A Riemannian Newton Optimization Framework for the Symmetric Tensor Rank Approximation Problem

Rima Khouja, Houssam Khalil, Bernard Mourrain

A RIEMANNIAN NEWTON OPTIMIZATION FRAMEWORK FOR THE SYMMETRIC TENSOR RANK APPROXIMATION PROBLEM*

RIMA KHOUJA†‡, HOUSSAM KHALIL†, AND BERNARD MOURRAIN‡

Abstract. The symmetric tensor rank approximation problem (STA) consists in computing the best low rank approximation of a symmetric tensor. We describe a Riemannian Newton iteration with trust region scheme for the STA problem. We formulate this problem as a Riemannian optimization problem by parameterizing the constraint set as the Cartesian product of Veronese manifolds. We present an explicit and exact formula for the gradient vector and the Hessian matrix of the method, in terms of the weights and points of the low rank approximation and the symmetric tensor to approximate, by exploiting the properties of the apolar product. We introduce a retraction operator on the Veronese manifold. The Riemannian Newton iterations are performed for best low rank approximation over the real or complex numbers. Numerical experiments are implemented to show the numerical behavior of the new method against perturbation, to compute the best real rank-1 approximation and the spectral norm of a real symmetric tensor, and to compare with some existing state-of-the-art methods for higher rank sparse symmetric tensors.

Key words. symmetric tensor decomposition, homogeneous polynomials, Riemannian optimization, Newton method, retraction, complex optimization, trust region method, Veronese manifold.

AMS subject classifications. 15A69, 15A18, 53B20, 53B21, 14P10, 65K10, 65Y20, 90-08

1. Introduction.

A symmetric tensor T of order d and dimension n in $\mathcal{T}^d(\mathbb{C}^n) = \mathbb{C}^n \otimes \cdots \otimes \mathbb{C}^n := \mathcal{T}_n^d$, is a special case of tensors, where its entries do not change under any permutation of its d indices. We denote their set $\mathcal{S}^d(\mathbb{C}^n) := \mathcal{S}_n^d$. The symmetric tensor decomposition problem consists in decomposing a symmetric tensor $T \in \mathcal{S}_n^d$ into linear combination of symmetric tensors of rank one i.e.

$$T = \sum_{i=1}^{r} w_i \underbrace{v_i \otimes \ldots \otimes v_i}_{d \text{ times}}, \quad w_i \in \mathbb{C},\ v_i \in \mathbb{C}^n \tag{1.1}$$

We have a correspondence between $\mathcal{S}_n^d$ and the set of homogeneous polynomials of degree d in n variables denoted $\mathbb{C}[x_1, \ldots, x_n]_d =: \mathbb{C}[\mathbf{x}]_d$. Using this correspondence, (1.1) is equivalent to express the homogeneous polynomial P associated to T as a sum of powers of linear forms, which is by definition the classical Waring decomposition i.e.

$$P = \sum_{i=1}^{r} w_i (v_{i,1}x_1 + \cdots + v_{i,n}x_n)^d, \quad w_i \in \mathbb{C},\ v_i \in \mathbb{C}^n \tag{1.2}$$

The smallest r such that this decomposition exists is by definition the symmetric rank of P denoted by $\mathrm{rank}_s(P)$. Let $d \geq 3$. The generic symmetric rank denoted by r_g, is given by the Alexander-Hirschowitz theorem [5] as follows: $r_g = \left\lceil \frac{1}{n}\binom{n+d-1}{d} \right\rceil$ for all $n, d \in \mathbb{N}$, except for the following cases: $(d, n) \in \{(3,5), (4,3), (4,4), (4,5)\}$, where it should be increased by 1. We say that T is of subgeneric rank, if its rank $\mathrm{rank}_s(T) = r$ in (1.2) is strictly lower than r_g, In this case, a strong property of uniqueness of the

*This work was funded by the Lebanese University, Lebanon, and Inria, Sophia Antipolis, France.

†Laboratory of Mathematics and its Applications LaMa-Lebanon, Lebanese University, Faculty of Sciences, Lebanon. (houssam.khalil@ul.edu.lb).

‡Aromath, Inria Sophia Antipolis Méditerranée, Université Côte d'Azur, France. (bernard.mourrain@inria.fr, rima.khouja@inria.fr).

Waring decomposition holds [14], and the symmetric tensor T is called identifiable, unless in three exceptions which are cited in [14, Theorem 1.1], where there are exactly two Waring decompositions. This identifiability property forms an important key strength of the Waring or equivalently the symmetric tensor decomposition. It can explains why this decomposition problem appears in many applications for instance in the areas of mobile communications, in blind identification of under-determined mixtures, machine learning, factor analysis of k-way arrays, statistics, biomedical engineering, psychometrics, and chemometrics. See e.g. [15, 17, 18, 46] and references therein. The decomposition of the tensor is often used to recover structural information in the application problem.

The Symmetric Tensor Approximation problem (STA) consists in finding the closest symmetric tensor $\in \mathcal{S}_n^d$, which has a symmetric rank at most r for a given $r \in \mathbb{N}$. Equivalently, it consists in approximating a homogeneous polynomial P associated to a symmetric tensor T by an element in σ_r, where $\sigma_r = \{Q \in \mathbb{C}[\mathbf{x}]_d \mid rank_s(Q) \leq r\}$, i.e.

$$\text{(STA)} \quad \min_{Q \in \sigma_r} \tfrac{1}{2}||P - Q||_d^2.$$

Since in many problems, the input tensors are often computed from measurements or statistics, they are known with some errors on their coefficients and computing an approximate decomposition of low rank often gives better structural information than the exact or accurate decomposition of the approximate tensor [6, 7, 22].

An approach which has been investigated to address the STA problem is to extend the Singular Value Decomposition (SVD) to this problem, since the truncated SVD of a matrix yields its best approximation of a given rank. This so-called Highed Order Singular Value Decomposition (HOSVD) method has been studied for tensor decomposition [18] or for multi-linear rank computation based on matrix SVD [19, 36], or using iterative truncation strategies [52]. However, when dealing with best low rank approximations of tensors, these techniques do not provide the closest low rank tensor, since the structure is not taken into account in the flattening operations involved in these methods.

Another classical approach for computing an approximate tensor decomposition of low rank is the so-called Alternating Least Squares (ALS) method. It consists in minimizing the distance between a given tensor and a multilinear low rank tensor tensor by alternately updating the different factors of the tensor decomposition, solving a quadratic minimization problem at each step. See e.g. [11, 12, 25, 31]. This approach is well-suited for tensor represented in $\mathcal{T}_n^d$ but it looses the symmetry property in the internal steps of the algorithm. The space in which the linear operations are performed is of large dimension n^d compared to the dimension $\binom{n+d-1}{d}$ of $\mathcal{S}_n^d$ when n and d grow. Moreover the convergence is slow [21, 51].

Other iterative methods such as quasi-Newton methods have been considered for tensor low rank approximation problems to improve the convergence speed. See for instance [26, 42, 43, 45, 48, 50]. In [9, 10], a Gauss-Newton iteration on a real Riemannian manifold associated to a sum of rank-1 multilinear tensors is presented in order to approximate a given tensor by a tensor of low rank r i.e. by a tensor of canonical polyadic decomposition [27], abbreviated by CPD, equal to the sum of r tensors of rank-1. Optimization techniques based on quasi-Newton iterations for block term decompositions of multilinear tensors over the complex numbers have also been presented in [47, 48]. In [45] quasi-Newton and limited memory quasi-Newton methods for distance optimization on products of Grassmannian varieties are designed

to deal with the Tucker decomposition of a tensor and applied for best low multi-rank tensor approximation. In all these approaches, an approximation on the Hessian is used to compute the descent direction, and the local quadratic convergence cannot be guaranteed.

Specific investigations have been developed, in the case of best rank-1 approximation. The problem is equivalent to the optimisation of a polynomial on the product of unitary spheres (see e.g. [19, 55]). Global polynomial optimization methods can be employed over the real or complex numbers, using for instance convex relaxations and semidefinite programming [40]. However, the approach is facing scalability issues in practice for large size tensors.

In relation with polynomial representation and multivariate Hankel matrix properties, another least square optimization problem is presented in [39], for low rank symmetric tensor approximation. Good approximation of the best low rank approximation are obtained for small enough perturbations of low rank tensors. More recently, a method for decomposing real even-order symmetric tensors, called Subspace Power Method (SPM), has been proposed in [29]. It is based on a power method associated to the projection on subspaces of eigenvectors of the Hankel operators and has a linear convergence.

Contributions. In this paper, we present a new Riemannian Newton method with trust region scheme for the STA problem. Exploiting the dimension reduction of the problem by working in $\mathbb{C}[\mathbf{x}]_d$, considering a suitable representation of the points on the Riemannian manifold, and combining the properties of the apolar product with efficient tools from complex optimization, we give an explicit, exact and tractable formulation of the Newton iterations for the distance minimization on the Riemannian manifold, which is the Cartesian product of Veronese manifolds. The explicit formulation is provided for low rank symmetric tensor approximation over the real and complex numbers. We propose an approximation method for a given homogeneous polynomial in $\mathbb{C}[\mathbf{x}]_d$ into linear form to the d^{th} power, based on the rank-1 truncation of the SVD of Hankel matrices associated to the homogeneous polynomial. From this approximation method, we present a retraction operator on the Veronese manifold. To compute a rank-r approximation of a symmetric tensor, we propose to combine the Riemannian Newton method with an algebraic method called SHD (Spectral Hankel Decomposition) for choosing the initial point. The SHD method is based on the computation of common generalized eigenvectors and generalized eigenvalues of pencils of Hankel matrices [24, 37]. Numerical experiments show the good numerical behavior of the new method against perturbations. The good performance of the approach appears clearly in particular, for the best rank-1 approximation of real-valued symmetric tensors. Comparisons with existing state-of-the-art methods corroborate this analysis.

The paper is structured as follows. In section 2 we give the main notation and preliminaries. Section 3 describes the different steps of the construction of the Riemannian Newton with trust region scheme algorithm that we develop for the STA problem, called RNS-TR. In subsection 3.1, we formulate the STA problem as a Riemannian least square optimization problem. In subsection 3.2, we introduce the parameterization of the points on the Riemannian manifold that we use in the computation of the gradient vector and the Hessian matrix in subsection 3.3. In subsection 3.4, we present a retraction operator on the Veronese manifold with its analysis. In subsection 3.5 we discuss the choice of the initial point in the iterative algorithm. Our new algorithm RNS-TR is presented in subsection 3.6. Numerical experiments are featured in section 4. The final section is for our conclusions and outlook.

2. Notation and preliminaries. We use similar notation as in [16]. We denote by $\mathcal{T}_n^d$ the set of outer product d times of $\mathbb{C}^n$. The set of symmetric tensors in $\mathcal{T}_n^d$ is denoted $\mathcal{S}_n^d$. We have a correspondence between $\mathcal{S}_n^d$ and the set of the homogeneous polynomials of degree d in n variables $\mathbb{C}[x_1,\ldots,x_n]_d := \mathbb{C}[\mathbf{x}]_d$. This allows to reduce the dimension of the ambient space of the problem from n^d (dimension of $\mathcal{T}_n^d$) to $\binom{n+d-1}{d}$ (dimension of $\mathcal{S}_n^d \sim \mathbb{C}[\mathbf{x}]_d$). The capital letters P, Q and T denote the homogeneous polynomials in $\mathbb{C}[\mathbf{x}]_d$ or equivalently elements in $\mathcal{S}_n^d$. A homogeneous polynomial P in $\mathbb{C}[\mathbf{x}]_d$ can be written as: $P = \sum_{|\alpha|=d} \binom{d}{\alpha} p_\alpha x^\alpha$, where $x = (x_1,\ldots,x_n)$ is the vector of the variables $x_1,\ldots,x_n$, $\alpha = (\alpha_1,\ldots,\alpha_n)$ is a vector of the multi-indices in $\mathbb{N}^n$, $|\alpha| = \alpha_1 + \cdots + \alpha_n$, $p_\alpha \in \mathbb{C}$, $x^\alpha := x_1^{\alpha_1}\ldots x_n^{\alpha_n}$ and $\binom{d}{\alpha} := \frac{d!}{\alpha_1!\ldots\alpha_n!}$. The superscripts $.^T$, $.^*$ and $.^{-1}$ are used respectively for the transpose, Hermitian conjugate, and the inverse matrix. The complex conjugate is denoted by an overbar, e.g., $\bar{w}$. We use parentheses to denote vectors e.g. $W = (w_i)_{1\le i\le r}$, and the square brackets to denote matrices e.g. $V = [v_i]_{1\le i\le r}$ where v_i are column vectors.

DEFINITION 2.1. *For $P = \sum_{|\alpha|=d} \binom{d}{\alpha} p_\alpha x^\alpha$ and $Q = \sum_{|\alpha|=d} \binom{d}{\alpha} q_\alpha x^\alpha$ in $\mathbb{C}[\mathbf{x}]_d$, their apolar product is*

$$\langle P, Q\rangle_d := \sum_{|\alpha|=d} \binom{d}{\alpha} \bar{p}_\alpha q_\alpha.$$

The apolar norm of P is $||P||_d = \sqrt{\langle P, P\rangle_d} = \sqrt{\sum_{|\alpha|=d} \binom{d}{\alpha} \bar{p}_\alpha p_\alpha}$.

The following properties of the apolar product can be verified by direct calculus:

LEMMA 2.2. *Let $L = (v_1x_1 + \cdots + v_nx_n)^d := (v^t x)^d \in \mathbb{C}[\mathbf{x}]_d$ where $v = (v_i)_{1\le i\le n}$ is a vector in $\mathbb{C}^n$, $P \in \mathbb{C}[\mathbf{x}]_d, Q \in \mathbb{C}[\mathbf{x}]_{(d-1)}$, we have the following two properties:*

1. $\langle L, P\rangle_d = P(\bar{v})$,
2. $\langle P, x_i Q\rangle_d = \frac{1}{d}\langle \partial_{x_i} P, Q\rangle_{(d-1)}, \quad \forall 1 \le i \le n.$

3. Riemannian Newton optimization for the STA problem. We present in this section a Riemannian Newton optimization framework for the STA problem. The approach is similar to the one described in [10, 23, 33] for real multilinear tensors. However, it differs in several ways: we develop a Riemannian Newton iteration suited for symmetric tensors, by strictly connecting with the geometry of this case. We exploit the symmetric setting to obtain an exact Newton iteration with simplified computation. The retraction properties are deduced from a direct analysis of the Hankel operators. We consider distance minimization problem for symmetric tensors with real and complex decompositions.

Riemannian optimization methods are solving optimization problems over a Riemannian manifold $\mathcal{M}$ [2]. In our problem, we will consider the following least square minimization problem

$$\min_{y\in\mathcal{M}} \frac{1}{2}||F(y)||^2 \tag{3.1}$$

where $F : \mathcal{M} \mapsto \mathbb{R}^N$ is a smooth objective function with $N \ge \dim \mathcal{M}$ and $\mathcal{M}$ is a Riemannian manifold associated to rank-r symmetric tensors. A Riemannian Newton method (See Algorithm 3.1) for solving (3.1) requires a Riemannian metric, and a retraction operator R_y from the tangent space $T_y\mathcal{M}$ at $y \in \mathcal{M}$ to $\mathcal{M}$ [2, Chapter 6]. Since we will assume that $\mathcal{M}$ is embedded in some space $\mathbb{R}^M$, we will take the metric induced by the Euclidean space $\mathbb{R}^M$.

Algorithm 3.1 Riemannian Newton method

Data: Riemannian manifold $\mathcal{M}$; retraction R on $\mathcal{M}$; function $F : \mathcal{M} \to \mathbb{R}^N$; objective function $f = \frac{1}{2}||F||^2$.
Input: A starting point $y_0 \in \mathcal{M}$.
Output: A sequence of iterates y_k.
for $k = 0, 1, 2, \ldots$ **do**
1. Solve the Newton equation $\operatorname{Hess} f(y_k)[\eta_k] = -\operatorname{grad} f(y_k)$ for the unknown vector η_k in the tangent space $T_{y_k}\mathcal{M}$ of $\mathcal{M}$ at y_k;
2. Set $y_{k+1} \leftarrow R_{y_k}(\eta_k)$;

end for

3.1. Formulation of the Riemannian least square problem. We describe now the Riemannian manifold that we use for the Riemannian Newton method.

DEFINITION 3.1. *Let* $\Phi : \mathbb{C}^n \to \mathbb{C}[\mathbf{x}]_d,\ v \mapsto (v^t x)^d = \sum_{|\alpha|=d} \binom{d}{\alpha} v^\alpha x^\alpha$. *The Veronese manifold in* $\mathbb{C}[\mathbf{x}]_d$ *denoted by* $\mathcal{V}_n^d$ *[28, 54] is the set of linear forms in* $\mathbb{C}[\mathbf{x}]_d - \{0\}$ *to the* d^{th} *power. It is the image of* Φ *after removing the zero polynomial i.e.* $\mathcal{V}_n^d := \Phi(\mathbb{C}^n) - \{0\}$.

Let $\sigma_r \subset \mathcal{S}_n^d$ be the set of the symmetric tensors of symmetric rank bounded by r. It is given by the image of the following map:

$$\Sigma_r : \mathcal{V}_n^{d\times r} := \mathcal{V}_n^d \times \cdots \times \mathcal{V}_n^d \longrightarrow \mathbb{C}[\mathbf{x}]_d$$
$$((v_i^t x)^d)_{1\le i\le r} \longmapsto \Sigma_r((v_i^t x)^d)_{1\le i\le r}) = \sum_{i=1}^{r} (v_i^t x)^d$$

The closure of the image of Σ_r is called the r^{th} secant of the Veronese variety $\mathcal{V}_n^d$. We consider the case of r subgeneric rank i.e. $r < r_g$, where r_g is the generic symmetric rank given by the Alexander-Hirschowitz theorem [5]. We are interested in such cases because Σ_r has a differential map which is generically an embedding and provides a regular parametrization of σ_r (see "Terracini's lemma", e.g. in [35]).

The Riemannian manifold that we will use is the r^{th} cartesian product $\mathcal{M} = \mathcal{V}_n^{d\times r}$ of the Veronese variety $\mathcal{V}_n^d$. We reformulate the Riemannian least square problem for the symmetric tensor approximation problem as follows:

$$(STA^*) \quad \min_{y\in\mathcal{M}} f(y)$$

where $\mathcal{M} = \mathcal{V}_n^{d\times r}$, $y = ((v_i^t x)^d)_{1\le i\le r}$, $f(y) = \frac{1}{2}||F(y)||_d^2$ and $F(y) = \Sigma_r(y) - P$.

3.2. Parameterization. For a symmetric tensor P in the image of Σ_r, its decomposition can be rewritten as $P = \sum_{i=1}^r w_i\,(v_i^t x)^d$ with $w_i \in \mathbb{R}_+^*$ and $||v_i|| = 1$, for $1 \le i \le r$; since we have $\sum_{i=1}^r (\tilde{v}_i^t x)^d = \sum_{i=1}^r ||\tilde{v}_i||^d (\frac{\tilde{v}_i^t}{||\tilde{v}_i||} x)^d = \sum_{i=1}^r w_i (v_i^t x)^d$, with $w_i := ||\tilde{v}_i||^d$ and $v_i := \frac{\tilde{v}_i^t}{||\tilde{v}_i||}$ such that $||v_i|| = 1$.

The vector $(w_i)_{1\le i\le r} \in \mathbb{C}^r$ in this decomposition is called "the weight vector", and is denoted by W. The coefficient vectors of the linear polynomials $(v_i^t x)$ such that $||v_i|| = 1$ form a matrix denoted $V = [v_i]_{1\le i\le r} \in \mathbb{C}^{n\times r}$.

The objective function f in subsection 3.1 is a real valued function of complex variables; such function is non-analytic, because it cannot verify the Cauchy-Riemann

conditions [44]. To apply the Riemannian Newton method, we need the second order Taylor series expansion of f. As discussed in [47], we overcome this problem by converting the optimization problem to the real domain, regarding f as a function of the real and the imaginary parts of its complex variables.

Let $\mathcal{N} = \{(W, \Re(V), \Im(V)) \mid W \in {\mathbb{R}_+^*}^r, V \in \mathbb{C}^{n\times r}, (\Re(v_i), \Im(v_i)) \in \mathbb{S}^{2n-1}, \forall 1 \leq i \leq r\}$, where $\mathbb{S}^{2n-1}$ is the unit sphere in $\mathbb{R}^{2n}$. Routine calculation shows that $\mathcal{N}$ is a Riemannian submanifold of $\mathbb{R}^{r+2nr}$ of dimension $r + r(2n-1) = 2nr$. This is the manifold that we will use for the parametrization of $\mathcal{M}$. We will consider f as a function of $\mathcal{N}$, in order to compute effectively its gradient vector and Hessian matrix.

3.3. Computation of the gradient vector and the Hessian matrix. Transforming the pair $(\Re(z), \Im(z))$ of the real and imaginary parts of a given complex variable z into the pair $(z, \overline{z})$ is a simple linear transformation, which will allow us to achieve explicit and simple computation of the gradient and Hessian of f.

Let $R_r = \{(W, \Re(V), \Im(V)) \mid W \in \mathbb{R}^r, V \in \mathbb{C}^{n\times r}\}$, $C_r = \left\{(W, V, \overline{V}) \mid W \in \mathbb{R}^r, V \in \mathbb{C}^{n\times r}\right\}$ and

$$K = \begin{bmatrix} I_r & 0_{r\times 2nr} \\ 0_{2nr\times r} & J \end{bmatrix} \tag{3.2}$$

where $J = \begin{bmatrix} I_{nr} & iI_{nr} \\ I_{nr} & -iI_{nr} \end{bmatrix}$. The linear map K is an isomorphism from R_r to C_r and its inverse is given by $K^{-1} = \begin{bmatrix} I_r & 0_{r\times 2nr} \\ 0_{2nr\times r} & \frac{1}{2}J^* \end{bmatrix}$.

The computation of the gradient and the Hessian of f as a function of C_r yields more elegant expressions than considering f as a function of R_r. For this reason, we consider f as a function of C_r to compute them, and then we use the isomorphism K in (3.2) to define the gradient and the Hessian of f as a function of R_r (see Lemma 3.3 below).

We call the gradient and the Hessian of f as a function of R_r (resp. C_r) at a point $(W, \Re(V), \Im(V))$ (resp. $(W, V, \overline{V})$) the real gradient and the real Hessian (resp. the complex gradient and the complex Hessian) and we denote them by G^R and H^R (resp. by G^C and H^C).

Recall that f is a function on $\mathcal{N}$. We can define the gradient and the Hessian of f at a point $p \in \mathcal{N}$, denoted respectively by G and H, by:

$$G = Q^T G^R, H = Q^T H^R Q$$

where the columns of Q form an orthonormal basis of $T_p\mathcal{N}$ [2, Ch.5].

Hereafter, we detail the computation of Q, G^R and H^R.

PROPOSITION 3.2. *Let $p = (W, \Re(V), \Im(V)) \in \mathcal{N}$. For all i in $\{1, \ldots, r\}$, we denote by u_i the vector $(\Re(v_i), \Im(v_i)) \in \mathbb{R}^{2n}$, and by $Q_{i,re}$ (resp. $Q_{i,im}$) the matrix given by the first n rows (resp. the last n rows) and the first $2n-1$ columns of the factor Q_i of the QR decomposition of $I_{2n} - u_i u_i^t$: $(I_{2n} - u_i u_i^t)P_i = Q_i R_i$ such that $Q_i Q_i^T = I_{2n}$, R_i is upper triangular, and P_i is a permutation matrix. Let $M = \begin{bmatrix} Q_{re} \\ Q_{im} \end{bmatrix} \in \mathbb{R}^{2nr\times(2n-1)r}$, where $Q_{re} = \mathrm{diag}(Q_{i,re})_{1\leq i\leq r}$ and $Q_{im} = \mathrm{diag}(Q_{i,im})_{1\leq i\leq r}$. Then the columns of $Q = \mathrm{diag}(I_r, M)$ form an orthonormal basis of $T_p\mathcal{N}$.*

Proof. Denote $(\Re(V), \Im(V))$ by Z. Using that $T_p\mathcal{N} \simeq T_W({\mathbb{R}_+^*}^r) \times T_Z\mathcal{V}$, where $\mathcal{V} = \left\{(\Re(V), \Im(V)) \mid V \in \mathbb{C}^{n\times r}, \ ||v_i||^2 = 1, \ \forall 1 \leq i \leq r\right\}$. We have $T_W({\mathbb{R}_+^*}^r) = \mathbb{R}^r$ and I_r is an orthonormal basis of $T_W({\mathbb{R}_+^*}^r)$.

We verify that M is an orthonormal basis of $T_Z\mathcal{V}$. Firstly, for each $i \in \{1, \cdots, r\}$,

$u_i \in \mathbb{S}^{2n-1} \subset \mathbb{R}^{2n}$, thus the first $(2n-1)$ columns of the factor Q_i of the QR decomposition of $I_{2n} - u_i u_i^t$ give an orthonormal basis of the image of $(I_{2n} - u_i u_i^t)$, which is $T_{u_i}\mathbb{S}^{2n-1}$.

Secondly, $T_Z\mathcal{V}$ is a vector space of dimension $r(2n-1)$ which is the cartesian product of the tangent spaces $T_{u_i}\mathbb{S}^{2n-1}$. Therefore, by construction, the columns of M form an orthonormal basis of $T_Z\mathcal{V}$.

Finally $Q = \mathrm{diag}(I_r, M)$ is an orthonormal basis of $T_p\mathcal{N}$. ∎

LEMMA 3.3. *The complex gradient G^C can be transformed into the real gradient G^R as follows:*

$$G^R = K^T G^C \tag{3.3}$$

Similarly H^R and H^C are related by the following formula:

$$H^R = K^T H^C K \tag{3.4}$$

Proof. As the first block of the matrix K in (3.2) is equal to the identity matrix I_r, we just have to verify that: $\frac{\partial f}{\partial z^R} = J^T \frac{\partial f}{\partial z^C}$, where $z := (v_i)_{1\leq i\leq r}, x := \Re(z), y := \Im(z), z^R := (x, y)$, and $z^C := (z, \bar{z})$.
We have that $x = \frac{1}{2}(z + \bar{z})$ and $y = \frac{i}{2}(\bar{z} - z)$, thus $\frac{\partial f}{\partial z} = \frac{\partial x}{\partial z}\frac{\partial f}{\partial x} + \frac{\partial y}{\partial z}\frac{\partial f}{\partial y} = \frac{1}{2}(\frac{\partial f}{\partial x} - i\frac{\partial f}{\partial y})$, similarly we have that $\frac{\partial f}{\partial \bar{z}} = \frac{1}{2}(\frac{\partial f}{\partial x} + i\frac{\partial f}{\partial y})$, which implies that $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial z} + \frac{\partial f}{\partial \bar{z}}$, and $\frac{\partial f}{\partial y} = i(\frac{\partial f}{\partial z} - \frac{\partial f}{\partial \bar{z}})$, herein $\frac{\partial f}{\partial z^R} = J^T \frac{\partial f}{\partial z^C}$. The formula (3.4) follows from the fact that: $\frac{\partial^2}{\partial z^R \partial z^{R^T}} = \frac{\partial}{\partial z^R}(\frac{\partial}{\partial z^R})^T = J^T \frac{\partial}{\partial z^C}(\frac{\partial}{\partial z^{C^T}} J) = J^T \frac{\partial^2}{\partial z^C \partial z^{C^T}} J$. ∎

PROPOSITION 3.4. *The gradient G^R of f on R_r is the vector*

$$G^R = \begin{pmatrix} G_1 \\ \Re(G_2) \\ -\Im(G_2) \end{pmatrix} \in \mathbb{R}^{r+2nr},$$

where

- $G_1 = (\sum_{i=1}^r w_i \Re((v_j^* v_i)^d) - \Re(\bar{P}(v_j)))_{1\leq i\leq r} \in \mathbb{R}^r$
- $G_2 = (d\sum_{i=1}^r w_i w_j (v_i^* v_j)^{(d-1)} \bar{v}_i - w_j \nabla\bar{P}(v_j))_{1\leq j\leq r} \in \mathbb{C}^{nr}$.

Proof. We can write f as:

$$\frac{1}{2}(f_1 - f_2 - f_3 + f_4) \tag{3.5}$$

where:

$$f_1 = \Big|\Big|\sum_{i=1}^r w_i (v_i^t x)^d\Big|\Big|_d^2 = \Big|\Big|\sum_{i=1}^r w_i \sum_{|\alpha|=d} \binom{d}{\alpha} v_i^\alpha x^\alpha\Big|\Big|_d^2 = \Big|\Big| \sum_{|\alpha|=d} \binom{d}{\alpha} \Big(\sum_{i=1}^r w_i v_i^\alpha\Big) x^\alpha\Big|\Big|_d^2$$

$$= \sum_{|\alpha|=d} \binom{d}{\alpha} \Big(\sum_{i=1}^r w_i \bar{v}_i^\alpha\Big)\Big(\sum_{i=1}^r w_i v_i^\alpha\Big) \text{ (by using the norm in Definition 2.1)},$$

$$f_2 = \langle \sum_{i=1}^r w_i (v_i^t x)^d, P\rangle_d = \sum_{i=1}^r w_i P(\bar{v}_i) \text{ (by using 1. in Lemma 2.2)}, f_3 = \bar{f_2} = \sum_{i=1}^r w_i \bar{P}(v_i), \text{and } f_4 = ||P||_d^2.$$

Let us decompose G^C as $G^C = \begin{pmatrix} G_1 \\ \tilde{G}_2 \\ \tilde{G}_3 \end{pmatrix}$ with $G_1 = (\frac{\partial f}{\partial w_j})_{1\leq j\leq r}$, $\tilde{G}_2 = (\frac{\partial f}{\partial v_j})_{1\leq j\leq r}$ and $\tilde{G}_3 = (\frac{\partial f}{\partial \bar{v}_j})_{1\leq j\leq r}$. As f is a real valued function, we have that $\frac{\partial f}{\partial \bar{v}_j} = \overline{\frac{\partial f}{\partial v_j}}$ [38, 44],

thus $\tilde{G}_3 = \overline{\tilde{G}_2}$. Let us start by the computation of G_1:

$$\begin{aligned}
\frac{\partial f_1}{\partial w_j} &= \frac{\partial}{\partial w_j}\Big(\sum_{|\alpha|=d}\binom{d}{\alpha}\Big(\sum_{i=1}^r w_i\bar{v}_i^\alpha\Big)\Big(\sum_{i=1}^r w_i v_i^\alpha\Big)\Big)\\
&= \sum_{|\alpha|=d}\binom{d}{\alpha}\Big(\bar{v}_j^\alpha\Big(\sum_{i=1}^r w_i v_i^\alpha\Big) + v_j^\alpha\Big(\sum_{i=1}^r w_i\bar{v}_i^\alpha\Big)\Big)\\
&= \sum_{i=1}^r w_i\sum_{|\alpha|=d}\binom{d}{\alpha}v_i^\alpha\bar{v}_j^\alpha + \sum_{i=1}^r w_i\sum_{|\alpha|=d}\binom{d}{\alpha}\bar{v}_i^\alpha v_j^\alpha\\
&= \sum_{i=1}^r w_i\langle(v_j^t x)^d,(v_i^t x)^d\rangle_d + \sum_{i=1}^r w_i\langle(v_i^t x)^d,(v_j^t x)^d\rangle_d \text{ (by Definition 2.1)}\\
&= \sum_{i=1}^r w_i(v_j^* v_i)^d + \sum_{i=1}^r w_i(v_i^* v_j)^d = 2\sum_{i=1}^r w_i\Re((v_j^* v_i)^d) \text{ (by Lemma 2.2 (1))},\\
\frac{\partial f_2}{\partial w_j} &= \frac{\partial}{\partial w_j}(\sum_{i=1}^r w_i P(\bar{v}_i)) = P(\bar{v}_j), \frac{\partial f_3}{\partial w_j} = \bar{P}(v_j), \text{and } \frac{\partial f_4}{\partial w_j} = 0.
\end{aligned}$$

Thus: $\dfrac{\partial f}{\partial w_j} = \dfrac{1}{2}\Big(2\sum_{i=1}^r w_i\Re((v_j^* v_i)^d) - P(\bar{v}_j) - \bar{P}(v_j)\Big) = \sum_{i=1}^r w_i\Re((v_j^* v_i)^d) - \Re(\bar{P}(v_j))$

Now, for the computation of $\tilde{G}_2$, let $P = \sum_{|\alpha|=d}\binom{d}{\alpha}a_\alpha x_\alpha$, and $1 \le k \le n$:

$$\begin{aligned}
\frac{\partial f_1}{\partial v_{j,k}} &= \sum_{|\alpha|=d}\binom{d}{\alpha}\Big(\sum_{i=1}^r w_i\bar{v}_i^\alpha\Big)(w_j\alpha_k v_j^{\alpha-e_k})\\
&= w_j\sum_{i=1}^r w_i\sum_{|\alpha|=d}\binom{d}{\alpha}\alpha_k\bar{v}_i^\alpha v_j^{\alpha-e_k}\\
&= w_j\sum_{i=1}^r w_i\langle\partial_{x_k}(v_i^t x)^d,(v_j^t x)^{d-1}\rangle_{d-1} \text{ (by Lemma 2.2 (1))}\\
&= dw_j\sum_{i=1}^r w_i\langle(v_i^t x)^d,x_k(v_j^t x)^{d-1}\rangle_d \text{ (by Lemma 2.2 (2))}\\
&= dw_j\sum_{i=1}^r w_i\bar{v}_{i,k}(v_i^* v_j)^{d-1} \text{ (by Lemma 2.2 (1))},\\
\frac{\partial f_2}{\partial v_{j,k}} &= 0, \frac{\partial f_3}{\partial v_{j,k}} = w_j\sum_{|\alpha|=d}\binom{d}{\alpha}\bar{a}_\alpha\alpha_k v_j^{\alpha-e_k} = w_j\partial_{x_k}\bar{P}(v_j), \text{and } \frac{\partial f_4}{\partial v_{j,k}} = 0
\end{aligned}$$

Thus: $\dfrac{\partial f}{\partial v_j} = \dfrac{1}{2}\Big(dw_j\sum_{i=1}^r w_i(v_i^* v_j)^{(d-1)}\bar{v}_i - w_j\nabla\bar{P}(v_j)\Big)$

We have $G^R = K^T G^C$ from (3.3). By multiplication of these two matrices, we obtain: $G^R = \begin{pmatrix} G_1 \\ \tilde{G}_2 + \overline{\tilde{G}_2} \\ i(\tilde{G}_2 - \overline{\tilde{G}_2})\end{pmatrix} = \begin{pmatrix} G_1 \\ 2\Re(\tilde{G}_2) \\ -2\Im(\tilde{G}_2)\end{pmatrix}$. Finally dividing by 2, we get $G^R =$

$\begin{pmatrix} G_1 \\ \Re(G_2) \\ -\Im(G_2) \end{pmatrix}$ where $G_2 = 2\tilde{G}_2$, which ends the proof. ∎

PROPOSITION 3.5. *The real Hessian H^R is given by the following block matrix:*

$$H^R = \begin{bmatrix} A & \Re(B)^T & -\Im(B)^T \\ \Re(B) & \Re(C+D) & -\Im(C+D) \\ -\Im(B) & \Im(D-C) & \Re(D-C) \end{bmatrix} \in \mathbb{R}^{(r+2nr)\times(r+2nr)}$$

with

- $A = \Re([(v_i^* v_j)^d]_{1\leq i,j\leq r}) \in \mathbb{R}^{r\times r}$,
- $B = [dw_i(v_j^* v_i)^{d-1}\bar{v}_j + \delta_{i,j}(d\sum_{l=1}^r w_l(v_l^* v_i)^{d-1}\bar{v}_l - \nabla\bar{P}(v_j))]_{1\leq i,j\leq r} \in \mathbb{C}^{nr\times r}$,
- $C = \operatorname{diag}[d(d-1)[\sum_{i=1}^r w_i w_j \overline{v_{i,k}v_{i,l}}(v_i^* v_j)^{d-2}]_{1\leq k,l\leq n} - w_j\Delta\bar{P}(v_j)]_{1\leq j\leq r} \in \mathbb{C}^{nr\times nr}$,
- $D = [dw_i w_j(v_i^* v_j)^{d-2}((v_i^* v_j)I_n + (d-1)v_j v_i^*)]_{1\leq i,j\leq r} \in \mathbb{C}^{nr\times nr}$.

Proof. H^C is given by the following block matrix:

$$H^C = \begin{bmatrix} \left[\frac{\partial^2 f}{\partial w_i \partial w_j}\right]_{1\leq i,j\leq r} & \left[\frac{\partial^2 f}{\partial w_i \partial v_j^T}\right]_{1\leq i,j\leq r} & \left[\frac{\partial^2 f}{\partial w_i \partial \bar{v}_j^T}\right]_{1\leq i,j\leq r} \\ \left[\frac{\partial^2 f}{\partial v_i \partial w_j}\right]_{1\leq i,j\leq r} & \left[\frac{\partial^2 f}{\partial v_i \partial v_j^T}\right]_{1\leq i,j\leq r} & \left[\frac{\partial^2 f}{\partial v_i \partial \bar{v}_j^T}\right]_{1\leq i,j\leq r} \\ \left[\frac{\partial^2 f}{\partial \bar{v}_i \partial w_j}\right]_{1\leq i,j\leq r} & \left[\frac{\partial^2 f}{\partial \bar{v}_i \partial v_j^T}\right]_{1\leq i,j\leq r} & \left[\frac{\partial^2 f}{\partial \bar{v}_i \partial \bar{v}_j^T}\right]_{1\leq i,j\leq r} \end{bmatrix}.$$

We have that $\frac{\partial^2 f}{\partial \bar{z}\partial \bar{z}^T} = \overline{\frac{\partial^2 f}{\partial z \partial z^T}}$, and $\frac{\partial^2 f}{\partial z \partial \bar{z}^T} = \frac{\partial^2 f}{\partial \bar{z}\partial z^T}$, for a complex variable z and a real valued function with complex variables f. Using these two relations, we find that $\left[\frac{\partial^2 f}{\partial w_i \partial w_j}\right]_{1\leq i,j\leq r}$, $\left[\frac{\partial^2 f}{\partial v_i \partial w_j}\right]_{1\leq i,j\leq r}$, $\left[\frac{\partial^2 f}{\partial v_i \partial v_j^T}\right]_{1\leq i,j\leq r}$, and $\left[\frac{\partial f}{\partial \bar{v}_i \partial v_j^T}\right]_{1\leq i,j\leq r}$ determine H^C. We denote them respectively by A, $\tilde{B}$, $\tilde{C}$, and $\tilde{D}$. Herein, we can decompose H^C as:

$$H^C = \begin{bmatrix} A & \tilde{B}^T & \tilde{B}^* \\ \tilde{B} & \tilde{C} & \tilde{D}^T \\ \overline{\tilde{B}} & \tilde{D} & \overline{\tilde{C}} \end{bmatrix}.$$

The computation of these four matrices can be done by taking the formula of $\frac{\partial f}{\partial w_j}$ and $\frac{\partial f}{\partial v_j}$ obtained in Proof 3, and using the apolar identities in Lemma 2.2. Using (3.4) we obtain: $H^R = \begin{bmatrix} A & 2\Re(\tilde{B})^T & -2\Im(\tilde{B})^T \\ 2\Re(\tilde{B}) & 2\Re(\tilde{C}+\tilde{D}) & -2\Im(\tilde{C}+\tilde{D}) \\ -2\Im(\tilde{B}) & 2\Im(\tilde{D}-\tilde{C}) & 2\Re(\tilde{D}-\tilde{C}) \end{bmatrix}$. Finally, for the simplification by 2, as in the previous proof, we redefine the formula of H^R as it is given in the proposition above, where B, C, and D are respectively equal to two times $\tilde{B}$,$\tilde{C}$, and $\tilde{D}$.

3.4. Retraction. The retraction is an important ingredient of a Riemannian optimization and choosing an efficient retraction is crucial.

DEFINITION 3.6. *[2, 4, 33] Let $\mathcal{M}$ be a manifold and $p \in \mathcal{M}$. A retraction R_p is a map $T_p\mathcal{M} \to \mathcal{M}$, which satisfies the following properties :*

1. *$R_p(0_p) = p$;*
2. *there exists an open neighborhood $\mathcal{U}_p \subset T_p\mathcal{M}$ of 0_p such that the restriction on $\mathcal{U}_p$ is well-defined and a smooth map;*

3. *R_p satisfies the local rigidity condition*

$$DR_p(0_p) = id_{T_p\mathcal{M}}$$

where $id_{T_p\mathcal{M}}$ denotes the identity map on $T_p\mathcal{M}$.

We will use the following well-known lemma to construct a retraction on $\mathcal{M} = \mathcal{V}_n^{d^{\times r}}$ [3, 10].

LEMMA 3.7. *Let $\mathcal{M}_1, \ldots, \mathcal{M}_r$ be manifolds, $p_i \in \mathcal{M}_i$ and $\mathcal{M} = \mathcal{M}_1 \times \cdots \times \mathcal{M}_r$ and $p = (p_1, \ldots, p_r) \in \mathcal{M}$. Let $R_i : T_{p_i}\mathcal{M}_i \to \mathcal{M}_i$ be retractions. Then $R_p : T_p\mathcal{M} \to \mathcal{M}$ defined as follows: $R_p(\xi_1, \cdots, \xi_r) = (R_{p_1}(\xi_1), \cdots, R_{p_r}(\xi_r))$ for $\xi_i \in T_{p_i}\mathcal{M}_i$, $1 \leq i \leq r$, is a retraction on $\mathcal{M}$.*

Our construction of a retraction on $\mathcal{V}_n^d$ is described in the following definitions.

DEFINITION 3.8. *The Hankel matrix of degree $k, d-k$ associated to a polynomial P in $\mathbb{C}[\mathbf{x}]_d$ is given by:*

$$H_P^{k,d-k} = \left(\langle P, x^{\alpha+\beta}\rangle_d\right)_{|\alpha|=k, |\beta|=d-k}$$

In this definition, we implicit assume that we have choosen a monomial ordering (for instance the lexicographic ordering on the monomials indexing the rows and columns of $H_P^{k,d-k}$) to build the Hankel matrix. The properties of Hankel matrices that we will use are independent of this ordering. Such a matrix is called a Hankel matrix since, as in the classical case, the entries of the matrix depend on the sum of the exponents of the monomials indexing the corresponding rows and columns. This matrix construction is closely related to the classical matricization or flattening of a multilinear tensor, which can be defined in a similar way using an adequate apolar product.

When $k = 1$, using the apolar relations $\langle P, x_i x^\beta\rangle_d = \frac{1}{d}\langle \partial_{x_i} P, x^\beta\rangle_{d-1}$, we see that $H_P^{1,d-1}$ is nothing else than the transposed of the coefficient matrix of the gradient $\frac{1}{d}\nabla P$ in the basis $\left(x^\beta \binom{d-1}{\beta}^{-1}\right)_{|\beta|=d-1}$. When $P = (v^t x)^d \in \mathcal{V}_n^d$, $H_P^{1,d-1}$ can thus be written as the rank-1 matrix $v \otimes (v^t x)^{d-1}$.

DEFINITION 3.9. *For $v \in \mathbb{C}^n \setminus \{0\}$, let $\Pi_v : \mathbb{C}[\mathbf{x}]_d \to \mathcal{V}_n^d$ be the map such that $\forall Q \in \mathbb{C}[\mathbf{x}]_d$,*

$$\Pi_v(Q) = \frac{\langle \Phi(v), Q\rangle_d}{\|\Phi(v)\|_d^2}\Phi(v) \tag{3.6}$$

where $\Phi : v \in \mathbb{C}^n \mapsto (v^t x)^d \in \mathcal{V}_n^d$ is the parametrization of the Veronese variety $\mathcal{V}_n^d$. For $P \in \mathbb{C}[\mathbf{x}]_d$, let $\theta(P) \in \mathbb{C}^n$ be the first left singular vector of $H_P^{1,d-1}$. For $P \in \mathcal{V}_n^d$, let

$$\begin{aligned} R_P : T\mathcal{V}_n^d &\to \mathcal{V}_n^d \\ Q &\mapsto \Pi_{\theta(P+Q)}(P+Q) \end{aligned}$$

By the apolar identities, we check that $R_P(Q) = \left(P(\bar{u}) + Q(\bar{u})\right)(u^t x)^d$ where $u = \theta(P+Q)$. We also verify that $\Pi_{\lambda u} = \Pi_u$ for any $\lambda \in \mathbb{C} \setminus \{0\}$ and any $u \in \mathbb{C}^n \setminus \{0\}$.

By the relation (3.6), for any $v \in \mathbb{C}^n \setminus \{0\}$, $\Pi_v(Q)$ is the vector on the line spanned by $\Phi(v)$, which is the closest to Q for the apolar norm. In particular, we have $\Pi_v(\Phi(v)) = \Phi(v)$.

We verify now that R_P is a retraction on $\mathcal{V}_n^d$.

LEMMA 3.10. *Let $P \in \mathcal{V}_n^d$, P is a fixed point by Π_u where u is the first left singular vector of $H_P^{1,d-1}$.*

Proof. If $P = (v^t x)^d = \Phi(v) \in \mathcal{V}_n^d$ with $v \in \mathbb{C}^n \setminus \{0\}$, then the first left singular vector u of $H_P^{1,d-1}$ is up to a scalar equal to v. Thus we have $\Pi_u(P) = \Pi_v(\Phi(v)) = \Phi(v) = P$. ∎

PROPOSITION 3.11. *Let $P \in \mathcal{V}_n^d$. There exists a neighborhood $\mathcal{U}_P \subset \mathbb{C}[\mathbf{x}]_d$ of P such that the map $\rho : Q \in \mathcal{U}_P \mapsto \Pi_{\theta(Q)}(Q)$ is well-defined and C^∞ smooth.*

Proof. Let $P \in \mathcal{V}_n^d$ and $\theta : Q \in \mathbb{C}[\mathbf{x}]_d \to q \in \mathbb{C}^n$ where q is the first left singular vector of the SVD decomposition of $H_Q^{1,d-1}$. Let $\gamma : \mathbb{C}[\mathbf{x}]_d \to \mathcal{V}_n^d = \Phi \circ \theta$ be the composition map by the parametrization map Φ of $\mathcal{V}_n^d$.

By construction, we have $\rho : Q \mapsto \langle Q, \gamma(Q)\rangle_d \, \gamma(Q)$. Let $\mathcal{O}$ denotes the open set of homogeneous polynomials $Q \in \mathbb{C}[\mathbf{x}]_d$ such that the Hankel matrix $H_Q^{1,d-1}$ has a nonzero gap between the first and the second singular values. It follows from [13] that the map θ is well-defined and smooth on $\mathcal{O}$. As P is in $\mathcal{V}_n^d$ and $H_P^{1,d-1}$ is of rank 1, $P \in \mathcal{O}$. Let $\mathcal{U}_P$ be a neighborhood of P in $\mathbb{C}[\mathbf{x}]_d$ such that $\Phi_{|\mathcal{U}_P}$ is well-defined and smooth. As the apolar product $\langle ., . \rangle_d$ and the multiplication are well-defined and smooth on $\mathbb{C}[\mathbf{x}]_d \times \mathbb{C}[\mathbf{x}]_d$, ρ is well-defined and smooth on $\mathcal{U}_P$, which ends the proof. ∎

As $\Phi : v \in \mathbb{C}^n \mapsto (v^t x)^d \in \mathcal{V}_n^d$ is a parametrization of the Veronese variety $\mathcal{V}_n^d$, the tangent space of $\mathcal{V}_n^d$ at a point $\Phi(v)$ is spanned by the first order vectors $D\Phi(v)\, q$ of the Taylor expansion of $\Phi(v + t\, q) = \Phi(v) + t\, D\Phi(v)\, q + O(t^2)$ for $q \in \mathbb{C}^n$. We are going to use this observation to prove the rigidity property of R_p.

PROPOSITION 3.12. *For $P \in \mathcal{V}_n^d, Q \in T_P(\mathcal{V}_n^d)$,*

$$P + t\, Q - R_P(t\, Q) = O(t^2).$$

Proof. As $P \in \mathcal{V}_n^d, Q \in T_P(\mathcal{V}_n^d)$, there exist $v, q \in \mathbb{C}^n$ such that $P = \Phi(v), Q = D\Phi(v)\, q$. In particular, we have $P + t\, Q - \Phi(v + t\, q) = O(t^2)$. This implies that $H_{P+t\, Q}^{1,d-1} - H_{\Phi(v+t\, q)}^{1,d-1} = O(t^2)$. By continuity of the singular value decomposition, we have $u_t - v_t = O(t^2)$ where $u_t = \theta(P + t\, Q)$ and $v_t = \theta(\Phi(v + t\, q))$ are respectively the first left singular vectors of $H_{P+t\, Q}^{1,d-1}$ and $H_{\Phi(v+t\, q)}^{1,d-1}$.

Since $H_{\Phi(v+t\, q)}^{1,d-1}$ is a matrix of rank 1 and its image is spanned by $v + t\, q$, v_t is a non-zero scalar multiple of $v + t\, q$ and we have $\Pi_{v_t} = \Pi_{v+t\, q}$. By continuity of the projection on a line, we have

$$\Pi_{u_t}(P + t\, Q) = \Pi_{v_t}(P + t\, Q) + O(t^2) = \Pi_{v+t\, q}(P + t\, Q) + O(t^2).$$

Since $\Phi(v + t\, q) = \Phi(v) + t\, D\Phi(v)q + O(t^2) = P + t\, Q + O(t^2)$, we have

$$\Pi_{v+t\, q}(P + t\, Q) = \Pi_{v+t\, q}(\Phi(v + t\, q)) + O(t^2) = \Phi(v + t\, q) + O(t^2).$$

We deduce that

$$\begin{aligned} &P + t\, Q - R_P(t\, Q) = P + t\, Q - \Pi_{u_t}(P + t\, Q) \\ &= P + t\, Q - \Phi(v + t\, q) + (\Phi(v + t\, q) - \Pi_{v_t}(P + t\, Q)) + (\Pi_{v_t}(P + t\, Q) - \Pi_{u_t}(P + t\, Q)) \\ &= \Phi(v) + t\, D\Phi(v)q - \Phi(v + t\, q) + O(t^2) = O(t^2), \end{aligned}$$

which proves the proposition. ∎

PROPOSITION 3.13. *Let $P \in \mathcal{V}_n^d$. The map $R_P : T_P\mathcal{V}_n^d \to \mathcal{V}_n^d$, $Q \mapsto R_P(Q) = \Pi_{\theta(P+Q)}(P+Q)$ is a retraction operator on the Veronese manifold $\mathcal{V}_n^d$.*

Proof. We have to prove that R_P verifies the three properties in Definition 3.6.

1. $R_P(0_P) = \Pi_{\theta(P)}(P + 0_P) = \Pi_{\theta(P)}(P) = P$, by using Lemma 3.10.
2. Let $S_P : T_P\mathcal{V}_n^d \to \mathbb{C}[\mathbf{x}]_d$, $Q \mapsto P + Q$. The map S_P is well-defined and smooth on $T_P\mathcal{V}_n^d$. By Proposition 3.11, Π is well-defined and smooth in a neighborhood $\mathcal{U}_P$ of $P \in \mathcal{V}_n^d$. Thus $R_P = \rho \circ S_P$ is well-defined and smooth in a neighborhood $\mathcal{U}'_P \subset T\mathcal{V}_n^d$ of 0_P.
3. By Proposition 3.12,

$$(P + tQ) - R_P(t\,Q) = O(t^2),$$

which implies that $\frac{d}{dt}R_P(t\,Q)\mid_{t=0} = Q$, or equivalently $DR_P(0_P).(Q) = Q$. Therefore we have $DR_P(0_P) = id_{T_P\mathcal{V}_n^d}$. ∎

3.5. Choice of the initial point. The choice of the initial point is a crutial step in Riemannian Newton iterative methods. We use the direct algorithm of [24], based on the computation of generalized eigenvectors and generalized eigenvalues of pencils of Hankel matrices (see also [37]), to compute an initial rank-r approximation. This algorithm, denoted SHD, works only with r such that $r < r_g$ and $\iota \leq \lfloor \frac{d-1}{2} \rfloor$, where ι denotes the interpolation degree of the points in the rank-r decomposition [20, Ch.4]. The rationale behind choosing the initial point with this method is when the symmetric tensor is already of symmetric rank r with $r < r_g$ and $\iota \leq \lfloor \frac{d-1}{2} \rfloor$, then this computation gives a good numerical approximation of the exact decomposition, so that the Riemannian Newton algorithm needs few iterations to converge numerically. We will see in the numerical experiment in section 4 that this initial point is an efficient choice to get a good low rank approximation of a symmetric tensor.

3.6. Riemannian Newton algorithm with trust region scheme for the STA problem. In the previous sections we desribed all the ingredients of a Riemannian Newton algorithm for the STA problem. Unfortunately, the convergence of this algorithm may not occur from the beginning, that is because Newton method converges if the initial point is close enough to a fix point solution. By adding a trust region scheme to the Riemannian Newton algorithm, we enhance the algorithm, with the desirable global properties of convergence to a local minimum, with a local superlinear rate of convergence [2, Chapter 7], [1].

Let $p_k = (W_k, \Re(V_k), \Im(V_k)) \in \mathcal{N}$. The idea is to approximate the objective function f to its second order Taylor series expansion in a ball of center $0_{p_k} \in T_{p_k}\mathcal{N}$ and radius Δ_k denoted by $B_{\Delta_k} := \{u \in T_{p_k}\mathcal{N} \mid ||u|| \leq \Delta_k\}$, and to solve the subproblem

$$\min_{u \in B_{\Delta_k}} m_{p_k}(u) \tag{3.7}$$

where $m_{p_k}(u) := f(p_k) + G_k^T u + \frac{1}{2}u^T H_k u$ and G_k is the gradient at p_k (Proposition 3.4) and H_k is the Hessian of f at p_k (Proposition 3.5).

By solving (3.7), we obtain a solution $u_k \in T_{p_k}\mathcal{N}$. Accepting or rejecting the candidate new point $p_{k+1} = R_{p_k}(u_k)$ is based on the quotient $\rho_k = \frac{f(p_k) - f(p_{k+1})}{m_{p_k}(0) - m_{p_k}(u_k)}$. If ρ_k exceeds 0.2 then the current point p_k is updated, otherwise the current point p_k remains unchanged.
The radius of the trust region Δ_k is also updated based on ρ_k. We choose to update the trust region as in [10] with a few changes.

Let $\Delta_{p_0} := 10^{-1}\sqrt{\frac{d}{r}\sum_{i=1}^{r}||w_i^0||^2}$, $\Delta_{max} := \frac{1}{2}||P||_d$. We take the initial radius as $\Delta_0 = min\{\Delta_{p_0}, \Delta_{max}\}$, if $\rho_k > 0.6$ then the trust region is enlarged as follow: $\Delta_{k+1} = min\{2||u_k||, \Delta_{max}\}$. Otherwise the trust region is shrinked by taking $\Delta_{k+1} = min\left\{(\frac{1}{3} + \frac{2}{3}.(1 + e^{-14.(\rho_k - \frac{1}{3})})^{-1})\Delta_k, \Delta_{max}\right\}$.

We choose the so-called dogleg method to solve the subproblem (3.7) [41]. Let p_N be the Newton direction given by the Newton equation $Hp_N = -G$, and let p_c denotes the Cauchy point given by $p_c = -\frac{G^TG}{G^THG}G$. Then the optimal solution p^* of (3.7) by the dogleg method is given as follows:

$$p^* = \begin{cases} p_N & \text{if } ||p_N|| \leq \Delta \\ -\frac{\Delta}{||G||}G & \text{if } ||p_N|| > \Delta \text{ end } ||p_c|| \geq \Delta \\ p_I & \text{otherwise} \end{cases}$$

where p_I is the intersection of the boundary of the sphere B_Δ and the vector pointing from p_c to p_N.
The algorithm of the Riemannian Newton method with trust region scheme for the STA problem is denoted by RNS-TR, and it is given by:

Algorithm 3.2 Riemannian Newton algorithm with trust region sheme for the STA problem (RNS-TR)

Input: The homogeneous polynomial $P \in \mathbb{C}[\mathbf{x}]_d$ associated to the symmetric tensor to approximate, $r < r_g$
Choose initial point (W_0, V_0)
while the method has not converged **do**
1. Compute the vectors G_1 and G_2 and the matrices A, B, C and D
2. Compute the vector G^R and the matrix H^R
3. Compute the basis matrix Q
4. Solve the subproblem (3.7) for the search direction $u_k \in B_{\Delta_k}$ by using the dogleg method
5. Compute the candidate next new point $p_{k+1} = R_{p_k}(u_k)$
6. Compute the quotient ρ_k
7. Accept or reject p_{k+1} based on the quotient ρ_k
8. Update the trust region radius Δ_k

end while
Output: $(\tilde{W}, \tilde{V})$

The Algorithm 3.2 is stopped when $\Delta_k \leq 10^{-3}$, or when the maximum number of iterations exceeds N_{max}.

REMARK 3.14. *For the implementation of the retraction in the step 5 in Algorithm 3.2; let us assume that the subproblem (3.7) is solved at a point* $p = (W, \Re(V), \Im(V)) \in \mathcal{N}$, *such that* $W = (w_i)_{1\leq i\leq r}$, *and* $V = [v_i]_{1\leq i\leq r}$, *in local coordinates with respect to the basis* Q *as in Proposition 3.2. It yields a solution vector* $\hat{p} \in \mathbb{R}^{r+r(2n-1)}$. *The tangent vector* $p^* \in T_p\mathcal{N}$ *is given by:* $p^* = Q\,\hat{p}$, *of size* $r + 2nr$. *For each* $j \in \{1, \cdots, r\}$, *Now, let denote* $w_j^* = p^*[j]$, *and* $v_j^* = p^*[r + (j-1)n + 1 : r + jn] + p^*[r + rn + (j-1)n + 1 : r + rn + jn]i$.

Using Lemma 3.7 for each j *in* $\{1, \cdots, r\}$, *we denote* $P_j := w_j(v_j^t x)^d \in \mathcal{V}_n^d$, $tg_j := w_j^*(v_j^t x)^d + dw_j(v_j^t x)^{d-1}(v_j^{*t} x) \in T_{P_j}\mathcal{V}_n^d$.

Using the retraction in Proposition 3.13, we have $R_{P_j}(tg_j) = (P_j + tg_j)(\overline{q}_j)\ (q_j^t x)^d$, *where* q_j *is the first left singular vector of* $H^{1,d-1}_{P_j+tg_j}$.

Finally, we define the new point $(\tilde{W}, \Re(\tilde{V}), \Im(\tilde{V})) \in \mathcal{N}$ *with* $\tilde{W} = (\tilde{w}_j)_{1\leq j\leq r}$, $\tilde{V} = [\tilde{v}_j]_{1\leq j\leq r}$, *such that* $\tilde{w}_j = |z_j|$, $\tilde{v}_j = e^{i\frac{\theta_j}{d}} q_j$, *where* $z_j := (P_j + tg_j)(\overline{q}_j) \in \mathbb{C}$ *and* $\theta_j :=$ *argument of* z_j.

REMARK 3.15. *In order to handle ill-conditioned Hessian matrices in the RNS-TR iterations, we use the Moore-Penrose pseudoinverse [8, 32, 49], which slows down the speed of computation. Ill-conditioned Hessian matrices can appear in cases where some vectors* v_i *of the rank* r*-approximation span close lines, which yields a singularity problem in the iteration. This can be happen when the tensor is well approximated by a tensor of rank smaller than the rank* r *used in the iterations (e.g. Example 4.1, Example 4.3 with* $r > 2$*). Another singular case is when the border rank of the symmetric tensor is not equal to its symmetric rank [10, 16], [34, section 2.4]. For example, the tensor* $P = (v_0^t x)(v_1^t x)^{d-1} + \epsilon T$, *with* $v_0, v_1 \in \mathbb{R}^n$, $T \in \mathbb{R}[x]_d$ *and* ϵ *very small, is close to the tensor* $(v_0^t x)(v_1^t x)^{d-1} = \lim_{\delta\to 0} \frac{1}{d\,\delta}(((v_1 + \delta v_0)^t x)^d - (v_1^t x)^d)$ *of border rank 2 and symmetric rank* $d+1$. *It can be very well approximated by a tensor of rank* 2, *with two vectors of almost the same direction.*

4. Numerical experiments. In this section we present three numerical experiments using the RNS-TR algorithm. In subsection 4.1 we choose randomly some real and complex examples of symmetric tensors of low rank, that we perturb randomly. To analyze the practical behavior, we compare the RNS-TR algorithm with an initial point given by the SHD method, with other reference methods. In subsection 4.2 we explore the performance of the RNS algorithm (which is the RNS-TR algorithm after removing the trust region scheme) to find a best real rank-1 approximation of a real symmetric tensor. In subsection 4.3 we compare with some examples of real and complex valued symmetric tensors, the performance of the RNS-TR algorithm with state-of-the-art methods for best low rank approximation of tensors of high rank. The RNS-TR algorithm is implemented in Julia version 1.1.1 in the package `TensorDec.jl`[1]. We use a Julia implementation for the method SPM tested in section 4.1. The solvers from Tensorlab v3 [53] are run in MATLAB 7.10. The experimentation have been done on a Dell Window desktop with 8 GB memory and Intel 2.3 GHz CPU.

4.1. Approximation of perturbations of low rank symmetric tensors. In this section, we consider perturbations of random low rank tensors. For a given rank r, we choose r random vectors v_i of size n, obeying Gaussian distributions and compute the symmetric tensor $T = \sum_{i=1}^r (v_i^t x)^d$ of order d. We choose a random symmetric tensor T_{err} of order d, with coefficients also obeying Gaussian distributions, normalize it so that its apolar norm is ϵ and add it to T: $\tilde{T} = T + \epsilon \frac{T_{err}}{\|T_{err}\|_d}$. We apply the different approximation algorithms to $\tilde{T}$ and compute the relative error err$= \frac{\|T_* - T\|_d}{\epsilon}$ between the approximation T_* of rank r computed by the algorithm and the rank-r tensor T. We run this computation for 100 random instances and report the minimal, median and maximal relative error, the average number of iterations N (rounded to the closest integer) and the average time t (in seconds).

As the initial tensor $\tilde{T}$ is in a ball of radius ϵ centered at the tensor T of rank r, we expect T_* to be closer to T and the relative error to be less than 1.

[1]It can be obtained from "https://gitlab.inria.fr/AlgebraicGeometricModeling/TensorDec.jl". See functions RNS and RNS-TR.

We compare the RNS-TR method with the initial point computed by SHD algorithm, with the recent Subspace Power Method (SPM) of [29] and the state-of-the-art implementation CPD-NLS of the package Tensorlab v3 [53], which is designed for the canonical polyadic decomposition [27], but using it with symmetric initial point provides symmetric approximation. Since SPM works for even order tensors with real coefficients, the comparison in Table 1 is run for tensors in $\mathcal{S}^4(\mathbb{R}^{10})$. In Tables 2 and 3, we compare CPD-NLS and RNS-TR for tensors in $\mathcal{S}^d(\mathbb{C}^{10})$ of order $d = 3, 4$ and with complex coefficients. These tables also provide a numerical comparison with the low rank approximation methods tested in Example 5.4 of [39], since the setting is the same.

The timing for the method RNS-TR includes the computation of the initial point by the SHD algorithm. For the methods SPM and RNS-TR, the iterations are stopped when the distance between two consecutive iterates is less than 10^{-6} or when the maximal number of iterations ($N_{\max} = 200$ in this experimentation) is reached.

TABLE 1
*Computational results of **SPM** and **RNS-TR** for rank-r approximations in $\mathcal{S}^4(\mathbb{R}^{10})$.*

		err_{spm}			t_{spm}	N_{spm}	err_{rns}			t_{rns}	N_{rns}
r	ϵ	min	med	max	avg	avg	min	med	max	avg	avg
1	1	0.042	0.099	0.171	0.031	21	0.0389	0.102	2.607	0.094	3
	10^{-1}	0.048	0.102	0.16	0.034	23	0.047	0.102	0.16	0.082	2
	10^{-2}	0.05	0.095	0.164	0.031	21	0.05	0.096	0.165	0.071	2
	10^{-4}	0.055	0.331	3.9	0.036	21	0.062	0.104	0.185	0.081	2
	10^{-6}	0.631	27.7	673.8	0.032	19	0.053	0.103	0.162	0.074	2
2	1	0.076	0.151	2.1	0.079	49	0.098	0.154	0.233	0.122	3
	10^{-1}	0.097	0.148	0.226	0.074	49	0.091	0.156	0.23	0.124	2
	10^{-2}	0.083	0.153	0.222	0.062	44	0.085	0.156	0.217	0.102	2
	10^{-4}	0.136	0.774	12.7	0.069	48	0.092	0.155	0.252	0.107	2
	10^{-6}	3.2	91	943.5	0.066	47	0.1	0.157	0.238	0.104	2
3	1	0.134	0.183	0.325	0.097	78	0.117	0.19	0.265	0.225	4
	10^{-1}	0.122	0.191	0.248	0.091	76	0.146	0.199	1.9	0.202	3
	10^{-2}	0.13	0.193	0.898	0.076	78	0.125	0.206	0.375	0.125	2
	10^{-4}	0.225	1.2	12.1	0.093	76	0.148	0.206	0.394	0.13	2
	10^{-6}	24	148	1264.8	0.106	79	0.131	0.201	0.518	0.132	2
4	1	0.172	0.226	473482.3	0.147	109	0.168	0.231	0.705	0.359	4
	10^{-1}	0.152	0.217	0.285	0.151	112	0.16	0.227	13.4	0.258	3
	10^{-2}	0.162	0.224	0.376	0.149	110	0.169	0.251	0.401	0.17	2
	10^{-4}	0.385	2.3	16	0.153	115	0.18	0.24	0.789	0.162	2
	10^{-6}	31.6	231	4218.8	0.15	109	0.181	0.244	0.469	0.164	2
5	1	0.184	0.245	1500.9	0.192	143	0.171	0.252	2.5	0.75	6
	10^{-1}	0.179	0.243	0.322	0.19	141	0.188	0.255	0.339	0.327	3
	10^{-2}	0.177	0.252	0.329	0.196	147	0.198	0.287	0.478	0.221	2
	10^{-4}	0.339	3.1	29.6	0.19	141	0.207	0.297	3	0.2	2
	10^{-6}	48.8	330.9	1921.2	0.196	144	0.213	0.3	1	0.197	2

In all these experimentations (see Table 1, Table 2, Table 3), the number of iterations of the RNS-TR method is significantly smaller than the number of iterations of the other methods. In SPM, the number of iterations to get an approximation of a single rank-1 term of the approximation is about 20, indicating a practical linear convergence as predicted by the theory [29, Theorem 5.10]. As the method CPD-NLS is based on a quasi-Newton iteration, its local convergence is sub-quadratic, which also explains the relatively high number of iterations. The local quadratic convergence of RNS-TR is revealed, in these experimentations, by the low number of iterations.

The cost of an iteration appears to be higher in RNS-TR method than in the other methods. Nevertheless, the total time is of the same order. The timing of SPM method is better than the timing of RNS-TR, which is better than CPD-NLS timing.

TABLE 2
*Computational results of **CPD-NLS** and **RNS-TR** for rank-r approximations in $\mathcal{S}^3(\mathbb{C}^{10})$.*

r	ϵ	err_{cpd} min	err_{cpd} med	err_{cpd} max	t_{cpd} avg	N_{cpd} avg	err_{rns} min	err_{rns} med	err_{rns} max	t_{rns} avg	N_{rns} avg
1	1	0.382	32.9	311.7	0.035	23	0.137	0.207	0.298	0.012	2
	10^{-1}	0.155	9.6	324.4	0.024	11	0.132	0.208	0.306	0.015	2
	10^{-2}	0.047	9.5	116.5	0.02	9	0.158	0.206	0.304	0.015	2
	10^{-4}	0.331	10.9	58.4	0.023	9	0.14	0.205	0.348	0.046	5
	10^{-6}	0.015	8.6	74.5	0.024	9	0.15	0.206	0.278	0.047	6
2	1	10.7	135.7	315	0.09	35	0.219	0.306	0.379	0.055	3
	10^{-1}	8.7	35.5	318.4	0.044	15	0.239	0.312	0.392	0.029	2
	10^{-2}	5.8	32.4	112.2	0.037	12	0.237	0.326	0.624	0.026	2
	10^{-4}	4.8	33	125.2	0.04	12	0.223	0.314	0.627	0.026	2
	10^{-6}	4.3	33.2	99.2	0.037	12	0.225	0.313	0.482	0.026	2
3	1	33	173.8	318.5	0.193	47	0.29	0.362	0.425	0.096	4
	10^{-1}	19.2	54.1	147.2	0.072	16	0.3	0.375	0.53	0.064	3
	10^{-2}	17.9	57.9	152.3	0.066	14	0.331	0.444	0.695	0.039	2
	10^{-4}	17.3	57.2	120	0.063	14	0.331	0.448	0.729	0.04	2
	10^{-6}	17	54.8	129.4	0.062	13	0.313	0.435	0.745	0.039	2
4	1	92.4	188.3	372.5	0.254	53	0.347	0.429	0.489	0.126	4
	10^{-1}	32.3	83.2	535.2	0.101	19	0.352	0.425	0.512	0.088	3
	10^{-2}	34.9	84.9	500	0.085	16	0.388	0.555	0.803	0.054	2
	10^{-4}	40.2	79.8	4.2e+4	0.08	15	0.446	0.563	0.671	0.048	2
	10^{-6}	38.1	90.1	4.6e+6	0.081	15	0.438	0.603	0.94	0.048	2
5	1	89.8	214.5	344.8	0.37	73	0.415	0.475	0.55	0.163	4
	10^{-1}	50.3	109.4	352.2	0.13	24	0.38	0.46	0.554	0.116	3
	10^{-2}	48.7	101.5	223	0.106	19	0.402	0.525	0.761	0.085	3
	10^{-4}	39.2	97.6	215.8	0.1	17	0.547	0.805	1.1	0.061	2
	10^{-6}	47.6	101.9	178.3	0.111	19	0.525	0.782	0.972	0.061	2

TABLE 3
*Computational results of **CPD-NLS** and **RNS-TR** for rank-r approximations in $\mathcal{S}^4(\mathbb{C}^{10})$.*

r	ϵ	err_{cpd} min	err_{cpd} med	err_{cpd} max	t_{cpd} avg	N_{rns} avg	err_{rns} min	err_{rns} med	err_{rns} max	t_{rns} avg	N_{rns} avg
1	1	0.004	339.5	1.4e+3	0.047	16	0.064	0.109	0.175	0.051	2
	10^{-1}	0.004	13.2	1.1e+3	0.033	11	0.066	0.108	0.153	0.05	2
	10^{-2}	0.001	9.9	688.2	0.033	11	0.064	0.12	0.150	0.05	2
	10^{-4}	0.016	17.1	127.6	0.032	10	0.055	0.118	0.154	0.05	2
	10^{-6}	0.007	12.5	155	0.032	11	0.064	0.111	0.157	0.05	2
2	1	10.3	636.6	2.2e+3	0.096	23	0.112	0.165	0.218	0.116	2
	10^{-1}	8.6	87	6.7e+3	0.072	17	0.115	0.166	0.217	0.086	2
	10^{-2}	4.3	58.3	448.5	0.064	14	0.118	0.159	0.257	0.085	2
	10^{-4}	3.8	62.8	9.9e+5	0.063	14	0.122	0.168	0.298	0.092	2
	10^{-6}	6.9	63.9	4.9e+3	0.061	14	0.111	0.167	0.221	0.086	2
3	1	76.2	753.8	2.4e+3	0.14	28	0.156	0.202	0.245	0.211	3
	10^{-1}	19.4	150.9	5e+3	0.096	19	0.155	0.203	0.272	0.139	2
	10^{-2}	19.3	118.9	416.3	0.087	16	0.156	0.204	0.256	0.129	2
	10^{-4}	16.2	132.3	512.6	0.087	16	0.15	0.209	0.294	0.145	2
	10^{-6}	17.7	138.1	4e+3	0.084	16	0.156	0.203	0.262	0.146	2
4	1	87.5	929.6	1.8e+3	0.229	39	0.191	0.232	0.297	0.34	3
	10^{-1}	38.8	224.4	3.9e+3	0.137	22	0.192	0.245	0.309	0.24	2
	10^{-2}	58.4	213.1	6.8e+3	0.115	18	0.198	0.256	0.337	0.188	2
	10^{-4}	36	195.8	828.8	0.137	20	0.199	0.248	0.354	0.188	2
	10^{-6}	257.3	193.2	689	0.117	18	0.196	0.252	0.344	0.191	2
5	1	424.8	1.1e+3	3.4e+3	0.291	41	0.213	0.262	0.314	0.399	3
	10^{-1}	118.4	318.7	5.3e+3	0.176	24	0.222	0.273	0.336	0.299	3
	10^{-2}	103.8	274	608.1	0.153	19	0.232	0.293	0.473	0.204	2
	10^{-4}	68.7	224.1	9.5e+3	0.154	20	0.247	0.296	0.469	0.176	2
	10^{-6}	56.5	269	2.9e+4	0.157	20	0.247	0.293	0.427	0.175	2

It should be noticed that CPD-NLS is not exploiting the symmetry of the tensors in these computations, whereas the other methods applies directly to symmetric tensors.

These experimentations also show that the numerical quality of low rank approximation is better for RNS-TR. In most of the computations, the relative error in RNS-TR method is less than 1, corroborating the idea that RNS-TR found the best low rank approximation in these cases. In the other methods, the local optimum found can be very far from the best low rank approximation. The same remark applies to the numerical results in [39, Example 5.4] for GP method and small perturbations ($\epsilon \in \{10^{-2}, 10^{-4}, 10^{-6}\}$), whereas the numerical quality in GP-OPT method of the same order as in RNS-TR though slightly worst. This can be explained by the choice of a random vector as initial point in SPM and CPD-NLS methods. The relatively high error in SPM method for small perturbations ($\epsilon = 10^{-6}$) can be explained by the fact that the threshold used to stop the iterations is of the same order as the perturbation ϵ, which deteriorates the approximation quality of the final low rank tensor.

4.2. Best rank-1 approximation and spectral norm. Let $P \in \mathcal{S}^d(\mathbb{R}^n)$, a best real rank-1 approximation of P is a minimizer of the optimization problem

$$\text{dist}_1(P) := \min_{X \in \mathcal{S}^d(\mathbb{R}^n), \text{rank} X = 1} ||P - X||_d^2 = \min_{(w,v) \in \mathbb{R} \times \mathbb{S}^{n-1}} ||P - w(v^t x)^d||_d. \tag{4.1}$$

where $\mathbb{S}^{n-1} = \{v \in \mathbb{R}^n \mid ||v|| = 1\}$ is the unit sphere. This problem is equivalent to $\min_{X \in \mathcal{T}^d(\mathbb{R}^n), \text{rank} X = 1} ||P^\tau - X||_F^2$ since at least one global minimizer is a symmetric rank-1 tensor [55].

The real spectral norm of $P \in \mathcal{S}^d(\mathbb{R}^n)$, denoted by $||P||_{\sigma,\mathbb{R}}$ is by definition:

$$||P||_{\sigma,\mathbb{R}}^2 := \max_{v \in \mathbb{S}^{n-1}} |P(v)| \tag{4.2}$$

The two problems (4.1) and (4.2) are related by the following equality:

$$\text{dist}_1(P)^2 = ||P||_d^2 - ||P||_{\sigma,\mathbb{R}}^2$$

which we deduce by simple calculus and properties of the apolar norm (see also [19, 55]):

$$\begin{aligned}
\text{dist}_1(P)^2 &= \min_{(w,v) \in \mathbb{R} \times \mathbb{S}^{n-1}} ||P - w(v^t x)^d||_d^2 \\
&= \min_{(w,v) \in \mathbb{R} \times \mathbb{S}^{n-1}} ||P||_d^2 - 2\langle P, w(v^t x)^d \rangle_d + ||w(v^t x)^d||_d^2 \\
&= \min_{(w,v) \in \mathbb{R} \times \mathbb{S}^{n-1}} ||P||_d^2 - 2wP(v) + w^2 \\
&= \min_{v \in \mathbb{S}^{n-1}} ||P||_d^2 - |P(v)|^2 = ||P||_d^2 - \max_{v \in \mathbb{S}^{n-1}} |P(v)|^2 = ||P||_d^2 - ||P||_{\sigma,\mathbb{R}}^2.
\end{aligned}$$

Therefore if v is a global maximizer of (4.2) such that $w = P(v)$, then $w\, v^{\otimes d}$ is a best rank-1 approximation of P. Herein, a rank-1 approximation $w\, v^{\otimes d}$, such that $w = P(v)$ and $||v|| = 1$, is better when $|w|$ is higher. Therefore, in the following experimentation, we report the weight w obtained by the different methods.

In [40] the authors present an algorithm called hereafter "SDP" based on semidefinite relaxations to find a best real rank-1 approximation of a real symmetric tensor by finding the global optimum of P on $\mathbb{S}^{n-1}$. In our second numerical experiment we choose to apply, on some of examples from [40] our method RNS with real initial point chosen according to the SHD algorithm as in subsection 3.5. Recall that the RNS algorithm is an exact Riemannian Newton method, and then it has a local

quadratic rate of convergence [2, Theorem 6.3.2]. Note that the computations in [40] are implemented in MATLAB 7.10 on a Dell Linux desktop with 8 GB memory and Intel 2.8 GHz CPU.

We denote by $w_{rns}(v_{rns}^t x)^d$ (resp. $w_{sdp}(v_{sdp}^t x)^d$) the rank-1 approximation of $P \in \mathbb{R}[\mathbf{x}]_d$ obtained by RNS method (resp. SDP method). Note that $|w_{sdp}|$ is the spectral norm of P, since SDP gives a best rank-1 approximation. We report the time spent by RNS method in the seconds (s) and we denote it by t_{rns}, while the computation time in SDP method, in the format hours:minutes:seconds, is denoted by t_{sdp}. We denote by N_{rns} the number of iterations needed for the convergence of RNS method. We denote by d_0 the norm between P and the initial point of RNS method, and by d_* the norm between P and the local minimizer obtained after the convergence of the RNS method.

EXAMPLE 4.1. [40, Example 3.5]. Consider the tensor $P \in \mathcal{S}^3(\mathbb{R}^n)$ with entries:

$$(P)_{i_1,i_2,i_3} = \frac{(-1)^{i_1}}{i_1} + \frac{(-1)^{i_2}}{i_2} + \frac{(-1)^{i_3}}{i_3}.$$

TABLE 4
Comparision of RNS-TR and SDP for Example 4.1

n	10	20	30	40	50
$\|w_{rns}\|$	17.8	34.2	50.1	65.9	81.6
d_0	32.4	80.8	133.5	188	290.3
d_*	28.4	64.9	102.2	139.8	177.8
t_{rns}	0.099	0.782	3.2	7.2	14.9
N_{rns}	5	6	8	7	6
$\|w_{SDP}\|$	17.8	34.2	50.1	65.9	81.6
t_{SDP}	0:00:02	0:00:06	0:00:30	0:04:05	0:32:45

In this example the RNS algorithm finds a best rank-1 approximation, which implies that the RNS algorithm found a minimizer in a neighborhood of the initial point chosen by the SHD algorithm, and it is in fact a global minimizer, and it converged quadratically to this point with very reduced time compared to the SDP algorithm especially when n grows.

EXAMPLE 4.2. [13, Example 3.7]. Consider the tensor $P \in \mathcal{S}^5(\mathbb{R}^n)$ given as:

$$(P)_{i_1,\ldots,i_5} = (-1)^{i_1}\log(i_1) + \cdots + (-1)^{i_5}\log(i_5).$$

TABLE 5
Comparison of RNS-TR and SDP for Example 4.2

n	10	20	30	40	50
$\|w_{rns}\|$	1.100e+2	8.833e+2	2.697e+3	6.237e+3	11.504e+3
d_0	526.1	6.559e+3	26.318e+3	64.268e+3	132.213e+3
d_*	477.5	6.096e+3	24.643e+3	60.435e+3	121.892e+3
t_{rns}	0.058	0.501	3.8	18.3	34.8
N_{rns}	5	5	6	6	6
$\|w_{sdp}\|$	1.100e+2	8.833e+2	2.697e+3	6.237e+3	
t_{sdp}	0:00:01	0:00:22	0:01:18	0:22:30	

In the experimentation results of Table 5 we find, as in the previous example, that RNS finds a best rank-1 approximation with very reduced time compared to

t_{sdp} which increases a lot with n. For example, for $n = 20$, SDP method needs 22.5 minutes to find a best rank-1 approximation. For the last column in Table 5, for $n = 25$, no experimental result is available for SDP method because of its difficulty to treat the problem due to its high complexity. Nevertheless, for $n = 25$, the RNS algorithm needs only 6 iterations to converge in 34.8 seconds to a local minimum, which we expect but cannot certify to be the best rank-1 approximation.

EXAMPLE 4.3. [40, Example 3.10]. Let $P \in \mathcal{S}^d(\mathbb{R}^n)$ such that

$$(P)_{i_1,\dots,i_d} = \sin(i_1 + \cdots + i_d).$$

In this example we compare the performance of RNS, SPD, SHOPM [30], SDF-NLS and CCPD-NLS, which are non-linear least-square solvers for the symmetric decomposition from Tensorlab v3, all together, for $d = 3$ with $n = 10, 15, 20, 25$, and for $d = 4$ with $n = 10, 15$. The numerical results of SPD and SHOPM (resp. SDF-NLS and CCPD-NLS) are taken from [40] (resp. the test of the methods implemented in Tensorlab V3.0 in MATLAB 7.10). For CCPD-NLS and SDF-NLS we run 50 instances and we take the mean of the absolute value of the 50 weights given by each of these two methods. The time spent by SHOPM and SDP methods is not available in [40]. We mention the time spent by the other methods in this comparison. We denote by t_{sdf} (resp. t_{ccpd}) the average of time in seconds(s) spent by SDF-NLS (resp. CCPD-NLS). We denote by N_{sdf} (resp. N_{ccpd}) the number of iterations in average needed in the SDF-NLS (resp. CCPD-NLS) method.

TABLE 6
Computational results for Example 4.3 with $d = 3, 4$.

	$d=3$				$d=4$	
n	10	15	20	25	10	15
$\lvert w_{rns}\rvert$	12.1	22.1	33	44.5	27.3	58.8
t_{rns}	0.071	0.278	0.86	1.701	0.221	0.85
N_{rns}	6	8	8	8	6	6
$\lvert w_{sdf}\rvert$	11.2	20.7	31.5	44.2	25.2	55.6
t_{sdf}	0.132	0.118	0.122	0.12	0.152	0.176
N_{sdf}	18	19	19	18	20	21
$\lvert w_{ccpd}\rvert$	11.2	20.7	31.6	44.2	24.9	55.1
t_{ccpd}	0.124	0.089	0.084	0.094	0.101	0.158
N_{ccpd}	18	18	18	19	20	20
$\lvert w_{shopm}\rvert$	3	13.7	27	33.3	1.2	50.4
$\lvert w_{sdp}\rvert$	12.1	22.1	33	44.5	27.3	61.4

The computational results presented in Table 6 show that SHOPM didn't find best rank-1 approximation. The mean, in all cases of n, of the absolute value of weights given by 50 instances with CCPD-NLS or SDF-NLS is not equal to $|w_{SDP}|$, which is as we mentioned before refree to the best rank-1 approximation, it is trivial since for each instance we start from a random initial point and we find one of the local-minima which is can be or not the best rank-1 approximation, this mean is in fact the mean of the absolute value of weights of local minima for rank-1 approximation. On the other side, RNS method found a best rank-1 approximation in all cases except when $d = 4$ and $n = 15$; but even with this case the rank-1 approximation given by the RNS method is better than SHOPM methods. Note that we run RNS only for one instance since SHD gives a unique rank-1 initial point.

These numerical experiments show that RNS algorithm with initial point chosen according to the SHD algorithm gives a best real rank-1 approximation for real symmetric tensor in all examples except Example 4.3 with $d = 4$, $n = 15$. We noticed that the combination between the RNS algorithm and this method to choose the initial point respectively outperforms the SDP method in term of the consumed time, and the SDF-NLS, CCPD-NLS and SHOPM methods in term of the rank-1 approximation given by each of these algorithms. Nevertheless, we have no certification that RNS-TR method converges to a best rank-1 approximation. All we can say is that RNS method with initial point chosen by using the SHD algorithm is an efficient method to get a good real rank-1 approximation of a real symmetric tensor.

4.3. RNS-TR, CCPD-NLS, and SDF-NLS for symmetric rank-r approximation. In this section, we consider two examples of a real and a complex valued sparse symmetric tensors of high rank, in order to compare the performance of RNS-TR with initial point computed by SHD, with state-of-the-art non-linear least-square solvers CPD-NLS, CCPD-NLS and SDF-NLS from Tensolab v3 with initial point following a standard normal distribution. These solvers employ factor matrices as parameterization and use a trust region method with dogleg steps called "NLS-GNDL" which means that they use the Gauss-Newton approximation of the Hessian matrix. Thus, mainly the difference between RNS-TR and the other solvers from tensorlab CPD-NLS, CCPD-NLS, and SDF-NLS is the parameterization using the Veronese manifolds, the use of the exact Hessian matrix and the choice of the initial point by SHD method. Note that we use CPD-NLS with symmetric initial point in order to obtain a symmetric approximation.

We fixe 200 iterations as maximal number of iterations, and we run 50 instances for these four methods and we report the minimal, median and maximal residual error, the average of time t (in seconds), and the average number of iterations N (rounded to the closet integer). The time of the RNS-TR method includes the time of the SHD method for the initial point.

EXAMPLE 4.4. *Let $P \in \mathcal{S}^3(\mathbb{R}^{10})$ such that:*

$$(P)_{i_1,i_2,i_3} = \begin{cases} i_1^2 + 1 & \text{if } i_1 = i_2 = i_3 \\ 1 & \text{if } i_1 = i_2 \neq i_3 \\ 0 & \text{otherwise} \end{cases}$$

This sparse symmetric tensor corresponds to the polynomial $P = \sum_{i=1}^{n} i^2 x_i^3 + (\sum_{i=1}^{n} x_i^2) \times (\sum_{i=1}^{n} x_i)$.

EXAMPLE 4.5. *Let $P \in \mathcal{S}^3(\mathbb{C}^{10})$ such that:*

$$(P)_{i_1,i_2,i_3} = \begin{cases} e^{\sqrt{i_1}+i_1^2\sqrt{-1}} + \frac{i_1}{n}\sqrt{-1} & \text{if } i_1 = i_2 = i_3 \\ \frac{i_1}{n}\sqrt{-1} & \text{if } i_1 = i_2 \neq i_3 \\ 0 & \text{otherwise} \end{cases}$$

This sparse symmetric tensor corresponds to the polynomial $P = \sum_{i=1}^{n} e^{\sqrt{i}+i^2\sqrt{-1}} x_i^3 + \sqrt{-1}(\sum_{i=1}^{n} \frac{i}{n} x_i^2) \times (\sum_{i=1}^{n} x_i)$.

The numerical results in Table 7 show that the number of iterations of RNS-TR method is low compared to the other methods, as expected for an exact Newton method with a local quadratic convergence, though the first steps are governed by the Trust-Region criteria and can progress sub-quadratically. The cost of the iterations

TABLE 7
*Comparison of **RNS-TR**, **CPD-NLS**, **CCPD-NLS**, **SDF-NLS** for Examples 4.4 and 4.5.*

Example 4.4						Example 4.5					
r	err_{rns}			t_{rns}	N_{rns}	r	err_{rns}			t_{rns}	N_{rns}
	min	med	max	avg	avg		min	med	max	avg	avg
3	70.6	96	134.3	0.084	4	3	22.4	28.8	30.9	0.127	6
5	33.3	54.2	91.8	0.453	10	5	14.1	17.1	22.2	0.195	6
10	0.884	0.884	0.884	0.465	6	10	0.162	0.162	0.162	0.335	5
r	err_{cpd}			t_{cpd}	N_{cpd}	r	err_{cpd}			t_{cpd}	N_{cpd}
	min	med	max	avg	avg		min	med	max	avg	avg
3	71.5	102	137.2	0.042	13	3	23.1	28.1	34.5	0.087	14
5	34.5	55.2	302.9	0.041	14	5	15.5	18.7	27.6	0.086	17
10	10.7	24	80.1	0.095	27	10	6.4	8.6	18.1	0.205	32
r	err_{ccpd}			t_{ccpd}	N_{ccpd}	r	err_{ccpd}			t_{ccpd}	N_{ccpd}
	min	med	max	avg	avg		min	med	max	avg	avg
3	71	102	137.1	0.067	14	3	22.9	26.8	35.2	0.084	14
5	34.2	54.7	121	0.116	26	5	14.9	17	26.6	0.104	18
10	7.8	7.8	9.7	0.5	90	10	4.8	4.8	11.2	0.506	60
r	err_{sdf}			t_{sdf}	N_{sdf}	r	err_{sdf}			t_{sdf}	N_{sdf}
	min	med	max	avg	avg		min	med	max	avg	avg
3	71	96.3	136	0.155	14	3	22.9	27.4	35.2	0.254	15
5	34.2	49.4	105.3	0.212	16	5	14.9	17.8	26.5	0.35	19
10	7.8	8.2	38.3	2.3	158	10	4.8	6.2	12.6	2.5	144

in RNS-TR is more expensing, resulting in a total timing which is bigger than the fastest method CPD-NLS, though it remains competitive. The numerical quality of approximation of RNS-TR method is the best one in this test. It is of the same order than the other methods for $r = 3, 5$ but much better for $r = 10$. This can be explained by the fact that the initial point provided by SHD method is close to the best rank-10 approximation. In the other methods the initial point is chosen at random and over 50 trials, these methods have not found one initial point in the region of convergence to the best rank-10 approximation of P. Notice that when $r = 3, 5$ the initial point provided by SHD method, based on truncated SVD and eigenvector computations, yields the same behavior as a random initial point (a random linear combination of the matrices of a pencil is used to compute the eigenvectors in SHD method).

5. Conclusions. We presented the first Riemannian Newton optimization framework for approximating a given complex-valued symmetric tensor by a low rank symmetric tensor. We parametrized in subsection 3.1 the constraint set as the Cartesian product of Veronese manifolds. We have developed an exact Riemannian Newton iteration without approximating the Hessian matrix. We proposed in subsection 3.2 a suitable representation of the points in the constraint set, and we exploited in subsection 3.3 the properties of the apolar product and of partial complex derivatives, to deduce a simplified and explicit computation of the gradient and Hessian of the square distance function in terms of the points, weights of the decomposition and the tensor to approximate. In subsection 3.4, we presented a retraction operator on the Veronese manifold. We show that, combined with SHD method for choosing the initial point, the new Riemannian Newton method has a good practical behavior in several experiments, in subsection 4.1 to compute low rank approximations of random perturbations of low rank symmetric tensors, in subsection 4.2 to compute a best real rank-1 approximation of a real symmetric tensor, and in subsection 4.3 to compute a low rank approximation of sparse symmetric tensors. In future works, we plan to investigate the computation of initial points for the Riemannanian Newton iterations applied to tensors of higher rank and the low rank approximation problem for other families of tensors, such as multi-symmetric or skew symmetric tensors.

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, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*, Princeton University Press, Princeton, NJ, 2008.

[3] P.-A. Absil and J. Malick, *Projection-like retractions on matrix manifolds*, SIAM Journal on Optimization, 22 (2012), pp. 135–158.

[4] R. L. Adler, J. Dedieu, J. Y. Margulies, M. Martens, and M. Shub, *Newton's method on Riemannian manifolds and a geometric model for the human spine*, IMA Journal of Numerical Analysis, 22 (2002), pp. 359–390.

[5] J. Alexander and A. Hirschowitz, *Polynomial interpolation in several variables*, vol. 4, 1995, pp. 201–222.

[6] E. S. Allman, C. Matias, and J. A. Rhodes, *Identifiability of parameters in latent structure models with many observed variables*, Ann. Statist., 37 (2009), pp. 3099–3132.

[7] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, *Tensor decompositions for learning latent variable models*, Journal of Machine Learning Research, 15 (2014), pp. 2773–2832.

[8] A. Bjorck, *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics, 1996.

[9] P. Breiding and N. Vannieuwenhoven, *The condition number of join decompositions*, SIAM Journal on Matrix Analysis and Applications, 39 (2018), pp. 287–309.

[10] P. Breiding and N. Vannieuwenhoven, *A Riemannian trust region method for the canonical tensor rank approximation problem*, SIAM Journal on Optimization, 28 (2018), pp. 2435–2465.

[11] J. D. Carroll and J.-J. Chang, *Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition*, Psychometrika, 35 (1970), pp. 283–319.

[12] B. Chen, S. He, Z. Li, and S. Zhang, *Maximum block improvement and polynomial optimization*, SIAM Journal on Optimization, 22 (2012), pp. 87–107.

[13] J.-L. Chern and L. Dieci, *Smoothness and periodicity of some matrix decompositions*, SIAM J. Matrix Analysis Applications, 22 (2000), pp. 772–792.

[14] L. Chiantini, G. Ottaviani, and N. Vannieuwenhoven, *On generic identifiability of symmetric tensors of subgeneric rank*, Transactions of the American Mathematical Society, 369 (2016), p. 4021–4042.

[15] P. Comon, *Tensor decompositions*, in Mathematics in Signal Processing V, J.G. McWhirter and I.K. Proudler, Eds., (2002), pp. 1–24.

[16] P. Comon, G. Golub, L.-H. Lim, and B. Mourrain, *Symmetric tensors and symmetric tensor rank*, SIAM Journal on Matrix Analysis and Applications, 30 (2008), pp. 1254–1279.

[17] P. Comon and M. Rajih, *Blind identification of under-determined mixtures based on the characteristic function*, Signal Processing, 86 (2006), pp. 2271 – 2281. Special Section: Signal Processing in UWB Communications.

[18] L. De Lathauwer, B. De Moor, and J. Vandewalle, *A multilinear singular value decomposition*, SIAM Journal on Matrix Analysis and Applications, 21 (2000), pp. 1253–1278.

[19] L. De Lathauwer, B. De Moor, and J. Vandewalle, *On the best rank-1 and rank-(r1 ,r2 ,. . .,rn) approximation of higher-order tensors*, SIAM Journal on Matrix Analysis and Applications, 21 (2000), pp. 1324–1342.

[20] D. Eisenbud, *The Geometry of Syzygies: A Second Course in Commutative Algebra and Algebraic Geometry*, Springer. OCLC: 249751633.

[21] M. Espig, W. Hackbusch, and A. Khachatryan, *On the convergence of alternating least squares optimisation in tensor format representations*, arXiv preprint arXiv:1506.00062, (2015).

[22] L. D. Garcia, M. Stillman, and B. Sturmfels, *Algebraic geometry of bayesian networks*, Journal of Symbolic Computation, 39 (2005), pp. 331–355.

[23] W. Hackbusch, *Tensor Spaces and Numerical Tensor Calculus*, Springer Series in Computational Mathematics, Springer Berlin Heidelberg, 2012.

[24] J. Harmouch, H. Khalil, and B. Mourrain, *Structured low rank decomposition of multivariate Hankel matrices*, Linear Algebra and Applications, (2017).

[25] R. Harshman, *Foundations of the parafac procedure: Models and conditions for an "explanatory" multi-modal factor analysis*, UCLA Working Papers in Phonetics, 16 (1970), pp. 1–84.

[26] C. Hayashi and F. Hayashi, *A new algorithm to solve parafac-model*, Behaviormetrika, 9 (1982), pp. 49–60.

[27] F. L. Hitchcock, *The expression of a tensor or a polyadic as a sum of products*, Journal of Mathematics and Physics, 6 (1927), pp. 164–189.
[28] J.Harris, *Algebraic Geometry: A First Course*, Graduate Texts in Mathematics, Springer-Verlag, New York, NY, 1998.
[29] J. Kileel and J. M. Pereira, *Subspace power method for symmetric tensor decomposition and generalized PCA*, https://arxiv.org/abs/1912.04007.
[30] E. Kofidis and P. A. Regalia, *On the best rank-1 approximation of higher-order supersymmetric tensors*, SIAM Journal on Matrix Analysis and Applications, 23 (2002), pp. 863–884.
[31] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Review, 51 (2009), pp. 455–500.
[32] K. Konstantinides and K. Yao, *Statistical analysis of effective singular values in matrix rank determination*, IEEE Transactions on Acoustics, Speech, and Signal Processing, 36 (1988), pp. 757–763.
[33] D. Kressner, M. Steinlechner, and B. Vandereycken, *Low-rank tensor completion by Riemannian optimization*, BIT Numer. Math., 54 (2014), pp. 447–468.
[34] J. Landsberg, *Tensors: Geometry and Applications*, Graduate studies in mathematics.
[35] J. M. Landsberg, *Tensors: Geometry and Applications: Geometry and Applications*, American Mathematical Soc.
[36] L. D. Lathauwer, P. Comon, B. D. Moor, and J. Vandewalle, *Higher-order power method - application in independent component analysis*, 1995.
[37] B. Mourrain, *Polynomial-Exponential Decomposition from Moments*, Foundations of Computational Mathematics, 18 (2018), pp. 1435–1492.
[38] Z. Nehari, *Introduction to Complex Analysis*, Allyn & Bacon, 1968.
[39] J. Nie, *Low Rank Symmetric Tensor Approximations*, SIAM Journal on Matrix Analysis and Applications, 38, pp. 1517–1540.
[40] J. Nie and L. Wang, *Semidefinite relaxations for best rank-1 tensor approximations*, SIAM Journal on Matrix Analysis and Applications, 35 (2014), pp. 1155–1179.
[41] J. Nocedal and S. Wright, *Numerical optimization*, Springer series in operations research and financial engineering, Springer, New York, NY, 2. ed., 2006.
[42] P. Paatero, *The multilinear engine—a table-driven, least squares program for solving multilinear problems, including the n-way parallel factor analysis model*, Journal of Computational and Graphical Statistics, 8 (1999), pp. 854–888.
[43] A.-H. Phan, P. Tichavský, and A. Cichocki, *Low complexity damped gauss–newton algorithms for candecomp/parafac*, SIAM Journal on Matrix Analysis and Applications, 34 (2013), pp. 126–147.
[44] R. Remmert and R. Burckel, *Theory of Complex Functions*, Graduate Texts in Mathematics, Springer New York, 1991.
[45] B. Savas and L.-H. Lim, *Quasi-newton methods on grassmannians and multilinear approximations of tensors*, SIAM Journal on Scientific Computing, 32 (2010), pp. 3352–3393.
[46] A. Smilde, R. Bro, and P. Geladi, *Multi-way Analysis with Applications in the Chemical Sciences*, John Wiley, West Sussex, UK, 2004.
[47] L. Sorber, M. V. Barel, and L. D. Lathauwer, *Unconstrained optimization of real functions in complex variables*, SIAM Journal on Optimization, 22 (2012), pp. 879–898.
[48] L. Sorber, M. Van Barel, and L. De Lathauwer, *Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank $-(l_r, l_r, 1)$ terms, and a new generalization*, SIAM Journal on Optimization, 23 (2013), pp. 695–720.
[49] G. W. Stewart, *Rank degeneracy*, SIAM Journal on Scientific and Statistical Computing, 5 (1984), pp. 403–413.
[50] G. Tomasi and R. Bro, *A comparison of algorithms for fitting the parafac model*, Comput. Stat. Data Anal., 50 (2006), pp. 1700–1734.
[51] A. Uschmajew, *Local convergence of the alternating least squares algorithm for canonical tensor approximation*, SIAM Journal on Matrix Analysis and Applications, 33 (2012), pp. 639–652.
[52] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen, *A New Truncation Strategy for the Higher-Order Singular Value Decomposition*, SIAM Journal on Scientific Computing, 34, pp. A1027–A1052.
[53] N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer, *Tensorlab 3.0*, Mar. 2016, https://www.tensorlab.net.
[54] F. Zak, *Tangents and Secants of Algebraic Varieties*, Translations of Mathematical Monographs, AMS, Providence, RI, 1993.
[55] X. Zhang, C. Ling, and L. Qi, *The best rank-1 approximation of a symmetric tensor and related spherical optimization problems*, SIAM Journal on Matrix Analysis and Applications,

33 (2012), pp. 806–821.