ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

A variable projection method for block term decomposition of higher-order tensors

Guillaume Olikier[1], P.-A. Absil[1], and Lieven De Lathauwer[2] *

1- Université catholique de Louvain - ICTEAM Institute
B-1348 Louvain-la-Neuve - Belgium

2- KU Leuven - Electrical Engineering Department (ESAT)
B-3000 Leuven - Belgium

Abstract. Higher-order tensors have become popular in many areas of applied mathematics such as statistics, scientific computing, signal processing or machine learning, notably thanks to the many possible ways of decomposing a tensor. In this paper, we focus on the best approximation in the least-squares sense of a higher-order tensor by a block term decomposition. Using variable projection, we express the tensor approximation problem as a minimization of a cost function on a Cartesian product of Stiefel manifolds. We present numerical experiments where variable projection makes a steepest-descent method approximately twice faster.

1 Introduction

Higher-order tensors have found numerous applications in signal processing and machine learning thanks to the many tensor decompositions available [1, 2]. In this paper, we focus on a recently introduced tensor decomposition called block term decomposition (BTD) [3, 4, 5]. The interest of working with BTD in blind signal separation was outlined in [6] and more specific examples are discussed in [7, 8, 9].

The BTD unifies the two most well known tensor decompositions which are the Tucker decomposition and the canonical polyadic decomposition (CPD). It also gives a unified view on how the basic concept of rank can be generalized from matrices to tensors. While in CPD, as well as in classical matrix decompositions, the components are rank-one terms, i.e., "atoms" of data, the terms in a BTD have "low" (multilinear) rank and can be thought of as "molecules" (consisting of several atoms) of data. Rank-one terms can only model data components that are proportional along columns, rows, ... and this assumption may not be realistic. On the other hand, block terms can model multidimensional

*This work was supported by (1) "Communauté française de Belgique - Actions de Recherche Concertées" (contract ARC 14/19-060), (2) Research Council KU Leuven: C1 project C16/15/059-nD, (3) F.W.O.: project G.0830.14N, G.0881.14N, (4) the Belgian Federal Science Policy Office: IUAP P7 (DYSCO II, Dynamical systems, control and optimization, 2012-2017), (5) the Fonds de la Recherche Scientifique – FNRS and the Fonds Wetenschappelijk Onderzoek – Vlaanderen under EOS Project no 30468160, (6) EU: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC Advanced Grant: BIOTENSORS (no. 339804). This paper reflects only the authors' views and the Union is not liable for any use that may be made of the contained information.

sources, variations around mean activity, mildly nonlinear phenomena, drifts of setting points, frequency shifts, mildly convolutive mixtures, and so on. Such a molecular analysis is not possible in the matrix setting. Furthermore, it turns out that, like CPDs, BTDs are still unique under mild conditions [4, 7].

In practice, it is more frequent to approximate a tensor by a BTD than to compute an exact BTD. More precisely, the problem of interest is to compute the best approximation in the least-squares sense of a higher-order tensor by a BTD. Only a few algorithms are currently available for this task. The Matlab toolbox Tensorlab [10] proposes the two following functions: (i) `btd_minf` uses L-BFGS with dogleg trust region (a quasi-Newton method), (ii) `btd_nls` uses nonlinear least squares by Gauss–Newton with dogleg trust region. Another available algorithm is the alternating least squares algorithm introduced in [5]. This algorithm is not included in Tensorlab and does not work better than `btd_nls` in general.

In this paper, we show that the performance of numerical methods can be improved using variable projection. Variable projection consists in exploiting the fact that, when the optimal value of some of the optimization variables is easy to find when the others are fixed, this optimal value can be injected in the objective function, yielding a new optimization problem where only the other variables appear. This technique has already been applied to the Tucker decomposition in [11] and exploited in [12, 13]. Here we extend it to the BTD approximation problem which is then expressed as a minimization of a cost function on a Cartesian product of Stiefel manifolds. Numerical experiments show that the gradient algorithm is sped up by almost a factor 2 for BTDs of two terms. This suggests that more sophisticated optimization algorithms such as those used in the Tensorlab functions could give much better performance if they were combined with variable projection. In the sequel, we focus on third-order tensors for simplicity but the generalization to tensors of any order is straightforward.

2 Preliminaries and notation

We let $\mathbb{R}^{I_1\times I_2\times I_3}$ denote the set of real third-order tensors of size (I_1, I_2, I_3). In order to improve readability, vectors are written in bold-face lower-case (e.g., $\mathbf{a}$), matrices in bold-face capitals (e.g., $\mathbf{A}$), and higher-order tensors in calligraphic letters (e.g., $\mathcal{A}$). For $n \in \{1, 2, 3\}$, the mode-n vectors of $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$ are obtained by varying the nth index while keeping the other indices fixed. The mode-n rank of $\mathcal{A}$, denoted $\operatorname{rank}_n(\mathcal{A})$, is the dimension of the linear space spanned by its mode-n vectors. The multilinear rank of $\mathcal{A}$ is the triple of the mode-n ranks. The mode-n product of $\mathcal{A}$ by $\mathbf{B} \in \mathbb{R}^{J_n\times I_n}$, denoted $\mathcal{A} \cdot_n \mathbf{B}$, is obtained by multiplying all the mode-n vectors of $\mathcal{A}$ by $\mathbf{B}$. We endow $\mathbb{R}^{I_1\times I_2\times I_3}$ with the standard inner product, defined by

$$\langle \mathcal{A}, \mathcal{B}\rangle := \sum_{i_1=1}^{I_1}\sum_{i_2=1}^{I_2}\sum_{i_3=1}^{I_3} \mathcal{A}(i_1, i_2, i_3)\mathcal{B}(i_1, i_2, i_3), \tag{1}$$

and we let $\|\cdot\|$ denote the induced norm, i.e., the Frobenius norm.

3 Variable projection

Let $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$. Consider positive integers R and R_i such that $R_i \leq \operatorname{rank}_i(\mathcal{A})$ for each $i \in \{1,2,3\}$ and $m := I_1 I_2 I_3 \geq R R_1 R_2 R_3 =: n$. The approximation of $\mathcal{A}$ by a BTD of R terms of multilinear rank (R_1, R_2, R_3) is a nonconvex minimization problem which can be expressed using variable projection as

$$\min_{\mathcal{S},\mathbf{U},\mathbf{V},\mathbf{W}} \underbrace{\left\| \mathcal{A} - \sum_{r=1}^{R} \mathcal{S}_r \cdot_1 \mathbf{U}_r \cdot_2 \mathbf{V}_r \cdot_3 \mathbf{W}_r \right\|^2}_{=: f_{\mathcal{A}}(\mathcal{S},\mathbf{U},\mathbf{V},\mathbf{W})} = \min_{\mathbf{U},\mathbf{V},\mathbf{W}} \underbrace{\min_{\mathcal{S}} f_{\mathcal{A}}(\mathcal{S},\mathbf{U},\mathbf{V},\mathbf{W})}_{=: g_{\mathcal{A}}(\mathbf{U},\mathbf{V},\mathbf{W})} \tag{2}$$

subject to the constraints $\mathbf{U}_r \in \operatorname{St}(R_1, I_1)$, $\mathbf{V}_r \in \operatorname{St}(R_2, I_2)$ and $\mathbf{W}_r \in \operatorname{St}(R_3, I_3)$ for each $r \in \{1, \ldots, R\}$, where given integers $p \geq q \geq 1$ we let $\operatorname{St}(q,p)$ denote the *Stiefel manifold*, i.e.,

$$\operatorname{St}(q,p) := \{\mathbf{U} \in \mathbb{R}^{p \times q} : \mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}_q\}. \tag{3}$$

A schematic representation of the BTD approximation problem is given in Fig. 1. The tensors $\mathcal{S}_r$ are called the core tensors while the matrices $\mathbf{U}_r, \mathbf{V}_r, \mathbf{W}_r$ are referred to as the factor matrices.

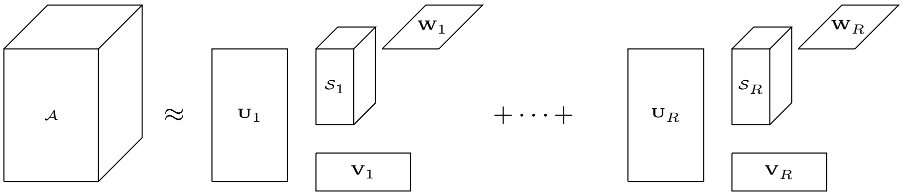

Fig. 1: Schematic representation of the BTD approximation problem.

Using vectorization, it can be shown that computing $g_{\mathcal{A}}(\mathbf{U},\mathbf{V},\mathbf{W})$ is a least squares problem; we denote its minimizer by $\mathcal{S}^*(\mathbf{U},\mathbf{V},\mathbf{W})$.[1] Thus,

$$g_{\mathcal{A}}(\mathbf{U},\mathbf{V},\mathbf{W}) = f_{\mathcal{A}}(\mathcal{S}^*(\mathbf{U},\mathbf{V},\mathbf{W}),\mathbf{U},\mathbf{V},\mathbf{W}). \tag{4}$$

Computing the partial derivatives of $g_{\mathcal{A}}$ reduces to the computation of partial derivatives of $f_{\mathcal{A}}$. Indeed, using the first-order optimality condition

$$\left.\frac{\partial f_{\mathcal{A}}(\mathcal{S},\mathbf{U},\mathbf{V},\mathbf{W})}{\partial \mathcal{S}}\right|_{\mathcal{S}=\mathcal{S}^*(\mathbf{U},\mathbf{V},\mathbf{W})} = \mathbf{0} \tag{5}$$

and the chain rule yields

$$\frac{\partial g_{\mathcal{A}}(\mathbf{U},\mathbf{V},\mathbf{W})}{\partial \mathbf{U}} = \left.\frac{\partial f_{\mathcal{A}}(\mathcal{S},\mathbf{U},\mathbf{V},\mathbf{W})}{\partial \mathbf{U}}\right|_{\mathcal{S}=\mathcal{S}^*(\mathbf{U},\mathbf{V},\mathbf{W})} \tag{6}$$

and likewise for the two other partial derivatives of $g_{\mathcal{A}}$. The partial derivatives of $f_{\mathcal{A}}$ can be computed in a convenient way using matricization [14].

[1] The minimizer is unique if and only if the matrix $[\mathbf{U}_j \otimes \mathbf{V}_j \otimes \mathbf{W}_j]_{i,j=1}^{1,R}$ has full column rank which is the case almost everywhere (with respect to the Lebesgue measure) since $m \geq n$.

4 Riemannian gradient algorithm

We briefly introduce the Riemannian gradient algorithm; our reference is [15]. Line-search methods to minimize a real-valued function F defined on a Riemannian manifold $\mathcal{M}$ are based on the update formula

$$x_{k+1} = R_{x_k}(t_k \eta_k), \tag{7}$$

where η_k is selected in the tangent space to $\mathcal{M}$ at x_k, denoted $\mathrm{T}_{x_k}\mathcal{M}$, R_{x_k} is a retraction on $\mathcal{M}$ at x_k, and $t_k \in \mathbb{R}$. The algorithm is defined by the choice of three ingredients: the retraction R_{x_k}, the search direction η_k and the step size t_k.

In our problem, we shall use the qf retraction [15, equation (4.8)]:

$$R_{\mathbf{X}}(\mathbf{Y}) := \mathrm{qf}(\mathbf{X} + \mathbf{Y}) \tag{8}$$

where $\mathrm{qf}(\mathbf{A})$ is the $\mathbf{Q}$ factor of the decomposition of $\mathbf{A} \in \mathbb{R}^{p\times q}$ with $\mathrm{rank}(\mathbf{A}) = q$ as $\mathbf{A} = \mathbf{QR}$ where $\mathbf{Q} \in \mathrm{St}(q,p)$ and $\mathbf{R}$ is an upper triangular $q \times q$ matrix with positive diagonal elements. The manifold in our problem is a Cartesian product of Stiefel manifolds; this is not an issue since the retraction can be performed componentwise. Next, the gradient method consists of choosing $\eta_k := -\operatorname{grad} F(x_k)$ where $\operatorname{grad} F$ is the Riemannian gradient of F. In the case where $\mathcal{M}$ is an embedded submanifold of a linear space $\mathcal{E}$ and F is the restriction on $\mathcal{M}$ of some function $\bar{F} : \mathcal{E} \to \mathbb{R}$, $\operatorname{grad} F(x)$ is simply the projection of the usual gradient of $\bar{F}$ at x on $\mathrm{T}_x\mathcal{M}$. For instance, $\mathrm{St}(q,p)$ is an embedded submanifold of $\mathbb{R}^{p\times q}$ and the projection of $\mathbf{Y} \in \mathbb{R}^{p\times q}$ on $\mathrm{T}_{\mathbf{X}}\mathrm{St}(q,p)$ is given by [15, equation (3.35)]

$$(\mathbf{I}_p - \mathbf{X}\mathbf{X}^{\mathrm{T}})\mathbf{Y} + \mathbf{X}\,\mathrm{skew}(\mathbf{X}^{\mathrm{T}}\mathbf{Y}) \tag{9}$$

where $\mathrm{skew}(\mathbf{A}) := \frac{1}{2}(\mathbf{A} - \mathbf{A}^{\mathrm{T}})$ is the skew-symmetric part of $\mathbf{A}$. At this point, it remains to specify the step size t_k. For that purpose, we will use the backtracking strategy presented in [15, section 4.2]. Assume we are at the kth iteration. We want to find $t_k > 0$ such that $F(R_{x_k}(-t_k \operatorname{grad} F(x_k)))$ is sufficiently small compared to $F(x_k)$. This can be achieved by the Armijo rule: given $\bar{\alpha} > 0$, $\beta, \sigma \in (0,1)$ and $\tau_0 := \bar{\alpha}$, we iterate $\tau_i := \beta\tau_{i-1}$ until

$$F(R_{x_k}(-\tau_i \operatorname{grad} F(x_k))) \leq F(x_k) - \sigma\tau_i \left\|\operatorname{grad} F(x_k)\right\|^2 \tag{10}$$

and then set $t_k := \tau_i$. In practice, we set $\bar{\alpha} := 0.2$, $\sigma := 10^{-3}$, $\beta := 0.2$ and we perform at most 10 iterations in the backtracking loop.

The procedure described in the previous paragraph corresponds to [15, Algorithm 1] with $c := 1$ and equality in [15, equation (4.12)]. In our problem, the domain of the cost function is compact since it is a Cartesian product of Stiefel manifolds. Therefore, [15, Corollary 4.3.2] applies and ensures that

$$\lim_{k\to\infty} \|\operatorname{grad} F(x_k)\| = 0. \tag{11}$$

In view of this result, it seems natural to stop the algorithm as soon as the norm of the gradient becomes smaller than a given quantity $\epsilon > 0$.

5 Numerical results

We evaluated the ability of the gradient algorithm to recover a known BTD. This test has two advantages: on one hand, we try to recover a structure that is really present, and on the other hand, we know the exact solution of the problem, which is a big advantage to evaluate the methods. Let us describe the test in detail. We set $R := 2$, we select

- positive integers I_i and R_i such that $R_i \leq I_i$ for each $i \in \{1, 2, 3\}$,
- $\mathcal{S}_r \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ for each $r \in \{1, \ldots, R\}$ according to the standard normal distribution, i.e., $\mathcal{S}_r :=$ `randn(`R_1`,`R_2`,`R_3`)` in Matlab,
- $\mathbf{U}_r \in \mathrm{St}(R_1, I_1)$, $\mathbf{V}_r \in \mathrm{St}(R_2, I_2)$ and $\mathbf{W}_r \in \mathrm{St}(R_3, I_3)$ for all $r \in \{1, \ldots, R\}$ according to the standard normal distribution, i.e.,
  $\mathbf{U}_r :=$ `qf(randn(`I_1`,`R_1`))` in Matlab,

and we define

$$\mathcal{A} := \sum_{r=1}^{R} \mathcal{S}_r \cdot_1 \mathbf{U}_r \cdot_2 \mathbf{V}_r \cdot_3 \mathbf{W}_r. \tag{12}$$

Then, we compare the gradient algorithm with variable projection (i.e., on the cost function $g_{\mathcal{A}}$) and without variable projection (i.e., on the cost function $f_{\mathcal{A}}$) using the same 100 randomly selected starting iterates. The results are presented in Table 1.[2] A success means that the norm of the gradient and the cost function have been respectively brought below $5 \cdot 10^{-14}$ and 10^{-25}.

	with VP	without VP
successes	100	100
min(iter)	266	371
mean(iter)	330	651
max(iter)	481	2804
mean(backtracking iter)	2.05	2.24
min(time)	0.41	0.55
mean(time)	0.52	1.03
max(time)	0.84	8.85
mean(time spent in VP)	0.11	/
mean(time/iter)	0.0016	0.0015

Table 1: Running times (in seconds) of the gradient algorithm with and without variable projection with $(I_1, I_2, I_3) := (5, 5, 5)$ and $(R_1, R_2, R_3) := (2, 2, 2)$.

At first sight, it may seem surprising that the mean running times of an iteration with and without variable projection are comparable (0.0016 vs 0.0015) since a least squares problem is solved at each iteration of the backtracking loop

[2] The Matlab code that produced the results is available at `https://sites.uclouvain.be/absil/2017.11`.

in the former case. In fact, it is consistent with the fact that fewer iterations are performed in the backtracking loop with variable projection (2.05 vs 2.24 on average). We also observe that the gradient algorithm with variable projection needs fewer iterations to converge, about half on average (330 vs 651). As a result, it is about twice faster on average.

References

[1] A. Cichocki, D. Mandic, A.H. Phan, C. Caiafa, G. Zhou, Q. Zhao, and L. De Lathauwer. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32(2):145–163, March 2015.

[2] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos. Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, 65(13):3551–3582, July 2017.

[3] L. De Lathauwer. Decompositions of a higher-order tensor in block terms—Part I: Lemmas for partitioned matrices. *SIAM J. Matrix Anal. Appl.*, 30(3):1022–1032, 2008.

[4] L. De Lathauwer. Decompositions of a higher-order tensor in block terms—Part II: Definitions and uniqueness. *SIAM J. Matrix Anal. Appl.*, 30(3):1033–1066, 2008.

[5] L. De Lathauwer and D. Nion. Decompositions of a higher-order tensor in block terms—Part III: Alternating least squares algorithms. *SIAM J. Matrix Anal. Appl.*, 30(3):1067–1083, 2008.

[6] L. De Lathauwer. Block component analysis, a new concept for blind source separation. In F. Theis, A. Cichocki, A. Yeredor, and M. Zibulevsky, editors, *Latent Variable Analysis and Signal Separation: 10th International Conference, LVA/ICA 2012, Tel Aviv, Israel, March 12-15, 2012. Proceedings*, pages 1–8. Springer Berlin Heidelberg, 2012.

[7] L. De Lathauwer. Blind separation of exponential polynomials and the decomposition of a tensor in rank-$(L_r, L_r, 1)$ terms. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1451–1474, December 2011.

[8] O. Debals, M. Van Barel, and L. De Lathauwer. Löwner-based blind signal separation of rational functions with applications. *IEEE Transactions on Signal Processing*, 64(8):1909–1918, April 2016.

[9] B. Hunyadi, D. Camps, L. Sorber, W. Van Paesschen, M. De Vos, S. Van Huffel, and L. De Lathauwer. Block term decomposition for modelling epileptic seizures. *EURASIP Journal on Advances in Signal Processing*, 2014(1):139, September 2014.

[10] N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer. Tensorlab 3.0, Mar. 2016. Available online. URL: `https://www.tensorlab.net`.

[11] L. De Lathauwer, B. De Moor, and J. Vandewalle. 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(4):1324–1342, 2000.

[12] M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer. Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme. *SIAM J. Matrix Anal. Appl.*, 32(1):115–135, 2011.

[13] Berkant Savas and Lek-Heng Lim. Quasi-newton methods on grassmannians and multilinear approximations of tensors. *SIAM J. on Scientific Computing*, 32(6):3352–3393, 2010.

[14] G. Olikier. Tensor approximation by block term decomposition. Master's thesis, Ecole Polytechnique de Louvain, Université catholique de Louvain, 2017. Supervisors: P.-A. Absil and L. De Lathauwer.

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