ELSEVIER

Chemometrics and Intelligent Laboratory Systems 43 (1998) 25–42

Chemometrics and intelligent laboratory systems

Three-way (PARAFAC) factor analysis: examination and comparison of alternative computational methods as applied to ill-conditioned data

Philip K. Hopke [a,*], Pentti Paatero [b], Hong Jia [c,1], Robert T. Ross [d,2], Richard A. Harshman [e]

[a] *Department of Chemistry, Clarkson University, Potsdam, NY 13699-5810, USA*
[b] *Department of Physics, University of Helsinki, P.O. Box 9, FIN-00014, Helsinki, Finland*
[c] *Department of Civil and Environmental Engineering, Clarkson University, Potsdam, NY 13699-5810, USA*
[d] *Department of Biochemistry, Ohio State University, Columbus, OH 43210, USA*
[e] *Department of Psychology, University of Western Ontario, London, Ontario, Canada N6A 5C2*

Abstract

Four different approaches to solving the trilinear three-way factor analysis problem are compared, and their performance with 'difficult' (i.e., ill-conditioned) data is tested. These approaches are represented by four different computer programs: one using a simple alternating least squares (ALS) algorithm with only minimal extrapolation (HL-PARAFAC), one in which the ALS is supplemented by a sophisticated extrapolation to speed convergence (TPALS), one using a non-linear curve fitting method (PMF3), and one using a non-iterative closed-form approximation (DTDMR). The options provided by these programs (e.g., with regard to missing values, weighted least squares, non-negativity and other constraints) are compared. Criteria for choosing synthesized test data and a method for synthesizing exponential test data are described. A numerical index is introduced to characterize the ill-conditioning of n-way arrays ($n > 2$). Two well characterized synthetic data sets serve as 'difficult' (ill-conditioned) test data. Intercomparisons among HL-PARAFAC, TPALS, DTDMR and PMF3 were implemented with these test data. Consequently, their limitations and strengths are determined. In addition, these trilinear analysis approaches are applied to a difficult set of ill-conditioned real data: a set of fluorescence spectroscopy measurements that characterize the steady-state fluorescence of an amino acid in aqueous solution. When converged, the results produced by the three least-squares techniques (but not DTDMR) agree. However, there are large differences in convergence speed when these difficult problems are solved: TPALS is faster than PARAFAC by a factor of ten, and PMF3 is faster than TPALS, again by a factor of ten. The program DTDMR is the fastest, but it only solves half of the problems. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: Factor analysis; Trilinear; PARAFAC; Fluorescence spectroscopy; PMF3; TPALS; DTDMR

* Corresponding author.
[1] Present address: 1090 Sunnyvale-Saratoga Road, Apt. 35, Sunnyvale, CA 94087, USA.
[2] Present address: Computer Science Department, California Polytechnic State University, San Luis Obispo, CA 93407, USA.

0169-7439/98/$ - see front matter © 1998 Elsevier Science B.V. All rights reserved.
PII: S0169-7439(98)00077-X

1. Introduction

During the past 10 years, there has been an increasing use of three-way factor analysis in chemometrics. This growth has provided many useful methods for chemistry applications including second order multivariate calibration, receptor modeling for air quality management, and fluorescence spectroscopy characterizing macromolecular or membrane systems. There are many groups who are developing new or finding improvements to existing methods.

Several different algorithms have been implemented to fit the PARAFAC factor analysis model, and it seems likely that they might behave in different ways when applied to 'difficult' problems (e.g., problems where high collinearity of factors, and/or large difference in the relative sizes of factors, and/or near proportionality of factor changes across one or more modes cause the dataset to be ill-conditioned). It is proposed that a standardized intercomparison method, and well-characterized test data sets, be provided to serve as a tool for such study. This paper will discuss a method for synthesizing sets of (ill-conditioned) data for intercomparison purposes, and will use two such datasets to test the four PARAFAC algorithms. In addition, the algorithms were tested by applying them to a quite ill-conditioned set of real data obtained via fluorescence spectroscopy.

2. The PARAFAC / CANDECOMP trilinear model

We investigate the trilinear factor analysis model generally referred to as either the Parallel Factor Analysis (PARAFAC) model or the Canonical Decomposition (CANDECOMP) model [1,2]. For brevity, it will be referred to as the PARAFAC model. In this trilinear model [1–5], elements x_{ijk} of a three-way array **X** are expressed as sums of products of elements from 'factor loading' matrices **A**, **B**, and **C**, according to the equation

$$x_{ijk} = \sum_{h=1}^{p} a_{ih} b_{jh} c_{kh} + e_{ijk} \quad (1)$$

This model is a straightforward generalization of the two-dimensional principal component analysis or 'factor analysis' model. There are, of course, other generalizations but they are not discussed in this context. The unknown matrices **A**, **B**, and **C** are to be determined so that a 'best fit' of the array **X** is achieved. The original definition of best fit required that the unweighted sum-of squares

$$Q(E) = \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{o} e_{ijk}^2 \quad (2)$$

be minimized. In order to utilize information about expected sizes of data errors, the weighted least squares expression

$$Q(E) = \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{o} w_{ijk} e_{ijk}^2 \quad (3)$$

has sometimes been preferred to Eq. (2). One common weighting scheme is to choose the weights as

$$w_{ijk} = \frac{1}{\sigma_{ijk}^2} \quad (4)$$

where the σ_{ijk} are the expected standard deviations for the observed values x_{ijk}, based either on theory about the error process or on observed variability across replications of the data array.

The two-way model may sometimes become identifiable in the presence of sufficient constraints such as those imposed by non-negativity or orthogonality, but in general it is under-identified (except when there is only one factor), and so is said to possess 'rotational freedom.' In contrast, a sufficient condition for the three-way model to be fully identified (up to trivial differences in factor order and relative scaling across modes) is that the factor loading matrices, **A**, **B**, and **C**, obey the Kruskal conditions ([5], see Theorem 4a; [6]). These conditions can be described in terms of the *k*-rank of each matrix. The *k*-rank of an *R*-column matrix **A** is $k(a)$ if the columns are linearly independent in every set of $k(a)$ columns from **A**, and if either $k(a) = R$ or there is at least one set of $(k(a) + 1)$ columns of **A** that includes linearly dependent columns [7]. Then if I_0 is the *k*-rank of **A**, J_0 is the *k*-rank of **B**, and K_0 is the *k*-rank of **C** and if $I_0 + J_0 + K_0 >= 2R + 2$ (where R is the number of columns in each of **A**, **B**, and **C**), then there are

no free rotations. The decomposition is uniquely determined except for the trivial scale and factor order differences. A useful corollary is that the solution is unique if any two of the factor loading matrices are of full rank and no two columns of the third matrix are proportional to each other [8–10]. However, this criterion is more restrictive than the Kruskal conditions.

The uniqueness of the results obtained by three-way factorization makes it a very useful technique. In two-way analysis with rotational indeterminancy, it can be difficult to identify the valid solution. Whenever there are repeated two-way data sets so that each one could be analyzed by the customary two-way techniques, one should recognize the possibility of using the three-way array that may be assembled by stacking the two-way matrices. If this array truly follows the three-way model given in Eq. (1) and is likely to fulfill the conditions of factor loading independence needed for uniqueness (discussed above), then it may be better to analyze the whole set as a three-way array instead of performing repeated two-way analyses.

3. Solution methods

3.1. Least squares approach

Several programs are readily available for solving the PARAFAC model as a least squares (LS) problem. Here those properties of the programs that relate to the basic model will be reviewed. Later enhancements to this basic model will be presented.

The original programs were named 'PARAFAC' and 'CANDECOMP' (as part of the MDS routine INDSCAL [1]). The estimation approach of the two programs was very similar, so the PARAFAC program will be used to represent both. In this article it will be referred to as 'HL-PARAFAC' to distinguish this particular program from the general PARAFAC model. HL-PARAFAC was written by Harshman [2] and Harshman and Lundy [3]. The program has been developed further in recent years. There now exist both a FORTRAN 77 version and a MATLAB version of the code; the latter (newer) version has some added functionality, such as optional weighted least squares and/or fitting of multiple additive constants (for fitting one kind of 'extended PARAFAC' [see Ref. [4]]). Like many programs for fitting multilinear models, HL-PARAFAC's least-squares calculation is based on the alternating least squares (ALS) algorithm. Each substep of ALS fixes two of the factor matrices (**A**, **B**, and **C**), then uses linear regression to find the third factor matrix. The algorithm cycles among the three factor matrices, updating each in turn. We define an ALS step as one substep for each factor matrix. HL-PARAFAC extrapolates by a fixed fraction of each ALS step to speed convergence (see Ref. [2], p. 33).

Leurgans and Ross [8], Leurgans et al. [9], Ross and Leurgans [10], and Lee et al. [11] have developed a non-negative, weighted ALS algorithm 'TPALS' for fitting the PARAFAC model (available as FORTRAN code that calls IMSL subroutines). The algorithm contains a more sophisticated and powerful acceleration feature. After individual increments to all the factor matrices have been computed in two ordinary ALS steps, the program performs an extension based on the relative magnitude of the two ALS steps and the angle between them. When the two successive ALS steps are nearly identical, which is often the case, the projection can be up to 200 times the last step.

Paatero [12,13] has recently described a non-negative weighted LS algorithm 'PMF3' for fitting the PARAFAC model. In contrast to ALS, in this algorithm there is no alternation between the modes: all of the modes are acted upon at the same time. PMF3 is based on the well-known Gauss–Newton curve fitting technique with Levenberg–Marquardt control against exceedingly long steps. An additional improvement of convergence rate is achieved by a Newton–Raphson-like correction of steps. The process is iterative. One step only produces an approximate solution of the problem and multiple steps are needed to obtain convergence.

3.2. Eigenvector decomposition

The original 'direct trilinear decomposition' (DTD) approach was developed by Wilson et al. [14] and Sanchez and Kowalski [15]. The technique was developed further by Zeng and Hopke [16] and was called DTDMR (direct trilinear decomposition followed by a matrix reconstruction). A more detailed

description and discussion and the algorithms can be found in Wilson et al. [14]. For the special case in which the three-way data block contains only two slices in mode 3 ($K = 2$), a_n and b_n can be calculated using the generalized rank annihilation method [15]. In general, there are more than two slices ($K > 2$). In order to use the generalized rank annihilation method, two linear combinations of all slices need to be found. The process for developing these combinations is provided in the literature [9,14–16].

The DTD approach differs from the least squares methods in that its properties are only known for 'exact data'. If the array rank of the three-way array is r, then the r factor DTD decomposition of the array may be an exact representation of the array. However, an exact representation is usually not sought for experimental data that includes error. The experimental array is of some high rank and a low-rank representation (describing the 'reliable part') is needed. The approximating properties of DTD are not known.

3.3. Enhancements to PARAFAC estimation

3.3.1. Multiple starting points

It is well known that the trilinear factor analysis problem may have multiple locally optimal solutions. One should note that this is a property of the model and not of the algorithms used for solving it. It has been suggested by Paatero [13] that all local solutions should be inspected and considered. It may not be safe to assume that the solution with best fit would necessarily be the desired solution. Thus, an approach is needed for finding those local solutions. HL-PARAFAC and PMF3 are usually run from multiple pseudo-random starting points. It is recommended that 5 to 10 starts are made whenever one wishes to explore the presence of local minima. The local minima can provide confirmation that solutions are all close to the global minimum and thus, it is likely that the system has converged to the best available fit. The results of the analysis not only must fit the data, but must be physically interpretable in terms of the system being analyzed. The apparent global minimum may not necessarily be interpretable and thus, it is often useful to carefully examine the solutions obtained from the other starting points.

Because of its non-iterative closed-form approach, there are no 'starting points' in DTD, although one could investigate fits to different subsets of the data. It is not known how DTD will behave in the presence of multiple local solutions. The program TPALS obtains some efficiency boost by using a decomposition as the starting point for the LS iterations. It obtains multiple starting points by performing multiple decompositions, and then it abandons all but the start which gives the best fit after a few steps.

3.3.2. Data point weighting and treatment of missing values

It is possible to utilize information about the expected standard deviations of data points in a weighted least squares fit. In a two-way model, the usual (centered, standardized) Principal Components solution is implicitly based on assumptions about the standard deviations of data points that are often unrealistic (see Paatero and Tapper [17]). In the three-way case, the eigenanalysis has been abandoned in any case as the tool for achieving a LS fit. Thus there is no temptation to continue this questionable practice. It is relatively easy to configure any LS program so that it performs a weighted least squares fit according to Eq. (3). It may be much harder to persuade the end user that there would be a benefit in using the weighted fit. The need to recompute the ALS fit matrix separately for each slice of the array causes some extra work when performing a weighted fit by the ALS. This reweighting may increase the workload by 50 to 200% in comparison to the unweighted case. The programs PMF3, TPALS, and HL-PARAFAC (MATLAB) can perform a weighted fit if desired as in Eqs. (3) and (4). For the DTD approach, the concept of weighting does not seem applicable, as there is no LS fit to begin with.

HL-PARAFAC handles missing values by iterative re-estimation of cells with missing data. When performing a weighted LS fit with any of the LS programs the handling of missing values is quite straightforward: basically one simply gives zero weight for the missing value; then it does not matter what is used as the data value. There may be some advantage in specifying a non-zero small weight, however, and using a 'typical' value as the data. This weighting may prevent the appearance of unrealistic

large values for such factor elements which are badly defined or undefined because of the presence of many missing values.

3.3.3. Constraints on the solutions: non-negativity and orthogonality

In different sciences, there may be different constraints that are appropriately applied to the factors. In the social sciences, orthogonality or zero-correlation of the loadings for one of the modes (e.g., person weights) is often preferred, either for simplicity or to suppress 'degenerate solutions' (see Ref. [4], pp. 271–279). Orthogonality constraints are usually used with mean-centered data and the (weaker) zero-correlation constraints for those cases where factors in a given mode are expected to have all positive loadings. Orthogonality and/or zero correlation constraints can be applied to one or more modes as an option in HL-PARAFAC code. HL-PARAFAC also includes optional routines for preprocessing one's data by applying various combinations of mean adjustments (centering rows, columns, and/or 'tubes') and scale adjustments (size-standardization of the slices in a given mode or modes), including the ability to apply these adjustments iteratively until the preprocessed data converge on values which simultaneously fulfill the requested requirements.

Non-negativity is often required in the physical sciences in order that the factors be directly interpretable as concentrations, densities, spectra, masses, etc. Non-negativity is an option in PMF3, TPALS and HL-PARAFAC (MATLAB). ALS algorithms use the algorithm, NNLS, by Lawson and Hanson [18] or newer variants. A penalty function approach is used in PMF3 to enforce non-negativity. A new approach to imposing non-negativity constraints [19] has recently been proposed, but it has not been implemented in any of these programs.

3.4. Other enhancements

As one might expect, most programs have features unique to that program, which could not be used when performing analyses for the comparisons. For example, PMF3 can compute the matrix of covariances among all parameters, and can perform iterative reweighting of data points to provide a more robust fit [20]. Such unique features will not be presented in this article.

4. Test cases used

Previous experience had suggested that most factor analysis algorithms are likely to behave comparably with 'easy' cases, but will show distinct strengths and weaknesses with difficult datasets. 'Difficult' means datasets where (a) the contributions of individual factors overlap so much that the factors are almost collinear, and/or (b) factor sizes differ by two or more orders of magnitude, and/or (c) some subset of factors will show almost proportional changes in factor size across the levels of a mode, so that the unique axis orientation is only weakly determined.

Conditions (a) and (c) really are the same in a strict mathematical sense. However, it is useful to recognize that the problem of collinearity can arise either with factors that have quite similar physical properties in one or two property-modes (e.g., similar excitation and/or emission spectra), or instead are tested with samples where there is inadequate independence of two (or more) factors' variation in concentration across the 'third' mode. This latter circumstance can arise much more easily than the first, and can cause problems when the scientists doing the analysis do not realize the difficulty.

Such 'difficult' datasets are not infrequent in chemometrics, where high precision of measurements makes it possible to extract information of ill-conditioned cases. Ill-conditioned factor matrices also occur when an algorithm has to traverse a so-called 'swamp', i.e., a region where positive and negative contributions partially cancel each other (see Ref. [22]). In order to study performance differences in the algorithms, the test cases of the present investigation were constructed or chosen to be more or less ill-conditioned. The ability to handle so called 'degenerate solutions' [3] was not studied as such. However, fast convergence with ill-conditioned factors should also be helpful in such cases.

The ultimate test of factor analytic software is analyzing real data. However, real data contain surprises and one does not always know what is the 'correct result.' Also, real data from one field is usually not instructive to researchers in other fields.

Thus, relying on real-world examples forces one to republish in different fields. For these reasons, it is advisable to also test the techniques with well-documented simulated examples. The aim of the following sections is to define a suite of test arrays so that other researchers could also use them. By using the same test blocks, the results of different groups are more easily comparable.

4.1. Desirable properties of synthetic test data sets

4.1.1. Reasonable size

Test blocks should only be large enough so that they contain the necessary information for supporting the tests. Large matrices make computations slow, and more importantly, it is not possible to show the results in publications if they comprise too many values. Of course, larger matrices may be needed when one is testing how well various algorithms 'scale up'-how well they perform when the number of parameters to be estimated becomes large.

4.1.2. Visual quality

Test matrices should look nice when plotted, so that the human eye may spot changes in the factors easily. This is good for publishing test results, but it is especially important for such matrices which are used for algorithm development. Exponential and Gaussian shapes have proved to make good test factors. But of course, the programs must not be allowed to take advantage of these special shapes.

4.1.3. Range of Values

There should be both large and small values in the factors. It is useful to have some cases that have only non-negative values, in that way one may test both general-purpose programs and programs incorporating non-negativity constraints.

4.1.4. Adjustability

It has to be easy to produce data blocks of different sizes, consisting of factor matrices of varying condition numbers. In this way it is possible to test what happens when the data approach the non-identifiable case.

4.1.5. Transportability

In order that different laboratories could hope to get identical results when comparing their techniques, the data blocks should not get changed in transport, i.e., exactly the same digits should be input by different groups. Also, the rounding when the values are originally written should not change the statistical properties of the data. This criterion was not obeyed in this study when originally writing the difficult four-factor test case, below, with the format G12.5E1. Most of the values were thus written with four decimals which is not quite enough, considering the standard deviation of 0.001. A format of G12.6E1 should have been used. However, once this block has been circulated with the original format, it was probably better to keep it the same throughout the studies and not create confusion by offering versions with different formats. An alternative to transporting the blocks is for different laboratories to create identical pseudorandom test blocks programmatically. However, this is only possible if they all use exactly the same algorithms and the same portable pseudorandom number generator with the same seed.

4.1.6. Appropriate Internal Structure

Data sets should be generated or selected so that the following properties are all appropriate and well characterized: (a) the amount of collinearity between factors, (b) the difference in sizes of factors, (c) the strength or weakness of (characteristics determining) rotational uniqueness, (d) the amount of random error (e) nature of the error distribution and (f) the variations, if any, in error standard deviations across different data points. 'Appropriate' could mean useful either as representative of real data or as extreme and thus 'difficult' test cases for the algorithms under study. The current article focuses mainly on datasets that are difficult test cases with respect to ill-conditioning (to be explained below), but which are not difficult with respect to the level of random error.

4.2. Exponential test cases

Two test datasets were generated in which the factor loadings for each factor in each mode fell along exponential curves. The data were generated with the help of the exponential test function $\mathrm{EX}^n(r)$. Computing $\boldsymbol{x} = \mathrm{EX}^n(r)$ generates a column vector $\boldsymbol{x}$ of dimension n with the elements

$$\boldsymbol{x}_i = c \cdot \exp\left(\frac{n-i}{n-1} r\right) \tag{5}$$

where the constant c may be chosen so that a desired normalization is achieved. For the test cases PP1 and PP@, c was chosen so that the sum of elements of $\boldsymbol{x}$ equals unity. The ratio of the first and last elements of $\boldsymbol{x}$ equals $\exp(r)$. The notation is extended so that $\mathrm{EX}^n(r, s, \ldots, t)$ is defined to mean the matrix consisting of the column vectors $\mathrm{EX}^n(r)$, $\mathrm{EX}^n(s)$, …, $\mathrm{EX}^n(t)$. For example, the array, $\mathrm{EX}^{10}(0,2,5)$, would consist of three columns. All of the elements of the first column are the same and equal 1. Column 2 has elements with values of 1.0000, 0.8007, 0.6412, 0.5134, 0.4111, 0.3292, 0.2636, 0.2111, 0.1690, 0.1353. The elements of column 3 are 1.0000, 0.5738, 0.3292, 0.1889, 0.1084, 0.0622, 0.0357, 0.0205, 0.0117, and 0.0067. A three-factor data block **X** of dimensions 10,8,6 may then be denoted symbolically as

$$\mathbf{X} = 100 * \mathrm{EX}^{10}(0,2,5) * \mathrm{EX}^{8}(0,2,5) * \mathrm{EX}^{6}(0,2,5),$$

where the asterisk denotes the outer matrix product according to Eq. (1).

Two test cases ‘PP1’ and ‘PP2’ were constructed as follows:

$$\mathrm{PP1} = 100 * \mathrm{EX}^{10}(0,2,5) * \mathrm{EX}^{8}(0,2,5) * \mathrm{EX}^{5}(0,2,5) + 0.025\mathrm{RN}(0,1),$$

$$PP2 = 100 * EX^{10}(0,2,4,-4) * EX^{8}(0,2,4,-4) \times * EX^{5}(-4,4,2,0) + 0.001\, RN(0,1),$$

where $\mathrm{RN}(0,1)$ denotes a pseudorandom array of normally distributed values with a mean value of 0 and a standard deviation of 1. The normalized underlying vectors for PP1 are shown in Fig. 1 and the corresponding vectors for PP2 are presented in Fig. 2.

The relative accuracy of the data in the three-factor example PP1 is typically 8%, with 14 values better than 1%. This is a typical level of accuracy for physical sciences. On the other hand, the data in the four-factor example PP2 have a typical relative accuracy of 0.5% while the best value has 0.01%. These values are more accurate than are typically available. Measurements of such accuracy are possible but certainly are not common. These two datasets were deliberately constructed to be ‘difficult’ test cases. By generating factor loadings that fall along closely related exponential curves, a high degree of collinearity in the factor loadings was assured resulting in a high overlap in the contributed variance of some factors.

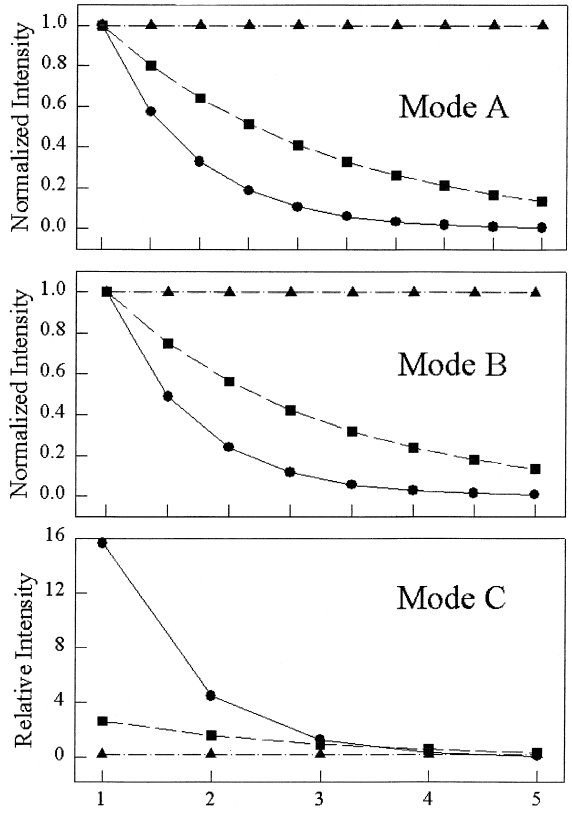

Fig. 1. Plot of the vectors used to generate the error free test data set PP1.

4.3. Real data

For our third and perhaps most challenging test case, an array of fluorescence spectrographic measurements was employed which was known from prior analysis to be both ill-conditioned and have factors that were only weakly identified due to the near proportionality of their loading profiles in at least one mode [11].

The fluorescence of any dilute specimen is separately linear in functions of each of the independent variables excitation wavelength, emission wavelength, and any treatment that alter concentration of fluorescence quantum yield [11]. This trilinear character of the data permits the mathematical decompo-

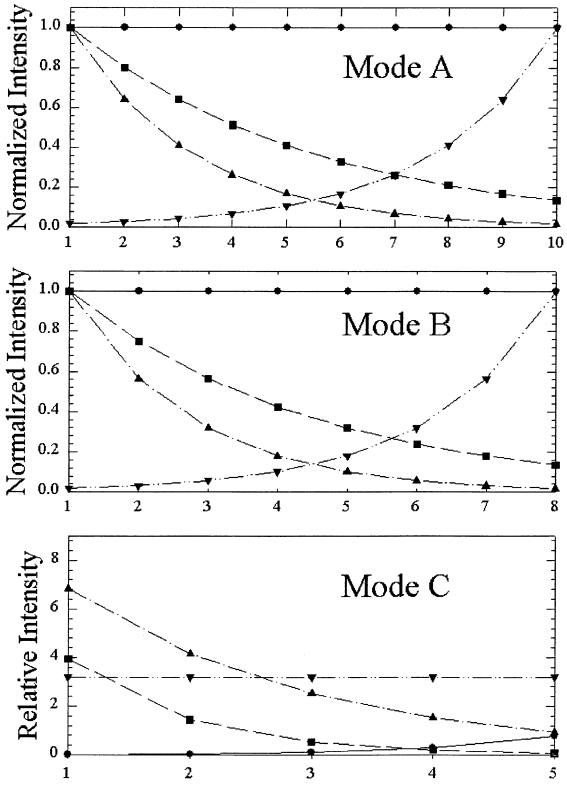

Fig. 2. Plot of the vectors used to generate the error free test data set PP2.

sition of spectra from complex specimens without using any other information about the properties.

Hence, the fluorescence x_{ijk} could be separated by three-way factor analysis as follows:

$$x_{ijk} = \sum_{n=1}^{N} a_n(\lambda_i)\, b_n(\lambda_j)\, c_n([\phi]_k) \qquad (6)$$

where $a_n(\lambda_i)$ is the relative absorbance of component n at excitation wavelength λ_i, $b_n(\lambda_j)$ is the relative fluorescence intensity of component n at emission wavelength λ_j, and $c_n([\phi]_k)$ is the relative fluorescence yield of component n at proton acceptor concentration $[\phi]_k$.

In a previous study [11], the steady-state fluorescence of *N*-acetyl-L-tyrosinamide (NAYA) was measured in aqueous solutions containing six different concentrations at (0.005–2.5 M) of a proton acceptor acetate. The excitation and emission wavelengths varied from 260 to 290 nm and from 300 to 400 nm, respectively. A $19 \times 20 \times 6$ three-way data block was thereby generated. Its characterization has also been studied theoretically [10]. These studies have shown that the emission maximum varies between that of ground-state tyrosine (305 nm) and that of the tyrosinate (345 nm), depending on the proton acceptor concentration. This data set comes with individual standard deviation values computed for all measured points; the data are of good precision, in which the standard deviations are typically a few percent of the value. There are a few missing values in one corner of the array; they are caused by the almost-overlapping wavelength ranges for excitation and emission.

This data block was used as a test dataset and (re)analyzed by all four programs.

4.4. Collinearity Diagnostics

To quantitatively measure and report the ill-conditioning of these datasets we will use two approaches. The first will be to give the singular values of the three individual factor loading matrices, and the second will be to use a numeric index of *array* ill-conditioning. For two-way matrices, it is common to describe how ill-conditioned a matrix is by means of a 'matrix condition number'—a number that gets larger as the matrix gets closer to singularity. This index is commonly defined as the ratio of the matrix's largest to smallest singular value:

$$\text{matrix condition number} = \frac{\text{svd}(l)}{\text{svd}(r)} \qquad (7)$$

To describe the degree of ill-conditioning in our test data arrays, an analogous condition number based on sums of squares attributable to the largest vs. the smallest systematic factor is defined as follows:

$$\text{array condition number} = \frac{\sqrt{\text{FSS}(1)}}{\text{FSS}(R) - \text{FSS}(R-1)} \qquad (8)$$

where FSS(R) is the Fitted Sum of Squares for the R-dimensional solution. To understand this formulation, consider that the Rth singular value when squared, represents the increment in the fitted sum of squares obtained by adding dimension R. The Rth FSSQ (Fitted Sum of Squares) represents the cumulative fitted sum of squares obtained by using dimen-

Table 1
Measures of array ill-conditioning for the test data sets

	Proportion of (fitted) mean square contributed by including the least important systematic factor	Array condition number, i.e., $[\mathrm{FSS}(1)/(\mathrm{FSS}(R)-\mathrm{FSS}(R-1))]^{1/2}$
PP1	0.000175	74.5
PP2	0.000258	54.8
Fluorescence data (4d)	0.0000151	256.8
Fluorescence data (3d)	0.0000711	118.4

sion 1 through R. To get the increment in fitted sum of squares resulting from dimension R, the difference between $\mathrm{FSSQ}(R)$ and $\mathrm{FSSQ}(R-1)$ is calculated. Since the standard definition of matrix condition number uses singular values rather than squared singular values, we take the square roots of FSSQ to obtain an equivalent measure of ill-conditioning.

In an error free synthetic dataset, R is the rank of the array (i.e., the minimum number of (trilinear) outer products needed to exactly fit the array); for synthetic data with error added, and for real data, R is taken to be the rank of the *systematic or fitted part* of the array. As with the conventional index of the ill-conditioning of the matrix, this index of ill-conditioning reflects both the amount of collinearity of factors and inequality of magnitude of the factors, and does so in terms of the relationships among the patterns of factor contribution in the array as a whole, rather than in terms of relationships among loadings in individual modes. Such an index does not appear to have been used previously, but it appears to be a natural generalization from two-way methods. This array condition number does not, however, reflect any 'difficulty' in the problems that arises in the three way case from weakness of the characteristics determining rotational uniqueness. The array condition numbers for the two synthetic datasets and for the real fluorescence data are given in Table 1. It is apparent that the contributions of the different ranks to the FSS are remarkably different, demonstrating that these test problems are indeed ill-conditioned. The ill-conditioning of the real fluorescence data is particularly severe.

The inspection of singular values confirms that these are 'difficult' factorization problems. If the last singular value is small in two of the three three-factor matrices, then these matrices are almost singular and the problem will thus be difficult. Similarly, occurrence of three small singular values indicates near non-identifiability in four-factor problems. Where two (or more) of columns are nearly proportional in any mode, then the axis orientation for that pair of columns is only weakly determined, and it may be difficult for the algorithm to establish the correct factor axes for the subspace spanned by those two (or more) factors. Table 2 shows the singular values for all three problems. It can be seen that the ratio of largest and smallest singular values for PP1 exceed 10 for both modes **A** and **B** and exceeds 25 in mode **C**. For PP2, the condition index for mode **A** is 37, for mode **B** is 38, and for mode **C** is 38. For the fluorescence data, the condition index for mode **A** is small, but for mode B, the value is 51 and for mode **C**, it is extremely large ($>33{,}000$). The singularity of the

Table 2
The singular values of factor matrices **A**, **B**, and **C** for the PP1, PP2, and fluorescence data sets

Singular value number	PP1			PP2			Fluorescence		
	A	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
1	4.53	5.07	2.55	4.77	5.34	2.19	6.06	7.11	3.09
2	1.83	2.04	0.55	2.89	3.22	1.03	4.46	3.94	0.21
3	0.35	0.36	0.099	0.95	1.05	0.37	4.42	0.92	0.011
4	–	–	–	0.13	0.14	0.058	1.96	0.14	9.2E − 5

C-mode factor matrix is not alarming. From the prior study of these data [11], it is clear that no two of the columns of matrix **C** are proportional in at least the three-factor solution. The reason of the singularity has not been investigated; it may well be of 'chemical' origin. Inequalities in factor sizes can also cause ill-conditioning of the array, but which loading matrices reveal this inequality depends on the scaling conventions. Two of the modes have been standardized to have unit mean squares in each column; the third mode then reflects the relative size of the different factors, as well as any collinearity. This standardization contributed to the large condition index of the **C** mode of the fluorescence data. Note that since the scale of all three modes is incorporated into the size of the rank − 1 (three-way) arrays of factor contributions to fitted array itself, this array conditioning index will be invariant across different scaling conventions for the individual loading matrices.

5. Test results

5.1. Problems in performance comparison

It would be possible to pick one test case and analyze it with the different programs on the same computer. Although such timings would give precise numerical values, they may not be representative. The interactions between the program, the data, the compiler, and the operating system are complex and further complicate any attempts at precise comparisons. In addition, the performance of any program for a small data set may be different from the performance for large sets of data, and this difference may be much larger with some programs than with others.

Further problems with any attempt at precise comparisons include the following.

(1) Omitting the non-negativity constraint (HL-PARAFAC, DTDMR) 'improves' the fit in those cases where noise tends to generate small negative values in some of the factors, but the positivity constraints can provide an important improvement in the strength of the uniqueness for a given dataset, and hence can greatly aid in the rate of convergence toward the unique solution. Should two programs be compared in terms of their unconstrained or constrained performance (a particularly difficult comparison issue when one program allows positivity constraints and another does not)?

(2) The presence of missing values in the acetate data caused two slices of the data array to be omitted from the analysis with HL-PARAFAC (FORTRAN) and with DTDMR. This put these two programs at a disadvantage when analyzing the real data, because they could not use information from the incomplete levels that was included for other programs and which made the problem easier by reducing factor collinearity and near-proportionality.

(3) It may be problematic to compare the rate of convergence in weighted vs. unweighted fitting, since changes in weighting patterns can, in effect, change the degree of ill-conditioning of a dataset. The importance of weighting depends strongly on the data (how the low/high weight values are distributed) but it is difficult to illustrate this with a few test results.

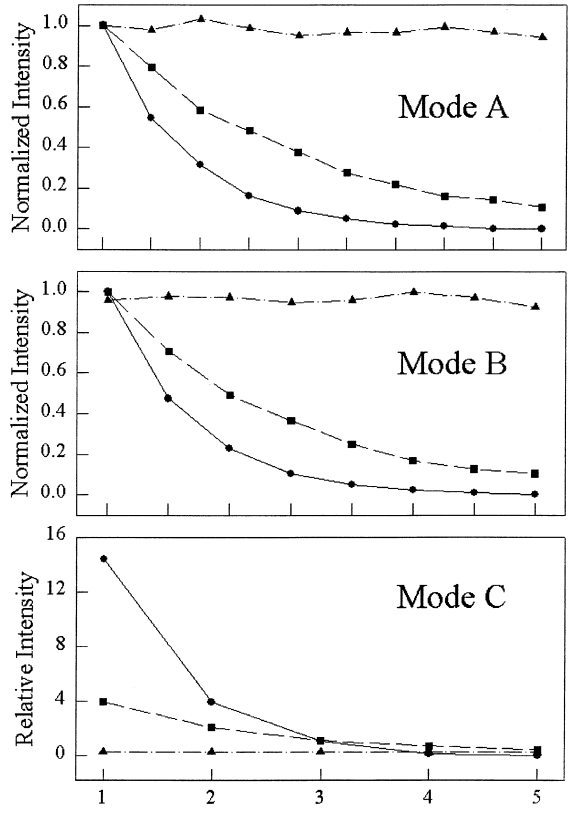

Fig. 3. Plot of the PMF3 resolution of the test data set PP1. Factors 1, 2, and 3 are drawn with a triangle, square, and circle.

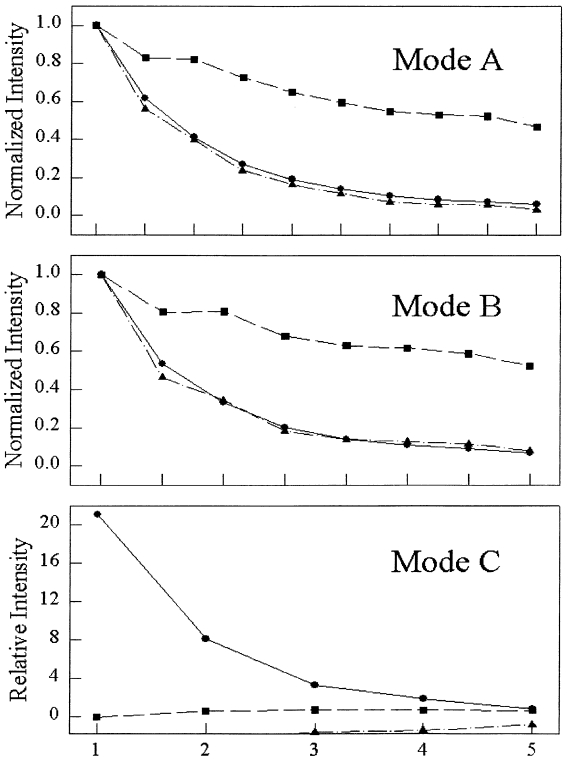

Fig. 4. Plot of the DTDMR resolution of the test data set PP1. Factors 1, 2, and 3 are drawn with a triangle, square, and circle.

Perhaps the most important difficulty in comparison of performance of different algorithms is that the numerical values used for convergence criteria have a strong influence both on the time taken and on the goodness-of-fit achieved. A fair comparison should report on the balance of these two properties, and/or systematically compare iteration counts at equivalent levels of fit (and vice versa).

Our main objectives in this article are more straightforward. They are: (1) determine how well the algorithms work in the presence of ill-conditioning, and to assess the extent to which the results obtained by different algorithms are equivalent (i.e., do the algorithms solve all, or only some. of these 'difficult' test cases, and if only some, do all the algorithms fail on the same test cases) and (2) because speed of the algorithms varied greatly, we can also give rough order-of-magnitude comparisons of the number of iterations that the programs required to solve these difficult cases.

5.2. Results for the synthetic data sets

All of the least-squares algorithms managed to solve these test problems from at least some starting positions, *when they were pushed to full convergence*, but this sometimes required many hundreds or even thousands of iterations, and could only be accomplished for some programs by resetting the convergence criteria to be much more strict than the default values provided in the program. At convergence, all the least squares solutions were identical, as would be expected. The closed-form algorithm DTDMR successfully recovered the factors for PP2, and for the fluorescence data when four factors were extracted, but did not give a reasonable three-factor approximation, and did not recover all of the factors built into PP1.

Figs. 3 and 4 present the three-factor resolution of the data set PP1. Fig. 3 gives the PMF3 solution

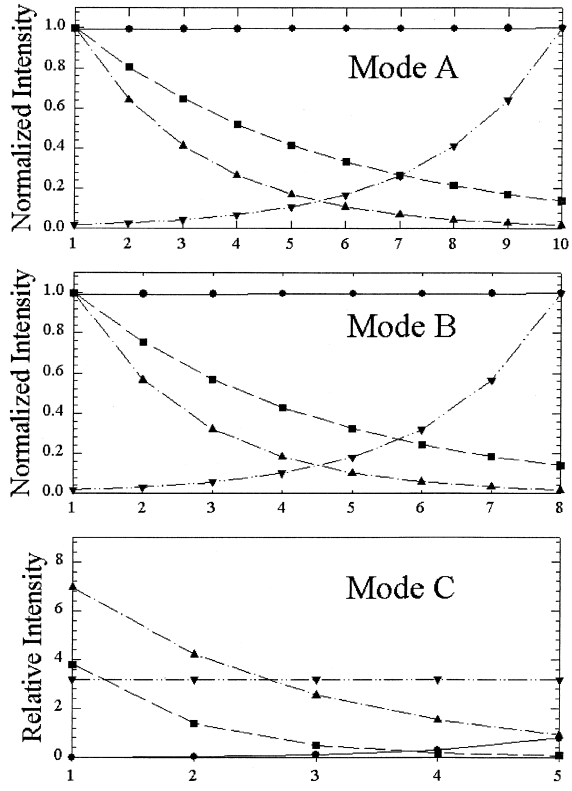

Fig. 5. Plot of the resolution of the test data set PP2 by either of the programs PMF3 or TPALS.

which represents the very similar solutions obtained with PMF3, HL-PARAFAC, and TPALS. Fig. 4 shows the DTDMR solution. It is clear that the direct decomposition solution (DTDMR) is different from that obtained by the other approaches (Fig. 4). Apparently, DTDMR has failed in this case; the MSE (mean squared error) value for the three-factor fit by DTDMR is *larger* than the MSE-value obtained by DTDMR with only two factors.

The four-factor test case PP2 was solved by all four techniques. Initial results showed that PMF3 and TPALS solutions were identical (see Fig. 5) but the HL-PARAFAC solution was not quite as good. It was necessary to tighten the convergence criteria to prevent premature convergence and over 2000 iterations were needed for even approximate convergence. The result (Fig. 6) was still not fully converged, and only becomes identical to Fig. 5 after setting the convergence criterion even more tightly and running additional iterations. This demonstrates the importance of full convergence. The solution for DTDMR (Fig. 7) was approximately correct, but not as close to the true loadings as the least squares solutions. The differences are most observable in the C mode. In a practical problem where the solution was not known a priori, one would probably accept all four results. The difficulty of the problem (which may result from weakly determined uniqueness in addition to array ill-conditioning) caused the differences in running times to be large. Typically, the timings were spaced with a factor of 10.

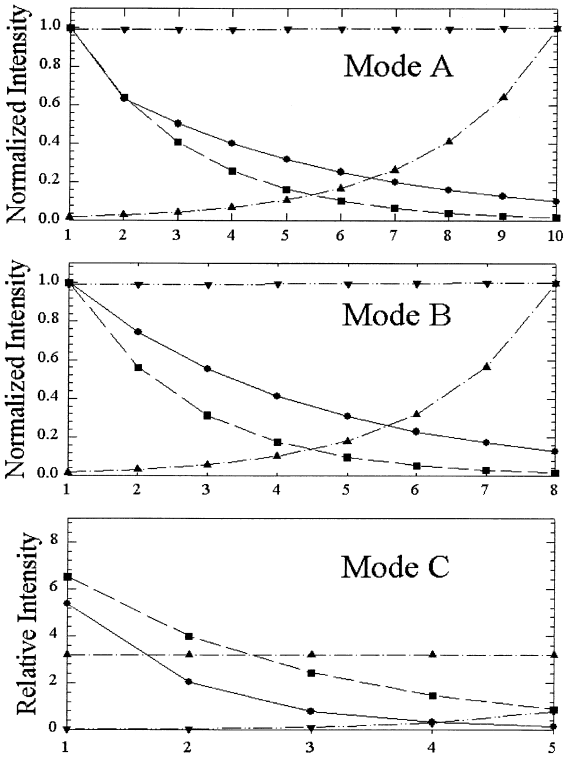

Fig. 6. Plot of the HL-PARAFAC resolution of the test data set PP2.

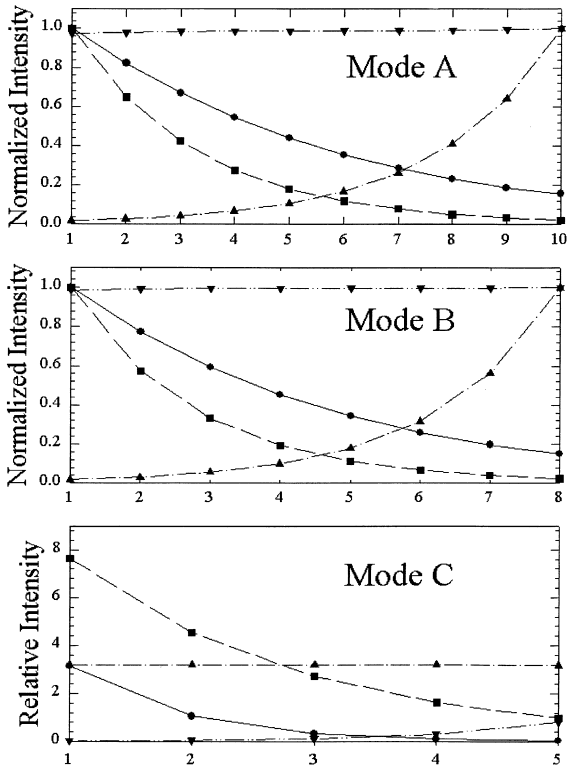

Fig. 7. Plot of the DTDMR resolution of the test data set PP2.

5.3. Results for the fluorescence data

It was known from earlier work [11] that this data contained three components. It was then initially analyzed with three factors. TPALS and PMF3 produced results essentially identical to those reported previously [11] (shown in Fig. 8). The main component has an excitation l_{max} at 302 nm and an emission l_{max} at

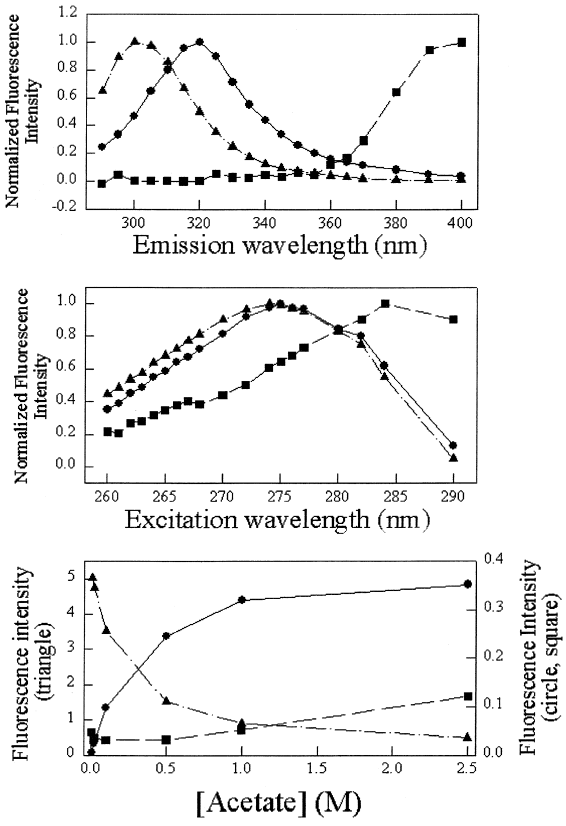

Fig. 8. Three factor solution for fluorescence data of dimensions $20\times19\times6$ from NAYA in acetate obtained by either program PMF3 or TPALS. Factors corresponding to the normal NAYA, NAYA acetate, and an impurity are plotted with a triangle, circle and square, respectively.

274 nm. The second component is observed with an excitation l_{max} at 320 nm and an emission l_{max} at 275 nm. These two components are assigned as the normally solvated side chain and side chain hydrogen-bonded to the added proton acceptor. The weak third component has a broad excitation spectrum peaking near 290 nm and a broad emission maximum peaking near 400 nm; it increases with increasing concentration of acetate. These characteristics of the third component suggest that it is impurity, primarily with the concentration of acetate.

The program DTDMR needed a different set-up for this problem because it does not have missing-data capability. Thus, because of the presence of some missing values at the two longest excitation wavelengths, all data for these two excitation wavelengths had to be omitted. It is worth noting that reduction in the data may have placed DTDMR at a slight disadvantage, since two of the factors are more similar/collinear in the reduced dataset. Due to an oversight, this handicap was also imposed on HL-PARAFAC. Although HL-PARAFAC can handle missing values, this option was overlooked and consequently the reduced dataset used for DTDMR was also used to test HL-PARAFAC. In addition, a non-negativity constraint was also omitted and constant weighting was applied to all points.

Fig. 9 gives the three-factor solution obtained with DTDMR. Note that the third component of the three factor resolution is clearly different from that of PMF3 and TPALS. DTDMR has missed the low-in-

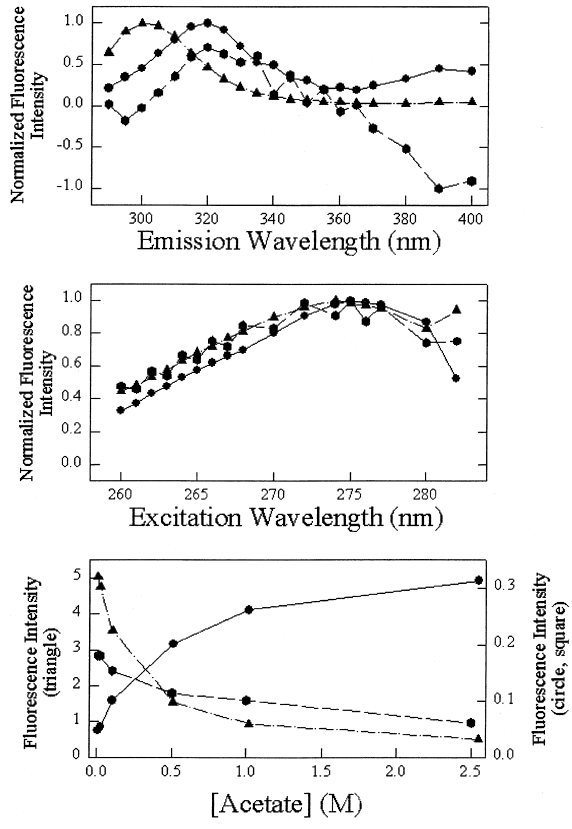

Fig. 9. The three factor DTDMR solution to the fluorescence data of dimensions $20\times17\times6$ from NAYA in acetate. Factors corresponding approximately to the normal NAYA, NAYA–acetate complex, are plotted with a triangle and a circle; the third factor which is not interpretable, is plotted with a hexagon, respectively.

tensity impurity component. Since DTDMR is not iterative, there is no question of a 'bad starting point'.

Fig. 10 gives the three-factor solution originally found by HL-PARAFAC. The loadings oscillate from point-to-point in the modes **A** and **B**. From this result, one would be tempted to say that this program has also failed to recover the solution. However, this is not, in fact, the case. One run by PMF3 on the complete data set had resulted in a strange-looking solution which was diagnosed as a local optimum (its MSE value was three times the optimal MSE). The result by HL-PARAFAC resembled this local optimum. However, upon further investigation it was found that unweighted PMF also always finds the oscillatory solution when applied to the reduced data set. Thus, this is not a case of finding a local mini-

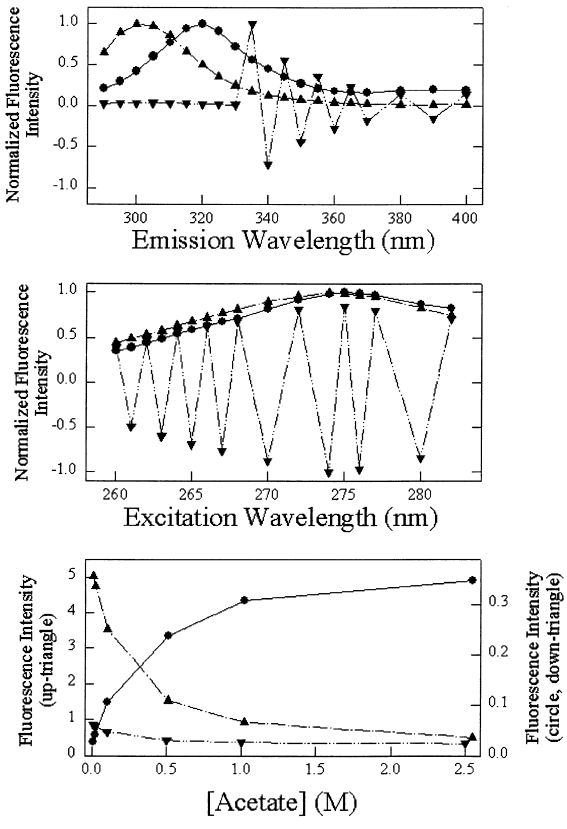

Fig. 10. The three factor HL-PARAFAC solution to fluorescence data of dimensions $20\times17\times6$ from NAYA in acetate. Factors corresponding to the normal NAYA, NAYA–acetate complex, and an oscillatory noise in the data are plotted with an up-pointing triangle, circle and down-pointing triangle, respectively.

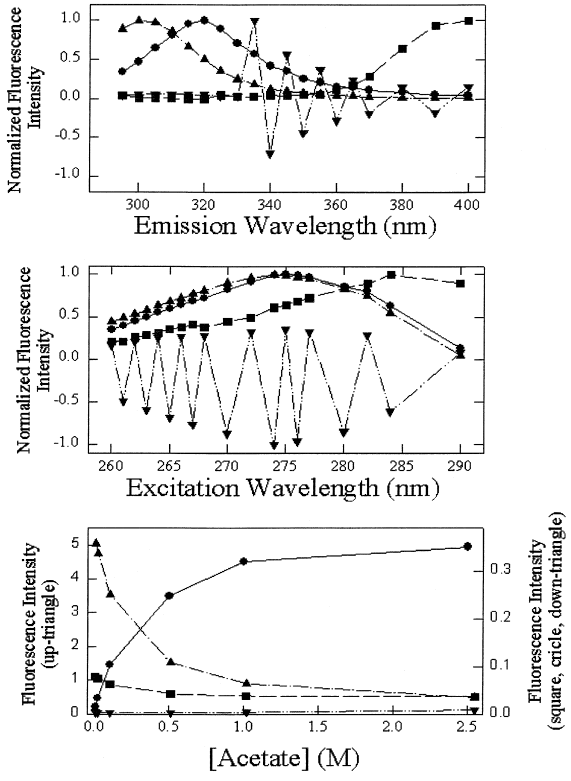

Fig. 11. Resolution of $20\times19\times6$ dimensioned fluorescence data of NAYA in acetate into four components by either program PMF3 or TPALS. The normal NAYA, NAYA–acetate complex, impurity, and oscillatory noise have been plotted with up-pointing triangle, circle, square, and down-pointing triangle, respectively.

mum, but rather reflects the change in the problem when two slices are removed from the data set. It appears that there is no local unweighted $20\times17\times6$ solution corresponding to the weighted full-array solution.

These results suggested that perhaps there would be a fourth factor in this data set. With four factors in the data but only three being extracted, different optimum solutions can be found (from some starting positions) in which factors 1, 2, and 4 are recovered, but factor 3 is not when slightly different data sets were employed. Four-factor runs (without the non-negativity constraints) produced the results shown in Figs. 11 and 12. The four-factor results obtained with PMF3 and TPALS were again identical; they are shown in Fig. 11. The programs HL-PARAFAC and DTDMR also succeeded, producing almost identical

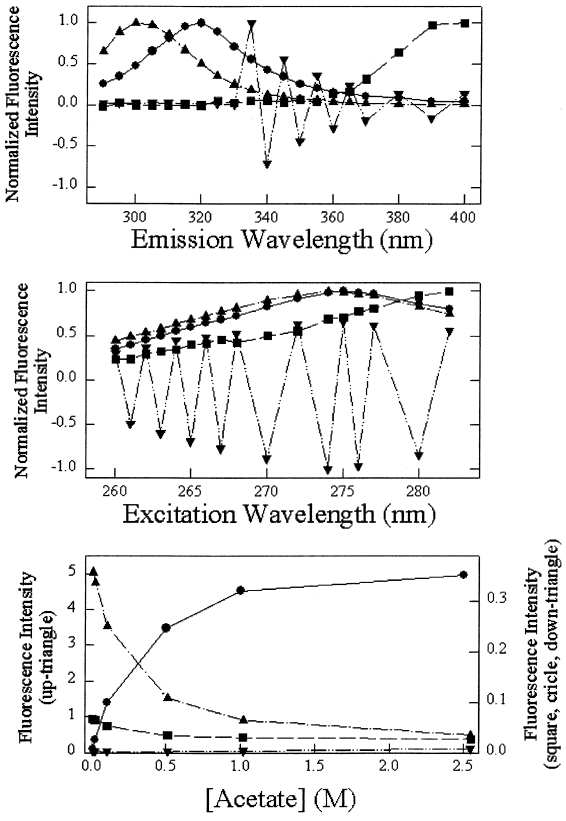

Fig. 12. Resolution of fluorescence of NAYA in acetate into four components by the program HL-PARAFAC when using the partial array of dimensions $20 \times 17 \times 6$. The normal NAYA, NAYA–acetate complex, impurity, and oscillatory noise have been plotted with up-pointing triangle, circle, square, and down-pointing triangle, respectively.

results. The HL-PARAFAC results are shown in Fig. 12. The small variations of the HL-PARAFAC results from the PMF3/TPALS solution is again attributable to omitting two slices from the data array.

In all these solutions, the fourth factor shows an oscillating behavior in the **A** and **B** modes. In practical work such a result could be easily dismissed as 'numerical instability.' However, in these data, this result is 'real'. Once it had been found by the programs, it could also be spotted visually in the data array. In the affected regions, there was a checkerboard pattern in which the values in the 'white' cells were larger than the values in the 'black' cells. The differences were a few percent of the affected data values. This result is a nice demonstration of the power of three-way factor analysis. It may reveal unexpected features from a data set without preconceptions or operator decisions. The reason behind factor four's oscillatory behavior is not known. Since the oscillation is periodic in data points (not in wavelength), it suggests that the reason might be the computations performed during data preprocessing.

Our initial runs with HL-PARAFAC did not find the best-fitting three-factor solution because we happened to use the reduced data set. This was not initially diagnosed until further study was made to determined if this was a local minima problem. It was found that the problem was really the loss of the information in the portion of the spectra containing missing values. Thus, the ultimate solution was the same as those obtained with the other least-squares approaches. However, the three-factor DTDMR solution differed from the three-factor LS solutions in a more fundamental fashion: its third factor did not resemble either the impurity or the oscillatory component.

6. Discussion

The foremost result of this study is that the three programs based on the least squares approach seem to agree well. However, this agreement was only obtained by adjusting the convergence criteria so that a sufficient number of iterations was guaranteed with all programs. An unwary user might easily obtain distorted results if not paying attention to the convergence. One may be certain that the minimum has been reached with the desired precision if one runs a few representative test cases with a very strict convergence criterion, so that the total iteration count is several times larger that the value which was first obtained. If these added iterations produce virtually no change of the solution, then the convergence may be assumed.

Multiple solutions may lead to surprises and to false conclusions. Thus, one lesson from this study is that it is essential to routinely use multiple starting points. In the tests in this study, the 'correct' solution also was the global minimum. However, this should not be assumed. It seems possible that the 'correct' solution need not be the one with the lowest mini-

mum. If error values for the data points (and resulting weights) are correct, then the lowest minimum would be expected to be the 'correct' one. However, if these error estimates are not correct (or the data are something different than they should be, for example, include some systematically biased values), then problems may arise. Such situations are not uncommon. As an example, assume that a non-robust analysis is performed and there is an outlier or a few outliers (without proper error values). Then there may be two competing solutions: (1) one with a bad fit for the outliers, and one factor representing a weak but true physical component, or (2) good fit of the outlier(s), using up one factor, and thus no factor available for representing the weak true component. Either alternative (1) or (2) may have the lower Q value. It is expected that one could construct an example, but that is really part of a further study.

The performance of the DTDMR program was different from the others. Although DTDMR was clearly the fastest algorithm, it did not solve all the test problems. Also, this program does not have the analysis options that the least-squares based programs have. For example, it is impossible to include individual weighting of data points or constraints on the solutions to be non-negative. Most seriously, one does not know what are the mathematical properties of the DTD solution for a model which contains noise or distortions from the ideal error-free model.

One important aspect of these programs is speed of execution. As described earlier, we are not inclined to present precise quantitative speed comparisons because they could easily be misleading. During this work it was confirmed that all three LS-based programs are equally fast when solving 'easy' cases where no collinearity is present, all factors are of similar weight or importance and uniqueness is strongly determined (all columns of factor loadings are far from proportionality). Typically, 20 or 30 steps were needed in such cases. Regarding our difficult test cases, which take much longer to solve with any of the programs, one may roughly summarize the differences among the LS programs as follows: HL-PARAFAC is the slowest in terms of the number of iterations needed. The TPALS program is faster by perhaps one order-of-magnitude. Finally, PMF3 is the fastest of the three and again there can be a margin of up to a factor of 10 or even more. The differences in computing time are similar but slightly smaller. Although each of the more sophisticated steps in PMF3 take longer to compute than the simpler ALS steps, it appears that the increased efficiency by far outweighs the increased workload per step, at least with moderate sized data arrays such as used here (problems with less than 300 unknown values to be determined). On a 486 PC computer, the running times for the test problems were typically as follows: PMF3—tens of seconds, TPALS—minutes, and HL-PARAFAC (F77)—tens of minutes.

The question then arises as to why PMF3 is so much faster than HL-PARAFAC in some tasks? It appears there are two main reasons and they may have different importance in different applications. The reasons are that PMF changes all variables simultaneously, and PMF sometimes avoids algorithmic 'swamps' because of its regularization features.

Recently, Paatero [13] discussed the question of changing all variables simultaneously as compared with changing discrete groups of variables one at a time. From the theory of optimization algorithms, it is well known that not changing all variables simultaneously may lead to slow algorithms whenever the eigenvalues of the fit matrix (the matrix of the 'normal equations' or the 'information matrix' of the variables) differ widely in their order of magnitude. Preliminary results suggest that the ratio of largest to the smallest significant eigenvalues can range (in chemometrics problems) from less than 10 to more than 10 000. The resulting number of steps may range from tens to tens of thousands if all variables are not evaluated together. Paatero [13] presents a weighted two-way example in which PMF converged in four steps where ALS needed more than 50 steps. This example is a one factor problem so degeneracy cannot play a role.

Degenerate solutions [21] and algorithmic 'swamps' [22] can severely lengthen the convergence time and an approach that avoids or reduces such problems will as a result have a speed advantage with some datasets. In degenerate solutions, positive and negative contributions from different factors almost cancel each other so that arrays of factor contributions arising from a single factor can have much larger norms than the original array $\mathbf{X}$. 'Swamps' are quasi-degenerate intermediate stages on the way to convergence, and may severely slow progress until

the program finds its way out. Such problems were not systematically investigated in this study; they did not belong to the original study plan. However, they may provide added reasons why some approaches were faster than others. Degeneracy and swamps are not a problem when all three loadings matrices are constrained to be non-negative since then there cannot be cancellations. The avoidance of 'swamps' may be one of the ways non-negative algorithms speed convergence. Another way of avoiding 'swamps' may be the use of some kind of regularization. It has been reported that applying regularization in the form of 'ridge regression' in ALS fits tends to prevent 'swamps' in the middle stage of iteration [22]. Regularization could have a similar effect. Since the sum-of-squares of factor elements is large in a degenerate solution, the penalty function involved with regularization (implicit in ridge regression and more explicit in the object function used in PMF3) imposes a substantial penalty on such solutions. In this way, regularization guides successive iterations in directions that tend to avoid 'swamps' [22].

However, if the true solution is non-degenerate, then the factors must be non-degenerate at least during the final stage of convergence. Nonetheless, it was found that the final convergence of the ALS-based techniques was still slow in our test cases. It seems that the speed advantage of PMF3 in these test cases may be due more to the fact that all variables are changed together. Of course, there may be some degeneracy and related slowness in mid-iteration in these test cases when negative values were allowed.

The performance of the TPALS algorithm may be understood as follows: during part of each iteration, this algorithm also changes all variables simultaneously. Hence, it uses more powerful tools than HL-PARAFAC and is expected to perform better. However, the movements of different factor elements are not entirely free because they are computed within a framework that builds on ALS substeps. A step by TPALS is not expected to be optimal whereas PMF3 computes a step that is optimal in first order approximation. Thus, TPALS may be slower than PMF3.

6.1. Availability of the programs

The program PMF3 was written in FORTRAN90. Normally it is only distributed in compiled form, as .exe files. Contact Pentti Paatero (pentti.paatero @helsinki.fi) about the availability of the program for various platforms. For PC computers (486 or Pentium), free evaluation licenses are available for testing the program for a period of 6 months.

The PARAFAC programs are available from Richard Harshman (harshman@uwo.ca) at a nominal cost, or free by FTP (internet File Transfer Protocal). The TPALS program is available from Robert Ross (rtr + @osu.edu). Note, however, that the IMSL library must be available in order that the TPALS code can be used without modification.

Acknowledgements

This work was supported in part by the US Environmental Protection Agency under grant R822482. The report has not been reviewed by the Agency, and the views and ideas expressed in this report are those of the authors and do not necessary reflect the views and policies of the Environmental Protection Agency nor does the mention of trade names or commercial products constitute an endorsement for use. Financial support to Pentti Paatero from the Vilho, Yrjö, and Kalle Väisälä Foundation and to Richard Harshman from the Natural Science and Engineering Research Council of Canada are gratefully acknowledged.

References

[1] J.D. Carroll, J.J. Chang, Analysis of individual differences in multidimensional scaling via *N*-way generalization of 'Eckart–Young' decomposition, Psychometrika 35 (1970) 283–319.

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

[3] R.A. Harshman, M.E. Lundy, The PARAFAC Model for Three-way Factor Analysis and Multidimensional Scaling, Research Methods for Multimode Data Analysis, Praeger, New York, NY, 1984, pp. 122–215.

[4] R.A. Harshman, M.E. Lundy, Data preprocessing and the extended PARAFAC model, Research methods for multimode data analysis, Praeger, New York, NY, 1984, pp. 216–284.

[5] R.A. Harshman, M.E. Lundy, PARAFAC: Parallel Factor Analysis, Computational Statistics and Data Analysis 18 (1994) 39–72.
[6] J.B. Kruskal, Three-way arrays: rank and uniqueness of trilinear decompositions, with applications to arithmetic complexity and statistics, Linear Algebra and Its Applications 18 (1977) 95–138.
[7] Law et al., Research Methods for Multimode Data Analysis, Chap. 6, Praeger, New York, NY, 1984, pp. 161–162.
[8] S.E. Leurgans, R.T. Ross, Multilinear models: applications in spectroscopy, Stat. Sci. 2 (1992) 289–319.
[9] S.E. Leurgans, R.T. Ross, R.B. Abel, A decomposition for three-way arrays, SIAM. J. Matrix Anal. Appl. 14 (1993) 1064–1083.
[10] R.T. Ross, S. Leurgans, Component resolution using multilinear models, Methods in Enzymology 246 (1995) 679–700.
[11] J.K. Lee, R.T. Ross, S. Thampi, S. Leurgans, Resolution of the properties of hydrogen-bonded tyrosine using a trilinear model of fluorescence, J. Phys. Chem. 96 (1992) 9158–9162.
[12] P. Paatero, Least squares formulation of robust non-negative factor analysis, Chemom. Intell. Lab. Syst. 37 (1997) 15–35.
[13] P. Paatero, A weighted non-negative least squares algorithm for three-way 'PARAFAC' factor analysis, Chemom. Intell. Lab. Syst. 38 (1997) 223–242.
[14] B.E. Wilson, E. Sanchez, B.R. Kowalski, An improved algorithm for the generalized rank annihilation method, J. Chemom 3 (1989) 493–498.
[15] E. Sanchez, B.R. Kowalski, Tensorial resolution: a direct trilinear decomposition, J. Chemom. 4 (1990) 29–45.
[16] Y. Zeng, P.K. Hopke, A new receptor model: a direct trilinear decomposition followed by a matrix reconstruction, J. Chemom. 6 (1992) 65–83.
[17] P. Paatero, U. Tapper, Analysis of different modes of factor analysis as least squares fit problems, Chemom. Intell. Lab. Syst. 18 (1993) 183–194.
[18] C.L. Lawson, R.J. Hanson, Solving least squares problems, Prentice-Hall, Englewood Cliffs, NJ, 1974.
[19] R. Bro, S. De John, A fast non-negativity constrained least-squares algorithm, J. Chemom. 11 (1997) 395–401.
[20] P. Paatero, User's guide for the programs PMF2 and PMF3 (1997) (Unpublished).
[21] J.B. Kruskal, R.A. Harshman, M.E. Lundy, How 3-MFA Data Can Cause Degenerate PARAFAC Solutions, Among Other Relationships, Multiway Data Analysis, North-Holland, 1989, pp. 115–122.
[22] W. Rayens, W. Mitchell, Two-factor degeneracies and a stabilization of PARAFAC, Chemom. Intell. Lab. Syst. 38 (1997) 173–181.