

SOFTWARE INSTRUCTIONS

The instructions do not include explanations about the chemometric methods on which the programs are based, although this information can be obtained reading the basic references given at the end of this document.

Please let us know any comment on the clarity of the instructions or on possible bugs of the programs. We greatly appreciate the feedback of the users. You may write to the address annaj@apolo.qui.ub.es for any query related to this section.

Exploratory analysis

- ◆ pure.m Selection of purest variables based on SIMPLISMA.
- ◆ efa.m Evolving Factor Analysis (EFA)
- ◆ efard.m Evolving Factor Analysis for rank-deficient data sets.
- ◆ eff.m Fixed Size Moving Window- Evolving Factor Analysis (FSMW-EFA)

Resolution

2- and 3-way

- ◆ als.m Multivariate Curve Resolution- Alternating Least Squares (MCR-ALS)

Exploratory analysis

◆ **Pure.m**

Selects the purest column variables in a data set based on the SIMPLISMA method.

Command line

```
» [sp, imp]=pure(d, nr, f)
```

Input

d: data set

nr: number of purest variables to be selected.

f: % of noise in the data set (with respect to maximum absorbance in mean spectrum).
Offset of purity.

Output

sp: purest column profiles.

imp: index of purest column variables.

Graphical output

- Figure with column means, column std. Deviations and first pure spectrum of column variables.
- Figures with pure spectrum and 'purity corrected' standard deviation spectrum related to each variable selection step.

◆ efa.m

Performs EFA along the row direction of a matrix, i.e., the windows are built increasing gradually the number of rows taken.

Command line

```
» [e,eforward,ebackward]=efa(d,ns)
```

Input

d: data set

ns: number of rows of the data matrix to be included in the analysis

Output

e: initial estimates (approximate process profiles) built from the combined forward and backward EFA plot (in eigenvalue scale).

eforward: results from forward EFA (in eigenvalue scale).

ebackward: results from backward EFA (in eigenvalue scale).

Execution queries

```
Press any key to select the min value of the log(eig) EFA plot  
min. value of log efa plots ?
```

This allows the user to set the minimum value in log(eigenvalue) scale to avoid large noise zones to be plotted.

```
Press any key to build the initial estimates from EFA results  
Number of factors to be considered:?
```

From the visual inspection of the combined EFA plot, the user can decide the number of compounds in the system and, therefore, the number of process profiles to be built.

```
Other num. of factors to be considered:? (y/n)
```

If the answer is 'y', a plot with a different number of generated process profiles can be visualised. If the answer is 'n', the process profiles displayed in the last plot are stored as initial estimates (**e**).

Graphical output

- Figure with separate forward and backward EFA subplots.
- Figure with combined forward and backward EFA subplots (in singular value scale and log(eigenvalue) scale). Black lines: forward EFA. Red lines: backward EFA.
- Combined EFA plot with log(eigenvalue) scale set by the user.
- Plot of EFA derived initial estimates.

◆ **efard.m**

Performs EFA on full rank matrices obtained by matrix augmentation of rank-deficient systems. It follows two steps:

- a) classical EFA on each individual full-rank matrix.
- b) adapted EFA on the full rank augmented data set to obtain local rank information from the rank-deficient data matrix.

Command line

```
» [efullrank,effullrank,ebfullrank,erd,efrd,ebrd]=efard(d,nmat,nrow,frank)
```

Input

d: original column-wise augmented data set. It must contain one or more full-rank matrices on top and only one rank-deficient matrix below.

nmat: total number of matrices appended in the column-wise augmented data set

nrow: row vector that contains the number of rows of each matrix in **d**. e.g., for two matrices with 20 and 34 rows, `nrow = [20 34]`.

frank: row vector that tells whether each matrix in the data set is full-rank (1) or rank-deficient (0). For the previous example, it would be: `[1 0]`. (note that the last element should always be zero).

Output

efullrank: initial estimates derived from EFA for each full rank matrix in the data set. When there is more than one full rank matrix, `efullrank` is a cell array with as many elements as full rank matrices used in the augmentation of the rank-deficient data set.

effullrank: matrix of eigenvalues from the forward EFA performed on each full rank matrix. When there is more than one full rank matrix, `effullrank` is a cell array with as many elements as full rank matrices.

ebfullrank: matrix of eigenvalues from the backward EFA performed on each full rank matrix. When there is more than one full rank matrix, `ebfullrank` is a cell array with as many elements as full rank matrices.

erd: initial estimates derived from EFA on the full rank data set obtained by matrix augmentation. The only contributions stored are related to the compounds in the rank-deficient matrix that are absent in the appended full-rank matrices.

efrd: matrix of eigenvalues from the forward EFA associated with the contributions in the rank-deficient matrix absent in matrices used for augmentation.

ebrd: matrix of eigenvalues from the backward EFA associated with the contributions in the rank-deficient matrix absent in matrices used for augmentation.

Execution queries

a) On the EFA analysis of the full rank matrices used for augmentation.

As in the **efa.m** program.

b) On the EFA of the column-wise augmented matrix.

After having shown the forward and backward EFA plots for the column-wise augmented matrix, we are asked:

```
>> Total number of components in full-rank matrices?
```

From the visual inspection of these plots, we must answer here the number of significant contributions appearing in the full rank matrices used for augmentation, i.e., those on the left hand side of the vertical line in the forward EFA and in the right hand side on the backward EFA.

The next figures will show the EFA plots related to the contributions exclusively present in the rank-deficient matrix in the data set.

On these plots, the queries are as in the routine **efa.m**.

Graphical output

For each full rank matrix in the augmented data set

- Figure with separate forward and backward EFA subplots.
- Figure with combined forward and backward EFA subplots (in singular value scale and $\log(\text{eigenvalue})$ scale). Solid black lines: forward EFA. Dashed black lines: backward EFA.
- Combined EFA plot with $\log(\text{eigenvalue})$ scale set by the user.
- Plot of EFA derived initial estimates.

For the column-wise augmented data set

- Forward EFA plot on the column-wise augmented data matrix (in singular value scale and $\log(\text{eigenvalue})$ scale). The vertical line separates the information related to the rank-deficient data set (right) and the full-rank matrices used in augmentation (left).
- Backward EFA plot on the column-wise augmented data matrix (in singular value scale and $\log(\text{eigenvalue})$ scale). The vertical line separates the information related to the rank-deficient data set (left) and the full-rank matrices used in augmentation (right).

For the rank-deficient matrix in the data set

- Combined forward (solid lines) and backward (dashed lines) EFA plot displaying the contributions exclusively linked to the rank-deficient matrix (in singular value and $\log(\text{eigenvalue})$ scale).
- Combined EFA plot with $\log(\text{eigenvalue})$ scale set by the user.
- Plot of EFA estimates linked to the contributions only present in the rank-deficient matrix.

◆ **eff.m**

Performs FSMW-EFA along the row direction of a matrix, i.e., the windows are built taking as many rows as indicated by the window size selected by the user.

Command line

```
» efeig=eff(d,ws,nr)
```

Input

d: data matrix

ws: window size

nr: number of rows of the data matrix to be included in the analysis

Output

efeig: results from FSMW-EFA (in eigenvalue scale).

Graphical output

- Figure with FSMW-EFA plot (in singular value scale).
- Figure with FSMW-EFA plot (in log(eigenvalue) scale).

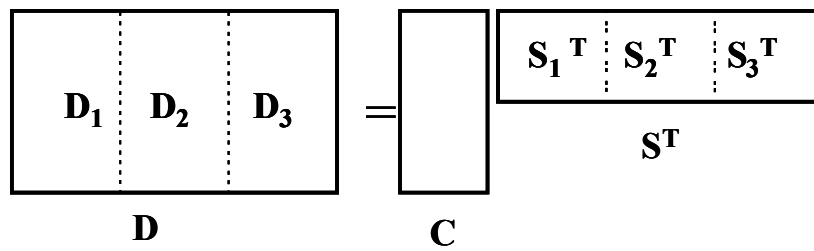
Resolution

2- and 3-way

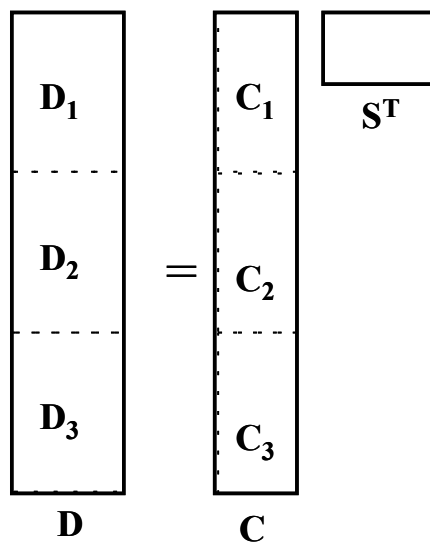
◆ als.m

Performs Multivariate Curve Resolution- Alternating Least Squares (MCR-ALS) in two- or three-way data sets (see below for different three-way examples).

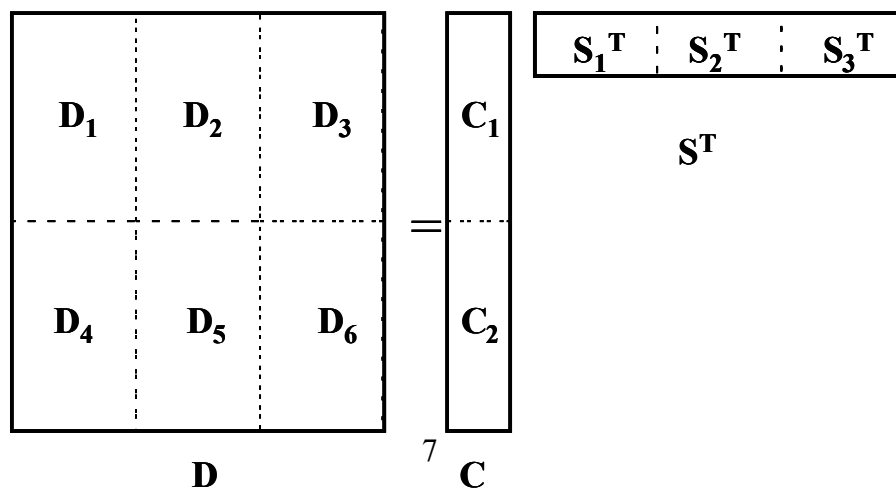
a)



b)



c)



Command line

```
» [copt,sopt,sdopt,ropt,areaopt,rtopt]=  
als(d,x0,nexp,nit,tolsigma,isp,csel,sse1,vclos1,vclos2);
```

Input

1. **d**. This is the name of your data set, sized ($r \times c$), where r are the number of rows and c the number of columns.

As we have seen, this **d** can be either a single data matrix or a three-way data set. We will take the three-way data sets in Figure 2 as examples to show the MATLAB notation that should be used in each case.

Single matrix. **d = D**

F. a) Row-wise augmented matrix. **d = [D₁ D₂ D₃]**

b) Column-wise augmented data matrix. **d = [D₁;D₂;D₃]**

c) Row-and column-wise augmented data matrix. **d = [D₁ D₂ D₃;D₄ D₅ D₆]**

*it is a common practice that the rows of the data set contain the instrumental responses collected during the experiment(s) (e.g., spectra, voltammograms,...) and the columns the information related to the variation of the concentration, so that the **C** matrix obtained relates to concentration profiles and the **S** matrix to spectra. Nevertheless, the program admits working in the opposite way.*

2. **x0**. This is the matrix of your initial estimates (starting concentration or spectral profiles for the iterative optimization process) which may be an approximation of the **C** matrix or of the **S** matrix. As in the case of your data set **d**, **x0** can be a single or an augmented matrix depending on the data set it comes from. If your initial estimates are concentration profiles, **x0** will be sized ($r \times ns$). If your initial estimates are spectra, **x0** is sized ($ns \times c$). ns is the number of components (pure contributions) you want to resolve in your data set.

Going to the examples in figure 1 and figure 2, **x0** would be:

Single matrix.

Conc. initial estimates. **x0 = C_{in}**

Spectra estimates. **x0 = S_{in}**

F.2. a) Row-wise augmented matrix.

Conc. initial estimates. **x0 = C_{in}**

Spectra estimates. **x0 = [S_{1in} S_{2in} S_{3in}]**

b) Column-wise augmented data matrix.

Conc. initial estimates. **x0 = [C_{1in};C_{2in};C_{3in}]**

Spectra estimates. **x0 = S_{in}**

c) Row-and column-wise augmented data matrix.

Conc. initial estimates. **x0 = [C_{1in};C_{2in}]**

Spectra estimates. **x0 = [S_{1in} S_{2in} S_{3in}]**

*In its current implementation, the program may interpret correctly your **C**- or **S**-type matrices, even if you input them transposed. To stabilise the optimization process, it is recommendable that you input initial estimates (based on chemical preknowledge or obtained by chemometric means) that be as close as possible to the real solutions expected, with either **C** or **S** profiles that fulfil the general properties (constraints) of the sought profiles, i.e., random initial estimates or PCA raw profiles should be avoided.*

3. nexp. Number of matrices forming the data set. This always equals (nr. of submatrices in **C** (**C_i**) x nr. of submatrices in **S** (**S_i**)) (default is 1).

Single matrix. **nexp** = 1

- F. . a) Row-wise augmented matrix. **nexp** = 3
 b) Column-wise augmented data matrix. **nexp** = 3
 c) Row-and column-wise augmented data matrix. **nexp** = 6

4. nit. Maximum number of iterations in the optimization step (default is 50).

5. tolsigma. Convergence criterion based on the relative change of lack of fit during the optimization between consecutive iterations (default is 0.1%).

6. isp. Small binary matrix containing the information related to the correspondence of species (components) among the matrices present in the data set. We use the binary notation to express the presence or absence of a species (1: present species; 0: absent species). The size of an isp matrix is always (nr. of **C_i** submatrices × ns). ns is in this case the total number of species in the data set.

According to the size of the isp matrix, we see that setting this matrix is very easy for single matrices and row-wise augmented matrices. Let's imagine a system with three components.

Single matrix. **isp** = [1,1,1]

- F. a) Row-wise augmented matrix. **isp** = [1,1,1]

Now, we show two possible isp matrices for examples in F. b) and c).

F. b) Column-wise augmented data matrix. Let's imagine that we have a system with 2 components in total, A and B. In **C₁**, only A is present; in **C₂**, only B is present and in **C₃**, A and B. **isp** = [1 0;0 1;1 1]

$$\begin{array}{c} \text{A B} \\ \text{C}_1 \begin{pmatrix} 1 & 0 \end{pmatrix} \\ \text{C}_2 \begin{pmatrix} 0 & 1 \end{pmatrix} \\ \text{C}_3 \begin{pmatrix} 1 & 1 \end{pmatrix} \end{array}$$

F. c) Row- and column-wise augmented data matrix. Let's imagine that we have a system with 3 components in total, A, B and C. In **C₁**, A and B are present and in **C₂**, A and C. **isp**=[1 1 0;1 0 1]

$$\begin{array}{c} \\ C_1 \\ C_2 \end{array} \begin{array}{ccc} A & B & C \\ \left(\begin{array}{ccc} 1 & 1 & 0 \\ 1 & 0 & 1 \end{array} \right) \end{array}$$

In systems like those in F. 2 b) and F. 2 c) that require more complex isp matrices, it may happen that we do not have the information related to the species present in each C_i . This problem is often solved looking at the results coming from the individual resolution of the matrices forming the data set. If, even so, we still have doubts upon this point, we may use an isp with no null elements in the first resolution attempt and modify it in next resolution trials according to the results obtained.

7. **csel** and **sstel**. Those matrices have analogous meaning and relate to the **C** and the **S** matrix, respectively. They are used to introduce information available about **C** and **S** in the resolution process, i.e. the elements of these matrices that are exactly (or at least, very approximately) known beforehand. **csel** and **sstel** are always sized as **C** and **S**, respectively. When an element in **C** or in **S** is unknown, the related element in **csel** or **sstel** can be either 'Inf' or 'NaN' (in MATLAB notation, *Inf* holds for infinite and *NaN* for Not-a-number); when the element is known, it appears with its true value in **csel** or **sstel**. Finite values in **csel** and **sstel** remain invariant in the resulting **C** and **S**.

Let's see an example of a **csel** matrix.

	A	B	C	
Selective region for compound C	0	0	1.32	Known profile
	0	0	1.28	
	NaN	NaN	1.26	
	NaN	NaN	1.24	
	NaN	NaN	1.22	
	NaN	NaN	1.04	
	NaN	NaN	1.04	
	NaN	NaN	1.04	
	NaN	NaN	1.04	
	NaN	NaN	1.04	
Local rank information (absence of compound B)	NaN	0	1.04	
	NaN	0	1.04	
	NaN	0	1.04	
	NaN	0	1.04	

As you can see, the known information can have different origins. We may know the pure profile of a certain species (a concentration profile or a spectrum, for instance). In a local rank analysis of the data set, we may have detected selective regions of our data set in the concentration and in/or the spectral direction: in this case, what we know is that the rest of the species are absent and, therefore, the values of their related elements should be zero. It may also happen that we know about the absence of a certain species in a certain region of **C** or **S**: then, their elements can be set to zero.

8. **vclos1** and **vclos2**. These input parameters are only used when we deal with certain cases of closed systems. In a closed system, the closure constraint imposes the fulfilment of a mass balance equation for one or more components along a process or reaction. In real cases, though, we may have closed systems, whose total concentration varies along the process in a perfectly known way (e.g., because of the dilution caused by the addition of a certain measured amount of a reagent). **vclos1** is a vector whose elements indicate the value of the total concentration at each stage of the process (for each row of **C** matrix). When we have two closure conditions in the same data set (e.g., two independent mass balance equations) we may use **vclos1** and **vclos2** to refer to the two different closed systems.

GENERAL INPUT INFORMATION

- For two-way data sets, the compulsory input parameters are only *d* and *xo*. For three-way data sets, *nexp* and *isp* should also be added. The rest are optional and, if needed, the program uses default values.
- *Default values for some input parameters.* **nexp** = 1; **nit** = 50; **tolsigma** = 0.1;
- When an input parameter is skipped (we do not need it or we use the default values) and we want to introduce a parameter which is present afterwards in the input sequence, we should fill the place of the skipped parameter with a [] symbol to preserve the right correspondence between the input information and the meaning of the related parameters. [] symbols are not needed for all ignored input parameters after the last introduced.

Output

Essentially, the output of the resolution program consists of matrices of pure concentration profiles and spectra (augmented or not, depending on the architecture of the data set) and some figures of merit related to the model fit obtained with the pure resolved profiles. For column-wise and row- and column-wise data matrices, some other parameters provide additional information for quantitative purposes.

1. **copt**. Matrix of resolved pure concentration profiles (single or augmented)
2. **sopt**. Matrix of pure spectra (single or augmented).
3. **sdopt**. Optimal percent of lack of fit in relative standard deviation units.

$$\% \text{ lack of fit} = 100 \times \sqrt{\frac{\sum_{i,j} e_{ij}^2}{\sum_{i,j} d_{ij}^2}}$$

where e_{ij} are the residuals obtained in the data set reproduction using the pure profiles in **copt** and **sopt** matrices and d_{ij} are the elements of the data set to be reproduced. This output parameter contains two figures.

sdopt = [a b]

a → compares the matrix obtained from the resolution results, $\mathbf{d}^* = \mathbf{copt} * \mathbf{sopt}$, with the matrix obtained by PCA reproduction using the same number of components as the raw data set has (\mathbf{d}_{PCA}). Therefore,

$$e_{ij} = d_{\text{PCA } ij} - d^*_{ij}$$

$$d_{ij} = d_{\text{PCA } ij}$$

b → compares the matrix obtained from the resolution results, $\mathbf{d}^* = \mathbf{copt} * \mathbf{sopt}$, with the raw data set (\mathbf{d}). Therefore,

$$e_{ij} = d_{ij} - d^*_{ij}$$

$$d_{ij} = d_{ij}$$

In low-noise systems where all possible contributions can be described with a bilinear model, these two figures should be similar. Differences may arise from the presence of structured noise or from the presence of unmodellable contributions (minor or from non-chemical sources). Intermediate situations could be present for large amounts of white noise or for other problems like non-linearities.

4. **ropt**. Matrix of residuals obtained from the comparison of the PCA reproduced data set (\mathbf{d}_{PCA}) using the pure resolved concentration and spectra profiles.

$$\mathbf{ropt} = \mathbf{d}_{\text{PCA}} - (\mathbf{copt} * \mathbf{sopt})$$

5. **areaopt**. This matrix is sized as the isp matrix and contains the area under the concentration profile of each species in each \mathbf{C}_i matrix. Let's imagine the following isp matrix:

$$\text{isp} = \begin{matrix} & \begin{matrix} A & B \end{matrix} \\ \begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} \end{matrix}$$

A possible **areaopt** matrix could be as follows:

$$\text{areaopt} = \begin{matrix} & \begin{matrix} A & B \end{matrix} \\ \begin{pmatrix} 1.05 & 3.40 \\ 0 & 2.56 \\ 1.30 & 0 \end{pmatrix} \end{matrix}$$

In this case, we may easily recognise a data set whose first matrix is a mixture (A + B) and the second and the third, standards of B and A, respectively. If external information about

the concentration of the standards is available, we may use the information in **areaopt** for quantitative purposes (e.g., in a univariate calibration model).

6. **rtopt**. Matrix providing relative quantitative information. **rtopt** is a matrix of area ratios between components (species) in different matrices. The first data matrix is always taken as a reference. Sticking to the example presented in parameter nr. 5 (**areaopt**), the related **rtopt** matrix would be:

$$\text{rtopt} = \begin{pmatrix} & \text{A} & \text{B} \\ \text{A} & 1.05/1.05 & 3.40/3.40 \\ \text{B} & 0/1.05 & 2.56/3.40 \\ & 1.30/1.05 & 0/3.40 \end{pmatrix} \quad \text{rtopt} = \begin{pmatrix} & \text{A} & \text{B} \\ \text{A} & 1.00 & 1.00 \\ \text{B} & 0 & 0.75 \\ & 1.24 & 0 \end{pmatrix}$$

*Whenever possible, it is recommendable organizing the data set so that the first matrix contains all the components present in the data set to avoid the presence of undetermined values (Inf) in the data matrix. If not possible, this relative quantitative information can be externally calculated taking the suitable reference for each component. Note that the program considers the area under the concentration profiles as proportional to the concentration. If this is not your case, **rtopt** lacks chemical sense and you can use other information contained in the concentration profiles (e.g., peak height) to get quantitative information.*

Screen output

The program shows information on the resolution process during the iterative optimisation procedure.

```
PCA CPU time: 0.06
Number of components: 3
percent of lack of fit (PCA lof): 0.25983
Data reproduction with the considered number of species has an error
0.25983
```

First, details on the PCA model equal in size to the real resolution model are shown.

```
ITERATION 1
Sum of squares respect PCA reprod. = 1359.3465 (1)
Old sigma = 692715.8947 -----> New sigma = 0.52692 (2)
Sigma respect experimental data = 25.7288 (3)

FITING IS IMPROVING !!! (4)
Change in sigma (%) = 131465087.5808 (5)
Fitting error (lack of fit, lof) in % (PCA) = 0.0053224 (6)
Fitting error (lack of fit, lof) in % (exp) = 0.25989 (7)
```

Percent of variance explained (r2) is 99.9993 (8)

Per each iteration n , the following results are shown:

- (1) Sum of squares of residuals between the MCR-ALS reproduction and the PCA model.
- (2) Std. deviation of the residuals (MCR-ALS vs. PCA) for iteration $n-1$ (old sigma) and n (new sigma).
- (3) Std. deviation of the residuals (MCR-ALS vs. experimental data) for iteration n .
- (4) Diagnostic on the evolution of the fit based on comparison of iteration n and last converging iteration.
- (5) % of sigma change between iteration n and last converging iteration.
- (6) sdopt (1) for iteration n . See output section.
- (7) sdopt (2) for iteration n . See output section.
- (8) % of variance explained (calculated as in **parafac.m** routine, see p. 7)

CONVERGENCE IS ACHIEVED! (1)

Fitting error (lack of fit, lof) in % at the optimum =
0.0049391 (PCA) 0.25988 (exp) (2)

Percent of variance explained (r2) at the optimum is 99.9993 (3)

Relative species conc. areas respect matrix (sample) 1 at the optimum

1 1 1 (4)

Plots are at optimum in the iteration 3 (5)

The program ends summarising the evolution of the optimisation process and the final results obtained.

- (1) It is detailed whether the process ended successfully (convergence!), was stopped after a certain number of diverging iterations or was stopped because the maximum number of iterations was exceeded.
- (2) sdopt. (see output section).
- (3) % of variance explained (calculated as in **parafac.m** routine, see p. 7).
- (4) rtopt (see output section).
- (5) The iteration number at which the optimal results are obtained is shown.

Graphical output

- Subplots of raw data set (plot of rows and columns).
- Subplots of initial estimates (**C** or **S** matrix) and estimated unconstrained least-squares **S** or **C** matrices.
- Subplots of scores and loading profiles for the PCA model with a number of compounds equal to the resolution model.
- Subplots of resolved concentration and signal profiles at each iterative cycle.
- Subplots of optimal concentration and signal profiles.

Execution queries

The **als.m** program can be used to resolve different kinds of three-way data sets (single matrices or three-way data sets, see F. a), b) and c)). The constraints selected during the optimisation process can be diverse and be applied in different manners. All this information is asked during the execution of the MCR-ALS program.

1. Definition of the structure of the data set.

This is asked when we are working with a three-way data set.

CLASSIFICATION OF THREE-WAY DATA SETS

Column-wise augmented data matrix (1)

Row-wise augmented data matrix (2)

Column- and row-wise augmented data matrix (3)

'Define your data set '

(1), (2) and (3) refer to data sets in F. a), b) and c), respectively.

Since we are not forced to work with three-way data sets formed by equally sized matrices, we should specify:

a) In column-wise augmented data matrices, the number of rows that each of the matrices has (equal to the number of rows of the related C submatrix).

C matrix, submatrix number 1

Enter the number of rows:

b) In row-wise augmented data matrices, the number of columns that each of the matrices has (equal to the number of rows of the related S submatrix).

S matrix, submatrix number 1

Enter the number of columns:

c) In row- and column-wise augmented data matrices, the structure of the data set, i.e. how many matrices have been appended column-wisely (equal to the number of submatrices in C) and how many matrices have been appended row-wisely (equal to the number of submatrices in S).

How many submatrices has the C matrix?

How many submatrices has the S matrix?

As in a) and b), the size of each of the appended matrices should afterwards be defined (providing the size of their derived C and S submatrices).

C matrix, submatrix number 1
Enter the number of rows:

S matrix, submatrix number 1
Enter the number of columns:

2. Selection of constraints.

The constraints to be applied somewhere in the concentration and/or in the spectral direction should be specified.

INPUT TYPE OF CONSTRAINTS TO BE APPLIED

None (0)

Non-negativity (1)

Unimodality (2)

Closure (3)

Equality (known) / Lower than (selectivity) constraints in conc profiles (4)

Equality (known) / Lower than (selectivity) constraints in spectra profiles (5)

Shape constraints (6)

Three-way data structure constraints (7)

Enter a vector with the selected constraints, e.g. [1,3,5]

Enter the constraints to be applied in the optimization:

Here you have a brief description of each constraint. For more detailed information, go to related literature.

- *Non-negativity (1)*: forces the elements in a profile to be positive.
- *Unimodality (2)*: allows for the presence of only one maximum per profile.
- *Closure (3)*: fulfilment of a mass balance condition. The different profiles (compounds) involved in the closed system are simultaneously constrained.
- *Equality constraints for concentration and spectra (4,5)*: the elements from matrices **C** and **S** known beforehand replace those calculated during the optimization process. The selection of these constraints is necessary to activate the introduction of the information in the **csel** and **ssel** matrices, when present.
- *Shape constraints*: at present, under implementation. This makes reference to fit a profile according to a definite shape (either based on a physicochemical model or on a mathematical function).
- *Three-way data structure constraints*: allows for selecting between a trilinear and a non-trilinear model to describe the data set. From an operational point of view, we may say that a trilinear structure implies a common shape for all profiles related to the same compound in the submatrices of an augmented direction (e.g., in a column-wise

augmented matrix, the profile from the i^{th} compound would have the same shape in all C_i submatrices).

All the constraints to be used during the resolution process are indicated in a vector, e.g., [1,3], non-negativity and closure are going to be applied somewhere in the data set (C and/or S).

3. *Specification of the mode of application of constraints.*

- *Where?*

In three-way data sets, we are asked whether all C submatrices (column-wise augmented data sets) and whether all S submatrices (row-wise augmented data sets) should be constrained the same way.

Do you want to apply the same constraints to all C submatrices?

Do you want to apply the same constraints to all S submatrices?
(y/n)

For each of the constraints selected, we should answer:

- *Where (C and/or S direction) and how to apply the constraint?*

Non-negativity

Enter the non-negativity constraints (1=conc / 2=conc and spectra / 3=spectra)

Direction(s) of application of the constraint.

SELECTION OF THE NON-NEGATIVITY IMPLEMENTATION FOR CONC PROFILES.

Warning: nnls or fnnls algorithms can constrain all or none of the present species.

Constraint of some selected species is only possible with the "forced to zero" option

Enter the selected algorithm for conc (0 = forced to zero / 1 = nnls / 2 = fnnls)

Selection of the algorithm used in the application of the non-negativity constraint.

- 0 The negative values in a profile are updated to zero.
- 1 Non-negative least squares.
- 2 Fast non-negative least-squares.

(1 and 2 are smoother. 0 is the only one that may be applied to all or some of the compounds).

For the 'forced to zero' option, we may select the profiles to be constrained in C.

How many conc profiles should be positive?

Enter a vector with the positive conc profiles (e.g. conc 1 and 3 [1,0,1]) '

If the number of profiles to be constrained is lower than the total, we specify which ones should be constrained in a vector with a length equal to the number of profiles in C. We use a binary code (0 for profiles unconstrained and 1 for profiles to be constrained).

E.g., for a 4-component system, the vector [1,1,0,0] indicates that the first and second profiles should be constrained and the third and the fourth should not.

Analogous questions would be answered for the application of non-negativity to the signal profiles (if needed).

Unimodality

Unimodality constraint applied to:

- 1 : concentration profiles
- 2 : spectra
- 3 : concentration profiles and spectra

Unimodal conc (1), spec(2), conc and spec (3)?

Direction of application of the constraint.

How many species are constrained to have unimodal concentration profiles?

Species with unimodal profiles, e.g. [0,1,..]?

If the number of profiles to be constrained is lower than the total, we specify which ones should be constrained in a vector with a length equal to the number of profiles in C. We use a binary code (0 for profiles unconstrained and 1 for profiles to be constrained).

Unimodal constraint tolerance for the conc?

Defines how strictly the constraint should be applied. If the answer is 1.0, no departures of the unimodal condition are allowed; if the value is higher than 1.0, slight departures from unimodality are allowed. (e.g., 1.05, secondary maxima exceeding in less than a 5% the neighbour value are allowed).

Unimodality implementation: vertical (0), horizontal (1), average (2)?

Algorithms used in the application of unimodality. From the roughest to the softest: vertical, horizontal and average.

Analogous questions would be answered for the application of unimodality to the signal profiles (if needed).

Closure

DIRECTION OF THE CLOSURE

Closure for concentration profiles (1)

Closure for spectra (2)

Specify closure direction

Direction of application of the constraint.

Number of closure constants to be included (select for each experiment)

0 = no closure

1 = one closure for the species concentrations

2 = two closures for species concentrations

Number of closed systems in the same data matrix. E.g. HA / HB, 2.

(first) closure constant is?

Value of the total concentration in the closed system. If not known, you may enter 1.

Closure condition

Equal condition (1) or "lower or equal than" condition (2)?

Defines the tolerance of the constraint. Equal condition (1) forces the sum of the concentrations in the closed system to equal exactly the total concentration at each stage of the process. (2) allows for some departures of this condition, i.e. slight variations of the total concentration in the system may be allowed.

which species are in (first) closure [1,0,1,...]

The compounds in the data set analysed that belong to the closed system should be specified (0 for compounds absent and 1 for compounds present).

Analogous questions would be answered for the application of closure to the signal profiles (if needed). NEVER CLOSE IN BOTH DIRECTIONS!!!

Normalisation

NO CLOSURE, NORMALIZATION OF THE S MATRIX (PURE SPECTRA) CAN BE RECOMMENDED:

Types of normalization: 0 = none

1 = spectra (rows) equal height

2 = spectra (rows) equal length

normalization 0, 1 or 2?:

Direction of application of the constraint. When the system is not closed in any of the directions, normalisation is usually recommended.

Equality constraints

When selected, they activate the use of the information in **csel** and/or **sse1** (see *Input* section) in the resolution process.

CONC EQUALITY (known) /LOWER THAN (selectivity) CONSTRAINTS WILL BE APPLIED !!!!!'),

are they equality (0) or lower than (1) constraints?

Defines the tolerance of the constraint. Equality (0) forces the defined values in **csel** and/or **sse1** to be exactly updated in the concentration profiles and/or spectra. (1) allows for some departures of this condition, i.e, values very close to the ones in **csel/sse1** are allowed.

Three-way structure

Defines the trilinear or non-trilinear structure of the data set. When in doubt, select non-trilinear.

STRUCTURE OF THREE-WAY DATA SETS

No trilinear (0)

trilinear, equal shape and synchronization (all species) (1)

trilinear without synchronization (all species), (2)

trilinear and synchronization for only some species, (3)

Option selected (0/1/2/3)?

In trilinear systems, (2) corrects for trilinear systems with shifted, but equally shaped profiles between matrices for the same compound. (3) allows for the application of partial trilinearity; only to some of the compounds in the data set.

APPLICATION OF THE TRILINEARITY CONSTRAINT')

Application to C matrix (1)
Application to S matrix (2)
Application to C and S matrices (3)

Select the mode of application of the trilinearity constraint

Direction of application of the constraint.

enter the vector of trilinear C profiles [1,0,1,..] 1=yes , 0=no

In situations of partial trilinearity, the profiles meant to be constrained should be specified.

4. *Display of graphical output.*

Do you want graphic output during the ALS optimization (y/n)?

If yes, the evolution of the concentration profiles and pure signals is shown at each iteration.

Bibliography

EFA

H. Gampp, M. Maeder, C.J. Meyer and A.D. Zuberbühler. "Calculation of equilibrium constants from multiwavelength spectroscopic data". III. Model-free analysis of spectrophotometric and ESR titrations". *Talanta*, 32, 1133-1139 (1985).

M. Maeder. "Evolving Factor Analysis for the resolution of overlapping chromatographic peaks". *Anal. Chem.*, 59, 527-530 (1987).

A. de Juan, S. Navea, J. Diewok and R. Tauler. "Local rank exploratory analysis of evolving rank-deficient systems". (submitted, preprint available on request).

FSMW-EFA

H.R. Keller and D.L. Massart. "Peak purity control in liquid chromatography with photodiode array detection by fixed size moving window evolving factor analysis". *Anal. Chim. Acta*, 246, 379-390 (1991).

SIMPLISMA

W. Windig and J. Guilment. "Interactive self-modeling mixture analysis". *Anal. Chem.*, 63, 1425-1432 (1991).

MCR-ALS

R. Tauler, A.K. Smilde and B.R Kowalski. "Selectivity, local rank, three-way data analysis and ambiguity in multivariate curve resolution". *J. Chemometr.* 9, 31-58 (1995).

R. Tauler. "Multivariate curve resolution applied to second order data". *Chemom. Intell. Lab. Sys.*, 30, 133-146 (1995).

General

A. de Juan, E. Casassas and R. Tauler. WILEY ENCYCLOPEDIA OF ANALYTICAL CHEMISTRY: INSTRUMENTATION AND APPLICATIONS. *Soft-modelling of analytical data*. Vol. 11 (2000) 9800-9837. Wiley Interscience.