Research Article

JOURNAL OF CHEMOMETRICS

Received: 28 July 2008, Revised: 3 February 2009, Accepted: 14 February 2009, Published online in Wiley Interscience: 16 March 2009

(www.interscience.wiley.com) DOI: 10.1002/cem.1232

# Using the simultaneous generalized Schur decomposition as a Candecomp/Parafac algorithm for ill-conditioned data

**Alwin Stegeman**[a]*

**The Candecomp/Parafac (CP) method decomposes a three-way array into a prespecified number *R* of outer product arrays, by minimizing the sum-of-squares of the residual array. The practical use of CP is sometimes complicated by the occurrence of so-called 'degenerate' sequences of solutions, in which several outer product arrays become highly correlated in all three modes and some elements of the outer product arrays become very large in magnitude. It is known that for $I \times J \times 2$ arrays, fitting a simultaneous generalized Schur decomposition (SGSD) avoids the problems of 'degeneracy' due to the non-existence of an optimal CP solution. In this paper, we consider the application of the SGSD method also for other array formats, when the array has a best fitting CP decomposition with ill-conditioned component matrices, in particular such that it resembles the pattern of a 'degeneracy'. For these cases, we compare the performance of two SGSD algorithms and the alternating least squares (ALS) CP algorithm in a series of numerical experiments. Copyright © 2009 John Wiley & Sons, Ltd.**

**Keywords:** Candecomp; Parafac; Schur decomposition; degenerate solutions; diverging components

## 1. INTRODUCTION

Carroll and Chang [1] and Harshman [2] independently proposed the same method for component analysis of three-way arrays, and named it Candecomp and Parafac, respectively. In the Candecomp/Parafac (CP) model, an $I \times J \times K$ array $\underline{\mathbf{X}}$ is decomposed into a prespecified number of $R$ outer product arrays and a residual term, i.e.

$$\underline{\mathbf{X}} = \sum_{r=1}^{R} \omega_r (\mathbf{a}^{(r)} \circ \mathbf{b}^{(r)} \circ \mathbf{c}^{(r)}) + \underline{\mathbf{E}} \tag{1}$$

where $\circ$ denotes the outer product, $\omega_r$ are the weights of the outer product arrays and $\mathbf{a}^{(r)}$, $\mathbf{b}^{(r)}$ and $\mathbf{c}^{(r)}$ are the $I$-, $J$- and $K$-vectors forming the outer product arrays, with $\|\mathbf{a}^{(r)}\| = \|\mathbf{b}^{(r)}\| = \|\mathbf{c}^{(r)}\| = 1$ for $r = 1, \ldots, R$, where $\|\cdot\|$ denotes the Frobenius norm. The $(i, j, k)$-element of the outer product array $\mathbf{a}^{(r)} \circ \mathbf{b}^{(r)} \circ \mathbf{c}^{(r)}$ equals $a_i^{(r)} b_j^{(r)} c_k^{(r)}$. We assume the weights $\omega_r$ to be nonnegative. For fixed $R$, the CP decomposition (1) is found by minimizing the sum-of-squares of $\underline{\mathbf{E}}$. Several iterative algorithms exist for this purpose, see e.g. Tomasi and Bro [3].

We consider the real-valued CP model, i.e. we assume the array $\underline{\mathbf{X}}$ and the decomposition to be real-valued. The real-valued CP model is used in a majority of applications in psychology and chemistry; see Kroonenberg [4] and Smilde, Bro and Geladi [5]. Complex-valued applications of CP occur in e.g. signal processing and telecommunications research; see Sidiropoulos, Giannakis and Bro [6] and Sidiropoulos, Bro and Giannakis [7].

The three-way rank of $\underline{\mathbf{X}}$ is usually defined as the following generalization of matrix rank: the smallest number of rank-1 arrays whose sum equals $\underline{\mathbf{X}}$. A three-way array has rank 1 if it is the outer product of three vectors. Hence, in the CP decomposition (1) each of the $R$ outer product arrays has rank 1. The three-way rank of $\underline{\mathbf{X}}$ is equal to the smallest number of components for which a CP decomposition exists with perfect fit, i.e., with an all-zero residual term $\underline{\mathbf{E}}$. Moreover, it follows that solving the CP model boils down to finding a best rank-$R$ approximation of $\underline{\mathbf{X}}$ in terms of smallest sum-of-squares distance. Since we consider the real-valued CP model, the rank of any array is assumed to be the rank over the real field.

A matrix notation of the CP model (1) is as follows. Let $\mathbf{X}_k$ ($I \times J$) and $\mathbf{E}_k$ ($I \times J$) denote the $k$th frontal slices of $\underline{\mathbf{X}}$ and $\underline{\mathbf{E}}$, respectively. Then Equation (1) can be written as

$$\mathbf{X}_k = \mathbf{A}\mathbf{C}_k\boldsymbol{\Omega}\mathbf{B}^{\mathrm{T}} + \mathbf{E}_k, \quad k = 1, \ldots K \tag{2}$$

where $\mathbf{A}$ ($I \times R$) and $\mathbf{B}$ ($J \times R$) have the vectors $\mathbf{a}^{(r)}$ and $\mathbf{b}^{(r)}$ as columns, respectively, $\boldsymbol{\Omega}$ ($R \times R$) is the diagonal matrix with the weights $\omega_r$ on its diagonal, and $\mathbf{C}_k$ ($R \times R$) is the diagonal matrix with the $k$th elements of the vectors $\mathbf{c}^{(r)}$ on its diagonal. The model part of the CP model is characterized by ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\boldsymbol{\Omega}$), where $\mathbf{C}$ ($K \times R$) has the vectors $\mathbf{c}^{(r)}$ as columns. We refer to $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ as the component matrices and to $\boldsymbol{\Omega}$ as the weights matrix.

* *Correspondence to: A. Stegeman, Heijmans Institute for Psychological Research, University of Groningen, Grote Kruisstraat 2/1, 9712 TS Groningen, The Netherlands.*
*E-mail: a.w.stegeman@rug.nl*

a *A. Stegeman*
*Heijmans Institute for Psychological Research, University of Groningen, Grote Kruisstraat 2/1, 9712 TS Groningen, The Netherlands*

For later use, we mention that the CP model (1) is a special case of the Tucker3 model of Tucker [8]. The latter is defined as

$$\underline{\mathbf{X}} = \sum_{r=1}^{R}\sum_{p=1}^{P}\sum_{q=1}^{Q} g_{rpq}\left(\mathbf{a}^{(r)} \circ \mathbf{b}^{(p)} \circ \mathbf{c}^{(q)}\right) + \underline{\mathbf{E}} \tag{3}$$

Clearly, the case with $R = P = Q$ and $g_{rpq} = 0$ if $(r, p, q) \neq (r, r, r)$ yields Equation (1). The $R \times P \times Q$ array $\underline{\mathbf{G}}$ with entries $g_{rpq}$ is referred to as the core array.

The most attractive feature of the CP model is its uniqueness property. Kruskal [9] has shown that, for fixed residuals $\underline{\mathbf{E}}$, the vectors $\mathbf{a}^{(r)}$, $\mathbf{b}^{(r)}$ and $\mathbf{c}^{(r)}$ and the weights $\omega_r$ are unique up to sign changes and a reordering of the summands in Equation (1) if

$$k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \geq 2R + 2 \tag{4}$$

where $k_{\mathbf{A}}$, $k_{\mathbf{B}}$, $k_{\mathbf{C}}$ denote the k-ranks of the component matrices. The k-rank of a matrix is the largest number $x$ such that every subset of $x$ columns of the matrix is linearly independent. Hence, contrary to the matrix principal components model, the CP components are rotationally unique if Equation (4) holds. See Stegeman and Sidiropoulos [10] for a more accessible proof of Kruskal's [9] uniqueness condition.

The practical use of CP algorithms to fit the CP model is sometimes complicated by the occurrence of so-called 'degenerate' sequences of CP updates. In such cases, convergence of a CP algorithm becomes very slow (it seems to be caught in a 'swamp', see Mitchell and Burdick [11]), some columns of the component matrices become more and more correlated as the CP algorithm runs longer, and the weights $\omega_r$ of the corresponding outer product arrays become very large. These phenomena were first reported in Harshman and Lundy [12]. Since the term degeneracy has a different meaning in Mathematics, we will refer to the symptoms described above as 'degeneracy' throughout the paper.

In this paper, we present an overview of the literature on the occurrence of different types of 'degeneracy' when fitting the CP model (Section 2). In Section 3, we discuss how to avoid the problems of 'degeneracy' due to the non-existence of an optimal CP solution. For $I \times J \times 2$ arrays, fitting a simultaneous generalized Schur decomposition (SGSD) is a cure against this type of 'degeneracy'. In Section 4, we argue for the application of the SGSD method also for other array formats, when the array has a best fitting CP decomposition with ill-conditioned component matrices, in particular such that it resembles the pattern of a 'degeneracy'. In Section 5, we compare the performance of two SGSD algorithms and the alternating least squares (ALS) CP algorithm in a Monte Carlo study. Finally, Section 6 contains a discussion of our findings.

## 2. THE OCCURRENCE OF 'DEGENERACY' WHEN RUNNING A CP ALGORITHM

Here, we present an overview of the literature on the occurrence of 'degeneracy' when fitting the CP model. When 'degeneracy' is observed, most often exactly two components are involved and they show the following pattern:

- In $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$, the columns $s$ and $t$ become nearly equal up to a sign change, the product of these sign changes being $-1$.
- The weights $\omega_s$ and $\omega_t$ become very large.

This pattern is referred to as a 'two-factor degeneracy'. The two outer product arrays involved in the 'degeneracy' diverge in nearly opposite directions, while their sum still contributes to a better CP fit. Also 'degeneracies' involving three or more components can be encountered, see Paatero [13] and Stegeman [14–16].

In applications of the CP model the following two types of 'degeneracy' have been encountered (for a fixed number of components $R$ and data array $\underline{\mathbf{X}}$):

**I** As the CP algorithm runs, the CP objective value decreases slower and slower, and the pattern of 'degeneracy' becomes more and more severe when the algorithm runs longer. Choosing different starting values cannot change this behavior.
**II** After going through a phase of slow convergence (a 'swamp') with highly correlated components, the CP algorithm terminates with an optimal solution.

In the formulations above, we assume that the CP algorithm used is designed to minimize the sum-of-squares of the residual array $\underline{\mathbf{E}}$.

Type I 'degeneracy' is also referred to as 'strong degeneracy', 'unbounded degeneracy' or 'diverging CP components'. This is a serious problem to the application of the CP model, since these type of outcomes of the CP algorithm do not yield directly interpretable results. Recent work has shown that type I 'degeneracy' is a mathematical property of the approximation problem the CP algorithm is trying to solve. Recall that the CP algorithm is trying to find a best rank-$R$ approximation to the array $\underline{\mathbf{X}}$. In practice, usually $\underline{\mathbf{X}}$ has rank higher than $R$ and, hence, lies outside of the set of arrays of rank at most $R$. Let

$$\mathcal{D}_R = \{\underline{\mathbf{Y}} \in \mathbb{R}^{I\times J\times K} : \underline{\mathbf{Y}} \text{ has rank at most } R\} \tag{5}$$

Then the CP problem can be formulated as

$$\begin{aligned} &\text{Minimize} && \|\underline{\mathbf{X}} - \underline{\mathbf{Y}}\|^2 \\ &\text{subject to} && \underline{\mathbf{Y}} \in \mathcal{D}_R \end{aligned} \tag{6}$$

When $\underline{\mathbf{X}}$ does not lie inside the set $\mathcal{D}_R$, an optimal CP solution (if it exists) will be a boundary point of $\mathcal{D}_R$. Hence, problems may arise when the boundary points of $\mathcal{D}_R$ do not lie in $\mathcal{D}_R$, i.e. when the set $\mathcal{D}_R$ is not closed. De Silva and Lim [17] show that $\mathcal{D}_1$ is closed but $\mathcal{D}_R$ is not closed for $R = 2, \ldots, \min(I, J, K)$. It follows that a best rank-1 approximation of $\underline{\mathbf{X}}$ always exists, but a best rank-$R$ approximation (i.e. an optimal CP solution) may not exist for $2 \leq R \leq \min(I, J, K)$. Also, De Silva and Lim [17] show that no $2 \times 2 \times 2$ array of rank 3 has a best rank-2 approximation. Numerical experiments and mathematical analysis yield the conjecture that $I \times I \times 2$ arrays of rank $I + 1$ do not have a best rank-$I$ approximation (Stegeman [14]). In Stegeman [16] such conjectures are formulated for all $I \times J \times 2$ arrays and all values of $R$, while Stegeman [15] considers several $I \times J \times 3$ arrays.

Kruskal *et al.* [18] conjectured that the non-existence of an optimal CP solution is the cause for type I 'degeneracy'. They reason that any sequence of CP updates produced by a CP algorithm, that approaches the boundary of $\mathcal{D}_R$ such that the



  

objective value approaches to the infimum of the CP objective function, must fail to converge and displays the pattern of 'degeneracy'. This reasoning has been proven correct by Krijnen *et al*. [19].

Further analysis of type I 'degeneracy' is done by Stegeman [14,16], who characterizes the boundary points of $\mathcal{D}_R$ for all $I \times J \times 2$ arrays and all values of $R$, and shows that if a sequence of CP updates converges to a boundary point of $\mathcal{D}_R$ not lying in $\mathcal{D}_R$ itself, then the pattern of 'degeneracy' appears. Analogous results for several $I \times J \times 3$ arrays can be found in Stegeman [15].

Type II 'degeneracy' is also referred to as 'weak degeneracy' or 'bounded degeneracy'. It is first described in Mitchell and Burdick [11], who note that 'swamps' and 'degeneracy' tend to occur together. Type II 'degeneracy' occurs while the CP problem has an optimal solution. To get to the optimal solution faster, several attempts have been made to design modified CP algorithms that spend less time in 'swamps', see e.g. Rayens and Mitchell [20], Kiers [21], Cao *et al*. [22], Rajih *et al*. [23], Zhao [24] and Navasca *et al*. [25]. Type II 'degeneracy' may also be overcome by choosing a different starting value. Paatero [13] constructs a $2 \times 2 \times 2$ example in which $R = 2$ and $\underline{\mathbf{X}}$ has rank 2, but the region between $\underline{\mathbf{X}}$ and the starting value is a 'swamp', i.e. a region where the CP updates of **A**, **B** and **C** have nearly collinear columns. As a result, a type II 'degeneracy' occurs. For some starting values the sequence of CP updates does not even get to $\underline{\mathbf{X}}$, but terminates inside the 'swamp'. This is evidence that also 'swamps', like type I 'degeneracy', are a mathematical property of the set $\mathcal{D}_R$. In particular, the analysis of Stegeman [14–16] shows that 'swamps' occur near parts of the boundary of $\mathcal{D}_R$ where most of the boundary points do not lie in $\mathcal{D}_R$ itself. This confirms the ideas expressed in Mitchell and Burdick [11] and Paatero [13].

Although it is generally believed that slow convergence of CP algorithms occurs due to near linear dependence in the columns of the updates of **A**, **B** and **C**, no theoretical results are known to the author that confirm this claim. It would be interesting to see how the pattern of 'degeneracy' in **A**, **B** and **C** affects the updates in the ALS CP algorithm, or in the gradient-based CP algorithms described in Tomasi and Bro [3]. This, however, is beyond the scope of this paper.

## 3. AVOIDING THE PROBLEM OF TYPE I 'DEGENERACY'

Here, we review the literature on ways to overcome the problem of type I 'degeneracy'. As explained in Section 2, type I 'degeneracy' occurs due to the non-existence of an optimal CP solution.

The CP model is shown to have an optimal solution when orthogonality constraints are imposed on (one of) the components matrices, see Krijnen *et al*. [19]. Also, Lim [26] shows that an optimal solution exists when $\underline{\mathbf{X}}$ is non-negative and the component matrices are restricted to be nonnegative. In both cases, the restrictions on the component matrices make the pattern of 'degeneracy' impossible. The idea to impose orthogonality restrictions is originally due to Harshman and Lundy [12].

To guarantee the existence of an optimal CP solution in the unrestricted CP model, De Silva and Lim [17] propose to consider the closure of the set $\mathcal{D}_R$ in the optimization problem (6). For each array format $I \times J \times K$ and value of $R$ this involves characterizing the boundary arrays of $\mathcal{D}_R$. For $I \times J \times 2$ arrays, this has been done by Stegeman [14,16]. Using these results, a method to avoid type I 'degeneracy' for $I \times J \times 2$ arrays is proposed in Stegeman and De Lathauwer [27]. Instead of the CP model, they propose to fit the SGSD introduced in De Lathauwer *et al*. [28]. In matrix notation, the SGSD model for an array $\underline{\mathbf{X}}$ is

$$\mathbf{X}_k = \mathbf{Q}_a \mathbf{R}_k \mathbf{Q}_b^{\mathrm{T}} + \mathbf{E}_k, \quad k = 1, \ldots K \tag{7}$$

where $\mathbf{Q}_a$ ($I \times R$) and $\mathbf{Q}_b$ ($J \times R$) are column-wise orthonormal and $\mathbf{R}_k$ are $R \times R$ upper triangular, $k = 1, \ldots, K$. The matrices $\mathbf{Q}_a$, $\mathbf{Q}_b$ and $\mathbf{R}_k$ are determined by minimizing the sum-of-squares of the residuals $\mathbf{E}_k$, $k = 1, \ldots, K$. For $R = I = J$, a Jacobi-type algorithm is presented in De Lathauwer *et al*. [28], and an extended QZ algorithm is developed by Van der Veen and Paulraj [29]. In Stegeman and De Lathauwer [27] the Jacobi-type algorithm is generalized to $R \leq I$ and $R \leq J$.

Analogous to Equation (5), let

$$\mathcal{S}_R = \{\underline{\mathbf{Y}} \in \mathbb{R}^{I \times J \times K} : \underline{\mathbf{Y}} \text{ has a full SGSD with } R \text{ components}\} \tag{8}$$

In general, there holds $\mathcal{D}_R \subset \mathcal{S}_R$. Indeed, suppose $\underline{\mathbf{X}}$ satisfies the CP model (2) with perfect fit. Let $\mathbf{A} = \mathbf{Q}_a \mathbf{R}_a$ be a QR-decomposition of **A**, with $\mathbf{Q}_a$ column-wise orthonormal and $\mathbf{R}_a$ upper triangular. Analogously, let $\mathbf{B} = \mathbf{Q}_b \mathbf{L}_b$ be a QL-decomposition of **B**, with $\mathbf{Q}_b$ column-wise orthonormal and $\mathbf{L}_b$ lower triangular. Then $\mathbf{X}_k = \mathbf{Q}_a(\mathbf{R}_a \mathbf{C}_k \mathbf{L}_b^T)\mathbf{Q}_b^T$, $k = 1, \ldots, K$, is a full SGSD for $\underline{\mathbf{X}}$. The converse is not generally true, however. A class of $I \times J \times 2$ arrays $\underline{\mathbf{X}}$ with a full $R$-component SGSD but rank larger than $R$ can be found in Stegeman and De Lathauwer [27].

Stegeman and De Lathauwer [27] show that the SGSD model (7) always has an optimal solution. Moreover, for $I \times J \times 2$ arrays the set $\mathcal{S}_R$ is the closure of $\mathcal{D}_R$. Hence, for $I \times J \times 2$ arrays and $\underline{\mathbf{X}}$ outside of $\mathcal{D}_R$, the optimal SGSD solution, if it is unique, is the limit point of the sequence of CP updates, whether the latter becomes 'degenerate' or not. Stegeman and De Lathauwer [27] show that the optimal SGSD solution may be written as the sum of the 'non-degenerate' rank-1 arrays from the sequence of CP updates, and a Tucker3 part with a sparse core array. The latter part is the limit point of the 'degenerate' part of the sequence of CP updates. This representation of the optimal SGSD solution is obtained from the Jordan normal form of $\mathbf{R}_2 \mathbf{R}_1^{-1}$. Although the representation is not a decomposition into rank-1 arrays, it is a sparse and rotationally unique decomposition and its constituting parts may be interpretable to the researcher. During the execution of the SGSD algorithm in Stegeman and De Lathauwer [27] no problems of slow convergence occurred, while applying a CP algorithm directly would have often resulted in a type I 'degeneracy'.

The fact that a 'degenerate' sequence of CP updates converges to a Tucker3 limit is mathematical proof of the data-analytic explanation for 'degeneracy' proposed by Harshman and Lundy [12], Lundy *et al*. [30] and Harshman [31]. These authors state that 'degeneracy' occurs when 'Parafac is trying to model Tucker variation'. Examples of 'degenerate' sequences of CP updates converging to a Tucker3 limit can also be found in Paatero [13].

## 4. USING THE SGSD TO OVERCOME TYPE II 'DEGENERACY'?

As mentioned in Section 2, several modified CP algorithms have been proposed with the aim of getting through 'swamps'

faster. Here, we discuss the possibility of using the SGSD model (7) instead of the CP model for this purpose. Due to the orthogonality restrictions on $\mathbf{Q}_a$ and $\mathbf{Q}_b$, these matrices cannot have nearly linearly dependent columns. Moreover, the SGSD model is guaranteed to have an optimal solution. These facts suggest that using the SGSD model instead of the CP model may prevent 'degeneracy' of type I as well as type II from occurring. Two observations argue against this optimism, however.

The first observation is the following. As explained in the previous section, there holds $\mathcal{D}_R \subset \mathcal{S}_R$. To obtain a 'good' CP solution from the optimal SGSD solution, the latter is approximated from the set $\mathcal{D}_R$. This can be done by solving a system of linear equations, as described in De Lathauwer *et al*. [28]. However, the thus obtained CP solution may not be optimal, especially when a considerable amount of noise is present. This is because, contrary to the $I \times J \times 2$ case, in general $\mathcal{S}_R$ is not the closure of $\mathcal{D}_R$.

The second observation is made by considering the numerical experiments in De Lathauwer *et al*. [28]. These show that cases of slow convergence also occur when fitting the SGSD model. In particular, Reference [28] consider the case where the array $\underline{\mathbf{X}}$ satisfies a CP decomposition (with some noise added) and vary the condition number of one component matrix. Slow convergence when fitting the SGSD model occurs for large condition numbers. The Jacobi-type algorithm of [28] is more prone to this behavior than the extended QZ algorithm of Van der Veen and Paulraj [29], however.

All things considered, the SGSD model seems worth a try. In the next section, we construct an array $\underline{\mathbf{X}}$ from a CP decomposition in which the first two columns of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are nearly proportional, with some noise added to it. Based on the discussion on type II 'degeneracy' in Section 2, we conjecture that when fitting the CP model to $\underline{\mathbf{X}}$, the algorithm must pass through a 'swamp' region before it reaches the underlying CP solution. We fit the SGSD model to $\underline{\mathbf{X}}$ and compute the CP solution closest to the optimal SGSD solution. For varying degrees of collinearity in $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and several different noise levels, this method is compared with a standard CP algorithm.

## 5. NUMERICAL EXPERIMENTS

Here, we conduct a Monte Carlo study to evaluate the SGSD method as explained in the previous section. Our study is modeled after the one conducted by Tomasi and Bro [3]. We construct arrays $\underline{\mathbf{X}}$ of size $20 \times 20 \times 20$ as follows. The matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are $20 \times 4$ and randomly sampled from the normal distribution with mean 0 and variance 1. Next, their first columns are determined as

$$\mathbf{a}_1 = \mathbf{a}_2 + d\mathbf{y}_a, \quad \mathbf{b}_1 = \mathbf{b}_2 + d\mathbf{y}_b, \quad \mathbf{c}_1 = -\mathbf{c}_2 + d\mathbf{y}_c \tag{9}$$

where $\mathbf{y}_a$, $\mathbf{y}_b$ and $\mathbf{y}_c$ are randomly sampled from the normal distribution with mean 0 and variance 1, and $d$ is a scalar whose value is varied during the experiments. Clearly, smaller values of $d$ result in higher degrees of collinearity in $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. The array $\underline{\mathbf{X}}$ is then constructed by the CP model (1), where the noise array $\underline{\mathbf{E}}$ is randomly sampled from the normal distribution with mean 0 and variance 1, and normalized such that the noise percentage ($\|\underline{\mathbf{E}}\|$ relative to $\|\underline{\mathbf{X}}\|$) has a fixed value, see Tomasi and Bro [3, Appendix D]. We consider the values $d = 0.25$, $d = 0.4$ and $d = 0.7$ and noise percentages 1, 5 and 10. To diminish the influence of chance on our results, we generate 20 arrays $\underline{\mathbf{X}}$ as above, for each combination of $d$ and noise percentage. Hence, a total of 180 arrays are generated in our Monte Carlo study.

It can be seen that the k-ranks of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are all equal to four. Since $R = 4$, the uniqueness of the CP decomposition follows from Equation (4).

The degree of 'degeneracy' of two CP components is usually measured by the 'triple cosine' or 'congruence'. For the first two components, this quantity is defined as

$$\frac{\mathbf{a}_1^{\mathrm{T}}\mathbf{a}_2}{\|\mathbf{a}_1\|\,\|\mathbf{a}_2\|} \quad \frac{\mathbf{b}_1^{\mathrm{T}}\mathbf{b}_2}{\|\mathbf{b}_1\|\,\|\mathbf{b}_2\|} \quad \frac{\mathbf{c}_1^{\mathrm{T}}\mathbf{c}_2}{\|\mathbf{c}_1\|\,\|\mathbf{c}_2\|} \tag{10}$$

where each term is the cosine of the angle between the first and second vector. The congruence lies between $-1$ and 1, and a congruence close to $-1$ indicates a two-factor 'degeneracy' as defined in Section 2. Table I contains the median values of the congruence between the first two components and the condition numbers of the generated $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. As can be seen, $d = 0.25$ yields a congruence of approximately $-0.9$, while $d = 0.4$ and $d = 0.7$ correspond to a congruence of $-0.8$ and $-0.5$, respectively. The values $-0.5$ and $-0.9$ were also used in the Monte Carlo study of Tomasi and Bro [3]. The condition numbers in Table I are slightly larger than in Reference [3]. In Figure 1, the variation of the congruence for the three values of $d$ can be seen. For each $d$, 60 decompositions are generated (20 for each noise level). Also, the relation between the congruence and the sum of the condition numbers of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ is plotted for all 180 generated decompositions. As expected, smaller congruences and larger condition numbers occur together.

We apply three algorithms to $\underline{\mathbf{X}}$ with the aim of obtaining the true $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ of the CP decomposition of $\underline{\mathbf{X}}$. The first two algorithms fit the four-component SGSD model to $\underline{\mathbf{X}}$, and compute the closest CP solution to the optimal SGSD solution. As SGSD algorithms we use the Jacobi-type algorithm of De Lathauwer *et al.* [28] (modified for $R \leq I$ and $R \leq J$ as described in Stegeman and De Lathauwer [27]) and the extended QZ algorithm of Van der Veen and Paulraj [29] (modified for $R \leq I$ and $R \leq J$ analogously). For the computation of the closest CP solution to the optimal SGSD solution, we use the method described in De Lathauwer *et al*. [28]. We compare these two algorithms to the standard ALS CP algorithm with $R = 4$. Despite its simplicity, the latter proves to be a fast and accurate algorithm in the Monte Carlo study of Tomasi and Bro [3]. Note that our Monte Carlo study differs from the numerical experiments of De Lathauwer *et al*. [28], who compare the two SGSD algorithms for arrays with a CP decomposition in which the condition number of only one component matrix is varied.

**Table I.** Median values of the congruence between the first two components and the condition numbers of the component matrices

| | $d = 0.25$ | $d = 0.4$ | $d = 0.7$ |
|---|---|---|---|
| Congruence | −0.91 | −0.81 | −0.52 |
| Cond($\mathbf{A}$) | 9.5 | 6.2 | 3.5 |
| Cond($\mathbf{B}$) | 8.5 | 6.4 | 3.5 |
| Cond($\mathbf{C}$) | 10.0 | 6.5 | 5.0 |



  

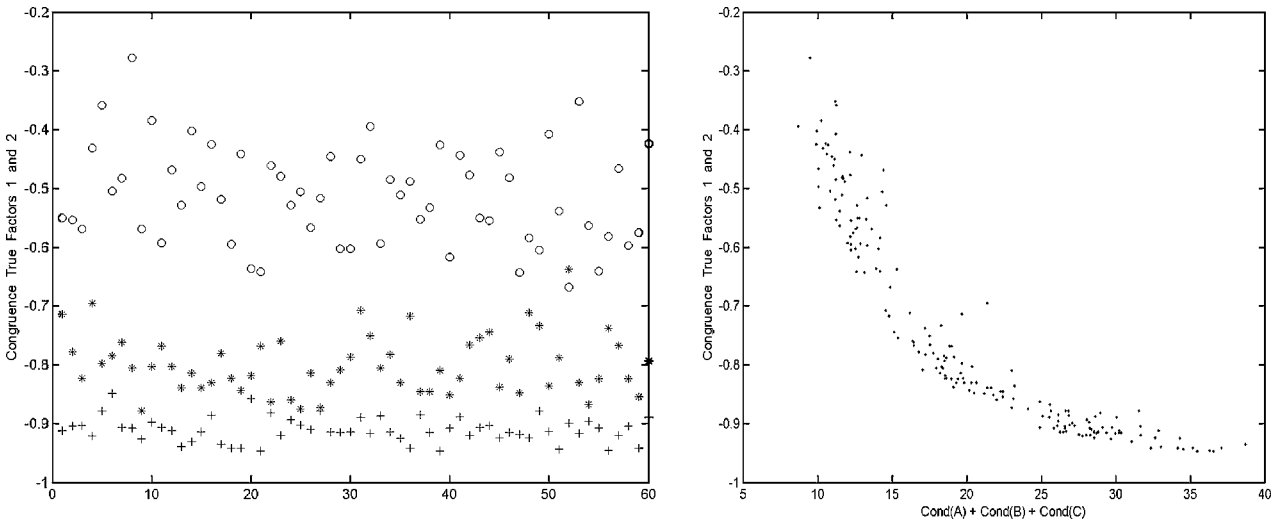


**Figure 1.** Congruence between the first two components for the values $d = 0.25$ (+), 0.4 (∗) and 0.7 (○) in the left figure, and the relation between the congruence and the sum of the condition numbers of the true component matrices in the right figure.

The SGSD Jacobi algorithm monotonically decreases the sum-of-squares of the below-diagonal parts of $\mathbf{R}_k$, $k = 1, \ldots, 20$. A relative decrease of this quantity below `1e-9` is used as a stopping criterion. For the extended QZ algorithm monotonic convergence does not hold and a different stopping criterion needs to be chosen. For this, the Frobenius norm of the difference between the rotations to update $\mathbf{Q}_a$ and $\mathbf{Q}_b$ and diagonal matrices, is used. The algorithm stops when this quantity falls below a threshold of `1e-9`. For the ALS CP algorithm the stopping criterion is a relative decrease of the error sum-of-squares below `1e-9`.

For each run of the algorithms, their initial values are determined as follows. For the SGSD algorithms, five iterations of the SGSD Jacobi algorithm are performed for 10 random initial values. The best result from these 10 runs is used as initial value for both the SGSD Jacobi and extended QZ algorithms. For the ALS CP algorithm, five ALS iterations are performed for 10 random initial values. The best result is used as initial value.

To determine whether the true CP decomposition is recovered, we compute the congruence of the obtained CP solution with the true CP decomposition. For a full recovery of the true CP decomposition, the congruence must be higher than some threshold value for all four true components. The same criterion is used in Tomasi and Bro [3]. Table II contains the percentages of full recovery for the three algorithms, different values of $d$ and noise percentages, and congruence thresholds of 0.95 and 0.97. For $d = 0.4$ and 0.7 there is not much difference between the methods. For $d = 0.25$ the SGSD methods are less accurate than the ALS CP algorithm, especially for higher noise levels. The SGSD extended QZ algorithm yields the least accurate results in this case. Note that all three methods were able to recover the third and fourth components in all runs. The differences in accuracy occur due to failure to recover the first two nearly collinear components. The accuracy of the SGSD extended QZ algorithm did not improve for a stopping criterion of `1e-12` or when relative decrease of error sum-of-squares is used as a stopping criterion.

The error sum-of-squares between $\underline{\mathbf{X}}$ and the obtained CP decomposition was smallest for ALS CP in all 180 runs. This is as expected since the SGSD methods need an additional approximation step to obtain a CP decomposition. In 158 of the 180 runs, the error sum-of-squares was smaller for the SGSD Jacobi algorithm than for the SGSD extended QZ algorithm.

Next, we consider the time consumption of the three algorithms. For the SGSD algorithms, this includes the time to compute a CP decomposition from the obtained SGSD solution. In Table III, the median values of time consumption are given for the different values of $d$ and noise percentage. As can be

**Table II.** Percentages of full recovery for the three algorithms. Each cell contains the percentages for noise levels 1, 5 and 10

| | $d = 0.25$ | $d = 0.4$ | $d = 0.7$ |
|---|---|---|---|
| SGSD Jacobi | 100, 90, 75 | 100, 100, 100 | 100, 100, 100 |
| SGSD ext QZ | 100, 85, 65 | 100, 100, 100 | 100, 100, 100 |
| ALS CP | 100, 100, 95 | 100, 100, 100 | 100, 100, 100 |
| SGSD Jacobi | 100, 85, 60 | 100, 100, 100 | 100, 100, 100 |
| SGSD ext QZ | 100, 70, 45 | 100, 100, 80 | 100, 100, 95 |
| ALS CP | 100, 100, 90 | 100, 100, 100 | 100, 100, 100 |

Congruence thresholds are 0.95 (top rows) and 0.97 (bottom rows).

**Table III.** Median values of time consumption in seconds for the three algorithms

| | $d = 0.25$ | $d = 0.4$ | $d = 0.7$ |
|---|---|---|---|
| SGSD Jacobi | 3.1, 1.3, 1.3 | 1.2, 0.6, 0.8 | 0.9, 0.8, 0.7 |
| SGSD ext QZ | 2.7, 1.9, 1.2 | 1.1, 1.1, 1.2 | 0.5, 0.4, 0.4 |
| ALS CP | 3.3, 3.6, 2.9 | 1.4, 1.1, 0.9 | 0.2, 0.2, 0.2 |

Each cell contains the times for noise levels 1, 5 and 10.

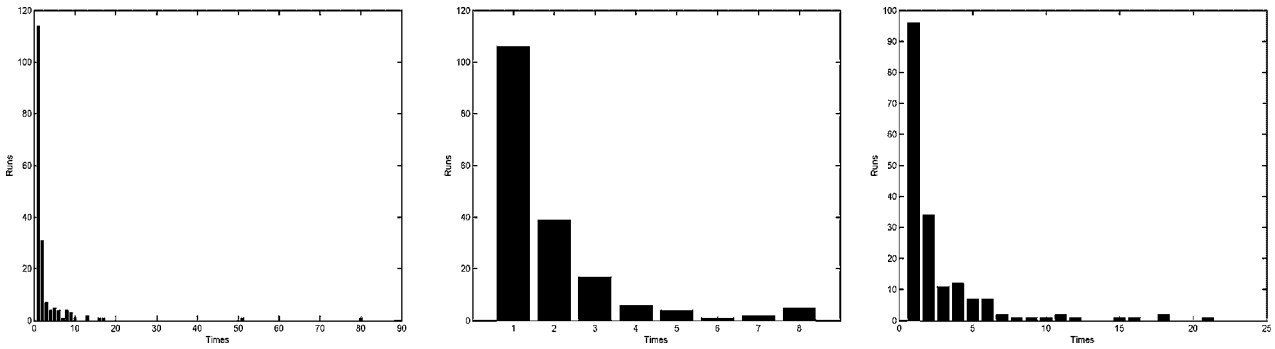


**Figure 2.** Histograms of time consumption for all 180 runs of the SGSD Jacobi (left), SGSD extended QZ (middle) and ALS CP (right) algorithms.

seen, the times go down when more noise is added. Fitting an array with a less 'degenerate' CP decomposition takes less time for all three algorithms. For $d = 0.25$ both SGSD algorithm are faster than ALS CP, for $d = 0.4$ one of the SGSD algorithms is faster than ALS CP, and ALS CP is fastest for $d = 0.7$. A striking difference between the algorithms is seen in Figure 2, where histograms of the time consumption are depicted. Both the SGSD Jacobi and ALS CP algorithms have long tails on their times distribution. The SGSD extended QZ algorithm on the other hand does not seem to be affected by slow convergence at all. This was also observed in the numerical experiments conducted by De Lathauwer *et al*. [28]. Note that apart from two extremely long runs (51 and 80 s) the histogram of the time consumption of the SGSD Jacobi algorithm is about the same as that of the ALS CP algorithm.

Although the SGSD methods are faster for higher degrees of collinearity in the true **A**, **B** and **C**, they are less accurate than ALS CP in these cases. Hence, our Monte Carlo study does not yield good results for the SGSD methods. Next, we assess the performance of the three algorithms in a situation where the true **A**, **B** and **C** have high condition numbers but the pattern of a 'degeneracy' is absent. Also, very little noise is added. This case is considered in Hopke *et al*. [32]. We consider their simulated datasets PP1 and PP2. For PP1, we take the true **A**, **B** and **C** from Hopke *et al.* [32, Figure 1]. The array $\underline{\mathbf{X}}$ has size $10 \times 8 \times 5$ and $R = 3$ in its CP decomposition. The added noise is 0.025 times a randomly sampled array from the normal distribution with mean 0 and variance 1. For PP2, we take the true **A**, **B** and **C** from Hopke *et al.* [32, Figure 2]. The array $\underline{\mathbf{X}}$ has size $10 \times 8 \times 5$ and $R = 4$ in its CP decomposition. The added noise is 0.001 times a randomly sampled array as for PP1. The condition numbers for **A**, **B** and **C** in PP1 are 18, 18 and 83, respectively. The smallest congruence between two components equals 0.22. Hence, we have high condition numbers for the component matrices but no pattern of 'degeneracy' as described in Section 2. For PP2, the condition numbers for **A**, **B** and **C** are 49, 48 and 52, respectively. The smallest congruence between two components equals 0.01. Note that the condition numbers for PP1 and PP2 are much larger than in our Monte Carlo study above. For both datasets the component matrices have full k-rank and uniqueness of the CP decompositions follows from Equation (4).

For PP1 and PP2, we generated 20 arrays with random noise and applied the three algorithms with the correct number of components. The stopping criteria were set at `1e-6`. Table IV contains the percentages of full recovery and time consumption of the three algorithms. The algorithms are approximately equally accurate, but as in our Monte Carlo study their time consumption differs dramatically. While the SGSD Jacobi and ALS CP algorithms have a highly varying time consumption, the SGSD extended QZ algorithm is consistently much faster. Hence, it seems that in cases of little noise and large condition numbers of **A**, **B** and **C** the SGSD extended QZ algorithm is to be preferred, while in cases of more noise and up to moderate condition numbers the ALS CP algorithm (or possibly another CP algorithm) is the best choice.

Another possibility was suggested by an anonymous reviewer: try to combine the best of the SGSD extended QZ algorithm (i.e. speed) with the best of the ALS CP algorithm (i.e. accuracy). For example, one could use the outcome of the SGSD extended QZ algorithm as initial values for ALS CP. We tried this in our first Monte Carlo study (results not reported). Although, this increases the accuracy of ALS CP somewhat, extremely long runs still occur in ALS CP. Moreover, the total time consumption of SGSD extended QZ and ALS CP together is about 1.5 times as large as when ALS CP is run with initial values chosen as in the simulations described above.

**Table IV.** Percentages of full recovery (congruence $\geq 0.97$) and median, minimum and maximum values of time consumption in seconds for the three algorithms and datasets PP1 (top rows) and PP2 (bottom rows)

| | Full recovery (%) | Median time | Min time | Max time |
|---|---|---|---|---|
| SGSD Jacobi | 100 | 9.9 | 5.5 | 27.0 |
| SGSD ext QZ | 85 | 0.14 | 0.09 | 0.47 |
| ALS CP | 100 | 2.4 | 0.03 | 7.0 |
| SGSD Jacobi | 80 | 9.0 | 5.6 | 36.3 |
| SGSD ext QZ | 95 | 0.25 | 0.20 | 0.34 |
| ALS CP | 90 | 9.7 | 4.9 | 62.3 |



  

## 6. DISCUSSION

In this paper, we discussed the phenomenon of 'degeneracy' when fitting the CP model. We mentioned that two types of 'degeneracy' can be distinguished. The first type is due to the non-existence of an optimal CP solution. To overcome this type of 'degeneracy' the CP model must be changed somehow, either by imposing additional restrictions (such as orthogonality or non-negativity) or by solving a closely related model (such as the SGSD model) and approximating its optimal solution by a CP decomposition. The second type of 'degeneracy' occurs when the sequence of CP updates passes through a region of slow convergence (a 'swamp') before terminating at the optimal solution. In this case, modified CP algorithms designed to spend less time in 'swamps' may be a cure.

We proposed to use the SGSD method for the latter purpose. In our Monte Carlo study, we assessed the performance of two SGSD algorithms and compared it to the standard ALS CP algorithm. In cases where the true CP decomposition resembles a two-factor 'degeneracy' the SGSD algorithms are faster but less accurate than ALS CP. However, the accuracy of the SGSD algorithms improves when the component matrices have larger condition numbers and the noise level is lower. We conjecture that this is due to the true CP decomposition being closer to the boundary of $\mathcal{D}_R$. Then the distance between the optimal SGSD solution and its approximation by a CP decomposition is smaller.

As in the numerical experiments of De Lathauwer *et al*. [28], we observed that the SGSD extended QZ algorithm has a very consistent low time consumption as compared to the SGSD Jacobi and ALS CP algorithms. For the difference between the two SGSD algorithms, we suggest the following explanation. Both algorithms are designed to minimize the sum-of-squares of the below-diagonal parts of $\mathbf{R}_k$, $k = 1, \ldots, K$. The SGSD Jacobi algorithm decreases this quantity monotonically, while the SGSD extended QZ algorithm does not. In each iteration, the SGSD Jacobi algorithm applies orthogonal rotations to the rows and columns of the $\mathbf{R}_k$ such that the most energy is transferred from their below-diagonal parts to their upper-triangular parts, averaged over all upper-triangular elements and all $k$. The SGSD extended QZ algorithm works in the same way, except that the rotations are determined such that the most energy is transferred to the diagonals of $\mathbf{R}_k$. This mechanism within the SGSD extended QZ algorithm may function as a tendency to 'move towards a CP decomposition with orthogonal **A** and **B**' (the latter has diagonal $\mathbf{C}_k$ instead of upper-triangular $\mathbf{R}_k$). Combined with its non-monotonic convergence, this may explain why the SGSD extended QZ algorithm is less affected by 'swamps' than the SGSD Jacobi algorithm.

For the computation of a 'good' approximation of the optimal SGSD solution by a CP decomposition we solved an overdetermined system of linear equations, as proposed by De Lathauwer *et al*. [28]. When the obtained CP decomposition has ill-conditioned **A**, **B** or **C**, this linear system also becomes ill-conditioned. In our Monte Carlo experiments, however, we encountered no problems when solving this linear system. In the numerical experiments of Stegeman and De Lathauwer [27] 'degeneracy' of type I occurs and the linear system becomes too ill-conditioned to solve numerically. This problem is circumvented by using the Jordan transform of $\mathbf{R}_2\mathbf{R}^{-1}$ instead. In cases where the true CP decomposition features exact proportional columns of **A**, **B** or **C**, the SGSD algorithms can still be used. The linear system, however, then becomes singular and has no unique solution. This reflects the non-uniqueness of the CP decomposition in this case.

### Acknowledgements

The research is supported by the Dutch Organisation for Scientific Research (NWO), VENI grant 451-04-102. The author thanks Lieven De Lathauwer for sharing the Jacobi and extended QZ algorithms for the SGSD model.

## REFERENCES

1. Carroll JD, Chang JJ. Analysis of individual differences in multidimensional scaling via an *n*-way generalization of Eckart-Young decomposition. *Psychometrika* 1970; **35**: 283–319.
2. Harshman RA. Foundations of the Parafac procedure: models and conditions for an "explanatory" multimodal factor analysis. *UCLA Working Papers in Phonetics* 1970; **16**: 1–84.
3. Tomasi G, Bro R. A comparison of algorithms for fitting the Parafac model. *Comput. Stat. Data Anal.* 2006; **50**: 1700–1734.
4. Kroonenberg PM. *Applied Multiway Data Analysis*. Wiley Series in Probability and Statistics, 2008.
5. Smilde A, Bro R, Geladi P. *Multi-way Analysis: Applications in the Chemical Sciences*. Wiley: Chichester, 2004.
6. Sidiropoulos ND, Giannakis G, Bro R. Blind Parafac receivers for DS-CDMA systems. *IEEE Trans. Signal Process.* 2000; **48**: 810–823.
7. Sidiropoulos ND, Bro R, Giannakis G. Parallel factor analysis in sensor array processing. *IEEE Trans. Signal Process.* 2000; **48**: 2377–2388.
8. Tucker LR. Some mathematical notes on three-mode factor analysis. *Psychometrika* 1966; **31**: 279–311.
9. Kruskal JB. Three-way arrays: rank and uniqueness of trilinear decompositions, with applications to arithmetic complexity and statistics. *Linear Algebra Appl.* 1977; **18**: 95–138.
10. Stegeman A, Sidiropoulos ND. On Kruskal's uniqueness condition for the Candecomp/Parafac decomposition. *Linear Algebra Appl.* 2007; **420**: 540–552.
11. Mitchell BC, Burdick DS. Slowly converging Parafac sequences: swamps and two-factor degeneracies. *J. Chemometr.* 1994; **8**: 155–168.
12. Harshman RA, Lundy ME. Data preprocessing and the extended Parafac model. In *Research Methods for Multimode Data Analysis*, Law HG, Snyder CW, Jr., Hattie JA, McDonald RP (eds). Praeger: New York, 1984; 216–284.
13. Paatero P. Construction and analysis of degenerate Parafac models. *J. Chemometr.* 2000; **14**: 285–299.
14. Stegeman A. Degeneracy in Candecomp/Parafac explained for $p \times p \times 2$ arrays of rank $p + 1$ or higher. *Psychometrika* 2006; **71**: 483–501.
15. Stegeman A. Degeneracy in Candecomp/Parafac and Indscal explained for several three-sliced arrays with a two-valued typical rank. *Psychometrika* 2007; **72**: 601–619.
16. Stegeman A. Low-rank approximation of generic $p \times q \times 2$ arrays and diverging components in the Candecomp/Parafac model. *SIAM J. Matrix Anal. Appl.* 2008; **30**: 988–1007.
17. De Silva V, Lim L-H. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM J. Matrix Anal. Appl.* 2008; **30**: 1084–1127.
18. Kruskal JB, Harshman RA, Lundy ME. How 3-MFA data can cause degenerate Parafac solutions, among other relationships. In *Multiway Data Analysis*, Coppi R, Bolasco S (eds). North-Holland: Amsterdam, 1989; 115–121.
19. Krijnen WP, Dijkstra TK, Stegeman A. On the non-existence of optimal solutions and the occurrence of "degeneracy" in the Candecomp/Parafac model. *Psychometrika* 2008; **73**: 431–439.
20. Rayens WS, Mitchell BC. Two-factor degeneracies and a stabilization of Parafac. *Chemometr. Intell. Lab. Syst.* 1997; **38**: 173–181.
21. Kiers HAL. A three-step algorithm for Candecomp/Parafac analysis of large data sets with multicollinearity. *J. Chemometr.* 1998; **12**: 155–171.
22. Cao YZ, Chen ZP, Mo CY, Wu HL, Yu RQ. A Parafac algorithm using penalty diagonalization error (PDE) for three-way data array resolution. *Analyst* 2000; **125**: 2303–2310.

23. Rajih M, Comon P, Harshman RA. Enhanced line search: a novel method to accelerate Parafac. *SIAM J. Matrix Anal. Appl.* 2008; **30**: 1128–1147.
24. Zhao H-G. A heuristic method for computing the best rank-*r* approximation to higher-order tensors. *Int. J. Contemp. Math. Sci.* 2008; **3**: 471–476.
25. Navasca C, De Lathauwer L, Kindermann S. Swamp reducing technique for tensor decomposition. *Proceedings of the 16th European Signal Processing Conference (EUSIPCO 2008)*, Lausanne, Switzerland, 25–29 August 2008.
26. Lim L-H. Optimal solutions to non-negative Parafac/multilinear NMF always exist. Talk at the *Workshop on Tensor Decompositions and Applications*, CIRM, Luminy, Marseille, France, August 29–September 2 2005.
27. Stegeman A, De Lathauwer L. A method to avoid diverging components in the Candecomp/Parafac model for generic $I \times J \times 2$ arrays. *SIAM J. Matrix Anal. Appl.* 2009; **30**: 1614–1638.
28. De Lathauwer L, De Moor B, Vandewalle J. Computation of the canconical decomposition by means of a simultaneous generalized Schur decomposition. *SIAM J. Matrix Anal. Appl.* 2004; **26**: 295–327.
29. Van der Veen A-J, Paulraj A. An analytical constant modulus algorithm. *IEEE Trans. Signal Process.* 1996; **44**: 1136–1155.
30. Lundy ME, Harshman RA, Kruskal JB. A two-stage procedure incorporating good features of both trilinear and quadrilinear models. In *Multiway Data Analysis*, Coppi R, Bolasco S (eds). North-Holland: Amsterdam, 1989; 123–130.
31. Harshman RA. The problem and nature of degenerate solutions or decompositions of 3-way arrays. Talk at the *Workshop on Tensor Decompositions*, AIM, Palo Alto, USA, 19–23 July 2004.
32. Hopke PK, Paatero P, Jia H, Ross RT, Harshman RA. Three-way (Parafac) factor analysis: examination and comparison of alternative computational methods as applied to ill-conditioned data. *Chemometr. Intell. Lab. Syst.* 1998; **43**: 25–42.



  