Journal of Statistical Software

April 2014, Volume 57, Issue 7. *http://www.jstatsoft.org/*

Three-Way Component Analysis Using the R Package ThreeWay

Paolo Giordani
Sapienza University of Rome

Henk A. L. Kiers
University of Groningen

Maria Antonietta Del Ferraro
Sapienza University of Rome

Abstract

The R package **ThreeWay** is presented and its main features are illustrated. The aim of **ThreeWay** is to offer a suit of functions for handling three-way arrays. In particular, the most relevant available functions are `T3` and `CP`, which implement, respectively, the Tucker3 and Candecomp/Parafac methods. They are the two most popular tools for summarizing three-way arrays in terms of components. After briefly recalling both techniques from a theoretical point of view, the functions `T3` and `CP` are described by considering three real life examples.

Keywords: multiway analysis, Tucker3, Candecomp/Parafac, R, **ThreeWay**.

1. Introduction

In statistics data generally refer to the observations of some variables on a set of units and are stored in a (two-way) matrix, say $\mathbf{X}$ of order $(I \times J)$, where I and J denote the numbers of units and variables, respectively. The generic element of $\mathbf{X}$ is x_{ij} and, therefore, data are indexed by $i \in \{1,\ldots,I\}$ and $j \in \{1,\ldots,J\}$ concerning the unit and variable modes, respectively. Here, we use the term 'mode' to refer to a set of entities. However, in several situations, the available data can be indexed by $i \in \{1,\ldots,I\}$, $j \in \{1,\ldots,J\}$ and $k \in \{1,\ldots,K\}$ where K denotes the number of occasions. In this case, the available information consists of some variables collected on a set of units on different occasions and is usually stored in a (three-way) array, say $\underline{\mathbf{X}}$ of order $(I \times J \times K)$, with generic element x_{ijk}. The array can then be seen as a box in which the ways (or indices) correspond to the vertical, horizontal and depth axis. For the sake of generality, in the following we decided not to refer to unit, variable and occasion modes. Rather, we refer to, respectively, the A-mode (with I entities), B-mode (with J entities) and C-mode (with K entities).

Multiway data analysis concerns the cases in which the number of indices is higher than two (three-way data analysis when the number of indices is three). In this paper we shall limit our attention to the three-way case. In the literature there exist several proposals for performing component models on three-way data. The two most popular techniques are the Tucker3 (T3) model (Tucker 1966) and the Candecomp/Parafac (CP) model (independently proposed by Carroll and Chang 1970; Harshman 1970). The R package **PTAk** (Leibovici 2010) and the functions `PCAn` and `CANDPARA` therein can be considered for applying three-way analysis. However, as far as we saw, some limitations can be found. For instance, it is not possible to extract one component for the A- and B-mode in a T3 analysis or to perform CP with one component. This may limit the applicability of the functions as we shall see in the following sections.

The aim of this work is to illustrate the R (R Core Team 2013) package **ThreeWay** (Del Ferraro, Kiers, and Giordani 2014) for performing a complete three-way analysis. The paper is organized as follows. In Section 2, we provide some preliminary notions on three-way data and we introduce the T3 and CP models. Then, Sections 3 and 4 are devoted to the main features of **ThreeWay** with particular reference to the implementations of T3 and CP considering three benchmark data sets. Finally, in Section 5 some concluding remarks are given.

2. Methodological background

The data array $\underline{\mathbf{X}}$ can be seen as a collection of K matrices of order $(I \times J)$. It can be convenient to explicitly take into account this by expressing the available information in terms of a new matrix, which we may call supermatrix, containing such a collection. This can be done for instance as $\mathbf{X}_A = [\mathbf{X}_{..1} \cdots \mathbf{X}_{..k} \cdots \mathbf{X}_{..K}]$, where $\mathbf{X}_{..k}$ stands for the matrix of order $(I \times J)$ concerning the k-th entity of the C-mode. $\mathbf{X}_{..k}$ is usually called the k-th frontal slab or slice of $\underline{\mathbf{X}}$. In other words, $\mathbf{X}_A$ is the matrix with I rows (corresponding to the A-mode entities) and JK columns (corresponding to all the combinations of the B- and C-mode entities) obtained juxtaposing next to each other the frontal slabs of $\underline{\mathbf{X}}$ (A-mode matricization of $\underline{\mathbf{X}}$). The process of transforming a three-way array into a two-way matrix is usually denoted as matricization or, especially in chemometrics, unfolding. It must be noted that there exist several ways to matricize an array (see, for more details, Kiers 2000). However, in this context, it is sufficient to consider $\mathbf{X}_A$ for describing the T3 and CP models.

In principle standard Principal Component Analysis (PCA) applied to $\mathbf{X}_A$ or to the matrix containing the mean values of the $\mathbf{X}_{..k}$'s can be used for summarizing the data in $\underline{\mathbf{X}}$. The former, sometimes called PCA on supermatrices (PCA-SUP), and the latter are available in the functions `pcasup3` and `pcamean` of **ThreeWay**. Their outputs give a preliminary insight into the data. However, these tools offer information which is incomplete at best because the three-way interactions among the data are arbitrarily ignored. Ad-hoc methods, such as T3 and CP, should be thus considered. In **ThreeWay**, the CP model is implemented in the function `CP` and the T3 model in the function `T3`. Some variants of T3, namely the Tucker2 (T2) and Tucker1 (T1) models are also available in the functions `T2` and `T1`, respectively. In the remaining part of this section, the CP and T3 models (and its variants T2 and T1) will be briefly recalled. Refer to Bro (1998), Kroonenberg (1983, 2008), Smilde, Bro, and Geladi (2004) for extensive monographs on T3 and CP and, in general, on multiway analysis with applications in several domains.

2.1. The Tucker3 model

The Tucker3 (T3) model (Tucker 1966) is a multi-linear model summarizing $\underline{\mathbf{X}}$ by extracting different components for every mode. The Tucker3 model with P components for the A-mode, Q for the B-mode and R for the C-mode, can be formalized as

$$\mathbf{X}_A = \mathbf{A}\mathbf{G}_A(\mathbf{C}^\top \otimes \mathbf{B}^\top) + \mathbf{E}_A, \tag{1}$$

where $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are the component matrices for the A-, B- and C-modes, respectively, the generic elements of which are a_{ip}, b_{jq} and c_{kr} expressing the component scores of the i-th entity on the p-th component for the A-mode, of the j-th entity on the q-th component for the B-mode and of the k-th entity on the r-th component for the C-mode, respectively. Furthermore, $\mathbf{G}_A$ is the matricized version of the so-called core array $\underline{\mathbf{G}}$ of order $(P \times Q \times R)$. Its generic element g_{pqr} gives the interaction among the components of the three modes. Finally $\mathbf{E}_A$ denotes the matricized array of the errors. The symbol $\otimes$ is the Kronecker product between two matrices (given two matrices $\mathbf{U}$ and $\mathbf{V}$, it is $\mathbf{U} \otimes \mathbf{V} = \begin{bmatrix} u_{11}\mathbf{V} & \cdots & u_{1J}\mathbf{V} \\ \vdots & \ddots & \vdots \\ u_{I1}\mathbf{V} & \cdots & u_{IJ}\mathbf{V} \end{bmatrix}$). A scalar formulation of the T3 model equivalent to Equation 1 can also be provided:

$$x_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{r=1}^{R} a_{ip}b_{jq}c_{kr}g_{pqr} + e_{ijk}. \tag{2}$$

In the T3 model, limited numbers of components for *all* the three modes are sought. However, it can be useful to reduce only two modes or just one mode. In these cases, the Tucker2 (T2) or Tucker1 (T1) models can be introduced, respectively. Reducing without loss of generality the A- and B-modes, the T2 model (T2-AB) can be written as

$$x_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q} a_{ip}b_{jq}g_{pqk} + e_{ijk}. \tag{3}$$

If, without loss of generality, our interest relies in reducing the A-mode, we obtain the T1 model (T1-A):

$$x_{ijk} = \sum_{p=1}^{P} a_{ip}g_{pjk} + e_{ijk}. \tag{4}$$

The optimal parameter matrices of the T3, T2 and T1 models are found by minimizing $\|\mathbf{E}_A\|^2 = \sum\limits_{i=1}^{I}\sum\limits_{j=1}^{J}\sum\limits_{k=1}^{K} e_{ijk}^2$. For the T3 and T2 models, this can be done implementing an Alternating Least Squares (ALS) algorithm, which alternatingly updates every parameter matrix keeping fixed the remaining ones upon convergence. It is assumed that an algorithm converges when the values of the loss function in two consecutive iterations differ less than a pre-specified threshold. It can be shown that these algorithms converge to at least a local minimum in a limited number of iterations. To limit the risk of hitting local optima, more than one start is recommended. Since T1 is equivalent to a PCA on $\mathbf{X}_A$, the T1 solution can be obtained by the SVD of $\mathbf{X}_A$. Different variants of T1 can be obtained by choosing different matricizations.

It can be shown that the obtained solution in terms of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{G}$ is not unique. In fact, equally well-fitting solutions can be obtained considering $\tilde{\mathbf{A}} = \mathbf{AS}$, $\tilde{\mathbf{B}} = \mathbf{BT}$, $\tilde{\mathbf{C}} = \mathbf{CU}$ and $\tilde{\mathbf{G}}_A = \mathbf{S}^{-1}\mathbf{G}_A({\mathbf{U}^{\top}}^{-1} \otimes {\mathbf{T}^{\top}}^{-1})$, where $\mathbf{S}$, $\mathbf{T}$ and $\mathbf{U}$ are rotation matrices of appropriate order. Such a rotational freedom holds provided that the rotations of the component matrices are compensated in the core. Although it might represent a limitation of the analysis, the indeterminacy of the solution can be used in order to obtain simple structure solutions. See, for instance, Kiers (1998) in which the procedure called orthomax for jointly rotating to simple structure the component matrices and the core is proposed. The estimation of the T3 model parameters can be carried out using the function `T3` of the R package **ThreeWay**. Its main features shall be illustrated in Section 3. The function `T2` and `T1` are implemented in order to perform T2 and T1, respectively.

2.2. The Candecomp/Parafac model

The Candecomp/Parafac (CP) model (Carroll and Chang 1970; Harshman 1970) aims at reducing $\underline{\mathbf{X}}$ by extracting the same number of components, say S, for every mode. In scalar notation the CP model can be written as

$$x_{ijk} = \sum_{s=1}^{S} a_{is} b_{js} c_{ks} + e_{ijk}, \tag{5}$$

with obvious notation. Although the CP model is rather different from T3 and satisfies different properties, it can be seen as a constrained version of T3 with $P = Q = R = S$ and $g_{pqr} = 1$, if $p = q = r$, and $g_{pqr} = 0$, otherwise. The matrix formulation of CP helps to highlight such a relationship. In fact, it is

$$\mathbf{X}_A = \mathbf{A}\mathbf{I}_A(\mathbf{C}^{\top} \otimes \mathbf{B}^{\top}) + \mathbf{E}_A, \tag{6}$$

where $\mathbf{I}_A$ is the matricized version of the three-way identity array $\underline{\mathbf{I}}$ ($i_{pqr} = 1$, if $p = q = r$, and $i_{pqr} = 0$, otherwise).

By comparing Equation 1 and Equation 6 we can conclude that the CP model can be seen as a T3 model with $\mathbf{G}_A = \mathbf{I}_A$. The most relevant difference between CP and T3 concerns the so-called intrinsic axis property. By this, under mild conditions, the CP solution is unique up to rescaling and permutation of the columns of the component matrices. It is easy to see that uniqueness holds because rotations of the component matrices cannot be compensated into the core which is constrained to be equal to $\underline{\mathbf{I}}$. A nice property of the CP model is that its solution with S components is the best S-rank approximation of $\underline{\mathbf{X}}$, where the rank of an array is defined according to Kruskal (1977). It follows that the rank of $\underline{\mathbf{X}}$ is equal to the smallest number of components for which a perfect fit CP solution is obtained.

A suitable ALS algorithm can be adopted for obtaining the optimal CP solution. Again the optimal parameter matrices are found such that the residual sum of squares is minimized. In order to apply the CP model, the function `CP` of the R package **ThreeWay** can be considered and its main features will be described in Section 4.

3. The function `T3` of the **R** package **ThreeWay**

We now apply the T3 model to the 'Learning to read' data (Bus 1982). The data set refers to the process of learning to read of seven pupils ($I = 7$). Five tests ($J = 5$) are used to evaluate the learning process: each test measures different reading aspects. The tests are 'Letter knowledge' (L), the ability to read 'Regular orthographic short words' (P), 'Regular orthographic long words' (Q), 'Regular orthographic long and short words within context' (S), 'Irregular orthographic long and short words' (R). The pupils are tested weekly from week 3 to week 47 except for eight holidays weeks, hence $K = 37$. The aim of the study is to investigate the learning process and whether the performances of the pupils are equal over time. Of course, the first step of the analysis consists of loading the data.

```
R> library("ThreeWay")
R> data("Bus")
```

The function `T3` requires the data set to be analyzed and, if available, the labels `laba`, `labb` and `labc` for the entities of the three modes (if not available, `T3` first asks to the user for adding them by keyboard and then generates them automatically when the user decides not to add them). The data set can be an object of class `array`, `data.frame` or `matrix`. In the latter two cases, the A-mode matricization of the original array must be given as input and the numbers of entities of the A-, B- and C-modes must be given interactively by the user. `Bus` is an object of class `matrix` with the names of the rows corresponding to the labels of the A-mode (pupils) and the names of the columns corresponding to a combination of the labels of the B- and C-modes (tests and time occasions, respectively).

```
R> laba <- rownames(Bus)
R> labb <- substr(colnames(Bus)[1:5], 1, 1)
R> labc <- substr(colnames(Bus)[seq(1, ncol(Bus), 5)], 3, 8)
```

A relevant point to be addressed concerns preprocessing. In fact, prior to fitting a model to the data, it is fundamental to decide how to preprocess the data. Preprocessing can be done by centering within a mode or a combination of modes and normalizing across a mode. Differently from a two-way analysis in which data are usually centered and/or normalized across the rows, in a three-way context, it is not straightforward how to preprocess the data since different options are available. We can say that the main aim of the preprocessing step is to eliminate artificial differences in levels and scales. Centering is helpful in order to get ratio-scale (rather than interval-scale) data, i.e., the observed values must be proportional and a zero value denotes a lack of the property being measured. Note that the CP and T3 (and its variants) models require ratio-scale data. The normalization step does not make the data consistent with the model but allows us to avoid that the results are affected by differences in range among entities of one or more modes. In the package **ThreeWay** centering and normalization are done according to Kiers (2000) and can be performed using functions `cent3` and `norm3`, respectively, specifying the mode(s) within or across we want to center or normalize the data. These functions are automatically implemented when launching `T3`. For a deeper insight into preprocessing in three-way analysis see Harshman and Lundy (1984) and Bro and Smilde (2003).

In `norm3`, data are normalized to sum of squares equal to product of size of other modes. Alternatively, one can scale the data so that they range from 0 to 1. Since the five tests have different score ranges, we decide to rescale them as was done by Kroonenberg (1983) and Timmerman (2001), who already analyzed the data using T3.

```
R> max.scale <- c(47, 10, 10, 15, 15)
R> maxBus <- rep(max.scale, 37)
R> BusN <- t(t(Bus)/maxBus)
```

Here, `max.scale` contains the maximum value for every test (the minimum value is 0). We have not centered the data because they have a meaningful zero point in 0. In fact, when the score of the pupil i on the test j at the occasion k is 0, we can conclude that the reading ability of such a pupil is absent. The preprocessed data set is available in `BusN` and the function `T3` can be run.

```
R> t3bus <- T3(BusN, laba, labb, labc)
```

In the following, we are going to summarize the most relevant steps for carrying out a three-way analysis process. Thus, some steps are omitted for the sake of brevity.

Prior to choosing the numbers of components, `T3` gives a suggestion based on the generalized scree test (Timmerman and Kiers 2000; Kiers and der Kinderen 2003) according to the Convex Hull procedure (Ceulemans and Kiers 2006). To save time, PCA-SUP analyses are considered. In fact, the eigenvalues resulting from such PCA analyses of supermatrices give an indication as to the required number of components for each mode. The corresponding functions called in `T3` are `T3runsApproxFit` and `DimSelector`. However, in our analysis we decide to operate as in Timmerman (2001) summarizing the data using two components for the A-mode ($P = 2$), one for the B-mode ($Q = 1$) and two for the C-mode ($R = 2$).

```
How many A-mode components do you want to use?
1: 2
Read 1 item
How many B-mode components do you want to use?
1: 1
Read 1 item
How many C-mode components do you want to use?
1: 2
Read 1 item
```

In this way the fit of the model is very high (96.26%) as one can see inspecting the following output obtained considering two additional random starts. Note that the T3 solution is obtained in `T3` calling the function `T3func`.

```
If you want additional runs, specify how many (e.g., 4):
1: 2
Read 1 item
Run no. 1
Rational ORTHONORMALIZED start
Tucker3 function value at start is 26.5805045184081
Tucker3 function value is 25.7735931853048 after 4 iterations
Fit percentage is 96.26469729604 %
```

```
Procedure used 0.03 seconds
Run no. 2
Random ORTHONORMALIZED starts
Tucker3 function value at start is 688.092368487312
Tucker3 function value is 25.7735931724334 after 6 iterations
Fit percentage is 96.2646972979055 %
Procedure used 0.03 seconds
Run no. 3
Random ORTHONORMALIZED starts
Tucker3 function value at start is 689.615935107658
Tucker3 function value is 25.7735931361569 after 5 iterations
Fit percentage is 96.264697303163 %
Procedure used 0.03 seconds

Fit (%) values from all runs:
Start n.1 Start n.2 Start n.3
    96.26     96.26     96.26

Tucker3 analysis with  2 x 1 x 2  components, gave a fit of 96.26 %
```

The next step of `T3` is about simple structure rotation by Orthomax (Kiers 1998) implemented in the function `varimcoco`. However, in the current analysis this rotation is not carried out because when $P = QR$ we can rotate to simple structure by simply transforming $\mathbf{G}_A$ of order (2×2) into the identity matrix of order two using the rotational freedom and compensating the transformation in the component matrices. Therefore, we ignore the Orthomax rotation by choosing relative weights equal to 0 (this does not rotate the current solution).

```
Specify (range of) relative weight(s) for A (default=0):
1:
Read 0 items

Warning: as the number of B-mode components is 1, no simple structure for B
will be sought (relative weight=0)

Specify (range of) relative weight(s) for C (default=0):
1:
Read 0 items
```

Prior to transforming $\mathbf{G}_A$ into the identity matrix, we permute and reflect the solution in such a way that the results are coincident with those in Timmerman (2001).

```
If you want to reflect/permute columns/rows, specify '1':
1: 1
Read 1 item
Give a vector for reflection of columns of A (e.g., 1 -1 -1 1 ..)
1: -1 1
Read 2 items
```

```
Give a vector with new order of columns of A (e.g., 3 1 4 2 ..)
1: 2 1
Read 2 items
Give a vector for reflection of columns of B (e.g., 1 -1 -1 1 ..)
1:
Read 0 items

Warning: the columns of B will not be reflected

Give a vector for reflection of columns of C (e.g., 1 -1 -1 1 ..)
1:
Read 0 items

Warning: the columns of C will not be reflected

Give a vector with new order of columns of C (e.g., 3 1 4 2 ..)
1: 2 1
Read 2 items
```

We then exit from `T3`. The resulting output is an object of class `list` called `t3bus`. In order to rotate the core to an identity matrix (denoted by $\mathbf{I}$) it is sufficient to premultiply it by its inverse matrix and to compensate such a transformation in $\mathbf{A}$ so as to obtain an equally well-fitting solution. Using Equation 1, this consists of postmultiplying $\mathbf{A}$ by $\mathbf{G}_A$:

$$\mathbf{A}\mathbf{G}_{\mathbf{A}}(\mathbf{C}^\top \otimes \mathbf{B}^\top) = \mathbf{A}\mathbf{G}_{\mathbf{A}}\mathbf{G}_{\mathbf{A}}^{-1}\mathbf{G}_{\mathbf{A}}(\mathbf{C}^\top \otimes \mathbf{B}^\top) = \tilde{\mathbf{A}}\mathbf{I}(\mathbf{C}^\top \otimes \mathbf{B}^\top), \tag{7}$$

being $\tilde{\mathbf{A}} = \mathbf{A}\mathbf{G}_{\mathbf{A}}$ and $\tilde{\mathbf{G}}_{\mathbf{A}} = \mathbf{I}$. In R, the transformation of $\mathbf{G}_A$ (denoted by `t3bus$core`) and that of $\mathbf{A}$ (`t3bus$A`) can be done as:

```
R> t3bus$A <- t3bus$A %*% t3bus$core
R> t3bus$core <- solve(t3bus$core) %*% t3bus$core
```

Just as in Timmerman (2001) we rescale the component matrices so that the maximum value of the second A-mode component, of the B-mode component and of the first C-mode component is equal to 1. The last three rescalings are compensated in the first A-mode component and in the second C-mode component.

```
R> t3bus$A <- t3bus$A %*% t3bus$core
R> colnames(t3bus$A) <- c("A1", "A2")
R> t3bus$core <- solve(t3bus$core) %*% t3bus$core
R> maxA2 <- max(t3bus$A[, 2])
R> t3bus$A[, 2] <- t3bus$A[, 2] / maxA2
R> maxB <- max(t3bus$B)
R> t3bus$B <- t3bus$B / maxB
R> maxC1 <- max(t3bus$C[, 1])
R> t3bus$C[, 1] <- t3bus$C[, 1] / maxC1
R> t3bus$A[, 1] <- t3bus$A[, 1] * maxB * maxC1
R> t3bus$C[, 2] <- t3bus$C[, 2] * maxA2 * maxB
```

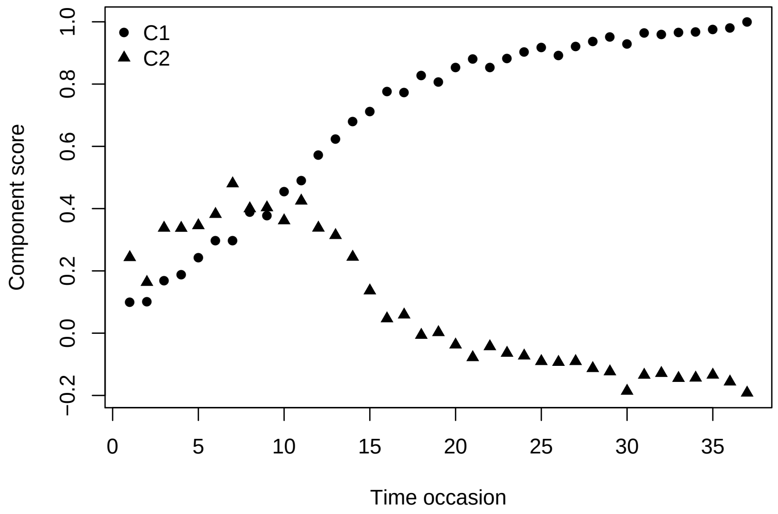

Figure 1: Component scores for the C-mode from T3 applied to the Bus data.

The so-obtained solution is very easy to interpret. To analyze the dynamics of the occasion component scores, we represent them in Figure 1.

```
R> plot(t3bus$C[, 1], type = "n", ylim = range(t3bus$C),
+    xlab = "Time occasion", ylab = "Component score")
R> points(t3bus$C[, 1], pch = 16)
R> points(t3bus$C[, 2], pch = 17)
R> legend("topleft", legend = c("C1", "C2"),  pch = c(16, 17), bty = "n")
```

The first C-mode component can be interpreted as the ‘General performance level’ because the scores are close to 0 in the beginning and close to 1 at the end of the testing time. The second C-mode component is more complex due to the negative scores in the second half of the occasions. It is interpreted approximately as the ‘Learning rate’ but the negative values do not indicate that the learning rate decreases in the end: it is due only to the rescaling procedure. The B-mode component (see printed values below) is connected with the ‘Difficulties of the items’. P (score 1.00) and S (score 0.99) have the highest scores. This is so because the judgments of these tests become positive faster than the other tests. The most difficult test is R with a component score equal to 0.58.

```
R> round(t3bus$B, 2)
```

```
    B1
L 0.91
P 1.00
Q 0.87
S 0.99
R 0.58
```

Taking into account that the core is an identity matrix, we can deduce that the first and second C-mode components are related, respectively, to the first and second A-mode components. The above connection helps us in the interpretation of the components A1 and A2 (see printed values below). Therefore, the pupils whose component scores are high are those who have a performance level (with respect to A1) and a learning rate (with respect to A2) above average. We can conclude that Pupil 4 is the best one: his (her) scores are the highest (scores 1.28 and 1.00). Pupil 5 follows. In fact, his (her) scores are the highest after the previous pupil. After them, Pupils 6 and 1 have almost the same high scores on the first component, but Pupil 1 has a very low score (the last but one) on the learning rate component. At last, in order, Pupils 3, 2 and 7 appear. Pupil 7 is the worst having the lowest scores (0.89 and −0.42).

```
R> round(t3bus$A, 2)
```

```
      A1    A2
n.1 1.06 -0.42
n.2 0.96 -0.30
n.3 0.99 -0.38
n.4 1.28  1.00
n.5 1.16  0.19
n.6 1.09 -0.01
n.7 0.89 -0.42
```

Finally, we are interested in assessing the statistical validity of the obtained component matrices. In fact, `T3` allows us to carry out a bootstrap procedure for computing confidence intervals for the current solution (Kiers 2004) by automatically calling the function `bootstrapT3`. As we are outside from `T3`, it is sufficient to write

```
R> set.seed(1)
R> t3busBoot <- bootstrapT3(BusN, t3bus$A, t3bus$B, t3bus$C, t3bus$core,
+    7, 5, 37, 2, 1, 2, 1e-6, 0, 0, 1, laba, labb, labc)
```

The analysis is based on 1000 (default value, but the user can make a different choice) bootstrap samples and matching via optimal transformation towards full solutions. Despite the very small sample size, the sample solution is a good estimate of the population parameters except for the second C-mode component. The bounds of the confidence intervals are given below (only the first ten rows are reported for the C-mode component matrix and details about the core are not reported because the transformation to the identity matrix can always be done).

```
R> round(t3busBoot$fpint, 2)
```

```
LB Fit (%) UB Fit (%)
     95.22      97.52
```

```
R> round(t3busBoot$B, 2)
```

```
  LB B1 UB B1
L  0.88  0.95
P  0.97  1.03
Q  0.84  0.89
S  0.97  1.02
R  0.46  0.65
```

```
R> round(t3busBoot$C[1:10, ], 2)
```

```
       LB C1 UB C1 LB C2 UB C2
Occ.1   0.04  0.13  0.03  0.28
Occ.2   0.05  0.13 -0.02  0.19
Occ.3   0.05  0.21 -0.06  0.41
Occ.4   0.10  0.22  0.05  0.37
Occ.5   0.15  0.29  0.00  0.40
Occ.6   0.20  0.35 -0.01  0.45
Occ.7   0.16  0.32  0.06  0.51
Occ.8   0.27  0.42  0.07  0.43
Occ.9   0.29  0.42  0.08  0.43
Occ.10  0.37  0.48  0.10  0.39
```

4. The function `CP` of the **R** package ThreeWay

This section is devoted to the function `CP` implementing CP, which is applied to two well-known three-way data sets.

4.1. `TV` data

The so-called TV data (Lundy, Harshman, and Kruskal 1989) (contained in `TV`) refer to the ratings on 15 American TV programs with respect to 16 bipolar scales given by a group of 30 students at the Ontario University. `TV` is an object of class `list` holding the data and the labels of the three modes. Lundy *et al.* (1989) analyzed the data by means of CP with three components. Here, we report the results found using `CP`. As we are interested in carrying out a stability check of the obtained solution, we need to rearrange the data so that the A-mode corresponds to the students, who are the C-mode. This is required because in the routines, if necessary, the random sample is assumed to be formed by the A-mode entities. It can be done using the function `permnew`.

```
R> data("TV")
R> TVdata <- TV[[1]]
R> labSCALE <- TV[[2]]
R> labPROGRAM <- TV[[3]]
R> labSTUDENT <- TV[[4]]
R> TVdata <- permnew(TVdata, 16, 15, 30)
R> TVdata <- permnew(TVdata, 15, 30, 16)
```

In this way the object of class `data.frame` called `TVdata` is such that the entities of the A-, B- and C-modes are, respectively, the students, the scales and the TV programs. We can now run CP.

```
R> CP(TVdata, labSTUDENT, labSCALE, labPROGRAM)

Specify the number of A-mode entities
1: 30
Read 1 item
Specify the number of B-mode entities
1: 16
Read 1 item
Specify the number of C-mode entities
1: 15
Read 1 item
```

Prior to fitting the model to the data, these are centered across TV programs and scales and normalized within the scales.

```
How do you want to center your array?
0 = none (default)
1 = across A-mode
2 = across B-mode
3 = across C-mode
12 = across A-mode and across B-mode
13 = across A-mode and across C-mode
23 = across B-mode and across C-mode
1: 23
Read 1 item
Data have been centered across B-mode
Data have been centered across C-mode

How do you want to normalize your array?
0 = none (default)
1 = within A-mode
2 = within B-mode
3 = within C-mode
1: 2
Read 1 item
Data have been normalized within B-mode
```

In order to compare the goodness of fit resulting from CP, the Convex Hull procedure is computed. By inspecting the output of the Convex Hull procedure (not reported here) we can see that the use of two components is suggested, but also extracting three components seems to be a reasonable choice. This is consistent with Lundy *et al.* (1989), who realize that the increase of fit passing from two to three components is not remarkable, but, at the same time, believe that a third, small but real, component exists. See also Bro and Kiers (2003).

```
How many components do you want to use?
1: 3
Read 1 item

Do you want to use constraints? If so, enter '1':
1:
Read 0 items
No constraints imposed

Specify convergence criterion (default=1e-6)
1: 1e-9
Read 1 item

If you want additional runs, specify how many (e.g., 4):
1: 2
Read 1 item

Specify the maximum number of iterations you allow (default=10000).
1: 50000
Read 1 item
```

The details about the algorithm (function `CPfunc`) for all the three runs can be found below.

```
Run no. 1
Candecomp/Parafac function value at Start is 7131.60689021309
Candecomp/Parafac function value is 3749.0362603670 after 6389 iterations
Fit percentage is 47.930051939348 %
Procedure used 35.89 seconds
Run no. 2
Candecomp/Parafac function value at Start is 7202.18335296148
Candecomp/Parafac function value is 3749.0362612989 after 6440 iterations
Fit percentage is 47.9300519264042 %
Procedure used 36.2 seconds
Run no. 3
Candecomp/Parafac function value at Start is 7208.70115055083
Candecomp/Parafac function value is 3749.0362611666 after 7521 iterations
Fit percentage is 47.9300519282422 %
Procedure used 42.31 seconds

Fit (%) values from all runs:
Start n.1 Start n.2 Start n.3
    47.93     47.93     47.93

Candecomp/Parafac analysis with  3  components, gave a fit of  47.93 %
Simple check on degeneracy: inspect matrix of triple congruences
        Comp.1  Comp.2  Comp.3
Comp.1  1.0000  0.0521 -0.9645
Comp.2  0.0521  1.0000 -0.1222
Comp.3 -0.9645 -0.1222  1.0000
```

The analysis of the results highlights that the algorithm always attains the purported global optimum, but takes a very long time to converge. Furthermore, the matrix of triple congruences shows that components 1 and 3 are highly collinear. If we inspected the component matrices, we could see that the elements of columns 1 and 3 are diverging. This is a typical pattern of CP degeneracy (see, for instance, Harshman and Lundy 1984; Stegeman 2006, 2007; Silva and Lim 2008; Rocci and Giordani 2010; Giordani and Rocci 2013). If so, the computational burden can be extremely high even if the data size is small. For this reason, the allowed maximum number of iterations must be given by the user (otherwise, default is 10,000). Degeneracies only occur with CP. T3 usually converges very quickly, as one can for instance realize inspecting the output of `T3` in Section 3.

A useful and recognized remedy to degeneracy is to considering a CP model with orthogonality constraints on one of the component matrices (Harshman and Lundy 1984). Here, we re-run `CP` with the same choices as described above and constraining the component matrix for the B-mode to be orthogonal.

```
Do you want to use constraints? If so, enter '1':
1: 1
Read 1 item
Digit:
1 = No constraints (default)
2 = Orthogonality constraints
3 = Zero correlations constraints
Specify the A-mode constraints:
1: 1
Read 1 item
Specify the B-mode constraints:
1: 2
Read 1 item
Specify the C-mode constraints:
1: 1
Read 1 item
```

The summary about the performance of the algorithm shows that the purported global optimum is always attained and the computation time dramatically decreases. Furthermore, the fit of the constrained CP model slightly decreases if compared with that of the unconstrained solution and all the obtained components are orthogonal.

```
Run no. 1
Candecomp/Parafac function value at Start is 7131.60689021309
Candecomp/Parafac function value is 3796.89450346562 after 230 iterations
Fit percentage is 47.265354118533 %
Procedure used 1.29 seconds
Run no. 2
Candecomp/Parafac function value at Start is 7198.76728242366
```

```
Candecomp/Parafac function value is 3796.89449182879 after 215 iterations
Fit percentage is 47.2653542801556 %
Procedure used 1.17 seconds
Run no. 3
Candecomp/Parafac function value at Start is 7200.20256213156
Candecomp/Parafac function value is 3796.89450650374 after 349 iterations
Fit percentage is 47.265354076337 %
Procedure used 1.97 seconds

Fit (%) values from all runs:
Start n.1 Start n.2 Start n.3
    47.27     47.27     47.27

Candecomp/Parafac analysis with  3  components, gave a fit of 47.27 %
Simple check on degeneracy: inspect matrix of triple congruences
       Comp.1 Comp.2 Comp.3
Comp.1      1      0      0
Comp.2      0      1      0
Comp.3      0      0      1
```

The solution is then normalized in such a way that the columns of the component matrices for the B- and C-modes have unit sum of squares (by means of the function `renormsolCP`).

```
If you want to scale components, specify '1':
1: 1
Read 1 item
What modes do you want to scale?
1 = B- and C-modes (scaling compensated in A)
2 = A- and C-modes (scaling compensated in B)
3 = A- and B-modes (scaling compensated in C)
1: 1
Read 1 item
```

As in `T3`, we can permute and reflect the solution. Here, we decide to reflect the first two A-mode components allowing us to obtain a final component matrix for the A-mode with all positive scores (we omit the details about the script). The resulting solution for the B- and C-mode component matrices is given below (negative scores of B pertain to the left side of the bipolar scale).

```
Solution for A, B and C after permutation and reflection.
B
                                              Comp.1 Comp.2 Comp.3
Thrilling-Boring                               -0.09   0.29  -0.12
Intelligent-Idiotic                             0.30   0.17   0.12
Erotic-Not Erotic                              -0.25  -0.06  -0.27
Sensitive-Insensitive                           0.03  -0.43   0.12
Interesting-Uninteresting                       0.00   0.37   0.26
```

```
Fast-Slow                                        -0.08  0.19 -0.29
Intellectually Stimulating-Intellectually Dull    0.27  0.20  0.25
Violent-Peaceful                                  0.08  0.04 -0.68
Caring-Callous                                    0.04 -0.47  0.02
Satirical-Not Satirical                          -0.46  0.09  0.18
Informative-Uninformative                         0.31  0.20  0.19
Touching-Leave Me Cold                           -0.07 -0.31  0.15
Deep-Shallow                                      0.21 -0.14  0.16
Tasteful-Crude                                    0.19 -0.31  0.02
Real-Fantasy                                      0.38  0.06 -0.06
Funny-Not Funny                                  -0.46  0.06  0.30
C
                            Comp.1 Comp.2 Comp.3
Mash                          0.14   0.03  -0.11
Charlie's angels              0.21   0.13   0.54
All in the family             0.20   0.04  -0.18
60 minutes                   -0.35  -0.19  -0.18
The tonight show              0.16  -0.23  -0.19
Let's make a deal             0.19   0.00   0.25
The Waltons                  -0.13   0.55  -0.23
Saturday night live           0.35  -0.31   0.11
News                         -0.39  -0.27   0.00
Kojak                         0.09  -0.04   0.37
Mork and Mindy                0.38   0.12  -0.10
Jacques Cousteau             -0.40  -0.10  -0.29
Football                     -0.07  -0.32   0.39
Little house on the prairie  -0.06   0.54  -0.18
Wild kingdom                 -0.31   0.04  -0.21
```

By inspecting the component matrices and following Lundy *et al.* (1989) we can interpret components 1, 2 and 3 as ‘Humor’, ‘Sensitivity’ and ‘Violence’, respectively.

We conclude our analysis by a stability check of the selected solution based on split-half analysis (function `splithalfCP`). It should be noted that CP offers the possibility to perform a bootstrap analysis calling function `bootstrapCP`, which is the CP counterpart of `bootstrapT3` described in Section 3. Since in the CP case a bootstrap analysis can be very time consuming, we prefer to consider a split-half analysis. Thus, two equally sized random (or based on odd vs. even sequence numbers) subsamples are drawn and two separate CP analyses using the same set-up considered for the full data are applied to these halves. Comparisons between the two solutions and with the full data solution are carried out. If the obtained components are stable, then the two solutions should give approximately the same results. Note that the function `splithalfCP` allows us to run a split-half analysis on the current solution or on a different one, specifying different choices with respect to those previously made.

```
If you want to carry out a STABILITY CHECK on current or different
solution, specify '1':
1: 1
Read 1 item
```

```
The splitting into two halves can be done randomly (default), or into odd
vs. even sequence numbers
If you prefer odd/even split, specify '1':
1: 1
Read 1 item
Splitting has been done into odd vs. even sequence numbers
You will now enter the split half procedure with the options specified
so far
However, you may want to modify certain choices here (rather than
rerunning the full Candecomp/Parafac)
Do you want to modify certain choices? If so, specify '1':
1:
Read 0 items
```

The output of the split-half procedure reports the component matrices resulting from the three analyses and the correlation and congruence (Tucker 1951) coefficients between corresponding columns of component matrices. For the sake of brevity, only the latter indices are given below from which we can state that especially the estimates of components 1 and 2 are stable.

```
Congruences for A in splits and in appropriate part of Afull
       SPL1 SPL2
Comp.1    1 1.00
Comp.2    1 0.99
Comp.3    1 0.99
Correlations for A in splits and in appropriate part of Afull
       SPL1 SPL2
Comp.1 1.00 0.96
Comp.2 1.00 0.96
Comp.3 0.94 0.84
Congruence values for B-mode component matrix
Comp.1 Comp.2 Comp.3
  0.87   0.82   0.68
Congruence values for C-mode component matrix
Comp.1 Comp.2 Comp.3
  0.96   0.89   0.42
```

4.2. `Kinship` data

The Kinship data (Rosenberg and Kim 1975; Arabie, Carroll, and Sarbo 1987) refer to 6 groups of students producing a partition of 15 kinship terms. For every group, the number of times in which two kinship terms are assigned to the same cluster is observed. The data array is such that the A- and B-mode entities (the kinship terms) coincide. It corresponds to a three-way proximity array (three-way, two-mode data), namely a collection of proximity data matrices, one for each group of students. This particular array can still be analyzed by means of CP. In this case, the CP model is usually referred to as the Indscal model to highlight that the model represents a three-way extension of the classical (two-way) multidimensional scaling. In their original paper, Carroll and Chang (1970) introduce such a model as a tool for summarizing a three-way array of scalar products through underlying components. Thus, the Indscal model can be understood as the CP model for arrays with symmetric slabs. When the CP model is fitted to proximity arrays, the symmetry of the slabs implies that two component matrices (in our case the matrices **A** and **B**) are the same. Moreover, the remaining matrix (in our case **C**) contains nonnegative scores. Both these aspects cannot be guaranteed with the CP algorithm, but are almost always satisfied in practice (see, for more details, ten Berge, Kiers, and Krijnen 1993). It is important to note that, since it is not explicitly required that $\mathbf{A} = \mathbf{B}$ and that **C** has nonnegative elements, this generally holds up to reflecting some of the columns of the matrices involved. As we shall see in the following, this is the reason why we will reflect the obtained solution. `Kinship` is an object of class `array`. It is loaded and then the function `CP` is run.

```
R> data("Kinship")
R> CP(Kinship, dimnames(Kinship)[[1]], dimnames(Kinship)[[2]],
+    dimnames(Kinship)[[3]])
```

We decide to preprocess the data by centering across the kinship terms (i.e., across the A- and B-modes) and normalizing within the groups of students (i.e., within the C-mode). The solution with three components is chosen as the optimal one. In fact, as we are going to see, the extracted components reveal the most relevant bases of meaning for partitioning the kinship terms. In other words, they allow us to discover how the groups of students distinguish the kinship terms.

```
How many components do you want to use?
1: 3
Read 1 item

Do you want to use constraints? If so, enter '1':
1:
Read 0 items
No constraints imposed

Specify convergence criterion (default=1e-6)
1: 1e-9
Read 1 item

If you want additional runs, specify how many (e.g., 4):
1: 2
Read 1 item

Specify the maximum number of iterations you allow (default=10000).
1: 50000
Read 1 item
```

We get the following output (omitting the details about the runs of the algorithms).

```
Fit (%) values from all runs:
Start n.1 Start n.2 Start n.3
```

```
    78.59     78.59     78.59

Candecomp/Parafac analysis with  3  components, gave a fit of  78.59 %
Simple check on degeneracy: inspect matrix of triple congruences
       Comp.1 Comp.2 Comp.3
Comp.1  1e+00  2e-04  0e+00
Comp.2  2e-04  1e+00  2e-04
Comp.3  0e+00  2e-04  1e+00
```

We can see that the purported global optimum is always attained. The goodness of fit of the model is very high and the solution is not degenerate (the components are approximately orthogonal). Prior to interpreting the extracted components, we normalize them in such a way that the columns of the component matrices for the A- and B-modes have unit sum of squares; furthermore, in order to have **A** = **B** and **C** with nonnegative elements we reflect the components accordingly (details of both steps are not reported here). We then get the following component matrices (only the matrices **B** and **C** are displayed). Since **C** contains nonnegative scores, the higher the generic score is, the more the group of students involved produces a partition of the kinship terms on the basis of meaning associated with the corresponding component.

```
Solution for A, B and C after permutation and reflection.

B
              Comp.1 Comp.2 Comp.3
Aunt           -0.26   0.37  -0.04
Brother         0.27  -0.17  -0.25
Cousin          0.03   0.35  -0.06
Daughter       -0.27  -0.23  -0.22
Father          0.27  -0.23  -0.24
Granddaughter  -0.28  -0.13   0.38
Grandfather     0.26  -0.14   0.43
Grandmother    -0.27  -0.14   0.42
Grandson        0.27  -0.13   0.38
Mother         -0.28  -0.23  -0.24
Nephew          0.26   0.36  -0.02
Niece          -0.26   0.36  -0.03
Sister         -0.27  -0.17  -0.25
Son             0.27  -0.23  -0.22
Uncle           0.26   0.37  -0.04
C
               Comp.1 Comp.2 Comp.3
First female   12.84   5.36   3.76
Second female   7.35   8.53   7.07
First male      8.91   7.40   6.75
Second male     9.64   6.19   5.74
Single female   2.12   9.78   8.63
Single male     4.79   8.88   7.48
```

Component 1 is sex-defined; the male relatives take positive scores and the female ones take negative scores. The score of cousin is close to zero. By looking at the first column of **C** we can conclude that sex is the basis of meaning for partitioning the kinship terms for the First female group and, to a lesser extent, for the Second male, First male and Second female groups. Component 2 distinguishes the kinship terms between the collateral relatives (Aunt, Cousin, Nephew, Niece and Uncle) with positive sign and the other ones (with negative sign), especially the nuclear family (Brother, Daughter, Father, Mother, Sister, Son) taking the lowest scores. Such a basis of meaning mainly refers to the Single female, Single male and Second female groups. Finally, Component 3 contrasts the direct ancestors and descendants with two generations removed (Granddaughter, Grandfather, Grandmother, Grandson) having positive scores with the nuclear family having negative scores. This basis of judgmental organization is emphasized for the Single female, Single male and Second male groups.

5. Final remarks

The most relevant features of the R package **ThreeWay** have been introduced by examples. **ThreeWay** offers a suite of about fifty functions for handling three-way arrays. Among them, emphasis has been paid to the two most famous techniques for summarizing three-way data, namely the CP and T3 methods, implemented in the functions `CP` and `T3`, respectively. Such functions carry out an interactive three-way analysis calling several additional functions to further extract relevant information from the data under investigation. The need for an interactive analysis arises because all the steps of a three-way analysis should not be done automatically. The process requires that the user inspects step-by-step the outputs of the functions and decides how to proceed accordingly. Nonetheless, if one is interested in running the CP and T3 algorithms the functions `T3func` and `CPfunc` can be used. For simulation experiments, the package also contains the functions `T3funcrep` and `CPfuncrep` which compute the solutions of the two methods in a single step. For instance, `CPfuncrep` is the same as `CPfunc` except that all printings are suppressed. Finally, also note that **ThreeWay** offers the T2 and T1 methods (functions `T2` and `T1`, respectively) in addition to the T3 method.

Acknowledgments

Paolo Giordani and Maria Antonietta Del Ferraro would like to thank the grant "Metodi e modelli statistici per l'analisi di traiettorie multivariate" (Progetti di ricerca Universitari 2009, Sapienza Università di Roma) for the financial support. Paolo Giordani is also grateful to the grant "Mixture and latent variable models for causal inference and analysis of socio-economic data" (FIRB Futuro in Ricerca 2012, Italian Ministry for Education, University and Research).

References

Arabie P, Carroll JD, Sarbo WSD (1987). *Three-Way Scaling and Clustering.* Sage Publications, Newbury Park.

Bro R (1998). *Multi-Way Analysis in the Food Industry – Models Algorithms and Applications.* Ph.D. thesis, University of Amsterdam.

Bro R, Kiers HAL (2003). "A New Efficient Method for Determining the Number of Components in PARAFAC Models." *Journal of Chemometrics*, **17**, 274–286.

Bro R, Smilde AK (2003). "Centering and Scaling in Component Analysis." *Journal of Chemometrics*, **17**, 16–33.

Bus AG (1982). "A Longitudinal Study in Learning to Read." Paper Presented at the 9th World Congress on Reading, July 26-30, Dublin, Ireland.

Carroll JD, Chang JJ (1970). "Analysis of Individual Differences in Multidimensional Scaling via an n-Way Generalization of Eckart-Young Decomposition." *Psychometrika*, **35**, 283–319.

Ceulemans E, Kiers HAL (2006). "Selecting among Three-Mode Principal Component Models of Different Types and Complexities: A Numerical Convex Hull Based Method." *British Journal of Mathematical and Statistical Psychology*, **59**, 133–150.

Del Ferraro MA, Kiers HAL, Giordani P (2014). ***ThreeWay**: Three-Way Component Analysis.* R package version 1.1.2, URL `http://CRAN.R-project.org/package=ThreeWay`.

Giordani P, Rocci R (2013). "Constrained Candecomp/Parafac via the Lasso." *Psychometrika*, **78**, 669–684.

Harshman RA (1970). "Foundations of the Parafac Procedure: Models and Conditions for an "Explanatory" Multimodal Factor Analysis." *UCLA Working Papers in Phonetics*, **16**, 1–84.

Harshman RA, Lundy ME (1984). "Data Preprocessing and the Extended PARAFAC Model." In HG Law, CWS Jr, JA Hattie, RP McDonald (eds.), *Research Methods for Multimode Data Analysis*, pp. 216–284. Praeger, New York.

Kiers HAL (1998). "Joint Orthomax Rotation of the Core and Component Matrices Resulting from Three-Mode Principal Components Analysis." *Journal of Classification*, **15**, 245–263.

Kiers HAL (2000). "Towards a Standardized Notation and Terminology in Multiway Analysis." *Journal of Chemometrics*, **14**, 105–122.

Kiers HAL (2004). "Bootstrap Confidence Intervals for Three-Way Methods." *Journal of Chemometrics*, **18**, 22–36.

Kiers HAL, der Kinderen A (2003). "A Fast Method for Choosing the Numbers of Components in Tucker3 Analysis." *British Journal of Mathematical and Statistical Psychology*, **56**, 119–125.

Kroonenberg PM (1983). *Three-Mode Principal Component Analysis: Theory and Applications.* DSWO, Leiden.

Kroonenberg PM (2008). *Applied Multiway Data Analysis.* John Wiley & Sons, Hoboken.

Kruskal JB (1977). "Three-Way Arrays: Rank and Uniqueness of Trilinear Decompositions, with Applications to Arithmetic Complexity and Statistics." *Linear Algebra and Its Applications*, **18**, 95–138.

Leibovici DG (2010). "Spatio-Temporal Multiway Decompositions Using Principal Tensor Analysis on k-Modes: The R Package **PTAk**." *Journal of Statistical Software*, **34**(10), 1–34. URL http://www.jstatsoft.org/v34/i10/.

Lundy ME, Harshman RA, Kruskal JB (1989). "A Two Stage Procedure Incorporating Good Features of Both Trilinear and Quadrilinear Models." In R Coppi, S Bolasco (eds.), *Multiway Data Analysis*, pp. 123–130. Elsevier, Amsterdam.

R Core Team (2013). *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/.

Rocci R, Giordani P (2010). "A Weak Degeneracy Decomposition for the CANDECOMP/PARAFAC Model." *Journal of Chemometrics*, **24**, 57–66.

Rosenberg S, Kim MP (1975). "The Method of Sorting as a Data-Gathering Procedure in Multivariate Research." *Multivariate Behavioral Research*, **10**, 489–502.

Silva VD, Lim LH (2008). "Tensor Rank and the Ill-Posedness of the Best Low-Rank Approximation Problem." *SIAM Journal on Matrix Analysis and Applications*, **30**, 1084–1127.

Smilde A, Bro R, Geladi P (2004). *Multi-way Analysis: Applications in the Chemical Sciences*. John Wiley & Sons, Chichester.

Stegeman A (2006). "Degeneracy in Candecomp/Parafac Explained for $p \times p \times 2$ Arrays of Rank $p + 1$ or Higher." *Psychometrika*, **71**, 483–501.

Stegeman A (2007). "Degeneracy in Candecomp/Parafac and Indscal Explained for Several Three-Sliced Arrays with a Two-Valued Typical Rank." *Psychometrika*, **72**, 601–619.

ten Berge JMF, Kiers HAL, Krijnen WP (1993). "Computational Solutions for the Problem of Negative Saliences and Nonsymmetry in INDSCAL." *Journal of Classification*, **10**, 115–124.

Timmerman ME (2001). *Component Analysis of Multisubject Multivariate Longitudinal Data*. Ph.D. thesis, University of Groningen.

Timmerman ME, Kiers HAL (2000). "Three-Mode Principal Components Analysis: Choosing the Numbers of Components and Sensitivity to Local Optima." *British Journal of Mathematical and Statistical Psychology*, **53**, 1–16.

Tucker LR (1951). "A Method for Synthesis of Factor Analysis Studies." *Personnel Research Section Report 984*, Department of the Army, Washington, DC.

Tucker LR (1966). "Some Mathematical Notes on Three-Mode Factor Analysis." *Psychometrika*, **31**, 279–311.

Affiliation:

Paolo Giordani
Department of Statistical Sciences
Sapienza University of Rome
P.le Aldo Moro, 5
00185 Rome, Italy
E-mail: `paolo.giordani@uniroma1.it`

Journal of Statistical Software `http://www.jstatsoft.org/`
published by the American Statistical Association `http://www.amstat.org/`

Volume 57, Issue 7 *Submitted:* 2011-11-03
April 2014 *Accepted:* 2013-11-15