Evidential clustering of time series data

Thierry Denoeux

02/09/2022

Data analysis

We consider the "Synthetic control chart" data:

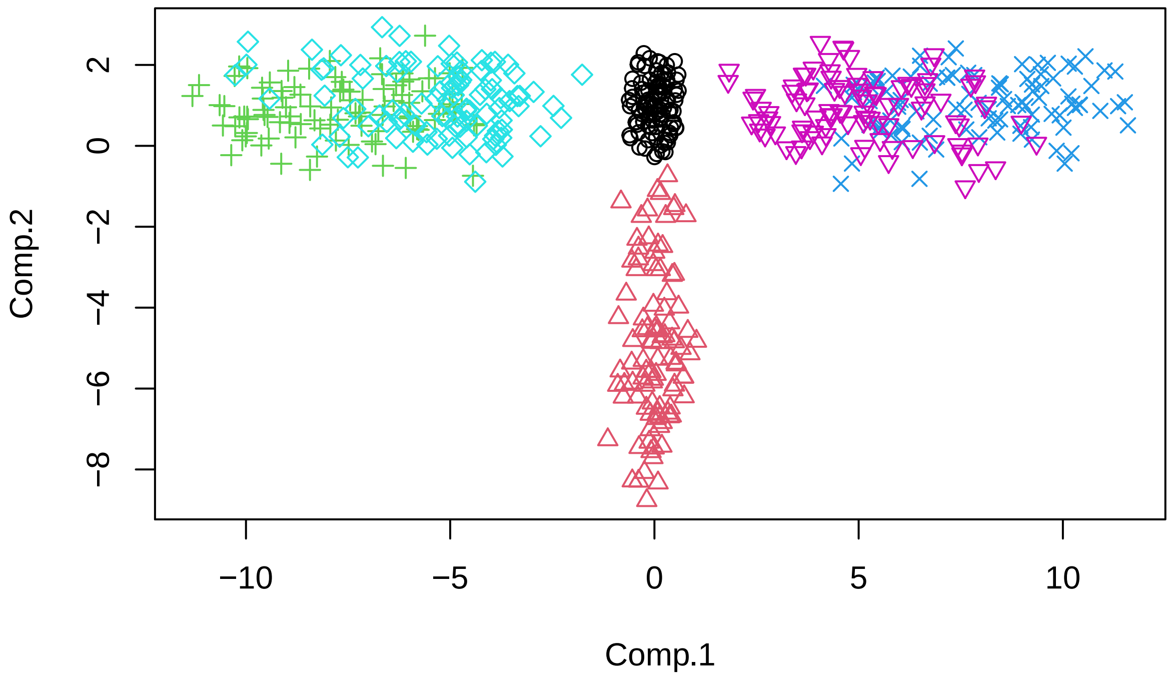
```
data<-read.table(file="/Users/Thierry/Documents/R/Data/Economics/synthetic_control.data")
ytrue<-c(rep(1,100),rep(2,100),rep(3,100),rep(4,100),rep(5,100),rep(6,100))
x<-as.matrix(data)
```

There are 6 classes with the following descriptions:

1. Normal
2. Cyclic
3. Increasing trend
4. Decreasing trend
5. Upward shift
6. Downward shift

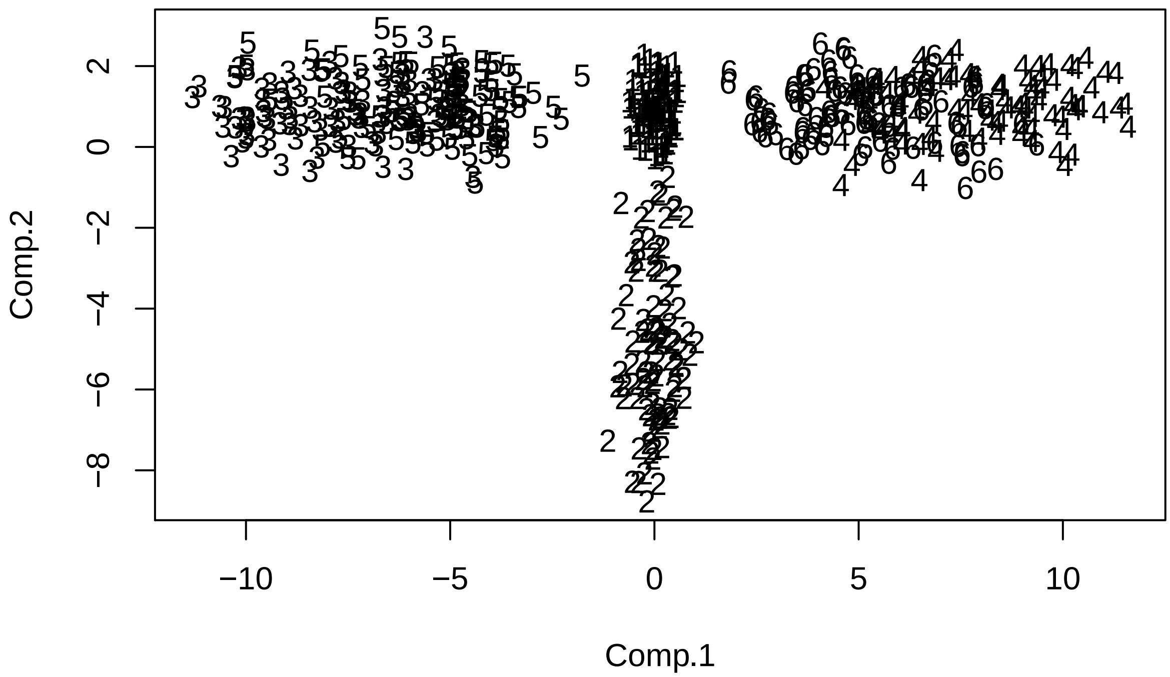
Let us plot the data in the space of the first two principal components:

```
xn<-scale(x)
pca<-princomp(xn)
z<-pca$scores[,1:2]
plot(z,pch=ytrue,col=ytrue)
```



We can see the definitions of the six classes in this plot:

```
plot(z,type="n")
text(z,as.character(ytrue))
```

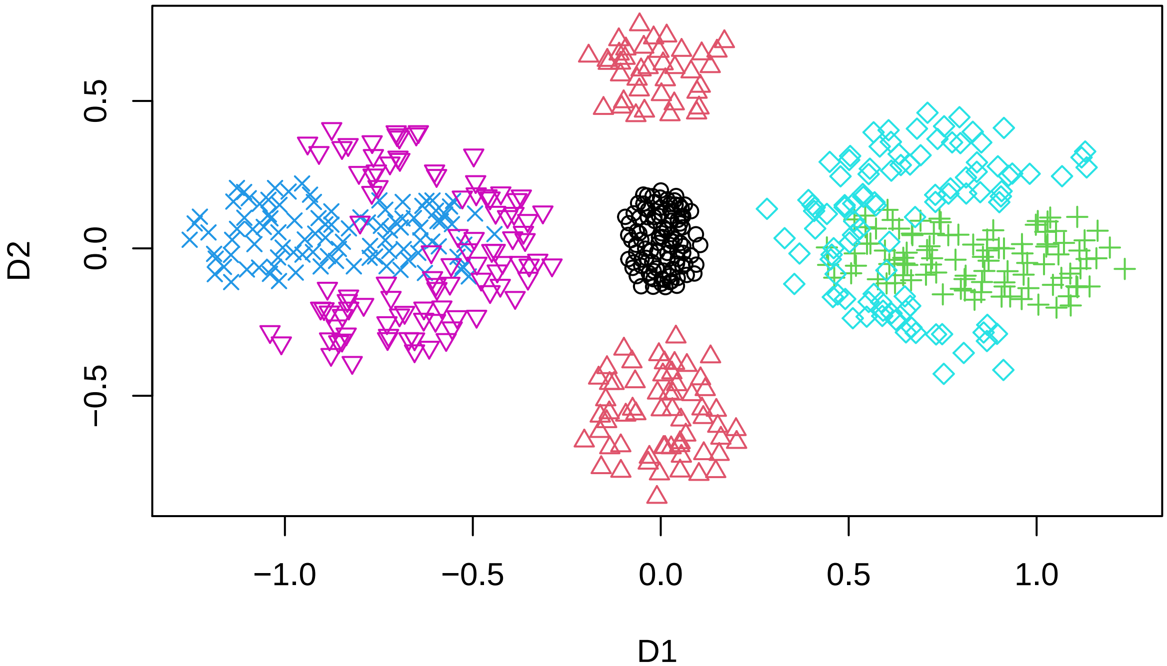


Classes 1 and 2 are well separated, but there is ambiguity between classes 3 and 5 on the one hand, and classes 4 and 6 on the other hand.

We can try to apply MDS to the Euclidean distance matrix:

```
library(smacof)
fit.mds1<-mds(dist(x),ndim=2,type="ratio")
```

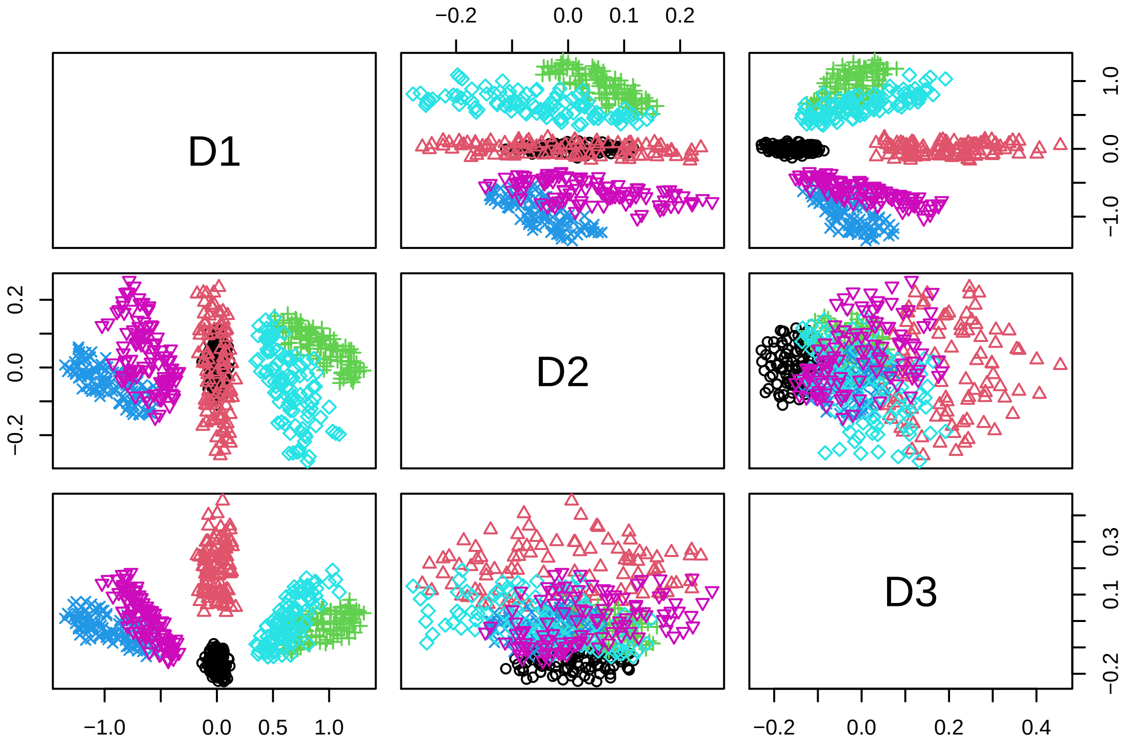
```r
plot(fit.mds1$conf,pch=ytrue,col=ytrue)
```



Classes 3/5 and 4/6 are even more intertwined, and classes 1 and 2 are not linearly separable.

We can now try to use the dynamic time warping distance:

```r
library(dtw)
#D<-dtwDist(x)
#save(D,file="DTW_synthetic_control.RData")
# The distance matrix is saved in a file to avoid recomputing it at each compilation
load("/Users/Thierry/Documents/R/Scripts/teaching/belief/DTW_synthetic_control.RData")
fit.mds<-mds(D,ndim=3,type="ratio")
pairs(fit.mds$conf,pch=ytrue,col=ytrue)
```

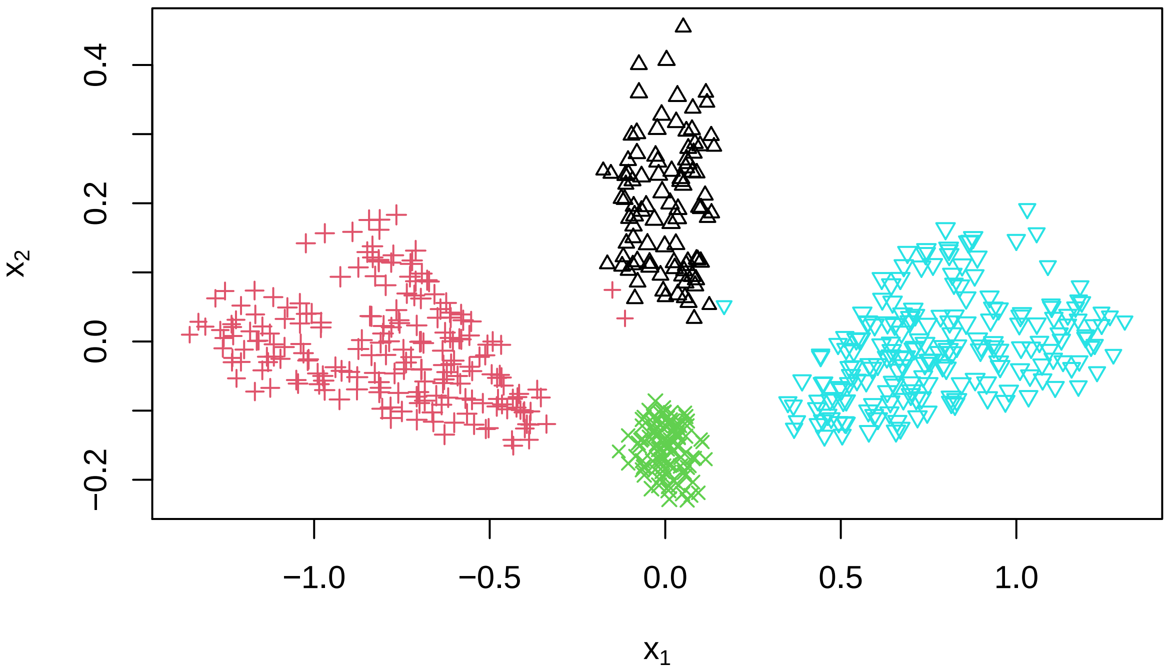


In the subspace (1,3), Classes 3/5 and 4/6 are now better separated, and class 2 appears as a more compact cluster (with some instances still mixed with class 1).

Using EVCLUS

Without constraints

```
library(evclust)
clus0<-kevclus(D=D,c=6,ntrials=1,disp=FALSE)
plot(clus0,X=fit.mds$conf[,c(1,3)],plot_approx=FALSE)
```



EVCLUS finds only 4 clusters: it does not disambiguate classes 3/5 and 4/6. Let us compute the Rand index:

```
library(clusterCrit)
R0<-extCriteria(as.integer(ytrue), clus0$y.pl, crit="Rand")
print(R0)
```

```
## $rand
## [1] 0.883734
```

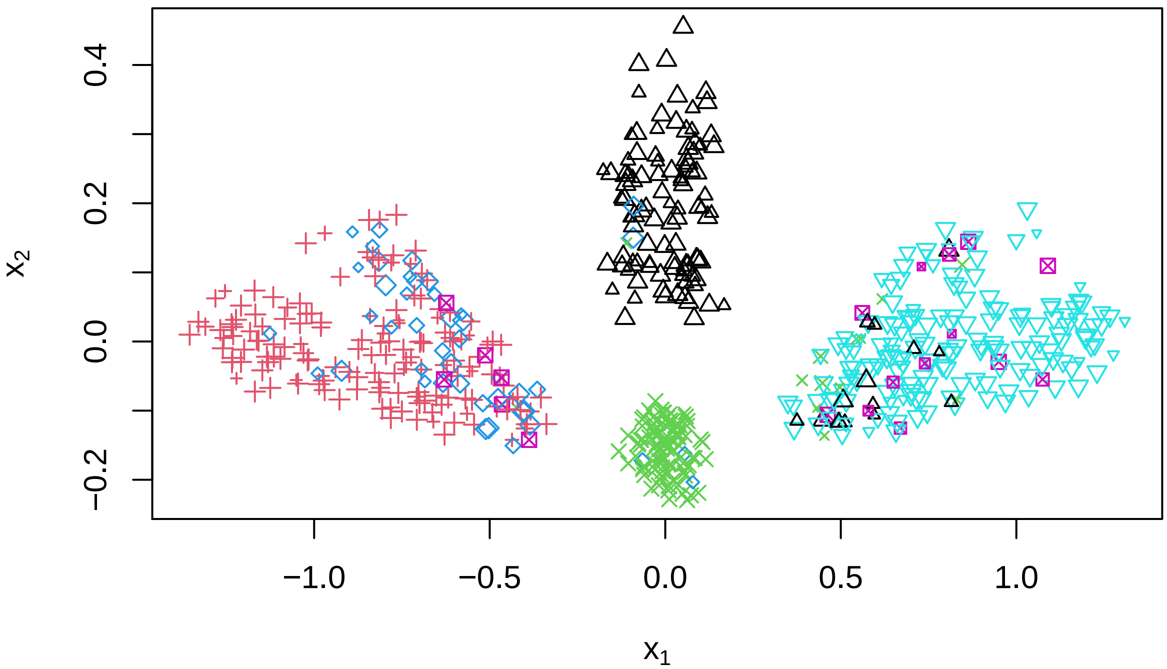
With pairwise constraints

Let us generate 2000 pairwise constraints:

```
set.seed(123)
const<-create_MLCL(ytrue, nbConst=2000)
```

We can then use function `kcevclus`:

```
clus1<-kcevclus(D=D,c=6,disp=FALSE,ML=const$ML,CL=const$CL,m0=clus0$mass,xi=0.1)
clus2<-kcevclus(D=D,c=6,disp=FALSE,ML=const$ML,CL=const$CL,m0=clus1$mass,xi=1)
plot(clus2,X=fit.mds$conf[,c(1,3)],plot_approx=FALSE)
```

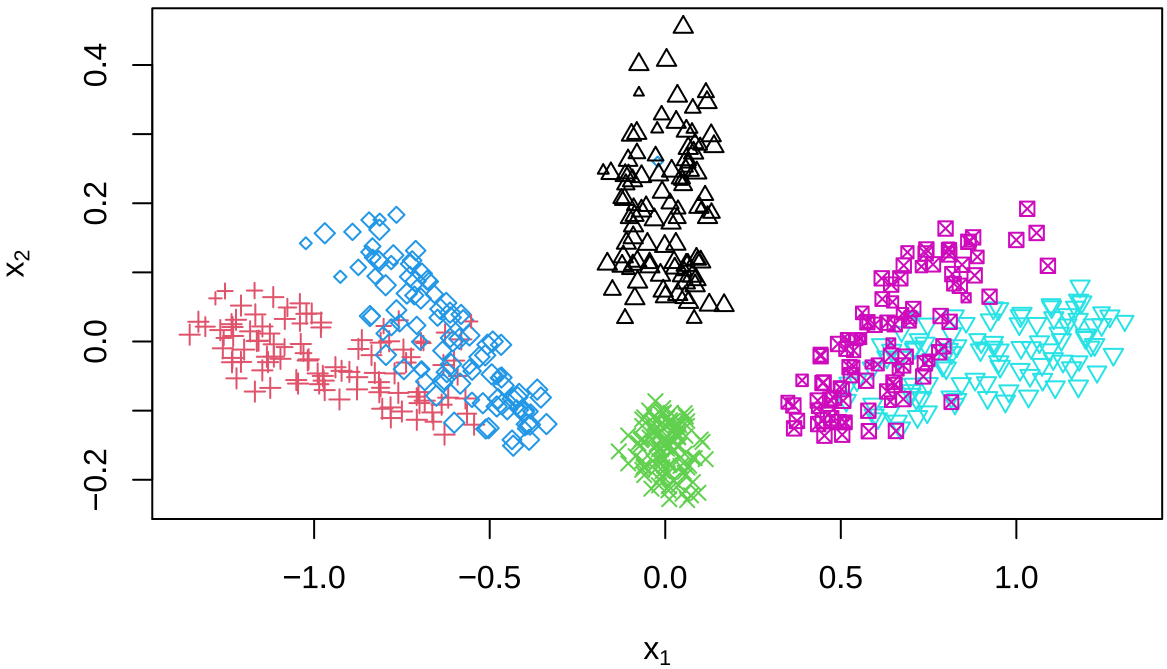



```
R1<-extCriteria(as.integer(ytrue), clus2$y.pl, crit="Rand")
print(c(R0,R1))
```

```
## $rand
## [1] 0.883734
##
## $rand
## [1] 0.8786032
```

The results have not improved. Let us now expand the constraints with $K = 5$ neighbors:

```
nn<-knn_dist(D, K=5)
const1<-expandlink(const,nn$nn.index,nn$nn.dist)
clus3<-kcevclus(D=D,c=6,ntrials=1,disp=FALSE,ML=const1$ML,CL=const1$CL,m0=clus0$mass,xi=0.1)
clus4<-kcevclus(D=D,c=6,ntrials=1,disp=FALSE,ML=const1$ML,CL=const1$CL,m0=clus0$mass,xi=1)
plot(clus4,X=fit.mds$conf[,c(1,3)],plot_approx=FALSE)
```




```
R2<-extCriteria(as.integer(ytrue), clus4$y.pl, crit="Rand")
print(c(R0,R1,R2))
```

```
## $rand
## [1] 0.883734
##
## $rand
## [1] 0.8786032
##
## $rand
## [1] 0.9966834
```

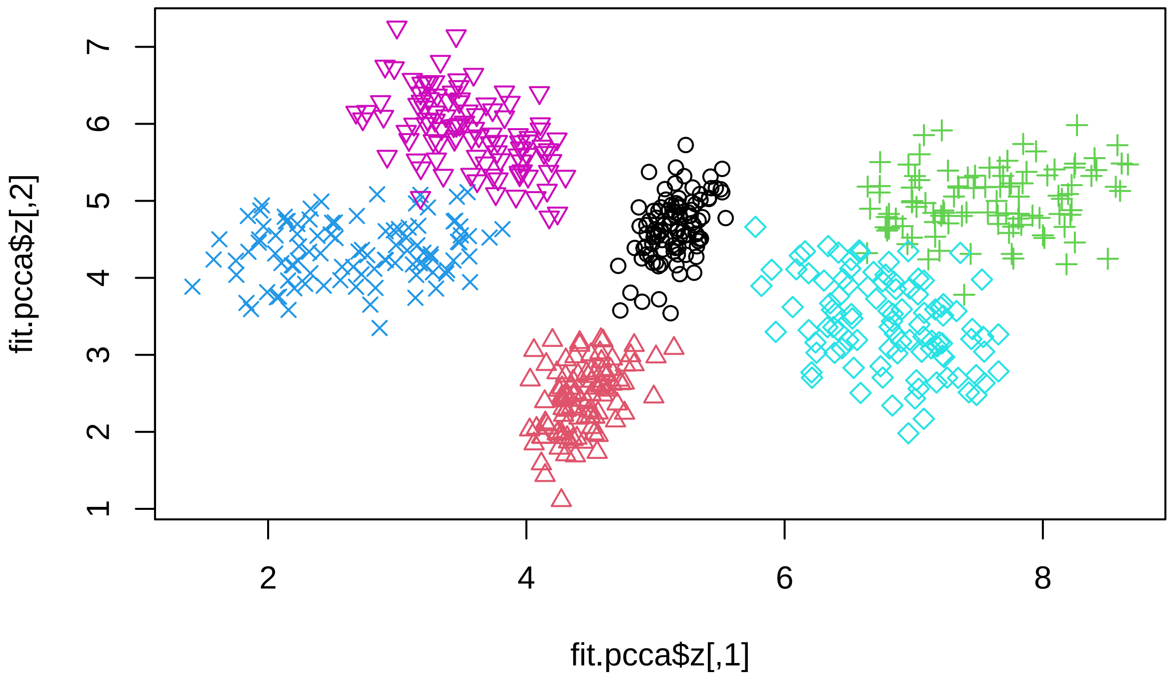
This time, we obtain a hard partition very close to the true one.

Using NN-EVCLUS

Using pairwise constraints

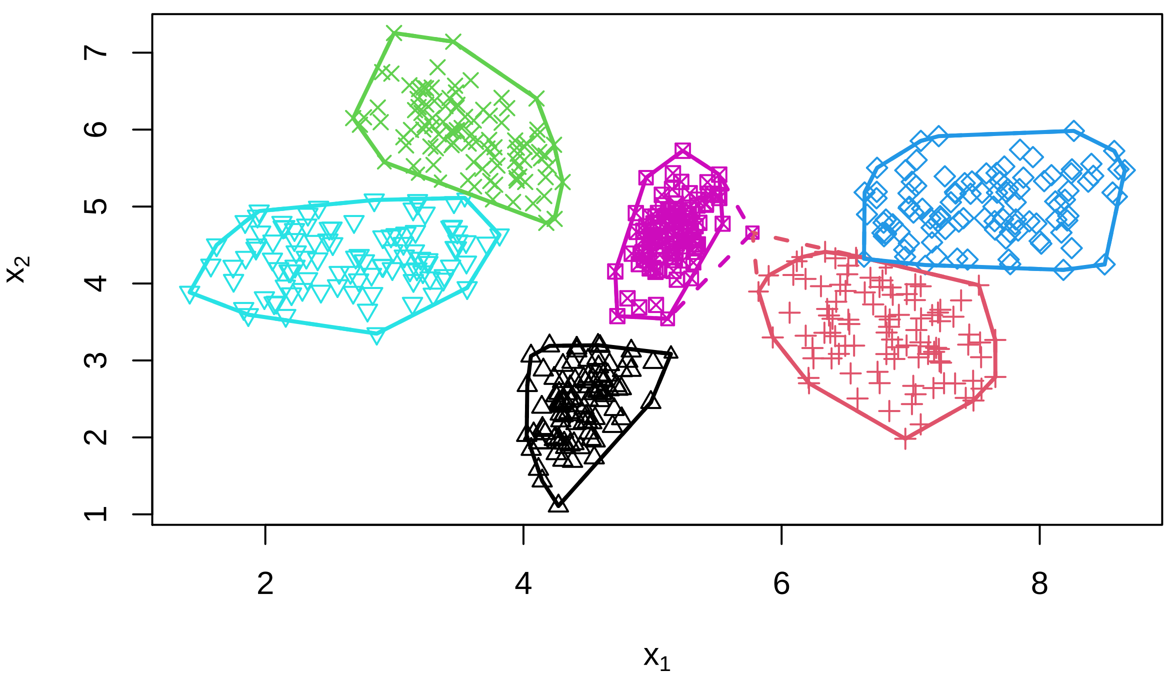
Let us now use NN-EVCLUS with the pairwise constraints. For that, we need to provide attributes. We will use PCCA, which linear transform the data using the pairwise constraints:

```
#fit.pcca<-pcca(x, d1=2, const1$ML, const1$CL)
#save(fit.pcca,file="pcca_synthetic_conrol.RData")
# The result is saved in a file to avoid recomputing it at each compilation
load(file="/Users/Thierry/Documents/R/Scripts/teaching/belief/pcca_synthetic_conrol.RData")
plot(fit.pcca$z,col=ytrue,pch=ytrue)
```



We can see that the classes are now much better separated, thanks to the pairwise constraints. Let us run function `nnevclus`:

```
set.seed(20220902)
clus5<-nnevclus(x=scale(fit.pcca$z),c=6,n_H=15,ML=const1$ML,CL=const1$CL,xi=1,
                options = c(0, 1000, 1e-04, 10))
plot(clus5,X=fit.pcca$z,plot_approx=TRUE)
```




```
R3<-extCriteria(as.integer(ytrue), clus5$y.pl, crit="Rand")
print(c(R0,R1,R2,R3))
```

```
## $rand
```

```
## [1] 0.883734
##
## $rand
## [1] 0.8786032
##
## $rand
## [1] 0.9966834
##
## $rand
## [1] 0.9945131
```

We obtain a similar result as that provided by EVCLUS. Howver, we now have a mapping from the 2D attribute space to mass functions.