# Do It Yourself!

S. Lê

## 1. Run HMFA first, to have a feeling of how it works

Please, note that in this example, the structure on the variables doesn't make any sense.

```
library(SensoMineR)
```

```
## Le chargement a nécessité le package : FactoMineR
```

```
?HMFA
data(wine)
dim(wine)
```

```
## [1] 21 31
```

```
hierar <- list(c(2,5,3,10,9,2), c(4,2))
hierar
```

```
## [[1]]
## [1]  2  5  3 10  9  2
##
## [[2]]
## [1] 4 2
```

```
res.hmfa <- HMFA(wine, H = hierar, type=c("n",rep("s",5)),graph=FALSE)
names(res.hmfa)
```

```
## [1] "eig"        "group"      "ind"        "partial"    "quanti.var"
## [6] "quali.var"  "call"
```

```
res.hmfa$eig
```

```
##         eigenvalue percentage of variance cumulative percentage of variance
## comp 1  1.850020562            41.85452103                          41.85452
## comp 2  0.850890396            19.25038601                          61.10491
## comp 3  0.407053504             9.20910274                          70.31401
## comp 4  0.327584774             7.41121698                          77.72523
## comp 5  0.211693454             4.78931333                          82.51454
## comp 6  0.184243479             4.16829019                          86.68283
## comp 7  0.151762324             3.43344259                          90.11627
## comp 8  0.092312127             2.08845238                          92.20473
```

```
## comp 9  0.085169030          1.92684829                     94.13157
## comp 10 0.070400479          1.59272733                     95.72430
## comp 11 0.049700818          1.12442206                     96.84872
## comp 12 0.038018106          0.86011455                     97.70884
## comp 13 0.028136547          0.63655600                     98.34539
## comp 14 0.021966790          0.49697257                     98.84237
## comp 15 0.018103389          0.40956771                     99.25193
## comp 16 0.010366410          0.23452773                     99.48646
## comp 17 0.008498444          0.19226722                     99.67873
## comp 18 0.005681544          0.12853820                     99.80727
## comp 19 0.004329531          0.09795050                     99.90522
## comp 20 0.004189505          0.09478257                    100.00000
```

```
res.hmfa$ind$coord
```

```
##             Dim.1      Dim.2       Dim.3       Dim.4        Dim.5
## 2EL   0.115589635 -0.3839978 -0.63247146 -0.89712973  0.196409100
## 1CHA -1.073384013 -0.8748015 -0.71491035 -1.06137438  0.119528809
## 1FON -0.514932834 -0.8435529 -0.80089351  0.43090044 -0.158675600
## 1VAU -3.312082471  0.1086326  1.12896933  0.19090306  0.426946486
## 1DAM  1.815930443  0.2803452  0.16248808  0.19651032  0.006922444
## 2BOU  0.900713023 -0.3683414 -0.29699797  0.73410583 -0.144255380
## 1BOI  1.210405669 -0.2746202 -0.09748263  0.98165504  0.011017133
## 3EL   0.007179299  0.3242391 -0.85108356 -0.89775551  0.490101883
## DOM1 -0.072338901 -0.4220109  0.30194184 -0.03644779  0.796175745
## 1TUR -0.801117197 -0.0893716  0.69905376 -0.50327664 -0.902505845
## 4EL   0.574079195  0.3282879  0.65002992 -0.37930835 -0.736464625
## PER1  0.666829046  0.5689914  0.55925116 -0.50415500 -0.658246992
## 2DAM  1.608893769 -0.2569175  0.03101310 -0.37739160 -0.103462228
## 1POY  1.516979460 -0.1533753 -0.16734776 -0.20632349 -0.190260417
## 1ING  0.746530209 -0.4027172 -0.63274012  0.30378792  0.175696283
## 1BEN  0.650360906 -0.7940512 -0.04433944  0.86124867 -0.290717306
## 2BEA  0.909112855 -0.3959351  0.93416390  0.26700017  0.970153829
## 1ROC -0.193470297 -0.3921543  1.19642392 -0.26464525  0.246037835
## 2ING -3.441970035 -1.0447129 -0.62878261  0.57610991 -0.433744213
## T1   -0.731136637  2.5119064 -0.16403392  0.46926827 -0.166978738
## T2   -0.582171124  2.5741573 -0.63225169  0.11631811  0.346321799
```

## 2. Perform your MFAs, at the finest grain, i.e. at the first level of the hierarchy

First, run the separate analyses.

```
res.g1 <- MCA(wine[,1:2],graph = F)
res.g2 <- PCA(wine[,3:7],graph = F)
res.g3 <- PCA(wine[,8:10],graph = F)
res.g4 <- PCA(wine[,11:20],graph = F)
```

Then, get the dimensionality of the data sets.

```
dim(res.g1$eig)
```

```
## [1] 5 3
```

```r
dim(res.g2$eig)
```

```
## [1] 5 3
```

```r
dim(res.g3$eig)
```

```
## [1] 3 3
```

```r
dim(res.g4$eig)
```

```
## [1] 10  3
```

Save the results with the proper number of dimensions.

```r
res.g1 <- MCA(wine[,1:2],graph = F)
res.g2 <- PCA(wine[,3:7],graph = F)
res.g3 <- PCA(wine[,8:10],graph = F)
res.g4 <- PCA(wine[,11:20],graph = F,ncp=10)
```

Build a vector of weights (first level of the hierarchy, first node)

```r
w.L1.1 <- c(1/rep(res.g1$eig[1,1],5),1/rep(res.g2$eig[1,1],5),
            1/rep(res.g3$eig[1,1],3),1/rep(res.g4$eig[1,1],10))
```

From the original 4 blocks of data to the 4 blocks of coordinates…

```r
L1.1 <- cbind(res.g1$ind$coord,res.g2$ind$coord,res.g3$ind$coord,res.g4$ind$coord)
```

Run your first MFA with a PCA program.

```r
res.pca.w.L1.1 <- PCA(L1.1,scale.unit = F,col.w = w.L1.1,graph = F,ncp = 20)
res.pca.w.L1.1$ind$coord[,1:5]
```

```
##            Dim.1      Dim.2       Dim.3        Dim.4       Dim.5
## 2EL    0.3297217 -0.3578156 -0.47475980 -1.772718606  0.20494894
## 1CHA -1.9921118 -0.5669002 -0.67531935 -2.132658250 -0.18856738
## 1FON -1.6244165 -0.6163498 -1.49970567  0.566496746 -0.13722855
## 1VAU -3.8392802  0.8580030  1.52406233  0.503972805  0.60386976
## 1DAM  2.8777837 -0.4384994  0.14794681  0.026945728 -0.10629942
## 2BOU  0.6591295 -1.1234093 -0.89904777  0.909888684 -0.29839212
## 1BOI  1.4442031 -1.0612182 -0.68301031  1.204860213 -0.01914242
## 3EL  -0.1886607  0.8581779 -0.85159139 -1.851929127  0.56044692
## DOM1 -0.6582326 -0.4270815  0.48455047  0.248117782  1.69897162
## 1TUR -1.0690730  0.2301602  1.49245782 -0.194288316 -1.50392162
## 4EL   0.4768726  0.4852073  1.23008094 -0.078196117 -1.20411213
## PER1  1.1591611  0.6625783  1.17334777 -0.298600619 -1.05161409
## 2DAM  1.8503275 -0.7224950  0.13474072 -0.680443110 -0.28833529
## 1POY  1.5443544 -0.4589019  0.02493772 -0.316373161 -0.36560125
## 1ING  0.5498577 -1.0520191 -1.23586229  0.007324811  0.23526535
```

```
## 1BEN  0.1880902 -1.2609589 -0.57777633  1.358947942 -0.36773244
## 2BEA  1.4100775 -1.5077947  1.23996264  0.235908680  1.58499095
## 1ROC -0.8738803 -0.3378355  2.01267085  0.251176289  0.63357323
## 2ING -3.9973507 -0.5605301 -1.17647506  0.799222995 -0.63818454
## T1    1.1623182  3.5408746 -0.32016898  1.079161117  0.02639492
## T2    0.5911084  3.8568076 -1.07104113  0.133183514  0.62066955
```

```r
res.pca.w.L1.1$eig
```

```
##         eigenvalue percentage of variance cumulative percentage of variance
## comp 1  2.953254141           33.312935920                          33.31294
## comp 2  1.873149351           21.129270057                          54.44221
## comp 3  1.082348860           12.208979140                          66.65119
## comp 4  0.881901164            9.947913573                          76.59910
## comp 5  0.604481735            6.818600886                          83.41770
## comp 6  0.392231431            4.424400984                          87.84210
## comp 7  0.332176653            3.746978431                          91.58908
## comp 8  0.231162135            2.607526828                          94.19661
## comp 9  0.161102662            1.817250536                          96.01386
## comp 10 0.100884078            1.137980227                          97.15184
## comp 11 0.083294954            0.939573542                          98.09141
## comp 12 0.070252190            0.792450153                          98.88386
## comp 13 0.033660364            0.379691514                          99.26355
## comp 14 0.021481970            0.242318287                          99.50587
## comp 15 0.019210626            0.216697354                          99.72257
## comp 16 0.012241490            0.138084960                          99.86065
## comp 17 0.008559353            0.096550162                          99.95720
## comp 18 0.001801935            0.020325971                          99.97753
## comp 19 0.001585659            0.017886359                          99.99541
## comp 20 0.000406479            0.004585115                         100.00000
```

Compare with MFA

```r
res.mfa.L1.1 <- MFA(wine[,1:20],group = c(2,5,3,10),type = c("n","s","s","s"),graph = F)
res.mfa.L1.1$ind$coord
```

```
##           Dim.1      Dim.2       Dim.3        Dim.4       Dim.5
## 2EL   0.3297217 -0.3578156 -0.47475980  1.772718606  0.20494894
## 1CHA -1.9921118 -0.5669002 -0.67531935  2.132658250 -0.18856738
## 1FON -1.6244165 -0.6163498 -1.49970567 -0.566496746 -0.13722855
## 1VAU -3.8392802  0.8580030  1.52406233 -0.503972805  0.60386976
## 1DAM  2.8777837 -0.4384994  0.14794681 -0.026945728 -0.10629942
## 2BOU  0.6591295 -1.1234093 -0.89904777 -0.909888684 -0.29839212
## 1BOI  1.4442031 -1.0612182 -0.68301031 -1.204860213 -0.01914242
## 3EL  -0.1886607  0.8581779 -0.85159139  1.851929127  0.56044692
## DOM1 -0.6582326 -0.4270815  0.48455047 -0.248117782  1.69897162
## 1TUR -1.0690730  0.2301602  1.49245782  0.194288316 -1.50392162
## 4EL   0.4768726  0.4852073  1.23008094  0.078196117 -1.20411213
## PER1  1.1591611  0.6625783  1.17334777  0.298600619 -1.05161409
## 2DAM  1.8503275 -0.7224950  0.13474072  0.680443110 -0.28833529
## 1POY  1.5443544 -0.4589019  0.02493772  0.316373161 -0.36560125
## 1ING  0.5498577 -1.0520191 -1.23586229 -0.007324811  0.23526535
## 1BEN  0.1880902 -1.2609589 -0.57777633 -1.358947942 -0.36773244
```

```
## 2BEA  1.4100775 -1.5077947  1.23996264 -0.235908680  1.58499095
## 1ROC -0.8738803 -0.3378355  2.01267085 -0.251176289  0.63357323
## 2ING -3.9973507 -0.5605301 -1.17647506 -0.799222995 -0.63818454
## T1    1.1623182  3.5408746 -0.32016898 -1.079161117  0.02639492
## T2    0.5911084  3.8568076 -1.07104113 -0.133183514  0.62066955
```

```
res.mfa.L1.1$eig
```

```
##          eigenvalue percentage of variance cumulative percentage of variance
## comp 1  2.953254141           33.312935920                          33.31294
## comp 2  1.873149351           21.129270057                          54.44221
## comp 3  1.082348860           12.208979140                          66.65119
## comp 4  0.881901164            9.947913573                          76.59910
## comp 5  0.604481735            6.818600886                          83.41770
## comp 6  0.392231431            4.424400984                          87.84210
## comp 7  0.332176653            3.746978431                          91.58908
## comp 8  0.231162135            2.607526828                          94.19661
## comp 9  0.161102662            1.817250536                          96.01386
## comp 10 0.100884078            1.137980227                          97.15184
## comp 11 0.083294954            0.939573542                          98.09141
## comp 12 0.070252190            0.792450153                          98.88386
## comp 13 0.033660364            0.379691514                          99.26355
## comp 14 0.021481970            0.242318287                          99.50587
## comp 15 0.019210626            0.216697354                          99.72257
## comp 16 0.012241490            0.138084960                          99.86065
## comp 17 0.008559353            0.096550162                          99.95720
## comp 18 0.001801935            0.020325971                          99.97753
## comp 19 0.001585659            0.017886359                          99.99541
## comp 20 0.000406479            0.004585115                         100.00000
```

Now, your turn. Do the same thing for the second node (still, at the first level of the hierachy, the finest one).

You have just performed two MFAs: one with 4 blocks, one with 2 blocks.

### 3. MFA at the second (last) level of the hierarchy : my first HMFA

As explained in the lecture, an HMFA is nothing else but a sequence of MFAs: run an MFA (based on PCAs) on the two matrices of coordinates obtained from the MFAs on the first 4 blocks, and on the 2 remainig blocks. Don't forget to build a vector of weights for your PCA. Compare the results with the **HMFA()** function.

Good luck.