Chemometrics and Intelligent Laboratory Systems 207 (2020) 104145

Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemometrics

Candecomp/Parafac with zero constraints at arbitrary positions in a loading matrix

Henk A.L. Kiers [a,*], Paolo Giordani [b]

[a] University of Groningen, the Netherlands
[b] Sapienza University of Rome, Italy

A B S T R A C T

When one interprets Candecomp/Parafac (CP) solutions for analyzing three-way data, small loadings are often ignored, that is, considered to be zero. Rather than just considering them zero, it seems better to actually model such values as zero. This can be done by successive modeling approaches as well as by a simultaneous modeling approach. This paper offers algorithms for three such approaches, and compares them on the basis of empirical data and a simulation study. The conclusion of the latter was that, under realistic circumstances, all approaches recovered the underlying structure well, when the number of values to constrain to zero was given. Whereas the simultaneous modeling approach seemed to perform slightly better, differences were very small and not substantial. Given that the simultaneous approach is far more time consuming than the successive approaches, the present study suggests that for practical purposes successive approaches for modeling zeros in the CP model seem to be indicated.

1. Introduction

Candecomp/Parafac (CP [1,2]; is a well-known and often applied method for the exploratory analysis of three-way data. The idea is to find a number (R) of components that jointly represent the data well. Specifically, let $\underline{\mathbf{X}}$ denote the (preprocessed) three-way data, with elements x_{ijk} for $i = 1, \ldots, I$, $j = 1, \ldots, J$ and $k = 1, \ldots, K$, where I, J and K denote the number of units for the three modes, labelled modes A, B and C, respectively. Then CP models the data as

$$x_{ijk} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} + e_{ijk}, \quad (1)$$

where a_{ir}, b_{jr}, and c_{kr} denote loadings for units from each of the three modes, on component r, $r = 1, \ldots, R$, and e_{ijk} denotes the residual or error terms. For ease of reference, the loadings are collected in the matrices **A**, **B** and **C**, and likewise error terms are collected in the three-way array $\underline{\mathbf{E}}$.

The idea of fitting the model to data is that the R components jointly give a simple summary of the data. The quality of the summary can be assessed by means of the model fit, expressed as $1-\sum e_{ijk}^2/\sum x_{ijk}^2$. Each component can be interpreted on the basis of the loadings. So, a component may represent a particular pattern of scores on modes A, B and C, the product of which represents the contribution of this component to the data. For instance, in sensory analysis, the data may represent scores of I experts (mode A), on J attributes (mode B) of K products (mode C). In such cases, the A-mode loadings may indicate that one component is strongly related to a particular subset of experts, while another component is related to a different or partly different subset of experts. The B-mode loadings may indicate that a component strongly relates to a particular set of attributes, and less or not at all to other attributes. The C-mode in turn may indicate that a component strongly relates to a subset of products. Such interpretations will then be based on the heights of the loadings, and small loadings (positive or negative) of a unit on a component will then typically be considered as absence of importance of this unit for the component at hand. Thus, an A-mode component will be interpreted mainly in relation to the experts that do *not* have small loadings, and experts with small loadings will be ignored. However, then the question arises, what can be considered small enough? One way of dealing with this is to consider that, ignoring small loadings in the interpretation, actually boils down to considering them equal to zero. But when loadings are tacitly considered to be zero, this implies that the loadings are *implicitly adjusted* while interpreting the results. Hence it seems more reasonable to *explicitly adjust* the loadings to zero, and then reassess the fit. If the fit does not change much, indeed, the loadings apparently were small enough to be modified into zero without much harm. So the rationale of this approach is to see if a model with zero loadings, which conforms well to how results are interpreted in practice (i.e., not taking into account low values at all), indeed gives sufficient fit of the model, and thus warrants this simplified interpretation.

Above, it has been described how, when interpreting loadings as

* Corresponding author.
E-mail addresses: h.a.l.kiers@rug.nl (H.A.L. Kiers), paolo.giordani@uniroma1.it (P. Giordani).

https://doi.org/10.1016/j.chemolab.2020.104145
Received 13 December 2019; Received in revised form 17 August 2020; Accepted 28 August 2020
Available online 15 October 2020

being so small that they can actually be considered zero, one should actually set these loadings to 0 and reassess the fit. However, there is room for improvement now, since the other model parameters are not optimal *given the zero loadings*. So, an obvious next step is to find the other loadings such that *given the zero loadings* the model optimally fits the data[1]. To our knowledge this approach has not formally been developed any further. Instead alternative approaches have been developed that take yet another step: Rather than starting from a CP model in which smallest values are set to zero, the CP model is fitted to data with a prespecified *number* of loadings (p) constrained to zero, while their locations are not specified in advance. The idea is that the locations are established such that the model with zero loadings at these locations yield the best possible fit. In the literature various proposals have been made for such constrained CP methods. In the present paper, we restrict ourselves to the situation where a number of loadings in only one of the three modes will be constrained to zeros. For this situation, the approach where in each row all but one of the loadings are constrained to zero has been studied repeatedly (e.g., [3–6]. These approaches are usually denoted as clustering approaches, or more appropriately, nonoverlapping clustering or partitioning approaches. In this case, the model assumes different underlying components for the A-mode units belonging to different clusters. In fact, a loading equal to zero identifies no importance for the component involved. In this respect, the method aims at discovering the structural differences among the A-mode units.

A more general approach of zero constrained CP is obtained when the limitation of "only one nonzero loading per row" is alleviated. Then the idea is to find model estimates for the loadings in (say) **A** such that p of them are zero, and the others are found such that the fit is optimal. Thus, overlapping clusters are allowed in this approach, and, as a consequence, this approach allows for discovering not only structural differences among A-mode units, but also structural similarities. Structural similarities emerge at its most complete when a column of **A** does not contain zeros at all, implying the importance of all the A-mode units for the component at hand. However, it also allows for distinguishing different levels of model complexity for the A-mode units. For further details on the overlapping clustering approach in component-based models, see [7].

The overlapping clustering approach applied to CP has been considered by [4]; but, up to our knowledge, never elaborated upon. The purpose of the present paper is to elaborate this particular method, and to compare its performance to the simplified approach mentioned earlier, in which the location of elements to be set to zero is determined by CP (i.e., by locating the smallest p loadings in the CP solution for **A**), and next the other parameters are estimated optimally.

This paper starts in Section 2 with a description of the zero constrained CP methods to be considered here. As will be seen, in line with the above two methods, we actually develop three approaches, two of which have the additional constraint that all rows will have at least one non-zero loading. The algorithms for minimizing these methods will be described in detail, and will be proven to yield monotone convergence of the function value. The methods will be illustrated on an empirical example in Section 3. Next in Section 4, the performance of the methods will be compared by means of a simulation study. Comparison criteria are recovery of the underlying structure, sensitivity to local optima, and computation time. The paper will be concluded by a discussion in Section 5.

2. Three methods for zero constrained CP

2.1. Successive approach to fitting CP with zero constraints at locations of smallest p loadings in A

The first approach to fitting CP models with the constraint that p loadings in **A** are zero works as follows: First a CP analysis is carried out, then the smallest p loadings are identified, and next the CP model is refitted subject to the constraint that these p loadings are zero. In the present paper, we use least squares fitting procedures. In other words, we aim to minimize the loss function

$$f(A,B,C) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(x_{ijk} - \sum_{r=1}^{R} a_{ir}b_{jr}c_{kr}\right)^2. \quad (2)$$

For minimizing (2), we can use any CP algorithm. In the present paper, we use the original algorithm independently proposed by [1,2]. It consists of generating starting values for **A**, **B**, and **C**, then iteratively updating the matrices **A**, **B**, and **C**, while keeping the others fixed, and continuing doing this until the function value can be considered converged. Each step of this so called alternating least squares (ALS) algorithm decreases the loss function f, and because $f \geq 0$, this process must converge to a stable function value. The solution associated with the converged function value usually refers to a stable solution in terms of **A**, **B**, and **C**, which can be considered a local or global minimum of f. Sometimes, however, the CP algorithm leads to the so called degeneracy, in which the parameter matrices **A**, **B**, and **C** do not converge, but (at least in one of the matrices) grow unboundedly. In such cases, two columns in, e.g., **A** do not only get ever larger values, but also become increasingly proportional to each other, as can be expressed by the cosine between these two columns approaching ± 1; at the same time, the associated columns in, e.g., **B**, and in **C**, display the same behavior of increasing proportionality, except that in one of the three matrices or in all three, these columns become inversely proportional, that is their cosine approaches -1. This phenomenon has been studied in depth by many researchers (e.g., see [8–13]). In order to detect it, one should either closely inspect the solution itself, or compute the triple products of cosines between all pairs of columns, and if the smallest one is approaching -1, this indicates a degenerate solution.

The updating steps for matrices **A**, **B**, and **C** have been described well in the literature, but as we need these steps for our constrained optimization purpose, the one for updating **A** of size $I \times R$ will now be explained here, and expressed in matrix terms. To update **A** considering the other parameters fixed, we have to minimize

$$f(\mathbf{A}) = ||\mathbf{X} - \mathbf{A}\mathbf{F}'||^2, \quad (3)$$

where **X** denotes the $I \times JK$ matrix with frontal slices of $\underline{\mathbf{X}}$ placed next to each other, and **F** denotes the $JK \times R$ matrix with elements $b_{jr}c_{kr}$, ordered in column r as $b_{1r}c_{1r}$, …, $b_{Jr}c_{1r}$, $b_{1r}c_{2r}$, …, $b_{Jr}c_{2r}$, etc. Differently put, the columns of **F** consist of the Kronecker products $c_r \otimes b_r$, $r = 1,..,R$. To minimize (3) over **A** is a simple multiple regression problem, the minimum of which is found for $\mathbf{A} = \mathbf{X}\mathbf{F}(\mathbf{F}'\mathbf{F})^{-1}$, provided that **F** has full column rank, as is usually the case. Hence, to update **A**, given **B** and **C**, we have to compute **F** from **B** and **C**, and next update **A** as $\mathbf{A}^{\mathbf{u}} = \mathbf{X}\mathbf{F}(\mathbf{F}'\mathbf{F})^{-1}$, and because $f(\mathbf{A}^{\mathbf{u}})$ gives the minimum of $f(\mathbf{A})$ we will have $f(\mathbf{A}^{\mathbf{u}},\mathbf{B},\mathbf{C}) \leq f(\mathbf{A},\mathbf{B},\mathbf{C})$. For updating **B** and **C** analogous updating steps can be derived, each also decreasing f. As mentioned, repeating this until converge will lead to a local or global minimum (ignoring degeneracies for the moment). To increase the chance of finding the global rather than just a local minimum, this procedure is typically repeated a number of times with different starting values, and the best solution is then kept (hopefully giving the global minimum of f). Here we typically use one rationally started run and ten randomly started runs.

Having described the CP fitting approach, we are now in a position to describe the procedure of next setting loadings in **A** to zero, and refitting the CP model keeping these values equal to zero. The basic set up then is:

Step 1. Find the matrices **A**,**B**,**C** that minimize f(**A**,**B**,**C**) by means of 11 runs of the CP algorithm.

Step 2. Locate the p (in absolute sense) smallest loadings in **A** and indicate these in the $I \times R$ binary matrix **W** by 0, while the other elements are indicated by 1.

[1] This idea may be fairly old, and at least was mentioned in 2012 by Nikos Sidiropoulos in a discussion session at the TRICAP 2012 meeting in Bruges.

Step 3. Find the matrices **A**,**B**,**C** that minimize $f(\mathbf{A},\mathbf{B},\mathbf{C})$ subject to the constraint that $\mathbf{A} = \mathbf{W}*\mathbf{A}$, where * denotes the elementwise product; alternatively we could phrase this as "subject to the constraint that those p loadings in **A** that are at the same position as where **W** has 0's must be constrained to zero".

To perform Step 2 is straightforward: First collect the absolute values of all elements of **A** in a vector of size IR, and make a vector **w** of the same size, filled with 1's. Then sort them, locate the position of the p smallest elements, and set the corresponding elements in **w** to 0. Finally, reorganize **w** into the matrix **W** of size $I \times R$.

Step 3 consists of minimizing $f(\mathbf{A},\mathbf{B},\mathbf{C})$ subject to the constraint $\mathbf{A} = \mathbf{W}*\mathbf{A}$. For this purpose we propose to use a variant of the ALS algorithm for CP, in which only the updating procedure of **A** has to be adjusted. Specifically, to update **A**, we now have to minimize $f(\mathbf{A}) = ||\mathbf{X}-\mathbf{A}\mathbf{F}'||^2$ subject to $\mathbf{A} = \mathbf{W}*\mathbf{A}$. This problem can be solved by realizing that $||\mathbf{X}-\mathbf{A}\mathbf{F}'||^2 = \sum_i ||\mathbf{x}_i'-\mathbf{a}_i'\mathbf{F}'||^2$ where $\mathbf{x}_i'$ and $\mathbf{a}_i'$ denote the ith row of **X** and **A**, respectively. Hence to minimize $||\mathbf{X}-\mathbf{A}\mathbf{F}'||^2$ we have to minimize $f_i(\mathbf{a}_i) = ||\mathbf{x}_i'-\mathbf{a}_i'\mathbf{F}'||^2$ independently, for $i = 1,..,I$. This is simply solved as follows (e.g., see [14]. Let $\mathbf{v}_i$ consist of all nonzero values in $\mathbf{a}_i'$ and let $\mathbf{G}_i$ be the matrix containing the associated columns of **F**, then $\mathbf{v}_i'\mathbf{G}_i' = \mathbf{a}_i'\mathbf{F}'$, and the problem to solve boils down to minimizing $f_i(\mathbf{v}_i) = ||\mathbf{x}_i'-\mathbf{v}_i'\mathbf{G}_i'||^2$, which is solved by taking $\mathbf{v}_i' = \mathbf{x}_i'\mathbf{G}_i\,(\mathbf{G}_i'\mathbf{G}_i)^{-1}$, and hence the vector $\mathbf{a}_i$ minimizing $f_i(\mathbf{a}_i)$ is obtained by taking its nonzero elements equal to the subsequent elements of the vector $\mathbf{v}_i$ minimizing $f_i(\mathbf{v}_i)$. In case *all* elements of $\mathbf{a}_i$ equal 0, obviously, no update of this row has to be carried out. Thus, each row of **A** can be updated, and the resulting update of the matrix **A** will give the minimum of $f(\mathbf{A}) = ||\mathbf{X}-\mathbf{A}\mathbf{F}'||^2$ subject to $\mathbf{A} = \mathbf{W}*\mathbf{A}$. Given the fact that the unconstrained CP solution gives the best possible fit, we assume that starting our algorithm with the non-zero loadings from this solution will usually lead to the global optimum of the constrained fitting problem, and we will not use different starting configurations. We denote this first method as "CP_Succ".

2.2. Successive approach to fitting CP with zero constraints at locations of smallest p loadings in A, while not allowing the occurrence of zero rows in A

In Section 2.1, we described our approach to fitting CP models with the constraint that p loadings in **A** are zero. However, it is possible that this approach leads to one or more rows consisting of only zero values. This implies that the data for the associated A-mode unit i are modelled to be exactly zero, hence, say, consisting of only "noise". Because this to us seems to lead to rather unrealistic solutions, we developed a variant of the above approach in which zero rows in **A** are precluded. The set up for this variant is basically the same as in Section 2.1, except that Step 2 now is replaced by

Step 2'. First locate the I rowwise (in absolute sense) biggest loadings. Among the remaining $(I-1)R$ loadings locate the p smallest loadings in **A** and indicate these in the binary matrix **W** by 0, while the other elements are indicated by 1.

To perform Step 2′ is only slightly more complex than Step 2: First determine the row-wise absolute values of **A**. Indicate their locations by 1's in the associated locations in matrix **W**, while the others are temporarily filled with 0's. Collect the absolute values of the remaining $(I-1)R$ elements in a vector of size $(I-1)R$, and make a vector **w** of the same size, filled with 1's. Then sort them, locate the position of the p smallest elements, and set the corresponding elements in **w** to zero. Finally, replace the $(I-1)R$ temporarily zero values in **W** by the subsequent values in **w**. We denote this second method as "CP_Succ_NoZRows".

2.3. Fitting CP with zero constraints at optimal locations of smallest p loadings in A, while not allowing the occurrence of zero rows in A

In Sections 2.1 and 2.2, we described approaches to fitting CP models with the constraint that p loadings in **A** are zero. Here, the location of the zero loadings was based on the CP solution for **A**. However, it is possible that solutions exist that give a better CP fit, while also having p loadings equal to zero, and also not allowing zero rows. To search such solutions, we have to minimize $f(\mathbf{A},\mathbf{B},\mathbf{C})$ subject to the mentioned constraints without fixing the location of the zero loadings. In other words, we have to minimize $f(\mathbf{A},\mathbf{B},\mathbf{C})$ subject to $\mathbf{A} = \mathbf{A}*\mathbf{W}$, where now **W** itself is *not fixed*, but denotes any binary matrix with p zeros provided that each row has at least one 1. We denote the set of such matrices **W** as W_p. In order to do so, we propose the following variant of the ALS algorithm for CP. As in the CP algorithm, **A**, **B**, and **C** will be updated, while keeping the others fixed, and this will be continued until the function value can be considered converged. The CP updating procedures for **B** and **C**, given the other matrices, are optimal, because they actually minimize $f(\mathbf{A},\mathbf{B},\mathbf{C})$ given the other parameter matrices. What remains is to find an update for **A** satisfying the zero loading constraints and minimizing or at least decreasing (3), that is $f(\mathbf{A}) = ||\mathbf{X}-\mathbf{A}\mathbf{F}'||^2$, subject to $\mathbf{A} = \mathbf{A}*\mathbf{W}$ across all binary **W** in set W_p. Because minimizing $f(\mathbf{A})$ subject to this constraint does not seem to have a closed form solution, we propose to update **A** by a majorization step (e.g., see [15,16]) which will decrease $f(\mathbf{A})$. For this purpose, we rewrite $f(\mathbf{A})$ as

$$f(\mathbf{A}) = c_1 + tr(-2\mathbf{F}'\mathbf{X}')\mathbf{A} + tr(\mathbf{A}\mathbf{F}'\mathbf{F}\mathbf{A}'), \tag{4}$$

where c_1 denotes a constant, and we can identify this as a special case of the general function for which [16], see also [17], described a majorizing function which is easier to minimize than the original function. Specifically, applying the general result in (11b) of [17] (with correction of the typo $\mathbf{X}^{\mathbf{c}}$, which should be $\mathbf{X}^{\mathbf{c}}$) we obtain for our special case that

$$f(A) \le m(\mathbf{A}) = c_2 + \alpha||\mathbf{H}-\mathbf{A}||^2, \tag{5}$$

where c_2 denotes a constant, α denotes the highest eigenvalue of $\mathbf{F}'\mathbf{F}$, and $\mathbf{H} = \mathbf{A}^{\mathbf{c}}-(2\alpha)^{-1}(-2\mathbf{X}\mathbf{F}+2\mathbf{A}^{\mathbf{c}}\mathbf{F}'\mathbf{F}) = \mathbf{A}^{\mathbf{c}}+\alpha^{-1}\mathbf{X}\mathbf{F}-\alpha^{-1}\mathbf{A}^{\mathbf{c}}\mathbf{F}'\mathbf{F}$, where $\mathbf{A}^{\mathbf{c}}$ denotes the current (not yet updated) version of the matrix **A**. As shown by [16]; if $\mathbf{A}^{\mathrm{upd}}$ is chosen such that it minimizes $m(\mathbf{A})$, then $f(\mathbf{A}^{\mathrm{upd}}) \le f(\mathbf{A}^{\mathrm{c}})$.

The problem of minimizing $m(\mathbf{A})$ subject to $\mathbf{A} = \mathbf{A}*\mathbf{W}$ across all binary **W** in set W_p is easy. It essentially boils down to applying Step 2' in Section 2.2 to H for finding **W** and then taking $\mathbf{A}^{\mathrm{upd}} = \mathbf{W}*\mathbf{H}$. To see why this is so, one should first realize that $||\mathbf{H}-\mathbf{A}||^2 = \sum_{ir}(h_{ir}-a_{ir})^2$. Clearly, for any **W** the optimal values of **A** subject to $\mathbf{A} = \mathbf{A}*\mathbf{W}$ are given by $a_{ir} = h_{ir}$ if $w_{ir} = 1$, and $a_{ir} = 0$ if $w_{ir} = 0$ (which effectively comes down to $\mathbf{A}=\mathbf{H}*\mathbf{W}$). Hence it remains to minimize

$$\begin{aligned}\sum_{ir}(h_{ir}-a_{ir})^2 &= \sum_{ir}w_{ir}(h_{ir}-h_{ir})^2 + \sum_{ir}(1-w_{ir})(h_{ir}-0)^2\\ &= \sum_{ir}(1-w_{ir})h_{ir}^2\end{aligned} \tag{6}$$

over all binary **W** in set W_p. This in turn is equivalent to *maximizing* $\sum_{ir}w_{ir}h_{ir}^2$ over all binary **W** in set W_p. The best way of satisfying the requirement that **W** has at least one 1 in each row, is to make sure that the I elements in **W** that correspond to the rowwise (in absolute sense) largest values of **H** are set to 1, because this gives the highest thus attainable sum of squares of $\sum_{ir}w_{ir}h_{ir}^2$. In the special case where $p = IR-I$, the solution then is to set the other elements in **W** equal to zero. This is the interesting special case in which the units of mode **A** can be considered to be partitioned in nonoverlapping clusters. Then the method is equivalent to the methods by [4,18]. In general, however, p will be chosen smaller than $IR-I$. In that case, of the remaining $IR-I$ elements, $IR-I-p$ should still be set to 1, in such a way that $\sum_{ir}w_{ir}h_{ir}^2$ is maximally increased. This is trivially attained by locating among the remaining elements in **H**, the $IR-I-p$ elements that are highest in absolute sense, and setting the corresponding elements in **W** equal to 1. This gives the highest value of $\sum_{ir}w_{ir}h_{ir}^2$, where in **W** in total $I + IR-I-p = IR-p$ elements have been set to 1, and the remaining p have been set to zero, and the constraint of not allowing for zero rows has been satisfied. Having obtained the optimal **W**, the optimal **A** is now obtained as **H***W, and this minimizes the majorization function $m(\mathbf{A})$. As mentioned above, updating **A** in this way,

will decrease the loss function $f(\mathbf{A})$.

It may happen that in this way a matrix **A** is obtained that has one or more zero columns. In that case, we know that it will usually be suboptimal because it would effectively lead to a model with one or more components less, and such a solution is usually suboptimal. Moreover, updates of **B** and **C** cannot be computed because these would then involve the inverse of rank deficient matrices. Therefore, if the computed update of **A** has one or more zero columns, it has been decided not to update **A** at that stage.

With this adjusted updating for **A**, an iterative algorithm can be set up that first initializes the matrices **A**, **B**, and **C**, and then in turn updates **A**, **B**, and **C**, while keeping the others fixed, and continues doing this until the function value can be considered converged. A monotone decrease of the function is again guaranteed, because each update will decrease the loss function. At convergence, one can expect to have found at least a local minimum. Practical experience demonstrated that different starting configurations can very well lead to different solutions, so it is recommended to use good starting configurations, and/or a large number of runs from different starting positions. One option for a (hopefully) good starting configuration could be the solution from the method described in Section 2.2. We denote this third method as "CP_Simult".

2.4. How to choose the number of components R and the number of zero loadings p

To choose the number of components R one may simply analyze the data by CP using different values for R up to a reasonable maximum, and then compare fit values, and see after what value of R further increases seriously level off. If desired, one can make a plot of fit against the number of components and look for an 'elbow' in the plot, thus carrying out a kind of 'scree test' [19]. This may, however, not always lead to a clear conclusion, and will require subjective judgement to decide on a good compromise between parsimony and fit.

Each of the methods discussed above search CP solutions where a fixed number (p) of loadings in **A** is constrained to be zero. However, in practice, it will usually not be clear *a priori* what number of loadings one would wish to set equal to zero. Since the aim of setting loadings to zero is to facilitate interpretation while the fit of the model is still good enough, it is clear that the choice for p must be such that a good compromise is found between parsimony and fit. Here parsimony is related directly to p. The higher p, the more parsimonious is the solution. Please note that, when increasing p to $p+1$ this does not mean that the same p loadings will be zero in both solutions, in other words, solutions are not necessarily nested. As mentioned in the introduction, the fit is defined as 1 minus the loss divided by the sum of squares of the data, hence as

$$Fit = 1 - \frac{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\left(x_{ijk} - \sum_{r=1}^{R} a_{ir}b_{jr}c_{kr}\right)^2}{\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} x_{ijk}^2}. \quad (7)$$

This is often multiplied by 100 to obtain the fit percentage.

We cannot give a general rule for striking a good compromise between parsimony and fit, because what is good is a subjective matter, depending on a researcher's judgment and goals. Therefore, it is recommended to plot the fit percentage against p. A typical such plot can be found in Fig. 1; in this case $I = 30$, $R = 3$, and the data were analyzed by means of the method in Section 2.2, varying p from 1 to 60. Clearly, when the number of zeros increases, the fit decreases, first only very slowly, but after $p = 20$ a bit faster, and the amount of decrease gradually increases. As said, the choice of a compromise is to be made by the researcher analyzing the data, but if a clear drop would occur, so that a kind of elbow can be recognized, then such a phenomenon may help selecting the compromise, because increasing p after the elbow will give relatively large fit decreases. Procedures for automatically finding such elbows have been proposed. In this context, it would consist of finding the point where the ratio of $(f_{p-1}-f_p)/(f_p-f_{p+1})$ is minimal, because this is where

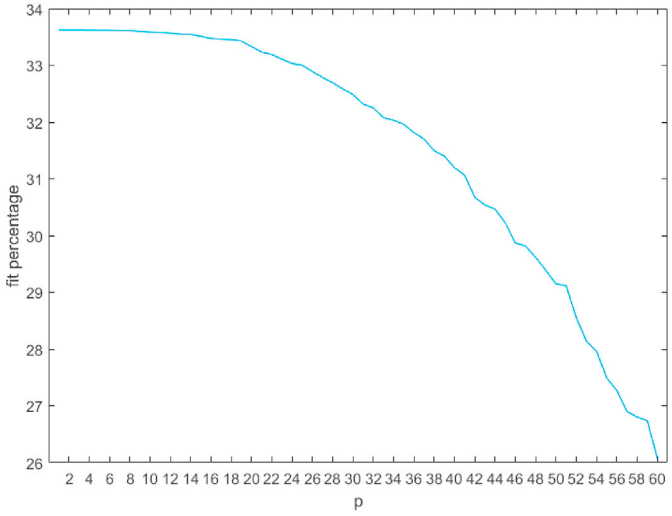

Fig. 1. Example of a plot of fit percentage against p.

the biggest change in direction of the curve takes place. This is a version of Cattell's scree test, and a special case of the Convex Hull procedure by [20]. The procedure should be handled with care, because at the beginning and/or end of the range it is well possible that instabilities in the ratio occur, see [20] for details.

In case it does not become clear what value for R should be used, it would be good to repeat the procedure for, for instance, two values of R, and determine good values for p for both cases. Then, to decide between the resulting solutions, it would be wise to actually compare the loading matrices, in order to see which solution is easiest to interpret. Indeed, as a reviewer suggested, taking more components could then result in a simpler structure (i.e., with fewer nonzeros per component). Whether or not this is more attractive depends on the goal of the analysis. If the focus is more on similarity, then taking fewer components would seem more attractive; if the focus is more on differences, taking more components seems more attractive.

3. Applying three methods for zero constrained CP on an empirical data set

To illustrate the methods we reanalyzed the Cider data[2] set analyzed by [18]. The data pertain to judgements of ten varieties of cider, by seven assessors on ten attributes. The 10x10x7 three-way array was structured as attributes (A) by products (B) by panelist (C), and our interest was, like Wilderjans and Cariou's, in clustering the attribute (mode A) loadings. The attributes were: sweet, acid, bitter, astringency, odor strength, pungent, alcohol, perfume, intensity, and fruity. The same preprocessing was used as by Wilderjans and Cariou, consisting of centering across products, and a particular scaling within the slices for the panelists (for details see p.48–49). The result of our preprocessing yielded the very same data as the preprocessed ones received from Tom Wilderjans.

We first analyzed the data by CP, for $R = 1,\ldots,5$, yielding fit percentages of: 41.2%, 53.4%, 58.5%, 63.5%, and 68.4%, respectively. Clearly, the fit strongly increased from $R = 1$ to $R = 2$ (12%), and after that consistently by roughly 5%. Therefore, the $R = 2$ solution seemed the one giving the last big fit increase after which the increase had levelled off, hence it seemed a good compromise between parsimony and fit. We report this CP solution in Table 1, with **B** and **C** normalized such that each column has a sum of squares equal to 1.

Next we analyzed the data by CP_Succ, CP_Succ_NoZRows, and

[2] Tom Wilderjans is gratefully acknowledged for sending the data to us in Matlab format, both the raw data and the preprocessed data.

Table 1
Results for Cider data.

Attributes	CP fit = 53.4%		CP_Succ, $p = 7$, fit 52.9%		CP_Succ_NoZRows $p = 5$, fit = 53.2%		CP_Simult, $p = 6$, fit = 52.9%		CP_Simult, $p = 10$ (=nonoverlap), fit = 49.5%	
	Comp.1	Comp.2	Comp.1	Comp.2	Comp.1	Comp.2	Comp.1	Comp.2	Comp.1	Comp.2
Intensity	−0.14	−6.41	0	−6.30	0	−6.26	0	−6.25	0	−6.38
Sweet	−12.37	−8.54	−14.09	−10.57	−12.92	−8.93	−10.42	−4.44	−9.45	0
Acid	1.31	1.20	0	0	0.58	0	0.84	0	1.10	0
Bitter	3.76	−0.51	4.08	0	4.07	0	3.67	0	2.89	0
Astringency	2.17	1.01	0	0	1.53	0	1.63	0	1.61	0
Odor strength	0.94	−3.77	0	−4.43	0	−4.40	0	−4.42	0	−4.03
Pungent	4.41	7.10	5.34	8.04	4.86	7.48	3.96	5.93	0	4.95
Alcohol	9.97	6.49	11.36	8.12	10.42	6.83	8.34	3.45	7.43	0
Perfume	−11.13	−4.72	−12.48	−6.37	−11.44	−4.92	−8.83	0	−8.45	0
Fruity	−13.95	−11.29	−16.02	−13.64	−14.67	−11.83	−11.80	−6.76	−10.53	0
Products										
Cider 1	−0.15	0.34	−0.18	0.33	−0.15	0.32	−0.04	0.30	0.09	0.36
Cider 2	0.13	0.23	0.09	0.22	0.12	0.22	0.26	0.20	0.36	0.28
Cider 3	−0.31	0.26	−0.31	0.26	−0.32	0.28	−0.29	0.30	−0.21	0.25
Cider 4	−0.39	0.13	−0.38	0.15	−0.40	0.16	−0.43	0.16	−0.42	0.02
Cider 5	0.30	−0.14	0.29	−0.16	0.30	−0.15	0.29	−0.12	0.27	−0.11
Cider 6	0.39	−0.08	0.36	−0.10	0.39	−0.11	0.44	−0.08	0.45	0.06
Cider 7	0.23	−0.01	0.21	−0.02	0.23	−0.02	0.28	−0.05	0.30	−0.02
Cider 8	−0.45	0.22	−0.45	0.23	−0.44	0.22	−0.45	0.19	−0.41	0.11
Cider 9	0.43	−0.81	0.49	−0.81	0.43	−0.80	0.20	−0.83	−0.12	−0.83
Cider 10	−0.16	−0.13	−0.14	−0.11	−0.16	−0.12	−0.26	−0.07	−0.31	−0.11
Panelists										
1	0.40	0.37	0.39	0.37	0.40	0.37	0.40	0.33	0.43	0.23
2	0.41	0.40	0.41	0.40	0.41	0.40	0.42	0.39	0.41	0.40
3	0.37	0.41	0.38	0.41	0.37	0.42	0.36	0.46	0.35	0.48
4	0.32	0.30	0.32	0.29	0.32	0.29	0.33	0.30	0.32	0.36
5	0.32	0.22	0.31	0.24	0.31	0.23	0.32	0.18	0.30	0.18
6	0.41	0.51	0.42	0.50	0.41	0.51	0.40	0.52	0.42	0.50
7	0.40	0.37	0.40	0.37	0.40	0.37	0.40	0.37	0.40	0.38

CP_Simult. In all cases we used $R = 2$ components; all solutions were aligned with the CP solution, using permissible permutations and scalings. We ran CP_Succ varying p from 1 to 10, yielding fit percentages of 53.4, 53.4, 53.4, 53.3, 53.2, 53.1, 52.9, 51.1, 49.0 and 48.1%, respectively. Clearly, taking up to $p = 7$ zeros hardly affected the fit, but after that the fit decreased much more quickly. Therefore, we selected the $p = 7$ solution as the best compromise between parsimony and fit. The loadings are given in Table 1.

Analogously, running CP_Succ_NoZRows with p from 1 to 10, yielded fit percentages of 53.4, 53.4, 53.4, 53.3, 53.2, 52.1, 51.8, 51.1, 50.6 and 49.5%, and we selected $p = 5$, because now $p = 6$ gave a considerably poorer fit. Lastly, we ran CP_Simult varying p from 1 to 10, yielding fit percentages of 53.4, 53.4, 53.4, 53.3, 53.2, 52.9, 52.4, 51.8, 50.7 and 49.5%. Just for illustrative purposes, these fit percentages were plotted in Fig. 2. Here $p = 5$ or $p = 6$ seemed to give good compromises between parsimony and fit. Here, we chose to report the $p = 6$ solution with 52.9% fit. For comparative purposes we also report the $p = 10$ solution, because this actually constrained the loadings such that there was one zero and one nonzero per row[3], which was exactly the same constraint as used by the [18] method, yielding non-overlapping clusters. The fit for this solution was 49.5%.

Comparing all solutions, first of all we can note that the first four have virtually the same fit. So, it can be seen that without noticeable loss of fit, 5, 6 or 7 loadings can be constrained to be 0, yielding an overlapping clustering solution for the attributes. Clearly, however, the non-overlapping clustering solution (i.e., by the [18] method) showed an appreciable fit decrease. So for this example, it seems fruitful to relax the constraints imposed by Wilderjans and Cariou, and allow for partly overlapping clusters. Then choosing the CP_Simult solution with $p = 6$ seems a useful compromise between parsimony and fit. CP_Succ with $p =$ 7 could also be chosen, but it leads to two rows of zeros for 'acid' and 'astringency', which means that these attributes are not represented at all by this solution. Whether this is deemed problematic or not, of course is a matter for substantive experts.

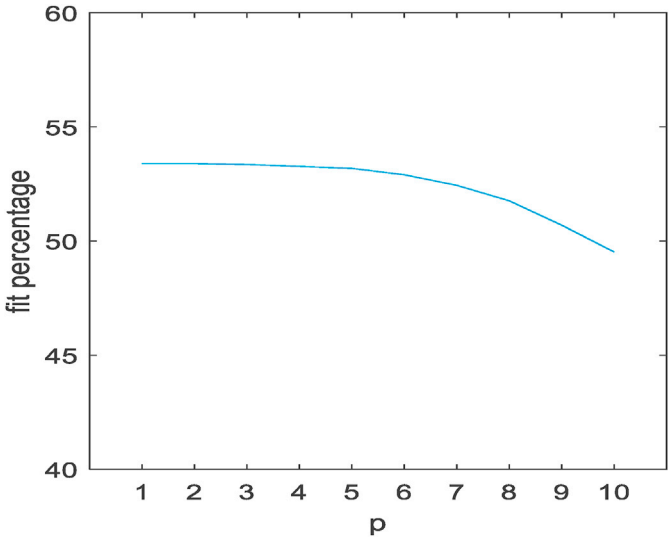

Fig. 2. Fit percentage versus number of zeros for CP_Simul applied to the Cider data.

Detailed interpretation of the component loadings is not taken up here. It can be seen that the interpretations of the components by all five methods are rather similar. The reader can refer to [18] for substantive interpretations. In the present paper, the main purpose of the analyses was to show the effect of constraining loadings to zero. Indeed, the methods can be seen to lead to different compromises between parsimony and fit, and hence different numbers of zeros.

[3] This is because there must be $p = 10$ zeros in total in a 10×2 matrix, and, as there must always be at least one nonzero in each of the ten rows, there must also be one zero in each row.

Table 2
Five different choices of construction of **W**, when $I = 15$. When $I = 30$, matrix **W** was defined as the matrix in the table below repeated once by itself.

*W*1			*W*2			*W*3			*W*4			*W*5		
0	1	1	1	0	1	1	1	1	1	1	0	1	0	0
0	1	1	1	0	1	1	1	1	1	1	0	1	0	0
1	0	1	1	0	1	1	1	1	1	1	0	1	0	0
1	0	1	1	0	1	1	1	1	1	1	0	1	0	0
1	1	0	1	0	1	1	1	1	1	1	0	1	1	0
1	1	0	1	1	0	1	1	0	1	1	0	1	1	0
1	1	1	1	1	0	1	1	0	1	0	0	1	1	0
1	1	1	1	1	0	1	1	0	1	0	0	0	1	0
1	1	1	1	1	0	1	1	0	1	0	0	0	1	1
1	1	1	1	1	0	1	1	0	1	0	1	0	1	1
1	1	1	0	1	1	1	0	0	1	0	1	0	1	1
1	1	1	0	1	1	1	0	0	1	0	1	0	0	1
1	1	1	0	1	1	1	0	0	1	0	1	0	0	1
1	1	1	0	1	1	1	0	0	1	0	1	0	0	1
1	1	1	0	1	1	1	0	0	1	0	1	0	0	1
p=6			*p*=15			*p*=15			*p*=18			*p*=24		

4. Comparison of three CP methods with *p* zero constraints on A

In the present section, we wish to compare the performance of the three approaches developed in this paper. By means of a simulation study, we generated data on the basis of the CP model with p zero loadings in **A**, and our aim was to see which of these methods best recover the underlying structure. In addition, we studied the computer time needed for applying the methods to a data set.

4.1. Generating data and design of the study

In this simulation study, data have been generated according to the CP model in (1). Specifically, for various combinations of conditions, $I \times J \times K$ data arrays $\underline{\mathbf{X}}$ have been computed from constructed matrices **A**, **B** and **C** as well as arrays $\underline{\mathbf{E}}$ with noise values e_{ijk}. In the construction of data we made some fixed choices, and varied several other choices, so as to get a fairly realistic and broad coverage of possible data structures.

The fixed choices in our design were as follows. In all cases, the number of B mode units J was fixed to 10, and the number of components R was fixed to 3. The matrices **B** and **C** were both constructed by randomly drawing all elements from the standard normal distribution, and next transforming them such that their inner product matrices equaled $\begin{pmatrix} 1 & .5 & .5 \\ .5 & 1 & .5 \\ .5 & .5 & 1 \end{pmatrix}$, that is, they had intermediate values for their mutual inner products, which seemed a realistic choice to us. Specifically, this was done by first orthogonalizing the matrices and normalizing their columns, and then multiplying them by the result of the Cholesky decomposition of the mentioned inner product matrix. The matrix **A** was constructed such that it had p values equal to exactly zero. The other elements were drawn randomly from a uniform distribution on the conjunction of the intervals $[-1, -0.5]$ and $[0.5,1]$.

The following choices were varied:

SizeA (2 levels): I was varied as $I = 15$ or $I = 30$.

SizeC (2 levels): K was varied as $K = 10$ or $K = 20$.

Wstruct (5 levels): The matrices **W** were varied in such a way that the form of overlap among components was varied, as well as the number of zeros. The five choices, labelled *W1*–*W5*, are described in Table 2. The choices were made such that we would vary complexity both in number of non-zeros and in location of non-zeros; specifically, the second and third structures were given the same number of non-zeros, but in the first they were spread evenly across components, while in the second they were very unevenly spread across components.

Noise (4 levels): The elements of the array were drawn from the standard normal distribution, and next multiplied by a constant such that the sum of squares of $\underline{\mathbf{E}}$ equaled 0, 1, 4 or 9 times that of the noise-free part of $\underline{\mathbf{X}}$ (i.e., with elements $\sum_{r=1}^{R} a_{ir}b_{jr}c_{kr}$). Specifically, we first computed the noise-free data, and their sum of squares (SS_f), then drew the random noise values, and computed their sum of squares (SS_n), then multiplied the noise values by $\sqrt{(SS_f/SS_n)}$ and by 0, 1, 2 or 3, to get data with, respectively 0% noise, and (approximately) 50%, 80% and 90% noise[4]. It should be realized that the latter two conditions are extremely noisy. Indeed, the 80% noise condition implies that there is four times more noise than signal, while the 90% condition even has 9 times more noise than signal. We incorporated these conditions to explore the range of situations under which still good results could be obtained, and also to see whether under such conditions differences between the results of the methods could be discerned.

[4] The idea is that the total sum of squares of $\underline{\mathbf{X}}$ is approximately equal to the sum of sums of squares of $\underline{\mathbf{X}}_f$ (noise-free part of the data) and of $\underline{\mathbf{E}}$, because the noise values were drawn randomly. The multiplication of the randomly drawn noise values by $\sqrt{(SS_f/SS_n)}$ makes their sum of squares equal to that of $\underline{\mathbf{X}}_f$. Hence multiplying the noise part by 1, makes the ratio 1:1, hence 50% noise, multiplying the noise by 2 makes the ratio 4:1, so 80% noise, and multiplying by 3 makes the ratio 9:1, hence 90% noise.

For each of the $2 \times 2 \times 5 \times 4$ combinations of levels of the factors *SizeA, SizeC, Wstruct* and *Noise*, 100 data sets were generated. To each data set, all three methods were applied, taking p equal to p^{true}, that is, the number of zeros in the matrix used in the construction of the data at hand. In addition, solutions were inspected for p values up to 3 values lower or higher than p^{true}. For each analysis with CP, 11 runs were used, one using the rational start based on the SVD's of appropriately arranged matricized versions of $\underline{\mathbf{X}}$, and ten using random starts. The CP_Simult analyses employed 51 starts, the first of which started with the solution from CP_Succ_NoZRows, while the others were started randomly. In all algorithms, the convergence criterion was that subsequent loss values differed by less than 10^{-6}% of the previous loss values, and the maximum number of iterations was set at 500.

4.2. Evaluation criteria

The main goal of the simulation study was to compare the methods as to how well they recover the underlying structure. For this purpose, two measures have been inspected: The average congruence (as measured by [21] congruence coefficient) of the columns of the estimated A with those of the underlying matrix A (both including the zero elements), and the analogously defined recovery of the matrix [5]. Because the methods do not uniquely determine the order or the sign of the loading vectors, before assessing these values, the columns of the estimated matrices **A** have been permuted and reflected in all possible permissible ways and the one yielding the best recovery of **A** was kept, and used for computing both the recovery of **A** and of **W**.

The analyses were run in Matlab on an ordinary desktop computer under Windows. Computation time was assessed for the 11 CP runs, and for all runs involved in the analyses for CP_Succ, CP_Succ_NoZRows, and CP_Simult using $p = p^{true}-3,\ldots, p^{true}$. Note that for the two CP_Succ methods, the time needed for obtaining the CP solution was also incorporated. Likewise, for CP_Simult the computation of the solution of CP_Succ_NoZRows which was used as the starting configuration in one run of CP_Simult was taken into account.

The CP analyses sometimes yielded degenerate solutions. In such cases, it makes no sense to use the CP solution as a start. Degeneracies as indicated by a minimal triple cosine below -0.90 were therefore monitored and dealt with separately in the report of the outcomes.

The numbers of local optima were recorded for the CP analyses and the CP_Simult analyses. These actually were *minimal* numbers of local optima, because they referred to the number of solutions in which the fit percentage was at least .01% lower than that of the best solution from all runs, while in principle, even the best solution may be a local optimum.

Finally, it has been seen to what extent, within the range of p values used, the scree test described in Section 2.4 was helpful in recovering the actually underlying true number of zero loadings.

4.3. Results

First an exploratory check on the results was made. For analyses using the correct number of zero loadings, all individual results for the recovery of A-mode loadings and the binary weights, as well as the fit percentages and numbers of local optima were plotted on one screen for each of the 80 conditions for quick visual inspection. This indicated that fit percentages were as expected, numbers of local optima for CP_Simult varied much (also within conditions), and recovery results often showed a big majority of fairly equal results and a small set of clearly poorer results. The poorest results tended to be associated with degenerate CP solutions (i.e. with very low triple cosine values). Upon inspecting these in detail, we saw that in some cases results based on CP solutions with very low triple cosines led to a zero column in **W** and hence in **A**. This in turn implied that the CP estimates for **B** and **C** were undefined, and hence no constrained CP solution was obtained. This happened in 11 (out of 8000) cases. The explanation is that, in case of degeneracy, loadings on two components tend to become high in absolute sense, and as a consequence the loadings on the third may often all be lower than the other ones. If such a CP solution is being used to indicate the smallest values which are to be set to zero, this can easily result in a complete column of zeros. However, in actual practice, one would recognize a degenerate solution, and next *replace it* by a non-degenerate solution, for instance by applying remedies suggested in [13] or the Error Preserving Correction approach by [22]. As it is unrealistic to use degenerate solutions in the two successive methods, we decided to ignore results for all 141 (out of 8000) cases where CP led to a triple cosine below -0.90. This effectively eliminated the solutions with zero columns, but also other poor CP solutions. It should be realized, however, that this does not preclude all *near* degenerate solutions leading to poor indications of locations of zero loadings. As it is difficult to define near degeneracy, we decided to stay on the safe side, and kept -0.90 as our inclusion threshold. All results given below hence relate to cases where CP gave triple cosines higher than -0.90. For each of the 80 conditions we had at least 84 remaining cases, while in all conditions with 0% and 50% noise all 100 cases remained.

Our main interest was in recovery of the A-mode loadings, and recovery of the location of the zero loadings (both in cases where the number of zero loadings is correct, that is $p = p^{true}$). The recovery of A-mode loadings for all conditions is displayed in Fig. 3, while that for **W** (i.e., recovery of the structure) is displayed in Fig. 4. To give an idea of the sampling uncertainty of these results, we computed standard deviations per condition, and found these (across all conditions) to be at most 0.13, hence all averages pertained to standard errors of at most 0.014, and often much less. In the noise-free condition (0% noise) and the 50% noise conditions, the results were strikingly good for all methods. Specifically, in the noise-free condition, average recovery of both the loadings and the structure was perfect in all conditions by all methods. This can be seen as an unrealistic, but useful test condition because we can thus verify whether, if a perfect solution *can* be attained, the methods actually do attain it. It has been verified that actually each of the 2000 individual cases led to perfect recovery (i.e., value over 0.9999) of both loadings and structure by all methods. The 50% noise condition seems to be the most practically realistic condition in our study, and also in this condition, recovery results were extremely good: Mean recoveries of A-mode loadings were never lower than 0.977, and those for the structure never below 0.99. Individual cases led only once to a recovery of the A-mode loadings lower than 0.95, and only twice to a recovery of the structure lower than 0.95, out of 2000 cases. Clearly, all methods gave extremely good recovery under these conditions, and differences between methods were ignorable.

In order to see if differences did show up in conditions with more noise, we incorporated the conditions with 80% and 90% noise in our design. Uncovering structure in such noisy data may be too much to ask for, but in case of strong theories of an underlying CP structure, one might still want to use CP to uncover such a structure even with very noisy data. From Figs. 3 and 4 it can be seen that, for 80% noise, in almost all conditions the mean recovery of all methods was higher than 0.85 (which is often considered as a threshold for considering the vectors compared by a congruence coefficient as fairly similar, see [23]. With the 90% noise condition, we seemed to have reached the breakdown level: Now in many conditions average recovery was poor (say, under 0.80), and given that results were poor and hence not useful, it did not really matter whether one method gives less poor results than the other. The main exceptions to this occurred for conditions with the largest values for I and K, that is $I = 30$ and $K = 20$, and the conditions with the simplest structure (*W5*), combined with $I = 30$ and/or $K = 20$, as can be discerned

[5] Congruence between binary vectors may seem a strange measure. However, this value is easily interpreted as the number of overlapping nonzeros, divided by the square roots of the numbers of nonzeros in the one and the other column, respectively. Clearly, the minimum is 0 (in case of no overlapping nonzeros), and the maximum is 1 (in case of identical binary vectors).

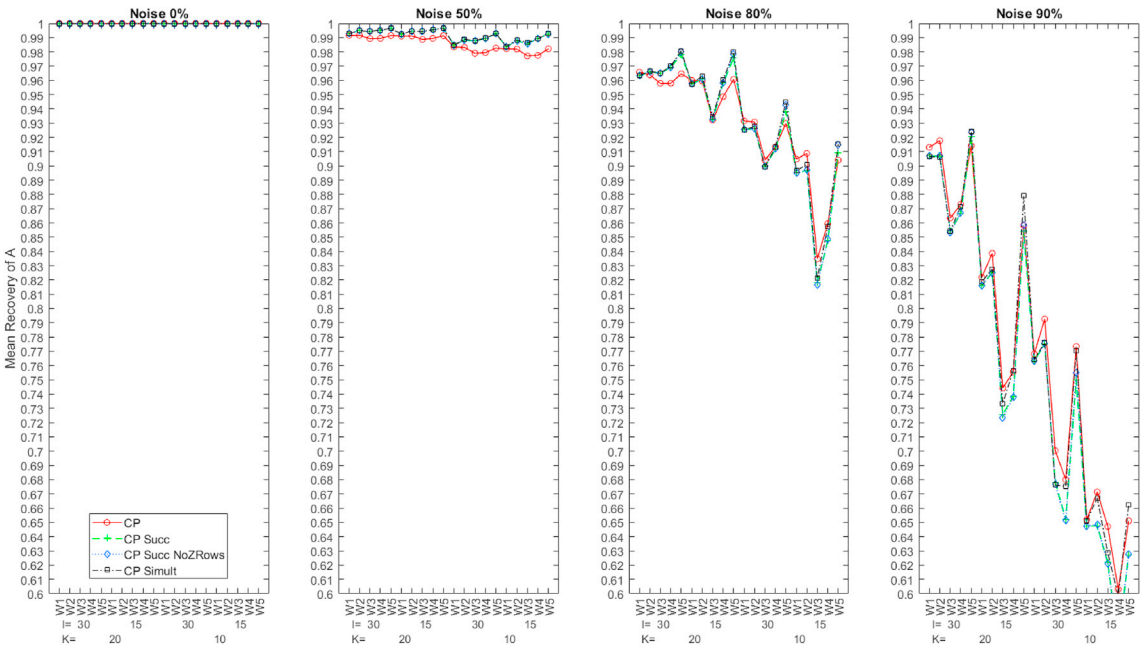

Fig. 3. Mean recovery of A mode loadings by CP, CP_Succ, CP_Succ_NoZRows and CP_Simult under all 80 conditions.

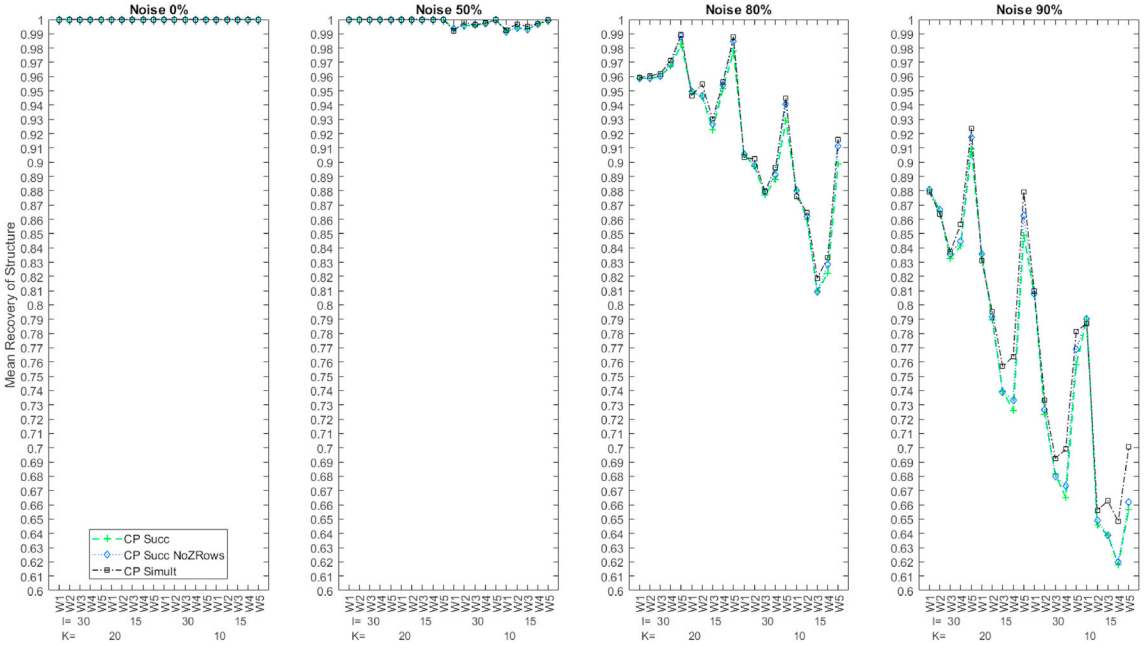

Fig. 4. Mean Recovery of structure by CP, CP_Succ, CP_Succ_NoZRows and CP_Simult under all 80 Conditions.

easily from Figs. 3 and 4. This can be explained by the fact that in such conditions the structure is better consolidated (i.e. based on more data), and of a less confounded nature (clearest structure in **W**). This explanation entails that recovery improves both as noise decreases and as size of I and K increases. Because recovery cannot increase endlessly (as it is maximized at 1), the increase in recovery due to one of these factors must diminish when it increases due to another factor, hence the clear *interaction* between these factors. The effect of structure is less easy to understand, although it seems clear that *W5* had most zeros and hence gave the simplest structure, which led to the least confounding of components. As to why, in the 90% noise conditions, *W1* and *W2* led to relatively good recovery too, we can only guess.

We now describe the results for the 80% noise conditions in a bit more detail. From Fig. 4 it is clear that the average recovery of the structure by the three constrained approaches was virtually equal (lines almost overlap, and mean differences reached up to 0.01 at most, which is negligible for practical purposes). However, the differences in mean recovery of loadings was for some conditions more substantial (although still not clearly over 0.02). In all but three conditions, the three constrained approaches gave better or equally good recovery as unconstrained CP did, while in the conditions with $I = 15$, $K = 10$, and where the population **W** had structures *W1*, *W2* or *W3*, unconstrained CP gave somewhat better recovery; the three constrained methods gave very similar results. In the 90% noise condition, focusing on cases with sufficiently good recovery, a similar conclusion can be drawn as for the 80% condition, except that now the difference between CP and the

constrained methods was a bit more in favor of CP. But it should be noted that all such differences were really small, and, as will be seen below, should actually be taken with a big grain of salt.

Since differences in means were small, we decided to inspect numbers of individual cases where substantial mean differences (i.e. at least 0.03) in the loadings occurred. We did so for the comparison of CP_Simult and CP in Table 3, and CP_Simult and CP_Succ_NoZRows in Table 4, while we focused on the 80% noise conditions; results for the other conditions were given as well for comparative purposes. It can be seen first of all that substantial differences virtually never occurred for noise condition 0% (as we already knew) and noise condition 50%, as we already expected. For the condition with 80% noise, we now see that substantial differences did occur, but never in a majority of cases in the same direction. Rather, the broad picture in both comparisons is that only slight tendencies can be discerned, which however by no means seem systematic. First of all, we considered the earlier mentioned slightly better performance of CP than the constrained methods, in the $I = 15$, $K = 10$, conditions with structures *W1*, *W2* and *W3*. From Table 3 we now see that actually in 5, 18 and 26 cases indeed CP had more than 0.03 better recovery than CP_Simult. However, in the great majority of cases differences were smaller than 0.03, and in 2, 6 and 10 cases, respectively, CP actually performed poorer than CP_Simult. Thus, the earlier mentioned results were far from consistent, and at most reflected a slight tendency. Upon further scrutinizing Table 3, for 80% noise, we see that the most systematic pattern (i.e., with most advantages for the one method, while none or few for the other) occurred for $K = 20$, $I = 15$, *W5,* where CP_Simult 10 times gave substantially better results, while 90 times the difference was less than 0.03, and never CP gave substantially better results. This, however, still indicated that in individual cases, in the great majority of cases differences were unsubstantial, but sometimes CP_Simult gave really better result than CP. In other cases, the differences were even less outspoken, or sometimes quite contradictory, e.g., for $K = 10$, $I = 15$, *W4,* where the frequencies were 14, 59, and 24 conveying the message: we can quite often expect a substantial difference in the one direction, or the other, and in roughly 60% of the time none at all. All in all, these results indicate that, in the 80% noise conditions, differences in recovery do show up, but can never be seen to be really systematically in favor of any of the methods. In the 90% noise case, the situation was very similar: No clear patterns of differences in either direction can be discerned. On a note of interest, we also inspected the recovery of the B-mode and C-mode loadings. These gave very much the same picture, except that differences were even less than for recovery of **A.** The results can be inspected in the supplementary material of this paper, and on our osf site https://osf.io/ymxgc/.

Next, we studied computation times. Obviously, computation times differed quite a bit between the methods: The successive approaches used only one run of their algorithm, while the simultaneous approach needed many (in our study 51). To get an idea about differences in computation times, in Table 5, we report these for the four methods. For CP we used 11 runs, for the two successive fitting approaches we used both the 11 CP runs, and 7 (single run) analyses for different values of p, while for the simultaneous approach, we used 11 CP runs, 7 (single run) analyses of CP_Succ_NoZRows, and 7 analyses involving 51 runs of the Simultaneous fitting algorithm. In the report of the average computation times, we distinguish only between noise conditions and size of K, as the biggest differences were seen across levels of these factors.

It can be seen that the extra runs for the successive approaches were very cheap compared to the CP analyses. The simultaneous fitting procedure, however, in this set up, was roughly a factor 100 times as time consuming as unconstrained CP and the successive approaches.

As far as local optima are concerned, we first report average frequencies for local optima among the 11 runs of the CP analyses (Table 6). We see that, for the low noise conditions, local optima occurred rarely, while for the highest noise conditions they occurred considerably more frequently. In all cases, the choice of as many as 11 random starts seems on the safe side, with little chance to miss the global optimum.

The CP_Simult analyses led quite frequently to local optima, although this did depend strongly on the noise level, and, interestingly, on the value of p. Table 7 gives a full overview of average frequencies of local optima in all conditions, and for three values of p: $p = p^{true}-1$, $p = p^{true}$, and $p = p^{true}+1$. Clearly, in the low noise conditions, local optima occurred far more frequently when p exceeded p^{true}, in all 20 conditions. In the 80% and 90% noise cases, local optima occurred very frequently, irrespective of the value of p.

Table 3
Frequencies of CP_Simult giving substantially better recovery (more than 0.03) of the loadings, more or less equal recovery (between ± 0.03), or substantially worse recovery (more than 0.03) than CP, in the three respective columns in each block. (The frequencies do not always add up to 100, because of exclusion of degenerate solutions).

			Noise 0%			Noise 50%			Noise 80%			Noise 90%		
K = 10	*I* = 15	*W1*	0	100	0	0	100	0	2	90	5	2	80	2
		W2	0	100	0	0	100	0	6	75	18	15	34	38
		W3	0	100	0	0	99	1	10	57	26	4	59	24
		W4	0	100	0	0	100	0	14	59	24	16	35	33
		W5	0	100	0	0	100	0	23	71	6	26	29	29
	I = 30	*W1*	0	100	0	0	100	0	0	100	0	0	98	1
		W2	0	100	0	0	100	0	0	99	1	5	66	24
		W3	0	100	0	0	100	0	2	91	7	6	55	35
		W4	0	100	0	0	100	0	1	97	2	10	49	31
		W5	0	100	0	0	100	0	6	94	0	12	64	20
K = 20	*I* = 15	*W1*	0	100	0	0	100	0	0	100	0	2	88	6
		W2	0	100	0	0	100	0	0	100	0	4	70	24
		W3	0	100	0	0	100	0	6	85	8	14	50	29
		W4	0	100	0	0	100	0	6	93	1	21	41	30
		W5	0	100	0	0	100	0	10	90	0	27	55	14
	I = 30	*W1*	0	100	0	0	100	0	0	100	0	0	99	1
		W2	0	100	0	0	100	0	0	100	0	0	97	3
		W3	0	100	0	0	100	0	1	99	0	2	84	13
		W4	0	100	0	0	100	0	1	99	0	5	80	13
		W5	0	100	0	0	100	0	1	99	0	10	88	2

Table 4
Frequencies of CP_Simult giving substantially better recovery (more than 0.03) of the loadings, more or less equal recovery (between ± 0.03), or substantially worse recovery (more than 0.03) than CP_Succ_NoZRows, in the three respective columns in each block. (The frequencies do not always add up to 100, because of exclusion of degenerate solutions).

			Noise 0%			Noise 50%			Noise 80%			Noise 90%		
K = 10	*I* = 15	*W1*	0	100	0	0	100	0	1	96	0	5	77	2
		W2	0	100	0	0	100	0	6	87	6	12	65	10
		W3	0	100	0	0	100	0	12	68	13	11	64	12
		W4	0	100	0	0	100	0	14	69	14	18	58	8
		W5	0	100	0	0	100	0	13	82	5	27	41	16
	I = 30	*W1*	0	100	0	0	100	0	0	100	0	0	99	0
		W2	0	100	0	0	100	0	0	100	0	6	82	7
		W3	0	100	0	0	100	0	2	93	5	13	71	12
		W4	0	100	0	0	100	0	0	100	0	15	65	10
		W5	0	100	0	0	100	0	1	99	0	16	71	9
K = 20	*I* = 15	*W1*	0	100	0	0	100	0	0	100	0	3	92	1
		W2	0	100	0	0	100	0	1	99	0	8	83	7
		W3	0	100	0	0	100	0	10	82	7	18	61	14
		W4	0	100	0	0	100	0	4	95	1	18	57	17
		W5	0	100	0	0	100	0	1	99	0	19	69	8
	I = 30	*W1*	0	100	0	0	100	0	0	100	0	0	99	1
		W2	0	100	0	0	100	0	0	100	0	0	100	0
		W3	0	100	0	0	100	0	0	100	0	3	91	5
		W4	0	100	0	0	100	0	0	100	0	4	87	7
		W5	0	100	0	0	100	0	0	100	0	2	97	1

Table 5
Mean computation times.

	CP 11 runs		CP_Succ 11 runs CP+7 analyses		CP_Succ_NoZRows 11 runs CP+7 analyses		CP_Simult 11 runs CP+7 succ. analyses + 7x51 runs	
Noise	*K* = 10	*K* = 20	*K* = 10	*K* = 20	*K* = 10	*K* = 20	*K* = 10	*K* = 20
0 %	0.07	0.09	0.09	0.12	0.09	0.12	21.89	24.44
50 %	0.04	0.05	0.07	0.09	0.07	0.09	12.32	12.69
80 %	0.09	0.08	0.13	0.12	0.13	0.12	15.07	14.46
90 %	0.21	0.18	0.27	0.23	0.27	0.23	18.81	18.02

Table 6
Average frequencies of local optima in 11 CP runs.

	CP 11 runs	
Noise	*K* = 10	*K* = 20
0%	0.02	0.02
50%	0.04	0.01
80%	1.92	0.94
90%	5.89	4.34

Finally, for all cases we applied the scree test to the results from the seven analyses with values of p ranging from $p = p^{true}-3$, $p = p^{true}$, and $p = p^{true}+3$. We inspected them in detail for CP_Succ_NoZRows and CP_Simult, because these methods use the same constrained model. For CP_Succ, similar results were found, but not reported here. The frequencies of cases in which the scree test indicated p_{true} as the most desirable value for p are reported in Table 8, where the noise free condition is ignored, because in this case, subsequent fit values for $p = p^{true}-3$, …, $p = p^{true}$ should all be 100%, but the scree test involved division by fit differences, which in this case were zero or very close to zero. In practice, however, upon visual inspection, the noise free cases with perfect fit outcomes up to $p = p^{true}$ would automatically lead to p^{true} as the desired solution.

The results (in Table 8) show that for the 50% noise conditions, and $K = 20$, the scree test very often indicated the underlying number of zeros as the preferred number. For $K = 10$ this was the case in far fewer instances. With the high noise percentages, the scree test did not have a clear preference for the underlying number of zeros.

5. Discussion

This paper has described three constrained versions of CP, in which a fixed number (p) of loadings of one mode is set equal to zero. The methods were applied to an empirical data set, to illustrate their effect, and how one can work with them. Next, in a simulation study, their performance was assessed. It was observed that up to large amounts of noise, the methods all performed well in terms of recovering the underlying structure given the number of zero loadings, and differences were small and not really systematic. The simultaneous method is much more time consuming than the successive approaches, and the fit improvement one can obtain with it does not systematically lead to improved recovery. For these reasons, it seems that if zero loadings are desired, the successive approach is the preferred approach. Only if optimal data fit is essential, the simultaneous approach seems to be

Table 7
Frequencies of local optima for CP_Simult using 51 runs, reported in each block of three columns for analyses with $p = p^{true}-1$, $p = p^{true}$, and $p = p^{true}+1$.

			Noise 0%			Noise 50%			Noise 80%			Noise 90%		
K = 10	I = 15	W1	2.0	3.0	40.8	17.3	17.9	33.6	39.4	42.8	44.8	42.8	45.1	45.9
		W2	5.1	6.8	39.3	25.5	19.3	31.4	47.1	47.5	48.4	49.3	49.4	49.6
		W3	7.2	8.3	41.1	29.9	26.5	39.5	49.0	48.9	49.3	49.0	49.3	49.5
		W4	6.0	7.6	40.0	27.2	22.0	38.1	48.5	48.7	48.6	49.7	49.7	49.8
		W5	3.6	4.0	34.2	22.7	11.3	32.3	46.8	47.1	47.3	49.9	49.8	49.8
	I = 30	W1	1.9	2.2	41.4	23.4	21.4	34.0	43.0	44.5	44.6	44.3	45.3	46.0
		W2	4.6	5.2	38.5	25.5	18.6	31.3	48.7	48.6	49.0	49.7	49.8	49.8
		W3	6.5	7.7	41.7	29.3	25.1	37.5	49.3	49.5	49.6	49.8	49.9	49.9
		W4	4.8	5.8	39.0	26.8	21.2	34.6	49.2	49.1	49.4	49.9	49.9	49.9
		W5	2.7	3.1	34.3	23.0	8.3	27.2	48.3	48.5	48.5	49.9	50.0	49.9
K = 20	I = 15	W1	2.1	3.3	41.4	12.0	3.8	33.0	32.1	35.9	40.9	44.2	44.8	46.1
		W2	5.7	6.5	39.9	19.3	4.6	31.8	39.9	41.7	43.5	49.3	49.5	49.5
		W3	7.3	8.3	41.3	21.8	12.4	38.9	45.5	46.1	47.8	49.3	49.5	49.5
		W4	6.3	7.5	40.1	21.2	9.9	38.9	44.6	44.8	46.2	49.6	49.7	49.8
		W5	3.5	4.2	32.7	17.8	5.5	32.0	37.3	35.0	42.1	48.9	48.9	49.1
	I = 30	W1	2.1	2.2	40.7	14.4	3.4	33.1	37.8	41.1	44.4	44.4	46.1	47.2
		W2	5.2	5.4	39.4	18.0	3.5	28.2	44.7	44.9	45.5	49.6	49.5	49.5
		W3	7.0	8.4	40.9	18.5	6.2	36.3	46.1	45.4	46.5	49.6	49.8	49.8
		W4	5.1	6.0	40.9	19.4	5.5	34.3	44.0	44.4	45.9	49.8	49.7	49.9
		W5	2.4	3.0	34.3	17.1	3.4	29.1	38.8	36.1	40.0	49.3	49.4	49.5

Table 8
Frequencies of times the scree test indicated p^{true}, as the most desired value for p.

			CP_Succ_NoZRows			CP_Simult		
			50%	80%	90%	50%	80%	90%
K = 10	I = 15	W1	36	13	19	36	13	11
		W2	64	21	20	63	20	19
		W3	56	17	24	57	26	17
		W4	74	20	16	74	24	16
		W5	90	19	15	92	20	17
	I = 30	W1	40	16	26	40	18	23
		W2	62	25	17	63	20	18
		W3	51	19	17	53	15	20
		W4	60	19	15	63	25	24
		W5	87	21	20	85	26	18
K = 20	I = 15	W1	75	24	18	75	23	16
		W2	95	21	19	95	19	19
		W3	93	24	16	93	29	16
		W4	97	25	26	98	22	19
		W5	98	50	24	99	53	19
	I = 30	W1	84	31	25	84	33	20
		W2	94	23	18	94	28	17
		W3	95	16	21	96	24	17
		W4	96	22	18	97	21	9
		W5	99	45	17	100	41	20

preferred. Actually, in such cases it is important to realize that we cannot be sure that the global optimum has been reached. It is well known that the problem of constraining a fixed number of elements to zero without knowing which ones, is difficult to solve. For tackling this problem, also known as an L_0 norm constrained optimization problem, various other strategies have been developed (e.g., see [24,25]. Future research could be devoted to applying such approaches in the present context, and comparing them to our approach.

In practice, one often does not know what number of zeros to choose. If the aim is to discover an underlying true model, our results show that this is reasonably well possible by means of the scree test in the 50% noise case; with 80% or 90% noise, however, this becomes really difficult, as we can derive from Table 8. However, if the main aim is to find a good compromise between parsimony and fit, the choice of p need not be essential. And, as one can see in the analysis of the empirical data, the interpretations of the components do not really depend much on the zero loading constraints employed.

In the present paper we have reported the development and comparison of methods that simplify interpretation by employing zero loadings. As has been mentioned, the methods in this way actually can be seen as methods for overlapping clustering, or in special cases nonoverlapping clustering. Thus, there is a relation with the clusterwise CP approach proposed by [5]. Their approach offers nonoverlapping clusters, with for each cluster one or more CP components. In the special case of their method using one component per cluster, and of CP_Simult using $(I-1)R$ zero loadings (implying non-overlapping clusters), these methods are equivalent to each other, and actually also to the methods by [4,18]. Which of these methods to use in practice, obviously depends on the goal of one's analysis.

The goal of the methods in the present paper resembles that of methods for finding sparse component loadings (e.g., see [26]). Sparseness constrained CP methods have been proposed, for instance, by [27] for obtaining approximate sparseness, and by [28] for Lasso based sparseness. In the present paper, we restrict ourselves to methods aiming for exact sparseness, *and* we wish to get optimal fit of the other parameters, so we wish to avoid the shrinkage entailed by the Lasso approach. This shrinkage problem has been noted by Rasmussen & Bro (2012, p.23), and they suggested to fix that by first using Lasso to find the locations of the zero loadings, and then optimally estimating the remaining elements. This of course closely resembles our Successive fitting approach, except that we suggested to use the CP solution itself to indicate zero loadings, rather than the Lasso. Whether the latter would work better than our approach is an empirical question, which could be interesting for further research.

Author statement

Kiers: Conceptualization, Methodology, Software, Data analysis, Writing - Original Draft, Writing - Review & Editing. Giordani: Conceptualization, Methodology, Writing - Review & Editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

We thank the three anonymous reviewers for their valuable input to our paper.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.chemolab.2020.104145.

Supplementary material

All Matlab scripts used in this study are available on https://osf.io/ymxgc/.

References

[1] J.D. Carroll, J.J. Chang, Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition, Psychometrika 35 (1970) 283–319.
[2] R.A. Harshman, Foundations of the Parafac procedure: models and conditions for an 'explanatory' multimodal factor analysis, UCLA Work. Pap. Phonetics 16 (1970) 1–84.
[3] W.P. Krijnen, The Analysis of Three-Way Arrays by Constrained PARAFAC Methods, DSWO Press, Leiden University, 1993.
[4] W.P. Krijnen, H.A.L. Kiers, Clustered variables in PARAFAC, in: Advances in Longitudinal and Multivariate Analysis in the Behavioral Sciences: Proceedings of the SMABS 1992 Conference, JHL Oud, RAW Van Blokland-Vogelesang), vol. 166, 1993. Nijmegen: ITS.
[5] T.F. Wilderjans, E. Ceulemans, Clusterwise Parafac to identify heterogeneity in three-way data, Chemometr. Intell. Lab. Syst. 129 (2013) 87–97.
[6] V. Cariou, T.F. Wilderjans, Consumer segmentation in multi-attribute product evaluation by means of non-negatively constrained CLV3W, Food Qual. Prefer. 67 (2018) 18–26.
[7] K. De Roover, E. Ceulemans, P. Giordani, Overlapping clusterwise simultaneous component analysis, Chemometr. Intell. Lab. Syst. 156 (2016) 249–259.
[8] J.M.F. ten Berge, H.A.L. Kiers, J. De Leeuw, Explicit Candecomp/Parafac solutions for a contrived $2 \times 2 \times 2$ array of rank three, Psychometrika 53 (1988) 579–584.
[9] J.B. Kruskal, R.A. Harshman, M.E. Lundy, How 3-MFA data can cause degenerate PARAFAC solutions, among other relationships, in: R. Coppi, S. Bolasco (Eds.), Multiway Data Analysis, Elsevier, Amsterdam, 1989, pp. 115–122.
[10] B.C. Mitchell, D.S. Burdick, Slowly converging Parafac sequences: swamps and two-factor degeneracies, J. Chemometr. 8 (1994) 155–168.
[11] A. Stegeman, Degeneracy in Candecomp/Parafac explained for $p \times p \times 2$ arrays of rank $p + 1$ or higher, Psychometrika 71 (2006) 483–501.
[12] R. Rocci, P. Giordani, A weak degeneracy revealing decomposition for the CANDECOMP/PARAFAC model, J. Chemometr. 24 (2010) 57–66.
[13] P. Giordani, R. Rocci, Remedies for degeneracy in Candecomp/Parafac, in: Quantitative Psychology Research, Springer, Cham, 2016, pp. 213–227.
[14] H.A.L. Kiers, A.K. Smilde, Constrained three-mode factor analysis as a tool for parameter estimation with second-order instrumental data, J. Chemometr. 12 (1998) 125–147.
[15] W.J. Heiser, Convergent computation by iterative majorization: theory and applications in multidimensional data analysis, in: W.J. Krzanowski (Ed.), Recent Advances in Descriptive Multivariate Analysis, Oxford University Press, Oxford, 1995, pp. 157–189.
[16] H.A.L. Kiers, Majorization as a tool for optimizing a class of matrix functions, Psychometrika 55 (1990) 417–428.
[17] H.A.L. Kiers, Setting up alternating least squares and iterative majorization algorithms for solving various matrix optimization problems, Comput. Stat. Data Anal. 41 (2002) 157–170.
[18] T.F. Wilderjans, V. Cariou, CLV3W: a clustering around latent variables approach to detect panel disagreement in three-way conventional sensory profiling data, Food Qual. Prefer. 47 (2016) 45–53.
[19] R.B. Cattell, The scree test for the number of factors, Multivariate Behav. Res. 1 (2) (1966) 245–276.
[20] T.F. Wilderjans, E. Ceulemans, K. Meers, CHull: a generic convex-hull-based model selection method, Behav. Res. Methods 45 (2013) 1–15.
[21] L.R. Tucker, A method for synthesis of factor analysis studies. Personnel Research Section Report No.984, Dept. of the Army, Washington D.C., 1951.
[22] A.H. Phan, P. Tichavský, A. Cichocki, Error preserving correction: a method for CP decomposition at a target error bound, IEEE Trans. Signal Process. 67 (5) (2018) 1175–1190.
[23] U. Lorenzo-Seva, J.M.F. Ten Berge, Tucker's congruence coefficient as a meaningful index of factor similarity, Methodology 2 (2006) 57–64.
[24] R. Peharz, F. Pernkopf, Sparse nonnegative matrix factorization with ℓ0-constraints, Neurocomputing 80 (2012) 38–46.
[25] G.J. van den Burg, P.J. Groenen, A. Alfons, Sparsestep: Approximating the Counting Norm for Sparse Regularization. arXiv Preprint arXiv:1701.06967, 2017.
[26] M.A. Rasmussen, R. Bro, A tutorial on the Lasso approach to sparse modeling, Chemometr. Intell. Lab. Syst. 119 (2012) 21–31.
[27] E.E. Papalexakis, C. Faloutsos, N.D. Sidiropoulos, ParCube: sparse parallelizable CANDECOMP-PARAFAC tensor decomposition, ACM Trans. Knowl. Discov. Data 10 (1) (2015) 1–25.
[28] E.E. Papalexakis, N.D. Sidiropoulos, Co-clustering as Multilinear Decomposition with Sparse Latent factors, in: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Prague, 2011, 2011, pp. 2064–2067, https://doi.org/10.1109/ICASSP.2011.5946731.