Journal of Classification 29:277-296 (2012)
DOI: 10.1007/s00357-012-9113-4

Block-Relaxation Approaches for Fitting the INDCLUS Model

Tom F. Wilderjans
KU Leuven, Belgium

Dirk Depril
suAzio Consulting, Belgium

Iven Van Mechelen
KU Leuven, Belgium

Abstract: A well-known clustering model to represent $I \times I \times J$ data blocks, the J frontal slices of which consist of $I \times I$ object by object similarity matrices, is the INDCLUS model. This model implies a grouping of the I objects into a prespecified number of overlapping clusters, with each cluster having a slice-specific positive weight. An INDCLUS model is fitted to a given data set by means of minimizing a least squares loss function. The minimization of this loss function has appeared to be a difficult problem for which several algorithmic strategies have been proposed. At present, the best available option seems to be the SYMPRES algorithm, which minimizes the loss function by means of a block-relaxation algorithm. Yet, SYMPRES is conjectured to suffer from a severe local optima problem. As a way out, based on theoretical results with respect to optimally designing block-relaxation algorithms, five alternative block-relaxation algorithms are proposed. In a simulation study it appears that the alternative algorithms with overlapping parameter subsets perform best and clearly outperform SYMPRES in terms of optimization performance and cluster recovery.

Keywords: INDCLUS; ADCLUS; Alternating least squares; Block-Relaxation algorithms; Additive clustering; Overlapping clusters; Proximity data; Three-Way data.

The research in this paper was partially supported by the Research Fund of KU Leuven (PDM-kort project 3H100377, dr. Tom F. Wilderjans; GOA $2005/04$, Prof. dr. Iven Van Mechelen), by the Belgian Science Policy (IAP P$6/03$, Prof. dr. Iven Van Mechelen), and by the Fund of Scientific Research (FWO)-Flanders (project G.0546.09, Prof. dr. I. Van Mechelen). The simulation study was conducted using high performance computational resources provided by KU Leuven (http://ludit.kuleuven.be/hpc). We would like to thank Prof. dr. J. de Leeuw for his helpful advice on the topic of block-relaxation algorithms. We further also would like to thank three anonymous reviewers for their useful comments and suggestions which considerably improved earlier versions of this manuscript.

Corresponding Author's Address: Tom F. Wilderjans, Faculty of Psychology and Educational Sciences, KU Leuven, Andreas Vesaliusstraat 2, Box 3762, B-3000 Leuven, Belgium, Tel: +32.16.32.61.23, Fax: +32.16.32.62.00, e-mail: tom.wilderjans@ppw.kuleuven.be

Published online 17 August 2012

1. Introduction

In statistical practice, one is often interested in finding structures within a given set of I objects, based on certain object characteristics. One such a characteristic is the resemblance of each object with all other objects.[1] Possible representation of object resemblance data include feature models (Tversky 1977), with features corresponding to groupings of the objects. An important special case of such models is the ADCLUS model as proposed by Shepard and Arabie (1979). In this model the similarity between two objects equals the sum of the weights of all common features of those objects.

For the case of a three-way two-mode $I \times I \times J$ data array $\underline{\mathbf{S}}$, the J frontal slices $\mathbf{S}_j$ of which each constitute an $I \times I$ object by object proximity matrix as stemming from each of J sources, Carroll and Arabie (1983) proposed the INDCLUS model as an extension of the ADCLUS model. In the INDCLUS model, the $I \times I \times J$ data array $\underline{\mathbf{S}}$ is approximated by an $I \times I \times J$ model array $\underline{\mathbf{M}}$, which consists of J matrices $\mathbf{M}_j$ ($j = 1, \ldots, J$) of size $I \times I$. Each matrix $\mathbf{M}_j$ (with entries $m^j_{i_1 i_2}$) approximates the corresponding $\mathbf{S}_j$ (with entries $s^j_{i_1 i_2}$) and can be represented by an ADCLUS model (Shepard and Arabie 1979) that includes a binary (0/1) $I \times K$ matrix $\mathbf{A}$, a non-negative real-valued $J \times K$ matrix $\mathbf{W}$ and real-valued constants c_j ($j = 1, \ldots, J$):

$$\mathbf{M}_j = \mathbf{A} diag(\mathbf{w}_{j.}) \mathbf{A}' + c_j \mathbf{1}\mathbf{1}', \tag{1}$$

with K denoting the smallest number for which such a decomposition is possible, $\mathbf{1}$ a column vector of length I with all entries equal to 1, and $\mathbf{w}_{j.}$ the j-th row of $\mathbf{W}$. The columns of the matrix $\mathbf{A}$ define K (possibly overlapping) clusters or features; hence, $\mathbf{A}$ is called a *cluster membership matrix*, with entries a_{ik} denoting whether object i belongs to cluster k ($a_{ik} = 1$) or not ($a_{ik} = 0$). The rows $\mathbf{w}_{j.}$ of $\mathbf{W}$ contain the *weights* of the features for source j. The number c_j is an overall constant for slice j and can take any real value. Equation (1) then implies that, for source j, the similarity between two objects is the sum of the weights of the clusters those objects jointly belong to plus the constant c_j, that is:

$$m^j_{i_1 i_2} = \sum_{k=1}^{K} a_{i_1 k} w_{jk} a_{i_2 k} + c_j. \tag{2}$$

In the associated data analysis, an INDCLUS model with a prespecified number of clusters K is fitted to a given data set by minimizing a least

1. Another characteristic may be the score of each object on a set of variables. For the case of object by variable data, Mirkin (1987, 1990) developed a model to find (overlapping) groups of objects based on their scores on several variables.

squares loss function. This means that, given a data array, and given a prespecified number of clusters K, an $I \times I \times J$ model array $\underline{\mathbf{M}}$ is looked for such that

$$\begin{aligned} L^2 &= \sum_{j=1}^{J} \|\mathbf{S}_j - \mathbf{A}diag(\mathbf{w}_{j.})\mathbf{A}' - c_j\mathbf{1}\mathbf{1}'\|_F^2 \\ &= \sum_{j=1}^{J} \sum_{i_1,i_2=1}^{I} (s_{i_1 i_2}^{j} - \sum_{k=1}^{K} a_{i_1 k} w_{jk} a_{i_2 k} - c_j)^2, \end{aligned} \tag{3}$$

is minimized over $\mathbf{A}$, $\mathbf{W}$, and c_j ($j = 1, \ldots, J$), subject to $\mathbf{A}$ being binary, $\mathbf{W}$ being a non-negative real-valued matrix, and c_j being real-valued. Note that $\|.\|_F$ denotes the Frobenius norm of a matrix, that is, the square root of the sum of its squared entries.

Unfortunately, no closed form expression for the global minimum of loss function (3) is available, and therefore to find this global optimum one must rely on heuristic procedures. In the past, several proposals for such procedures have been launched in the literature. However, all of these have been subjected to some criticism.

The first two proposals of heuristic procedures were the original ADCLUS algorithm (Shepard and Arabie 1979) and the MAPCLUS algorithm (Arabie and Carroll 1980; Carroll and Arabie 1983).[2] Both, however, appeared to be very computationally intensive, which makes their practical use somewhat prohibitive. As a way out, Chaturvedi and Carroll (1994) proposed a fast and convenient alternating least squares (ALS) algorithm called SINDCLUS. This algorithm fits a slightly altered model, namely,

$$\mathbf{M}_j = \mathbf{A}diag(\mathbf{w}_{j.})\tilde{\mathbf{A}}' + c_j\mathbf{1}\mathbf{1}', \tag{4}$$

with $\tilde{\mathbf{A}}$ denoting an $I \times K$ binary matrix that can be different from $\mathbf{A}$. Subsequently, in the data analysis, the loss function

$$\tilde{L}^2 = \sum_{j=1}^{J} \|\mathbf{S}_j - \mathbf{A}diag(\mathbf{w}_{j.})\tilde{\mathbf{A}}' - c_j\mathbf{1}\mathbf{1}'\|_F^2 \tag{5}$$

is minimized. Chaturvedi and Carroll (1994) conjectured that the global optimum of loss function (5) would always be an INDCLUS model (i.e., $\mathbf{A} = \tilde{\mathbf{A}}$). ten Berge and Kiers (2005) showed that there is at least one case in which this conjecture is true: when $K = 1$, $c_j = 0$ and the matrices $\mathbf{S}_j$ ($j = 1, \ldots, J$) are positive definite; yet, Kiers (1997) and ten Berge and Kiers (2005) further also showed by means of counterexamples that

2. Another heuristic procedure that has been proposed is the SEFIT method (Mirkin 1987, 1990, 1996).

the conjecture is in general not always true. As a way out, Kiers (1997) proposed a new ALS algorithm, called SYMPRES, which fits the INDCLUS model in a computationally fast way by means of alternatingly updating the clusters one by one. At present, it is the best available option to minimize the INDCLUS loss function (3). Unfortunately, ten Berge and Kiers (2005) also found that SYMPRES suffers from a local optima problem.

In the present paper, we will propose five alternative novel ALS strategies to minimize the least squares loss function (3), based on theoretical results regarding optimally designing block-relaxation algorithms. To evaluate the performance of these alternative algorithms, we will conduct a simulation study in which the alternative algorithms will be compared to each other and to the existing SYMPRES algorithm.

The remainder of this paper is structured as follows. In Section 2, we will recapitulate the SYMPRES algorithm and discuss its drawbacks. In Section 3, we will propose the alternative algorithmic strategies to minimize loss function (3). In Section 4, a simulation study will be presented in which the alternative algorithms and SYMPRES will be evaluated. Concluding remarks will be given in Section 5.

2. Fitting the INDCLUS Model by the SYMPRES Algorithm

2.1 Alternating Least Squares Algorithms

SYMPRES, as proposed by Kiers (1997) is an alternating least squares (ALS) type of algorithm. In ALS algorithms, which belong to the broader class of block-relaxation algorithms (de Leeuw 1994), a complex optimization problem is solved by alternatingly tackling a series of subproblems which, considered separately, are easy to handle. To this end, the model parameters are grouped together (e.g., partitioned) into a number of subsets, in such a way that for each subset, conditional on the parameters not belonging to this subset, optimal parameter values can be determined easily (e.g., in terms of closed-form expressions). In the algorithm, after determining initial estimates for all parameters, the parameter subsets are then updated (in a given order) while keeping the other parameters fixed; this procedure is continued until there is no further improvement in the loss function value. Algorithms like SYMPRES differ from each other in three aspects, which pertain to three nontrivial choices that need to be made when constructing any ALS algorithm: (1) the way in which initial values for the parameters are obtained, (2) the way in which the parameters are grouped together in subsets, and (3) the order in which the parameter subsets are updated (conditionally upon the parameters not belonging to the set in question).

2.2 The SYMPRES Algorithm

In the SYMPRES algorithm, as described and applied by ten Berge and Kiers (2005), the loss function (3) is minimized as follows:

1. Obtain initial estimates:
   (a) Randomly generate an initial solution for $\mathbf{A}$ by independently drawing entries a_{ik} from a Bernoulli distribution with parameter $\pi = .5$.
   (b) Randomly generate an initial solution for $\mathbf{W}$ by means of independently drawing entries w_{jk} uniformly from the interval $[0, 1]$.
   (c) Calculate the constants c_j ($j = 1, \ldots, J$) as the means of the entries of $\mathbf{S}_j - \mathbf{A}diag(\mathbf{w}_{j.})\mathbf{A}'$, and calculate the loss value (3) for the obtained initial solution.
2. Update the clusters one by one, that is, for $k = 1, \ldots, K$, do the following:
   (a) For $j = 1, \ldots, J$, calculate $\mathbf{S}_{j(-k)} = \mathbf{S}_j - \mathbf{a}_{.1}w_{j1}\mathbf{a}'_{.1} - \cdots - \mathbf{a}_{.(k-1)}w_{j(k-1)}\mathbf{a}'_{.(k-1)} - \mathbf{a}_{.(k+1)}w_{j(k+1)}\mathbf{a}'_{.(k+1)} - \cdots - \mathbf{a}_{.K}w_{jK}\mathbf{a}'_{.K} - c_j\mathbf{1}\mathbf{1}'$, with $\mathbf{a}_{.k}$ denoting the k-th column of $\mathbf{A}$. To obtain the residual data $\mathbf{S}_{j(-k)}$, the similarities based on all but the k-th cluster are subtracted from the data. The loss function (3) then reduces to
   $$\sum_{j=1}^{J} \|\mathbf{S}_{j(-k)} - w_{jk}\mathbf{a}_{.k}\mathbf{a}'_{.k}\|_F^2. \tag{6}$$
   (b) Calculate w_{jk} ($j = 1, \ldots, J$) by means of a simple regression using each of the J constituents of (6). Should one of the resulting values be negative, then put it equal to zero. Plug in the new estimate into (6).
   (c) Update column $\mathbf{a}_{.k}$ element-wise: For $i = 1, \ldots, I$ evaluate the loss value for $a_{ik} = 1$ and $a_{ik} = 0$ (while keeping all other elements of $\mathbf{a}_{.k}$ fixed) and plug in (after evaluating a single a_{ik}) the value for a_{ik} that yields the lowest loss.
3. Update the additive constants c_j ($j = 1, \ldots, J$) as in step 1c.
4. Evaluate the loss value. If it has decreased return to step 2, otherwise the algorithm stops.

In order to avoid the (many) locally optimal solutions for the loss function (3), a multistart procedure has been implemented that consists of repeating steps 1 to 4 a prespecified number of times and subsequently retaining the result that yields the lowest loss function value as the final estimate.

It appears that for the SYMPRES algorithm the following choices have been made: First, initial parameter values are obtained by randomly generating values for the membership entries and the weights, and by computing subsequently conditionally optimal associated constants. Second, the set of parameters (i.e., the membership parameters a_{ik}, the weights w_{jk}, and the constants c_j) has been partitioned into the following subsets:

1. $I \times K$ subsets (singletons) of individual membership parameters a_{ik} (i.e., the $I \times K$ elements of $\mathbf{A}$);
2. K subsets of weights $\mathbf{w}_{.k}$ (i.e., [w_{1k}, ..., w_{Jk}], $k = 1, \ldots, K$), with each subset of weights being associated with one cluster;
3. a subset containing the constants $\mathbf{c}$ (i.e., [c_1, ..., c_J]).

Third, the parameter subsets are updated in the following order: [$\mathbf{w}_{.1}$, a_{11}, ..., a_{I1}, $\mathbf{w}_{.2}$, a_{12}, ..., a_{I2}, ..., $\mathbf{w}_{.K}$, a_{1K}, ..., a_{IK}, $\mathbf{c}$]. Note that the membership entries are updated cluster by cluster (i.e, column by column in the membership matrix $\mathbf{A}$) in an element-wise fashion (i.e., each entry separately).

2.3 Drawbacks of the SYMPRES Algorithm

In the construction of block-relaxation algorithms (e.g., alternating least squares algorithms) one may wish to take into account the following recommendations to enhance optimization performance:

1. With regard to the starting procedure, ideally initial parameter estimates are generated that are as close as possible to the optimal ones, resulting in a faster convergence of the algorithm. This ideal can be approached by determining the initial estimates in a (more or less) rational or optimal way (Milligan 1980; Schepers, Van Mechelen, and Ceulemans 2006; Ceulemans, Van Mechelen, and Leenen 2007).
2. It is well-known that dependencies between parameters may interfere with algorithmic performance (Harshman and Lundy 1984; Gelman, Carlin, Stern and Rubin 1995). This effect has been made especially clear for the convergence rate of algorithms (Goodman and Sokal 1989; Smith and Roberts 1993). In particular, for block-relaxation algorithms, de Leeuw (1994) has demonstrated that slower convergence may be expected when high dependencies exist between parameters that belong to different subsets. Based on this work and extrapolating it to other aspects of algorithmic performance, it may be recommended to group together the parameters in such a way that parameters that belong to the same subset are allowed to be

highly dependent, while parameters that belong to different subsets are as independent as possible.[3]

3. With respect to the updating order, in cases in which it is not possible to group together all dependent parameters, it may be advised, when a set of parameters has been updated, to re-estimate as fast as possible the other sets of parameters that are highly dependent on the first. Otherwise, the algorithm may be directed to and get stuck in a suboptimal solution. Note that a similar recommendation applies to *K*-means analysis.[4]

Although the recommendations as outlined above originate from other contexts, we conjecture that they are also relevant for constructing suitable algorithms for fitting INDCLUS and other models that imply mixed discrete-integer optimization problems. When applying the recommendations outlined above to the INDCLUS case, three drawbacks of the SYMPRES algorithm may be identified (which might also explain its local minima problem as demonstrated by ten Berge and Kiers 2005):

A first drawback is that the initial values for the weights and the constants almost never will be conditionally optimal given the (randomly generated) initial membership entries. As a consequence, the SYMPRES starting values may fairly strongly deviate from the (globally) optimal solution.

A second drawback is that SYMPRES in four respects does not fully take into account the strong dependencies between the INDCLUS parameters that may be expected from model equation (1): (a) The mutual dependencies among the weights are neglected to a large extent, because only weights that are associated to the same cluster are grouped together in a subset. (b) The dependencies between the weights and the constants are neglected totally, because the constants are considered as a separate subset. (c) The dependencies among the binary membership entries are totally neglected, because each membership parameter is dealt with as a separate subset that is updated separately. (d) The dependencies between membership entries, on the one hand, and weights and constants, on the other hand, are totally neglected, because no single subset contains a combination of membership entries, weights and constants.

A third drawback is that, when updating an object cluster, not all weights (and constants) that depend on the membership entries of that clus-

3. Note that a similar recommendation has been given for algorithms based on Gibbs sampling (see, e.g., Gelman, Carlin, Stern, and Rubin 1995; Roberts and Sahu 1997; Smith and Roberts 1993; Seewald 1992).

4. In *K*-means the (continuous) centroids are highly dependent on the (binary) clustering of the objects; as a consequence, re-estimating the centroids immediately after re-allocating an object to a cluster (i.e., after changing a binary membership parameter) yields a better optimization performance than updating the centroids only after re-allocating all objects (see, e.g., Dhillon, Guan, and Kogan 2002; Larsen and Aone 1999; Steinbach, Karypis, and Kumar 2000).

ter are re-estimated as fast as possible; in SYMPRES, indeed, only the weights associated with the cluster in question are re-estimated, while the weights associated with other clusters (and the constants) are only updated much later.

3. Five Alternative ALS Approaches

Based on the drawbacks of the SYMPRES algorithm that have been identified in the previous section, different modifications for the SYMPRES algorithm may be proposed. Through (combinations of) these modifications five alternative ALS algorithms for fitting the INDCLUS model may be obtained. Below we will first discuss the different modifications. Next, the alternative ALS algorithms will be outlined.

3.1 Proposed Modifications for the SYMPRES Algorithm

Four modifications for the SYMPRES algorithm will be proposed. The first modification is based on the first drawback of SYMPRES, while the other three modifications try to account for the second and third drawback.

A first modification consists of obtaining (better) initial values for the weights and constants by estimating them optimally, conditional on the initial membership parameters.

A second modification aims at taking into account the mutual dependencies among the weights and the dependencies between the weights and the constants. For this purpose, one may group all weights and constants in a single subset, implying a considerable decrease in the number of subsets.

A third modification consists of better accounting for the mutual dependencies among membership entries, by updating these entries row-wise (i.e., each row of $\mathbf{A}$ is re-estimated in turn); again, this implies a considerable decrease in the number of subsets. Note that Depril, Van Mechelen, and Mirkin (2008), in the context of a one-mode additive clustering model, already successfully used an ALS procedure with row-wise updating of the membership entries.

A fourth modification pertains to taking care of the dependencies between membership entries, on the one hand, and weights and constants, on the other hand. It may be achieved by taking into account the consequences for the weights and constants (and the loss function value) during any re-estimation of the memberships. Therefore, with respect to the grouping of the parameters, subsets are created that consist of a number of membership entries together with all the weights and the constants. Consequently, the number of subsets decreases, while the subsets are now allowed to overlap (i.e., the subsets in question have the weights and constants in common).

Note that taking into account weights and constants during the updating of memberships is equivalent to optimizing an alternative loss function with the membership entries as only arguments (which can be obtained by substituting in loss function (3) the weights and constants by their conditionally optimal value, given the membership entries). Optimization of this alternative loss function implies that, when re-estimating the memberships, the weights and constants are updated implicitly.

To achieve the different modifications outlined above, one may rely on the following algorithmic adaptations:

- The first and second modification necessitate an estimation of the optimal weights and constants conditionally upon the membership entries. For this purpose, loss function (3) may be rewritten as

$$L^2 = \sum_{j=1}^{J} \|\mathbf{S}_j - \sum_{k=1}^{K} w_{jk}\mathbf{a}_{.k}\mathbf{a}'_{.k} - c_j\mathbf{1}\mathbf{1}'\|_F^2. \tag{7}$$

  Hence, given initial estimates for the memberships, conditionally optimal weights w_{jk} and constants c_j can be computed by solving a set of J regression problems with $\mathbf{a}_{.k}\mathbf{a}'_{.k}$ acting as predictor variables and with non-negativity constraints on the regression weights w_{jk}. Ignoring for a moment the constants c_j, the minimization of (7) over all positive w_{jk} implies a set of J independent non-negative least squares (NNLS) problems. These problems can be solved with the NNLS algorithm of Lawson and Hanson (1974), for which a fast computational implementation, called FAST-NNLS, has been provided by Bro and de Jong (1997). Including the simultaneous estimation of the unconstrained constants c_j in FAST-NNLS, can be done easily.
- A row-wise (re-)estimation of the memberships as implied by the third modification may be performed in an enumerative way because for each row only 2^K row patterns need to be evaluated.
- The fourth modification involves the estimation of parameter subsets that consist of a mix of binary membership entries and continuous weights and constants. This mixed problem may be solved by computing for each possible pattern of membership parameters the optimal weights and constants, and by subsequently selecting the pattern that yields the optimal value for the loss function.

3.2 Overview of the Alternative ALS Algorithms

Five alternative ALS algorithms to estimate the INDCLUS model may be considered by adding in a stepwise manner to the SYMPRES algorithm the four modifications that have been proposed in the previous section.

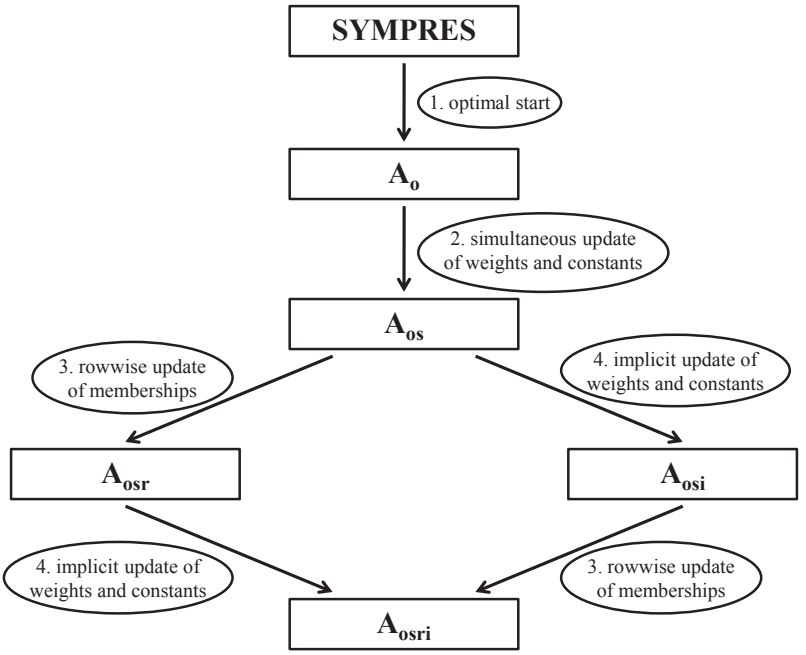


Figure 1. Overview of the alternative algorithms. Within the ellipses the modification is represented that has been added in each step.

Figure 1 presents an overview of these alternative algorithms with within an ellipse an indication of the modification (see Section 3.1) that has been added in each step, and the name of each algorithm containing the modifications that have been added to the original SYMPRES algorithm (with o = optimal start, s = simultaneous update of weights and constants, r = rowwise update of memberships, and i = implicit update of weights and constants). Table 1 further presents an overview of the starting procedure, the parameter subsets, and the order in which the subsets are being updated (within one iteration) for each of the algorithms. It should be noted that the five proposed algorithms are only a subset of all possible algorithms that may be developed by combining SYMPRES with the four modifications (i.e., there is, for instance, no algorithm $\mathbf{A_{sr}}$ which -only- combines the second and the third modification).

4. Simulation Study

4.1 Introduction

In this section, we will present a simulation study to evaluate the performance of SYMPRES and the five alternative algorithms proposed in the

Table 1. Starting procedure, grouping of the parameters in subsets, and updating order (for one iteration) for SYMPRES and the alternative ALS algorithms.

algorithm	starting procedure	grouping of the parameters	updating order
SYMPRES	random $\mathbf{A}$, $\mathbf{W}$ rational $\mathbf{c}$	a_{ik} $(i = 1, \ldots, I; k = 1, \ldots, K)$ $\mathbf{w}_{.k}$ $(k = 1, \ldots, K)$ $\mathbf{c}$	$\mathbf{w}_{.1}$ $a_{11}, \ldots, a_{I1}$ $\vdots$ $\mathbf{w}_{.K}$ $a_{1K}, \ldots, a_{IK}$ $\mathbf{c}$
$\mathbf{A_o}$	random $\mathbf{A}$ optimal $\mathbf{W}$, $\mathbf{c}$	a_{ik} $(i = 1, \ldots, I; k = 1, \ldots, K)$ $\mathbf{w}_{.k}$ $(k = 1, \ldots, K)$ $\mathbf{c}$	$\mathbf{w}_{.1}$ $a_{11}, \ldots, a_{I1}$ $\vdots$ $\mathbf{w}_{.K}$ $a_{1K}, \ldots, a_{IK}$ $\mathbf{c}$
$\mathbf{A_{os}}$	random $\mathbf{A}$ optimal $\mathbf{W}$, $\mathbf{c}$	a_{ik} $(i = 1, \ldots, I; k = 1, \ldots, K)$ $(\mathbf{W}, \mathbf{c})$	$a_{11}, \ldots, a_{I1}$ $(\mathbf{W}, \mathbf{c})$ $\vdots$ $a_{1K}, \ldots, a_{IK}$ $(\mathbf{W}, \mathbf{c})$
$\mathbf{A_{osr}}$	random $\mathbf{A}$ optimal $\mathbf{W}$, $\mathbf{c}$	$\mathbf{a}_{i.}$ $(i = 1, \ldots, I)$ $(\mathbf{W}, \mathbf{c})$	$\mathbf{a}_{1.}$ $(\mathbf{W}, \mathbf{c})$ $\vdots$ $\mathbf{a}_{I.}$ $(\mathbf{W}, \mathbf{c})$
$\mathbf{A_{osi}}$	random $\mathbf{A}$ optimal $\mathbf{W}$, $\mathbf{c}$	$(a_{ik}, \mathbf{W}, \mathbf{c})$ $(i = 1, \ldots, I; k = 1, \ldots, K)$	$(a_{11}, \mathbf{W}, \mathbf{c})$ $\vdots$ $(a_{I1}, \mathbf{W}, \mathbf{c})$ $\vdots$ $(a_{IK}, \mathbf{W}, \mathbf{c})$
$\mathbf{A_{osri}}$	random $\mathbf{A}$ optimal $\mathbf{W}$, $\mathbf{c}$	$(\mathbf{a}_{i.}, \mathbf{W}, \mathbf{c})$ $(i = 1, \ldots, I)$	$(\mathbf{a}_{1.}, \mathbf{W}, \mathbf{c})$ $\vdots$ $(\mathbf{a}_{I.}, \mathbf{W}, \mathbf{c})$

In this table, a_{ik} denotes the individual membership entries from $\mathbf{A}$, $\mathbf{a}_{i.}$ the memberships in row i of $\mathbf{A}$, $\mathbf{W}$ the weights, $\mathbf{w}_{.k}$ the weights associated with cluster k (i.e., column k of $\mathbf{W}$), and $\mathbf{c}$ the constants ($i = 1, \ldots, I$; $k = 1, \ldots, K$). Note that parameters between parentheses (i.e., $()$) are updated simultaneously.

previous section. At this point, we are interested in two aspects of algorithmic performance: optimization and recovery. With regard to optimization, we will examine to which extent each algorithm finds the global optimum of the loss function. Concerning recovery, we will investigate to which extent each algorithm succeeds in recovering the true structure underlying a given data set. Regarding both aspects we will not only study to which extent the algorithms differ from one another globally (i.e., considered over all simulated data sets), but also how both algorithmic performance and possible performance differences among the algorithms may depend on characteristics of the data. At this point, we want to focus on three such characteristics: (1) the number of underlying clusters, (2) the amount of cluster overlap in the underlying clustering, and (3) the level of noise. That allows us to investigate whether possible performance differences among the algorithms are more pronounced in more difficult situations (e.g., with many underlying clusters, with a large amount of cluster overlap, and/or with a large level of noise).

In Subsection 4.2 we will outline the design of the simulation study. The results will be presented in Subsection 4.3 and discussed in Subsection 4.4.

4.2 Design

To generate a data array $\underline{\mathbf{S}}$, a binary $I \times K$ membership matrix $\mathbf{A}$, a non-negative real-valued $J \times K$ weight matrix $\mathbf{W}$, real-valued constants c_j, and (given the symmetric nature of the data) symmetric $I \times I$ real-valued noise matrices $\mathbf{E}_j$ ($j = 1, \ldots, J$) are generated independently. The rows of the $I \times K$ matrix $\mathbf{A}$ are independently drawn from a multinomial distribution on all possible 2^K binary row patterns (with a probability of .05 for the zero pattern[5] and with the probabilities for row patterns with a single 1 put equal to each other). The entries w_{jk} of $\mathbf{W}$ are independently drawn from a uniform distribution on the interval $(10, 100)$. A symmetric $\mathbf{E}_j$ is generated by drawing the diagonal and lower triangular entries from a univariate normal distribution $N(0, \sigma_E^2)$ and by taking the upper triangle as a copy of the lower triangle. The constants c_j are uniformly generated from the interval $(-10, 10)$; subsequently a fixed number is added to all constants which is such that all data entries are positive. A data array $\underline{\mathbf{S}}$ is then obtained as J matrices $\mathbf{S}_j$, with $\mathbf{S}_j = \mathbf{M}_j + \mathbf{E}_j$, and with $\mathbf{M}_j = \mathbf{A}diag(\mathbf{w}_{j.})\mathbf{A}' + c_j\mathbf{1}\mathbf{1}'$ being the true INDCLUS model underlying the data.

5. An overlapping clustering of the objects implies that there are no constraints on the binary membership matrix, except that clusters cannot be empty (because in that case the number of clusters is less than the desired number). As such, it may happen that an object does not belong to any cluster. From a substantial point of view, this may make sense: For example, when the objects refer to patients and the clusters to underlying syndromes, it is possible that a patient does not suffer from any of the syndromes under study.

In the data generation, the following factors were manipulated:

- *The shape of the data array:* The shape of the data array $\underline{\mathbf{S}}$ is defined in terms of the ratio I/J (i.e., the ratio of the number of objects to the number of raters) and takes three different values: 1/8, 1, and 8. Given that the total number of entries in $\underline{\mathbf{S}}$ is kept fixed at $32,768$, this implies three different values for $I \times I \times J$: $16 \times 16 \times 128$, $32 \times 32 \times 32$, $64 \times 64 \times 8$.
- *The number of underlying clusters, K:* 2, 3, or 4.
- *The amount of cluster overlap:* This factor is defined as the probability for an object to belong to more than one cluster and it is put equal to 25%, 50%, or 75%; for this purpose, the multinomial probabilities of all row patterns in $\mathbf{A}$ that contain more than a single 1 (i.e., objects belonging to more than one cluster) were put equal to one another with a total equal to the percentage in question.
- *The level of noise, ε:* This factor is defined as the proportion ε of the total variance in the data $\underline{\mathbf{S}}$ accounted for by $\underline{\mathbf{E}}$, which is controlled through σ_E^2 (for more information about this procedure, see Wilderjans, Ceulemans, Van Mechelen, and van den Berg 2011; Wilderjans, Ceulemans, and Van Mechelen 2009; Schepers, Van Mechelen, and Ceulemans 2006). The proportion ε will be either 0, .10, .20, .30, .40, .50, .60, .70, .80, .90, .95, .975, .99, or .995.

All design factors were fully crossed. It yields 3 (*Shape of the data array*) $\times$ 3 (*Number of underlying clusters*) $\times$ 3 (*Amount of cluster overlap*) $\times$ 14 (*Level of noise*) $=$ 378 data conditions for each of which 10 replicates will be generated, which yields $3,780$ simulated data sets.

Each data set is then analyzed with SYMPRES and with all five alternative ALS procedures. As the result of an ALS algorithm may strongly depend on the initial parameter estimates, a multistart procedure is implemented for all algorithms with 100 random initial starting values for the parameters.

4.3 Results

4.3.1 Optimization Performance

In this section we want to study to which extent the different algorithms were able to find the global optimum of the loss function. However, because noise $\mathbf{E}_j$ has been added to a true model $\mathbf{M}_j$ ($j = 1, \ldots, J$), this global optimum is unknown. Therefore, we introduce the concept of *proxy* or *pseudo-optimum*, which may be determined for each data set separately

and which may act as an approximation of the global optimum. In particular, the value of the proxy for each data set equals the loss function value associated with the best encountered solution among: (1) the solutions obtained by analyzing the data set at hand with the different algorithms under study, (2) the solutions obtained by running the different algorithms under consideration seeded with the true membership matrix $\mathbf{A}^{\mathbf{true}}$, and (3) the true underlying solution (i.e., $\mathbf{M}_j$). This definition implies that either no, one, or several algorithms can reach the proxy for a given data set at hand.

First of all, we examined the overall optimization performance of each algorithm. To this end, in Figure 2 for each algorithm the percentage of data sets for which the best multistart run equals the proxy (i.e., the estimated global optimum of the loss function) is displayed. In this figure, one can see that the first three modifications do not enhance optimization performance, while the fourth modification does so to a large extent. Moreover, the third modification appears to function as a catalyst for the fourth modification in that the third modification, without affecting the optimization performance in its own, makes the effect of the fourth modification more pronounced.

We also studied for each data set and for each algorithm the percentage of multistart runs that hit the proxy. It appears that modifying SYMPRES with the third and/or fourth modification enhances this aspect of optimization performance (with the number of multistart runs that hit the proxy ranging, on average, from 30% to 58%).

Second, we studied the optimization performance as a function of data characteristics. Therefore, we performed a repeated measures analysis of variance with the percentage of multistart runs that ended in the proxy as the dependent variable and the data characteristics and the algorithm-factor as independent variables. This analysis revealed a large main effect of the noise level, with an effect size η^2_G (Bakeman 2005) of $.42$: When the level of noise becomes larger, the different algorithms are more sensitive to local optima. Further, this analysis revealed a sizeable main effect ($\eta^2_G = .10$) of the algorithm (see above) and the amount of cluster overlap ($\eta^2_G = .07$): The optimization performance decreases when clusters are more overlapping.

4.3.2 Cluster Structure Recovery

In this section we will evaluate to which extent each algorithm is able to disclose the true cluster structure underlying a data set. To this end, we calculated for each data set and for each algorithm the kappa coefficient (Cohen 1960) between the entries of $\hat{\mathbf{A}}$ (i.e., the clustering found by the algorithm under study) and the entries of $\mathbf{A}^{\mathbf{true}}$ (i.e., the true underlying

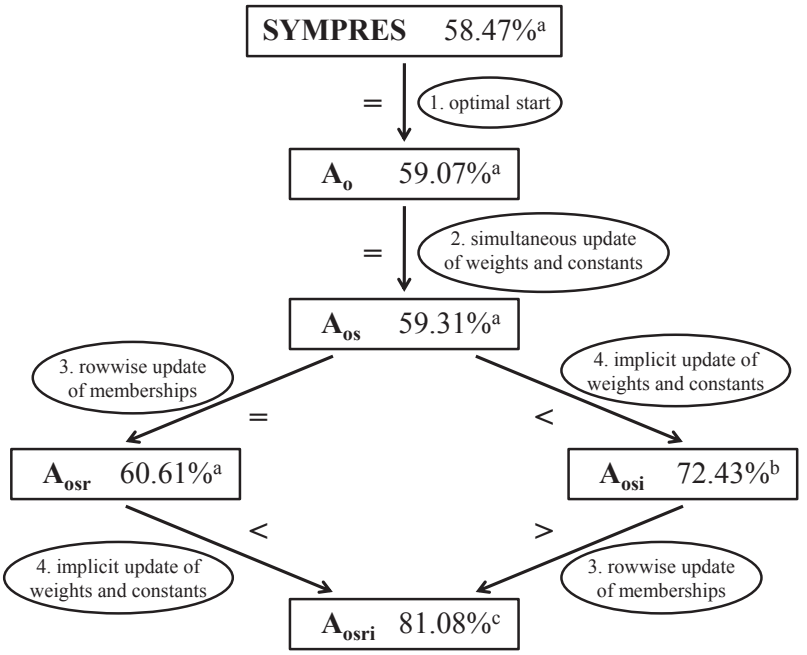


Figure 2. The percentage of data sets for which each algorithm hits the proxy, computed over all data sets. Algorithms indicated with a different character differ significantly from each other at .05 level, tested with a generalized mixed model and taking multiple (contrasts) testing into account.

clustering)[6]. To take the permutational freedom of the (order of the) clusters into account, we defined the *goodness-of-clustering* (GOC)-statistic as the maximum value of the kappa coefficient over all column permutations of $\hat{\mathbf{A}}$. As such, GOC equals the percentage of entries of the true clustering, adjusted for chance, that have been recovered correctly. Note that GOC takes values in the interval $[0, 100]$, with a value of 100 meaning perfect recovery.

6. The kappa coefficient κ between two binary matrices equals the proportion of agreement between the two matrices (i.e., proportion of corresponding cells that have the same value), corrected for chance. In particular, κ can be computed as follows:

$$\kappa = \frac{(p_{00} + p_{11}) - (p_{0.}p_{.0} + p_{1.}p_{.1})}{1 - (p_{0.}p_{.0} + p_{1.}p_{.1})}, \tag{8}$$

with p_{00} (p_{11}) being the proportion of corresponding cells that both are zero (one) and $p_{0.}$ and $p_{1.}$ ($p_{.0}$ and $p_{.1}$) the marginal proportion of zero- and one-cells for the first (second) matrix. Note that $p_{00} + p_{11}$ equals the (uncorrected) proportion of corresponding cells that have the same value. In the results, this coefficient is presented as a percentage (i.e., $\kappa \times 100$).

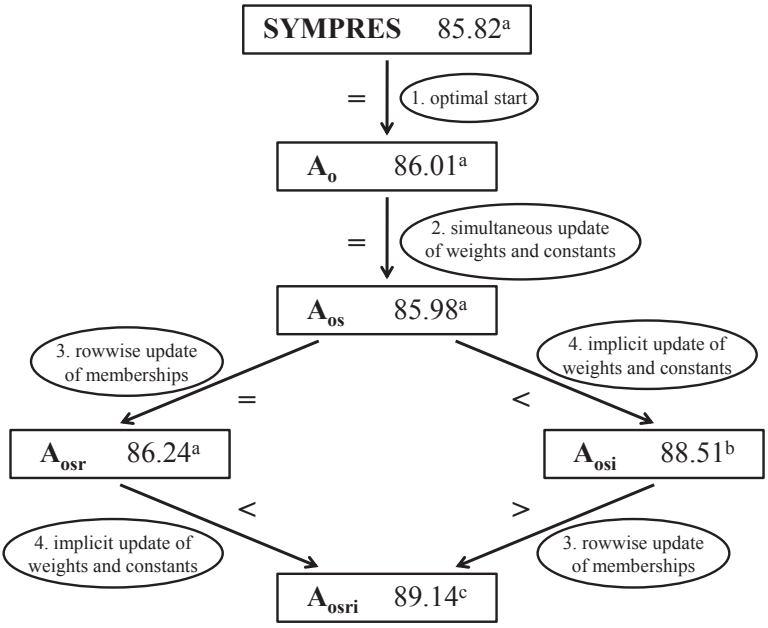


Figure 3. Mean *goodness-of-clustering* (GOC) value, computed over all data sets, for each algorithm. Algorithms indicated with a different character differ significantly from each other at .05 level, tested with Tukey-HSD procedure (Tukey, 1953).

First, we focus on the overall recovery performance of the different algorithms. To this end, in Figure 3, for each algorithm, the mean GOC over all data sets is displayed. From this figure, it appears that only the fourth modification increases the recovery performance, while the third modification again functions as a catalyst for the effect of the fourth one.

Second, to study how the recovery performance varies as a function of data characteristics, we performed a repeated measures analysis of variance with the GOC value as the dependent variable and the data characteristics and the algorithm-factor as independent variables. In this analysis, only taking effects into account with $\eta^2_G \geq .10$ (Bakeman 2005), it appears that most of the variance in recovery performance is explained by a large main effect of the level of noise ($\eta^2_G = .89$): Recovery performance decreases when the level of noise becomes larger, with low recovery values especially being encountered for large noise levels. This main effect, however, is qualified by a level of noise by amount of cluster overlap interaction ($\eta^2_G = .10$), which, in turn, is qualified by an interaction between the noise level, the amount of cluster overlap and the number of underlying clusters ($\eta^2_G = .12$): The decrease in recovery performance with increasing noise levels is more pro-

nounced when clusters overlap to a large extent, with this especially being the case when the number of underlying clusters is small.

4.4 Discussion

The simulation study demonstrated that the fourth modification enhances the optimization and recovery performance, with the third modification in each case figuring as a catalyst. However, when evaluating this result it is important to note that up to now differences between the algorithms in computational effort (i.e., computation time) have not been taken into account (with all algorithms being applied with a multistart procedure with 100 starts). Regarding computation time, it is useful to know that the different algorithms were all implemented in MATLAB-code (version $7.11.0.584$, $R2010b$) and run on a supercomputer consisting of INTEL XEON $L5420$ processors with a clock frequency of 2.5 GHz and with 8 GB RAM. Under these circumstances, it appears that the SYMPRES algorithm needs about half a second to perform one multistart run. Adding the first and/or second modification does not increase the computation time, whereas the third modification lengthens the computation time, on average, with 18 seconds, while the fourth modification implies a lengthening of the processing time with 4 seconds. As such, on average, the total computation time for the algorithms $\mathbf{A_o}$, $\mathbf{A_{os}}$, $\mathbf{A_{osr}}$, $\mathbf{A_{osi}}$, and $\mathbf{A_{osri}}$ (for a single run) amounts to $.5$, $.5$, 18.5, 4.5, and 22.5 seconds, respectively.

To examine to what extent the obtained optimization and recovery results are to be attributed to differences in computational effort, we re-analyzed a part of the simulated data sets, in such a way that each algorithm was given the same amount of time to arrive at a solution. To achieve this, we computed for each condition the mean time each algorithm needed to run 100 multistart runs. Next, we re-analyzed each data set in each condition by each algorithm and manipulated the number of multistart runs such that each algorithm was given the same amount of time. Since large differences in performance especially appeared for large noise conditions, we limited these re-analyses to the data sets that had large levels of noise. The results are summarized in Figure 4. From this figure it appears that, when keeping time constant, the (catalyzing) effect of the third modification vanishes, while the effect of the fourth modification, although becoming smaller, remains present.

5. Concluding Remarks

In this paper, we presented four algorithmic modifications and five novel ALS algorithms to fit the INDCLUS model to a given data set. In an extensive simulation study it was demonstrated that some of the alternative algorithms significantly outperform the SYMPRES algorithm with respect

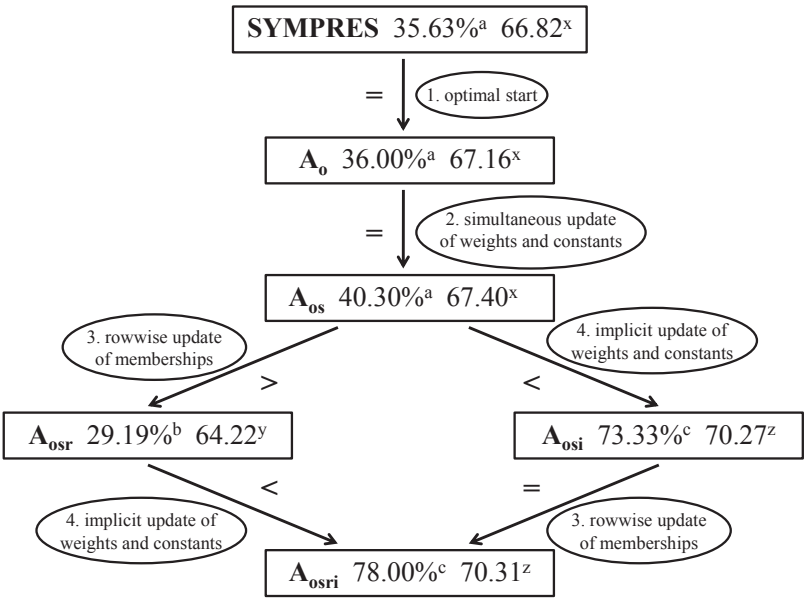


Figure 4. Overview of the performance differences between the alternative algorithms (after keeping differences in computation time constant). The values next to the name of the algorithm refer to, computed over all data sets with $\varepsilon \geq .90$, the mean percentage of data sets for which the corresponding algorithm hits the proxy and the mean GOC value for the corresponding algorithm. Algorithms indicated with a different character differ significantly from each other at .05 level, tested with a generalized mixed model and taking multiple (contrasts) testing into account or with Tukey-HSD procedure (Tukey, 1953).

to optimization and recovery performance (also after keeping differences in computation time constant). With respect to the algorithms, the fourth modification, which consists of simultaneously updating the weights and the constants (in an implicit way) when re-estimating the memberships, seems to be the key modification that accounts for the enhanced optimization and recovery performance; this modification implies that the weights and constants are always at their optimal value given the memberships. Hence, the dependencies between the (binary) membership entries, on the one hand, and the (continuous) weights and constants, on the other hand, appear to be the most critical ones that need to be taken into account.

Interestingly, there are some indications that similar conclusions might also hold for other clustering problems, like *K*-means analysis (Dhillon, Guan, and Kogan 2002; Larsen and Aone 1999; Steinbach, Karypis and Kumar 2000) and three-mode partitioning (Schepers, Van Mechelen, and Ceulemans 2006). For example, with respect to the latter, Schepers, Van

Mechelen, and Ceulemans (2006) have demonstrated that implicitly updating the continuous parameters when re-estimating the binary cluster memberships (i.e., DRIFT algorithms) enhances algorithmic performance.

In general, it would be interesting to investigate in further research whether our findings on the importance of dependencies between binary and continuous parameters also apply to the broader category of mixed discrete-integer optimization problems.

References

ARABIE, P., and CARROLL, J.D. (1980), "MAPCLUS: A Mathematical Programming Approach to Fitting the ADCLUS Model," *Psychometrika, 45*, 211–235.

BAKEMAN, R. (2005), "Recommended Effect Size Statistics for Repeated Measures Designs," *Behavior Research Methods, 37*, 379–384.

BRO, R., and DE JONG, S. (1997), "A Fast Non-Negativity-Constrained Least Squares Algorithm," *Journal of Chemometrics, 11*, 393–401.

CARROLL, J.D., and ARABIE, P. (1983), "INDCLUS: An Individual Differences Generalization of the ADCLUS Model and the MAPCLUS Algorithm," *Psychometrika, 48*, 157–169.

CEULEMANS, E., VAN MECHELEN, I., and LEENEN, I. (2007), "The Local Minima Problem in Hierarchical Classes Analysis: An Evaluation of a Simulated Annealing Algorithm and Various Multistart Procedures," *Psychometrika, 72*, 377–391.

CHATURVEDI, A., and CARROLL, J.D. (1994), "An Alternating Combinatorial Optimization Approach to Fitting the INDCLUS and Generalized INDCLUS Models," *Journal of Classification, 11*, 155–170.

COHEN, J. (1960), "A Coefficient of Agreement for Nominal Scales," *Educational and Psychological Measurement, 20*, 37-46.

DE LEEUW, J. (1994), "Block-Relaxation Algorithms in Statistics", in: *Information Systems and Data Analysis*, eds. H.H. Bock, W. Lenski, and M.M. Richter, Berlin: Springer-Verlag, pp. 308–325.

DEPRIL, D., VAN MECHELEN, I., and MIRKIN, B.G. (2008), "Algorithms for Additive Clustering of Rectangular Data Tables," *Computational Statistics and Data Analysis, 52*, 4923–4938.

DHILLON, I.S., GUAN, Y., and KOGAN, J. (2002), "Refining Clusters in High-Dimensional Text Data", in: *Proceedings of the Workshop on Clustering High Dimensional Data and its Applications at the Second SIAM International Conference on Data Mining*, eds. I.S. Dhillon, and J. Kogan, SIAM 2002, pp. 71–82.

GELMAN, A., CARLIN, J.B., STERN, H.S., and RUBIN, D.B. (1995), *Bayesian Data Analysis*, London: Chapman and Hall.

GOODMAN, J., and SOKAL, A.D. (1989), "Multigrid Monte Carlo Method: Conceptual Foundations," *Physical Review D, 40*, 2035–2071.

HARSHMAN, R.A., and LUNDY, M.E. (1984), "The PARAFAC Model", in: *Research Methods for Multimode Data Analysis*, eds. H.G. Law, C.W. Snyder, Jr., J.A. Hattie, and R.P. McDonald, New York: Praeger, pp. 122–215.

KIERS, H.A.L. (1997), "A Modification of the SINDCLUS Algorithm for Fitting the ADCLUS and INDCLUS Models," *Journal of Classification, 14*, 297–310.

LARSEN, B., and AONE, C. (1999), "Fast and Effective Text Mining Using Linear-Time Document Clustering", in: *Proceedings of the Fifth ACM SIGKDD*, San Diego, CA, pp. 16–22.

LAWSON, C.L., and HANSON, R.J. (1974), *Solving Least Squares Problems*, Englewood Cliffs, NJ: Prentice-Hall Inc.

MILLIGAN, G.W. (1980), "An Examination of the Effect of Six Types of Error Perturbation on Fifteen Clustering Algorithms," *Psychometrika, 45*, 325–342.

MIRKIN, B.G. (1987), "The Method of Principal Clusters," *Automation and Remote Control, 10*, 131–143.

MIRKIN, B.G. (1990), "A Sequential Fitting Procedure for Linear Data Analysis Models," *Journal of Classification, 7*, 167–195.

MIRKIN, B.G. (1996), *Mathematical Classification and Clustering (Nonconvex Optimization and its Applications)*, Boston-Dordrecht: Kluwer Academic Press.

ROBERTS, G.O., and SAHU, S.K. (1997), "Updating Schemes, Correlation Structure, Blocking and Parameterization for the Gibbs Sampler," *Journal of the Royal Statistical Society: Series B, 59*, 291–317.

SCHEPERS, J., VAN MECHELEN, I., and CEULEMANS, E. (2006), "Three-Mode Partitioning," *Computational Statistics and Data Analysis, 51*, 1623–1642.

SEEWALD, W. (1992), "Discussion on Parameterization Issues in Bayesian Inference (by S. E. Hills and A. F. M. Smith)", in: *Bayesian Statistics 4*, eds. J.M. Bernardo, J.O. Berger, A.P. Dawid, and A.F.M. Smith, Oxford: Oxford University Press, pp. 241–243.

SHEPARD, R.N., and ARABIE, P. (1979), "Additive Clustering: Representation of Similarities as Combinations of Discrete Overlapping Properties," *Psychological Review, 86*, 87–123.

SMITH, A.F.M., and ROBERTS, G.O. (1993), "Bayesian Computation via the Gibbs Sampler and Related Markov Chain Monte Carlo Methods," *Journal of the Royal Statistical Society: Series B, 55*, 3–23.

STEINBACH, M., KARYPIS, G., and KUMAR, V. (2000), "A Comparison of Document Clustering Techniques", in: *Proceedings of the Sixth ACM SIGKDD, World Text Mining Conference*, Boston, MA.

TEN BERGE, J.M.F., and KIERS, H.A.L. (2005), "A Comparison of Two Methods for Fitting the INDCLUS Model," *Journal of Classification, 22*, 273–286.

TUKEY, J.W. (1953), "The Problem of Multiple Comparisons", Mimeographed monograph.

TVERSKY, A. (1977), "Features of Similarity," *Psychological Review, 84*, 327–352.

WILDERJANS, T.F., CEULEMANS, E., VAN MECHELEN, I., and VAN DEN BERG, R.A. (2011), "Simultaneous Analysis of Coupled Data Matrices Subject to Different Amounts of Noise," *British Journal of Mathematical and Statistical Psychology, 64*, 277–290.

WILDERJANS, T.F., CEULEMANS, E., and VAN MECHELEN, I. (2009), "Simultaneous Analysis of Coupled Data Blocks Differing in Size: A Comparison of Two Weighting Schemes," *Computational Statistics and Data Analysis, 53*, 1086–1098.