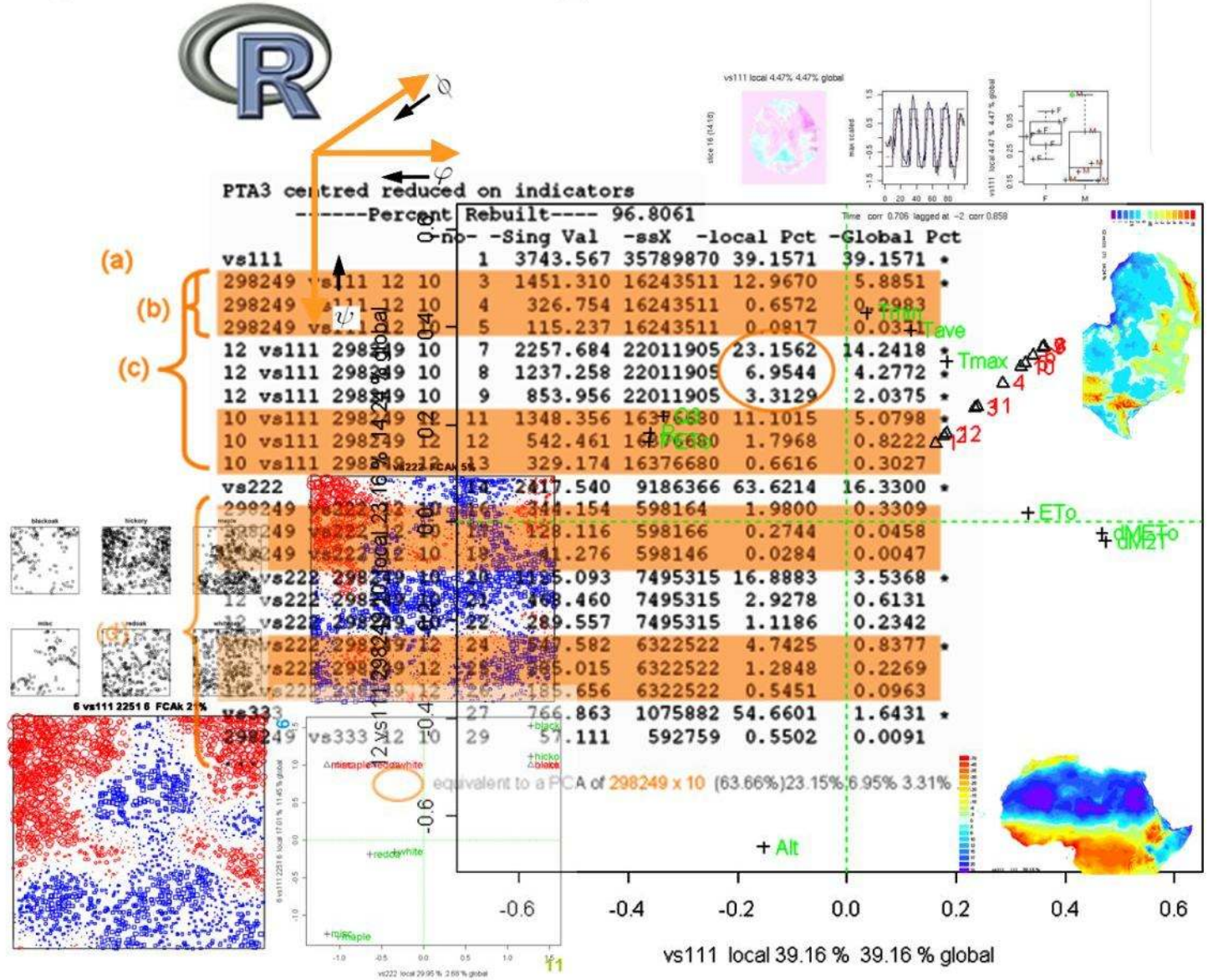
- Principal Tensor Analysis on k modes -



Didier Leibovici - University of Nottingham

1. Introduction

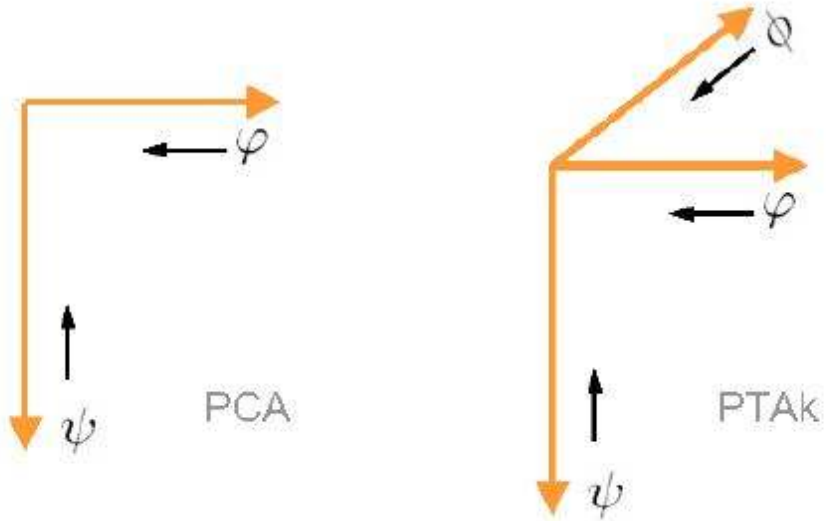
Multidimensional methods may be restricted to dealing with multiway data. As analysing two-way tables, the multi-way table has to be collapsed or unfolded in a table with two modes, thereby looking at interactions of order 2 in a multiple fashion instead of looking at multiple interactions. This is the case, for example, for Multiple Correspondence Analysis as compared to Simple Correspondence Analysis. The former is not the extension of the latter when dealing with more than 2 categorical variables, but rather a "flatter" extension of it where only all 2 ways marginal lack of independence are considered. Within **R**, R Development Core Team (2007), the add-on package **PTAk** Leibovici (2001, 2007) aims at decomposing interactions of order $k > 2$, and, for example, the method `FCAk` within the package decomposes the lack of independence measured by a χ^2 for the k variables in the k-way table. This particular `PTAk` application will be described in section 6, for general purposes but also towards analysing spatial patterns of occurrences.

Beforehand the algebraic background extending matrix calculus will be shortly described in section 2, together with the optimisation procedure of the main method of the package: `PTAk`. A brief comparison with some other well known multiway methods will also be made in this section. Sections 3 and 4 will give an overview of using the package, whilst sections 5, 6, and 7 will describe some generic approaches to derive decomposition models useful in a spatio-temporal context. The framework used within the package **PTAk** is indeed extending some duality principles Cailllez (1976); Escoufier (1987); Dray and Dufour (2007), therefore approaches in multidimensional analysis focusing on spatial data and on temporal data, such as methods decomposing local and global variances as in **ade4**, Chessel, Dufour, Dray, Lobry, Ollier, Pavoine, and Thioulouse (2007), can be reused.

2. Understanding PTAk relatively to PCA

PTAk offers a decomposition similar to what is obtained from matrices with a Principal Component Analysis, but working on tensors, *i.e.* in mathematical algebra they are multilinear maps, seen here simply as multiple-entries tables (k entries). Tensor algebra properties allow to derive multiple-entries table calculus, extending matrix calculus, Leibovici and Sabatier (1998); Dauxois, Roomain, and Viguier-Pla (1994). In order to describe the generalisation proposed with the PTAk model, let us first rewrite the PCA method within a tensorial framework. For a given matrix X of dimension $n \times p$, the first principal component is a linear combination (given by a p-dimensional vector φ_1) of the p columns ensuring maximum sum of squares of the coordinates of the n-dimensional vector obtained. The square root of this

Figure 1: Illustrative comparison between PCA and PTAk (here with $k = 3$) when computing singular values by Complete Contractions given in the equations 1 and 2: the basis of the RPVSCC algorithm.



sum of squares is called the first singular value σ_1. One has: ${}^t(X\varphi_1)(X\varphi_1) = \sigma_1^2$ and $X\varphi_1/\sigma_1$ is the principal component normed to 1. This maximisation problem can be written either in matrix form or tensor form:

$$\begin{aligned}\sigma_1 &= \max_{\substack{\|\psi\|_n=1 \\ \|\varphi\|_p=1}} ({}^t\psi X\varphi) = \max_{\substack{\|\psi\|_n=1 \\ \|\varphi\|_p=1}} X..(\psi \otimes \varphi) \\ &= {}^t\psi_1 X\varphi_1 = X..(\psi_1 \otimes \varphi_1)\end{aligned} \tag{1}$$

In equation 1 X is used either for the matrix or the tensor. An easy way of understanding computationally the algebraic operators ".." and "⊗" is to see them as the following operations: $\psi_1 \otimes \varphi_1$ is a np vector of the n blocks of the p vectors $\psi_{1i}\varphi_1$, $i = 1, ...n$; ".." called a contraction generalises the multiplication of a matrix by a vector and in the case like here of equal dimensions of the two tensors (np) corresponds to the natural inner product (X is then also seen an np vector). ψ_1 is termed first principal component, φ_1 first principal axis, $(\psi_1 \otimes \varphi_1)$ is called first principal tensor. Within **R**, tensor products can be utilised with the outer product (%o%) or the Kronecker product (%x%). The tensor being in fact an algebraic operation, it is up to the computational step to choose one or the other. The computational description of "⊗", given above, is using the Kronecker product:

```
> c(1,2,3)%x%c(4,5)

[1]  4  5  8 10 12 15
```

The result with the outer product is an array which could be preferred for tensor product representation, as here a matrix emphasised the bilinear property. The vectorisation of the array is a permuted version of the Kronecker product:

```
> c(1,2,3)%o%c(4,5)

     [,1] [,2]
[1,]    4    5
[2,]    8   10
[3,]   12   15

> dim(c(1,2,3)%o%c(4,5)%o%c(3,1))

[1] 3 2 2

>  all(as.vector(t(c(1,2,3)%o%c(4,5)))==c(1,2,3)%x%c(4,5))

[1] TRUE
```

Notice here the description of a tensor of order 2, a bilinear map, as associated to a matrix which is usually associated to one linear map. The duality diagram Cailllez (1976); Escoufier (1987); Dray and Dufour (2007) comes to complete the association with another linear map

on the dual spaces involved to define the other linear map: expressed by the transposed matrix. The contraction, "..", is implemented within the function `CONTRACTION` and it uses the package **tensor**. Now if X is a tensor of higher order, say 3 here we can look for the first principal tensor associated with the singular value with the optimisation form:

$$\begin{aligned} \sigma_1 &= \max_{\substack{\|\psi\|_s=1 \\ \|\varphi\|_v=1 \\ \|\phi\|_t=1}} X..(\psi \otimes \varphi \otimes \phi) \\ &= X..(\psi_1 \otimes \varphi_1 \otimes \phi_1) \end{aligned} \tag{2}$$

This is a direct extension of equation 1 which practically, both being expressed by practical schemas on figure 1 with contractions made either on a matrix table or on a tensor of order 3. The further extension to $k > 3$ is straightforward. `CONTRACTION.list` is convenient relatively to equations 1 and 2 as it performs the contraction without computing the tensor product of the vectors in the first place as in fact algebraically:

$$X..(\psi \otimes \varphi \otimes \phi) = (X..\psi)..(\varphi \otimes \phi) = (X..\varphi)..(\psi \otimes \phi) = (X..\phi)..(\psi \otimes \varphi) = ((X..\psi)..\varphi))..\phi \tag{3}$$

The function `SINGVA` computes the best rank-one approximation of the given tensor X

together with its singular value, given by equation 2 (and for higher orders). The therein algorithm, called *RPVSCC*, is inspired from the algorithm of Reciprocal Averaging Hill (1973) also known as the transition formulae in Correspondence Analysis and in the signal processing community as the "power method". Notice `PTAk`, `CANDPARA` (PARAFAC/CANDECOMP) and `PCAn` (Tucker$_n$ model) are equivalent when looking for best rank-one approximation. (References for the last two methods are given in the package.

```
> PTAk(X,nbPT=1,nbPT2=0)==CANDPARA(X,dim=1)==PCAn(X,dim=rep(1,length(dim(X))))
```

This cannot be strictly verified using **PTAk** as `CANDPARA` and `PCAn` in their implementation only accepts rank approximation greater than 1. Working around is :

```
> X=c(1,2,3)%o%c(2,4,6)%o%c(3,7) +rnorm(18,sd=0.0001)
> sol1=PTAk(X,nbPT=2,nbPT2=0) ; sol2=CANDPARA(X,dim=2); sol3=PCAn(X,dim=c(2,2,2))
> sol1[[1]]$v[1,] ; sol2[[1]]$v[1,];sol3[[1]]$v[1,] ; sol1[[3]]$d


[1] 0.2672617 0.5345234 0.8017830
 [1] -0.2672617 -0.5345234 -0.8017830
  [1] -0.2672617 -0.5345234 -0.8017830

[1] 2.132416e+02 2.086484e-04
```

Where the first mode component for the first Principal Tensor given by `sol1[[1]]$v[1,]` is equivalent to the other approximations, with a nearly rank-one tensor: `sol1[[3]]$d` are the singular values.

Adding an orthogonality constraint (projection onto the orthogonal tensorial of the principal tensor, see equation 4) allows us to carry on the algorithm to find the second principal tensors and so on. Following this algorithm schema, the `PTAk` decomposition obtained offers a way of synthesising the data according to uncorrelated sets of components. Within this schema implemented for the functions `PTA3` and `PTAk` one can distinguish main Principal Tensors from associated Principal Tensors. The latter are associated to a main principal tensor as they show one or more component of this principal tensor in their sets of components. The associated principal tensors are obtained by a $PTA(k-1)$-modes decomposition once the k-modes data has been "contracted" by the given component. This makes the algorithm a recursive algorithm with the following procedure, where here $k = 3$:

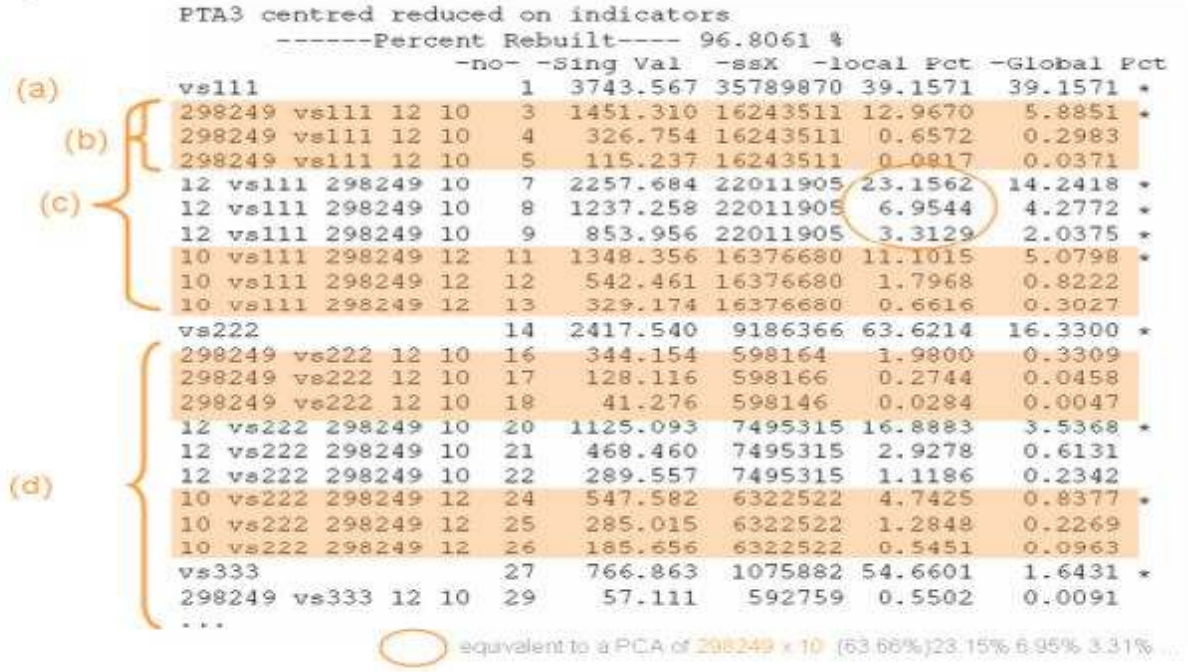
$$\begin{aligned} PTA_3(X) &= \sigma_1(\psi_1 \otimes \varphi_1 \otimes \phi_1) \\ &+ \psi_1 \otimes_1 PTA_2(P(\varphi_1^{\perp} \otimes \phi_1^{\perp})X..\psi_1) \\ &+ \varphi_1 \otimes_2 PTA_2(P(\psi_1^{\perp} \otimes \phi_1^{\perp})X..\varphi_1) \\ &+ \phi_1 \otimes_3 PTA_2(P(\psi_1^{\perp} \otimes \varphi_1^{\perp})X..\phi_1) \\ &+ PTA_3(P(\psi_1^{\perp} \otimes \varphi_1^{\perp} \otimes \phi_1^{\perp})X) \end{aligned} \tag{4}$$

The notation $\otimes_i$ means that the vector on the left hand will take the ith place, among the k places here, in each full tensorial product, *e.g.* $\varphi_1 \otimes_2 \alpha \otimes \beta = \alpha \otimes \varphi_1 \otimes \beta$. More details on the properties of the method and on each function of the package is given in the references Leibovici and Sabatier (1998); Leibovici (2007). The equation 4 and figure 2 illustrates the multi-hierarchical decomposition obtained with the `PTAk` method. On figure 2, in almost the

same way as for PCA, one gets a hierarchy of principal tensors corresponding to a hierarchy of sum of squares, *i.e.* by the squared of the singular values (σ) under the column `-Sing Val` associated to each principal tensor. It is a multilevel hierarchy in agreement with the equation 4. Percents of variability associated to each Principal tensors can be used to retain main variability within the data tensor X. These percentages are in the `-Global Pct` column of figure 2 whereas `-local Pct` are relative to the sum of squares given in column `-ssX` linked to the current tensorial optimisation as defined in equation 4. Plots of the vector components of a particular principal tensor allows the description of the extracted variability for each Principal Tensor.

We have seen that *PTA-k*modes, PARAFAC/CANDECOMP and Tucker$_n$ were equivalent when looking for the best rank-one approximation. Then the methods differs as also differs the rank definitions attached to the models. *PTA-k*modes will *try* to look for best approximation according to the orthogonal rank (*i.e.* the rank-one tensors (of the decomposition) are orthogonal), Tucker$_n$ or PCA-*n*modes will look for best approximation according to the space-ranks (*i.e.* ranks of every bilinear form deducted from the original tensor

Figure 2: Output summary from the function `summary()` on a `PTAk` object, here the climatic data described in section 3.1: (a) is the first principal tensor, (c) represents all the associated principal tensors to first one such like (b) are the spatial-mode associated principal tensors, (d) corresponds to the `PTAk` decomposition on the projection onto the orthogonal tensorial of the first principal tensor.



(folding the multi-array into a matrix), that is the number of components in each space), PARAFAC/CANDECOMP will look for best approximation according to the rank (*i.e.* the rank-one tensors are not necessarily orthogonal). It is said here "*PTA-k*modes will *try*" as it has been proven recently on an example that the orthogonal-rank was not providing necessarily a nested decomposition as *PTA-k*modes implies, Kolda (2003). One can also notice that *PTA-k*modes extends the PARAFAC-orthogonal if one considers only main Principal Tensors (not associated ones) *i.e.* by setting `nbPT2=0` in the `PTAk` call or by ignoring them. The function `REBUILD` will return the approximated or filtered dataset according to the method used, either `PTAk`, `CANDPARA`, or `PCAn`. `REBUILD` allows to choose the list of tensors and also by selected a global threshold for percentage of variability explained by each elementary tensors. For `PCAn` the function calls `REBUILDPCAn` which is not using the previously described parameters.

```
> Xapp=REBUILD(sol1,nTens=c(1,2),testvar=1e-12)

-- Variance Percent rebuilt X  at  100 %
-- MSE  4.378514e-09
-- with  2  Principal Tensors out of  2  given
-- compression     0  %
```

For PTAk and CANDPARA, the approximation is done according to the equation model, here written for a tensor of order 4:

$$X = \sum_{i \in \varsigma} \sigma_i \psi_i \otimes \varphi_i \otimes \phi_i \otimes \xi_i + \epsilon \tag{5}$$

where ς is a set of the selected elementary tensors. The PCAn rebuilt approximation is a direct generalisation of model from Kroonenberg and De Leeuw (1980):

$$X = (\psi \otimes \varphi \otimes \phi \otimes \xi)..C + \epsilon \tag{6}$$

where the components here are matrices of components with as many columns in each mode-space as asked for during the optimisation analysis (the space-ranks), and C being the *core* tensor with dimensions corresponding to the space-ranks. The algorithm written in the function PTAk is fully recursive therefore slower when $k = 3$ than PTA3 which takes benefit from knowing the order of the tensor. PROJOT is the function within PTAk performing the orthogonal tensor projection but can also be used for any structure or design associated with each mode to perform a linear constrained analysis in the same way as for PCAIV (Principal Component Analysis on Instrumental Variables), see Leibovici (2000) for a full description of using PTAIVk and in the **PTAk** manual for PROJOT where a quick implementation is given as example.

3. Running a general PTAk

SINGVA, PROJOT and other functions also used for 2 modes analysis, SVDGen, in **PTAk** stand in the package as used within PTAk, but also to perform particular analysis. Indeed, the main method is the PTAk along with the other multiway models implemented in the package. So once you have loaded or scanned the dataset from other sources or format, put it in a multi-array, an array object in R you can run the PTAk decomposition. This is illustrated, below, with the dataset related to ecoclimatic delineation problem, Leibovici, Quillevere, and Desconnets (2007), where dynamics over a typical year of 10 climatic indicators were analysed in the circum-saharan zone, using their monthly average estimates. Here the studied zone has been limited to Tunisia; the shapefile contains a regular grid with the multivariate values:

```
> library(PTAk)
> library(tensor)
> library(maptools)
> library(RColorBrewer)
> Yl=brewer.pal(11,"PuOr")
> Zone_climTUN<-read.shape( "E:\\R_GIS\\R_GilHF\\TUN\\tunisie_climat.shp")
> plot(Zone_climTUN,ol=NA,auxvar=Zone_climTUN$att.data$PREC_OCTO,nclass=20,
                colrmp=colorRampPalette(Yl)(21)))
  #indicators 84 +3 to repeat
> Zone_clim<-Zone_climTUN$att.data[,c(2:13,15:26,28:39,42:53,57:80,83:95,55:56)]
> Zot <-Zone_clim[,85:87] ;temp <-colnames(Zot)
> Zot <- as.matrix(Zot)%x%t(as.matrix(rep(1,12)))
> colnames(Zot) <-c(paste(rep(temp [1],12),1:12),paste(rep(temp [2],12),1:12),
                paste(rep(temp [3],12),1:12))
> Zone_clim <-cbind(Zone_clim[,1:84],Zot)
  # 2599 120 space x (mois x var)
```

```
> dim(Zone_clim)

[1] 2599 120

Zone3w <- array(as.vector(as.matrix(Zone_clim)),c(2599,12,10))
  # space x mois x var
> dim(Zone3w);dimnames(Zone3w)<-list(rownames(Zone3w),1:12,
                c("P","Tave","ETo","PETo","Tmax","Tmin","Q3","Alt","dM2T","dMETo"))
```

```
[1] 2599   12   10

> Zone3w.PTA3<-PTA3(Zone3w,nbPT=3,nbPT2=3,minpct=0.1)

 ---Final iteration---  7
 --Singular Value--  59898.86  -- Local Percent --  97.62936 %
 ---Final iteration---  26
 --Singular Value--  2860.392  -- Local Percent --  68.66842 %
 ---Final iteration---  39
 --Singular Value--  401.1593  -- Local Percent --  38.09571 %
  ++ Last 3-modes vs <  0.1 % stopping this level and under ++
 -----Execution Time----- 7.43

> summary(Zone3w.PTA3,testvar=0.01)

 ++++ PTA-  3 modes ++++
               data=  Zone3w   2599 12 10
    PTA3 centree reduite sur var
                ------Percent Rebuilt---- 99.97716 %
                ------Percent Rebuilt from Selected ---- 99.95512 %
                 -no- --Sing Val--    --ssX-- --local Pct-- --Global Pct--
vs111               1      59898.9 3674994157      97.62936      97.629361
2599 vs111 12 10    3       3243.0 3598688392       0.29226       0.286187
12 vs111 2599 10    6       7354.4 3652184965       1.48097       1.471774
12 vs111 2599 10    7       3142.0 3652184965       0.27031       0.268629
vs222              11       2860.4   11915003      68.66842       0.222636
12 vs222 2599 10   16       1677.1   11037709      25.48250       0.076536
 ++++               ++++
 Shown are selected over  15  PT  with var> 0.01 % total
```

The first Principal Tensor is capturing most of the variability 97.6% which is nearly as much as the decomposition up to 3 main Principal Tensors and 3 for each associated, *i.e.* at each second level analysis (a PCA). One should have had for each main Principal Tensor 9 associated Principal tensors, making the listing 30 lines long, but having always the first Principal Component redundant that makes only 6, so out of the 21 potential Principal it is shown only the one with `Global Pct` >0.01%. The listing `summary` mentions `...over 15 PT` as in the call function, the parameter `minpct=0.1` forces the algorithm to stop a $k >= 3$-level (no sub-level analysis), if this percentage of variability is not met: it happened here for `vs333`. The full description of the ouput `summary` is explained in the section 3.2 where the listing ouput provides a more complete form.

This first `PTAk` analysis is not very useful as the variations and range of values can be very different from one climatic variable to another one. So the main variations captured by the principal tensors will be towards this differentiation without expressing necessarily the interactions between the variables and them with the spatio-temporal domain which may only be detected in some principal tensors (main or associated) with comparatively very small singular values. As usually done in PCA: centring and scaling the variables, preprocessing transformation may be crucial as part of the modelling and analysis process.

A complete presentation with particular issues for spatio-temporal data can be found in Leibovici (2009b) as well as the references quoted here. Some other examples and references about `PTAk` can also be found in `http://c3s2i.free.fr`.

References

Leibovici DG (2009a). ***PTAk:*** Principal Tensor Analysis on k modes, **R**-package version 1. 2-0. edition. URL http://cran.r-project.org/web/packages/PTAk/index.html.

Leibovici DG (2009b). "Spatio-temporal Multiway Decompositions using Principal Tensor Analysis on k-modes: the **R** package **PTAk**." *Journal of Statistical Software*, **xx**(x), 1–33. URL http://www.jstatsoft.org.

Leibovici D (2001). "**PTAk**: Principal Tensor Analysis on k modes." *Contributing R-package*, **version 1.1-4**. URL http://c3s2i.free.fr.

Leibovici D, EL Maach H (1997). "Une décomposition en Valeurs Singulières d'un élément d'un produit Tensoriel d' k espaces de Hilbert Séparables." *Compte Rendus de l'Académie des Sciences / Statistiques and Probabilités*, **tome 325, série I**, 779–782.

Leibovici D, Quillevere G, Desconnets JC (2007). "A Method to Classify Ecoclimatic Arid and Semi-Arid Zones in Circum-Saharan Africa Using Monthly Dynamics of Multiple Indicators." *IEEE Transactions on Geoscience and Remote Sensing*, **45**(12), 4000–4007.

Leibovici D, Sabatier R (1998). "A Singular Value Decomposition of k-Way Array for a Principal Component Analysis of Multiway Data, PTA-k." *Linear Algebra and Its Applications*, **269**, 307–329.

R Development Core Team (2007). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

functions /UML structure

- This is just a quick description …

A true UML will come with for future versions … (not necessary here)

- The “class” PTAk is described in the manual is the top class for the package
- PCAn, FCAk, CANDPARA inherit from PTAk
- SVDgen output are also PTAk class (k=2)

functions /UML structure

- PTAk, PCAn, FCAk, CANDPARA, SVDgen are the main functions
- Methods: plot, summary, REBUILD
- other functions, MultCent etc …

- That’s it for now!