BEST LOW MULTILINEAR RANK APPROXIMATION OF HIGHER-ORDER TENSORS, BASED ON THE RIEMANNIAN TRUST-REGION SCHEME *

MARIYA ISHTEVA†, LIEVEN DE LATHAUWER†‡,
P.-A. ABSIL§, AND SABINE VAN HUFFEL†

Abstract. Higher-order tensors are used in many application fields, such as statistics, signal processing and scientific computing. Efficient and reliable algorithms for manipulating these multi-way arrays are thus required. In this paper, we focus on the best rank-(R_1, R_2, R_3) approximation of third-order tensors. We propose a new iterative algorithm based on the trust-region scheme. The tensor approximation problem is expressed as minimization of a cost function on a product of three Grassmann manifolds. We apply the Riemannian trust-region scheme, using the truncated conjugate-gradient method for solving the trust-region subproblem. Making use of second order information of the cost function, superlinear convergence is achieved. If the stopping criterion of the subproblem is chosen adequately, the local convergence rate is quadratic. We compare this new method with the well-known higher-order orthogonal iteration method and discuss the advantages over Newton-type methods.

Key words. multilinear algebra, higher-order tensor, rank reduction, singular value decomposition, trust-region scheme, Riemannian manifold, Grassmann manifold.

AMS subject classifications. 15A69, 65F99, 90C48, 49M37, 53B20

1. Introduction. Higher-order tensors are multi-way arrays of numbers, i.e., higher-order generalizations of vectors (first order) and matrices (second order). They are used as a tool in higher-order statistics [32, 28, 43, 31], independent component analysis [10, 11, 14, 8] and harmonic analysis [36, 35]. Fields of application of higher-order tensors also include chemometrics [41], biomedical signal processing [30, 15, 16, 4], telecommunications [39, 40, 10], scientific computing [7, 6, 20, 34] and image processing [47]. The main applications of the tensor approximation discussed in this paper are dimensionality reduction of tensors with high dimensions [14, 4, 15, 16, 26, 41, 25] and signal subspace estimation [36, 35, 26, 41, 25, 19].

Even some basic matrix concepts cannot be generalized to higher-order tensors in a unique manner. For example, generalization of different matrix-rank properties leads to different tensor-rank definitions. In this paper, we consider a definition of tensor-rank introduced in [21, 22] as a direct generalization of row and column rank of a matrix. The so-called mode-n vectors ($n = 1, 2, \ldots, N$) of an Nth-order tensor

*Research supported by: (1) Research Council K.U.Leuven: GOA-Ambiorics, CoE EF/05/006 Optimization in Engineering (OPTEC), (2) F.W.O.: (a) project G.0321.06, (b) Research Communities ICCoS, ANMMM and MLDM, (3) the Belgian Federal Science Policy Office: IUAP P6/04 (DYSCO, "Dynamical systems, control and optimization", 2007–2011), (4) EU: ERNSI. M. Ishteva is supported by a K.U.Leuven doctoral scholarship (OE/06/25, OE/07/17, OE/08/007), L. De Lathauwer is supported by "Impulsfinanciering Campus Kortrijk (2007-2012)(CIF1)" and STRT1/08/023. The scientific responsibility rests with the authors.

†Department of Electrical Engineering - ESAT/SCD, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10/2446, B-3001 Leuven, Belgium, tel.: +32-16-321143, +32-16-328651, +32-16-321703, fax: +32-16-321970 (Mariya.Ishteva@esat.kuleuven.be, Lieven.DeLathauwer@esat.kuleuven.be, Sabine.VanHuffel@esat.kuleuven.be).

‡Group Science, Engineering and Technology, Katholieke Universiteit Leuven Campus Kortrijk, E. Sabbelaan 53, 8500 Kortrijk, Belgium, tel.: +32-56-326062, fax: +32-56-246999 (Lieven.DeLathauwer@kuleuven-kortrijk.be).

§Department of Mathematical Engineering, Université catholique de Louvain, Bâtiment Euler - Parking 13, Av. Georges Lemaître 4, B-1348 Louvain-la-Neuve, Belgium, tel.: +32-10-472597, fax: +32-10-472180 (http://www.inma.ucl.ac.be/∼absil/).

are vectors obtained by varying the n-th index, while keeping the other indices fixed. The dimension of the vector space spanned by the mode-n vectors is called "mode-n rank". The multilinear rank of a tensor is then the $n-$tuple of the mode-n ranks. Contrary to the matrix case, different mode-n ranks are not necessarily equal to each other. In this paper, we look for the best low rank-$(R_1, R_2, \ldots, R_N)$ approximation of an Nth-order tensor. This is a tensor with bounded mode-n ranks as close as possible to the original tensor. In the sense of multilinear rank, a generalization of the singular value decomposition, giving the best low rank approximation of a matrix, is the higher-order singular value decomposition (HOSVD) [12], also known as Tucker decomposition [45, 46]. Its truncation results in a suboptimal solution of the best low rank-$(R_1, R_2, \ldots, R_N)$ approximation problem, which can serve as a good starting point for iterative algorithms.

The most widely used iterative algorithm is the higher-order orthogonal iteration (HOOI) [13, 26, 27]. Recently, Newton-type algorithms for the best low multilinear rank approximation of tensors have been developed in [24, 17, 38], the first one being based on [2]. In this paper, we develop an algorithm based on the trust-region scheme on Riemannian manifolds; we refer to [1] for the theory of the Riemannian trust-region method and to the GenRTR[1] package for a generic Matlab implementation. A summary of the first version of our algorithm has been proposed in [23]. The main advantage over HOOI of our new algorithm is its superlinear convergence speed. Morover, for a well-chosen stopping criterion, the algorithm has quadratic convergence. The convergence speed of HOOI is only linear. On the other hand, the HOOI iterations are cheaper in general. However, if the multilinear rank is small compared to the tensor dimensions, which is often the case in practice, the cost per iteration of both algorithms is comparable. A detailed comparison of these two algorithms in given in Section 5. Compared to the Newton-type algorithms, the advantage of the proposed algorithm is its more stable behavior. For the trust-region algorithm, convergence to stationary points is guaranteed for all initial points. This is not the case for the Newton-type methods. Moreover, in practice, the trust-region algorithm converges to a local minimum, whereas the Newton-type methods may converge to any type of stationary point (minimum, maximum or saddle point). Finally, we mention other related methods that have appeared in the literature. A Krylov method has been proposed in [37] for large sparse tensors. Fast HOSVD algorithms for symmetric, Toeplitz and Hankel tensors have been proposed in [5]. For tensors with large dimensions, a Tucker-type decomposition is developed in [34]. In the latter paper, it is required that a good Tucker approximation exists with low multilinear rank. The computation then involves only few entries of the given tensor and takes only linear time.

This paper is organized as follows. In Section 2, some basic definitions are given and the problem of approximating a tensor by a low multilinear rank tensor is formulated in mathematical terms. Both HOSVD and HOOI are briefly presented. We consider third-order tensors, for simplicity. In Section 3, the basics of the trust-region method in $\mathbb{R}^n$, in a Euclidean space and on a Riemannian manifold are given. Our trust-region based algorithm for the best rank-(R_1, R_2, R_3) approximation of third-order tensors is described in section 4. Section 5 makes a comparison between the trust-region algorithm and HOOI. A comparison with Newton-type methods is made in Section 6. We draw our conclusions in Section 7.

Throughout this paper, we use the following notation. Tensors are denoted by

[1] http://www.math.fsu.edu/~cbaker/GenRTR/

calligraphic letters $(\mathcal{A}, \mathcal{B}, \ldots)$, bold-face capitals $(\mathbf{A}, \mathbf{B}, \ldots)$ correspond to matrices, and scalars are written as lower-case letters $(a, b, \ldots)$. This means that the elements of a third-order tensor $\mathcal{A}$ are $a_{ijk} = (\mathcal{A})_{ijk}$. Some special scalars and upper bounds are denoted by capital letters $(I, I_1, I_2, I_3, N, R, \ldots)$. We use "$\times$" for the Cartesian product of sets and "$\otimes$" for the Kronecker product. The identity matrix is $\mathbf{I}$, $O_R = \{\mathbf{X} \in \mathbb{R}^{R\times R} : \mathbf{X}^T\mathbf{X} = \mathbf{X}\mathbf{X}^T = \mathbf{I}\}$ denotes the orthogonal group (the set of all orthogonal $R \times R$ matrices), and $St(R, I) = \{\mathbf{X} \in \mathbb{R}^{I\times R} : \mathbf{X}^T\mathbf{X} = \mathbf{I}\}$ denotes the Stiefel manifold (the set of all column-wise orthonormal $I \times R$ matrices).

2. Problem formulation. In this section, we start with some basic definitions and notations. Further, we formulate the main problem and then we briefly summarize HOSVD and HOOI.

2.1. Basic definitions. We denote the *mode-n products* of a tensor $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$ with matrices $\mathbf{M}^{(n)} \in \mathbb{R}^{J_n\times I_n}$, $n = 1, 2, 3$ by $\mathcal{A} \bullet_n \mathbf{M}^{(n)}$, where

$$(\mathcal{A} \bullet_1 \mathbf{M}^{(1)})_{j_1 i_2 i_3} = \sum_{i_1} a_{i_1 i_2 i_3} m^{(1)}_{j_1 i_1},$$

$$(\mathcal{A} \bullet_2 \mathbf{M}^{(2)})_{i_1 j_2 i_3} = \sum_{i_2} a_{i_1 i_2 i_3} m^{(2)}_{j_2 i_2},$$

$$(\mathcal{A} \bullet_3 \mathbf{M}^{(3)})_{i_1 i_2 j_3} = \sum_{i_3} a_{i_1 i_2 i_3} m^{(3)}_{j_3 i_3}$$

and $1 \le i_n \le I_n$, $1 \le j_n \le J_n$. To make computations easier, it is convenient to define *matrix representations* $\mathbf{A}_{(n)}, n = 1, 2, 3$ of a tensor $\mathcal{A}$ in the following way

$$(\mathbf{A}_{(1)})_{i_1,(i_2-1)I_3+i_3} = (\mathbf{A}_{(2)})_{i_2,(i_3-1)I_1+i_1} = (\mathbf{A}_{(3)})_{i_3,(i_1-1)I_2+i_2} = a_{i_1 i_2 i_3},$$

with $1 \le i_1 \le I_1$, $1 \le i_2 \le I_2$, $1 \le i_3 \le I_3$. The *scalar product* of two tensors $\mathcal{A}$ and $\mathcal{B}$ is defined as $\langle \mathcal{A}, \mathcal{B}\rangle = \sum_{i_1}\sum_{i_2}\sum_{i_3} a_{i_1 i_2 i_3}\, b_{i_1 i_2 i_3}$. Finally, the *Frobenius norm* of a tensor $\mathcal{A}$ is $\|\mathcal{A}\| = \sqrt{\langle \mathcal{A}, \mathcal{A}\rangle}$.

2.2. The problem. In this paper, we look for the best rank-(R_1, R_2, R_3) approximation $\hat{\mathcal{A}} \in \mathbb{R}^{I_1\times I_2\times I_3}$ of an (unstructured) third-order tensor $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$. $\hat{\mathcal{A}}$ has to minimize the least-squares cost function $F : \mathbb{R}^{I_1\times I_2\times I_3} \to \mathbb{R}$,

$$F : \hat{\mathcal{A}} \mapsto \|\mathcal{A} - \hat{\mathcal{A}}\|^2 \tag{2.1}$$

under the constraints $\mathrm{rank}_1(\hat{\mathcal{A}}) \le R_1$, $\mathrm{rank}_2(\hat{\mathcal{A}}) \le R_2, \mathrm{rank}_3(\hat{\mathcal{A}}) \le R_3$, where $\mathrm{rank}_n(.)$ stands for the mode-n rank.

Instead of minimizing the cost function (2.1) we will solve the equivalent problem (see [13]) of maximizing the function $\overline{g} : St(R_1, I_1) \times St(R_2, I_2) \times St(R_3, I_3) \to \mathbb{R}$,

$$\overline{g} : (\mathbf{U}, \mathbf{V}, \mathbf{W}) \mapsto \|\mathcal{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T\|^2 \tag{2.2}$$

over the column-wise orthonormal matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$. It is sufficient to determine $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$ in order to optimize (2.1). Knowing these matrices, the optimal tensor $\hat{\mathcal{A}}$ is computed as

$$\hat{\mathcal{A}} = \mathcal{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W}, \tag{2.3}$$

where $\mathcal{B} \in \mathbb{R}^{R_1\times R_2\times R_3}$ is given by

$$\mathcal{B} = \mathcal{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T.$$

2.3. Higher-order singular value decomposition. For matrices, the solution of the best low rank approximation problem is given by the truncated singular value decomposition (SVD). The higher-order singular value decomposition (HOSVD) [12, 45, 46] is a generalization of SVD to higher-order tensors. It decomposes a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ as a product of a so-called core-tensor $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and three orthogonal matrices $\mathbf{U}^{(i)} \in \mathbb{R}^{I_i \times I_i}$, $i = 1, 2, 3$, i.e.,

$$\mathcal{A} = \mathcal{S} \bullet_1 \mathbf{U}^{(1)} \bullet_2 \mathbf{U}^{(2)} \bullet_3 \mathbf{U}^{(3)} .$$

In general, it is impossible to obtain a diagonal core tensor. However, the core tensor has certain properties implying a diagonal matrix in the second-order case.

Computing HOSVD is straightforward. The singular matrices $\mathbf{U}^{(i)}, i = 1, 2, 3$ are obtained as the matrices of left singular vectors of the matrix representations $\mathbf{A}_{(i)}, i = 1, 2, 3$. The core tensor is computed as

$$\mathcal{S} = \mathcal{A} \bullet_1 \mathbf{U}^{(1)^T} \bullet_2 \mathbf{U}^{(2)^T} \bullet_3 \mathbf{U}^{(3)^T} .$$

Unfortunately, in general, truncated HOSVD gives a suboptimal solution of (2.1). However, it is often a good starting value for iterative algorithms.

2.4. Higher-order orthogonal iteration. The higher-order orthogonal iteration method (HOOI) [13, 26, 27] is an alternating least-squares algorithm for finding the best rank-(R_1, R_2, R_3) approximation of a tensor. The initial matrices $\mathbf{U}_0$, $\mathbf{V}_0$, $\mathbf{W}_0$ are taken from the truncated HOSVD. At each step, one of the matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$ in (2.2) is updated, while the other two are kept constant. One cycles through updates of $\mathbf{U}, \mathbf{V}, \mathbf{W}$ in an alternating manner. In order to optimize $\mathbf{U}$, the left R_1-dimensional dominant subspace of $\mathbf{A}_{(1)}(\mathbf{V} \otimes \mathbf{W})$ has to be computed. Updates of $\mathbf{V}$ and $\mathbf{W}$ are obtained in a similar way. The convergence rate of HOOI is at most linear.

3. Riemannian trust-region scheme. In this section, we recall the idea behind the trust-region methods. We consider cost functions in $\mathbb{R}^n$, then move to a general Euclidean space and finally to the more general case of a Riemannian manifold.

The trust-region method (see [9, 33]) is an iterative method for minimizing a cost function. At each iteration step a quadratic model of the cost function is obtained. This model is assumed to be adequate in a region (the trust-region) around the current iterate. Then, an update is computed as the minimizer of the model in the trust region. The quality of the updated iterate is evaluated, it is consequently accepted or rejected and the trust-region radius is adjusted.

3.1. Basic trust-region method. In $\mathbb{R}^n$, given $f : \mathbb{R}^n \to \mathbb{R}$, the trust-region subproblem for finding the update $\eta \in \mathbb{R}^n$ for the current iterate $x \in \mathbb{R}^n$ is given by

$$\min_{\eta \in \mathbb{R}^n} m(\eta), \quad m(\eta) = f(x) + \partial f(x)\eta + \frac{1}{2}\eta^T \partial^2 f(x)\eta, \quad \|\eta\| \leq \Delta,$$

where $\partial f(x)$ is the gradient row vector of the first partial derivatives of $f(x)$, $\partial^2 f(x)$ is the Hessian matrix of the second partial derivatives of $f(x)$, and Δ is the trust-region radius. The quality of the model m is evaluated by means of the quotient

$$\rho = \frac{f(x) - f(x + \eta)}{m(0) - m(\eta)} . \tag{3.1}$$

If ρ is close to 0 or negative, then the model is very inaccurate, i.e., the step must be rejected and the trust-region radius must be reduced. If ρ is larger but still small, the step is accepted and the trust-region radius is reduced. Finally, if ρ is close to 1, then there is a good correspondence between the model and the cost function, the step is accepted and the trust-region radius can be increased.

In a Euclidean space E, if $f : E \to \mathbb{R}$ and $\langle \cdot, \cdot \rangle$ is the inner product in E, then the trust-region subproblem can be formulated as follows without resorting to a basis of E :

$$\begin{aligned} \min_{\eta \in E} m(\eta), \quad m(\eta) &= f(x) + \mathrm{D}f(x)[\eta] + \tfrac{1}{2}\mathrm{D}^2 f(x)[\eta, \eta] \\ &= f(x) + \langle \mathrm{grad} f(x), \eta \rangle + \tfrac{1}{2}\langle \mathrm{Hess} f(x)[\eta], \eta \rangle, \quad \langle \eta, \eta \rangle \leq \Delta_x^2, \end{aligned} \tag{3.2}$$

where $\mathrm{D}F(x)[z]$ denotes the directional derivative of the function F at x in the direction of z; $\mathrm{grad} f(x)$ and $\mathrm{Hess} f(x)$ stand for the gradient and the Hessian of f at x.

3.2. Trust-region method on a Riemannian manifold. Let now the function f be defined on a Riemannian manifold M. In this case, the trust-region subproblem at a point $x \in M$ takes place on the tangent space $T_x M$, which is a Euclidean space. The update vector $\eta \in T_x M$ is then a tangent vector. Its direction corresponds to the direction in which the next iterate is to be found and its length indicates how far in this direction to go. However, the new iterate should be on the manifold and not on the tangent space. The actual correspondence between η and the new point on the manifold is given through a smooth mapping, called retraction. The retraction at point x is denoted by R_x and its application on a vector $\xi \in T_x M$ is illustrated in Figure 3.1. The retraction has to satisfy certain conditions as described in [3, 1] but these conditions do not determine a unique function. The so-called exponential map is always an option. There are, however, often computationally better choices depending on the particular problem.

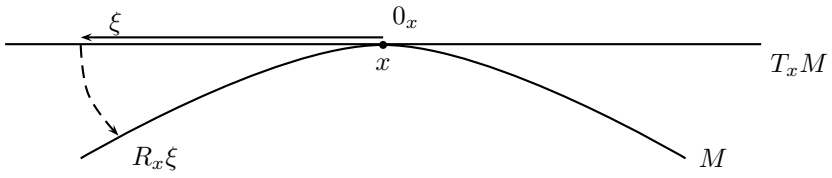


FIGURE 3.1. *Retraction* R_x

Given a cost function $f : M \to \mathbb{R}$ and a current iterate $x_k \in M$ we look for $x^* \in M$, such that $f(x^*)$ is minimal (see [1]). Using R_{x_k}, the minimization problem for f on M is locally mapped onto a minimization problem on $T_{x_k} M$ for the cost function $\hat{f}_{x_k} : T_{x_k} M \to \mathbb{R}$,

$$\hat{f}_{x_k} : \xi \mapsto f(R_{x_k}(\xi)).$$

Following (3.2), the trust-region subproblem for $T_{x_k}M$ becomes

$$\begin{aligned}\min_{\eta\in T_{x_k}M} m_{x_k}(\eta),\ \ m_{x_k}(\eta) &= \hat{f}_{x_k}(0_{x_k}) + \mathrm{D}\hat{f}_{x_k}(0_{x_k})[\eta] + \tfrac{1}{2}\mathrm{D}^2\hat{f}_{x_k}(0_{x_k})[\eta,\eta] \\ &= \hat{f}_{x_k}(0_{x_k}) + \langle \mathrm{grad}\hat{f}_{x_k}(0_{x_k}),\eta\rangle + \tfrac{1}{2}\langle \mathrm{Hess}\hat{f}_{x_k}(0_{x_k})[\eta],\eta\rangle \\ &= f(x_k) + \langle \mathrm{grad}f(x_k),\eta\rangle + \tfrac{1}{2}\langle \mathrm{Hess}f(x_k)[\eta],\eta\rangle\,,\end{aligned} \tag{3.3}$$

where $\langle\eta,\eta\rangle \leq \Delta_k^2$ and $\mathrm{Hess}f(x_k)$ is the Riemannian Hessian [3]. An approximate but sufficiently accurate solution to the trust-region subproblem is given by the truncated conjugate gradient algorithm (tCG) as suggested in [42, 44]. An advantage of this algorithm is that the Hessian matrix is not required explicitly but only its application to a tangent vector has to be computed. Other possible methods for (approximately) solving the trust-region subproblem (3.3) can be found in [29, 9]. The candidate for the new iterate is given by

$$x_+ = R_{x_k}(\eta_k),$$

where η_k is the solution of (3.3). The quotient

$$\rho_k = \frac{f(x_k) - f(R_{x_k}(\eta_k))}{m_{x_k}(0_{x_k}) - m_{x_k}(\eta_k)} = \frac{\hat{f}_{x_k}(0_{x_k}) - \hat{f}_{x_k}(\eta_k)}{m_{x_k}(0_{x_k}) - m_{x_k}(\eta_k)} \tag{3.4}$$

is evaluated in the same way as (3.1), the new iterate is accepted or rejected and the trust-region radius is updated.

Under mild conditions, the trust-region algorithm converges globally (i.e., for all initial points) to stationary points [1]. Furthermore, since the cost function is monotonically decreasing, it is expected that convergence to a local minimum is achieved in practice. Convergence to a saddle point or a local maximum is only observed in specially designed numerical examples.

4. Riemannian trust-region based rank-(R_1, R_2, R_3) approximation of a tensor. In this section, we apply the Riemannian trust-region scheme described in Section 3 to the problem (2.2). For this purpose, we need to go through the "checklist" in [1, §5.1] and give closed-form expressions for all the necessary components. Due to an invariance property of $\overline{g}$, we consider a Riemannian quotient manifold on which the computations take place. This will first be explained. Then, we present the closed-form expressions for the inner product on the manifold, the retraction function and the gradient and Hessian of the cost function, as required by [1, §5.1]. Note that in order to maximize $\overline{g}$ using the trust-region scheme, we, in fact, minimize the function $-\overline{g}$.

4.1. The quotient manifold. For third-order tensors, the cost function $\overline{g}$ from (2.2) can be computed as

$$\begin{aligned}\overline{g}(\mathbf{U},\mathbf{V},\mathbf{W}) = \|\mathcal{A}\bullet_1\mathbf{U}^T\bullet_2\mathbf{V}^T\bullet_3\mathbf{W}^T\|^2 &= \|\mathbf{U}^T\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})\|^2 \\ &= \|\mathbf{V}^T\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})\|^2 \\ &= \|\mathbf{W}^T\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})\|^2.\end{aligned} \tag{4.1}$$

It is easily seen that $\overline{g}$ has the following invariance property:

$$\overline{g}(\mathbf{U},\mathbf{V},\mathbf{W}) = \overline{g}(\mathbf{U}\mathbf{Q}^{(1)},\mathbf{V}\mathbf{Q}^{(2)},\mathbf{W}\mathbf{Q}^{(3)})\,,$$

where $\mathbf{Q}^{(i)} \in O_{R_i}, i = 1,2,3$ are orthogonal matrices. It is thus advisable to work on the manifold $M = St(R_1,I_1)/O_{R_1} \times St(R_2,I_2)/O_{R_2} \times St(R_3,I_3)/O_{R_3}$, which can be

thought of as a product of three Grassmann manifolds. The function $\overline{g}$ has one and only one projection $g : M \to \mathbb{R}$,

$$g : (\mathbf{U}O_{R_1}, \mathbf{V}O_{R_2}, \mathbf{W}O_{R_3}) \mapsto \overline{g}(\mathbf{U}, \mathbf{V}, \mathbf{W}). \tag{4.2}$$

Although the elements of M are more difficult to comprehend because they are equivalence classes of matrices, simplifications in the formulas and better convergence results follow, which justify (4.2). It will also be useful to define an extension of $\overline{g}$, $\tilde{g} : \mathbb{R}^{I_1 \times R_1} \times \mathbb{R}^{I_2 \times R_2} \times \mathbb{R}^{I_3 \times R_3} \to \mathbb{R}$,

$$\tilde{g} : (\mathbf{U}, \mathbf{V}, \mathbf{W}) \mapsto \|\mathcal{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T\|^2,$$

the domain of which is a vector space, so that all classical techniques of real analysis are applicable. Another advantage of the latter function is that it is suitable for computer computations since matrices are readily implemented as two-dimensional arrays.

Even though we can work with $\tilde{g}(\mathbf{U}, \mathbf{V}, \mathbf{W})$ as a representation of $g(\mathbf{U}, \mathbf{V}, \mathbf{W})$, we still have to keep in mind the structure of M. For this purpose, some additional notions have to be taken into account (see [3] for details). For simplicity, we first consider a manifold $M_1 = St(p, n)/O_p$. The vertical space at $\mathbf{X} \in St(p, n)$ is defined to be the tangent space at $\mathbf{X}$ to the equivalence class $\mathbf{X}O_p$, i.e.,

$$\mathcal{V}_{\mathbf{X}} = \{\mathbf{X}\mathbf{M} : \mathbf{M} = -\mathbf{M}^T \in \mathbb{R}^{p \times p}\}.$$

The tangent space to $St(p, n)$ at $\mathbf{X}$ is

$$\begin{aligned} T_{\mathbf{X}} St(p, n) &= \{\dot{\mathbf{X}} : \mathbf{X}^T\dot{\mathbf{X}} + \dot{\mathbf{X}}^T\mathbf{X} = 0\} \\ &= \{\mathbf{X}\mathbf{M} + \mathbf{X}_\perp\mathbf{K} : \mathbf{M} = -\mathbf{M}^T \in \mathbb{R}^{p \times p}, \mathbf{K} \in \mathbb{R}^{(n-p) \times p}\}, \end{aligned}$$

where $\mathbf{X}_\perp$ is any matrix such that $\begin{bmatrix} \mathbf{X} & \mathbf{X}_\perp \end{bmatrix} \in O_n$. Let $\dot{\mathbf{X}}_1, \dot{\mathbf{X}}_2 \in T_{\mathbf{X}} St(p, n)$. We define an inner product on $T_{\mathbf{X}} St(p, n)$ in the following way

$$\langle \dot{\mathbf{X}}_1, \dot{\mathbf{X}}_2 \rangle_{\mathbf{X}} = \operatorname{trace}(\dot{\mathbf{X}}_1^T \dot{\mathbf{X}}_2).$$

We choose the horizontal space at $\mathbf{X}$ to be the orthogonal complement of the vertical space in $T_{\mathbf{X}} St(p, n)$,

$$\mathcal{H}_{\mathbf{X}} = \{\mathbf{X}_\perp \mathbf{K} : \mathbf{K} \in \mathbb{R}^{(n-p) \times p}\}.$$

Given $\mathbf{Z}_{\mathbf{X}O_p} \in T_{\mathbf{X}O_p} M_1$, there exists a unique $\overline{\mathbf{Z}}_{\mathbf{X}} \in \mathcal{H}_{\mathbf{X}}$, such that $\mathrm{D}\pi(\mathbf{X})[\overline{\mathbf{Z}}_{\mathbf{X}}] = \mathbf{Z}_{\mathbf{X}O_p}$, where

$$\begin{array}{rccc} \pi : & St(p, n) & \to & M_1 \\ & \mathbf{X} & \mapsto & \mathbf{X}O_p \end{array}$$

is the quotient map. $\overline{\mathbf{Z}}_{\mathbf{X}}$ is termed the horizontal lift of $\mathbf{Z}_{\mathbf{X}O_p}$ at $\mathbf{X}$. It can be shown that $\overline{\mathbf{Z}}_{\mathbf{X}\mathbf{Q}} = \overline{\mathbf{Z}}_{\mathbf{X}}\mathbf{Q}$ for all $\mathbf{Q} \in O_p$. Observe that $\langle \overline{\mathbf{Z}}^{(1)}_{\mathbf{X}\mathbf{Q}}, \overline{\mathbf{Z}}^{(2)}_{\mathbf{X}\mathbf{Q}} \rangle = \langle \overline{\mathbf{Z}}^{(1)}_{\mathbf{X}}, \overline{\mathbf{Z}}^{(2)}_{\mathbf{X}} \rangle$, hence $\langle\langle \mathbf{Z}^{(1)}_{\mathbf{X}O_p}, \mathbf{Z}^{(2)}_{\mathbf{X}O_p} \rangle\rangle = \langle \overline{\mathbf{Z}}^{(1)}_{\mathbf{X}}, \overline{\mathbf{Z}}^{(2)}_{\mathbf{X}} \rangle$ is a well-defined inner product on $T_{\mathbf{X}O_p} M_1$. Finally, the orthogonal projection onto the horizontal space at $\mathbf{X}$ is given by

$$P_{\mathbf{X}}(\dot{\mathbf{X}}) = (\mathbf{I} - \mathbf{X}\mathbf{X}^T)\dot{\mathbf{X}}. \tag{4.3}$$

The elements of the vertical space can be thought of as variations that do not modify the column space of $\mathbf{X}$, whereas the elements of the horizontal space, applied to $\mathbf{X}$, modify the span of $\mathbf{X}$. The projection (4.3) reflects the fact that only variations of $\mathbf{X}$ that are orthogonal to the column space of $\mathbf{X}$ are important. Any other variations are irrelevant, since they are variations in the same equivalence class so they do not affect points in $St(p,n)/O_p$.

For the more general manifold M, we apply the following orthogonal projection

$$P_{(\mathbf{U},\mathbf{V},\mathbf{W})}(\dot{\mathbf{U}},\dot{\mathbf{V}},\dot{\mathbf{W}}) = ((\mathbf{I}-\mathbf{U}\mathbf{U}^T)\dot{\mathbf{U}},(\mathbf{I}-\mathbf{V}\mathbf{V}^T)\dot{\mathbf{V}},(\mathbf{I}-\mathbf{W}\mathbf{W}^T)\dot{\mathbf{W}}). \tag{4.4}$$

The inner product on the horizontal space $\mathcal{H}_{(\mathbf{U},\mathbf{V},\mathbf{W})}$ is then defined as the sum of the inner products of its components. A *retraction* R defines a unique point on the manifold corresponding to a given tangent vector $(\mathbf{Z}_{\mathbf{U}O_{R_1}},\mathbf{Z}_{\mathbf{V}O_{R_2}},\mathbf{Z}_{\mathbf{W}O_{R_3}})$ at a point $(\mathbf{U}O_{R_1},\mathbf{V}O_{R_2},\mathbf{W}O_{R_3})$. We use the following definition [3, §4.1.2]:

$$\begin{aligned}&R_{(\mathbf{U}O_{R_1},\mathbf{V}O_{R_2},\mathbf{W}O_{R_3})}(\mathbf{Z}_{\mathbf{U}O_{R_1}},\mathbf{Z}_{\mathbf{V}O_{R_2}},\mathbf{Z}_{\mathbf{W}O_{R_3}})\\ &= (\mathrm{qf}(\mathbf{U}+\overline{\mathbf{Z}}_{\mathbf{U}})O_{R_1},\mathrm{qf}(\mathbf{V}+\overline{\mathbf{Z}}_{\mathbf{V}})O_{R_2},\mathrm{qf}(\mathbf{W}+\overline{\mathbf{Z}}_{\mathbf{W}})O_{R_3}),\end{aligned}$$

where qf denotes the Q factor of the thin QR decomposition [18]. As mentioned in the previous section, other possibilities exist for the retraction function. However, our choice is computationally efficient and motivated by the fact that we are only interested in column spaces of the matrices $\mathbf{U},\mathbf{V}$ and $\mathbf{W}$ and not in their actual values.

Finally, closed-form expressions for the gradient $\mathrm{grad}\, g$, and the Hessian $\mathrm{Hess}\, g$ are obtained in the next two subsections.

From now on, we abuse notation and use g for $\overline{g}$, $\mathbf{X}$ for $\mathbf{X}O_p$, $\mathbf{Z}_{\mathbf{X}}$ for $\overline{\mathbf{Z}}_{\mathbf{X}}$ and $\langle\cdot,\cdot\rangle$ for $\langle\langle\cdot,\cdot\rangle\rangle$. This is admissible since $g(\mathbf{X}O_p)=\overline{g}(\mathbf{X})$ and $\langle\langle\mathbf{Z}^{(1)}_{\mathbf{X}O_p},\mathbf{Z}^{(2)}_{\mathbf{X}O_p}\rangle\rangle = \langle\overline{\mathbf{Z}}^{(1)}_{\mathbf{X}},\overline{\mathbf{Z}}^{(2)}_{\mathbf{X}}\rangle$.

4.2. Computation of the gradient. We have the following link between the *gradient* of g and the gradient of $\tilde{g}$ [3, (3.39)]

$$\mathrm{grad} g(\mathbf{U},\mathbf{V},\mathbf{W}) = P_{(\mathbf{U},\mathbf{V},\mathbf{W})}\mathrm{grad}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W}), \tag{4.5}$$

where $P_{(\mathbf{U},\mathbf{V},\mathbf{W})}(\mathbf{Z}_{\mathbf{U}},\mathbf{Z}_{\mathbf{V}},\mathbf{Z}_{\mathbf{W}})$ projects the columns of $\mathbf{Z}_{\mathbf{U}},\mathbf{Z}_{\mathbf{V}}$ and $\mathbf{Z}_{\mathbf{W}}$ onto the orthogonal complements of the column spaces of $\mathbf{U},\mathbf{V}$ and $\mathbf{W}$ respectively. Next, based on (4.1), we derive the differential of $\tilde{g}$ as

$$\begin{aligned}\mathrm{D}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z}_{\mathbf{U}},\mathbf{Z}_{\mathbf{V}},\mathbf{Z}_{\mathbf{W}})] &= 2\mathrm{trace}(\mathbf{Z}_{\mathbf{U}}^T\,\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U})\\ &\quad +2\mathrm{trace}(\mathbf{Z}_{\mathbf{V}}^T\,\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V})\\ &\quad +2\mathrm{trace}(\mathbf{Z}_{\mathbf{W}}^T\,\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{W}).\end{aligned}$$

The formula for the gradient of $\tilde{g}$ follows directly from the latter expression, namely

$$\begin{aligned}\mathrm{grad}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W}) &= (\mathrm{grad}_{\mathbf{U}}\tilde{g},\mathrm{grad}_{\mathbf{V}}\tilde{g},\mathrm{grad}_{\mathbf{W}}\tilde{g})\\ &= (2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U},\\ &\quad 2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V},\\ &\quad 2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T\,(\mathbf{A}_{(3)})^T\mathbf{W}).\end{aligned} \tag{4.6}$$

Combining (4.5), (4.4) and (4.6), a closed form expression for the gradient of g is obtained.

4.3. Computation of the Hessian. A starting point for computing the *Hessian* of g is the following formula [3, (7.2) and Prop. 5.3.4]

$$\text{Hess } g(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] = P_{(\mathbf{U},\mathbf{V},\mathbf{W})}\text{D}(\text{grad}g)(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]. \tag{4.7}$$

Taking into account equations (4.5) and (4.4), we can transform the first component of $\text{D}(\text{grad}g)(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]$ to

$$\begin{aligned}
&\text{D}((\mathbf{I}-\mathbf{U}\mathbf{U}^T)\text{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]\\
&= \text{D}(\text{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \text{D}(\mathbf{U}\mathbf{U}^T\text{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]\\
&= \text{D}(\text{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_U}\mathbf{U}^T\text{grad}_{\mathbf{U}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\\
&\quad -\mathbf{U}\text{D}\{\mathbf{U}^T\text{grad}_{\mathbf{U}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\}[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]\,.
\end{aligned}$$

Similar expressions are obtained for the other two components. Substituting these expressions in (4.7), using (4.4) and the equality $(\mathbf{I}-\mathbf{U}\mathbf{U}^T)\mathbf{U}=0$, (4.7) is simplified to

$$\begin{aligned}
&\text{Hess } g(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] =\\
&\Big((\mathbf{I}-\mathbf{U}\mathbf{U}^T)\Big(\text{D}(\text{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_U}\mathbf{U}^T\text{grad}_{\mathbf{U}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\Big),\\
&\ (\mathbf{I}-\mathbf{V}\mathbf{V}^T)\Big(\text{D}(\text{grad}_{\mathbf{V}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_V}\mathbf{V}^T\text{grad}_{\mathbf{V}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\Big),\\
&(\mathbf{I}-\mathbf{W}\mathbf{W}^T)\Big(\text{D}(\text{grad}_{\mathbf{W}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] - \mathbf{Z_W}\mathbf{W}^T\text{grad}_{\mathbf{W}}\tilde{g}(\mathbf{U},\mathbf{V},\mathbf{W})\Big)\Big).
\end{aligned} \tag{4.8}$$

The only quantities in (4.8) that still lack a closed-form expression are of the form $\text{D}(\text{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]$. From (4.6) we have:

$$\begin{aligned}
\text{D}(\text{grad}_{\mathbf{U}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \;=\;& 2\mathbf{A}_{(1)}(\mathbf{Z_V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U}\\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{Z_W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U}\\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{Z_V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{U}\\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{Z_W})^T(\mathbf{A}_{(1)})^T\mathbf{U}\\
&+2\mathbf{A}_{(1)}(\mathbf{V}\otimes\mathbf{W})(\mathbf{V}\otimes\mathbf{W})^T(\mathbf{A}_{(1)})^T\mathbf{Z_U}\,.
\end{aligned}$$

By analogy,

$$\begin{aligned}
\text{D}(\text{grad}_{\mathbf{V}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \;=\;& 2\mathbf{A}_{(2)}(\mathbf{Z_W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V}\\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{Z_U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V}\\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{Z_W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{V}\\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{Z_U})^T(\mathbf{A}_{(2)})^T\mathbf{V}\\
&+2\mathbf{A}_{(2)}(\mathbf{W}\otimes\mathbf{U})(\mathbf{W}\otimes\mathbf{U})^T(\mathbf{A}_{(2)})^T\mathbf{Z_V}
\end{aligned}$$

and

$$\begin{aligned}
\text{D}(\text{grad}_{\mathbf{W}}\tilde{g})(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})] \;=\;& 2\mathbf{A}_{(3)}(\mathbf{Z_U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{W}\\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{Z_V})(\mathbf{U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{W}\\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{Z_U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{W}\\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{Z_V})^T(\mathbf{A}_{(3)})^T\mathbf{W}\\
&+2\mathbf{A}_{(3)}(\mathbf{U}\otimes\mathbf{V})(\mathbf{U}\otimes\mathbf{V})^T(\mathbf{A}_{(3)})^T\mathbf{Z_W}\,.
\end{aligned}$$

A closed form expression for $\text{Hess } g(\mathbf{U},\mathbf{V},\mathbf{W})[(\mathbf{Z_U},\mathbf{Z_V},\mathbf{Z_W})]$ is derived by substituting the last three expressions and (4.6) in (4.8).

4.4. Summary of the algorithm. In this subsection, we present a summary of the whole algorithm. In order to simplify the presentation, let $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$ and $\mathbf{Z} = (\mathbf{Z_U}, \mathbf{Z_V}, \mathbf{Z_W})$.

1. Obtain initial iterate $\mathbf{X}_0 \in M$, e.g., using the truncated HOSVD from Section 2.3.
   Set initial value for Δ_0.
2. Iterate until convergence ($k = 0, 1, 2, \ldots$)
   - Acquire a solution $\mathbf{Z}_k$ of the trust-region subproblem

$$\min_{\mathbf{Z} \in T_{\mathbf{X}_k} M} m_{\mathbf{X}_k}(\mathbf{Z}), \quad \text{subject to } \langle \mathbf{Z}, \mathbf{Z} \rangle \leq \Delta_k^2,$$

     where

$$m_{\mathbf{X}_k}(\mathbf{Z}) = -g(\mathbf{X}_k) + \langle -\operatorname{grad} g(\mathbf{X}_k), \mathbf{Z} \rangle + \frac{1}{2} \langle -\operatorname{Hess} g(\mathbf{X}_k)[\mathbf{Z}], \mathbf{Z} \rangle.$$

     An approximate solution is given by the tCG algorithm.
   - Form the quotient

$$\rho_k = \frac{-g(\mathbf{X}_k) + g(R_{\mathbf{X}_k}(\mathbf{Z}_k))}{m_{\mathbf{X}_k}(0_{\mathbf{X}_k}) - m_{\mathbf{X}_k}(\mathbf{Z}_k)}.$$

     If ρ_k is big enough, $\mathbf{X}_{k+1} = R_{\mathbf{X}_k}(\mathbf{Z}_k)$; otherwise $\mathbf{X}_{k+1} = \mathbf{X}_k$.
     If ρ_k is too small, $\Delta_{k+1} = \frac{1}{4}\Delta_k$; if ρ_k is large $\Delta_{k+1} = 2\Delta_k$; otherwise $\Delta_{k+1} = \Delta_k$.
3. The best low multilinear rank approximation $\hat{\mathcal{A}}$ is given by

$$\hat{\mathcal{A}} = \mathcal{B} \bullet_1 \mathbf{U} \bullet_2 \mathbf{V} \bullet_3 \mathbf{W},$$

   where

$$\mathcal{B} = \mathcal{A} \bullet_1 \mathbf{U}^T \bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T$$

   and $\mathbf{X} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$ is the converged triplet from Step 2.

5. Comparison with HOOI. In this section, we perform numerical experiments illustrating the properties of the proposed algorithm. All experiments are carried out in parallel for the trust-region and HOOI algorithms. The simulations were run using MATLAB. The machine precision is approximately 2.10^{-16}.

5.1. Convergence speed. We start our comparison with the convergence speed of both algorithms. HOOI only has linear convergence, whereas the local rate of convergence of the trust-region based algorithm is superlinear [1]. This improvement in the trust-region based algorithm is due to using second order information of the cost function. For illustration, we considered tensors $\mathcal{A} \in \mathbb{R}^{20 \times 20 \times 20}$ with elements uniformly distributed in the interval $[0, 1]$. We set $R_1 = R_2 = R_3 = 5$. Initial matrices were taken from the truncated HOSVD. Then both algorithms were run until convergence ($\|\operatorname{grad} g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) or until a maximum of 100 iterations was reached. In Fig. 5.1, the results of this experiment are shown for one representative example. Three stages of the trust-region algorithm evolution can be distinguished. Due to a large initial trust-region radius, a poor correspondence between the cost function and the quadratic model was obtained for the first two updates and as a result they were rejected. In the plots this corresponds to constant values of the cost

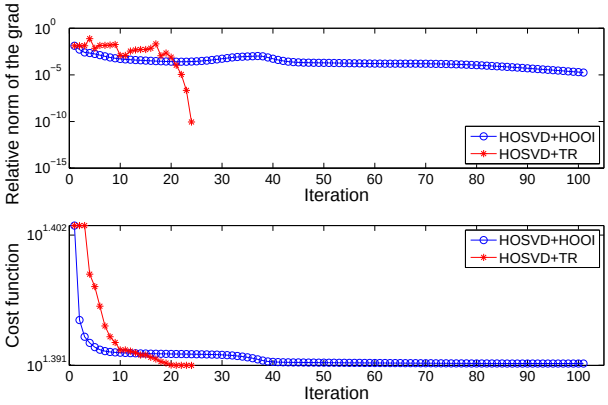
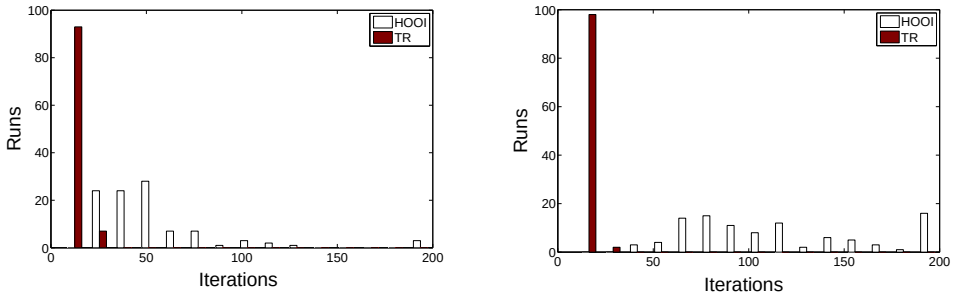


FIGURE 5.1. *Evolution of the relative norm of the gradient* $\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X})$ *and the cost function (2.1). The tensor* $\mathcal{A} \in \mathbb{R}^{20\times 20\times 20}$ *has elements uniformly distributed in the interval* $[0, 1]$*;* $R_1 = R_2 = R_3 = 5$*. The starting values are taken from the truncated HOSVD.*

function and of the relative norm of the gradient. After reducing the trust-region radius, the values of the quotient (3.4) were considerably improved, so the majority of the following steps were successful. As a consequence, during this second phase, the value of the cost function decreased monotonically as it can be seen from the bottom plot. The relative norm of the gradient did not decrease monotonically because of the complex nature of the cost function. However, its decrease is not required in order to improve the cost function value. Finally, in the last steps, the iterates were close enough to the solution for superlinear convergence, as can be observed in the gradient plot.

The number of iterations necessary for the algorithms to converge is strongly related to the convergence speed. In Fig. 5.2, we compare the two algorithms for



(a) $(R_1, R_2, R_3) = (7, 8, 9)$ (b) $(R_1, R_2, R_3) = (2, 2, 2)$

FIGURE 5.2. *Number of iterations for HOOI and trust-region algorithms for tensors* $\mathcal{A} \in \mathbb{R}^{10\times 10\times 10}$ *with elements taken from a normal distribution with zero mean and unit standard deviation. Performed are 100 runs with initial values* $\mathbf{U}_0, \mathbf{V}_0$*, and* $\mathbf{W}_0$ *taken from the truncated HOSVD. Each run was stopped if the corresponding algorithm did not converge in 200 iterations* ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$).

tensors $\mathcal{A} \in \mathbb{R}^{10\times 10\times 10}$ with elements taken from a normal distribution with zero

mean and unit standard deviation and for two multilinear ranks – $(7, 8, 9)$ and $(2, 2, 2)$, respectively. The results shown are for 100 runs and initial values were taken from the truncated HOSVD. Both algorithms were run until convergence ($\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$) or until a maximum of 200 iterations was reached. The trust-region algorithm did not reach the maximum number of iterations in neither test. HOOI had 3 runs in (a) and 13 runs in (b) with 200 iterations. It can be concluded that in order to converge, the trust-region based algorithm requires less iterations than HOOI. The difference seems to be stronger for smaller multilinear rank.

5.2. Computational cost. While the trust-region method converges faster than HOOI, the cost for one iteration is higher than the cost for one HOOI iteration, which is $O(I^3R + IR^4 + R^6)$ as shown in [17, 24]. For simplicity, we assume that $R_1 = R_2 = R_3 = R$ and $I_1 = I_2 = I_3 = I$.

In order to compute the cost for one trust-region iteration, we need to estimate the cost of one inner tCG iteration and multiply it by the number of tCG iterations within one trust-region step. The cost for the operations outside the inner tCG loop is smaller and can be neglected. The computationally heaviest operation in the trust-region algorithm is the computation of the Hessian, or more precisely, the application of the Hessian on a tangent vector. This is dominated by expressions such as $\mathbf{A}_{(1)}(\mathbf{Z_V} \otimes \mathbf{W})$. Another representation of these expressions is $(\mathcal{A} \bullet_2 \mathbf{Z}_\mathbf{V}^T) \bullet_3 \mathbf{W}^T = (\mathcal{A} \bullet_3 \mathbf{W}^T) \bullet_2 \mathbf{Z}_\mathbf{V}^T$, which requires $O(I^3R)$ flops. Such expressions are computed in each inner iteration, i.e., in each step of tCG. However, the products $\mathcal{A}\bullet_1 \mathbf{U}^T$, $\mathcal{A}\bullet_2 \mathbf{V}^T$, $\mathcal{A}\bullet_3 \mathbf{W}^T$, $\mathcal{A}\bullet_2 \mathbf{V}^T \bullet_3 \mathbf{W}^T$, etc., which involve only the tensor $\mathcal{A}$ and the matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$ and which do not involve the components $\mathbf{Z_U}, \mathbf{Z_V}, \mathbf{Z_W}$ of tangent vectors, can be computed at the beginning of each trust-region step. They require $O(I^3R)$ flops. Thus, within the inner iterations only expressions of the form $(\mathcal{A}\bullet_3 \mathbf{W}^T)\bullet_2 \mathbf{Z}_\mathbf{V}^T$, given $\mathcal{A}\bullet_3 \mathbf{W}^T$, are computed. The cost of such an operation is only $O(I^2R^2)$. Finally, the maximum number of inner iteration per trust-region iteration is $3(I - R)R$, which equals the dimension of the manifold. This leads us to a total cost of $O(I^3R) + 3(I - R)R.O(I^2R^2) = O(I^3R^3)$ for one trust-region step. It should be taken into account that in applications, the values of R_n are often much smaller than the corresponding values of I_n.

Note, however, that one can reduce the computational cost of the trust-region algorithm without losing its fast local convergence rate. This can be done by choosing a stopping criterion for the inner iteration based on the gradient of the cost function as in [1]. In this case, few inner tCG steps are taken when the outer iterate is far away from the solution, i.e., when the gradient is large, and more inner tCG steps are taken close to the solution. This behavior is illustrated in Figure 5.3. The tCG inner iteration is stopped according to

$$\|r_j\| \leq \|r_0\| \min(\|r_0\|^\theta, \kappa),$$

where r_j is the residual at the jth step of tCG; see [1, §3] for details. We chose $\theta = 1$, which yields quadratic convergence of the trust-region algorithm. We used this inner stopping criterion for all our numerical experiments involving the trust-region method. For Figure 5.3, the entries of $\mathcal{A} \in \mathbb{R}^{20\times20\times20}$ are taken from a normal distribution with zero mean and unit standard deviation. We set $R_1 = R_3 = R_3 = 2$ and $\mathbf{U}_0, \mathbf{V}_0$, and $\mathbf{W}_0$ are obtained from the truncated HOSVD. The algorithm was stopped when $\|\text{grad } g(\mathbf{X})\|/g(\mathbf{X}) < 10^{-9}$.

6. Comparison with Newton-type methods. Besides HOOI, Newton-type methods for the best rank-(R_1, R_2, R_3) approximation have appeared in the literature [24, 17, 38]. These methods have local quadratic convergence rate and their

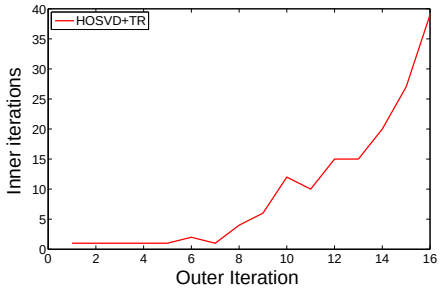


FIGURE 5.3. *Number of inner tCG steps per trust-region step. The tensor* $\mathcal{A} \in \mathbb{R}^{20\times 20\times 20}$ *has elements taken from a normal distribution with zero mean and unit standard deviation;* $R_1 = R_3 = R_3 = 2$ *and* $\mathbf{U}_0, \mathbf{V}_0$, *and* $\mathbf{W}_0$ *are taken from the truncated HOSVD. Few tCG iterations are performed in the first trust-region steps and more in the neighborhood of the solution.*

computational cost per iteration is of the same order as the one of the trust-region method. However, they are not globally convergent and depend strongly on the initial values of the matrices $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$. In practice, these methods might diverge. The truncated HOSVD often gives good initial values but sometimes these values are not good enough. On the other hand, the trust-region method converges globally to local optima except for very special examples that are artificially constructed.

Another advantage of the trust-region method is that the cost function is monotonically decreasing. This explains why convergence to saddle points or local maxima is not observed in practice. Newton methods do not distinguish between minima, maxima and saddle points. This means that if the stationary points are close to each other, even if a relatively good starting point is chosen, these algorithms might converge to a maximum or to a saddle point instead of to a minimum.

To illustrate the above, we considered tensors $\mathcal{A} \in \mathbb{R}^{20\times 20\times 20}$ with elements uniformly distributed in the interval $[0, 1]$ and estimated their best rank-$(5, 5, 5)$ approximation. We started from values taken from the truncated HOSVD and performed 20 iterations of HOOI. Then we ran HOOI, the geometric Newton [24] and the trust-region algorithms until convergence, with a maximum of 500 iterations. In Figure 6.1, one example is shown where HOOI and the trust-region algorithm both converge to a local minimum of (2.1) but the geometric Newton algorithm converges to a saddle point. For clarity, the figure presents only the iterations in the interval $[20, 100]$. The type of the end points was evaluated using the Hessian of the cost function. Cases like this seem rather rare. Often, the Newton-type and the trust-region algorithms perform in a similar way in the same setting, i.e., converge to a local minimum with a similar number of iterations. In our simulations, it also happened that both types of algorithms converged to a local minimum but that the Newton-type algorithm needed considerably more iterations. This occurred especially when the starting point of the Newton algorithm had an indefinite Hessian.

In tests with low multilinear rank tensors affected by additive noise with a modest signal-to-noise ratio, the performance of the Newton-type and trust-region algorithms was similar. We considered rank-$(5, 5, 5)$ tensors $\mathcal{T} \in \mathbb{R}^{20\times 20\times 20}$, affected by additive noise in the following way

$$\mathcal{A} = \mathcal{T}/\|\mathcal{T}\| + 0.1\, \mathcal{E}/\|\mathcal{E}\|, \tag{6.1}$$

in which $\mathcal{E} \in \mathbb{R}^{20\times 20\times 20}$ represents noise with elements uniformly distributed in the

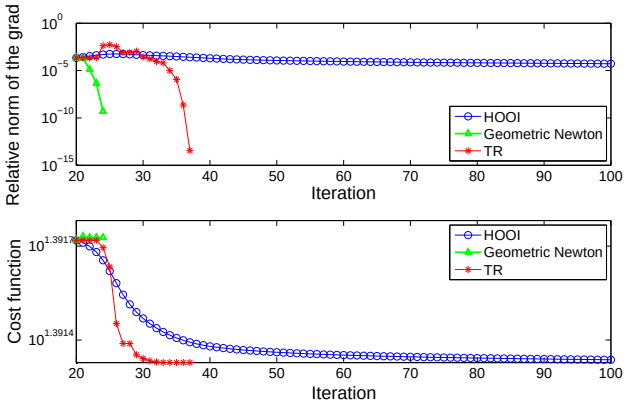


FIGURE 6.1. *Evolution of the relative norm of the gradient* $\|grad\ g(\mathbf{X})\|/g(\mathbf{X})$ *and the cost function (2.1). The tensor* $\mathcal{A} \in \mathbb{R}^{20\times 20\times 20}$ *has elements uniformly distributed in the interval* $[0, 1]$*;* $R_1 = R_2 = R_3 = 5$*. The starting point of the plots is obtained by initializing with matrices taken from the truncated HOSVD and performing 20 additional HOOI iterations.*

interval $[0, 1]$. The tensor $\mathcal{T}$ was constructed as the product of a tensor $\mathcal{C} \in \mathbb{R}^{5\times 5\times 5}$ and three matrices $\mathbf{M}^{(i)} \in \mathbb{R}^{20\times 5}$, $i = 1, 2, 3$, all with elements uniformly distributed in the interval $[0, 1]$, i.e.,

$$\mathcal{T} = \mathcal{C} \bullet_1 \mathbf{M}^{(1)} \bullet_2 \mathbf{M}^{(2)} \bullet_3 \mathbf{M}^{(3)}.$$

In Figure 6.2, one representative example is given. The initial matrices $\mathbf{U}_0$, $\mathbf{V}_0$

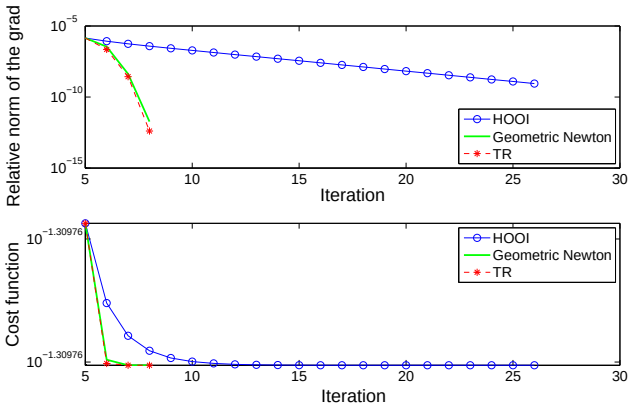


FIGURE 6.2. *Evolution of the relative norm of the gradient* $\|grad\ g(\mathbf{X})\|/g(\mathbf{X})$ *and the cost function (2.1). The tensor* $\mathcal{A} \in \mathbb{R}^{20\times 20\times 20}$ *is as in (6.1), with* $R_1 = R_2 = R_3 = 5$*. The starting point of the plots is obtained by initializing with matrices taken from the truncated HOSVD and performing 5 additional HOOI iterations.*

and $\mathbf{W}_0$ were taken from the truncated HOSVD and 5 additional HOOI iterations were performed. As it can be seen from the figure, the differential-geometric Newton method and the trust-region algorithm have similar convergence behavior whereas HOOI needs more but computationally less expensive iterations.

We conclude this section by reporting that in our simulations the Newton methods

in [24, 17] generally behaved in the same way, although for a particular dataset the behavior was sometimes different.

7. Conclusions. We have developed a new algorithm for computing the best rank-(R_1, R_2, R_3) approximation of higher-order tensors. It is based on the Riemannian trust-region scheme on quotient manifolds. In order to solve the trust-region subproblems, the truncated conjugate gradient method is applied. The algorithm converges globally to local minima of the cost function (2.1). Its convergence rate is superlinear and even quadratic, for a suitable stopping criterion.

In comparison with the higher-order orthogonal iteration method [13], the advantage of the new algorithm is its faster local convergence rate. The maximal computational cost per iteration of these two algorithms is comparable for small multilinear rank. This is the case in many applications. Moreover, the trust-region based algorithm reaches its maximum number of inner iterations and hence its maximal computational cost per iteration only in iterations close to the solution. Thus, the overall performance of the proposed method is to be preferred in many cases.

In the literature also Newton-type methods have been presented [24, 17, 38]. Their computational cost per iteration is of the same order as the one of the trust-region method. However, in rather difficult cases, these algorithms have problems distinguishing between minima, maxima and saddle points and might even diverge. The trust-region algorithm overcomes these difficulties.

REFERENCES

[1] P.-A. Absil, C. G. Baker, and K. A. Gallivan, *Trust-region methods on Riemannian manifolds*, Found. Comput. Math., 7 (2007), pp. 303–330.

[2] P.-A. Absil, M. Ishteva, L. De Lathauwer, and S. Van Huffel, *A geometric Newton method for Oja's vector field*, Neural Comput., 21 (2009), pp. 1415–1433.

[3] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*, Princeton University Press, Princeton, NJ, January 2008.

[4] E. Acar, C. A. Bingol, H. Bingol, R. Bro, and B. Yener, *Multiway analysis of epilepsy tensors*, ISMB 2007 Conference Proceedings, Bioinformatics, 23 (2007), pp. i10–i18.

[5] R. Badeau and R. Boyer, *Fast multilinear singular value decomposition for structured tensors*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1008–1021. Special Issue on Tensor Decompositions and Applications.

[6] G. Beylkin and M. J. Mohlenkamp, *Numerical operator calculus in higher dimensions*, Proc. Natl. Acad. Sci. USA, 99 (2002), pp. 10246–10251.

[7] ———, *Algorithms for numerical analysis in high dimensions*, SIAM J. Sci. Comput., 26 (2005), pp. 2133–2159.

[8] P. Comon, *Independent component analysis, a new concept?*, Signal Processing, 36 (1994), pp. 287–314. Special issue on Higher-Order Statistics.

[9] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *Trust-Region Methods*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, PA, 2000.

[10] L. De Lathauwer and J. Castaing, *Tensor-based techniques for the blind separation of DS-CDMA signals*, Signal Processing, 87 (2007), pp. 322–336. Special Issue on Tensor Signal Processing.

[11] L. De Lathauwer, B. De Moor, and J. Vandewalle, *An introduction to independent component analysis*, J. Chemometrics, 14 (2000), pp. 123–149. Special Issue: Multi-way Analysis.

[12] ———, *A multilinear singular value decomposition*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1253–1278.

[13] ———, *On the best rank-1 and rank-$(R_1, R_2, \ldots, R_N)$ approximation of higher-order tensors*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1324–1342.

[14] L. De Lathauwer and J. Vandewalle, *Dimensionality reduction in higher-order signal processing and rank-$(R_1, R_2, \ldots, R_N)$ reduction in multilinear algebra*, Linear Algebra Appl., 391 (2004), pp. 31–55. Special Issue on Linear Algebra in Signal and Image Processing.

[15] M. De Vos, L. De Lathauwer, B. Vanrumste, S. Van Huffel, and W. Van Paesschen, *Canonical decomposition of ictal scalp EEG and accurate source localization: Princi-*

ples and simulation study, Journal of Computational Intelligence and Neuroscience, 2007 (2007), pp. 1–10. Special Issue on EEG/MEG Signal Processing.

[16] M. De Vos, A. Vergult, L. De Lathauwer, W. De Clercq, S. Van Huffel, P. Dupont, A. Palmini, and W. Van Paesschen, *Canonical decomposition of ictal scalp EEG reliably detects the seizure onset zone*, NeuroImage, 37 (2007), pp. 844–854.

[17] L. Eldén and B. Savas, *A Newton–Grassmann method for computing the best multi-linear rank-(r_1, r_2, r_3) approximation of a tensor*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 248–271.

[18] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, 3rd ed., 1996.

[19] M. Haardt, F. Roemer, and G. Del Galdo, *Higher-order SVD-based subspace estimation to improve the parameter estimation accuracy in multidimensional harmonic retrieval problems*, IEEE Transactions on Signal Processing, 56 (2008), pp. 3198–3213.

[20] W. Hackbusch and B. Khoromskij, *Tensor-product approximation to multidimensional integral operators and Green's functions*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1233–1253. Special Issue on Tensor Decompositions and Applications.

[21] F. L. Hitchcock, *The expression of a tensor or a polyadic as a sum of products*, Journal of Mathematical Physics, 6 (1927), pp. 164–189.

[22] ———, *Multiple invariants and generalized rank of a p-way matrix or tensor*, Journal of Mathematical Physics, 7 (1927), pp. 39–79.

[23] M. Ishteva, L. De Lathauwer, P.-A. Absil, and S. Van Huffel, *Dimensionality reduction for higher-order tensors: algorithms and applications*, International Journal of Pure and Applied Mathematics, 42 (2008), pp. 337–343.

[24] ———, *Differential-geometric Newton method for the best rank-(R_1, R_2, R_3) approximation of tensors*, Numerical Algorithms, 51 (2009), pp. 179–194. Tributes to Gene H. Golub Part II.

[25] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Review, 51 (2009 (to appear)).

[26] P. M. Kroonenberg, *Applied Multiway Data Analysis*, Wiley, 2008.

[27] P. M. Kroonenberg and J. de Leeuw, *Principal component analysis of three-mode data by means of alternating least squares algorithms*, Psychometrika, 45 (1980), pp. 69–97.

[28] P. McCullagh, *Tensor Methods in Statistics*, Chapman and Hall, London, 1987.

[29] J. J. Moré and D. C. Sorensen, *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 553–572.

[30] M. Mørup, L. K. Hansen, C. S. Herrmann, J. Parnas, and S. M. Arnfred, *Parallel factor analysis as an exploratory tool for wavelet transformed event-related EEG*, NeuroImage, 29 (2006), pp. 938–947.

[31] C. L. Nikias and J. M. Mendel, *Signal processing with higher-order spectra*, IEEE Signal Process. Mag., 10 (1993), pp. 10–37.

[32] C. L. Nikias and A.P. Petropulu, *Higher-Order Spectra Analysis. A Nonlinear Signal Processing Framework*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[33] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Verlag, New York, 2nd ed., 2006. Springer Series in Operations Research.

[34] I. V. Oseledets, D. V. Savostianov, and E. E. Tyrtyshnikov, *Tucker dimensionality reduction for three-dimensional arrays in linear time*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 939–956. Special Issue on Tensor Decompositions and Applications.

[35] J.-M. Papy, L. De Lathauwer, and S. Van Huffel, *Exponential data fitting using multilinear algebra: The decimative case*, J. Chemometrics. in press; available on-line (DOI: 10.1002/cem.1212).

[36] ———, *Exponential data fitting using multilinear algebra: The single-channel and the multi-channel case*, Numer. Linear Algebra Appl., 12 (2005), pp. 809–826.

[37] B. Savas and L. Eldén, *Krylov subspace methods for tensor computations*, Tech. Report LITH-MAT-R-2009-02-SE, Department of Mathematics, Linköpings Universitet, 2009.

[38] B. Savas and L.-H. Lim, *Best multilinear rank approximation of tensors with quasi-Newton methods on Grassmannians*, Tech. Report LITH-MAT-R-2008-01-SE, Department of Mathematics, Linköpings Universitet, 2008.

[39] N. Sidiropoulos, R. Bro, and G. Giannakis, *Parallel factor analysis in sensor array processing*, IEEE Trans. Signal Process., 48 (2000), pp. 2377–2388.

[40] N. Sidiropoulos, G. Giannakis, and R. Bro, *Blind PARAFAC receivers for DS-CDMA systems*, IEEE Trans. Signal Process., 48 (2000), pp. 810–823.

[41] A. Smilde, R. Bro, and P. Geladi, *Multi-way Analysis. Applications in the Chemical Sciences*, John Wiley and Sons, Chichester, U.K., 2004.

[42] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp. 626–637.
[43] A. Swami and G. Giannakis, *Bibliography on HOS*, Signal Process., 60 (1997), pp. 65–126.
[44] Ph. L. Toint, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, I. S. Duff, ed., Academic Press, London, 1981, pp. 57–88.
[45] L. R. Tucker, *The extension of factor analysis to three-dimensional matrices*, in Contributions to mathematical psychology, H. Gulliksen and N. Frederiksen, eds., Holt, Rinehart & Winston, NY, 1964, pp. 109–127.
[46] ———, *Some mathematical notes on three-mode factor analysis*, Psychometrika, 31 (1966), pp. 279–311.
[47] M. A. O. Vasilescu and D. Terzopoulos, *Multilinear subspace analysis for image ensembles*, in Proc. Computer Vision and Pattern Recognition Conf. (CVPR '03), Madison, WI, vol. 2, 2003, pp. 93–99.