Computing the nonnegative 3-way tensor factorization using Tikhonov regularization

Jean-Philip Royer, Pierre Comon, Nadège Thirion-Moreau

COMPUTING THE NONNEGATIVE 3-WAY TENSOR FACTORIZATION USING TIKHONOV REGULARIZATION

Jean-Philip Royer, Pierre Comon[*]

I3S, Algorithmes/Euclide-B,
2000 route des Lucioles, BP 121, F-06903,
Sophia Antipolis Cédex,
Tel/Fax: +33 4 9294 2717/2896

Nadège Thirion-Moreau

ISITV, LSEET, UMR CNRS 6017,
Université du Sud Toulon Var,
F-83957, La Garde Cédex, France,
Tel/Fax: +33 4 9414 2456/2671

ABSTRACT

This paper deals with the minimum polyadic decomposition of a nonnegative three-way array. The main advantage of the nonnegativity constraint is that the approximation problem becomes well posed. To tackle this problem, we suggest the use of a cost function including penalty terms built with matrix exponentials. Gradient components are then derived, allowing to efficiently implement the decomposition using classical optimization algorithms. In our case ALS and conjugate gradient algorithms are studied and compared with another existing algorithm, thanks to computer simulations performed in the context of data analysis.

Index Terms— Data analysis, data mining, non linear conjugate gradient, nonnegative 3-way array, Canonical Polyadic decomposition, Tikhonov regularization.

1. INTRODUCTION

Performing the canonical polyadic decomposition of a tensor (sometimes also called Parafac, CP, or CanD) is of interest in many fields such as data analysis/mining, signal and image processing, chemometrics [1, 2, 3, 4]. But some difficulties may arise when searching for the zeros of a polynomial of degree six or larger, in a very large number of variables. Hence the CP calculation can be numerically difficult to complete. When approximating a tensor by another of lower rank, we can also face an ill-posed problem, for which solutions may contain diverging components cancelling each other [5, 6]. Recently, researchers have been paying more and more attention to nonnegative matrices or tensors for at least two reasons: first their usefulness in fields where it is dealt with images and/or spectra (such as hyperspectral imaging, chemometrics, etc), and second because their low-rank approximation becomes well-posed. A book is even entirely dedicated to that problem [7]. In this paper, we present an approach that takes into account this constraint by modifying the cost fonction. Then, the three gradient components are derived allowing to efficiently implement the decomposition using classical optimization algorithms (ALS and conjugate gradient algorithms in our case). Computer simulations are performed in the context of data analysis to illustrate the behavior of the suggested approach and to compare it with another existing algorithm described in [7]. The paper is organized as follows. The problem is stated in Section 2. Next, our approach to the nonnegativity constraint is described in Section 3, and optimization algorithms in Section 4. Finally, Section 5 is devoted to computer simulations in the context of data mining.

[*]This work has been supported by a "PRES euro-méditerranéen" grant.

2. PROBLEM STATEMENT

In a given basis, a third order tensor can be represented by a three-way array. The order of a tensor hence corresponds to the number of indices of the associated array (its *ways* or *modes* [4]). In this paper, due to the considered applications, including fluorescence spectroscopy [8][4] or hyperspectral imaging [9], we focus on real positive 3-way arrays denoted by $\mathbf{T} = (t_{ijk}) \in \mathbb{R}^{I\times J\times K}$, admitting the following trilinear decomposition, also known as a triadic decomposition [10] of $\mathbf{T}$: $\mathbf{T} = \sum_{f=1}^{F} \mathbf{a}_f \circledast \mathbf{b}_f \circledast \mathbf{c}_f$, where the three involved matrices $\mathbf{A} = (a_{if}) = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_F] \in \mathbb{R}^{I\times F}$, $\mathbf{B} = (b_{jf}) = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_F] \in \mathbb{R}^{J\times F}$, $\mathbf{C} = (c_{kf}) = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_F] \in \mathbb{R}^{K\times F}$ are the so-called *loading matrices*, whose columns are the *loading factors*. F is a sufficiently large integer and $\circledast$ stands for the outer (tensor) product. Equivalently, $\forall i = 1, \ldots, I, \quad \forall j = 1, \ldots, J, \quad \forall k = 1, \ldots, K$, we have the relation between array entries:

$$t_{ijk} = \sum_{f=1}^{F} a_{if} b_{jf} c_{kf} \tag{1}$$

The smallest integer F that can be found such that the equality above holds exactly is called the *tensor rank* [11]. For this value of F, the above decomposition will be called the Canonical Polyadic decomposition (CP) of tensor $\mathbf{T}$. The aforementioned model can be written in a compact matrix form using

the Khatri-Rao product $\odot$, as

$$\mathbf{T}_{(1)}^{I,JK} = \mathbf{A}(\mathbf{C} \odot \mathbf{B})^T, \quad \mathbf{T}_{(2)}^{J,KI} = \mathbf{B}(\mathbf{C} \odot \mathbf{A})^T, \\ \mathbf{T}_{(3)}^{K,JI} = \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T, \tag{2}$$

where $\mathbf{T}_{(1)}^{I,JK}$ (resp. $\mathbf{T}_{(2)}^{J,KI}$ and $\mathbf{T}_{(3)}^{K,JI}$) is the matrix of size $I \times JK$ (resp. $J \times KI$ and $K \times JI$) obtained by unfolding the array $\mathbf{T}$ of size $I \times J \times K$ in the first mode (resp. the second and third mode).

2.1. CP decomposition of 3-way tensors

The goal of the CP decomposition for three-way tensors is to estimate the 3 loading matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$. A rather standard approach consists of (i) assuming that the rank F is known or overestimated, and (ii) minimizing the Frobenius norm of the error between the data and its model:

$$\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) = \|\mathbf{T}_{(1)}^{I,JK} - \mathbf{A}(\mathbf{C} \odot \mathbf{B})^T\|_F^2 \tag{3} \\ = \|\mathbf{T}_{(2)}^{J,KI} - \mathbf{B}(\mathbf{C} \odot \mathbf{A})^T\|_F^2 = \|\mathbf{T}_{(3)}^{K,JI} - \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T\|_F^2,$$

where $\|\cdot\|_F^2$ stands for the Frobenius norm. The matrix gradient calculation leads to:

$$\nabla_{\mathbf{A}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) = \left[-\mathbf{T}_{(1)}^{I,JK} + \mathbf{A}(\mathbf{C} \odot \mathbf{B})^T\right](\mathbf{C} \odot \mathbf{B}), \tag{4}$$

$$\nabla_{\mathbf{B}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) = \left[-\mathbf{T}_{(2)}^{J,KI} + \mathbf{B}(\mathbf{C} \odot \mathbf{A})^T\right](\mathbf{C} \odot \mathbf{A}), \tag{5}$$

$$\nabla_{\mathbf{C}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) = \left[-\mathbf{T}_{(3)}^{K,JI} + \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T\right](\mathbf{B} \odot \mathbf{A}). \tag{6}$$

3. NONNEGATIVE 3-WAY ARRAY

3.1. Tikhonov regularization

One possible way to take into account the nonnegative constraint is to add "well-chosen" penalty terms to the cost function *i.e.* negative entries from the loading matrices have to considerably increase the value of the cost function, whereas positive entries must let it nearly unchanged. Thus, the minimization problem (3) is replaced by the minimization of cost function: $\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + \alpha f(\mathbf{A}) + \beta f(\mathbf{B}) + \gamma f(\mathbf{C})$,where α, β, γ are positive scalars which play the role of weights for the three loading matrices. And $f(\cdot)$ is chosen such that: (i) $f(\cdot)$ is continuous, (ii) $f(\mathbf{A}) \geq 0$ for all $\mathbf{A}$ in $\mathbb{R}^{I\times F}$ and (iii) $f(\mathbf{A}) = 0$ if and only if $\mathbf{A} \in \mathbb{R}^{+I\times F}$.

3.2. Cichocki's ALS algorithm

In [7], it was suggested to use the cost function below:

$$\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C}) = \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + \alpha\|\mathbf{A}\|_F^2 + \beta\|\mathbf{B}\|_F^2 + \gamma\|\mathbf{C}\|_F^2.$$

The gradient components found in (4)-(6) are replaced by:

$$\nabla_{\mathbf{A}}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{A}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + \alpha\mathbf{A}, \tag{7}$$

$$\nabla_{\mathbf{B}}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{B}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + \beta\mathbf{B}, \tag{8}$$

$$\nabla_{\mathbf{C}}\mathcal{G}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{C}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + \gamma\mathbf{C}. \tag{9}$$

By equating the gradient components to zero, the following solutions are found allowing the ALS technique to be implemented ($\mathbf{I}_F$ is the identity matrix of size $F \times F$):

$$\widehat{\mathbf{A}} = \mathbf{T}_{(1)}^{I,JK}(\mathbf{C} \odot \mathbf{B})\left[(\mathbf{C} \odot \mathbf{B})^T(\mathbf{C} \odot \mathbf{B}) + \alpha\mathbf{I}_F\right]^\dagger, \\ \widehat{\mathbf{B}} = \mathbf{T}_{(2)}^{J,KI}(\mathbf{C} \odot \mathbf{A})\left[(\mathbf{C} \odot \mathbf{A})^T(\mathbf{C} \odot \mathbf{A}) + \beta\mathbf{I}_F\right]^\dagger, \\ \widehat{\mathbf{C}} = \mathbf{T}_{(3)}^{K,JI}(\mathbf{B} \odot \mathbf{A})\left[(\mathbf{B} \odot \mathbf{A})^T(\mathbf{B} \odot \mathbf{A}) + \gamma\mathbf{I}_F\right]^\dagger. \tag{10}$$

Finally, a "projection operator" $[\cdot]_+$ is applied, whose aim is to enforce positive entries.

$$\widehat{\mathbf{A}} \leftarrow \left[\widehat{\mathbf{A}}\right]_+, \quad \widehat{\mathbf{B}} \leftarrow \left[\widehat{\mathbf{B}}\right]_+, \quad \widehat{\mathbf{C}} \leftarrow \left[\widehat{\mathbf{C}}\right]_+, \tag{11}$$

where $[\mathbf{M} = (m_{ij})]_+$ returns a matrix of the same size as $\mathbf{M}$ whose (i,j) entry is $\max\{\epsilon, m_{ij}\}$ if ϵ is a small constant (typically 10^{-16}).

3.3. Suggested approach

Instead, we suggest to use the following cost function:

$$\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + \alpha\|e^{-\gamma_A\mathbf{A}}\|_F^2 \\ + \beta\|e^{-\gamma_B\mathbf{B}}\|_F^2 + \gamma\|e^{-\gamma_C\mathbf{C}}\|_F^2, \tag{12}$$

where γ_A, γ_B, γ_C are positive constants. The goal is to constraint the cost function to reach high values if the loading matrices contain negative entries, so that the minimization problem is achieved only for nonnegative loading matrices. Gradients are shown to be (see Appendix):

$$\nabla_{\mathbf{A}}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{A}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + 4\alpha\gamma_A^2\left[\mathbf{A} - \frac{\mathbf{1}_{I,F}}{2\gamma_A}\right], \tag{13}$$

$$\nabla_{\mathbf{B}}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{B}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + 4\beta\gamma_B^2\left[\mathbf{B} - \frac{\mathbf{1}_{J,F}}{2\gamma_B}\right], \tag{14}$$

$$\nabla_{\mathbf{C}}\mathcal{I}(\mathbf{A},\mathbf{B},\mathbf{C}) = \nabla_{\mathbf{C}}\mathcal{F}(\mathbf{A},\mathbf{B},\mathbf{C}) + 4\gamma\gamma_C^2\left[\mathbf{C} - \frac{\mathbf{1}_{K,F}}{2\gamma_C}\right] \tag{15}$$

(where $\mathbf{1}_{I,F}$ is the $I \times F$ matrix of ones), allowing the construction of the two following $(I+J+K)F \times 1$ vectors:

$$\mathbf{g}^{(k)} = \begin{pmatrix} \mathsf{vec}\{\nabla_{\mathbf{A}}\mathcal{I}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\} \\ \mathsf{vec}\{\nabla_{\mathbf{B}}\mathcal{I}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\} \\ \mathsf{vec}\{\nabla_{\mathbf{C}}\mathcal{I}(\mathbf{A}^{(k)},\mathbf{B}^{(k)},\mathbf{C}^{(k)})\} \end{pmatrix}, \mathbf{x}^{(k)} = \begin{pmatrix} \mathsf{vec}\{\mathbf{A}^{(k)}\} \\ \mathsf{vec}\{\mathbf{B}^{(k)}\} \\ \mathsf{vec}\{\mathbf{C}^{(k)}\} \end{pmatrix} \tag{16}$$

where the $\mathsf{vec}\{\cdot\}$ operator applied on a given matrix stacks its columns into a column vector.

4. OPTIMIZATION ALGORITHMS

4.1. ALS algorithm

By equating the gradient components to zero, we can estimate successively the three loading matrices in the case of the Alternated Least Squares (ALS) algorithm:

$$\begin{aligned}
\widehat{\mathbf{A}} &= \left[\mathbf{T}_{(1)}^{I,JK}(\mathbf{C}\odot\mathbf{B}) + 2\alpha\gamma_A \mathbf{1}_{I,F}\right]\left[(\mathbf{C}\odot\mathbf{B})^T(\mathbf{C}\odot\mathbf{B}) \right.\\
&\quad \left. +4\gamma_A^2\alpha\,\mathbf{I}_F\right]^\dagger,\\
\widehat{\mathbf{B}} &= \left[\mathbf{T}_{(2)}^{J,KI}(\mathbf{C}\odot\mathbf{A}) + 2\beta\gamma_B \mathbf{1}_{J,F}\right]\left[(\mathbf{C}\odot\mathbf{A})^T(\mathbf{C}\odot\mathbf{A}) \right.\\
&\quad \left. +4\gamma_B^2\beta\,\mathbf{I}_F\right]^\dagger,\\
\widehat{\mathbf{C}} &= \left[\mathbf{T}_{(3)}^{K,JI}(\mathbf{B}\odot\mathbf{A}) + 2\gamma\gamma_C \mathbf{1}_{K,F}\right]\left[(\mathbf{B}\odot\mathbf{A})^T(\mathbf{B}\odot\mathbf{A}) \right.\\
&\quad \left. +4\gamma_C^2\gamma\,\mathbf{I}_F\right]^\dagger. \qquad (17)
\end{aligned}$$

This process is repeated until an error criterion is satisfied, or until a given number of iterations.

4.2. Non linear conjugate gradient

To estimate the three loading matrices A, B, and C, $\mathcal{I}$ given in (12) has to be minimized with respect to those three matrices. With this aim, several descent algorithms could be used. We focus on the non linear conjugate gradient algorithm, in which a vector $\mathbf{x}$ given in (16) is updated at each step k according to the following adaptation rule:

$$\left\{\begin{array}{ll} \mathbf{x}^{(k+1)} & = \mathbf{x}^{(k)} + \mu^{(k)}\mathbf{d}^{(k)} \\ \mathbf{d}^{(k+1)} & = -\mathbf{g}^{(k+1)} + \beta^{(k)}\mathbf{d}^{(k)}. \end{array}\right. \qquad (18)$$

Two expressions of β can be used [12]. As noticed in [12], if a conjugate gradient method is reinitialized, from time to time, by setting $\mathbf{d}^{(i)} = -\mathbf{g}^{(i)}$, we might get better performance that is why in our case we have chosen to perform this "restart" every $(I+J+K)F$ iterations. With regard to the choice of the step size $\mu^{(k)}$ see [13].

5. COMPUTER SIMULATIONS

Simulations are now provided in the context of data analysis (more precisely in chemometrics with fluorescence images). **The application field -** Considering a water sample containing one component of dissolved organic matter called fluorophore, and excited by an optical light, we can establish a relation. The intensity of fluorescence depends on the concentration of the component for the considered sample k (c_k), the excitation spectrum ($\epsilon(\lambda_e)$), and the re-emitted spectrum ($\gamma(\lambda_f)$): $I(\lambda_f,\lambda_e,k) = I_o\,\gamma(\lambda_f)\,\epsilon(\lambda_e)\,c_k$. When several components are dissolved, we face a problem of mixture separation. Yet, when their concentration is low enough, the Beer-Lambert law is linear and reads:

$$I(\lambda_f,\lambda_e,k) = I_o \sum_\ell \gamma_\ell(\lambda_f)\,\epsilon_\ell(\lambda_e)\,c_{k,\ell} \qquad (19)$$

Comparing (19) and (1) shows that $\gamma_\ell(\lambda_f(i))$ corresponds to a_{if}, $\epsilon_\ell(\lambda_e(j))$ to b_{jf} and $c_{k,\ell}$ matches c_{kf}. For each excitation wavelength absorbed by the component, an intensity of fluorescence depending on the emission wavelength is measured, so that a fluorescence image can be reconstructed.

The results - We have simulated a tensor $\mathcal{T}$, with $F=4$ components, whose fluorescence images are of size 47×71. A matrix $\mathbf{C}$ of size 50×4 is randomly generated. It contains positive values only. Thus, $\mathcal{T}$ is of size $47\times 71\times 50$. To be able to compare the behaviour of the different algorithms, we introduce an error index: $E = \|\mathbf{T}-\widehat{\mathbf{T}}\|_F^2$, or in dB, $E_{dB} = 10\log_{10}(E)$, where $\widehat{\mathbf{T}} = \sum_{f=1}^F \widehat{\mathbf{a}}_f \circledast \widehat{\mathbf{b}}_f \circledast \widehat{\mathbf{c}}_f$ and $\widehat{\mathbf{a}}$, $\widehat{\mathbf{b}}$ and $\widehat{\mathbf{c}}$ are the estimated factors. The three aforementioned algorithms are compared in Fig. 1. For Cichocki's ALS algorithm, we have applied $\alpha=\beta=\gamma=10^{-6}$. Regarding our algorithms, it was chosen to "normalize" *i.e.* $\gamma_A = \frac{1}{\|\mathbf{A}\|}$, $\gamma_B = \frac{1}{\|\mathbf{B}\|}$, $\gamma_C = \frac{1}{\|\mathbf{C}\|}$. Moreover, α, β, γ are decreased at each iteration: $\alpha(k+1) = \beta(k+1) = \gamma(k+1) = \frac{\alpha(k)}{m}$, with m a constant and $\alpha(0)=0.5$), so that the influence of the penalty terms becomes more and more weak. We can observe that the three algorithms are quite fast (convergence reached in less than 100 iterations). The reconstruction error is smaller with our algorithms than with the Cichocki's ALS. Its performance is indeed bounded because of the addition of penalty terms; such a problem is avoided in our case by diminishing the influence of penalty terms as more iterations are run. However, a small reconstruction error does not necessarily imply a good estimation of the loading matrices, as now demonstrated. Assume we still have a mixture of 4 components, but set $F=5$ in the model (the rank is overestimated). The obtained results are displayed in Fig. 2. On the left, the images have been obtained thanks to our algorithm (combining conjugate gradient and Tikhonov regularization), whereas on the right, they have been reconstructed thanks to the algorithm of [7] (combining ALS, regularization and projection). In both cases, it is difficult to assert there are 4 components. The artefact remains nevertheless less important with our approach. Note that our other approach proposed in [13] did a better job in this respect.

6. CONCLUSION

An algorithm has been proposed, combining Tikhonov regularization and either the ALS or the conjugate gradient algorithm, in order to compute the minimal polyadic decomposition of nonnegative three-way arrays. First, a new cost function has been introduced, then the gradient matrices have calculated and two different optimization algorithms have been used. Computer simulations have been provided to illustrate the behaviour of the suggested approaches and to compare them to other algorithms found in the literature.

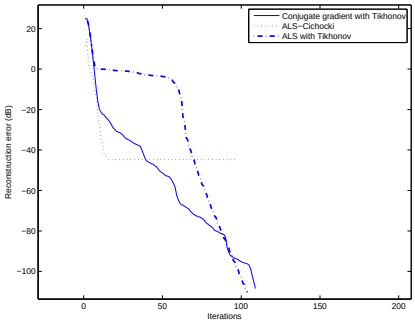

Fig. 1. Reconstruction error (dB) versus number of iterations using a nonnegative $71 \times 47 \times 50$ tensor.

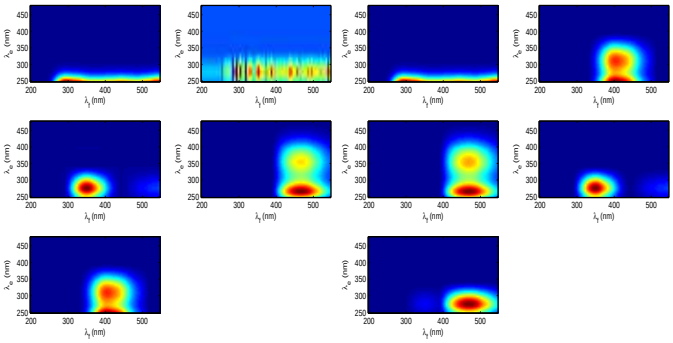

Fig. 2. Mixture of 4 factors, assuming $F = 5$; the 5 estimated emission-excitation images using the conjugate gradient algorithm with Tikhonov regularization (left) and the ALS algorithm of [7] (regularization + projection) (right).

Appendix

Our aim is now to calculate the derivatives of the penalty terms: $\mathsf{d}f(\mathbf{A}) = \mathsf{d}\|e^{-\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2$. Considering a square $N \times N$ matrix $\mathbf{A}$, we have the following property: $\mathbf{D}_0.\, e^{\mathbf{A}} = \sum_{k=0}^{\infty} \frac{\mathbf{A}^k}{k!}$. Or:

$$\|e^{-\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 = \mathsf{trace}\{(e^{-\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}})^T . e^{-\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\} = \mathsf{trace}\{e^{-2\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\}.$$

where $\mathsf{trace}\{\cdot\}$ stands for the trace of the matrix given in argument. Using $\mathbf{D}_0$, we have:

$$\|e^{-2\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 = \sum_{k=0}^{\infty} \frac{(-2\gamma)^k}{k!} \mathsf{trace}\{[\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}]^k\}$$

Considering only the three first terms in the series expansion, we find that:

$$\begin{aligned}
&\mathsf{d}\|e^{-\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}}\|_F^2 \\
&= \frac{\partial \left[\mathbf{I}_{IF} - 2\gamma \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\} + 2\gamma^2 [\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}]^2\right]}{\partial \mathsf{vec}\{\mathbf{A}\}} \mathsf{dvec}\{\mathbf{A}\} \\
&\simeq \left[-2\gamma \mathbf{I}_{IF} + 4\gamma^2 \mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\}\right] \mathsf{dvec}\{\mathbf{A}\} \\
&\simeq 4\gamma^2 \left[\mathsf{diag}\{\mathsf{vec}\{\mathbf{A}\}\} - \frac{\mathbf{I}_{IF}}{2\gamma}\right] \mathsf{dvec}\{\mathbf{A}\} \qquad (20)
\end{aligned}$$

or in its matrix (and not vectorial) form $4\gamma^2 \left[\mathbf{A} - \frac{\mathbf{1}_{I,F}}{2\gamma}\right] \mathsf{d}\mathbf{A}$.

7. REFERENCES

[1] P. Comon, "Tensors, usefulness and unexpected properties," in *IEEE Workshop on Statistical Signal Processing (SSP'09)*, Cardiff, UK, Sept. 1-3 2009, keynote.

[2] P. Comon, X. Luciani, and A. L. F. De Almeida, "Tensor decompositions, alternating least squares and other tales," *Jour. Chemometrics*, vol. 23, pp. 393–405, Aug. 2009.

[3] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *Siam Review*, vol. 51, no. 3, pp. 455–500, Sept. 2009.

[4] A. Smilde, R. Bro, and P. Geladi, *Multi-Way Analysis with applications in the chemical sciences*, Wiley, 2004.

[5] P. Paatero, "Construction and analysis of degenerate Parafac models," *J. Chemometrics*, vol. 14, no. 3, pp. 285–299, 2000.

[6] L. H. Lim and P. Comon, "Nonnegative approximations of nonnegative tensors," *Jour. Chemometrics*, vol. 23, pp. 432–441, Aug. 2009.

[7] A. Cichocki, R. Zdunek, A. H. Phan, and S. I. Amari, *Non negative matrix and tensor factorizations: Application to exploratory multi-way data analysis and blind separation*, Wiley, 2009.

[8] R. Bro, "Parafac: tutorial and applications," *Chemometr. Intell. Lab.*, vol. 38, pp. 149–171, 1997.

[9] Q. Zhang, H. Wang, R. Plemmons, and P. Pauca, "Tensors methods for hyperspectral data processing: a space object identification study," *Journal of Optical Society of America A*, Dec. 2008.

[10] F. L. Hitchcock, "The expression of a tensor or a polyadic as a sum of products," *J. Math. and Phys.*, vol. 6, pp. 165–189, 1927.

[11] T. Lickteig, "Typical tensorial rank," *Linear Algebra Appl.*, vol. 69, pp. 95–120, 1985.

[12] E. Polak, *Optimization algorithms and consistent approximations*, Springer, 1997.

[13] J.-P. Royer, N. Thirion-Moreau, and P. Comon, "Nonnegative 3-way tensors factorization via conjugate gradient with globally optimal stepsize," in *submitted to ICASSP'11*, Prague, Czech Republic, May 2011.