2.5 Multivariate Curve Resolution (MCR)

Lecturer: Dr. Lionel Blanchet

The Multivariate Curve Resolution (MCR) methods are widely used in the analysis of mixtures in chemistry and biology. The main interest of this method is to perform a matrix decomposition allowing obtaining the individual characteristics (spectra, chromatograms …) of the chemical compounds of mixtures. Although MCR was mainly developed and used in Matlab, the main functions have been transposed in *R* into the *ALS* package.

Exercise 2.5.1 Load the "NIRdata" and plot the data set

```
> data(NIRdata)
> matplot(wv, t(NIRdata), type="l", lty=1)
```

The data set is here composed of 2000 variables. The 30 near infrared spectra represent 30 mixtures of chemical compounds. The vector "wv" contains the corresponding wavelengths. Do you observe any evolution in the spectral shape between the different samples? Are you able using this graphical representation to imagine the number and the shapes of the spectra of the pure chemical compounds forming the mixture?

Exercise 2.5.2 Perform exploratory analysis

In the previous chapter, you have performed Principal Component Analysis (PCA) on multiple examples. Use the same approach to visualize the "NIRdata".

> NIRdata.mc <- scale(NIRdata, scale=FALSE)

> PC.NIRdata.mc <- PCA(NIRdata.mc)

> scoreplot(PC.NIRdata.mc, main = "Scores")

> loadingplot(PC.NIRdata.mc, main = "Loadings")

Look at the score plot obtained. Do you visualize a logical evolution between the samples? From this observation are able to estimate the number of chemical compounds present in the mixture?

The score plot provides you information about the different samples. The information corresponding to variables is also of interest in this problem. Visualize the loading plot as you learn in chapter 2. Are you able to determine the chemical rank of this data set (i.e. number of compounds present)?

The resulting plot is difficult to read. Two reasons can explain it. First, the number of variables is very high (2000 variables), so the loading plot is too crowded. Second, a lot of variables are colinear: the different wavelengths belonging to the same peak appear grouped on the loading plot. However, the information about variables may be visualized in a more usual way for a chemist: as a spectrum.

> matplot(wv, loadings(PC.NIRdata.mc)[,1:2], type="l", lty=1)

Can you identify the most important peaks for the PCA model? Are you able to interpret these loadings as spectra? In such a “toy-example”, the interpretation is still doable. However, this task remains quite difficult and non intuitive for most of the chemists or biologists.

One of the objectives of MCR is to obtain more readable results. To do so the main constraint of PCA is removed: the orthogonality of the different components. Indeed two spectra of two pure chemical compounds can share multiple features and are therefore non orthogonal. The objective is to obtain an interpretable solution directly after

statistical analysis. The components extracted must therefore stay within a meaningful "chemical subspace".

Exercise 2.5.3 Construct an initial guess of the pure spectra

The first approach in MCR is trying to estimate the pure spectra directly from the data. This is the approach proposed in SIMPLISMA (SIMPLe to use Interactive Self-Modelling Algorithm). The idea is to find the most different spectra within the dataset and assume that they are the most pure ones.

> simplisma(t(NIRdata), 2, 0.1)

Apply SIMPLISMA to the "NIRdata" for two components. Observe the resulting estimation of the pure spectra. Do they look like NIR spectra?

> simplisma(t(NIRdata), 3, 0.1)

Redo the SIMPLSIMA to the "NIRdata" but for three components. Observe the resulting estimation of the pure spectra. Do they still they look like NIR spectra? Do the two first components resemble the ones obtained before?

Exercise 2.5.4 Construct an initial guess of the concentration profiles

In the previous exercise pure spectra were estimated. One could be more interested in the evolution of the compound concentrations in the mixture. One possible approach is to apply SIMPLISMA on the transposed NIRdata. (You can do it as an extra exercise.)

A second approach aims to detect and characterize evolution in the data set. This is done using Evolving Factor Analysis (EFA). As previously the calculation can be done using different chemical rank. Apply EFA for 2 and 3 components.

> efa(NIRdata, nf=2, plot=1)
> efa(NIRdata, nf=3, plot=1)

Can you interpret the results obtained? Does it make sense? How comparable are the two results obtained?

Exercise 2.5.5 Unconstrained Multivariate Curve Resolution

The two precedent methods only use explicitly one of the two directions available in the NIRdata: the concentrations or the spectra. The algorithm MCR ALS (Multivariate Curve Resolution by Alternating Least Squares) is using both directions alternatively until it converges to a stable solution.

Use the initial estimates for EFA as initial guess for MCR-ALS. Calculate the best decomposition for 2 and 3 components without constraints.

> bla2 <- efa(NIRdata, nf=2)
> test0 <- als(CList=list(bla2$E), PsiList=list(NIRdata), S=matrix(1, nrow=2000, ncol=2), nonnegC=FALSE, nonnegS=FALSE, normS=2, x=wv)
> matplot(test0$CList[[1]], type="l")
> x11()
> matplot(test0$S, type="l")
> bla3 <- efa(NIRdata, nf=3)
> test1 <- als(CList=list(bla3$E), PsiList=list(NIRdata), S=matrix(1,nrow=2000 ,ncol=3), nonnegC=FALSE, nonnegS=FALSE, normS=2, x=wv)

> x11()

> matplot(test1$CList[[1]], type="l")

> x11()

> matplot(test1$S, type="l")

Compare the results first graphically and then using the statistical criterions R^2 and lack of fit (lof). Which solution appears to be the best? What would you conclude about the chemical rank of this data set?

Exercise 2.5.6 Effect of a wrong initial guess and constrained MCR-ALS

We will repeat the previous example using different estimates. The calculations produce different results. Let's try using a bad starting point. Load the initial guess IG_wrong, and use it in MCR-ALS.

> test0 <- als(CList=list(IG_wrong), PsiList=list(NIRdata), S=matrix(1,nrow=2000, ncol=3), nonnegC=FALSE, nonnegS=FALSE, normS=2, x=wv)

> x11()

> matplot(test0$CList[[1]], type="l")

> x11()

> matplot(test0$S, type="l")

What can you conclude about the results?

Now have a look at the initial guess you used.

> matplot(IG_wrong, type="l")

As you can see one component is entirely negative, which means that this component is really far from a real chemical contribution. Try now to use again IG_wrong in MCR-ALS but this time using the non negativity constraint.

```
> test1 <- als(CList=list(IG_wrong), PsiList=list(NIRdata), S=matrix(1,nrow=2000,
ncol=3), nonnegC=TRUE, nonnegS=FALSE, normS=2, x=wv)
>  x11()
> matplot(test1$CList[[1]], type="l")
> x11()
> matplot(test1$S, type="l")
```

Is the constrained result better than the previous one? How do you explain it?

Extra exercise (can be executed as exercise 2.5.7). The intensity ambiguities

The drawback of the loss of orthogonality (compare to PCA) is that multiple solutions are possible for one data set, i.e. multiple sets of components equally good in terms of statistics but leading to different interpretations.

The simplest problem is the called “intensity ambiguities”. The interpretation of the results can be troubled by this effect but in most of the case it is easy to detect.
Let’s multiply the concentration output of the exercise 2.5.5 for three components by a vector [1 5 1].

```
> bla3 <- efa(NIRdata, nf=3)
> test1 <- als(CList=list(bla3$E), PsiList=list(NIRdata), S=matrix(1,nrow=2000,
ncol=3), nonnegC=FALSE, nonnegS=FALSE, normS=2, x=wv)
> C <- test1$CList[[1]]
> C[,2] <- C[,2] * 5
```

> S <- test1$S

> S[,2] <- S[,2] / 5

The spectra must be divided by the same vector. The matrix of residuals remains the same i.e. the new model is as good as the previous one (same R^2 and lof).

Now look at the concentration profile. The results look quite different with an higher concentration of one of the compounds. However if you look at the spectra it is easy to detect a scale problem.

> x11()

> matplot(C, type="l")

> x11()

> matplot(S, type="l")

Extra exercise (can be executed as exercise 2.5.8). The rotation ambiguities

The second problem is more complex. Instead of multiplying (or dividing) the different components by a scalar, the model is now multiply (or divided) by a rotation matrix. Use the matrix “m_rotation” to transform the MCR model form the exercise 3.3.1 (with 3 components)

> bla3 <- efa(NIRdata, nf=3)

> test1 <- als(CList=list(bla3$E), PsiList=list(NIRdata), S=matrix(1,nrow=2000, ncol=3), nonnegC=FALSE, nonnegS=FALSE, normS=2, x=wv)

> rot.mat <- matrix(c(0.36,0.48,-0.8,-0.8,0.6,0,0.48,0.64,0.6), ncol=3, nrow=3, byrow=TRUE)

> C <- test1$CList[[1]]

> C[,2] <- C[,2] %% rot.mat*

```
> S <- test1$S
> S[,2] <- S[,2] %*% ginv(rot.mat)
> x11()
> matplot(C, type="l")
> x11()
> matplot(S, type="l")
```

Again you can check the concentration and the spectra. In this case you observed a strong effect. Do you think that the use of constraint could help to limit or prohibit this effect completely?
