eNote 10

Multidimensional Scaling: Metric and non-metric (PCO and NMS)

Indhold

10.1 Reading material

You can find something on Wikipedia:

http://en.wikipedia.org/wiki/Multidimensional_scaling

You can also find something in the Wehrens book:

- 4.6.1 Multidimensional Scaling

Or a little related thing in the Varmuza book:

- 3.8.4 SAMMON'S NONLINEAR MAPPING

Or a little something as a part of the HSAUR3-package supporting the book:

A Handbook of Statistical Analyses Using R (Torsten Hothorn and Brian S. Everitt, Chapman & Hall/CRC, 2014):

http://cran.r-project.org/web/packages/HSAUR3/vignettes/Ch_multidimensional_scaling.pdf

10.2 QuickR

From http://www.statmethods.net/advstats/mds.html

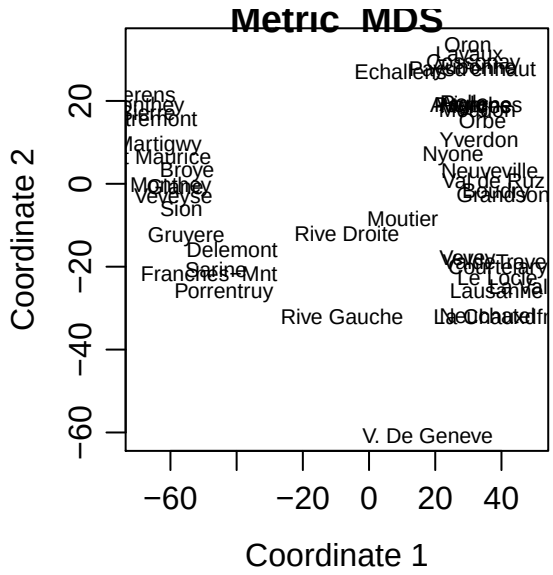
we get the advice:

You can perform a classical MDS using the `cmdscale` function: (using here the data `swiss` as an example - 5 socio-economic vartiables for each of 47 French-speaking provinces of Switzerland at about 1888)

```
# Classical MDS
# N rows (objects) x p columns (variables)
# each row identified by a unique row name

mydata <- swiss[-1]
d <- dist(mydata) # euclidean distances between the rows
fit <- cmdscale(d,eig=TRUE, k=2) # k is the number of dim
fit # view results


$points
                   [,1]        [,2]
Courtelary    38.959229 -20.4050378
Delemont     -41.355148 -15.7512320
Franches-Mnt -48.301104 -21.6604943
Moutier       10.213911  -8.2701294
Neuveville    36.492876   3.2193826
Porrentruy   -43.841141 -26.1327238
Broye        -54.955969   2.8607597
Glane        -58.571774  -0.5535762
Gruyere      -55.233296 -12.7889552
Sarine       -46.179341 -20.6685623
Veveyse      -59.133043  -3.2799165
Aigle         28.443006  18.8672216
Aubonne       31.668563  28.3422952
Avenches      32.388508  19.1854054
Cossonay      31.492152  29.0042126
Echallens      9.566146  27.0254651
Grandson      40.531787  -2.4261476
Lausanne      38.524115 -25.7824883
La Vallee     49.403878 -24.6801735
```

```
Lavaux         30.289534  31.5217925
Morges         32.176260  18.3380262
Moudon         32.753425  17.8815434
Nyone          25.396855   6.8745590
Orbe           34.312715  15.2555024
Oron           29.891663  33.7488563
Payerne        31.511744  18.7369553
Paysd'enhaut   31.331441  27.3752618
Rolle          28.822708  19.5441310
Vevey          29.754085 -18.2082210
Yverdon        33.240504  10.7240944
Conthey       -67.665349  18.6374853
Entremont     -66.688133  15.7332926
Herens        -68.802770  21.4984357
Martigwy      -63.510289   9.1680196
Monthey       -59.850151  -0.5754368
St Maurice    -63.127868   6.7253853
Sierre        -67.016009  17.2303805
Sion          -56.692503  -5.8562753
Boudry         38.405649  -2.0545973
La Chauxdfnd   39.587521 -31.8758352
Le Locle       38.784377 -22.5165825
Neuchatel      35.751474 -31.6926632
Val de Ruz     37.631896   0.9083478
ValdeTravers   40.668340 -18.6393992
V. De Geneve   17.737682 -60.6632384
Rive Droite    -6.667866 -11.8821291
Rive Gauche    -8.140292 -32.0429967

$eig
 [1]  8.648449e+04  2.112744e+04  2.706138e+03  6.392245e+02  3.480073e+02
 [6]  6.366270e-12  5.049680e-12  4.806995e-12  3.742979e-12  3.299345e-12
[11]  2.582295e-12  1.282168e-12  1.237885e-12  9.035419e-13  6.749936e-13
[16]  5.995660e-13  5.146883e-13  4.363656e-13  4.269951e-13  4.086160e-13
[21]  3.924628e-13  3.546247e-13  2.852941e-13  2.493990e-13  2.277657e-13
[26]  2.084816e-13  1.397871e-13  1.183306e-13  2.008992e-14 -2.573070e-13
[31] -3.324318e-13 -5.271718e-13 -7.493299e-13 -7.671986e-13 -8.638812e-13
[36] -1.059992e-12 -1.101320e-12 -1.133792e-12 -1.308953e-12 -1.566165e-12
[41] -1.733716e-12 -1.843208e-12 -2.075134e-12 -2.079606e-12 -2.157792e-12
[46] -6.311337e-12 -8.764151e-12
```

```
$x
NULL

$ac
[1] 0

$GOF
[1] 0.9668177 0.9668177


# plot solution
x <- fit$points[,1]
y <- fit$points[,2]
plot(x, y, xlab="Coordinate 1", ylab="Coordinate 2",
  main="Metric  MDS",  type="n")
text(x, y, labels = row.names(mydata), cex=.7)
```

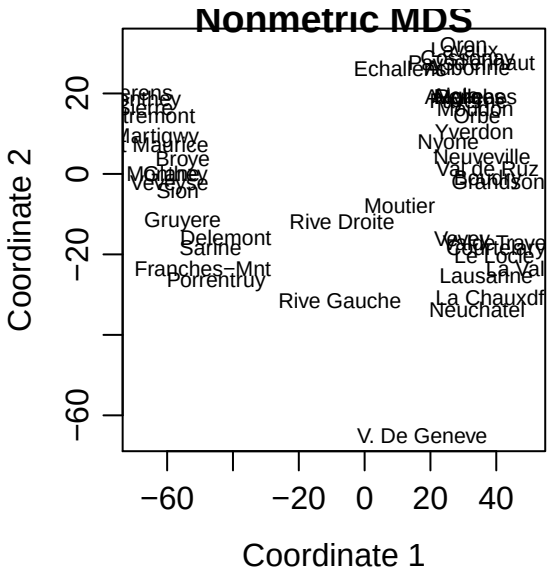


Nonmetric MDS is performed using the `isoMDS` function in the MASS package.

```
# Nonmetric MDS
# N rows (objects) x p columns (variables)
# each row identified by a unique row name

library(MASS)
d <- dist(mydata) # euclidean distances between the rows
```

```
fit <- isoMDS(d, k=2) # k is the number of dim


initial  value 2.979731
iter   5 value 2.431486
iter  10 value 2.343353
final  value 2.338839
converged


fit # view results


$points
                   [,1]        [,2]
Courtelary    39.976648 -18.6652658
Delemont     -42.144880 -15.8354737
Franches-Mnt -49.183472 -23.5017064
Moutier       10.640047  -7.7948873
Neuveville    35.713299   4.2241084
Porrentruy   -45.069549 -26.7640938
Broye        -55.273717   3.1907784
Glane        -58.641853   0.1769567
Gruyere      -55.274386 -11.8305230
Sarine       -46.800689 -18.2672079
Veveyse      -59.271433  -2.7666474
Aigle         27.639864  18.9592510
Aubonne       31.354629  26.3662771
Avenches      32.353450  19.3520700
Cossonay      31.342613  28.5036282
Echallens     10.799996  26.1967959
Grandson      40.595874  -1.7999750
Lausanne      36.972991 -25.1351295
La Vallee     50.122185 -23.5248585
Lavaux        30.346198  30.8619325
Morges        31.803525  18.6695352
Moudon        33.621794  16.2266437
Nyone         25.423297   7.4959696
Orbe          34.176435  14.5941476
Oron          30.076709  32.2309096
Payerne       31.559866  17.6683208
Paysd'enhaut  32.631374  27.0802839
```

```
Rolle          28.769137  19.1245372
Vevey          29.599603 -16.5009238
Yverdon        33.286505  10.5935022
Conthey       -67.755640  18.1702425
Entremont     -66.402167  14.4997009
Herens        -68.766153  20.1878849
Martigwy      -63.370151   8.9517543
Monthey       -60.005854  -0.5353731
St Maurice    -62.872702   7.3903698
Sierre        -66.745414  16.5445662
Sion          -56.775840  -4.1215356
Boudry         37.427364  -1.3796084
La Chauxdfnd   41.684937 -30.7139972
Le Locle       39.469769 -20.2862591
Neuchatel      34.177795 -33.7884464
Val de Ruz     37.403309   1.2494697
ValdeTravers   42.204515 -17.0228055
V. De Geneve   17.524061 -65.0238668
Rive Droite    -6.829233 -11.9126783
Rive Gauche    -7.514655 -31.3383738

$stress
[1] 2.338839


# plot solution
x <- fit$points[,1]
y <- fit$points[,2]
plot(x, y, xlab="Coordinate 1", ylab="Coordinate 2",
  main="Nonmetric MDS", type="n")
text(x, y, labels = row.names(mydata), cex=.7)
```

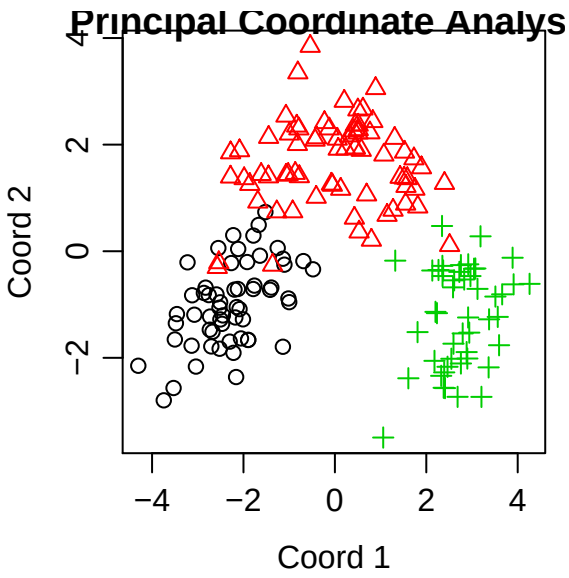


At http://gastonsanchez.com/blog/how-to/2013/01/23/MDS-in-R.html we learn that there are 7 Functions to do Metric Multidimensional Scaling in R.

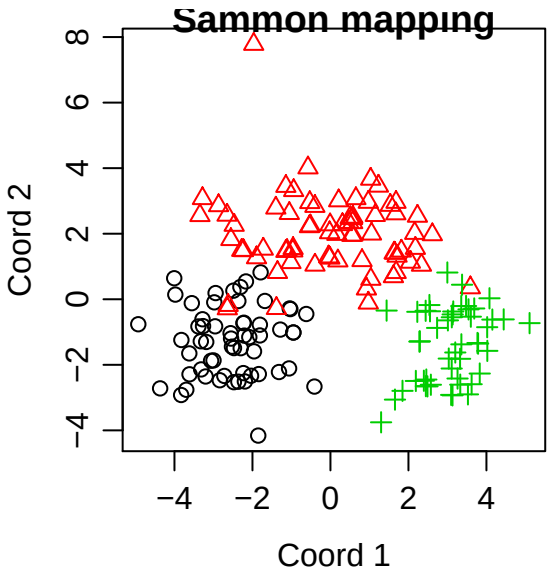
10.3 Example from Wehren's book:

(A little adaptation of the code has been necessary to match the newest version of the `cmdscale` result structure.)

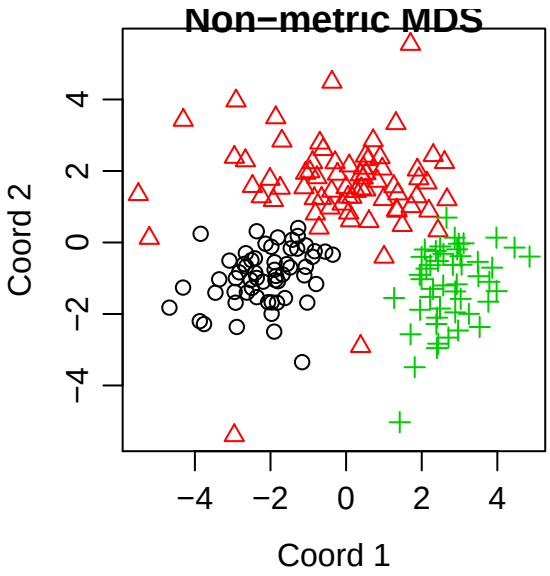
```
library(ChemometricsWithRData, warn.conflicts=FALSE, quietly = TRUE)
# First metric:
data(wines)
wines.dist <- dist(scale(wines))
wines.cmdscale <- cmdscale(wines.dist)
plot(wines.cmdscale, pch = wine.classes, col = wine.classes,
     main ="Principal Coordinate Analysis",
     xlab = "Coord 1", ylab = "Coord 2")
```




```
# Then Non- metric with Sammon's method:
wines.sammon <- sammon(wines.dist)


Initial stress        : 0.14753
stress after   0 iters: 0.14753


wines.sammon <- sammon(wines.dist, magic = .00003)


Initial stress        : 0.14753
stress after  10 iters: 0.14347, magic = 0.002
stress after  19 iters: 0.11251


plot(wines.sammon$points, main = "Sammon mapping",
     col = wine.classes, pch = wine.classes,
     xlab = "Coord 1", ylab = "Coord 2")
```



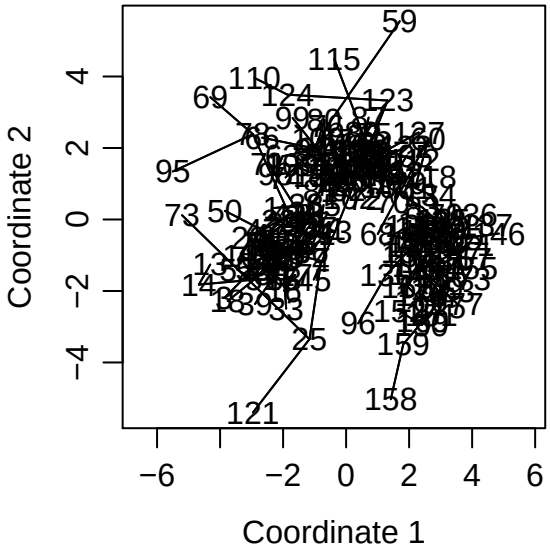

```
# Then Non- metric with Kruskal's method:

wines.isoMDS <- isoMDS(wines.dist)


initial  value 25.980817
iter   5 value 19.977649
iter  10 value 17.798597
iter  15 value 17.327216
iter  20 value 17.128918
iter  20 value 17.116929
iter  20 value 17.111706
final  value 17.111706
converged


plot(wines.isoMDS$points, main = "Non-metric MDS",
     col = wine.classes, pch = wine.classes,
     xlab = "Coord 1", ylab = "Coord 2")
```



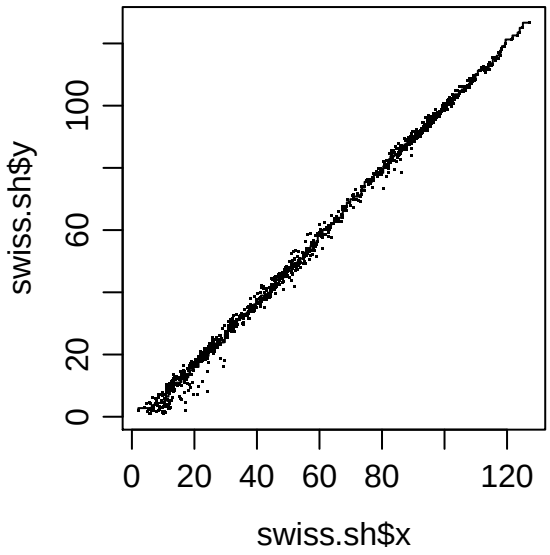
10.4 Misc

We can plot a Minimum spanning tree using the `mst` function of the `ape` package:

```
# Minimum spanning tree
library(ape)

x <- wines.isoMDS$points[,1]
y <- wines.isoMDS$points[,2]

st <- mst(as.matrix(wines.dist))

plot(x, y, xlab = "Coordinate 1", ylab = "Coordinate 2",
 xlim = range(x)*1.2, type = "n")
for (i in 1:nrow(as.matrix(wines.dist))) {
 w1 <- which(st[i, ] == 1)
 segments(x[i], y[i], x[w1], y[w1])
}
text(x, y, labels=colnames(as.matrix(wines.dist)))
```




```
# The Shepard diagram
swiss.sh <- Shepard(swiss.dist, swiss.mds$points)
plot(swiss.sh, pch = ".")
lines(swiss.sh$x, swiss.sh$yf, type = "S")
```



10.5 Distance measures

The `vegdist` of the vegan package:

http://cc.oulu.fi/~jarioksa/softhelp/vegan/html/vegdist.html

and the `dist.binary` of the ade4 package:

http://pbil.univ-lyon1.fr/ade4/ade4-html/dist.binary.html

can find all sorts of different dissimilarity-measures - e.g. Jaccard. The Pearson Phi is very close to the co-called "Yule-coefficient.

The definitions of most of the binary measures can e.g. be found here:

http://fitelson.org/coherence/deza_17.3.pdf

10.6 Exercises

|||| Exercise 1 Distances between cities on Sealand

Can you recreate the map from a distance matrix? (Data: `Sjaelland.txt`)

```
# Read the data and store it as a distance matrix
Sealand_dist <- as.dist(read.table("Sjaelland.txt",
                                   fill = TRUE, header = T))
```

a) Given the distance matrix, make a PCO of the data to recreate the map of Sealand. If necessary: multiply scores by -1 to achieve the proper directions.

b) How many dimensions do you need and what do the first two PCO axes explain concerning variation in the data?

c) Overlay the PCO score plot with a MST (minimum spanning tree). Would the MST give a good or poor idea of how to make railway connections between cities on Sealand?

d) Find out if the NMS solution will give a better result (use the PCO solution as the initial matrix). Is there a reason to believe NMS should be better than PCO in this case? What is the stress value?

|||| Exercise 2 Flying times

Use the data set `flying.txt`:

```
# Read the data and store it as a distance matrix
flying_dist <- as.dist(read.table("flying.txt",
                                  fill = TRUE, header = T))
```

a) How will the recreated map of Europe look like?

b) Does the MST give meaning?

c) Make a new file with ranked distances (1 for the shortest flying time, 2 for the next shortest etc.). Will such a matrix give any meaningful map of Europe?

d) Is the NMS solution better than the PCO solution in this regard?

|||| Exercise 3 Secondary metabolite data

Use the data set `sm.txt`:

```
# Read the data and store it as a 129-by-51 matrix
# of zeros and ones:
smdata <- matrix(scan("sm.txt"), ncol = 51, byrow = T)
```

a) Compare the PCO /NMS solutions to in three dimensions on the secondary metabolite data set based on the Jaccard and other similarity coefficients, e.g. the Pearson Phi. Which of the similarity measures give the highest percentage of explanation of variance on the first three PCO and NMS axes. Which give the best stress values?

b) Overlay the MST in all 6 cases. Do they appear to show any deviations between the (PCO) ordination and the MST clustering?