Computational Statistics & Data Analysis 16 (1993) 103–118
North-Holland

An alternating least squares algorithm for PARAFAC2 and three-way DEDICOM

Henk A.L. Kiers
University of Groningen, Netherlands

Received July 1991
Revised March 1992

Abstract: PARAFAC2 is a method for analyzing three-way data consisting of symmetric frontal slices. Three-way DEDICOM can be considered a generalization of PARAFAC2 in that it fits essentially the same model to three-way data consisting of square frontal slices that may be asymmetric. In the present paper, an alternating least squares algorithm is developed for three-way DEDICOM, and an algorithm for PARAFAC2 is derived from it. The performance of the algorithms is studied for some empirical and synthetical data sets.

Keywords: Three-way data; Longitudinal; Factor analysis.

PARAFAC is an interesting generalization of principal components analysis for the situation that the same variables have been observed on the same observation units a number of times (Harshman, 1970; Harshman and Lundy, 1984). One of the main features of the PARAFAC method is that it gives dimensions that (under mild conditions) are unique up to scaling and permutation. The standard PARAFAC method (also denoted as PARAFAC1) cannot be used for cross-sectional data. That is, PARAFAC1 cannot be used for analyzing data that have been observed on different samples of observation units. For an exploratory analysis of such cross-sectional data, taking into account that the same variables have been observed, Harshman (1972; see also Carroll and Wish, 1974, p. 94–96; Harshman and Lundy, 1984, p. 187) proposed to fit the PARAFAC2 model to the covariance matrices obtained in the different samples. If C_k denotes the $(m \times m)$ covariance matrix at occasion k, then the PARAFAC2 model can be described as

$$C_k = AD_kHD_kA' + E_k, \tag{1}$$

$k = 1, \dots, K$, where A denotes an $m \times r$ matrix of 'loadings' for the m variables

Correspondence to: Henk A.L. Kiers, Department of Psychology (P/OP), Grote Kruisstr. 2/1, 9712 TS Groningen, The Netherlands.

on a set of r 'latent factors', D_k denotes an $r \times r$ diagonal matrix containing 'saliences' for the different dimensions, H denotes a symmetric (and usually positive semi-definite) matrix with relations between the factors, and E_k is a matrix with error terms. For details on the interpretation of the various parameter sets the reader is referred to Harshman (1972), Carroll and Wish (1974, p. 94–96), Dunn and Harshman (1982), or Kroonenberg (1983), among others.

The PARAFAC2 model was proposed as an alternative for the PARAFAC1 model. As mentioned, the latter model is (under mild conditions) unique. Harshman (1972, p. 36) has found in analyses of synthetic data that PARAFAC2 has "unique solutions, if the number of factors extracted is not greater than the number 'actually in the data' (used to create the synthetic data), and if there are a sufficient number of independent C_k matrices" (Harshman, 1972, p. 36). Carroll and Wish (1974, p. 95–96) give some further results regarding uniqueness of the PARAFAC2 solution, among which the statement that dimensions are not unique in case K = 2. Neither of these have given sufficient conditions for uniqueness of the PARAFAC2 solution, but current research by Harshman and Lundy (in preparation) will probably fill this gap.

The PARAFAC2 model is fitted to a set of covariance matrices in the least squares sense by minimizing

$$\sigma_1(A, H, D_1, \dots, D_K) = \sum_{k=1}^{K} \| C_k - AD_k H D_k A' \|^2. \tag{2}$$

Harshman (1972, p. 40–41) briefly sketches an algorithm by Jennrich for minimizing this function. Carroll and Wish (1974, p. 95) mention that "it takes into account asymmetries in data and solutions in ways that may not be optimal", and that Carroll and Chang have developed an algorithm that seems to have some advantage over Jennrich's method. The status of these algorithms is unclear, and apparently a PARAFAC2 algorithm with good convergence properties has not yet been developed. One purpose of the present paper is to fill this gap, and propose an algorithm for minimizing (2) that converges monotonically to a stable function value.

As noted by Harshman and Lundy (1984), if the PARAFAC2 model is applied to a set of square matrices describing asymmetric relationships among n things (e.g., confusion data, or transition data), then we obtain the three-way DEDICOM model proposed by Harshman (1978; see also Harshman, Green, Wind and Lundy, 1982). That is, the three-way DEDICOM model (DEDICOM3) can be written as

$$X_k = AD_k R D_k A' + E_k, \tag{3}$$

$k = 1, \dots, K$, where A is an $m \times r$ matrix of coordinates of objects (persons, stimuli) on basic aspects ('factors'), R is a matrix with asymmetric relations between the factors, and the diagonal matrices $D_1, \dots, D_K$ contain weights for the 'factors' to allow for different saliences of factors at different occasions. Clearly, the only difference between (1) and (3) is that H (in (1)) is symmetric, or even p.s.d., whereas R (in (3)) can be any square (generally asymmetric) matrix. In fact, PARAFAC2 can be considered a constrained variant of DEDICOM3.

In order to fit the DEDICOM3 model we have to minimize the function

$$\sigma_2(A, R, D_1, \ldots, D_K) = \sum_{k=1}^{K} \| X_k - AD_k RD_k A' \|^2. \tag{4}$$

To the author's knowledge, an algorithm for this method is not available yet. The main purpose of the present paper is to propose an algorithm for minimizing (4). An algorithm for PARAFAC2 will later be derived by adjusting the DEDICOM3 algorithm.

An alternating least squares algorithm for DEDICOM3

The DEDICOM3 function σ_2 has to be minimized over the three parameter sets A, R, and $\{D_1, \ldots, D_K\}$. It is proposed here to decrease σ_2 alternatingly over A while R and $D_1, \ldots, D_K$ are fixed, over R while A and $D_1, \ldots, D_K$ are fixed, and over $D_1, \ldots, D_K$ while A and R are fixed. In this way, σ_2 decreases monotonically, and because σ_2 is bounded below, this procedure will converge to a stable function value of σ_2. We will now discuss *how* σ_2 is decreased over the three different parameter sets.

Updating A

The most straightforward way to decrease σ_2 over A would be to *minimize* it over A (while R and $D_1, \ldots, D_K$ are considered fixed). However, there does not seem to be a closed from solution for this minimization problem. In the context of a related problem, Kiers (1989) proposed to minimize his function over A columnwise. A similar approach is taken here. That is, writing σ_2 in terms of the columns $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_r$ of A, we find

$$\begin{aligned}
\sigma_2(A) &= \sum_{k=1}^{K} \| X_k - AD_k RD_k A' \|^2 \\
&= \sum_{k=1}^{K} \| X_k \|^2 - 2 \sum_{k=1}^{K} \operatorname{tr} A'X_k AD_k R'D_k \\
&\quad + \sum_{k=1}^{K} \operatorname{tr} A'AD_k RD_k A'AD_k R'D_k \\
&= \sum_{k=1}^{K} \| X_k \|^2 - 2 \sum_{k=1}^{K} \sum_{p=1}^{r} \sum_{q=1}^{r} \boldsymbol{a}'_p X_k \boldsymbol{a}_q d_{kq} r_{pq} d_{kp} \\
&\quad + \sum_{k=1}^{K} \sum_{p=1}^{r} \sum_{q=1}^{r} \sum_{s=1}^{r} \sum_{t=1}^{r} \boldsymbol{a}'_p \boldsymbol{a}_q d_{kq} r_{qs} d_{ks} \boldsymbol{a}'_s \boldsymbol{a}_t d_{kt} r_{pt} d_{kp},
\end{aligned} \tag{5}$$

where d_{kp} denotes the pth diagonal element of D_k, and r_{gh} denotes the element (g, h) of R. Expression (5) can be simplified by constraining the colums of A to have unit length. This constraint can be imposed without loss of generality, because any scaling of the columns of A can be compensated for by an inverse scaling of the matrices D_k. Using this constraint, and isolating one column of A, say $\boldsymbol{a}_l$, we can write σ_2 as a function of this single column as

$$\sigma_2(\boldsymbol{a}_l) = c + \boldsymbol{a}_l'C\boldsymbol{a}_l - 2\boldsymbol{z}'\boldsymbol{a}_l, \tag{6}$$

where the matrix C and the vector $\boldsymbol{z}$ are defined as

$$C \equiv 2\sum_{k=1}^{K}\left(\sum_{p\neq l}^{r}\sum_{s\neq l}^{r} d_{kp}d_{ks}d_{kl}^2(r_{ls}r_{pl} + r_{ll}r_{sp})\boldsymbol{a}_p\boldsymbol{a}_s' - d_{kl}^2 r_{ll}X_k\right) \tag{7}$$

and

$$\boldsymbol{z} \equiv \sum_{k=1}^{K}\sum_{p\neq l}^{r} d_{kp}d_{kl}\Bigg((r_{pl}X_k' + r_{lp}X_k)\boldsymbol{a}_p - d_{kl}^2 r_{ll}(r_{pl} + r_{lp})\boldsymbol{a}_p - \sum_{q\neq l}^{r}\sum_{s\neq l}^{r} d_{kq}d_{ks}(r_{qs}r_{pl} + r_{sq}r_{lp})\boldsymbol{a}_p'\boldsymbol{a}_q\boldsymbol{a}_s\Bigg), \tag{8}$$

as derived in the Appendix. The matrix C is generally asymmetric. However, C can be replaced by the symmetric matrix $S = \frac{1}{2}(C + C')$, without affecting the function value. Hence minimizing σ_2 over $\boldsymbol{a}_l$ reduces to minimizing the function

$$f(\boldsymbol{a}_l) = \boldsymbol{a}_l'S\boldsymbol{a}_l - 2\boldsymbol{z}'\boldsymbol{a}_l \tag{9}$$

over $\boldsymbol{a}_l$, subject to $\boldsymbol{a}_l'\boldsymbol{a}_l = 1$. The solution for a similar type of problem has been given by Ten Berge and Nevels (1977). In their problem S is a diagonal matrix. Our problem can be transformed into a special case of their problem by substituting the eigendecomposition $S = UDU'$ for S (where D has elements in weakly descending order, and U is an orthonormal matrix). If we define $\tilde{\boldsymbol{a}}_l \equiv U'\boldsymbol{a}_l$, and $\boldsymbol{x} \equiv U'\boldsymbol{z}$, then it is readily verified that

$$f(\boldsymbol{a}_l) = g(\tilde{\boldsymbol{a}}_l) = \tilde{\boldsymbol{a}}_l'D\tilde{\boldsymbol{a}}_l - 2\boldsymbol{x}'\tilde{\boldsymbol{a}}_l. \tag{10}$$

It follows that minimizing f over $\boldsymbol{a}_l$ subject to $\boldsymbol{a}_l'\boldsymbol{a}_l = 1$ is equivalent to minimizing g over $\tilde{\boldsymbol{a}}_l$ subject to $\tilde{\boldsymbol{a}}_l'\tilde{\boldsymbol{a}}_l = 1$. Ten Berge and Nevels (1977) have described a procedure for obtaining the global minimum of g. Having found the $\tilde{\boldsymbol{a}}_l$ that minimizes g, the $\boldsymbol{a}_l$ that minimizes f can be derived as $\boldsymbol{a}_l = U\tilde{\boldsymbol{a}}_l$. Minimizing f, and hence σ_2 over each of the colums of A successively, we decrease σ_2 monotonically (or at least, we do 'not increase' σ_2, which for convenience will be denoted as 'decrease' in the sequel). Hence one complete cycle of updating all colums of A decreases σ_2.

Updating R

The next problem is that of decreasing σ_2 over R while A, and $D_1, \ldots, D_K$ are fixed. For this problem there is a closed form solution for R that *minimizes* σ_2.

This solution is closely related to the one obtained by Kiers and Ten Berge (1989) for a similar problem. That is, first the matrices X_k and AD_kRD_kA' in function σ_2 are strung out row-wise into column vectors. The resulting vectors are denoted as $\text{Vec}(X_k)$ and $\text{Vec}(AD_kRD_kA')$. Next, $\text{Vec}(AD_kRD_kA')$ is further elaborated as $\text{Vec}(AD_kRD_kA') = (AD_k \otimes AD_k)\text{Vec}(R)$, where $\otimes$ denotes the Kronecker-product (see Henderson and Searle, 1981; Kiers and Ten Berge, 1989). Finally, stacking all vectorized matrices below each other we find

$$\sigma_2(R) = \left\| \begin{pmatrix} \text{Vec}(X_1) \\ \vdots \\ \text{Vec}(X_K) \end{pmatrix} - \begin{pmatrix} (AD_1 \otimes AD_1)\ \text{Vec}(R) \\ \vdots \\ (AD_K \otimes AD_K)\ \text{Vec}(R) \end{pmatrix} \right\|^2$$

$$= \left\| \begin{pmatrix} \text{Vec}(X_1) \\ \vdots \\ \text{Vec}(X_K) \end{pmatrix} - \begin{pmatrix} (AD_1 \otimes AD_1) \\ \vdots \\ (AD_1 \otimes AD_K) \end{pmatrix} \text{Vec}(R) \right\|^2. \tag{11}$$

Clearly, minimizing (11) over $\text{Vec}(R)$ is a multiple regression problem. Hence the solution for $\text{Vec}(R)$ is

$$\text{Vec}(R) = \left(\sum_{k=1}^{K} (D_kA'AD_k) \otimes (D_kA'AD_k) \right)^{-1} \sum_{k=1}^{K} (D_kA' \otimes D_kA')\ \text{Vec}(X_k)$$

$$= \left(\sum_{k=1}^{K} (D_kA'AD_k) \otimes (D_kA'AD_k) \right)^{-1} \sum_{k=1}^{K} \text{Vec}(D_kA'X_kAD_k). \tag{12}$$

If $(\sum_{k=1}^{K}(D_kA'AD_k) \otimes (D_kA'AD_k))$ is singular, then the generalized Moore–Penrose inverse is taken instead of the regular inverse. The solution for R can be obtained from (12) by 'unstacking' $\text{Vec}(R)$. Because this solution for R minimizes σ_2 over R, given A and $D_1, \ldots, D_K$, it decreases σ_2.

Updating $D_1, \ldots, D_K$

The final step in the algorithm is that of decreasing σ_2 by updating $D_1, \ldots, D_K$, for given A and R. Again, the most straight-forward way would be to minimize σ_2 over $D_1, \ldots, D_K$. In order to do so, it is useful to note that the problem consists of K independent minimization problems. That is, for each k we have to minimize a function of the form

$$\sigma_3(D_k) = \| X_k - AD_kRD_kA' \|^2. \tag{13}$$

Because a closed form solution for minimizing this function over D_k is not available, we propose the following elementwise procedure. Writing σ_3 in terms of the elements $d_{k1},\ldots,d_{kr}$ of D_k we find analogous to (5):

$$\sigma_3(D_k) = \| X_k \|^2 - 2\sum_{p=1}^{r}\sum_{q=1}^{r} d_{kp}d_{kq}r_{pq}\boldsymbol{a}_p' X_k \boldsymbol{a}_q$$
$$+ \sum_{p=1}^{r}\sum_{q=1}^{r}\sum_{s=1}^{r}\sum_{t=1}^{r} d_{kp}d_{kq}d_{ks}d_{kt}r_{qs}r_{pt}\boldsymbol{a}_p'\boldsymbol{a}_q\boldsymbol{a}_s'\boldsymbol{a}_t. \tag{14}$$

Writing σ_3 as a function of just one element of D_k, say d_{kl}, and writing f for the terms independent of d_{kl}, we find, in a similar way as in deriving (6)

$$\sigma_3(d_{kl}) = ad_{kl}^4 + bd_{kl}^3 + cd_{kl}^2 + ed_{kl} + f, \tag{15}$$

where

$$a \equiv r_{ll}^2, \tag{16}$$

$$b \equiv 2\sum_{p\neq l}^{r} d_{kp}r_{ll}(r_{pl}+r_{lp})\boldsymbol{a}_p'\boldsymbol{a}_l, \tag{17}$$

$$c \equiv \left(\sum_{p\neq l}^{r}\sum_{q\neq l}^{r} d_{kp}d_{kq}\big((r_{pl}r_{ql}+r_{lp}r_{lq})\boldsymbol{a}_p'\boldsymbol{a}_q + 2(r_{lp}r_{ql}+r_{ll}r_{pq})\boldsymbol{a}_p'\boldsymbol{a}_l\boldsymbol{a}_q'\boldsymbol{a}_l\big)\right) - 2r_{ll}\boldsymbol{a}_l'X_k\boldsymbol{a}_l \tag{18}$$

and

$$e \equiv 2\left(\sum_{p\neq l}^{r} d_{kp}\left(\sum_{q\neq l}^{r}\sum_{s\neq l}^{r} d_{kq}d_{ks}(r_{qs}r_{pl}+r_{sq}r_{lp})\boldsymbol{a}_p'\boldsymbol{a}_q\boldsymbol{a}_s'\boldsymbol{a}_l - \boldsymbol{a}_p'(r_{pl}X_k + r_{lp}X_k')\boldsymbol{a}_l\right)\right). \tag{19}$$

To determine the minimum of the fourth degree polynomial (15) in d_{kl}, we have to solve the normal equation

$$4ad_{kl}^3 + 3bd_{kl}^2 + 2cd_{kl} + e = 0. \tag{20}$$

This equation has three roots in general (of which at least one is real). These can be determined by standard procedures (e.g., Abramowitz and Stegun, 1964, p. 17). The minimum of σ_3 is attained for that root that gives the smallest function value. This procedure for updating d_{kl} can be used to update all elements of D_k, which will decrease the overall function σ_2 monotonically. In this way each matrix D_k can be updated.

Setting up the complete algorithm.

Alternating the above described three procedures for updating A, R, and $D_1,\ldots,D_K$, we can decrease function σ_2 monotonically. This iterative procedure can be considered an ALS algorithm because it alternates between least squares updates for the columns of A, for the matrix R and for the elements of $D_1, \ldots, D_K$. Because σ_2 is bounded from below, this algorithm will converge to a stable function value.

Some of the parameter sets in the algorithm have to be initialized before the iterative procedure can start. It is suggested here to initialize the matrices D_k as $D_k = I_r$, $k = 1, \ldots, K$, and to initialize A as the matrix containing the r dominant eigenvectors of $\Sigma_k(X_k + X_k')$. This intialization will be called the 'default start' henceforth.

The complete algorithm can be summarized as follows:

Step 1.

1a. Initialize A, and $D_1, \ldots, D_K$;

1b. Compute starting values for R according to (12);

1c. Evaluate σ_2 for these starting values.

Step 2.

2a. For $l = 1, \ldots, r$, update column $\boldsymbol{a}_l$ of A by

1. Computing C and $\boldsymbol{z}$ according to (7) and (8), respectively, and computing the eigendecomposition of $S = \frac{1}{2}(C + C') = UDU'$;
2. Minimizing $g(\tilde{\boldsymbol{a}}_l)$ according to the procedure by Ten Berge and Nevels (1977) and deriving the solution of $\boldsymbol{a}_l$ as $\boldsymbol{a}_l = U\tilde{\boldsymbol{a}}_l$;
3. Replace $\boldsymbol{a}_l$ by its update.

2b. For $k = 1, \ldots, K$, $l = 1, \ldots, r$, update d_{kl} by

1. Identifying the coefficients a, b, c and e ((16)–(19));
2. Finding the roots of the normal equation (20);
3. Choosing the root that gives the smallest value of σ_2 as the update for d_{kl};
4. Replace d_{kl} by its update.

2c. Update R according to (12).

2d. Evaluate the function value of σ_2.

If $(\sigma_2^{\text{old}} - \sigma_2^{\text{new}}) > \epsilon\sigma_2^{\text{old}}$ and $\sigma_2^{\text{new}} > \epsilon^* \Sigma_k \text{tr} X_k' X_k$, for small values ϵ and ϵ^* (determined by the user), then repeat Step 2.

An alternating least squares algorithm for PARAFAC2

As has been mentioned above, PARAFAC2 can be seen as a variant of DEDICOM3 in which H is constrained to be symmetric or even p.s.d. If H is merely constrained to be symmetric, then one may use the DEDICOM3 algorithm because if X_k is symmetric, then the solution for R will be symmetric too. This can be proven as follows. It has been shown above that the minimum of σ_2 over R given A and $D_1, \ldots, D_K$ is attained for

$$\text{Vec}(R) = \left(\sum_{k=1}^{K} (D_k A'AD_k) \otimes (D_k A'AD_k) \right)^{-1} \sum_{k=1}^{K} \text{Vec}(D_k A' X_k A D_k). \tag{12}$$

By a slightly different derivation for the (unique) minimizing R, that is after first transposing left and right hand terms in σ_2, we find

$$\operatorname{Vec}(R') = \left(\sum_{k=1}^{K} (D_k A'AD_k) \otimes (D_k A'AD_k) \right)^{-1} \sum_{k=1}^{K} \operatorname{Vec}(D_k A'X_k' AD_k). \tag{21}$$

Clearly, when X_k is symmetric, the right hand sides of (12) and (21) coincide, hence $\operatorname{Vec}(R) = \operatorname{Vec}(R')$, or equivalently, R is symmetric.

Although the texts that introduce PARAFAC2 do not explicitate that H should be p.s.d., both the fact that H is seen as a matrix with cosines of angles between the factors and that it is derived as TT' where T is a certain oblique transformation matrix (Harshman, 1972, p. 36; Carroll and Wish, 1974, p. 95) suggest that H should be p.s.d. For that reason, we have adjusted the DEDICOM3 algorithm such that it handles the constraint that R (denoted as H now) be p.s.d. For that purpose, we only need to develop a procedure for updating H subject to the constraint that H be p.s.d. The procedures for updating A, and $D_1, \ldots, D_K$ do not need to be adjusted, but will be simplified slightly by using the symmetry of R.

To find an update for H subject to the constraint that H be p.s.d. we first expand the function σ_1 as

$$\sigma_1(H) = \sum_{k=1}^{K} \| C_k \|^2 - 2 \operatorname{tr}\left(\sum_{k=1}^{K} D_k A'C_k AD_k \right) H + \sum_{k=1}^{K} \operatorname{tr} D_k A'AD_k HD_k A'AD_k H. \tag{22}$$

This function can be recognized as a member of the general class of matrix functions for which Kiers (1990) has given a minimization procedure. His procedure can be used to minimize

$$f(H) = \operatorname{tr} \tilde{A}H + \sum_{k=1}^{K} \operatorname{tr} \tilde{B}_k H\tilde{C}_k H', \tag{23}$$

for given matrices $\tilde{A}$, $\tilde{B}_k$, $\tilde{C}_k$, $k = 1, \ldots, K$, over H subject to a variety of constraints. Basically, his procedure comes down to an iterative procedure that monotonically decreases $f(H)$ by updating H as the matrix that minimizes $\| F + H^0 - H \|^2$, where

$$F \equiv -\left(2 \sum_{k=1}^{K} \alpha_k \right)^{-1} \left(\tilde{A}' + \sum_{k=1}^{K} \tilde{B}_k H\tilde{C}_k + \sum_{k=1}^{K} \tilde{B}_k' H\tilde{C}_k' \right), \tag{24}$$

with α_k chosen such that α_k is at least as large as the largest eigenvalue of $(\frac{1}{2}\tilde{B}_k \otimes \tilde{C}_k + \frac{1}{2}\tilde{B}_k' \otimes \tilde{C}_k')$, and H^0 denotes the previous value of H (Kiers, 1990, pp. 420–421). For the present case, $\tilde{A} = -2\Sigma_k D_k A'C_k AD_k$, and $\tilde{B}_k = \tilde{C}_k = D_k A'AD_k$, hence α_k can be taken as $\lambda_k^2(D_k A'AD_k)$, where $\lambda_k(\cdot)$ denotes the largest eigenvalue of the matrix between parentheses, and

$$F = \left(\sum_{k=1}^{K} \lambda_k^2(D_k A'AD_k) \right)^{-1} \times \left(\sum_{k=1}^{K} D_k A'C_k AD_k - \sum_{k=1}^{K} D_k A'AD_k H^0 D_k A'AD_k \right). \tag{25}$$

Although Kiers has not worked out the case where H is constrained to be p.s.d., he mentions (p. 428) that constraints other than the ones explicitly worked out can be imposed easily by updating H by the H that minimizes $\| F + H^0 - H \|^2$ subject to the constraint at hand. Hence, in the present case, we have to minimize $\| F + H^0 - H \|^2$ subject to the constraint that H be p.s.d. The solution for this problem has been given by Keller (1962). That is, let $(F + H^0) = K\Lambda K'$ be an eigendecomposition, let q be the number of positive eigenvalues, let Λ_q be the $q \times q$ diagonal matrix containing these positive eigenvalues, and let K_q $(r \times q)$ be the columnwise orthonormal matrix containing the corresponding eigenvectors. Then $\tilde{H} = K_q \Lambda_q K_q'$ minimizes $\| F + H^0 - H \|^2$. Hence by updating H as $\tilde{H}$, the function $f(H)$ and hence the function σ_1 is decreased.

If we combine the above procedure for updating H with the procedures for updating A and $D_1, \ldots, D_K$ in the DEDICOM3 algorithm, we have an algorithm that monotonically decreases the PARAFAC2 loss function σ_1. The procedures for updating A and $D_1, \ldots, D_K$ can be simplified slightly by using the symmetry of X_k (called C_k in PARAFAC2), $k = 1, \ldots, K$, and R (called H in PARAFAC2). That is, for the update of column $\boldsymbol{a}_l$ of A, it remains to minimize function

$$f(\boldsymbol{a}_l) = \boldsymbol{a}_l' S \boldsymbol{a}_l - 2\boldsymbol{z}'\boldsymbol{a}_l, \tag{9}$$

where

$$S = 2 \sum_{k=1}^{K} \sum_{p \neq l}^{r} \sum_{s \neq l}^{r} d_{kp} d_{ks} d_{kl}^2 (h_{ls} h_{pl} + h_{ll} h_{sp}) \boldsymbol{a}_p \boldsymbol{a}_s' - d_{kl}^2 h_{ll} C_k, \tag{26}$$

and

$$\boldsymbol{z} = 2 \sum_{k=1}^{K} \sum_{p \neq l}^{r} d_{kp} d_{kl} \left(h_{pl} C_k \boldsymbol{a}_p - d_{kl}^2 h_{ll} h_{pl} \boldsymbol{a}_p - \sum_{q \neq l}^{r} \sum_{s \neq l}^{r} d_{kq} d_{ks} h_{qs} h_{pl} \boldsymbol{a}_p' \boldsymbol{a}_q \boldsymbol{a}_s \right). \tag{27}$$

The update for the element d_{kl} of D_k, $k = 1, \ldots, K$, $l = 1, \ldots, r$ is again obtained from solving the normal equation (20), but now b, c, and e can be simplified as

$$b = 4 \sum_{p \neq l}^{r} d_{kp} h_{ll} h_{pl} \boldsymbol{a}'_p \boldsymbol{a}_l, \tag{28}$$

$$c = 2\left(\sum_{p \neq l}^{r} \sum_{q \neq l}^{r} d_{kp} d_{kq} \left(h_{pl} h_{ql} \boldsymbol{a}'_p \boldsymbol{a}_q + (h_{lp} h_{ql} + h_{ll} h_{pq}) \boldsymbol{a}'_p \boldsymbol{a}_l \boldsymbol{a}'_q \boldsymbol{a}_l \right) - H_{ll} \boldsymbol{a}'_l C_k \boldsymbol{a}_l \right) \tag{29}$$

and

$$e = 4\left(\sum_{p \neq l}^{r} d_{kp} \left(\sum_{q \neq l}^{r} \sum_{s \neq l}^{r} d_{kq} d_{ks} h_{qs} h_{pl} \boldsymbol{a}'_p \boldsymbol{a}_q \boldsymbol{a}'_s \boldsymbol{a}_l - h_{pl} \boldsymbol{a}'_p C_k \boldsymbol{a}_l \right) \right). \tag{30}$$

The complete PARAFAC2 algorithm consists of the updating of R as described at the beginning of this section (with key formula (25)), and updating A and $D_1, \ldots, D_K$ in essentially the same way as in the DEDICOM3 algorithm, but with the simplifications offered by (26) through (30) incorporated.

Performance of the DEDICOM3 and PARAFAC2 algorithms

The DEDICOM3 algorithm has been programmed (in PASCAL) and subsequently tested on a number of data that were constructed according to model (3), with zero error terms. For these data, we know that a perfect fit can be attained. We constructed 480 data sets by crossing the following characteristics of the data. First of all we chose $m = 6$ for the first 240 sets, and $m = 10$ for the next 240 sets. In both cases we took $k = 3$ for the first 120 sets, and $k = 6$ for the next 120 sets. All data sets were constructed according to (3), half of these were based on dimensionality $r = 2$, half on $r = 3$. Finally, in one third of the cases the matrix R used in the construction of the data was taken completely at random, in one third it was taken at random but symmetric, and in the final third it was constructed as a random p.s.d. matrix. This complicated design was used to ensure that a broad range of possible data sets was covered, and to be able to single out possible irregularities in any of the 'cells' of this design. In some pilot analyses we found that the procedure tended to converge rapidly at the beginning, yielding a function value which is relatively close to zero in a reasonable number of iterations, but then continues very slowly. Because of this, we aborted the iterative process after 100 iterations, even if the convergence criterion ($\epsilon = \epsilon^* = 0.0000001$) was not yet met. Our main interest is in seeing if the DEDICOM3 algorithm approximates the perfect solution closely enough. Therefore, we checked in how many instances the solution gave a fit value (expressed as percentage of explained sum of squares) smaller than 99%. In the first three columns of Table 1, for each cell (pertaining to 20 analyses) the number of cases with a fit less than 99% within 100 iterations is reported (with

Table 1
Number of cases (out of 20) where the fit is less than 99% (and less than 99.9% parenthesized), and average time per iteration

m	k	r	Completely Random R	Random Symmetric R	Random p.s.d. R	Time per Iteration (in sec.)	Random p.s.d. R PARAFAC2
6	3	2	3(4)	0 (4)	0 (0)	0.9	0 (0)
6	6	2	1(3)	1 (6)	0 (2)	1.5	0 (3)
6	3	3	2(9)	1 (8)	0 (2)	2.4	0 (5)
6	6	3	5(6)	6(14)	0(12)	4.2	1(16)
10	3	2	1(3)	1 (1)	0 (0)	2.7	0 (0)
10	6	2	0(0)	3 (7)	0 (2)	4.4	0 (1)
10	3	3	0(6)	3 (8)	0 (2)	6.7	0 (3)
10	6	3	5(9)	3(13)	0(12)	11.5	0(17)

in parentheses the number of cases with a fit less than 99.9%). In only 35 (out of 480) cases the fit was less than 99% and in 32 of these the procedure had not yet converged. So only in the remaining three cases, where the fit was smaller than 99% and the process had converged, we deal with cases where the algorithm converged to a local minimum. In these three cases the fit was 76.2%, 99.4%, and 98.2%, respectively. It seems safe to conclude that when started with our default starting procedure, the algorithm is not likely to converge to a local optimum. To make the chance of hitting local optima even smaller, one should use several other (random) starts in addition to the default one, and rerun the program. In order to get an idea of the time needed for such reanalyses, the fourth column gives the average computation time per iteration, for each of the data sizes. These computation times were found for a pc with 80386 processor (20 MHz) and 80387 coprocessor.

The data constructed with R p.s.d. have also been analyzed by the PARAFAC2 algorithm (that is, DEDICOM3 with R constrained to be p.s.d.). The number of cases with a fit less than 99% within 100 iterations is reported in the fifth column of Table 1 (with the number of cases with a fit less than 99.9% parenthesized). It should be noted that for these data the DEDICOM3 algorithm always gave solutions with R p.s.d. (which comes to no surprise because the perfect solution with R p.s.d. was almost always recovered). The PARAFAC2 algorithm converged only 9 times before the maximum of 100 iterations was reached, whereas the DEDICOM3 algorithm converged 16 times in less than 100 iterations. Apparently, at least for the data at hand the DEDICOM3 algorithm is more efficient, which can be explained by the fact that it updates R by *minimizing* the function over R, whereas the R update in the PARAFAC2 algorithm merely *decreases* the function.

For imperfect data the DEDICOM3 algorithm need not always give solutions with R p.s.d. To examine the behavior of DEDICOM3 in this respect, 160 data sets with random, but p.s.d. C_k, $k = 1, \dots, K$, were made of the same orders as in the previous study. Among the 2- and 3-dimensional solutions R was p.s.d.

throughout. Our conjecture that the p.s.d.-ness of $C_1, \ldots, C_K$ is a sufficient condition for DEDICOM3 to give solutions with R p.s.d. was disproved by a data set consisting of six 3×3 matrices where a 3-dimensional solution was obtained. It seems that DEDICOM3 gives solutions with R indefinite only in cases where the dimensionality is high relative to the size of the data, a situation of limited practical relevance. Based on these results, a useful strategy seems to be to use the DEDICOM3 algorithm for PARAFAC2 as well, and use the PARAFAC2 algorithm only in cases where DEDICOM3 fails to give a proper solution (with R p.s.d.).

From Table 1 it can be observed that the size of the data does not affect the number of inadequate solutions (with fit less than 99%) in a consistent way. The dimensionality of the data (r) does seem to affect the number of inadequate solutions. For $r = 2$ we found 11 and for $r = 3$ we found 25 inadequate solutions ($\chi^2 = 6.93$, $p < 0.01$). As far as the type of the data is concerned, it is interesting to note that the p.s.d. data gave no inadequate solution (out of 160), whereas the other types of data gave 17 and 18 inadequate solutions, respectively.

Apart from these analyses we analyzed three empirical data sets for which it was known that the different slabs had quite similar (two-dimensional) factor solutions. For a set with two 15×15 correlation matrices, the DEDICOM3 algorithm converged (with $\epsilon = \epsilon^* = 0.0000001$) in 5 iterations when the default start (as described in the algorithm section) was taken. The PARAFAC2 algorithm converged in 7 iterations to the same function value, albeit with different values for the parameter matrices. This demonstrates the nonuniqueness of PARAFAC2 solutions for data with only two frontal slices. The same function value was obtained when two sets of different starting values were used by the PARAFAC2 algorithm, but it took more iterations to find the solution; a third start yielded a local minimum.

The second data set consisted of four correlation matrices for 7 variables. We took $r = 2$ in all analysis. When the default start was used, the DEDICOM3 algorithm converged in 64 iterations to a local minimum ($\sigma_2 = 9.59$). With a second start the algorithm yielded $\sigma_2 = 9.21$ (after 183 iterations). Four additional random starts yielded values of $\sigma_2 = 10.01$ (aborted after 100 iterations), $\sigma_2 = 9.34$ (aborted after 100 iterations), $\sigma_2 = 9.21$ (converged in 30 iterations), and $\sigma_2 = 9.22$ (converged in 28 iterations), respectively. In the apparently globally minimal solution R was p.s.d., so the PARAFAC2 algorithm did not seem necessary here. The PARAFAC2 algorithm has yet been used for testing purposes, and we obtained the same local minimum when using the default start (after 63 iterations). Using the other starts again we found $\sigma_1 = 9.21$ (36 iterations), $\sigma_1 = 9.21$ (36 iterations), $\sigma_1 = 9.35$ (88 iterations), and $\sigma_1 = 9.22$ (42 iterations), respectively. For the present data set PARAFAC2 is more efficient than DEDICOM3. It is of interest to mention that the parameter matrices from the two solutions corresponding to $\sigma_1 = 9.21$ were equal up to admissible permutations and sign reflections, thus sustaining the conjecture of uniqueness of the PARAFAC2 solution.

The third data set consisted of six covariance matrices for six variables. The two-dimensional DEDICOM3 solution based on the default start consisted of a matrix R which is p.s.d. Five different random starts gave local minima (all quite close to the solution from the default start). The same six starts were used with the PARAFAC2 algorithm. The default start resulted in the same solution as found with DEDICOM3, both in terms of the function value, and in terms of the parameter matrices (sustaining the uniqueness conjecture again). The other starts produced results that were clearly better than the ones of DEDICOM3. More importantly, three of the five starts resulted in function values which were slightly smaller than the one obtained with the default start. Apparently, here again the PARAFAC2 algorithm outperformed the DEDICOM3 algorithm.

Discussion

The DEDICOM3 algorithm proposed here seems to work reasonably well in practice. The problems of local optima can be dealt with by using several restarts. Attempts have been made to overcome the slow convergence near the solution by using several acceleration procedures (Ramsay, 1975). These did not speed up the process considerably, and in some cases even overshot the solution in such a way that the minimization process had to start all over again, albeit from a different position.

Harshman (1972, p. 38) has mentioned that it seems reasonable to constrain the diagonal matrices $D_1, \ldots, D_K$ to be nonnegative. Such a constraint can be implemented in our procedure as follows. Instead of updating each element of D_k by that root of the normal equation (20) that gives the smallest function value, we update d_{kl} by the *positive* root that gives the smallest function value, or by zero if $d_{kl} = 0$ gives a function value which is yet smaller. This procedure has been programmed as well, and performed well in a few test analyses.

Procedures for assessing the statistical stability of the results, or robustness of the method, for instance with respect to sensitivity to outliers, have not been studied here. For PARAFAC2, one may deal with these matters by jackknife procedures in which observation units are left out one at a time, or bootstrap procedures where samples (with replacement) are drawn from the original sample of observation units. For DEDICOM3, these procedures can also be used if the data in each slab are based on aggregates over observation units. If the data consists of judgements of K observers on n stimuli, then a reasonable way to assess stability of the results seems to perform a jackknife study by leaving out one observer at a time.

Appendix

To derive expression (6), we first isolate all terms in (5) containing $\boldsymbol{a}_l$ (using $\boldsymbol{a}_l'\boldsymbol{a}_l = 1$):

$$\begin{aligned}
\sigma_2(\boldsymbol{a}_l) = {} & \sum_{k=1}^{K} \| X_k \|^2 - 2\sum_{k=1}^{K} \boldsymbol{a}_l' X_k \boldsymbol{a}_l d_{kl}^2 r_{ll} - 2\sum_{k=1}^{K}\sum_{p\neq l}^{r} \boldsymbol{a}_p' X_k \boldsymbol{a}_l d_{kl} r_{pl} d_{kp} \\
& - 2\sum_{k=1}^{K}\sum_{q\neq l}^{r} \boldsymbol{a}_l' X_k \boldsymbol{a}_q d_{kq} r_{lq} d_{kl} + \sum_{k=1}^{K} d_{kl}^4 r_{ll}^2 + \sum_{k=1}^{K}\sum_{p\neq l}^{r} d_{kp} d_{kl}^3 r_{ll} r_{pl} \boldsymbol{a}_p' \boldsymbol{a}_l \\
& + \sum_{k=1}^{K}\sum_{q\neq l}^{r} d_{kq} d_{kl}^3 r_{ll} r_{ql} \boldsymbol{a}_l' \boldsymbol{a}_q \\
& + \sum_{k=1}^{K}\sum_{s\neq l}^{r} d_{ks} d_{kl}^3 r_{ls} r_{ll} \boldsymbol{a}_s' \boldsymbol{a}_l + \sum_{k=1}^{K}\sum_{t\neq l}^{r} d_{kt} d_{kl}^3 r_{lt} r_{ll} \boldsymbol{a}_l' \boldsymbol{a}_t \\
& + \sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{q\neq l}^{r} d_{kp} d_{kq} d_{kl}^2 r_{ql} r_{pl} \boldsymbol{a}_p' \boldsymbol{a}_q + \sum_{k=1}^{K}\sum_{s\neq l}^{r}\sum_{t\neq l}^{r} d_{ks} d_{kt} d_{kl}^2 r_{ls} r_{lt} \boldsymbol{a}_s' \boldsymbol{a}_t \\
& + \sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{s\neq l}^{r} d_{kp} d_{ks} d_{kl}^2 r_{ls} r_{pl} \boldsymbol{a}_p' \boldsymbol{a}_l \boldsymbol{a}_s' \boldsymbol{a}_l \\
& + \sum_{k=1}^{K}\sum_{q\neq l}^{r}\sum_{t\neq l}^{r} d_{kq} d_{kt} d_{kl}^2 r_{lt} r_{ql} \boldsymbol{a}_l' \boldsymbol{a}_q \boldsymbol{a}_l' \boldsymbol{a}_t \\
& + \sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{t\neq l}^{r} d_{kp} d_{kt} d_{kl}^2 r_{ll} r_{pt} \boldsymbol{a}_p' \boldsymbol{a}_l \boldsymbol{a}_l' \boldsymbol{a}_t \\
& + \sum_{k=1}^{K}\sum_{q\neq l}^{r}\sum_{s\neq l}^{r} d_{kq} d_{ks} d_{kl}^2 r_{ll} r_{qs} \boldsymbol{a}_l' \boldsymbol{a}_q \boldsymbol{a}_s' \boldsymbol{a}_l \\
& + \sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{q\neq l}^{r}\sum_{s\neq l}^{r} d_{kp} d_{kq} d_{ks} d_{kl} r_{qs} r_{pl} \boldsymbol{a}_p' \boldsymbol{a}_q \boldsymbol{a}_s' \boldsymbol{a}_l \\
& + \sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{q\neq l}^{r}\sum_{t\neq l}^{r} d_{kp} d_{kq} d_{kt} d_{kl} r_{ql} r_{pt} \boldsymbol{a}_p' \boldsymbol{a}_q \boldsymbol{a}_l' \boldsymbol{a}_t \\
& + \sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{s\neq l}^{r}\sum_{t\neq l}^{r} d_{kp} d_{ks} d_{kt} d_{kl} r_{ls} r_{pt} \boldsymbol{a}_p' \boldsymbol{a}_l \boldsymbol{a}_s' \boldsymbol{a}_t \\
& + \sum_{k=1}^{K}\sum_{q\neq l}^{r}\sum_{s\neq l}^{r}\sum_{t\neq l}^{r} d_{kq} d_{ks} d_{kt} d_{kl} r_{qs} r_{lt} \boldsymbol{a}_l' \boldsymbol{a}_q \boldsymbol{a}_s' \boldsymbol{a}_t \\
& + \sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{q\neq l}^{r}\sum_{s\neq l}^{r}\sum_{t\neq l}^{r} \boldsymbol{a}_p' \boldsymbol{a}_q d_{kq} r_{qs} d_{ks} \boldsymbol{a}_s' \boldsymbol{a}_t d_{kt} r_{pt} d_{kp}.
\end{aligned} \tag{31}$$

By writing c for the sum of terms independent of $\boldsymbol{a}_l$, sorting and reindexing elements, and subsequently collecting terms, we can simplify (31) as

$$\begin{aligned}
\sigma_2(\boldsymbol{a}_l) &= c - 2\sum_{k=1}^{K} \boldsymbol{a}_l'(d_{kl}^2 r_{ll} X_k)\boldsymbol{a}_l - 2\sum_{k=1}^{K}\sum_{p\neq l}^{r} d_{kl}d_{kp}r_{pl}\boldsymbol{a}_p' X_k \boldsymbol{a}_l \\
&\quad - 2\sum_{k=1}^{K}\sum_{p\neq l}^{r} d_{kl}d_{kp}r_{lp}\boldsymbol{a}_p' X_k' \boldsymbol{a}_l + 2\sum_{k=1}^{K}\sum_{p\neq l}^{r} d_{kp}d_{kl}^3 r_{ll}r_{pl}\boldsymbol{a}_p'\boldsymbol{a}_l \\
&\quad + 2\sum_{k=1}^{K}\sum_{p\neq l}^{r} d_{kp}d_{kl}^3 r_{lp}r_{ll}\boldsymbol{a}_p'\boldsymbol{a}_l + 2\boldsymbol{a}_l'\sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{s\neq l}^{r} d_{kp}d_{ks}d_{kl}^2 r_{ls}r_{pl}\boldsymbol{a}_p\boldsymbol{a}_s'\boldsymbol{a}_l \\
&\quad + 2\boldsymbol{a}_l'\sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{s\neq l}^{r} d_{kp}d_{ks}d_{kl}^2 r_{ll}r_{sp}\boldsymbol{a}_p\boldsymbol{a}_s'\boldsymbol{a}_l \\
&\quad + 2\sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{q\neq l}^{r}\sum_{s\neq l}^{r} d_{kp}d_{kq}d_{ks}d_{kl}r_{qs}r_{pl}\boldsymbol{a}_p'\boldsymbol{a}_q\boldsymbol{a}_s'\boldsymbol{a}_l \\
&\quad + 2\sum_{k=1}^{K}\sum_{p\neq l}^{r}\sum_{q\neq l}^{r}\sum_{s\neq l}^{r} d_{kp}d_{kq}d_{ks}d_{kl}r_{sq}r_{lp}\boldsymbol{a}_p'\boldsymbol{a}_q\boldsymbol{a}_s'\boldsymbol{a}_l \\
&= c + 2\boldsymbol{a}_l'\sum_{k=1}^{K}\left(\sum_{p\neq l}^{r}\sum_{s\neq l}^{r} d_{kp}d_{ks}d_{kl}^2(r_{ls}r_{pl}+r_{ll}r_{sp})\boldsymbol{a}_p\boldsymbol{a}_s' - d_{kl}^2 r_{ll}X_k\right)\boldsymbol{a}_l \\
&\quad - 2\sum_{k=1}^{K}\sum_{p\neq l}^{r} d_{kp}d_{kl}\Bigg(\boldsymbol{a}_p'(r_{pl}X_k + r_{lp}X_k') - d_{kl}^2 r_{ll}(r_{pl}+r_{lp})\boldsymbol{a}_p' \\
&\qquad\qquad - \sum_{q\neq l}^{r}\sum_{s\neq l}^{r} d_{kq}d_{ks}(r_{qs}r_{pl}+r_{sq}r_{lp})\boldsymbol{a}_p'\boldsymbol{a}_q\boldsymbol{a}_s'\Bigg)\boldsymbol{a}_l,
\end{aligned} \tag{32}$$

which is (6) with (7) and (8) substituted for C and z, respectively.

Acknowledgement

This research has been made possible by a fellowshicp from the Royal Netherlands Academy of Arts and Sciences. The author is obliged to Jos ten Berge and Richard Harshman for stimulating comments.

References

Abramowitz, M. and I.A. Stegun (1964). *Handbook of Mathematical Functions*. Dover, New York.

Carroll, J.D. and M. Wish (1974). Models and methods for three-way multidimensional scaling. In: D.H. Krantz et al. (Eds.) *Contemporary Developments in Mathematical Psychology, Vol. II:*

Measurement, Psychophysics, and Neural Information Processing (pp. 57–105). Freeman and Co., San Francisco.

Dunn, T.R. and R.A. Harshman (1982). A multidimensional scaling model for the size-weight illusion. *Psychometrika*, **47**, 25–45.

Harshman, R.A. (1970). Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-mode factor analysis. *UCLA Working Papers in Phonetics*, **16**, 1–84.

Harshman, R.A. (1972). PARAFAC2: Mathematical and technical notes. *UCLA Working Papers in Phonetics*, **22**, 33–44.

Harshman, R.A. (1978). Models for analysis of asymmetrical relationships among N objects or stimuli. Paper presented at the First Joint Meeting of the Psychometric Society and the Society for Mathematical Psychology, Hamilton, Ontario, August.

Harshman, R.A. and M.E. Lundy (1984). The PARAFAC model for three-way factor analysis and multidimensional scaling. In: H.G. Law, C.W. Snyder, J.A. Hattie and R.P. McDonald (Eds.), *Research Methods for Multimode Data Analysis*. Praeger, New York. (pp. 122–215).

Harshman, R.A. and M.E. Lundy (In preparation). Uniqueness of proportional profile models. Manuscript.

Harshman, R.A., P.E. Green, Y. Wind and M.E. Lundy (1982). A model for the analysis of asymmetric data in marketing research, *Marketing Science*, **1**, 205–242.

Henderson, H.V. and S.R. Searle (1981). The vec-permutation matrix, the vec operator and Kronecker products: A review. *Linear and multilinear algebra*, **9**, 271–288.

Keller J.B., (1962). Factorization of matrices by least squares. *Biometrika*, **49**, 239–242.

Kiers, H.A.L. (1989). An alternating least squares algorithm for fitting the two- and three-way DEDICOM model and the IDIOSCAL model. *Psychometrika*, **54**, 515–521.

Kiers, H.A.L. (1990). Majorization as a tool for optimizing a class of matrix functions, *Psychometrika*, **55**, 417–428.

Kiers, H.A.L. and J.M.F. Ten Berge (1989). Alternating least squares algorithms for simultaneous components analysis with equal component weight matrices for all populations. *Psychometrika*, **54**, 467–473.

Kroonenberg, P.M. (1983). *Three-mode Principal Component Analysis*. DSWO Press, Leiden.

Ramsay, J.O. (1975). Solving implicit equations in psychometric data analysis. *Psychometrika*, **40**, 337–360.

Ten Berge, J.M.F. and K. Nevels (1977). A general solution to Mosier's oblique Procrustes problem. *Psychometrika*, **42**, 593–600.