
## V13 -0.09880436
## V14 36.85982092
## diag D:
##  [1] 2.294814e-01 5.201394e-01 2.174699e-03 4.031241e+00 1.021906e+02
##  [6] 6.332814e-02 5.952969e-02 1.083898e-03 1.419790e-01 8.267495e-02
## [11] 1.575141e-02 1.092685e-01 2.160316e+04
```

Secondly,the mcfa outputs are shown below:

q = 1:

```
mcfa.models[[1]]
```

```
## Call:
## mcfa(Y = wine_2, g = 3, q = q, itmax = 200, init_method = "eigen-A")
##
## Coefficients:
## pi_i :  0.58 0 0.42
##
## A:
##               [,1]
## V2  -0.0170756560
## V3  -0.0030621958
## V4  -0.0031088535
## V5  -0.0255218327
## V6  -0.1310851135
## V7  -0.0030251395
## V8  -0.0026838537
## V9  -0.0004728139
## V10 -0.0020952281
## V11 -0.0066852564
## V12 -0.0012562480
## V13 -0.0034338002
## V14 -0.9908459412
```

```
##
## xi_1:
## [1] -730.5898
## xi_2:
## [1] -726.365
## xi_3:
## [1] -803.6443
##
## omega_1:
## [1] 39.16348
## omega_2:
## [1] 23.56357
## omega_3:
## [1] 32.45907
##
## diag D:
##  0.3123729 1.253002 0.0614799 12.80715 170.5391 0.3415258 0.926245 0.01649224 0.306854 4.969421 0.05
```

q = 2:

```
mcfa.models[[2]]
```

```
## Call:
## mcfa(Y = wine_2, g = 3, q = q, itmax = 200, init_method = "eigen-A")
##
## Coefficients:
## pi_i :  0.443 0.204 0.353
##
## A:
##               [,1]         [,2]
## V2  -0.0150833787 -0.129267921
## V3  -0.0025743488 -0.031757353
## V4  -0.0027307561 -0.024498618
## V5  -0.0216188762 -0.253768428
## V6  -0.1164333925 -0.948943313
## V7  -0.0027600483 -0.016970462
## V8  -0.0025369069 -0.009169059
## V9  -0.0003961987 -0.005008046
## V10 -0.0019011841 -0.012444830
## V11 -0.0061044090 -0.037437295
## V12 -0.0011175756 -0.008993476
## V13 -0.0030817613 -0.022667040
## V14 -0.9928083709  0.119335729
##
## xi_1:
## [1] -626.18564  -26.04411
## xi_2:
## [1] -430.93783  -44.69873
## xi_3:
## [1] -1100.93629    22.82782
##
## omega_1:
##            [,1]       [,2]
```

```
## [1,] 14992.352 -1666.6446
## [2,] -1666.645   195.3603
## omega_2:
##           [,1]       [,2]
## [1,] 5652.8767 -648.07853
## [2,] -648.0785   75.04654
## omega_3:
##           [,1]       [,2]
## [1,] 53362.325 -5793.7043
## [2,] -5793.704   629.4721
##
## diag D:
##  0.2863384 1.161909 0.06322361 8.444685 166.6597 0.2926374 0.7513917 0.01378526 0.2910521 4.695942 0
```

q = 3:

```
mcfa.models[[3]]
```

```
## Call:
## mcfa(Y = wine_2, g = 3, q = q, itmax = 200, init_method = "eigen-A")
##
## Coefficients:
## pi_i :  0.049 0.454 0.498
##
## A:
##              [,1]         [,2]         [,3]
## V2  -0.0149256742 -0.113071038 -0.452722776
## V3  -0.0025352117 -0.027686982 -0.116276451
## V4  -0.0027019314 -0.021922948 -0.073259891
## V5  -0.0213147627 -0.224852208 -0.825467748
## V6  -0.1155762348 -0.958816841  0.259173955
## V7  -0.0027395865 -0.014970940 -0.056488650
## V8  -0.0025258495 -0.008094822 -0.030409143
## V9  -0.0003896176 -0.004197959 -0.023236295
## V10 -0.0018879163 -0.011585349 -0.020318514
## V11 -0.0060598681 -0.033509656 -0.118598760
## V12 -0.0011066513 -0.007866365 -0.030842891
## V13 -0.0030534927 -0.019541999 -0.086524957
## V14 -0.9929181434  0.118622678 -0.003840975
##
## xi_1:
## [1] -630.429779  -52.343863    7.264324
## xi_2:
## [1] -531.406332  -30.077114   -2.864941
## xi_3:
## [1] -968.6009938    5.7874590    0.8227325
##
## omega_1:
##             [,1]      [,2]       [,3]
## [1,] 27741.76863  34.29913 -897.22494
## [2,]    34.29913  58.24058  -16.06157
## [3,]  -897.22494 -16.06157   32.88236
## omega_2:
```

```
##              [,1]         [,2]       [,3]
## [1,] 14781.59370 -1583.827221 -35.223982
## [2,] -1583.82722   212.519828  -5.341948
## [3,]   -35.22398    -5.341948   2.654799
## omega_3:
##               [,1]         [,2]       [,3]
## [1,]  87225.53713 -10712.20253  76.988888
## [2,] -10712.20253   1422.28121 -32.303391
## [3,]     76.98889    -32.30339   4.981034
##
## diag D:
##  0.2982512 1.154812 0.05941899 8.082475 0.9630397 0.2919746 0.7511902 0.0130438 0.2887757 4.66601 0.0
```

q = 4:

```r
mcfa.models[[4]]
```

```
## Call:
## mcfa(Y = wine_2, g = 3, q = q, itmax = 200, init_method = "eigen-A")
##
## Coefficients:
## pi_i :  0.313 0.345 0.341
##
## A:
##               [,1]         [,2]         [,3]          [,4]
## V2  -0.0149486216 -0.114813870 -0.455153372 -0.1397854904
## V3  -0.0024886038 -0.027119107 -0.139902455  0.2115417238
## V4  -0.0027063195 -0.022211549 -0.072194842 -0.0249448094
## V5  -0.0213184806 -0.227125056 -0.812295194 -0.0578728388
## V6  -0.1155932197 -0.958070968  0.260555284  0.0247403791
## V7  -0.0027956428 -0.016242067 -0.039303142 -0.2708192645
## V8  -0.0026270823 -0.010161908  0.006540795 -0.4823996895
## V9  -0.0003841966 -0.004156539 -0.025112543  0.0243133682
## V10 -0.0019268631 -0.012403151 -0.007699312 -0.1862427823
## V11 -0.0059176917 -0.031111004 -0.184379463  0.6624061117
## V12 -0.0011260998 -0.008329380 -0.022872260 -0.0937080028
## V13 -0.0031316990 -0.021319595 -0.059373439 -0.3771383828
## V14 -0.9929159476  0.118630615 -0.004062872 -0.0002565771
##
## xi_1:
## [1] -651.4264386  -29.2452684   -0.6058739    2.3848142
## xi_2:
## [1] -1111.2134215    22.9562539     0.8543347    -0.5415876
## xi_3:
## [1] -486.255333  -35.309732   -1.784138   -1.150106
##
## omega_1:
##              [,1]         [,2]        [,3]        [,4]
## [1,] 14798.99408 -1126.464895 -202.438396 -26.6208637
## [2,] -1126.46490   291.087997  -46.284779   5.8135364
## [3,]  -202.43840   -46.284779   22.000461  -0.8523840
## [4,]   -26.62086     5.813536   -0.852384   0.1901671
## omega_2:
```

```
##               [,1]         [,2]        [,3]        [,4]
## [1,] 48559.02929 -6147.566856  99.1780422 -76.2296720
## [2,] -6147.56686   896.282761 -37.8980152   8.0998127
## [3,]    99.17804   -37.898015   5.6666963   0.2034422
## [4,]   -76.22967     8.099813   0.2034422   0.2460517
## omega_3:
##                [,1]        [,2]        [,3]        [,4]
## [1,] 12847.560246 -1596.38604  -3.1660631 -37.0680728
## [2,] -1596.386039   267.82763 -17.5832862   5.2072803
## [3,]    -3.166063   -17.58329   4.9026526  -0.4577342
## [4,]   -37.068073     5.20728  -0.4577342   1.2336556
##
## diag D:
##  0.2315644 0.9346633 0.06016731 8.334818 0.4637933 0.1096614 0.09956586 0.009864498 0.1961258 3.0991
```

q = 5:

```
mcfa.models[[5]]
```

```
## Call:
## mcfa(Y = wine_2, g = 3, q = q, itmax = 200, init_method = "eigen-A")
##
## Coefficients:
## pi_i :  0.346 0.341 0.313
##
## A:
##                [,1]         [,2]         [,3]         [,4]         [,5]
## V2  -0.0150416873 -0.117381347 -0.276547584  0.210434858 -0.851028506
## V3  -0.0025357365 -0.027898429 -0.107579875  0.248709876 -0.058155855
## V4  -0.0027149645 -0.022410946 -0.066103578 -0.008393714 -0.044584320
## V5  -0.0213893663 -0.227894637 -0.906945778 -0.178394220  0.296842341
## V6  -0.1155943145 -0.957498123  0.259339015  0.001998952  0.050137418
## V7  -0.0027794547 -0.016287852 -0.020358659 -0.206269782 -0.197085151
## V8  -0.0025815337 -0.009841861  0.010012058 -0.425641173 -0.228690973
## V9  -0.0003911088 -0.004277796 -0.019575839  0.031529996 -0.013626418
## V10 -0.0019108968 -0.012302171 -0.005401133 -0.164047814 -0.090751685
## V11 -0.0060208470 -0.032577922 -0.124194379  0.719669591  0.010802845
## V12 -0.0011225535 -0.008393442 -0.012985702 -0.066977262 -0.081238293
## V13 -0.0031099879 -0.021404780 -0.035001975 -0.296522256 -0.271038076
## V14 -0.9929123925  0.118662090 -0.005083238 -0.001561951  0.003132737
##
## xi_1:
## [1] -1111.0124522    23.0858809     1.3310497    -0.5691690    -0.2180626
## xi_2:
## [1] -486.0243831  -35.4177762   -2.0780970   -1.1324425   -0.5353119
## xi_3:
## [1] -651.4387370  -29.2489660   -0.4512819    2.1240813    0.9039361
##
## omega_1:
##               [,1]         [,2]        [,3]        [,4]        [,5]
## [1,] 48463.09821 -6094.438793 255.5462794 -52.6848913 -59.2287308
## [2,] -6094.43879   887.444440 -51.1527051   8.7506429  -0.9023760
## [3,]   255.54628   -51.152705  10.4981556   0.4305326  -1.4810360
```

```
## [4,]   -52.68489     8.750643   0.4305326   0.3108514  -0.4704613
## [5,]   -59.22873    -0.902376  -1.4810360  -0.4704613   1.6931336
## omega_2:
##             [,1]          [,2]       [,3]        [,4]        [,5]
## [1,] 12841.59006 -1582.3366707  62.990813 -18.3925171 -38.1832673
## [2,] -1582.33667   265.9471164 -21.720240   5.8394132  -0.2749253
## [3,]    62.99081   -21.7202399  11.919301   1.1216382  -1.6243057
## [4,]   -18.39252     5.8394132   1.121638   1.8059033  -0.5490031
## [5,]   -38.18327    -0.2749253  -1.624306  -0.5490031   1.2321662
## omega_3:
##             [,1]         [,2]        [,3]       [,4]        [,5]
## [1,] 14832.39656 -1123.527570 -153.176411 14.2148386 -68.0030043
## [2,] -1123.52757   289.543594  -49.505285  8.2601499  -5.0793379
## [3,]  -153.17641   -49.505285   25.925903 -2.4303586   2.3909202
## [4,]    14.21484     8.260150   -2.430359  0.4940994  -0.6666424
## [5,]   -68.00300    -5.079338    2.390920 -0.6666424   1.2117682
##
## diag D:
##  0.1871194 0.9290838 0.03801293 0.3225576 0.2788604 0.1159033 0.1047111 0.00960666 0.1956475 3.070190
```

q = 6:

```
mcfa.models[[6]]
```

```
## Call:
## mcfa(Y = wine_2, g = 3, q = q, itmax = 200, init_method = "eigen-A")
##
## Coefficients:
## pi_i :  0.319 0.335 0.346
##
## A:
##              [,1]         [,2]         [,3]         [,4]         [,5]
## V2  -0.0149936960 -0.115282352 -0.270371766  0.060350300 -0.821319240
## V3  -0.0025427386 -0.028036847 -0.106822333  0.170758977 -0.031528355
## V4  -0.0027077024 -0.022197913 -0.065831724 -0.005225554 -0.044970046
## V5  -0.0213723266 -0.227687245 -0.908794260 -0.156837901  0.287500125
## V6  -0.1155867203 -0.957877692  0.258122135  0.003024961  0.049124344
## V7  -0.0027395694 -0.015207638 -0.019399586 -0.093947835 -0.224858597
## V8  -0.0025179517 -0.008217746  0.011081272 -0.209127740 -0.289477643
## V9  -0.0003932529 -0.004302511 -0.019402734  0.010722569 -0.009726147
## V10 -0.0018827156 -0.011615941 -0.005393889 -0.060266780 -0.119176141
## V11 -0.0059844221 -0.032573974 -0.128134097  0.912745248 -0.061502845
## V12 -0.0011170440 -0.008119563 -0.012292511 -0.074966340 -0.078743420
## V13 -0.0030725608 -0.020153749 -0.033479953 -0.218010293 -0.280528849
## V14 -0.9929150414  0.118636431 -0.005017809 -0.002153294  0.003610524
##             [,6]
## V2   0.411597444
## V3   0.228619203
## V4  -0.018226575
## V5  -0.094800529
## V6  -0.003549118
## V7  -0.328755996
## V8  -0.651808861
```

```
## V9   0.052062879
## V10 -0.282631385
## V11 -0.335812206
## V12  0.013193769
## V13 -0.201287405
## V14  0.001409879
##
## xi_1:
## [1] -648.7032044  -29.4301580   -0.5461834    2.2819012    0.7066895
## [6]    0.4788753
## xi_2:
## [1] -486.0969475  -35.3057774   -2.0969895   -1.5839899   -0.6475506
## [6]   -0.2410702
## xi_3:
## [1] -1110.4780186    22.9864397     1.3097362    -0.4263324    -0.2344936
## [6]    -0.2818050
##
## omega_1:
##             [,1]         [,2]        [,3]      [,4]         [,5]         [,6]
## [1,] 14980.29038 -1137.134182 -151.626502 53.751530 -79.00053231 -22.08928731
## [2,] -1137.13418   288.914778  -49.125929  6.668137  -3.84534400   5.10967066
## [3,]  -151.62650   -49.125929   26.569693 -5.937204   2.61773392   0.41448503
## [4,]    53.75153     6.668137   -5.937204  5.130969  -1.18844992  -1.30085974
## [5,]   -79.00053    -3.845344    2.617734 -1.188450   1.31323395  -0.03798219
## [6,]   -22.08929     5.109671    0.414485 -1.300860  -0.03798219   0.68135542
## omega_2:
##             [,1]          [,2]        [,3]        [,4]        [,5]         [,6]
## [1,] 13124.03314 -1607.4575817  55.4902245 10.57514223 -46.1294604 -19.01456786
## [2,] -1607.45758   268.0587835 -19.9408457 -0.97492617   1.0660778   6.22274328
## [3,]    55.49022   -19.9408457  11.5838030  1.56774855  -1.6214608   0.38733521
## [4,]    10.57514    -0.9749262   1.5677485  0.57638990  -0.6659737   0.03443792
## [5,]   -46.12946     1.0660778  -1.6214608 -0.66597370   1.2799395  -0.14663515
## [6,]   -19.01457     6.2227433   0.3873352  0.03443792  -0.1466352   1.09909359
## omega_3:
##              [,1]          [,2]        [,3]        [,4]         [,5]        [,6]
## [1,] 48975.360425 -6158.2713299 252.858134 -56.1983278 -66.37261346 -4.41167081
## [2,] -6158.271330   895.4166208 -50.513155   6.9493549   0.29159568  3.50285089
## [3,]   252.858134   -50.5131549  10.097317   1.5627613  -1.52188304 -0.31216804
## [4,]   -56.198328     6.9493549   1.562761   0.9203833  -0.82725357 -0.20663077
## [5,]   -66.372613     0.2915957  -1.521883  -0.8272536   1.73823139 -0.09960593
## [6,]    -4.411671     3.5028509  -0.312168  -0.2066308  -0.09960593  0.33693302
##
## diag D:
##  0.1703854 0.9114773 0.03716657 0.3402454 0.1573449 0.08930012 0.01326491 0.009723062 0.1785701 0.435
```

(ii) Error rate and BIC:

For q=1: mfa:

```
mfa.models[[1]]$BIC
```

```
## [1] 7106.716
```

```
classError(mfa.models[[1]]$clust, wine$V1)$errorRate
```

```
## [1] 0.02808989
```

mcfa:

```
mcfa.models[[1]]$BIC
```

```
## [1] 8120.785
```

```
classError(mcfa.models[[1]]$clust, wine$V1)$errorRate
```

```
## [1] 0.3370787
```

For q=2: mfa:

```
mfa.models[[2]]$BIC
```

```
## [1] 7044.433
```

```
classError(mfa.models[[2]]$clust, wine$V1)$errorRate
```

```
## [1] 0.01685393
```

mcfa:

```
mcfa.models[[2]]$BIC
```

```
## [1] 7860.011
```

```
classError(mcfa.models[[2]]$clust, wine$V1)$errorRate
```

```
## [1] 0.241573
```

For q=3: mfa:

```
mfa.models[[3]]$BIC
```

```
## [1] 7034.18
```

```
classError(mfa.models[[3]]$clust, wine$V1)$errorRate
```

```
## [1] 0.02247191
```

mcfa:

```r
mcfa.models[[3]]$BIC
```

```
## [1] 7872.203
```

```r
classError(mcfa.models[[3]]$clust, wine$V1)$errorRate
```

```
## [1] 0.3202247
```

For q=4: mfa:

```r
mfa.models[[4]]$BIC
```

```
## [1] 7050.206
```

```r
classError(mfa.models[[4]]$clust, wine$V1)$errorRate
```

```
## [1] 0.005617978
```

mcfa:

```r
mcfa.models[[4]]$BIC
```

```
## [1] 7164.769
```

```r
classError(mcfa.models[[4]]$clust, wine$V1)$errorRate
```

```
## [1] 0.05617978
```

For q=5: mfa:

```r
mfa.models[[5]]$BIC
```

```
## [1] 7069.775
```

```r
classError(mfa.models[[5]]$clust, wine$V1)$errorRate
```

```
## [1] 0.03370787
```

mcfa:

```r
mcfa.models[[5]]$BIC
```

```
## [1] 7139.037
```

```r
classError(mcfa.models[[5]]$clust, wine$V1)$errorRate
```

```
## [1] 0.06179775
```

For q=6: mfa:

```r
mfa.models[[6]]$BIC
```

```
## [1] 7102.203
```

```r
classError(mfa.models[[6]]$clust, wine$V1)$errorRate
```

```
## [1] 0.005617978
```

mcfa:

```r
mcfa.models[[6]]$BIC
```

```
## [1] 6914.677
```

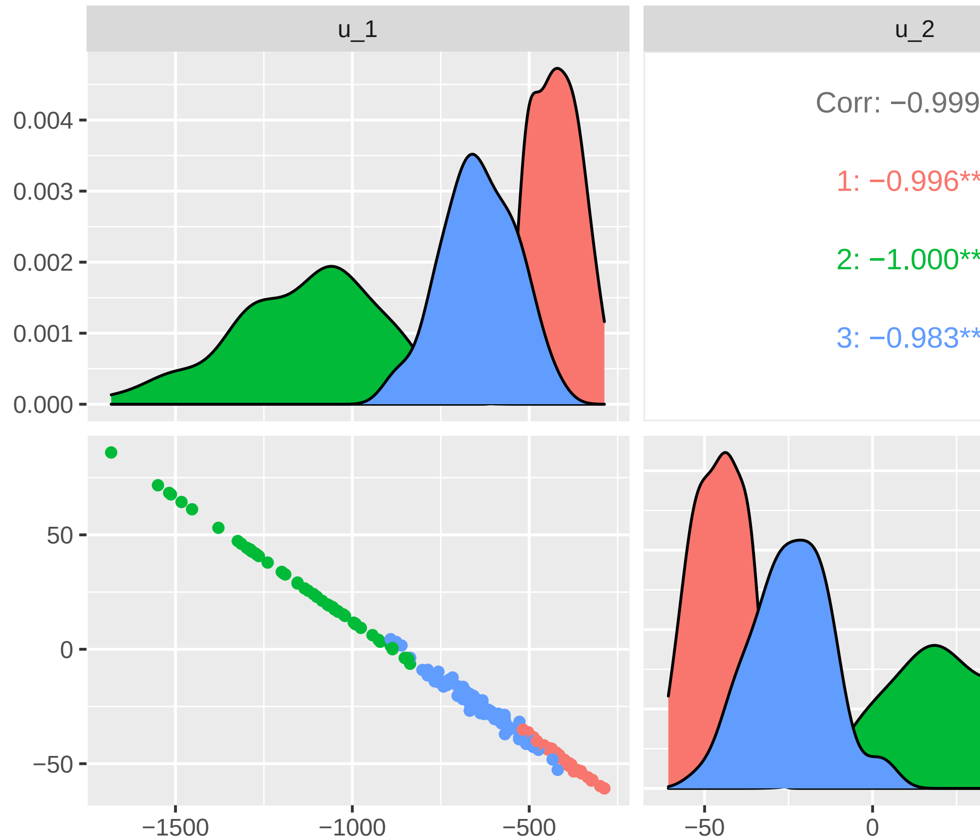
```r
classError(mcfa.models[[6]]$clust, wine$V1)$errorRate
```

```
## [1] 0.06179775
```

Of the mfa models, the model with q=3 has the least BIC (BIC = 7034.175). The error rate of the mfa models is the least for the model with q=4 (error rate = 0.005617978). On the other hand, the mcfa model with the least BIC is the model with q = 6 (BIC = 6914.808). The smallest error rate comes from the model with q = 4 (error rate = 0.05617978). Comparing both, we see that the overall smallest BIC comes from the mcfa models, while the smallest error rate comes from the mfa model.

```r
model <- mcfa(wine_2, g = 3, q = 2, itmax = 200, init_method = "eigen-A")
```

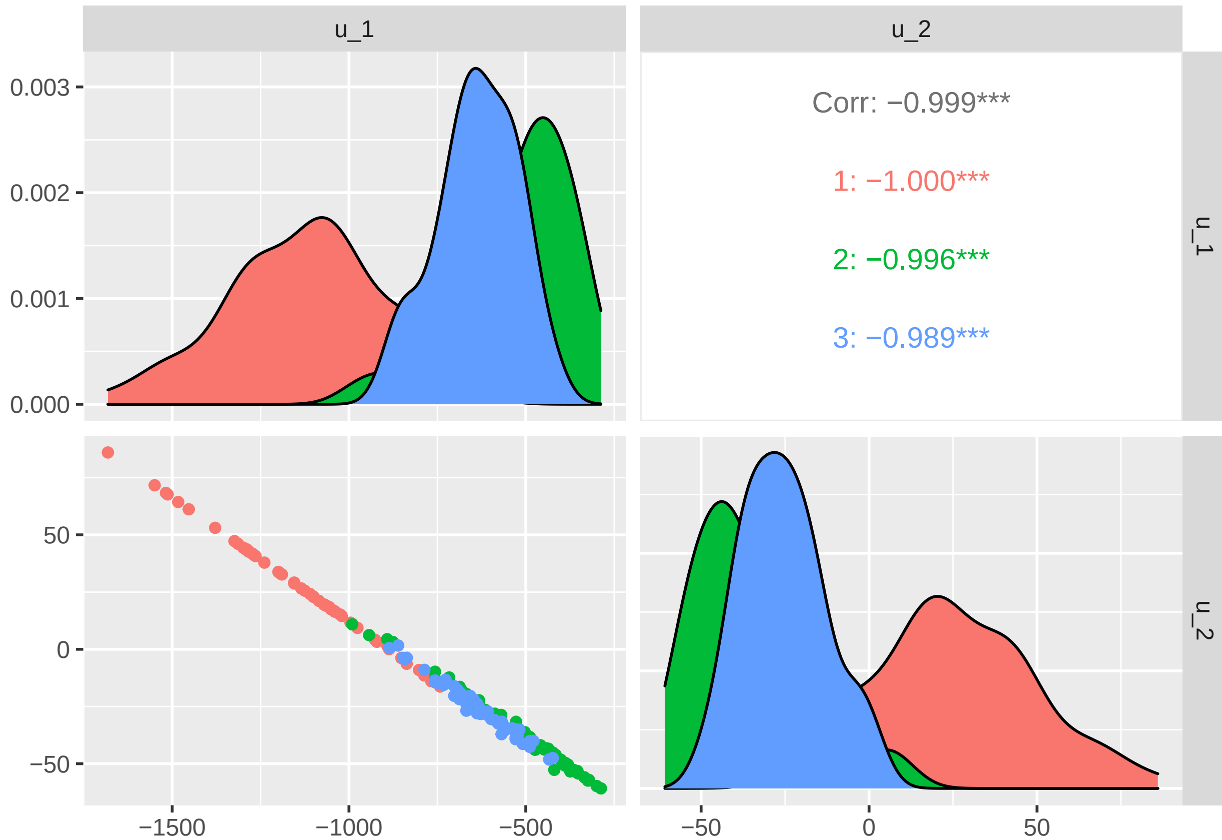
```r
plot_factors(model, type = "Umean")
```



(iii) Plot MCFA clusters for q=2:

The plot on the bottom left shows the clustering from the mcfa model. Next, we make a plot with the real labels shown:

```
plot_factors(model, type = "Umean", clust = wine$V1) #wine$V1 show real labels.
```

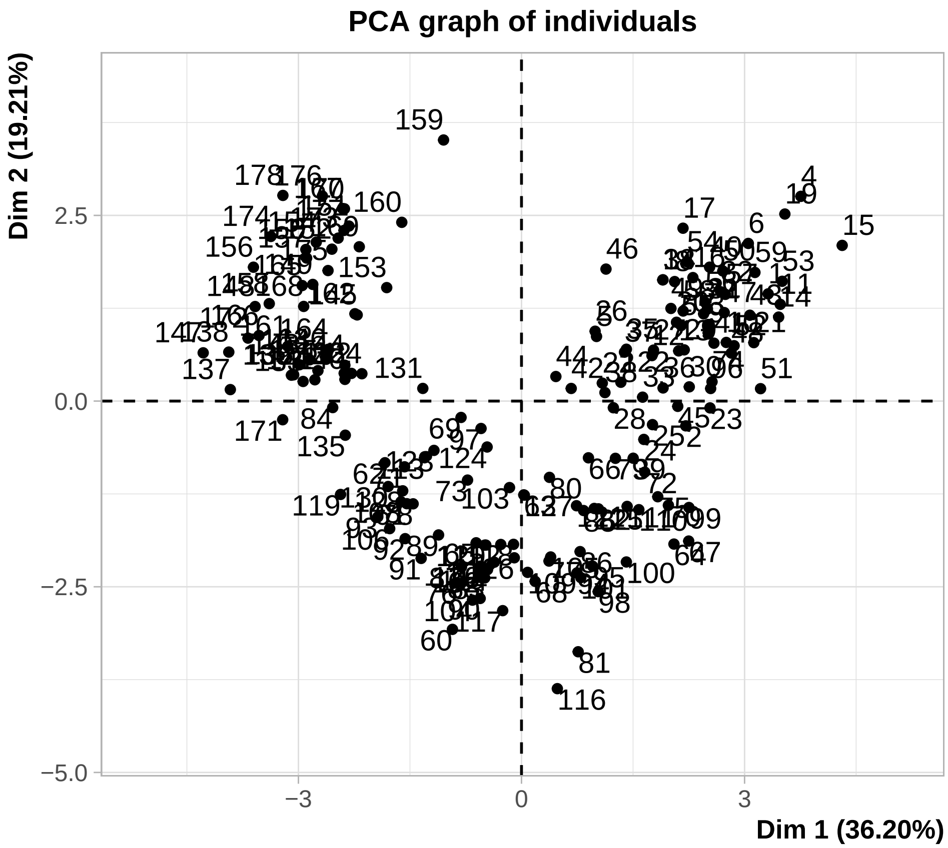


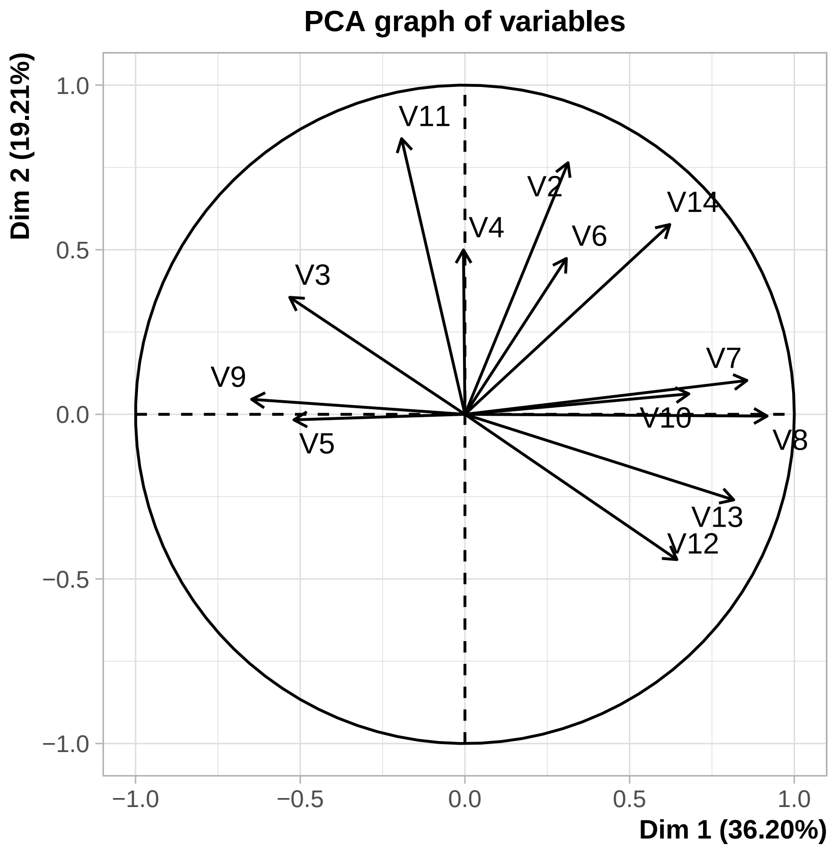
One can see from the plot above that that there are some misclassifications in our mcfa model, as some dots are classified with the wrong color compared to the real labelling.

c)

(i)

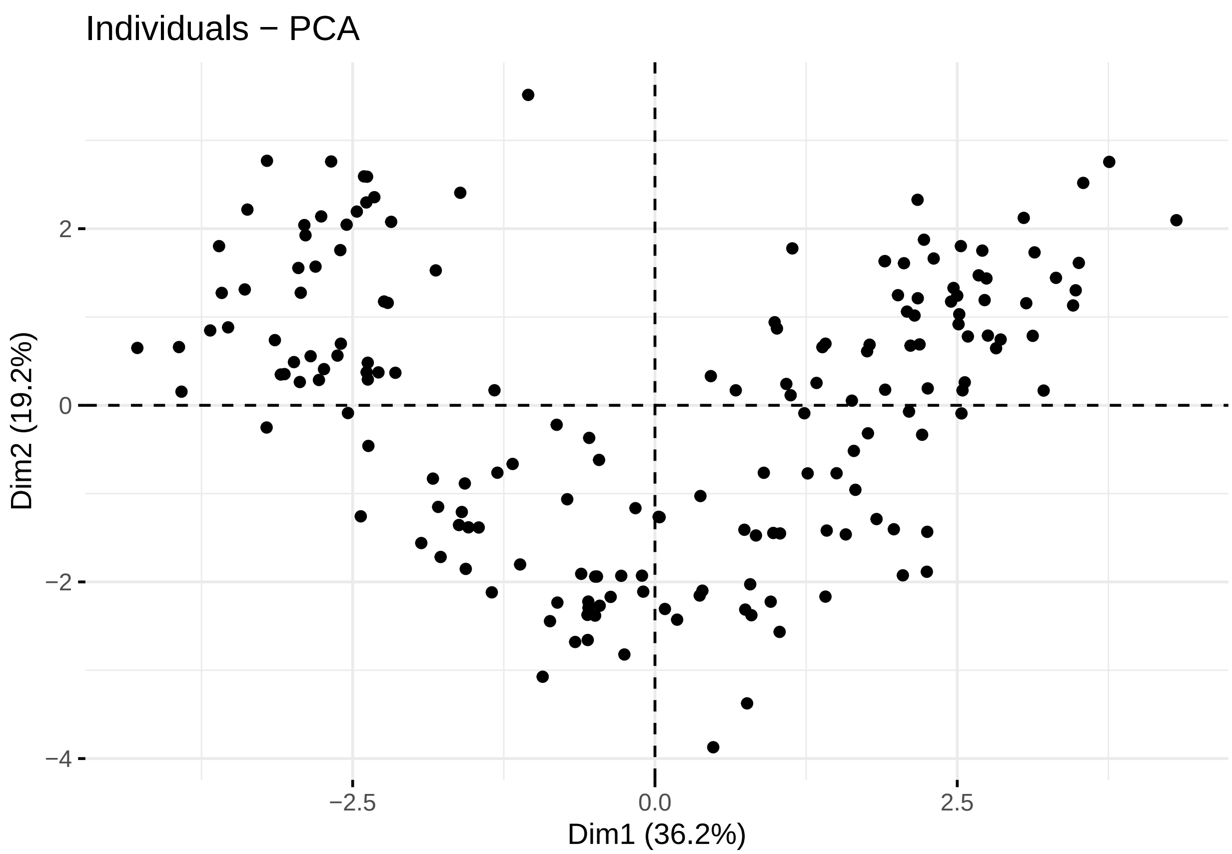
Now, the goal is to use PCA to reduce dimentionality before we fit a mixture model using Mclust. We use the PCA function from the FactoMineR-package.


PCA graph of individuals
Dim 2 (19.21%)
Dim 1 (36.20%)
2.5
0.0
−2.5
−5.0
−3
0
3



A visualization is shown below:

```
fviz_pca_ind(pca.model, label = "none")
```



Now that we have our PC's, we can fit our mixture model. The parameter values are shown below.

```
mixture.pca <- Mclust(data = pca.data, G = 3)

mixture.pca$parameters
```

```
## $pro
## [1] 0.3783078 0.3485848 0.2731074
##
## $mean
##             [,1]       [,2]      [,3]
## Dim.1 2.2093513 -0.2415496 -2.752083
## Dim.2 0.7634274 -1.7812940  1.216084
##
## $variance
## $variance$modelName
## [1] "EEV"
##
## $variance$d
## [1] 2
##
## $variance$G
## [1] 3
##
## $variance$sigma
## , , 1
```

```
##
##           Dim.1     Dim.2
## Dim.1 0.6703286 0.3611780
## Dim.2 0.3611780 0.8666074
##
## , , 2
##
##            Dim.1      Dim.2
## Dim.1  1.0880853 -0.1947453
## Dim.2 -0.1947453  0.4488507
##
## , , 3
##
##           Dim.1     Dim.2
## Dim.1 0.4449628 0.1882162
## Dim.2 0.1882162 1.0919732
##
##
## $variance$scale
## [1] 0.6711648
##
## $variance$shape
## [1] 1.7026249 0.5873284
##
## $variance$orientation
## , , 1
##
##           Dim.1      Dim.2
## Dim.1 0.6073661  0.7944221
## Dim.2 0.7944221 -0.6073661
##
## , , 2
##
##            Dim.1      Dim.2
## Dim.1 -0.9627996 -0.2702163
## Dim.2  0.2702163 -0.9627996
##
## , , 3
##
##            Dim.1      Dim.2
## Dim.1 -0.2604283 -0.9654932
## Dim.2 -0.9654932  0.2604283
```

(ii) Misclassification rate:

We can plot the misclassification rate, like we did in b(ii):

```
classError(mixture.pca$classification, wine$V1)$errorRate
```

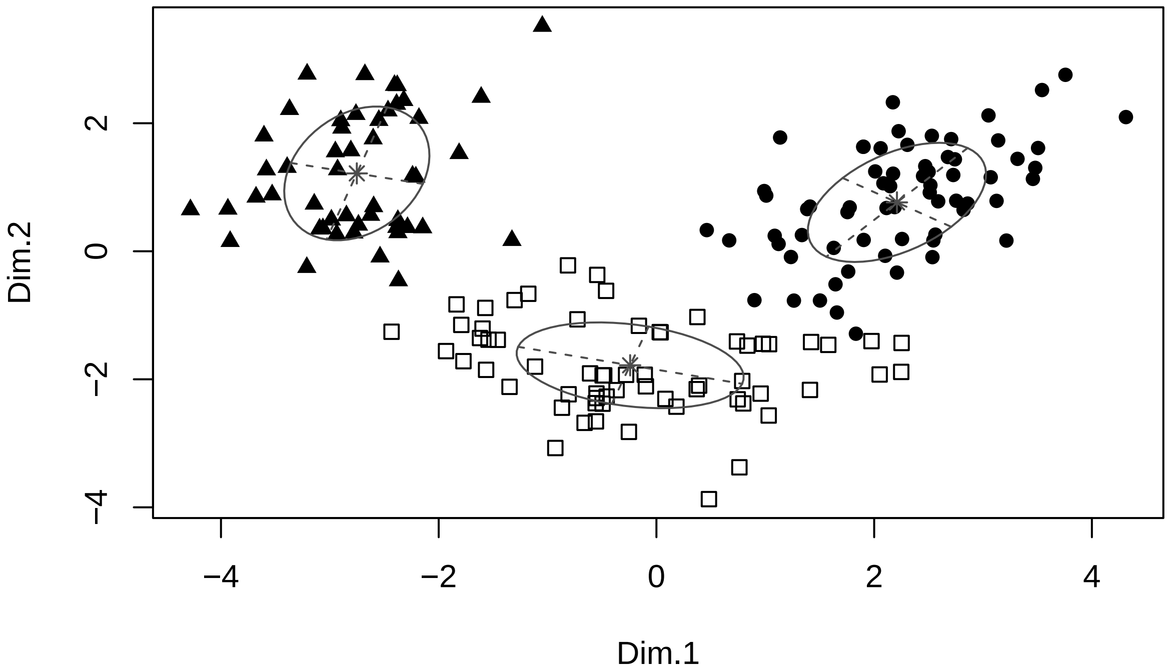
```
## [1] 0.04494382
```

The error rate in this case is larger than the smallest error rate from b(ii) given by the mfa model with q = 2 (error rate = 0.005617978). Hence, fewer misclassifications occur when using mfa on the wine dataset.

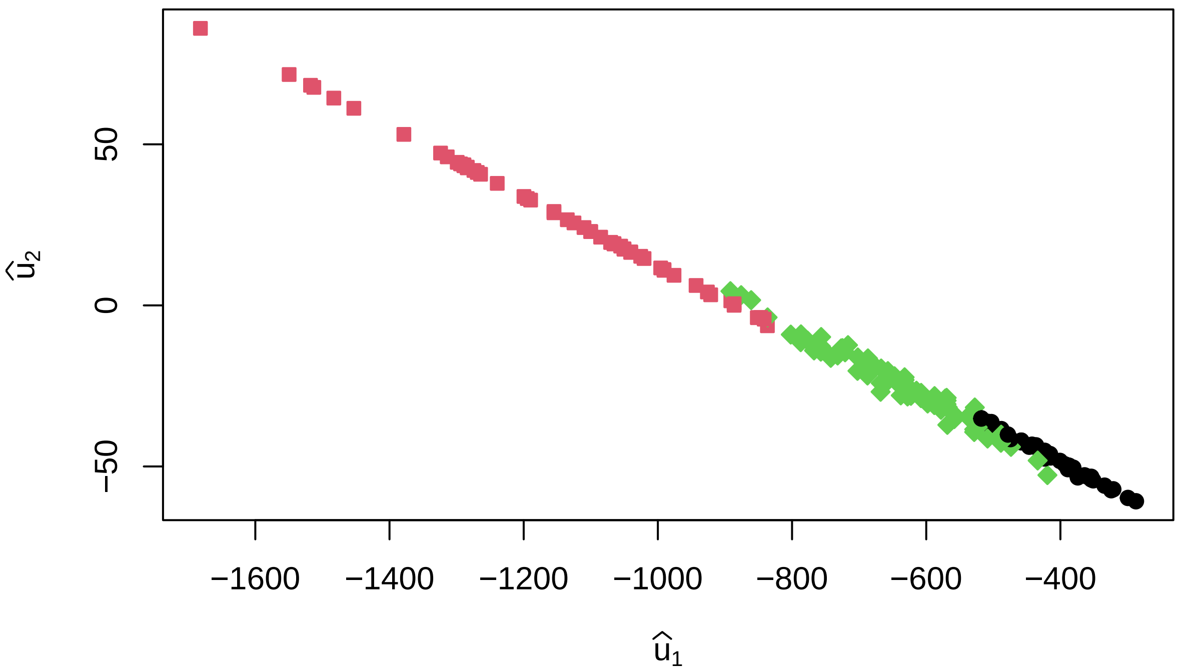
(iii)

Now we compare the PC-method to the mcfa by plotting both. Each point show the true class of origin.

```
plot(mixture.pca, what = "classification", col = wine$V1)
```




```
plot(model, col = wine$V1)
```



These two plots are quite different, as the mcfa model with q = 2 gives a linear shaped graph whereas the PC mixture model returns a cluster structure. Each plot show the class of origin for each point. In the PC plot, one can see that the three classes are clearly separated from each other. On the other hand, the classes in the mcfa are somewhat overlapping.