Data Mining by NonNegative Tensor Approximation

Rodrigo Cabral Farias, Pierre Comon, Roland Redon

DATA MINING BY NONNEGATIVE TENSOR APPROXIMATION

Rodrigo Cabral Farias, Pierre Comon[*]

GIPSA-lab, CNRS UMR5216,
11, rue des Mathématiques, BP 46
38402 Saint Martin d'Hères, France
{rodrigo.cabral-farias, pierre.comon}@gipsa-lab.grenoble-inp.fr

Roland Redon

Lab. PROTEE, EA3819
Université de Toulon,
83957 La Garde, France
roland.redon@univ-tln.fr

ABSTRACT

Inferring multilinear dependences within multi-way data can be performed by tensor decompositions. Because of the presence of noise or modeling errors, the problem actually requires an approximation of lower rank. We concentrate on the case of real 3-way data arrays with nonnegative values, and propose an unconstrained algorithm resorting to an hyperspherical parameterization implemented in a novel way, and to a global line search. To illustrate the contribution, we report computer experiments allowing to detect and identify toxic molecules in a solvent with the help of fluorescent spectroscopy measurements.

Index Terms— muti-way ; tensor ; CP ; low-rank ; approximation ; nonnegative ; line search ; fluorescence ; HAP

1. INTRODUCTION

Tensor decompositions are emerging as an alternative tool for machine learning and data mining. They permit to (but are also limited to) find multilinear dependences between hidden variables. For instance, they have been already used in the field of blind source separation, but less used as a deterministic tool for mining data.

In the present framework, we shall concentrate on real nonnegative data, which are organized in the form of a multi-way array. Several applications have already been identified, which fit in this framework, including machine learning [1], internet connection analysis [2, 3], hyperspectral imaging [4], spectrography [5], as well as a few others [6, 7]. Our description will focus on 3-way fluorescence spectrography, but could of course apply to arrays of higher order and to other application fields where nonnegativity is important.

Notation Vectors will be distinguished from scalar numbers by bold lowercases, *e.g.* $\mathbf{v}$; matrix and tensor arrays will be denoted by bold and calligraphic uppercases, respectively, *e.g.* $\mathbf{A}$ and $\boldsymbol{\mathcal{T}}$. Array entries are real scalar numbers, and are denoted by plain letters, *e.g.* v_i, A_{jk}, $T_{\ell mn}$, for the previous vector, matrix or tensor examples. Vectors will be represented by column arrays of coordinates. $\mathbf{A} \boxdot \mathbf{B}$ will denote the element-wise (Hadamard) product between two matrices of same dimensions, $\otimes$ will denote the outer product between arrays, *e.g.* $\mathbf{v} \otimes \mathbf{A} \otimes \boldsymbol{\mathcal{T}}$ is the object whose coordinates are $v_i\, A_{jk}\, T_{\ell mn}$. These are the notations assumed in *e.g.* [8].

2. TENSOR APPROXIMATION

2.1. Tensor decomposition

The objective is to decompose a 3-way data array $\mathcal{Y}$ in a finite sum of outer products of vectors (also known as tensor products). For example, if we consider an array of real values of size $I \times J \times K$, then we want to find the vectors $\mathbf{a}_r = [A_{1r}, \cdots A_{Ir}]^\top \in \mathbb{R}^I$, $\mathbf{b}_r = [B_{1r}, \cdots B_{Jr}]^\top \in \mathbb{R}^J$ and $\mathbf{c}_r = [C_{1r}, \cdots C_{Kr}]^\top \in \mathbb{R}^K$ with $r \in \{1, \cdots, R\}$ for which

$$Y_{ijk} = \sum_{r=1}^{R} A_{ir}\, B_{jr}\, C_{kr}, \tag{1}$$

or equivalently using the outer product

$$\boldsymbol{\mathcal{Y}} = \sum_{r=1}^{R} \mathbf{a}_r \otimes \mathbf{b}_r \otimes \mathbf{c}_r. \tag{2}$$

If R is the smallest number of components necessary to exactly decompose $\boldsymbol{\mathcal{Y}}$, then (2) is called the canonical polyadic (CP) decomposition of $\mathcal{Y}$; see [5, 6, 8] and references therein.

Most of the time, $\boldsymbol{\mathcal{Y}}$ cannot be exactly decomposed as in (2) for the desired R, either because the actual model behind $\boldsymbol{\mathcal{Y}}$ is not multilinear (linear in $\mathbf{a}_r$, $\mathbf{b}_r$ and $\mathbf{c}_r$ individually), or because the data are noisy. In this case, for a given R, the goal is to find $\{\mathbf{a}_r, \mathbf{b}_r, \mathbf{c}_r,\ 1 \le r \le R\}$ that best approximate the original array, for example, by solving the following minimization problem

$$\begin{array}{ll} \text{minimize} & \Upsilon = \left\| \boldsymbol{\mathcal{Y}} - \sum\limits_{r=1}^{R} \mathbf{a}_r \otimes \mathbf{b}_r \otimes \mathbf{c}_r \right\|_2^2 \\ \text{w.r.t.} & \mathbf{A} = [\mathbf{a}_1, \cdots \mathbf{a}_R] \in \mathbb{R}^{I\times R}, \\ & \mathbf{B} = [\mathbf{b}_1, \cdots \mathbf{b}_R] \in \mathbb{R}^{J\times R}, \\ & \mathbf{C} = [\mathbf{c}_1, \cdots \mathbf{c}_R] \in \mathbb{R}^{K\times R}, \end{array} \tag{P.1}$$

[*] This work has been funded by the European Research Council under the 7th Framework Programme FP7/2007–2013 Grant Agreement no. 320594.

where $\| \cdot \|_2$ is the Frobenius norm: $\|\boldsymbol{\mathcal{T}}\|_2 = \sqrt{\sum_{i,j,k} T_{ijk}^2}$. Matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are commonly called loading matrices [5].

Several approaches may be found in the literature, which search for a solution of problem (P.1). They range from simple alternating algorithms like the well-known alternating least squares (ALS) [5] to all-at-once minimization procedures like gradient descent, conjugate gradient and quasi-Newton algorithms [9, 10, 4, 11]. Interestingly, a solution for problem (P.1) does not necessarily exist [12], and, as a consequence, searching for it is an ill-posed problem.

2.2. Nonnegative CP and quadratic parameterization

A particular case for which problem (P.1) is well-posed is when the vectors in $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are nonnegative. In this case, it can be shown that a solution exists [13]. Not only this instance of the problem is well-posed, but it also naturally appears in many applications such as fluorescence spectrometry, as pointed out in Section 1. For these reasons, in what follows, we focus on the nonnegative version of problem (P.1), which may be stated as follows, for a given nonnegative tensor $\boldsymbol{\mathcal{Y}}$:

$$\begin{array}{ll} \text{minimize} & \Upsilon = \left\| \boldsymbol{\mathcal{Y}} - \sum_{r=1}^{R} \mathbf{a}_r \otimes \mathbf{b}_r \otimes \mathbf{c}_r \right\|_2^2 \\ \text{w.r.t.} & \mathbf{A}, \mathbf{B} \text{ and } \mathbf{C}, \\ \text{subject to} & \mathbf{A} \succeq \mathbf{0}, \mathbf{B} \succeq \mathbf{0} \text{ and } \mathbf{C} \succeq \mathbf{0}, \end{array} \tag{P.2}$$

where $\mathbf{F} \succeq \mathbf{0}$ means that all elements of $\mathbf{F}$ are nonnegative.

Common methods solving (P.2) include ALS with active sets [14, p. 170], minimization methods with logarithmic penalty functions [9] and extensions of algorithms used for nonnegative matrix factorization [15], [16] such as ALS, regularized ALS with projection, projected gradient, accelerated proximal gradient, gradient with nonnegative multiplicative update and projected quasi-Newton techniques.

Instead of solving the minimization problem (P.2), an interesting approach studied in [11] consists in reparameterizing the problem with the loading matrices $\mathbf{A}' \in \mathbb{R}^{I \times R}$, $\mathbf{B}' \in \mathbb{R}^{J \times R}$ and $\mathbf{C}' \in \mathbb{R}^{K \times R}$ such that

$$\mathbf{A} = \mathbf{A}' \boxdot \mathbf{A}' = \begin{bmatrix} a_{1,1}^2 & \cdots & a_{1,R}^2 \\ \vdots & & \vdots \\ a_{I,1}^2 & & a_{I,R}^2 \end{bmatrix}, \tag{3}$$

$$\mathbf{B} = \mathbf{B}' \boxdot \mathbf{B}' \quad \text{and} \quad \mathbf{C} = \mathbf{C}' \boxdot \mathbf{C}'.$$

Note that with this reparameterization the original minimization problem becomes unconstrained:

$$\begin{array}{ll} \text{minimize} & \left\| \boldsymbol{\mathcal{Y}} - \sum_{r=1}^{R} (\mathbf{a}'_r \boxdot \mathbf{a}'_r) \otimes (\mathbf{b}'_r \boxdot \mathbf{b}'_r) \otimes (\mathbf{c}'_r \boxdot \mathbf{c}'_r) \right\|_2^2 \\ \text{w.r.t.} & \mathbf{A}', \mathbf{B}' \text{ and } \mathbf{C}', \end{array} \tag{P.3}$$

since the components of the approximation are now nonnegative by construction. As a consequence, this approach does not require the introduction of projection techniques; for example, in [11] a conjugate gradient technique for unconstrained problems is used to solve problem (P.3).

2.3. Unit norm constraint and spherical parameterization

The nonnegativity constraint imposed above allows to guarantee the existence of a solution, but an issue that still persists is the lack of uniqueness of the parameterization. Non uniqueness of the solution comes from multiple ambiguities inherent in the multilinear model (P.1), which persist in (P.3).

The first ambiguity put forward in the literature is that of permutation, since the order in which the R terms are summed has no influence on the result. Actually, this issue is irrelevant in parameter estimation, even if it is important when assessing performance in computer experiments. On the other hand, there remains a scaling ambiguity, induced by the multiplicative aspect within components. In fact, the tensor $\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c}$ can be equally represented by the triplets of vectors $(\mathbf{a}, \mathbf{b}, \mathbf{c})$ or $(\lambda_1 \mathbf{a}, \lambda_2 \mathbf{b}, \mathbf{c}/\lambda_1 \lambda_2)$, for any nonzero numbers $\{\lambda_1, \lambda_2\}$; this indeterminacy is inherent in the definition of tensor spaces [8]. Contrary to permutation, scaling can have an effect on optimization algorithms used to solve (P.2). In particular, it can make iterations continuously move between various scaling-equivalent solutions. In practice, it is useful to fix these ambiguities.

To solve this issue, we can impose that all the columns of $\mathbf{A}$ and $\mathbf{B}$ have a unit $\mathcal{L}_1$ norm, as these matrices are already restricted to have nonnegative elements:

$$\|\mathbf{a}_r\|_1 = 1 \quad \text{and} \quad \|\mathbf{b}_r\|_1 = 1, \quad \text{for } r = \{1, \cdots, R\}. \tag{4}$$

On the other hand, matrix $\mathbf{C}$ is left unconstrained.

These $\mathcal{L}_1$-norm constraints impose the following $\mathcal{L}_2$-norm constraints in the nonnegative reparameterization (P.3):

$$\|\mathbf{a}'_r\|_2^2 = 1 \quad \text{and} \quad \|\mathbf{b}'_r\|_2^2 = 1, \quad \text{for } r = \{1, \cdots, R\}. \tag{5}$$

As a consequence, each column vector of $\mathbf{A}'$ and $\mathbf{B}'$ is constrained to lie on the unit $\mathcal{L}_2$-hypersphere. This can also be imposed by a parameterization, which consists of describing them using polar coordinates, namely by angles. For example, any vector $\mathbf{a}'$ with I components can be parameterized with angles $\boldsymbol{\theta} = [\theta_1 \cdots, \theta_{I-1}]^\top$ as

$$\left[\cos\theta_1, \ \cos\theta_2 \sin\theta_1, \ \ldots, \ \cos\theta_{I-1} \prod_{i=1}^{I-2} \sin\theta_i, \ \prod_{i=1}^{I-1} \sin\theta_i \right]^\top$$

But since these angles can be restricted to belong to the interval $[0, \pi/2]$, one can instead use their tangent $t_i \in [0, \infty[$, since there exists a bijection relating them: $\cos^2\theta_i = (1 + t_i^2)^{-1}$ and $\sin^2\theta_i = t_i^2(1 + t_i^2)^{-1}$. Finally, a generic vector $\mathbf{a} = \mathbf{a}' \boxdot \mathbf{a}'$ with nonnegative entries and unit $\mathcal{L}_1$-norm can be written as

$$\left[\frac{1}{1+t_1^2}, \frac{1}{1+t_2^2}\frac{t_1^2}{1+t_1^2}, \dots \frac{1}{1+t_{I-1}^2}\prod_{i=1}^{I-2}\frac{t_i^2}{1+t_i^2}, \prod_{i=1}^{I-1}\frac{t_i^2}{1+t_i^2}\right]^\top,$$

But it turns out that there is an even simpler and obvious way to do this, which will present significant advantages. In fact consider the simpler parameterization

$$\frac{1}{1+\sum_{i=2}^{I}\tau^2}\left[1,\, \tau_2^2,\, \tau_3^2, \dots,\, \tau_I^2\right],\ \tau_i \in [0,\, \infty[.$$

This parameterization also allows to represent any nonnegative vector on the unit $\mathcal{L}_1$-hypersphere. There is a relationship between $\{t_i\}$ and $\{\tau_i\}$, but it is irrelevant in our discussion. Clearly, there are I possible such parameterizations of the I-dimensional hypersphere, depending on the place where the value 1 is imposed. We assumed the value 1 was set in the first entry, but the reasoning is the same with a different location.

The inconvenience of this solution, which is also shared by tangents t_i, is that all τ_i's need to tend to infinity in order to approach zero in the first coordinate. The consequence is that zero cannot be exactly obtained, and that this parameterization is valid only generically. For this reason, we shall now consider projective coordinates:

$$\frac{1}{\sum_{i=1}^{I}\tau_i^2}\left[\tau_1^2,\, \tau_2^2, \dots,\, \tau_I^2\right]. \tag{6}$$

Clearly, we have now too many parameters, and the parameter vector $(\tau_1, \dots \tau_I)$ corresponds to the same point as $\lambda(\tau_1, \dots \tau_I)$ for any nonzero nonnegative λ. This is precisely what allows not only to reach all possible values, but also to better condition the parameterization because we may now impose $|\tau_i| \in [0, 1]$. In practice, setting $\lambda = |\tau_{max}|^{-1}$, the inverse of the largest $|\tau_i|$, will yield the best conditioning. In the remainder, we refer to this parameterization technique as Parameterization of Positive Orthant Hyperspheres (PAPOH). Then problem (P.3) reduces to:

$$\begin{array}{ll} \text{minimize} & \Upsilon \overset{\text{def}}{=} \left\|\boldsymbol{\mathcal{Y}} - \sum\limits_{r=1}^{R}\boldsymbol{\mathcal{D}}_r\right\|_2^2, \\ \text{w.r.t.} & \boldsymbol{\alpha},\ \boldsymbol{\beta} \text{ and } \boldsymbol{\gamma}, \end{array} \tag{P.4}$$

where $\boldsymbol{\mathcal{D}}_r$ denote decomposable (rank-1) tensors

$$\boldsymbol{\mathcal{D}}_r \overset{\text{def}}{=} \mathbf{a}_r(\boldsymbol{\alpha}_r) \otimes \mathbf{b}_r(\boldsymbol{\beta}_r) \otimes \mathbf{c}_r(\boldsymbol{\gamma}_r),$$

where $\mathbf{a}_r(\boldsymbol{\alpha}_r)$ and $\mathbf{b}_r(\boldsymbol{\beta}_r)$ are parameterized as in (6); vector $(\alpha_{1r}, \dots \alpha_{Ir})^\top$ is denoted by $\boldsymbol{\alpha}_r$,

$$\mathbf{a}_r(\boldsymbol{\alpha}_r) = \frac{1}{\sum_i \alpha_{ir}^2}\left[\alpha_{1r}^2, \dots \alpha_{Ir}^2\right]^\top, \quad \max_i \alpha_{ir}^2 = 1, \tag{7}$$

$\boldsymbol{\alpha}$ is the set of R vectors $\boldsymbol{\alpha}_r$, and a similar notation is used for $\boldsymbol{\beta}_r$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}_r$ and $\boldsymbol{\gamma}$. Here the entries of $\mathbf{c}_r(\boldsymbol{\gamma}_r)$ are simply $C_{kr} = \gamma_{kr}^2$, where for notation homogeneity γ_{kr} is an Ersatz of C'_{kr}.

2.4. Fundamental difference with projection

The algorithm proposed in this contribution will choose *at every iteration* the most appropriate parameterization by setting to 1 the absolute value of one parameter in every vector $\boldsymbol{\alpha}_r$ and $\boldsymbol{\beta}_r$, $1 \le r \le R$, as depicted in (7). Every unit-norm vector with nonnegative entries may hence have a 1 in the numerator at a *different location*, and this *may vary* with iterations.

There is a fundamental difference between this parameterization (PAPOH) and projection. In fact, the iterations we produce are by construction all on the hypersphere, and the most well conditioned parameterization is chosen at very iteration. On the other hand in projected iterative algorithms, iterations are generated outside the hypersphere and are projected onto it at every iteration.

3. ALGORITHM

3.1. Parameter update

To solve (P.4), we use a conjugate gradient algorithm. Let

$$\mathbf{x}_t \overset{\text{def}}{=} \begin{bmatrix} \hat{\boldsymbol{\alpha}} \\ \hat{\boldsymbol{\beta}} \\ \hat{\boldsymbol{\gamma}} \end{bmatrix}_t \text{ and } \mathbf{g}_t \overset{\text{def}}{=} \begin{bmatrix} \boldsymbol{\nabla}\Upsilon_{\boldsymbol{\alpha}} \\ \boldsymbol{\nabla}\Upsilon_{\boldsymbol{\beta}} \\ \boldsymbol{\nabla}\Upsilon_{\boldsymbol{\gamma}} \end{bmatrix}_t.$$

For this algorithm the estimates $\hat{\boldsymbol{\alpha}}$, $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\gamma}}$ are updated at iteration t using the Pollak-Ribière update formula:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{x}_t - \mu_t \mathbf{d}_t, \\ \nu_t &= (\mathbf{g}_{t+1} - \mathbf{g}_t)^\top \mathbf{g}_{t+1} / ||\mathbf{g}_t||_2^2, \\ \mathbf{d}_{t+1} &= \mathbf{M}_t\, g_{t+1} - \nu_t \mathbf{d}_t, \end{aligned}$$

where μ_t is a variable step size, $\mathbf{d}_t$ is the descent direction, which is initialized to $\mathbf{d}_1 = \mathbf{g}_1$, and $\mathbf{M}_t$ is a preconditioning matrix, usually diagonal, which we have set to identity in the present paper.

3.2. Gradient expressions

At each iteration t, one index $i(r,t)$ (resp. $j(r,t)$) in every vector $\boldsymbol{\alpha}_r$ (resp. $\boldsymbol{\beta}_r$) is chosen, so that $\alpha_{i(r,t),r}^2 = 1$ (resp. $\beta_{j(r,t),r}^2 = 1$). The gradient w.r.t. $\alpha_{i(r,t),r}$ does not exist at iteration t, and in practice it may be set to zero since $\alpha_{i(r,t),r}^2$ remains equal to 1. We eventually have the following derivatives:

$$\text{for } i \neq i(r,t) \quad \frac{\partial A_{ir}}{\partial \alpha_{ir}} = 2\,\frac{\alpha_{ir} u_{ir}}{(\sum_q \alpha_{qr}^2)^2}, \tag{8}$$

$$\text{for } i \notin \{i(r,t), p\} \quad \frac{\partial A_{pr}}{\partial \alpha_{ir}} = -2\,\frac{\alpha_{ir}\alpha_{pr}^2}{(\sum_q \alpha_{qr}^2)^2}, \tag{9}$$

$$\text{with} \qquad u_{ir} \overset{\text{def}}{=} \textstyle\sum_{q \neq i} \alpha_{qr}^2,$$

$$\text{and for } p = i(r,t) \quad \frac{\partial A_{pr}}{\partial \alpha_{ir}} = -2\,\frac{\alpha_{ir}}{(\sum_q \alpha_{qr}^2)^2}. \tag{10}$$

Following the notation assumed for $\boldsymbol{\alpha}_r$ in (7), β_{jr} and γ_{kr} will denote the jth and kth entries of vectors $\boldsymbol{\beta}_r$ and $\boldsymbol{\gamma}_r$, respectively. Now define $E_{ijk} = Y_{ijk} - \sum_r A_{ir}B_{jr}C_{kr}$, so that $\Upsilon = ||\boldsymbol{\mathcal{E}}||^2$. Then, $\forall p, i, 1 \leq p \leq I, i \in \{1, \ldots I\} - \{i(r,t)\}$:

$$\frac{\partial \Upsilon}{\partial \alpha_{ir}} = -2 \sum_{pjk} E_{pjk} \, \frac{\partial A_{pr}}{\partial \alpha_{ir}} B_{jr}C_{kr}. \tag{11}$$

Similar expressions hold for $\boldsymbol{\beta}$, by exchanging the roles of $\mathbf{A}$ and $\mathbf{B}$, with $\mathbf{b}_r = \boldsymbol{\beta}_r \boxdot \boldsymbol{\beta}_r / ||\boldsymbol{\beta}_r||_2^2$ as in (7). For the third loading matrix, there is no normalization, and $\mathbf{c}_r = \boldsymbol{\gamma}_r \boxdot \boldsymbol{\gamma}_r$ so that

$$\frac{\partial \Upsilon}{\partial \gamma_{kr}} = -4 \sum_{ij} E_{ijk} \, A_{ir}B_{jr}\gamma_{kr}. \tag{12}$$

3.3. Exact Line Search (ELS) by polynomial fitting

Since the objective function Υ defined in (P.4) is a rational function in the unknown parameters, for any fixed direction $\mathbf{g} = [\mathbf{g}_\alpha, \mathbf{g}_\beta, \mathbf{g}_\gamma]^\top$, defined *e.g.* by the gradient, the stationary points of Υ with respect to μ_t are the roots of a polynomial $\pi(\mu)$ in μ_t. This is a great advantage of the parameterization we have assumed: the value of the step size μ_t yielding the *globally* minimal value of the objective along the search direction can be found. More precisely, we proceed in three steps:

1. *Computation of the coefficients of the rational function.* From (1) and (6), it can be seen that the objective takes the form
$$\Upsilon(\boldsymbol{\alpha} + \mu\mathbf{g}_\alpha, \boldsymbol{\beta} + \mu\mathbf{g}_\beta, \boldsymbol{\gamma} + \mu\mathbf{g}_\gamma) =$$
$$\sum_{ijk} \left[Y_{ijk} - \sum_{r=1}^{R} \frac{\ell^2_{A,ir}(\mu)}{q_{A,r}(\mu)} \frac{\ell^2_{B,jr}(\mu)}{q_{B,r}(\mu)} \, \ell^2_{C,kr}(\mu) \right]^2,$$
where $\ell_{A,ir}$, $\ell_{B,jr}$ and $\ell_{C,kr}$ are linear forms, and $q_{A,r}$ and $q_{B,r}$ are quadratic forms in μ.

2. *Finding the roots of the first derivative*:
$$\frac{\partial \Upsilon}{\partial \mu} = 0 \Leftrightarrow \pi(\mu) = 0, \tag{13}$$
where the exact expression of π can be obtained from those of the above linear and quadratic forms.

3. *Finding the root corresponding to the global minimum*: just select the best root by evaluating Υ at those values.

However, to spare some computational load, this optimal step size does not need to be computed at every iteration, but can be executed only cyclically, for instance at every $IJKR$ iterations, or just before termination, with the hope that it could help escape from local minima in case the algorithm is stuck.

3.4. Complexity

Following the usual practice, the computational complexity is assimilated to the number of multiplications, being understood that a floating point operation (flop) is implemented as a multiplication followed by an addition, and that the number of multiplications is generally of same order as the number of additions [17].

Update rule. The update of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ is dominated by the gradient computation, whereas the complexity involved by (8-9) is negligible. In fact, the computation of the gradient requires the contraction of tensor E_{ijk} over two or three indices as shown in (11) and (12). More precisely, the update (8) requires an order of $O(IJKR(I + J + 1))$ multiplies.

Line Search. Next, the computation of the globally optimal step size (ELS) is dominated by the calculation of the coefficients of polynomial $\pi(\mu)$; the computation of its roots is comparatively negligible. More precisely, dropping indices for an easier writing, polynomial π takes the expression:

$$\pi(\mu) = \sum_{r=1}^{R} \left[\prod_{n \neq r} q^2_{An} q^2_{Bn} \right] (2F_r - G_r) \;\text{ with}$$

$$\begin{aligned} F_r &= q_{Ar}q_{Br} \sum_{ijk} E_{ijk}[\ell_A\ell'_A\ell^2_B\ell^2_C + \ell^2_A\ell_B\ell'_B\ell^2_C + \ell^2_A\ell^2_B\ell_C\ell'_C] \\ G_r &= (q'_{Ar}q_{Br} + q_{Ar}q'_{Br}) \sum_{ijk} E_{ijk}\ell^2_A\ell^2_B\ell^2_C. \end{aligned}$$

If we assume that the Fourier transform is not used to compute the product of two polynomials of degree m and n, with say $0 < m \leq n$, then the complexity of the product is $(n - m)(m+1) + (m+1)(m+2)/2$ flops. On this basis, one can then show that computing all the coefficients of $\pi(\mu)$ requires a computational complexity which increases as $O(IJKR)$.

3.5. Initialization and stopping

The initial value is drawn randomly and belongs to the feasible set; that is, every parameter vector $\mathbf{a}_r$ or $\mathbf{b}_r$ is nonnegative belongs to the unit $\mathcal{L}_1$ hypersphere.

One defines a maximal number of iterations, t_{max}, and a minimal relative variation of the objective, ε. The iterations are stopped if one of the two conditions below are satisfied:

1. Maximal number of iterations reached: $t \geq t_{max}$
2. The relative variations of the objective function have become negligible: $|\Upsilon_{t+1} - \Upsilon_t|\Upsilon^{-1}_{t+1} < \varepsilon$.

4. COMPUTER EXPERIMENTS

In what follows we test five different algorithms used to find tensor approximations: alternating least squares (ALS) without nonnegativity constraints[1], alternating least squares with

[1] A projection is carried out only at the last iteration after normalizing the factors A and B.

projection on the nonnegative orthant (NALS), gradient with nonnegative multiplicative update (NMU), conjugate gradient using the parameterization with squares (CGP), and conjugate gradient with PAPOH (PAPOH CG). The true nonnegative CP model has random elements such that A and B have unit L_1 norm. The model dimensions are $I = 20$, $J = 20$, $K = 15$ and $R = 8$ and it is measured with independent identically distributed (i.i.d.) Gaussian noise with standard deviation $\sigma = 5 \times 10^{-4}$. Algorithm initialization is also random but equal for all algorithms. In the simulations, we used the NMU algorithm from the *MATLAB Tensor Toolbox* [18]. Simulation results for two different realizations are shown in Fig.1, where the reconstruction error is plotted (in dB) as a function of the number of iterations. Note that if we compare the performance of the gradient-based algorithms, NMU is the slowest and PAPOH CG is the fastest. Notice that the alternating algorithms can be either fast or slow depending on the initial conditions. In the second realization we can see that NALS is blocked in a point far from the optimum.

The data used for the second test are generated by artificially mixing fluorescence excitation-emission spectra representative of 4 different polycyclic aromatic hydrocarbons (PAH): naphthalene, phenanthrene, pyrene, benzanthracene. The mixing matrices (10 in the present case) are generated randomly with nonnegative values. The algorithms are tested supposing $R = 5$ to simulate a misdetection of the number of solutes. The original and reconstructed spectra are shown in Fig.2 after 5000 iterations, and plotted as a function of wavelengths, as usual in the field of chemometrics. The results for the standard ALS algorithm are shown by normalizing and projecting the final result on the nonnegative orthant (but its iterations are unconstrained). Results for the NMU algorithm are not shown due to space limitation.

One can observe that the ALS algorithms generate a spurious component that does not exist. On the other hand, the CGP of [11] and PAPOH CG make a good job. If we look at the sums of the squared errors on the parameters (and not on the tensor reconstruction) we can clearly see the differences. They are the following: ALS - 2.00×10^{-3}, NALS - 2.36×10^{-4}, CGP - 2.82×10^{-5} and PAPOH CG - 3.12×10^{-9}. In practice, the gradient-based algorithms can be made even more robust to a misdetection of the number of solutes by adding an approximation of the mixed pseudo norm $\mathcal{L}_{\infty,0}$ as a penalty term (this will be reported in another paper). This can hardly be done in the ALS approach.

5. CONCLUSIONS

The $\mathcal{L}_1$ norm has been chosen since it is the natural norm for vectors with nonnegative entries. This led to a simple parameterization of the intersection of the unit $\mathcal{L}_1$-hypersphere and the nonnegative cone, which is symmetric and expressed as a rational function. Among useful by-products, we have simpler gradient expressions, a conditioning easy to monitor, and a step size that can be globally optimized. The efficiency of this algorithm is illustrated by a problem of fluorescence spectroscopy, which is important in environmental sciences. The CP decomposition of the nonnegative data tensor into rank-one terms permits to identify pure solutes, such as toxic polycyclic aromatic hydrocarbons highly diluted in water. In particular, the proposed algorithm better detects the number of fluorescent solutes, and hence better identifies them via their spectrum.

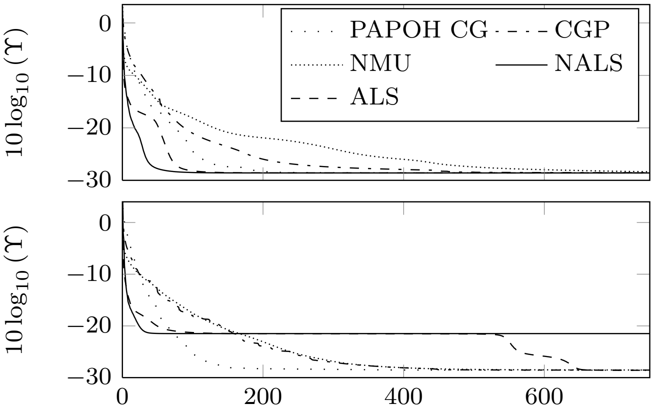

Fig. 1. Cost function Υ (dB) versus iterations. The true CP model is generated randomly such that all of its elements are nonnegative and the columns of the factors A and B have unit L_1 norm. The results obtained with two realizations are shown. The CP model is of dimension $20 \times 20 \times 15$, with rank $R = 8$. The model is measured with i.i.d. Gaussian noise with standard deviation $\sigma = 5 \times 10^{-4}$.

6. REFERENCES

[1] A. Shashua and T. Hazan, “Non-negative tensor factorization with applications to statistics and computer vision,” in *22nd Int. Conf. Machine Learning*, Bonn, 2005, pp. 792–799.

[2] D. M. Dunlavy, T. G. Kolda, and E. Acar, “Temporal link prediction using matrix and tensor factorizations,” *ACM Trans. Knowledge Discov. Data*, vol. 5, no. 2, Feb. 2011.

[3] E. Acar and B.Yener, “Unsupervised multiway data analysis: A literature survey,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 1, pp. 6–20, 2009.

[4] P. Zhang, H. Wang, R. Plemmons, and P. Pauca, “Tensor methods for hyperspectral data analysis: A space object material identification study,” *J. Optical Soc. Amer.*, vol. 25, no. 12, pp. 3001–3012, Dec. 2008.

[5] A. Smilde, R. Bro, and P. Geladi, *Multi-Way Analysis*, Wiley, Chichester UK, 2004.

[6] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications,” *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[7] L-H. Lim and P. Comon, “Nonnegative approximations of nonnegative tensors,” *Jour. Chemometrics*, vol. 23, pp. 432–441, Aug. 2009, hal-00410056.

[8] P. Comon, “Tensors: a brief introduction,” *IEEE Sig. Proc. Magazine*, vol.31, no.3, May 2014, special issue on BSS. hal-00923279.

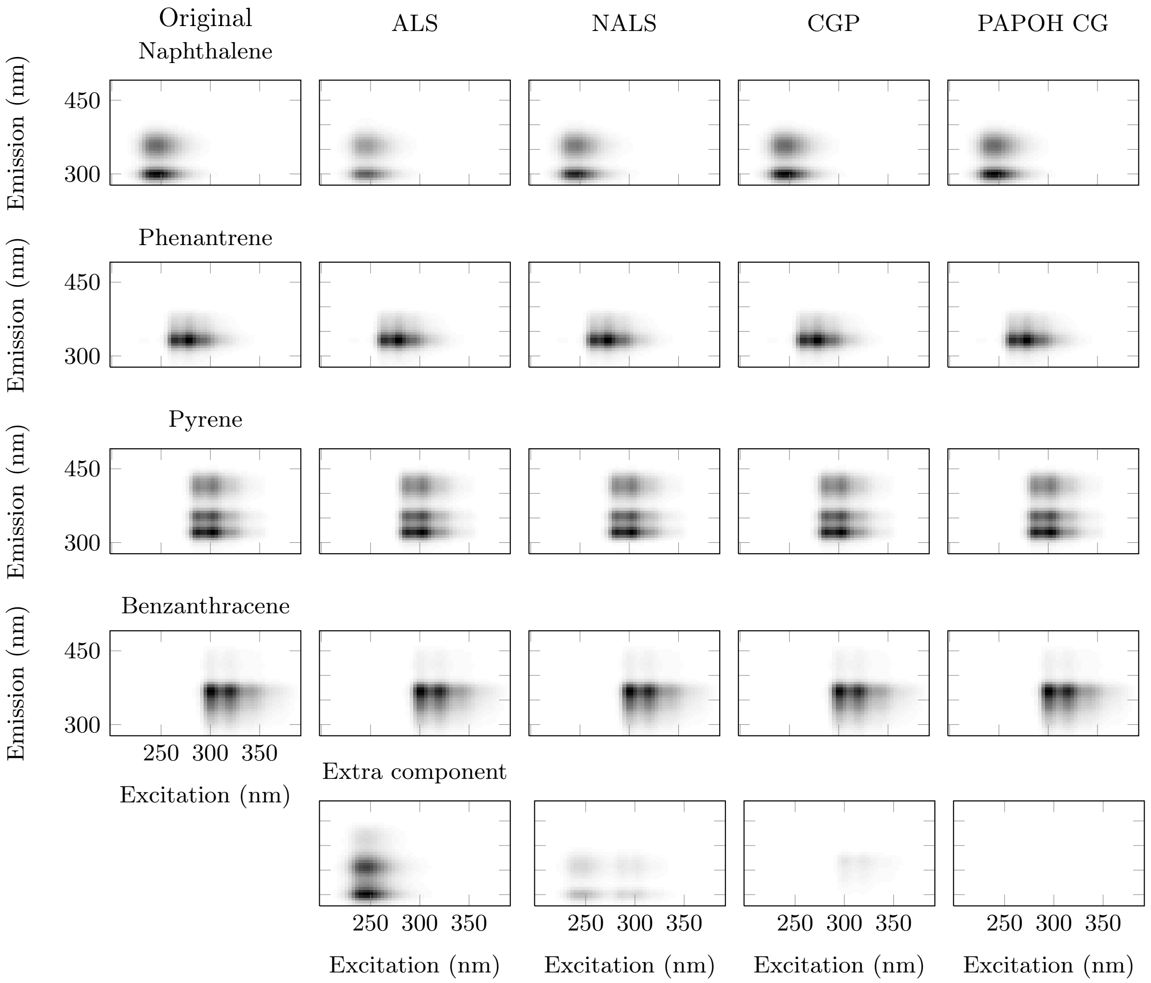

Fig. 2. Fluorescence spectra obtained with four different algorithms: alternating least squares (ALS) without nonnegativity constraints (negative values appear as white), ALS with projection on the nonnegative orthant (NALS), nonnegative conjugate gradient (CGP), and conjugate gradient with Parameterization of Positive Orthant Hyperspheres (PAPOH CG). As it is indicated in the column "Original", where the original unmixed spectra are shown, the spectra are representative of 4 different polycyclic aromatic hydrocarbons (PAH): naphthalene, phenanthrene, pyrene, benzanthracene. The algorithms are tested supposing $R = 5$ to simulate a misdetection of the number of components, the 5^{th} line thus represents an additional artificial component that may have been estimated. The black color represents the largest intensity value and white is zero.

[9] P. Paatero, "A weighted non-negative least squares algorithm for three-way 'parafac' factor analysis," *Chemo. Intel. Lab. Syst.*, vol.38, no.2, pp. 223–242, 1997.

[10] P. Comon, X. Luciani, and A. L. F. De Almeida, "Tensor decompositions, alternating least squares and other tales," *J. Chemometrics*, vol.23, pp. 393–405, 2009, hal-00410057.

[11] J.-P. Royer, N. Thirion-Moreau, and P. Comon, "Computing the polyadic decomposition of nonnegative third order tensors," *Signal Processing*, vol. 91, no.9, pp. 2159–2171, 2011.

[12] V. De Silva and L.-H. Lim, "Tensor rank and the ill-posedness of the best low-rank approximation problem," *SIAM J. Matrix Ana. Appl.*, vol.30, no.3, pp. 1084–1127, 2008.

[13] L.-H. Lim and P. Comon, "Nonnegative approximations of nonnegative tensors," *Journal of Chemometrics*, vol.23, no.7-8, pp. 432–441, 2009.

[14] R. Bro, *Multi-way analysis in the food industry: models, algorithms, and applications*, Ph.D. thesis, University of Amsterdam, The Netherlands, 1998.

[15] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*, John Wiley & Sons, 2009.

[16] G. Zhou, A. Cichocki, Q. Zhao, and S. Xie, "Nonnegative matrix and tensor factorizations: an algorithmic perspective," *IEEE Sig. Proc. Magazine*, vol.31, no.3, May 2014, special issue on BSS.

[17] G. H. Golub and C. F. Van Loan, *Matrix computations*, The John Hopkins University Press, 1989.

[18] B. W. Bader and T. G. Kolda and others, "MATLAB Tensor Toolbox Version 2.5," Available online, January 2012, `http://www.sandia.gov/~tgkolda/TensorToolbox/`.