SVD-BASED ALGORITHMS FOR THE BEST RANK-1 APPROXIMATION OF A SYMMETRIC TENSOR

YU GUAN*, MOODY T. CHU†, AND DELIN CHU‡

Abstract. This paper revisits the problem of finding the best rank-1 approximation to a symmetric tensor and makes three contributions. First, in contrast to the many long and lingering arguments in the literature, it offers a straightforward justification that generically the best rank-1 approximation to a symmetric tensor is symmetric. Second, in contrast to the typical workhorse in the practice for the low-rank tensor approximation, namely, the alternating least squares (ALS) technique which improves one factor a time, this paper proposes three alternative algorithms, based on the singular value decomposition (SVD) that modifies two factors a time. One step of SVD-based iteration is superior to two steps of ALS iterations. Third, it is proved that not only the generalized Rayleigh quotients generated from the three SVD-based algorithms enjoy monotone convergence, but also that the iterates themselves converge.

Key words. symmetric tensor, best rank-1 approximation, singular value decomposition, convergence analysis

AMS subject classifications. 15A15, 15A09, 15A23

1. Introduction.

A real-valued tensor of order k can be represented by a k-way array

$$T = [\tau_{i_1,\ldots,i_k}] \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_k}$$

with elements $\tau_{i_1,\ldots,i_k}$ accessed via k indices. A tensor of the form

$$\bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)} = \mathbf{u}^{(1)} \otimes \ldots \otimes \mathbf{u}^{(k)} := \left[u_{i_1}^{(1)} \ldots u_{i_k}^{(k)} \right],$$

where elements are the products of entries from vectors $\mathbf{u}^{(\ell)} \in \mathbb{R}^{I_\ell}$, $\ell = 1, \ldots, k$, is said to be of rank one. When $I_1 = \ldots = I_k$, we have a square tensor. An order-k square tensor T is said to be symmetric if

$$\tau_{i_1,\ldots,i_k} = \tau_{i_{\sigma(1)},\ldots,i_{\sigma(k)}} \tag{1.1}$$

with respect to all possible permutations σ over the integers $\{1, \ldots k\}$. A symmetric rank-1 tensor therefore necessarily implies that $\mathbf{u}^{(\ell)} = c_\ell \mathbf{u}^{(1)}$ for some scalar c_ℓ, $\ell = 2, \ldots, k$. In this case, we denote $I_1 = \ldots = I_k = n$ and write $\bigotimes_{\ell=1}^{k} \mathbf{u} = \mathbf{u}^k$.

The problem of finding a best rank-1 approximation to T is to determine unit vectors $\mathbf{u}^{(\ell)} \in \mathbb{R}^{I_\ell}$, $\ell = 1, \ldots k$, and a scalar λ such that the functional

$$f\left(\lambda, \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}\right) := \left\| T - \lambda \bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)} \right\|_F^2 = \sum_{i_1, i_2, \ldots, i_k} \left(t_{i_1,\ldots,i_k} - \lambda u_{i_1}^{(1)} \ldots u_{i_k}^{(k)} \right)^2 \tag{1.2}$$

is minimized. For any fixed unit vectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}$, the optimal value of λ for (1.2) is given precisely by the length of the projection of the "vector" T onto the direction of the "unit vector" $\bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_k}$, i.e.,

$$\lambda = \lambda\left(\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}\right) = \left\langle T, \bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)} \right\rangle. \tag{1.3}$$

*Department of Mathematics, National University of Singapore, Singapore 119076. (ricky7guanyu@gmail.com)

†Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205. (chu@math.ncsu.edu) This research was supported in part by the National Science Foundation under grant DMS-1316779.

‡Department of Mathematics, National University of Singapore, Singapore 119076. (matchudl@nus.edu.sg) This research was supported in part by NUS research grant R-146-000-236-114.

Thus, minimizing the orthogonal component of T, as is desired in (1.2), is equivalent to maximizing the length $|\lambda|$ of the parallel component. In [20], the expression (1.3) is called the generalized Rayleigh quotient of T corresponding to $\{\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}\}$. Many approaches for finding the extreme values of (1.3) have been proposed in the literature. See, for example, [3, 8, 9, 11, 12, 13]. Switching the signs of the variables $\mathbf{u}^{(\ell)}$, if necessary, we may restrict our attention without loss of generality to the case that $\lambda > 0$ only.

This paper is about finding the best rank-1 approximation to a symmetric tensor. It was conjectured in [16] and proved that the best symmetric rank-1 approximation to a symmetric tensor is its best rank-1 approximation [21, Theorem 2.1]. The proof was by induction. However, a more correct way of stating this result is that the best rank-1 approximation to a symmetric tensor "can be chosen" symmetric [7, Theorem 9], because there might be non-symmetric best rank-1 approximations [7, Section 4] for a symmetric tensor. Even more precisely, except for symmetric tensors lying on a specific real algebraic variety, a generic symmetric tensor has a unique rank-1 approximation which, hence, is symmetric [7]. With all these being said, we make an interesting remark that perhaps it was Stefan Banach who first noted in the context of homogeneous polynomials [1] that a best rank-1 approximation of a symmetric tensor could be chosen to be symmetric. Discussions on different aspects of rank-1 approximation to symmetric tensors can be found in [2, 4, 10, 11, 15]. Research endeavor on this subject is still ongoing. See, for example, some more recent work in [5, 18].

This paper contains two parts. First, we offer a simple argument that the symmetry of the best rank-1 approximation for a generic symmetric tensor can easily be understood with the notion of conventional singular value decomposition (SVD) for matrices. Second, we turn that argument into iterative SVD-based algorithms for computing the symmetric best rank-1 approximation. Our main focus in this paper is on the second part where we offer a convergence analysis that is new in the literature.

This paper is organized as follows. We begin with the introduction of some notations and basic facts in Section 2. Then, using well known properties of the SVD, we argue in a very concise way for the symmetry of the best rank-1 approximation in Section 3. Depending on how the permutations are chosen when applying the SVD successively to increase the objective value, we propose three algorithms for computing the best rank-1 approximation in Section 4. Among these, Algorithms 1 with cyclic permutation and Algorithm 2 with pre-assigned random permutation are formulated due to their theoretical simplicity. In turn, they highly motivate Algorithm 3 with post-assigned random permutation which is the easiest to implement. Convergence analysis is given in Section 5 where we first prove that the generalized Rayleigh quotients of all three algorithms converge monotonically and then detail the limiting behavior of the iterates generated by Algorithm 3. Since Algorithm 1 is of theoretical interest but is not implemented, its convergence analysis is given in the Appendix. Some numerical examples together with some interesting observations are presented in Section 6.

2. Basics. Tensors are multi-dimensional arrays. Thus, there are multiple ways to define tensor multiplications. Their appearances are often rather complex and perplexing. To facilitate the subsequence discussion, we first introduce a simple notation system that generalizes what we already know from the matrix theory. We also establish a few useful tools. Readers who are familiar with these basic operations may skip to the next section.

Let the symbol $[\![m]\!]$ denote henceforth the set of integers $\{1, \ldots m\}$ for a given positive integer m. Suppose that the set $[\![k]\!]$ is partitioned as the union of two disjoint nonempty subsets $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_s\}$ and $\boldsymbol{\beta} = \{\beta_1, \ldots, \beta_t\}$, where $s + t = k$. An element in the tensor $T \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_k}$ will be marked as $\tau^{(\boldsymbol{\alpha},\boldsymbol{\beta})}_{[\mathcal{I}|\mathcal{J}]}$ where $\mathcal{I} := (i_1, \ldots, i_s)$ and $\mathcal{J} := (j_1, \ldots, j_t)$ contain those indices at locations $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, respectively. Each index in the arrays $\mathcal{I}$ and $\mathcal{J}$ should be within the corresponding range of integers, e.g., $i_1 \in [\![I_{\alpha_1}]\!]$ and so on. If the reference to a specific partitioning $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is clear, then without causing ambiguity we abbreviate the element as $\tau_{[\mathcal{I}|\mathcal{J}]}$. For example, if $\boldsymbol{\alpha} = \{2, 4\}$ and $\boldsymbol{\beta} = \{1, 3, 5, 6\}$, then $\tau_{2,3,1,5,6,4} = \tau_{[3,5|2,1,6,4]}$. Note that the representation $\tau_{[\mathcal{I}|\mathcal{J}]}$ is $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ specific, but there is no preference of $\boldsymbol{\alpha}$ over $\boldsymbol{\beta}$. The partition $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ may be regarded as generalizing the notion of row and column which we are familiar with in matrices.

Given a fixed partitioning $[\![k]\!] = \boldsymbol{\alpha} \cup \boldsymbol{\beta}$, we shall regard an order-k tensor $T \in \mathbb{R}^{I_1 \times \ldots \times I_k}$ as a "matrix representation" of a linear operator mapping order-s tensors to order-t tensors [19]. Specifically, we identify T

with the linear map

$$\mathscr{T}_{\boldsymbol{\beta}} : \mathbb{R}^{I_{\alpha_1}\times\ldots\times I_{\alpha_s}} \to \mathbb{R}^{I_{\beta_1}\times\ldots\times I_{\beta_t}}, \tag{2.1}$$

such that for any $S \in \mathbb{R}^{I_{\alpha_1}\times\ldots\times I_{\alpha_s}}$, we have

$$\mathscr{T}_{\boldsymbol{\beta}}(S) := T \circledast_{\boldsymbol{\beta}} S = \left[\left\langle \tau_{[:|\ell_1,\ldots,\ell_t]}, S\right\rangle\right] \in \mathbb{R}^{I_{\beta_1}\times\ldots\times I_{\beta_t}}, \tag{2.2}$$

where

$$\left\langle \tau_{[:|\ell_1,\ldots,\ell_t]}, S\right\rangle := \sum_{i_1=1}^{I_{\alpha_1}} \cdots \sum_{i_s=1}^{I_{\alpha_s}} \tau_{[i_1,\ldots,i_s|\ell_1,\ldots,\ell_t]} s_{i_1,\ldots,i_s} \tag{2.3}$$

is the Frobenius inner product generalized to multi-dimensional arrays. In other words, the product $\circledast_{\boldsymbol{\beta}}$ generalizes the conventional matrix-to-vector multiplication by combining the $(\ell_1,\ldots,\ell_t)$-th "slice" in the $\boldsymbol{\beta}$ direction of the tensor T, i.e., $\tau_{[:|\ell_1,\ldots,\ell_t]}$, with S through the inner product (2.3).

Each of the following sequence of results is elementary but together they are helpful tool for algebraic manipulations throughout the discussion.

LEMMA 2.1. *Given a general tensor $T \in \mathbb{R}^{I_1\times I_2\times\ldots\times I_k}$, a partitioning $[\![k]\!] = \boldsymbol{\alpha}\cup\boldsymbol{\beta}$, and vectors $\mathbf{u}^{(\ell)} \in \mathbb{R}^{I_\ell}$, $\ell = 1,\ldots,k$, then it holds that*

$$\left\langle T, \bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)}\right\rangle = \left\langle T\circledast_{\boldsymbol{\beta}} \bigotimes_{i=1}^{s} \mathbf{u}^{(\alpha_i)}, \bigotimes_{j=1}^{t} \mathbf{u}^{(\beta_j)}\right\rangle. \tag{2.4}$$

Proof. Based on the definition (2.2), the right hand side of (2.4) is simply a rearrangement of terms in the summation by the associative law. ∎

LEMMA 2.2. *Given a general tensor $T \in \mathbb{R}^{I_1\times I_2\times\ldots\times I_k}$, arbitrary vectors $\mathbf{u}^{(\alpha_i)} \in \mathbb{R}^{I_{\alpha_i}}$, $i = 1,\ldots,k-2$, and $\mathbf{v} \in \mathbb{R}^{I_{\beta_2}}$, then*

$$\left(T\circledast_{\{\beta_1,\beta_2\}} \bigotimes_{i=1}^{k-2} \mathbf{u}^{(\alpha_i)}\right)\mathbf{v} = \left(T\circledast_{\{\beta_1,\alpha_j\}} \bigotimes_{i=1}^{j-1} \mathbf{u}^{(\alpha_i)} \otimes \mathbf{v} \otimes \bigotimes_{i=j+1}^{k-2} \mathbf{u}^{(\alpha_i)}\right)\mathbf{u}^{(\alpha_j)} \tag{2.5}$$

for any $j \in [\![k-2]\!]$.

Proof. The notion of a tensor entry $\tau^{(\boldsymbol{\alpha},\boldsymbol{\beta})}_{[\mathcal{I}|\mathcal{J}]}$ defined earlier will be informative here for tracking which index is being associated with which location. The μ-th entry of the vector $\left(T\circledast_{\boldsymbol{\beta}} \bigotimes_{i=1}^{k-2} \mathbf{u}^{(\alpha_i)}\right)\mathbf{v}$ is given by

$$\begin{aligned}
&\sum_{\nu=1}^{I_{\beta_2}} \left(\sum_{i_1=1}^{I_{\alpha_1}} \cdots \sum_{i_{k-2}=1}^{I_{\alpha_{k-2}}} \tau^{(\boldsymbol{\alpha},\boldsymbol{\beta})}_{[i_1,\ldots,i_{k-2}|\mu,\nu]} u^{(\alpha_1)}_{i_1}\cdots u^{(\alpha_{k-2})}_{i_{k-2}}\right) v_\nu \\
&= \sum_{i_1=1}^{I_{\alpha_1}} \cdots \sum_{i_{k-2}=1}^{I_{\alpha_{k-2}}} \sum_{\nu=1}^{I_{\beta_2}} \tau^{(\boldsymbol{\alpha}\cup\{\beta_2\},\{\beta_1\})}_{[i_1,\ldots,i_{k-2},\nu|\mu]} u^{(\alpha_1)}_{i_1}\cdots u^{(\alpha_{k-2})}_{i_{k-2}} v_\nu \\
&= \sum_{i_j=1}^{I_{\alpha_j}} \left(\sum_{i_1=1}^{I_{\alpha_1}} \cdots \sum_{i_{k-2}=1}^{I_{\alpha_{k-2}}} \sum_{\nu=1}^{I_{\beta_2}} \tau^{(\boldsymbol{\alpha}\cup\{\beta_2\}-\{\alpha_j\},\{\beta_1,\alpha_j\})}_{[i_1,\ldots,i_{k-2},\nu|\mu,i_j]} u^{(\alpha_1)}_{i_1}\cdots u^{(\alpha_{k-2})}_{i_{k-2}} v_\nu\right) u^{(\alpha_j)}_{i_j},
\end{aligned}$$

where the last equality is obtained by the associative law so that the summation inside the parentheses contains no $u^{(\alpha_j)}_{i_j}$ terms. ∎

LEMMA 2.3. *Given a symmetric tensor $T \in \mathbb{R}^{n\times\cdots\times n}$ of order k and two fixed positive integers s and t with $k = s + t$, let $\mathbf{v}^{(\ell)} \in \mathbb{R}^n$, $\ell = 1, \ldots, s$, be arbitrary vectors. Then the product $T\circledast_{\boldsymbol{\beta}} \bigotimes_{i=1}^{s} \mathbf{v}^{(\rho_i)}$ is a symmetric tensor of order t and is independent of any permutation ρ of $[\![s]\!]$ and any subset $\boldsymbol{\beta} \subset [\![k]\!]$ with cardinality t.*

Proof. Let $[\![k]\!] = \boldsymbol{\alpha} \cup \boldsymbol{\beta}$ be a partitioning where, without loss of generality, indices in $\boldsymbol{\beta}$ are arranged in ascending order. Then for $\ell_j \in [\![n]\!]$, $j = 1, \ldots, t$, we have

$$\begin{aligned}\left(T\circledast_{\boldsymbol{\beta}} \bigotimes_{i=1}^{s} \mathbf{v}^{(\rho_i)}\right)_{\ell_1,\ldots,\ell_t} &= \sum_{i_1=1}^{n} \cdots \sum_{i_s=1}^{n} \tau^{(\boldsymbol{\alpha},\boldsymbol{\beta})}_{[i_1,\ldots,i_s|\ell_1,\ldots,\ell_t]} v_{i_1}^{(\rho_1)} \cdots v_{i_s}^{(\rho_s)} \\ &= \sum_{i_1=1}^{n} \cdots \sum_{i_s=1}^{n} \tau_{i_1,\ldots,i_s,\ell_1,\ldots,\ell_t} v_{i_1}^{(\rho_1)} \cdots v_{i_s}^{(\rho_s)} \\ &= \sum_{i_{\rho^{-1}(1)}=1}^{n} \cdots \sum_{i_{\rho^{-1}(s)}=1}^{n} \tau_{i_{\rho^{-1}(1)},\ldots,i_{\rho^{-1}(s)},\ell_1,\ldots,\ell_t} v_{i_{\rho^{-1}(1)}}^{(1)} \cdots v_{i_{\rho^{-1}(s)}}^{(s)}.\end{aligned}$$

In the above, the symmetry of T implies that the location of $\boldsymbol{\beta}$ is immaterial and thus the second equation is obtained by moving $\ell_1, \ldots, \ell_t$ to the end of the index array, whereas ρ^{-1} denotes the inverse of the permutation ρ. By renaming $i_{\rho^{-1}(j)}$ as i_j, $j = 1, \ldots, s$, we see that the reference to ρ is also immaterial. ∎

COROLLARY 2.4. *Under the same condition of Lemma 2.3, the associative law*

$$T\circledast \bigotimes_{i=1}^{s} \mathbf{v}^{(\rho_i)} = \left(T\circledast \bigotimes_{i=1}^{j} \mathbf{v}^{(\rho_i)}\right)\circledast \bigotimes_{i=j+1}^{s} \mathbf{v}^{(\rho_i)} \tag{2.6}$$

holds for any $j \in [\![s]\!]$ and any permutation ρ of $[\![s]\!]$.

When the subset $\boldsymbol{\beta} \subset [\![k]\!]$ is of cardinality 2, then Lemma 2.3 can be generalized to arbitrary tensor of order $k - 2$.

LEMMA 2.5. *Given a symmetric tensor $T \in \mathbb{R}^{n\times\cdots\times n}$ of order k is symmetric and a subset $\boldsymbol{\beta} \subset [\![k]\!]$ with cardinality 2, then, with respect to any tensor $S \in \mathbb{R}^{n\times\cdots\times n}$ of order $k - 2$, the product $T\circledast_{\boldsymbol{\beta}} S$ is a symmetric matrix.*

Proof. For convenience, write $M := T\circledast_{\boldsymbol{\beta}} S$. By the definition (2.2), M is a matrix. Observer that

$$m_{ij} = \langle \tau_{[:|i,j]}, S\rangle = \langle \tau_{[:|j,i]}, S\rangle = m_{ji},$$

because $\tau_{[:|i,j]} = \tau_{[:|j,i]}$ by the symmetry of T. ∎

For latter usage in our proof for convergence, we also need the following results from real analysis.

LEMMA 2.6. *Let $\{a_k\}$ be a bounded sequence of real numbers with the property $|a_{k+1} - a_k| \to 0$ as $k \to \infty$. If the accumulation points for the sequence are isolated, then $\{a_k\}$ converges to a unique limit point.*

Proof. Suppose $\{a_{\alpha_k}\}$ and $\{a_{\beta_k}\}$ are two subsequences of $\{a_k\}$ which converge, respectively, to two distinct limit points, x and y. Let z denote any fixed real number between x and y. For a positive number r, let $B_x(r)$ denote the neighborhood $[x - r, x + r]$ of x.

For any $0 < \epsilon < \frac{1}{4}\min\{|x - z|, |y - z|\}$, there exists a large enough integer $K = K(\epsilon)$ such that $a_{\alpha_k} \in B_x(\epsilon)$, $a_{\beta_k} \in B_y(\epsilon)$, and $|a_{k+1} - a_k| < \epsilon$ for all $k \geq K$. Infinitely many elements of $\{a_k\}$ must leave $B_x(\epsilon)$ to enter $B_y(\epsilon)$ and vise versa. By doing so, there is an infinite subsequence of $\{a_k\}$ contained in $B_z(\epsilon)$. This shows that z is also an accumulation point. Since z is arbitrary, we have shown any number between x and y is an accumulate point. This contradicts the assumption that the accumulation points are isolated. ∎

Lemma 2.6 asserts the uniqueness. A slight variation requiring a weaken assumption and resulting only a local convergence will also fit our need.

LEMMA 2.7. *Assume that a^* is an isolated accumulation point of a sequence $\{a_k\}$ such that for every subsequence $\{a_{k_j}\}$ converging to a^*, there is an infinite subsequence $\{a_{k_{j_i}}\}$ such that $|a_{k_{j_i}+1} - a_{k_{j_i}}| \to 0$. Then the whole sequence $\{a_k\}$ converges to a^*.*

Proof. There are some ambiguities in the original proof [14, Lemma 4.10]. See also [21, Proposition 3.2]. We take this opportunity to clarify the dubiety. Suppose that the sequence $\{a_k\}$ does not converge to a^*. We prove by contradiction.

Since a^* is isolated, there exists a neighborhood $N_\epsilon(a^*) := \{x \in \mathbb{R} | |x - a^*| \le \epsilon\}$ such that a^* is the only accumulation point of the sequence $\{a_k\}$. Let $\{a_{k_j}\}$ be an arbitrary subsequence contained in $N_\epsilon(a^*)$. For each j, let $\{a_{k_j}, a_{k_j+1}, \ldots, a_{\ell_j}\}$ be the largest consecutive segment of $\{a_k\}$ that starts at a_{k_j} and stays inside $N_\epsilon(a^*)$, i.e.,

$$\ell_j := \max\{\ell | \, |a_i - a^*| \le \epsilon, i = k_j, k_j + 1, \ldots, \ell\}.$$

Reducing ϵ if necessary, note that ℓ_j must be finite because, otherwise, $|a_k - a^*| \le \epsilon$ for all k large enough and for arbitrary ϵ, implying that $\{a_k\}$ converges to a^*.

By construction, the subsequence $\{a_{\ell_j}\}$ has the property

$$|a_{\ell_j} - a^*| \le \epsilon, \quad |a_{\ell_j+1} - a^*| > \epsilon.$$

Being bounded, a subsequence $\{a_{\ell_{j_i}}\}$ of $\{a_{\ell_j}\}$ must converge. Being contained in $N_\epsilon(a^*)$, the limit point must be a^*. Therefore $|a_{\ell_{j_i}} - a^*| < \frac{\epsilon}{2}$ when i is large enough. In this way, we have found a convergent subsequence $\{a_{\ell_{j_i}}\}$, but element by element we always have the gap

$$|a_{\ell_{j_i}+1} - a_{\ell_{j_i}}| \ge |a_{\ell_{j_i}+1} - a^*| - |a_{\ell_{j_i}} - a^*| \ge \frac{\epsilon}{2}.$$

This is a contradiction. ∎

3. Symmetric Best rank-1 approximation. We now argue that the best rank-1 approximation to a generic symmetric tensor is symmetric. We shall not assume a priori that the best rank-1 approximation is unique, nor that a symmetric best rank-1 approximation always exists. All we need is the following fundamental fact from matrix theory.

LEMMA 3.1. *Given a matrix $A \in \mathbb{R}^{m \times n}$, then the global maximum of the generalized Rayleigh quotient*

$$\max_{\substack{\mathbf{y} \in \mathbb{R}^m, \|\mathbf{y}\| = 1 \\ \mathbf{z} \in \mathbb{R}^n, \|\mathbf{z}\| = 1}} \mathbf{y}^\top A \mathbf{z} \tag{3.1}$$

is precisely the largest singular value σ_1 of A, where the global maximizer $(\mathbf{y}_1, \mathbf{z}_1)$ consists of precisely the corresponding left and right singular vectors. The best rank-1 approximation to A is given by $\sigma_1 \mathbf{y}_1 \mathbf{z}_1^\top$. In the event that $A \in \mathbb{R}^{m \times m}$ is symmetric and that the largest singular value of A is simple, then $\mathbf{y} = \pm \mathbf{z}$ depending on the sign[1] of the dominant eigenvalue $\lambda_1 = \pm \sigma_1$ and, hence, the best rank-1 approximation to A is symmetric.

The condition that the largest singular value of A is simple is generic in the sense that the symmetric matrices with multiply eigenvalues form an algebraic variety of codimension two [6]. Consequently, the symmetric matrices that do not have a unique symmetric best rank-1 approximation form an algebraic variety of codimension one [7, Lemma 3].

Built upon Lemma 3.1, we explain in the argument below the kind of generic property we need for a symmetric tensor. We justify the symmetry by comparing two components a time in the rank-1 tensor.

Suppose that $\lambda \bigotimes_{\ell=1}^k \overline{\mathbf{u}}^{(\ell)}$ is the best rank-1 approximation to a given order-k symmetric tensor T. By (1.3), the generalized Rayleigh quotient $\lambda = \left\langle T, \bigotimes_{\ell=1}^k \overline{\mathbf{u}}^{(\ell)} \right\rangle$ is positive and maximal. Consider the case

[1] We shall use the symbol $\pm$ to indicate a proper sign selection in the subsequent discussion when there is no need to specify the sign.

Algorithm 4.1 (Best rank-1 approximation via SVD updating with cyclic progression.)

Require: An order-k, n-dimensional, symmetric tensor T and k starting unit vectors $\mathbf{u}_{[0]}^{(1)}, \ldots, \mathbf{u}_{[0]}^{(k)} \in \mathbb{R}^n$
Ensure: A local best rank-1 approximation to T

1: **for** $p = 0, 1, \cdots,$ **do**
2: **for** $\ell = 1, 2, \cdots, k-1,$ **do**
3: $\boldsymbol{\beta}_\ell = (\ell, \ell+1)$
4: $C_{[p]}^{(\ell)} = T \circledast_{\boldsymbol{\beta}_\ell} \bigotimes_{i=1}^{\ell-1} \mathbf{u}_{[p+1]}^{(i)} \otimes \bigotimes_{i=\ell+2}^{k} \mathbf{u}_{[p]}^{(i)}$
5: $[\mathbf{u}, s, \mathbf{v}] = \mathsf{svds}(C_{[p]}^{(\ell)}, 1)$ {Dominant singular value triplet via Matlab routine svds}
6: **if** $u_1 < 0$ **then**
7: $\mathbf{u} = -\mathbf{u}$ {Assume the generic case that $u_1 \neq 0$; otherwise, use another entry.}
8: **end if**
9: $\mathbf{u}_{[p+1]}^{(\ell)} := \mathbf{u}$ {If $\ell = 1$, this is $\widehat{\mathbf{u}}_{[p+1]}^{(1)}$; otherwise this is the second update $\mathbf{u}_{[p+1]}^{(\ell)}$, if $2 \leq \ell < k$.}
10: $\widehat{\mathbf{u}}_{[p+1]}^{(\ell+1)} := \mathbf{u}$ {Skipping this step will not affect $C_{[p]}^{(\ell+1)}$ at Line 4.}
11: $\lambda_{[p+1]}^{(\ell)} := s$
12: **end for**
13: $\boldsymbol{\beta}_k = (k, 1)$
14: $C_{[p]}^{(k)} = T \circledast_{\boldsymbol{\beta}_k} \bigotimes_{i=2}^{k-1} \mathbf{u}_{[p+1]}^{(i)}$
15: $[\mathbf{u}, s, \mathbf{v}] = \mathsf{svds}(C_{[p]}^{(k)}, 1)$ {Dominant singular value triplet via Matlab routine svds}
16: $\mathbf{u}_{[p+1]}^{(k)} := \mathbf{u}$ {Adjust the sign properly as in Line 6.}
17: $\mathbf{u}_{[p+1]}^{(1)} := \mathbf{u}$
18: $\lambda_{[p+1]}^{(k)} := s$
19: **end for**

$\boldsymbol{\beta} = \{1, 2\}$. By Lemma 2.1, we can write

$$\lambda = \left\langle T \circledast_{\boldsymbol{\beta}} \bigotimes_{\ell=3}^{k} \overline{\mathbf{u}}^{(\ell)}, \overline{\mathbf{u}}^{(1)} \otimes \overline{\mathbf{u}}^{(2)} \right\rangle.$$

By Corollary 2.5, the matrix $C := T \circledast_{\boldsymbol{\beta}} \bigotimes_{\ell=3}^{k} \overline{\mathbf{u}}^{(\ell)}$ is symmetric. Assume that the largest singular value, which is λ, of C is simple. Then, by Lemma 3.1, we conclude that $\overline{\mathbf{u}}^{(1)} = \pm \overline{\mathbf{u}}^{(2)}$. Moving to the choice $\boldsymbol{\beta} = \{2, 3\}$ and assuming again that λ is simple for the newly defined matrix C, we than have $\overline{\mathbf{u}}^{(2)} = \pm \overline{\mathbf{u}}^{(3)}$. Continuing this process, we conclude that $\overline{\mathbf{u}}^{(1)}, \ldots, \overline{\mathbf{u}}^{(k)}$ differ from each other by at most a negative sign. At the end, we may write $\lambda \bigotimes_{\ell=1}^{k} \overline{\mathbf{u}}^{(\ell)} = \pm \lambda \overline{\mathbf{u}}^{(1)^k}$. So the best rank-1 approximation to a symmetric tensor is necessarily symmetric.

4. Computation. The argument in the preceding section motivates an SVD-based way to calculate the symmetric best rank-1 approximation by iterations. The idea of using the SVD instead of the ALS is not new. It has been proposed for general tensors in [8], but so far as we know no convergence analysis has ever been established. The main contribution of this paper is to furnish the proof of convergence for symmetric tensors.

4.1. Update with cyclic progression. The most basic approach is outlined in Algorithm 4.1. For efficiency, we also propose a modification by random permutations in Algorithm 4.2 followed by a more simplified Algorithm 4.3. Two types of dynamics are involved in all algorithms. One is the dynamics of the objective values, of which the analysis is straightforward. The other is the dynamics of the iterates, which is much harder to characterize. We will discuss the convergence in the next section.

Algorithm 4.2 (Best rank-1 approximation via SVD updating with randomization.)

Require: An order-k, n-dimensional, symmetric tensor T and k starting unit vectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)} \in \mathbb{R}^n$
Ensure: A local best rank-1 approximation to T

1: $t \leftarrow 0$
2: $\lambda_0 \leftarrow \left\langle T, \bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)} \right\rangle$
3: **repeat**
4: $\quad t \leftarrow t + 1$
5: $\quad \sigma \leftarrow$ random permutation of $\{1, \ldots, k\}$
6: $\quad \boldsymbol{\beta}_t \leftarrow (\sigma_{k-1}, \sigma_k)$ {Randomly select two factors}
7: $\quad C_t \leftarrow T \circledast_{\boldsymbol{\beta}_t} \bigotimes_{i=1}^{k-2} \mathbf{u}^{(\sigma_i)}$
8: $\quad [\mathbf{u}_t, s_t, \mathbf{v}_t] = \mathsf{svds}(C_t, 1)$ {Dominant singular value triplet via Matlab routine svds}
9: $\quad$ **if** $(\mathbf{u}_t)_1 < 0$ **then**
10: $\quad\quad \mathbf{u}_t = -\mathbf{u}_t$
11: $\quad$ **end if**
12: $\quad \lambda_t \leftarrow s_t$
13: $\quad \mathbf{u}^{(\sigma_{k-1})}, \mathbf{u}^{(\sigma_k)} \leftarrow \mathbf{u}_t$
14: **until** λ_t meets convergence criteria

To convey the idea, it is convenient to adopt the subscript $_{[p]}$ in Algorithm 4.1 to indicate the quantity at the p-th iteration. Each sweep of p at Line 1 in Algorithm 4.1 involves k pairs of $\boldsymbol{\beta}$ ranging circularly from $(1,2), (2,3), \ldots, (k,1)$. Thus, each $\mathbf{u}_{[p+1]}^{(\ell)}$ is updated twice. The first updates for $\ell = 2, \ldots, k$, denoted by $\widehat{\mathbf{u}}_{[p+1]}^{(\ell)}$ at Line 10, are not essential and can be completely removed from the algorithm without affecting the calculation, but their presences help bridge the monotonicity in theory. The update $\widehat{\mathbf{u}}_{[p+1]}^{(1)}$ is temporarily overwritten as $\mathbf{u}_{[p+1]}^{(1)}$ at Line 9 for the computation of $C_{[p]}^{(\ell)}$ at Line 4 for $\ell = 2, \ldots, k-1$, but will be updated again at Line 17. The switch of sign at Line 7 conditioned upon Line 6 is to ensure that the iterates will be aligned in one direction and thus avoid jumping back and forth. Also, by Lemma 2.3, the reference to $\boldsymbol{\beta}_\ell$ in the multiplication by $\circledast_{\boldsymbol{\beta}_\ell}$ at Line 4 is entirely unnecessary. We include it in the description to help keep track of the procedure. We register the intermediate values $\lambda_{[p+1]}^{(\ell)}$ as well, even though only $\lambda_{[p+1]}^{(k)}$ at the final stage is crucial.

The above algorithm is different from the alternating least squares (ALS) approach that has been popular for computing the best rank-1 approximation [4, 11, 20]. The most significant difference is that, since the dominant singular vector $\mathbf{u}_{[p+1]}^{(\ell)}$ of the matrix $C_{[p]}^{(\ell)}$ gives rise to the absolute maximal value $\lambda_{[p+1]}^{(\ell)}$ for the functional

$$g(\mathbf{x}, \mathbf{y}) := \left\langle T, \bigotimes_{i=1}^{\ell-1} \mathbf{u}_{[p+1]}^{(i)} \otimes \mathbf{x} \otimes \mathbf{y} \otimes \bigotimes_{i=\ell+2}^{k} \mathbf{u}_{[p]}^{(i)} \right\rangle \tag{4.1}$$

among all possible vectors $\mathbf{x}$ and $\mathbf{y}$, the mechanism of updating $\mathbf{x}$ and $\mathbf{y}$ simultaneously in Algorithm 4.1 is going to increase the generalized Rayleigh quotient faster than the combination of two applications of ALS approach to $\mathbf{x}$ followed by $\mathbf{y}$. The gain is also better than the maximum of updating $\mathbf{x}$ or $\mathbf{y}$ separately as that discussed in [8, Preposition 4].

4.2. Update with random permutation. An alternative way to cut short the iterates required by the ℓ-loop in Algorithm 4.1 is to shuffle the columns $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}$ by a random permutation σ and generate a matrix C for updating. This randomized procedure is modified at Line 7 in Algorithm 4.2. To avoid confusion with data generated from Algorithm 4.1, we employ a slightly different notation when describing this algorithm. Again, by Lemma 2.3, the reference to $\boldsymbol{\beta}_t$ is irrelevant. For simplicity, we always choose to update the last two vectors $\mathbf{u}^{(\sigma_{k-1})}, \mathbf{u}^{(\sigma_k)}$ after the permutation. A classical result in probability theory asserts that the expected

Algorithm 4.3 (Best rank-1 approximation via SVD updating with post-randomization.)

Require: An order-k, n-dimensional, symmetric tensor T and k starting unit vectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)} \in \mathbb{R}^n$
Ensure: A local best rank-1 approximation to T

1: $t \leftarrow 0$
2: $\mu_0 \leftarrow \left\langle T, \bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)} \right\rangle$
3: **repeat**
4: $\quad t \leftarrow t + 1$
5: $\quad C_t \leftarrow T \circledast \bigotimes_{i=1}^{k-2} \mathbf{u}^{(i)}$
6: $\quad [\mathbf{u}_t, s_t, \mathbf{v}_t] = \mathsf{svds}(C_t, 1)$ {Dominant singular value triplet via Matlab routine svds}
7: $\quad \sigma \leftarrow$ random permutation of $\{1, \ldots, k\}$
8: $\quad$ **if** $(\mathbf{u}_t)_1 < 0$ **then**
9: $\quad\quad \mathbf{u}_t = -\mathbf{u}_t$
10: $\quad$ **end if**
11: $\quad \mu_t \leftarrow s_t$
12: $\quad \mathbf{u}^{(\sigma_{k-1})}, \mathbf{u}^{(\sigma_k)} \leftarrow \mathbf{u}_t$ {Randomly replace two factors}
13: **until** μ_t meets convergence criteria

number of trials for a permutation to recur is $\frac{k(k-1)}{2}$. But by the time that a repetition occurs, the vectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}$ should have been updated. If the convergence is ever to happen, the effect of reshuffling will gradually diminish.

4.3. Update with post-randomization. To carry out the permutation $\mathbf{u}^{(\sigma_i)}$ at Line 7 in Algorithm 4.2 is still cumbersome. Since the purpose of permutation is simply to mingle the vectors, we may consider the alternative by postponing the permutation to the end of calculation as is indicated in Algorithm 4.3. It can be argued that Algorithm 4.2 would be equivalent to Algorithm 4.3 in the sense that, if one could foresee the future permutation at Line 7 and prearrange the columns in the order $\left\{\mathbf{u}^{(\sigma_1)}, \ldots, \mathbf{u}^{(\sigma_k)}\right\}$ before Line 5 in Algorithm 4.3, then both algorithms would be using the same C_t. In reality, of course, such a rearrangement does not happen, so we distinguish the progress of the generalized Rayleigh quotient by a different notation μ_t. Though $\mu_0 = \lambda_0$ to begin with, this μ_t in general is not the same as the λ_t generated by Algorithm 4.2 when $t \geq 1$. Note the simplification at Line 5 in Algorithm 4.3 which utilizes only the first $k-2$ vectors. The permutation at Line 12 will help intermingle the vectors before the next step.

Another difference between Algorithm 4.2 and Algorithm 4.3 deserves noting. In Algorithm 4.2, the replacement at Line 13 does not interfere with the vectors $\mathbf{u}^{(\sigma_i)}$, $i \in [\![k-2]\!]$, used to define C_t at Line 7. But in Algorithm 3, the replacement at Line 12 may affect $0, 1$ or 2 many of the first $k-2$ vectors used to define C_t at Line 5.

Indeed, with probability $\frac{(k-2)(k-3)}{k(k-1)}$ the perturbation σ will ask to replace 2 such vectors, which is high when k is large. We may thus consider a subclass of Algorithm 4.3 by requiring the update at Line 12 be limited to $[\![k-2]\!]$. The limit points of $\{\mathbf{u}_t\}$ by this subclass iteration form a subset of those by the unmodified Algorithm 4.3. Our numerical experiments suggest that both versions have the same set of limit points.

4.4. Symmetric update. Finally, since the best rank-1 approximation of symmetric tensors is symmetric, all factors should be the same eventually. It is tempting to exploit the mechanism of keeping the symmetry at every iteration once an SVD is done. We outline the procedure in Algorithm 4.4. The contrast is at Line 5 where C_t is calculated based on one single factor $\mathbf{u}_t$. Indeed, a similar idea has been proposed in [11] as the symmetric high-order power method whose performance has been reported as poor. We shall demonstrate in our numerical experiment that Algorithm 4.4 does not perform competitively either. Though interesting, this algorithm is of little importance to us. We mention it in passing and we do not consider it as a contribution.

Algorithm 4.4 (Best rank-1 approximation via symmetric SVD.)

Require: An order-k, n-dimensional, symmetric tensor T and a starting unit vector $\mathbf{u}_0 \in \mathbb{R}^n$
Ensure: A local best rank-1 approximation to T

$t \leftarrow 0$
$\mu_0 \leftarrow \left\langle T, \bigotimes_{\ell=1}^{k} \mathbf{u}_0 \right\rangle$
repeat
  $t \leftarrow t + 1$
  $C_t \leftarrow T \circledast \bigotimes_{i=1}^{k-2} \mathbf{u}_0$ {Using the same factor for all}
  $[\mathbf{u}_t, s_t, \mathbf{v}_t] = \mathsf{svds}(C_t, 1)$ {Dominant singular value triplet via Matlab routine svds}
  if $(\mathbf{u}_t)_1 < 0$ **then**
    $\mathbf{u}_t = -\mathbf{u}_t$
  end if
  $\mu_t \leftarrow s_t$
  $u_0 \leftarrow \mathbf{u}_t$
until μ_t meets convergence criteria

5. Convergence analysis. In this section, we analyze the convergence for the above algorithms. We first show the monotonicity of the generalized Rayleigh quotients. Most importantly, we argue that the iterates themselves also converge. The latter answers an open question on convergence analysis for the SVD-based methods, as it offers an understanding of what was declared as "we do not have a complete understanding when this will happen" in [8, Page 947]. We concentrate mainly on Algorithm 4.3 for its practicality in implementation. We also include the behavior of Algorithm 4.1 in the Appendix for its elegance in theory. The analysis of Algorithm 4.2 is left to interested readers.

5.1. Convergence of objective values. Because the SVD is involved, where the dominant singular value and singular vector are selected at each update, all three algorithms enjoy the property that the corresponding sequences of the generalized Rayleigh quotients are bounded and monotone increasing.

LEMMA 5.1. *The scalars* $\left\{\lambda_{[p]}^{(\ell)}\right\}$ *generated in Algorithm 4.1 form a monotone convergent sequence for each* $\ell = 1, \ldots, k$ *and all converge to the same value.*

Proof. It suffices to prove the assertion for the case $\lambda_{[p]}^{(k)}$ because the following argument shows that all other cases are sandwiched in between.

By applying Lemma 3.1 to each of the matrices $C_{[p]}^{(\ell)}$ consecutively, we observe that at any stage of p the inequalities

$$\begin{aligned}
\lambda_{[p]}^{(k)} &= \left| \left\langle T, \bigotimes_{\ell=1}^{k} \mathbf{u}_{[p]}^{(\ell)} \right\rangle \right| = \left| \left\langle T \circledast_{\boldsymbol{\beta}_1} \bigotimes_{\ell=3}^{k} \mathbf{u}_{[p]}^{(\ell)}, \mathbf{u}_{[p]}^{(1)} \otimes \mathbf{u}_{[p]}^{(2)} \right\rangle \right| \leq \left| \left\langle T \circledast_{\boldsymbol{\beta}_1} \bigotimes_{\ell=3}^{k} \mathbf{u}_{[p]}^{(\ell)}, \widehat{\mathbf{u}}_{[p+1]}^{(1)} \otimes \widehat{\mathbf{u}}_{[p+1]}^{(2)} \right\rangle \right| \\
&= \lambda_{[p+1]}^{(1)} = \left| \left\langle T \circledast_{\boldsymbol{\beta}_2} \widehat{\mathbf{u}}_{[p+1]}^{(1)} \otimes \bigotimes_{\ell=4}^{k} \mathbf{u}_{[p]}^{(\ell)}, \widehat{\mathbf{u}}_{[p+1]}^{(2)} \otimes \mathbf{u}_{[p]}^{(3)} \right\rangle \right| \leq \left| \left\langle T \circledast_{\boldsymbol{\beta}_2} \widehat{\mathbf{u}}_{[p+1]}^{(1)} \otimes \bigotimes_{\ell=4}^{k} \mathbf{u}_{[p]}^{(\ell)}, \mathbf{u}_{[p+1]}^{(2)} \otimes \widehat{\mathbf{u}}_{[p+1]}^{(3)} \right\rangle \right| \\
&= \lambda_{[p+1]}^{(2)} \leq \cdots \\
&\leq \lambda_{[p+1]}^{(k-1)} = \left| \left\langle T, \widehat{\mathbf{u}}_{[p+1]}^{(1)} \otimes \bigotimes_{\ell=2}^{k-1} \mathbf{u}_{[p+1]}^{(\ell)} \otimes \widehat{\mathbf{u}}_{[p+1]}^{(k)} \right\rangle \right| = \left| \left\langle T \circledast_{\boldsymbol{\beta}_k} \bigotimes_{\ell=2}^{k-1} \mathbf{u}_{[p+1]}^{(\ell)}, \widehat{\mathbf{u}}_{[p+1]}^{(1)} \otimes \widehat{\mathbf{u}}_{[p+1]}^{(k)} \right\rangle \right| \\
&\leq \left| \left\langle T \circledast_{\boldsymbol{\beta}_k} \bigotimes_{\ell=2}^{k-1} \mathbf{u}_{[p+1]}^{(\ell)}, \mathbf{u}_{[p+1]}^{(1)} \otimes \mathbf{u}_{[p+1]}^{(k)} \right\rangle \right| = \lambda_{[p+1]}^{(k)}
\end{aligned}$$

are always maintained. The monotone sequence $\left\{\lambda_{[p]}^{(k)}\right\}$ is bounded above by $\|T\|_F$, so it must converge. The inequalities sandwich the sequences one after another, so their limit points must be the same. ∎

LEMMA 5.2. *The scalars $\{\lambda_t\}$ generated in Algorithm 4.2 form a monotone convergent sequence.*

Proof. By applying Lemma 3.1, it is still true that at any stage of t we always have

$$\lambda_t = \left|\left\langle T, \bigotimes_{\ell=1}^{k} \mathbf{u}^{(\ell)} \right\rangle\right| = \left|\left\langle T \circledast_{\boldsymbol{\beta}_t} \bigotimes_{i=1}^{k-2} \mathbf{u}^{(\sigma_i)}, \mathbf{u}^{(\sigma_{k-1})} \otimes \mathbf{u}^{(\sigma_k)} \right\rangle\right| \leq \left|\left\langle T \circledast_{\boldsymbol{\beta}_t} \bigotimes_{i=1}^{k-2} \mathbf{u}^{(\sigma_i)}, \mathbf{u}_{t+1} \otimes \mathbf{u}_{t+1} \right\rangle\right| = \lambda_{t+1},$$

since $\mathbf{u}_{t+1}$ is the dominant singular vector of the matrix $T \circledast_{\boldsymbol{\beta}_t} \bigotimes_{i=1}^{k-2} \mathbf{u}^{(\sigma_i)}$. Again, the monotone sequence $\{\lambda_t\}$ is bounded above by $\|T\|_F$, so it must converge. ∎

LEMMA 5.3. *The scalars $\{\mu_t\}$ generated in Algorithm 4.3 form a monotone convergent sequence.*

Proof. Suppose that $C_t = T \circledast \bigotimes_{i=1}^{k-2} \mathbf{u}^{(i)}$ has been defined in terms of vectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k-2)}$ from the previous step. Suppose also that its dominant singular vector $\mathbf{u}_t$ has been calculated. Then

$$\mu_t = \left|\left\langle T \circledast \bigotimes_{i=1}^{k-2} \mathbf{u}^{(i)}, \mathbf{u}_t \otimes \mathbf{u}_t \right\rangle\right|. \tag{5.1}$$

To proceed to the next step, some of these columns $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k-2)}$ might be replaced by $\mathbf{u}_t$. Let $\boldsymbol{\Theta}^o$ and $\boldsymbol{\Theta}^+$ denote subsets of $[\![k-2]\!]$ containing indices of those vectors that will not been changed and are to be updated, respectively. Depending on the current permutation σ_t, there are three possible scenarios – the cardinality $|\boldsymbol{\Theta}^+|$ of the set $\boldsymbol{\Theta}^+$ can be 0, 1, or 2. By Lemmas 2.1 and 2.3, we may write

$$\begin{aligned} \mu_t &= \left|\left\langle T, \bigotimes_{i\in\boldsymbol{\Theta}^o} \mathbf{u}^{(i)} \otimes \bigotimes_{j\in\boldsymbol{\Theta}^+} \mathbf{u}^{(j)} \otimes \mathbf{u}_t^2 \right\rangle\right| = \left|\left\langle T \circledast \bigotimes_{i\in\boldsymbol{\Theta}^o} \mathbf{u}^{(i)} \otimes \mathbf{u}_t^{|\boldsymbol{\Theta}^+|}, \bigotimes_{j\in\boldsymbol{\Theta}^+} \mathbf{u}^{(j)} \otimes \mathbf{u}_t^{2-|\boldsymbol{\Theta}^+|} \right\rangle\right| \\ &\leq \left|\left\langle T \circledast \bigotimes_{i\in\boldsymbol{\Theta}^o} \mathbf{u}^{(i)} \otimes \mathbf{u}_t^{|\boldsymbol{\Theta}^+|}, \mathbf{u}_{t+1} \otimes \mathbf{u}_{t+1} \right\rangle\right| = \mu_{t+1}. \end{aligned} \tag{5.2}$$

In the above, we have adopted the notion that a factor of either $\bigotimes_{j\in\emptyset} \mathbf{u}^{(j)}$ or $\mathbf{u}_t^0$ means that it does not occur in the multiplication. ∎

5.2. Convergence of Iterates. The above argument about the monotonicity of values $\lambda_{[p]}^{(k)}$, λ_t, or μ_t is interesting and important, but certainly not enough, because the convergence of objective values does not guarantee the convergence of iterates to a global minimizer nor even to a stationary point [13]. What need be done for both algorithms is to argue that generically the iterates of vectors themselves also converge. Since Algorithm 4.3 is our ultimate choice of scheme in view of its simplicity and effectiveness, we give a detailed account of its dynamical behavior in this section. A far more complicated analysis for Algorithm 4.1 is given in the Appendix.

For clarity, enumerate the column vectors at the end of each t-loop by $\left\{\mathbf{u}_t^{(1)}, \ldots, \mathbf{u}_t^{(k)}\right\}$ in accordance with their original order. By construction, for $t \geq 1$, at least two of these vectors are identical and the rest are unchanged from the previous step. The goal is to prove that these columns converge to the same vector as t goes to infinity, regardless how the random permutation σ which varies in t takes place. Toward this end, we establish a sequence of results.

Since the dominant singular vectors $\mathbf{u}_t$ are always normalized to unit length, the sequence $\{\mathbf{u}_t\}$ must have a convergent subsequence. We first argue that its limit point must propagate to all elements of $\left\{\mathbf{u}_{t_i}^{(1)}, \ldots, \mathbf{u}_{t_i}^{(k)}\right\}$ in the following sense.

LEMMA 5.4. *If* $\{\mathbf{u}_{t_i}\}$ *is a convergent subsequence and* $\lim_{i\to\infty}\mathbf{u}_{t_i}=\overline{\mathbf{u}}$*, then* $\lim_{i\to\infty}\mathbf{u}_{t_i}^{(\ell)}=\overline{\mathbf{u}}$ *for all* $\ell\in[\![k]\!]$.

Proof. Let $\left\{\sigma^{[t_i]}\right\}$ denote the corresponding sequence of permutations used in the algorithm for generating $\{\mathbf{u}_{t_i}\}$. Note that each time at least two vectors in the set $\left\{\mathbf{u}_{t_i}^{(1)},\ldots,\mathbf{u}_{t_i}^{(k)}\right\}$ are identical with $\mathbf{u}_{t_i}$. Specifically, by Line 12 in the algorithm, we have $\mathbf{u}_{t_i}^{\left(\sigma_{k-1}^{[t_i]}\right)}=\mathbf{u}_{t_i}^{\left(\sigma_{k}^{[t_i]}\right)}=\mathbf{u}_{t_i}$. As $\mathbf{u}_{t_i}$ gets close to $\overline{\mathbf{u}}$, so do these two vectors. However, since $\left\{\sigma^{[t_i]}\right\}$ is of uniform distribution varying through all possible permutations as i goes to infinity, the locations of the so called "these two vectors" must also pervade through all possible pairs in the set $[\![k]\!]$. That is, all vectors in $\left\{\mathbf{u}_{t_i}^{(1)},\ldots,\mathbf{u}_{t_i}^{(k)}\right\}$ are close to $\overline{\mathbf{u}}$ when i is large enough. ∎

LEMMA 5.5. *For almost all symmetric tensors* T*, the accumulation points of the sequence* $\{\mathbf{u}_t\}$ *generated by Algorithm 4.3 are geometrically isolated.*

Proof. Suppose that $\overline{\mathbf{u}}$ is an accumulation point. By Corollary 2.4, we may perform the following operation, and by Lemma 5.4, $\overline{\mathbf{u}}$ is a solution to the nonlinear equation

$$T\circledast\mathbf{u}^{k-2}=\left\langle T,\mathbf{u}^{k}\right\rangle\mathbf{u}^{2}. \tag{5.3}$$

The equation (5.3) is a polynomial system of degree $k+2$ in the unknown $\mathbf{u}\in\mathbb{R}^n$ with leading coefficient T. By the theory of parameter continuation [17, Theorem 7.1.1], we know that for almost all symmetric tensor T, except for an affine algebraic subset of codimension one, the solutions to (5.3) are isolated. ∎

We stress that the polynomial system (5.3) might have multiple solutions. We are interested in the real solution that maximizes the generalized Rayleigh quotient $\left\langle T,\mathbf{u}^{k}\right\rangle$. The monotone behavior of $\{\lambda_t\}$ or $\{\mu_t\}$ seems to suggest that this is happening. However, we can conclude only that a local maximum is being realized by the iteration.

THEOREM 5.6. *For almost all symmetric tensors* T*, the sequence* $\{\mathbf{u}_t\}$ *of dominant singular vectors generated in Algorithm 4.3 converges.*

Proof. Suppose that $\{\mathbf{u}_{t_i}\}$ is any subsequence converging to $\overline{\mathbf{u}}$. Suppose also by Lemma 5.5 that $\overline{\mathbf{u}}$ is isolated. By Lemma 5.4, the corresponding subsequences $\left\{\mathbf{u}_{t_i}^{(\ell)}\right\}$ converge to $\overline{\mathbf{u}}$ for all $\ell\in[\![k]\!]$. By construction (Line 5 in Algorithm 4.3), the subsequence $\{C_{t_i+1}\}$ of matrices converges. By continuity, the subsequence $\{\mathbf{u}_{t_i+1}\}$ must also converge to $\overline{\mathbf{u}}$. In particular, $\|\mathbf{u}_{t_i+1}-\mathbf{u}_{t_i}\|\to 0$. The condition in Lemma 2.7 therefore is satisfied. It follows that the whole sequence $\{\mathbf{u}_t\}$ converges to $\overline{\mathbf{u}}$. ∎

We conclude this section with a simple illustration. Given a symmetric tensor T of order 3 and an arbitrary unit vector $\mathbf{u}_0$, Algorithm 4.3 can be cast as fixed-point iteration defined by

$$\mathbf{u}_{t+1}=F(\mathbf{u}_t), \tag{5.4}$$

where $F(\mathbf{u})$ represents the dominant singular vector of $T\circledast\mathbf{u}$ (with consistent sign at the first entry). Then, by Theorem 5.6, the sequence $\{\mathbf{u}_t\}$ converges.

6. Numerical examples. In this section, we carry out some numerical experiments to demonstrate the working of our algorithms. We concentrate mostly on Algorithm 4.3. Because the size of data grows rapidly as n^k, we will not list the test data. At present, we pay no attention to fine tune the implementation for efficiency either. We simply describe how we set up our experiments and present some empirical observations.

Example 1. Our first concern is the complexity analysis of the algorithm. Given the abilities of high performance (vector or parallel) processors today, simple floating-point operations (flops) counts are not at all valid any more. On the other hand, on a dedicated machine, the CPU time should be approximately proportional to the number of flops, albeit the I/O will also cost time. We decide to measure the CPU time required in each of the major components in the algorithm. We divide the measurement as follows:

- T_{Outer} = the time needed to form the outer product $\bigotimes_{i=1}^{k-2} \mathbf{u}^{(i)}$.
- T_C = the time needed to perform the tensor multiplication $\circledast$ for creating C_t in Line 5.
- T_{SVD} = the time needed for the SVD at Line 6.
- T_{Total} = the total execution time, including every other possible details such as I/O.

For the case $k = 3$, we vary the dimension $n = 2^p$ for $p = 4, \ldots, 9$. For the case $k = 4$, an order-4 tensor of dimension $n = 2^8$ requires 32GB bytes. So we limit ourselves to $p = 4, \ldots, 7$ only. Each case of p is repeatedly tested 20 times with random starting unit vectors and we plot the average as the running time in Figure 6.1.

It should not be surprising that the overhead T_{Outer} remains almost constant for the case $k = 3$ because no outer product is needed except for swapping columns at Line 12. However, we find that neither T_{Outer} does vary significantly even for the case $k = 4$. What is interesting is that for small size problems, say, $n \leq 32$, the overhead T_{Total} is attributed mainly to T_{SVD}. But when n is sufficiently large, while the SVD should cost more time, the cost T_C of the tensor product $\circledast$ outweighs T_{SVD} of the SVD. It is seen in Figure 6.1 that when $n = 2^9$ and $k = 3$ or when $n = 2^7$ when $k = 4$, the main contribution to T_{Total} is from T_C.

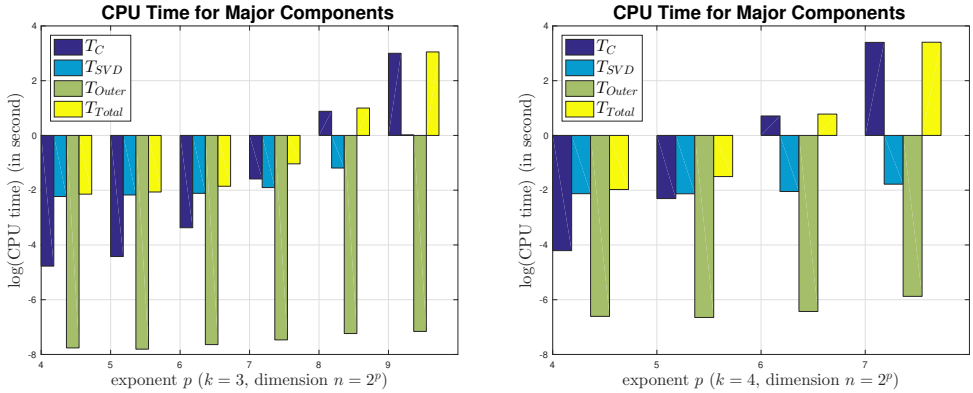

FIGURE 6.1. *Breakdown of CPU time needed for calculation of major parts in Algorithm 4.3.*

Example 2. We are curious about the performance of the SVD-based algorithms when comparing among themselves as well some of the popular methods. For this purpose, we apply the same stopping criteria to all methods — the iteration terminates when three consecutive generalized Rayleigh quotients do not vary more than the tolerance 10^{-8}. We measure the CPU time needed by our Algorithms 1-4, as well as the conventional ALS and symmetric ALS [11]. The problem sizes are chosen in the same way as in Example 1. We execute each algorithm by 20 runs with random initial unit vectors. They may converge to different limit points, but we give every algorithm the same criteria to reach convergence. We compute the average time. In all tests, we find that Algorithm 3 is fastest especially for large p in Figure 6.2.

In Example 1, we find that the $\boldsymbol{\beta}$-product $T \circledast_{\boldsymbol{\beta}} S$ is the most expensive part of the calculation for our Algorithm 4.3. Since the conventional ALS method involves a similar calculation, we make a rough complexity comparison between Algorithm 4.3 and the conventional ALS method. In the ALS method, forming $\bigotimes_{i=1}^{k-1} \mathbf{u}^{(i)}$ requires n^{k-1} entry-to-entry multiplications and there are k layers to form the inner product $T \circledast \bigotimes_{i=1}^{k-1} \mathbf{u}^{(i)}$. Together with the required normalization, the ALS method requires $n^k + n^{k-1} + n$ scalar multiplications per update. In contrast, the SVD-based method requires $n^k + n^{k-2}$ scalar multiplications per $\boldsymbol{\beta}$-product if $k > 3$, and n^3 if $k = 3$. Additionally, the svds involved in the SVD-based method requires more overhead than the normalization involved in the ALS method. It is difficult to estimate how many iterations are required inside the svds to generate the dominant singular value triplet, but the total cost should not be worse than $O(n^3)$. We thus estimate that the SVD-based method requires $O(n^k + n^{k-2} + n^3)$ scalar multiplications per update. These estimates seem compatible per update, however, the numerical evidence in Example 2 clearly demonstrates that overall Algorithm 4.3 is significantly faster than the ALS method.

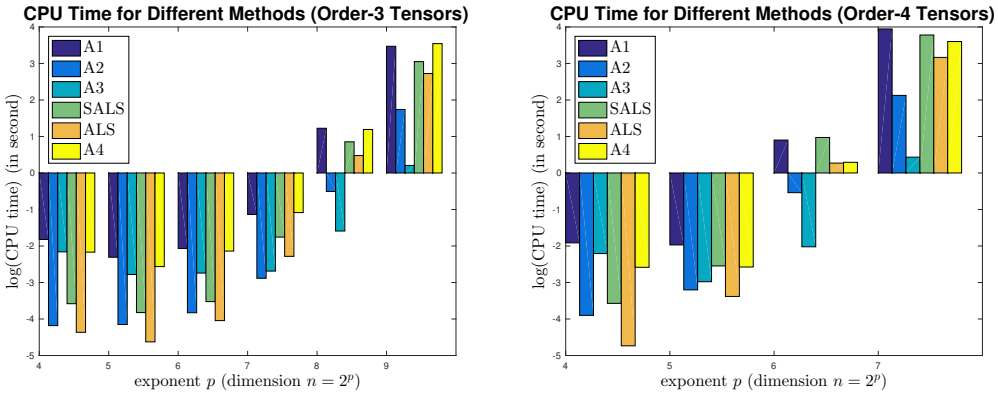

FIGURE 6.2. *Breakdown of CPU time for comparison among different methods.*

Though they are not in the same category, it might be interesting to compare the above-mentioned complexities of the iterative methods in one update to those of finite algorithms. The so called SeROAP method proposed in [5] requires $O(\frac{2p(n^{k+1}-n^2)}{n-1})$ scalar multiplications (where p is a user-defined parameter) for the decreasing order phase and $O(\frac{2(n^{k+1}-n^2)}{n-1})$ scalar multiplications for the projection phase. Similarly, the ST-HOSVD and T-HOSVD methods proposed in [18] require $O(\frac{n^{k+2}+n^k-n^2-1}{n-1})$ and $O(\frac{kn^{k+2}+(1-k)n^{k+1}-n}{n-1})$ operations, respectively. These finite algorithms give rise to some good approximations, but have no mechanism for further improvement. So, they might serve as a good starting point for further iteration such as by our SVD-based methods and the ALS method. We have not explored this hybrid approach in this study.

It is worthy of a further remark on whether or how a scheme preserves symmetry. At first glance, it might seem that in our algorithms A1, A2, and A3, we have imposed symmetry because we update two vectors simultaneously by the same dominant singular vector. The fact is that because only two vectors are updated a time while others are not affected, the symmetry is not required or even expected. One of the most important points in our theory is that at the end all factors in iterates converge to the same vector and, hence, symmetry shows up. On the other hand, as is reported in Figure 6.2, imposing symmetry in the ALS scheme, which is the SALS method, or imposing total symmetry in the SVD-based scheme, which is the A4 algorithm, has no advantage over the conventional ALS or the randomized A3 algorithm.

Finally, we need to point out that, although the symmetric limit point (by non-symmetric algorithms) for symmetric tensors is expected in theory, we have to set some stopping criteria for the iteration in practice. Consequently, the numerical results returned by non-symmetric algorithms might not be perfectly symmetric. In contrast, the symmetric ALS method in [11] and our totally symmetric SVD-based method (A4) always keep the symmetry in each step. They are slower, but have the advantage of keeping the symmetry in bay when comparing to non-symmetric algorithms.

Example 3. Even though we have taken the advantage of the SVD that produces the best approximation per iteration, its effect is limited to the locality. We generate 15 random vectors $\mathbf{x}_i \in \mathbb{R}^{10}$ and combine them into $T = \sum_{i=1}^{15} \mathbf{x}_i^7$. As these vectors $\{\mathbf{x}_i\}$ are linearly dependent, we no longer have a good way to estimate the rank of T [4]. Still, its best rank-1 approximation is guaranteed to exist. Applying Algorithm 4.3 with 20 distinct sets of random starting unit vectors, we plot history of iterations for each of the 20 tests. We continue to observe properties such as convergence and monotonicity discussed earlier in this paper. However, numerical results in Figure 6.3 indicate the possibility of having multiple local solutions. See Lemma 5.5. The number of locally best rank-1 approximations should be the same of real solutions to the polynomial system (5.3), but that depends on T. While the SVD-based algorithms seem capable of capturing the solution with "larger" objective values in most of the trials, this experiment demonstrates that the notion of the best rank-1 approximation should be interpreted only locally.

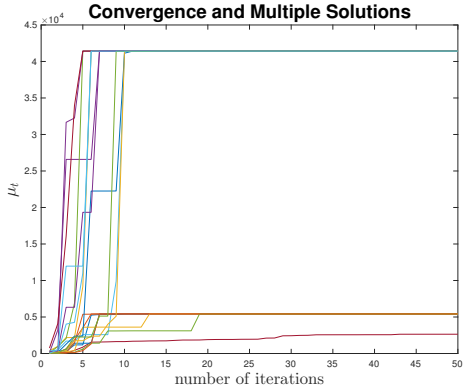

FIGURE 6.3. *History of monotone convergence and existence of multiple best rank-1 approximations.*

Example 4. To experiment the sensitivity of a rank-1 tensor subject to perturbations, we randomly generate six vectors $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_5 \in \mathbb{R}^{10}$ from the identical and independent standard normal distribution. Define $T_0 = \mathbf{x}_0^7$. This rank-1, order-7, dimension-10, and symmetric tensor T_0 will be fixed as our target. Define the unit tensor $B := \frac{\sum_{i=1}^5 \mathbf{x}_i^7}{\|\sum_{i=1}^5 \mathbf{x}_i^7\|_F}$ which generically is of rank 5 [4]. We perturb T_0 via an additive noise of the form

$$T_\sigma = T_0 + \sigma B$$

where $\sigma \in [0, 2]$ signifies the magnitude of the noise. By gradually increasing the strength of perturbation, we compute the rank-1 approximation tensor $\overline{T}_\sigma$ of T_σ by Algorithm 4.3. The noise level $\|T_\sigma - T_0\|_F = \sigma$ is low relative to $\|T_0\|_F$, but the added noise certainly disrupts the rank. Our goal is to compare the difference between $\overline{T}_\sigma$ and the original T_0 as well as the computed generalized Rayleigh quotient $\mu(\overline{T}_\sigma)$. Plotted in Figure 6.4 are the relative differences, showing that the computed rank-1 tensor $\overline{T}_\sigma$ is a reasonable approximation to T_0, but the discretion is large enough to suggest that it is not recovering T_0 exactly. It is interesting to note that, despite of the high nonlinearity involved, the quantities $|\mu(\overline{T}_\sigma) - \mu(T_0)|$ and $\|\overline{T}_\sigma - T_0\|_F$ are almost linearly in σ.

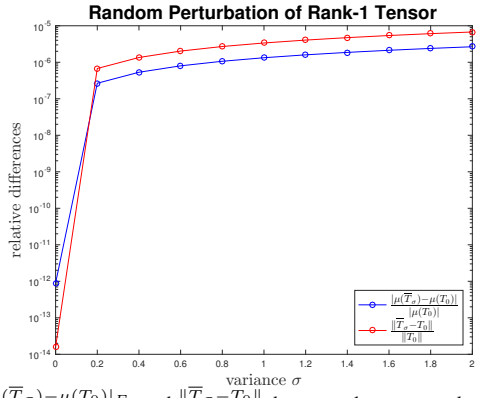

FIGURE 6.4. *Relative difference* $\frac{|\mu(\overline{T}_\sigma)-\mu(T_0)|_F}{|\mu(T_0)|_F}$ *and* $\frac{\|\overline{T}_\sigma-T_0\|}{\|T_0\|}$ *between the computed rank-1 tensor* $\overline{T}_\sigma$ *and the original* T_0.

7. Conclusion. The well developed notion of singular value decomposition can be used to offer a simpler alternative argument that generically the best rank-1 approximation to given symmetric tensor is symmetric. As a by-product, three SVD-based algorithms are proposed for computing the symmetric best rank-1 approximation, which should perform superior to the classical ALS methods. The main contribution of this paper is on the proof of convergence of both the objective values and the iterates generated by these methods.

8. Appendix. Algorithm 4.1 is somewhat too conservative for computation in practice. However, its fundamental structure is the basis of Algorithm 4.3. In this appendix, we prove its convergence which of mathematical interest in its own right.

The difficulty of Algorithm 4.1 is at the complexity that $\mathbf{u}_{[p+1]}^{(\ell)}$ depends on both the new vectors $\left\{\mathbf{u}_{[p+1]}^{(i)}\right\}$, $i = 1, \ldots, \ell-1$, and the old vectors $\left\{\mathbf{u}_{[p]}^{(i)}\right\}$, $i = \ell+2, \ldots, k$. We first make the following observation about the collective behavior.

LEMMA 8.1. *There is a subsequence* $\left\{\mathbf{u}_{[p_j]}^{(\ell)}\right\}$ *generated by Algorithm 4.1 that converges to the same limit point for all* $\ell = [\![k]\!]$.

Proof. For each fixed ℓ, it is always true that $\|C_{[p]}^{(\ell)}\|_F \leq \|T\|_F$ for all p because $\| \bigotimes_{i=1}^{\ell-1} \mathbf{u}_{[p+1]}^{(i)} \otimes \bigotimes_{i=\ell+2}^{k} \mathbf{u}_{[p]}^{(i)} \|_F = 1$. The Bolzano-Weierstrass theorem guarantees a convergent subsequence. There are only finitely many ℓ. Selecting a subsequence of a subsequence, if necessary, we can find a common subset $\{p_j\}$ of nonnegative integers so that $\left\{C_{[p_j]}^{(\ell)}\right\}$ converges simultaneously for all $\ell \in [\![k]\!]$. We claim that the assertion holds for the subsequence $\left\{\mathbf{u}_{[p_j]}^{(\ell)}\right\}$.

For clarity, we accomplish the proof in two steps. First, we argue that the sequences $\left\{\mathbf{u}_{[p_j]}^{(\ell)}\right\}$ converge simultaneously for all $\ell \in [\![k]\!]$. Second, we argue that they converge to the same limit point.

By continuity, the sequences of the corresponding dominant singular vectors $\left\{\widehat{\mathbf{u}}_{[p_j+1]}^{(1)}\right\}$ and $\left\{\mathbf{u}_{[p_j+1]}^{(\ell)}\right\}$ of $\left\{C_{[p_j]}^{(\ell)}\right\}$ converge simultaneously for all $\ell \in [\![k]\!]$. Denote $\lim_{j\to\infty} C_{[p_j]}^{(\ell)} = C^{(\ell)}$, $\lim_{j\to\infty} \widehat{\mathbf{u}}_{[p_j+1]}^{(1)} = \mathbf{u}_{\sharp}^{(1)}$ and $\lim_{j\to\infty} \mathbf{u}_{[p_j+1]}^{(\ell)} = \mathbf{u}_{\sharp}^{(\ell)}$ for $\ell = 2, \ldots, k$. The subscript $_\sharp$ is a handy way to remind us that these are the limit points corresponding to the subsequence $_{[p_j+1]}$. We shall assume the generic condition that $C^{(\ell)}$ is nonsingular for all ℓ.

To prove the simultaneous convergence, we consider separate cases:

Case 1. For $\ell = 4, \ldots k$, let $\eta = \ell - 2$ so that $2 \leq \eta \leq k-2$. By using Lemma 2.2, we obtain the equalities

$$\begin{aligned} C_{[p_j]}^{(\eta)} \mathbf{u}_{[p_j+1]}^{(\eta)} &= \left(T \circledast_{\boldsymbol{\beta}_\eta} \left(\widehat{\mathbf{u}}_{[p_j+1]}^{(1)} \otimes \ldots \otimes \mathbf{u}_{[p_j+1]}^{(\eta-1)} \otimes \mathbf{u}_{[p_j]}^{(\eta+2)} \otimes \mathbf{u}_{[p_j]}^{(\eta+3)} \otimes \ldots \otimes \mathbf{u}_{[p_j]}^{(k)} \right) \right) \mathbf{u}_{[p_j+1]}^{(\eta)} \\ &= \left(T \circledast_{\boldsymbol{\beta}_{\eta+1}} \left(\widehat{\mathbf{u}}_{[p_j+1]}^{(1)} \otimes \ldots \otimes \mathbf{u}_{[p_j+1]}^{(\eta-1)} \otimes \mathbf{u}_{[p_j+1]}^{(\eta)} \otimes \mathbf{u}_{[p_j]}^{(\eta+3)} \otimes \ldots \otimes \mathbf{u}_{[p_j]}^{(k)} \right) \right) \mathbf{u}_{[p_j]}^{(\eta+2)} \\ &= C_{[p_j]}^{(\eta+1)} \mathbf{u}_{[p_j]}^{(\eta+2)}. \end{aligned} \tag{8.1}$$

Taking the limits on both sides of (8.1), together with the non-singularity of $C^{(\eta+1)}$, we see that $\lim_{j\to\infty} \mathbf{u}_{[p_j]}^{(\ell)}$ exists for $\ell = 4, \ldots, k$. The algorithm entails that $\mathbf{u}_{[p_j]}^{(1)} = \mathbf{u}_{[p_j]}^{(k)}$, so the convergence of $\mathbf{u}_{[p_j]}^{(1)}$ is a by-product.

Case 2. For $\ell = 3$, consider the identities

$$\begin{aligned} C_{[p_j]}^{(1)} \widehat{\mathbf{u}}_{[p_j+1]}^{(1)} &= \left(T \circledast_{\boldsymbol{\beta}_1} \left(\mathbf{u}_{[p_j]}^{(3)} \otimes \mathbf{u}_{[p_j]}^{(4)} \otimes \ldots \otimes \mathbf{u}_{[p_j]}^{(k)} \right) \right) \widehat{\mathbf{u}}_{[p_j+1]}^{(1)} \\ &= \left(T \circledast_{\boldsymbol{\beta}_2} \left(\widehat{\mathbf{u}}_{[p_j+1]}^{(1)} \otimes \mathbf{u}_{[p_j]}^{(4)} \otimes \ldots \otimes \mathbf{u}_{[p_j]}^{(k)} \right) \right) \mathbf{u}_{[p_j]}^{(3)} = C_{[p_j]}^{(2)} \mathbf{u}_{[p_j]}^{(3)}. \end{aligned} \tag{8.2}$$

Taking the limits at both ends of (8.2), together with $C^{(2)}$ being nonsingular, we see that $\lim_{j\to\infty} \mathbf{u}_{[p_j]}^{(3)}$ exists.

Case 3. For $\ell = 2$, observe the relationship

$$\begin{aligned} C^{(k)}_{[p_j-1]}\mathbf{u}^{(k)}_{[p_j]} &= \pm\lambda^{(k)}_{[p_j]}\mathbf{u}^{(k)}_{[p_j]} \\ &= \left(T\circledast_{\boldsymbol{\beta}_k}\left(\mathbf{u}^{(2)}_{[p_j]}\otimes\ldots\otimes\mathbf{u}^{(k-1)}_{[p_j]}\right)\right)\mathbf{u}^{(k)}_{[p_j]} \\ &= \left(T\circledast_{\boldsymbol{\beta}_1}\left(\mathbf{u}^{(3)}_{[p_j]}\otimes\ldots\otimes\mathbf{u}^{(k)}_{[p_j]}\right)\right)\mathbf{u}^{(2)}_{[p_j]} = C^{(1)}_{[p_j]}\mathbf{u}^{(2)}_{[p_j]}. \end{aligned} \tag{8.3}$$

Though we do not know the convergence of $C^{(k)}_{[p_j-1]}$, the convergence of $\lambda^{(k)}_{[p_j]}$, $\mathbf{u}^{(k)}_{[p_j]}$ and $C^{(1)}_{[p_j]}$ implies that $\lim_{j\to\infty}\mathbf{u}^{(2)}_{[p_j]}$ exists, which in turn implies that $C^{(k)}_{[p_j-1]}$ converges.

Now we prove that sequences converge to the same limit point. Denote $\lim_{j\to\infty}\mathbf{u}^{(\ell)}_{[p_j]} = u^{(\ell)}_{\natural}$ for $\ell = 1,\ldots,k$. It thus becomes clear that

$$C^{(\ell)} = T\circledast_{\boldsymbol{\beta}_\ell}\left(\mathbf{u}^{(1)}_{\sharp}\otimes\ldots\otimes\mathbf{u}^{(\ell-1)}_{\sharp}\otimes\mathbf{u}^{(\ell+2)}_{\natural}\otimes\mathbf{u}^{(\ell+3)}_{\natural}\otimes\ldots\otimes\mathbf{u}^{(k)}_{\natural}\right). \tag{8.4}$$

Analogous to (8.1), we mention also the identity

$$\begin{aligned} C^{(k-1)}_{[p_j]}\mathbf{u}^{(k-1)}_{[p_j+1]} &= \pm\lambda^{(k-1)}_{[p_j+1]}\mathbf{u}^{(k-1)}_{[p_j+1]} \\ &= \left(T\circledast_{\boldsymbol{\beta}_{k-1}}\left(\widehat{\mathbf{u}}^{(1)}_{[p_j+1]}\otimes\mathbf{u}^{(2)}_{[p_j+1]}\otimes\ldots\otimes\mathbf{u}^{(k-2)}_{[p_j+1]}\right)\right)\mathbf{u}^{(k-1)}_{[p_j+1]} \\ &= \left(T\circledast_{\boldsymbol{\beta}_k}\left(\mathbf{u}^{(2)}_{[p_j+1]}\otimes\ldots\otimes\mathbf{u}^{(k-2)}_{[p_j+1]}\otimes\mathbf{u}^{(k-1)}_{[p_j+1]}\right)\right)\widehat{\mathbf{u}}^{(1)}_{[p_j+1]} = C^{(k)}_{[p_j]}\widehat{\mathbf{u}}^{(1)}_{[p_j+1]}. \end{aligned} \tag{8.5}$$

It follows, by construction and continuity, that we have the relationships

$$\pm\widetilde{\lambda}\mathbf{u}^{(1)}_{\natural} = \pm\widetilde{\lambda}\mathbf{u}^{(k)}_{\natural} = C^{(1)}\mathbf{u}^{(2)}_{\natural}, \tag{8.6}$$

$$C^{(\ell)}\mathbf{u}^{(\ell)}_{\sharp} = \pm\widetilde{\lambda}\mathbf{u}^{(\ell)}_{\sharp} = C^{(\ell+1)}\mathbf{u}^{(\ell+2)}_{\natural}, \quad \ell = 1,\ldots,k-2, \tag{8.7}$$

$$C^{(k-1)}\mathbf{u}^{(k-1)}_{\sharp} = \pm\widetilde{\lambda}\mathbf{u}^{(k-1)}_{\sharp} = C^{(k)}\mathbf{u}^{(1)}_{\sharp}, \tag{8.8}$$

$$C^{(k)}\mathbf{u}^{(k)}_{\sharp} = \pm\widetilde{\lambda}\mathbf{u}^{(k)}_{\sharp}. \tag{8.9}$$

By continuity again, note that $\widetilde{\lambda}$ is the dominant singular value of all matrices $C^{(\ell)}$, $\ell = 1,\ldots,k$. Under the assumption that $\widetilde{\lambda}$ is simple, the corresponding singular vector is unique up to a sign change. However, because in Lines 6 to 8 of Algorithm 4.1 we have required that the first entry of the dominant singular vector be positive, such a sign change does not exist. Recursively, we obtain the relationships

$$\begin{cases} \mathbf{u}^{(1)}_{\sharp} = \mathbf{u}^{(1)}_{\natural} = \mathbf{u}^{(k)}_{\natural} = \mathbf{u}^{(2)}_{\natural}, & \\ \mathbf{u}^{(\ell)}_{\sharp} = \mathbf{u}^{(\ell+2)}_{\natural}, & \ell = 1,\ldots,k-2, \\ \mathbf{u}^{(1)}_{\sharp} = \mathbf{u}^{(k-1)}_{\sharp} = \mathbf{u}^{(k)}_{\sharp}. & \end{cases} \tag{8.10}$$

These relationships allow us to write

$$\begin{aligned} \widetilde{\lambda} &= \left|\left\langle C^{(1)}, \mathbf{u}^{(1)}_{\sharp}\otimes\mathbf{u}^{(1)}_{\sharp}\right\rangle\right| = \left|\left\langle T\circledast_{\boldsymbol{\beta}_1}\left(\mathbf{u}^{(3)}_{\natural}\otimes\mathbf{u}^{(4)}_{\natural}\otimes\ldots\otimes\mathbf{u}^{(k)}_{\natural}\right), \mathbf{u}^{(1)}_{\natural}\otimes\mathbf{u}^{(2)}_{\natural}\right\rangle\right| \\ &= \left|\left\langle T\circledast_{\boldsymbol{\beta}_2}\left(\mathbf{u}^{(1)}_{\natural}\otimes\mathbf{u}^{(4)}_{\natural}\otimes\ldots\otimes\mathbf{u}^{(k)}_{\natural}\right), \mathbf{u}^{(2)}_{\natural}\otimes\mathbf{u}^{(3)}_{\natural}\right\rangle\right| = \left|\left\langle T\circledast_{\boldsymbol{\beta}_2}\left(\mathbf{u}^{(1)}_{\sharp}\otimes\mathbf{u}^{(4)}_{\natural}\otimes\ldots\otimes\mathbf{u}^{(k)}_{\natural}\right), \mathbf{u}^{(2)}_{\natural}\otimes\mathbf{u}^{(3)}_{\natural}\right\rangle\right|. \end{aligned}$$

But we also have

$$\widetilde{\lambda} = \left| \left\langle C^{(2)}, \mathbf{u}_\sharp^{(2)} \otimes \mathbf{u}_\sharp^{(2)} \right\rangle \right| = \left| \left\langle T \circledast_{\boldsymbol{\beta}_2} \left(\mathbf{u}_\sharp^{(1)} \otimes \mathbf{u}_\natural^{(4)} \otimes \ldots \otimes \mathbf{u}_\natural^{(k)} \right), \mathbf{u}_\sharp^{(2)} \otimes \mathbf{u}_\sharp^{(2)} \right\rangle \right|.$$

By the uniqueness of the dominant singular vector, since the first entry is kept positive, we conclude that

$$\mathbf{u}_\sharp^{(2)} = \mathbf{u}_\natural^{(2)}. \tag{8.11}$$

Repeating this process, we can prove that $\mathbf{u}_\sharp^{(\ell)} = \mathbf{u}_\natural^{(\ell)}$ for $\ell = 2, \ldots, k-1$. Together with (8.10), we finally prove that the sequence $\left\{ \mathbf{u}_{[p_j]}^{(\ell)} \right\}$ converges to the same limit point for all $\ell = 1, \ldots k$. ∎

COROLLARY 8.2. *If the sequences* $\left\{ C_{[p_j]}^{(\ell)} \right\}$ *of matrices converge simultaneously for all* $\ell \in [\![k]\!]$*, then they converge to the same limit point for all* $\ell \in [\![k]\!]$.

Proof. The assertion follows from (8.4) and the fact that $\mathbf{u}_\sharp^{(\ell)} = \mathbf{u}_\natural^{(\ell)}$ for $\ell \in [\![k]\!]$ which has been proved in the previous lemma. ∎

In the proof of Lemma 8.1, we make use of simultaneously convergent subsequences $\left\{ C_{[p_j]}^{(\ell)} \right\}$ to argue the simultaneously convergent subsequences $\left\{ \mathbf{u}_{[p_j]}^{(\ell)} \right\}$. We can also reverse the argument.

COROLLARY 8.3. *If subsequences* $\left\{ \mathbf{u}_{[p_j]}^{(\ell)} \right\}$ *converge simultaneously for all* $\ell \in [\![k]\!]$*, then so do subsequences* $\left\{ C_{[p_j]}^{(\ell)} \right\}$ *and* $\left\{ \mathbf{u}_{[p_j+1]}^{(\ell)} \right\}$.

Proof. The simultaneous convergence of $\left\{ \mathbf{u}_{[p_j]}^{(\ell)} \right\}$ for $\ell = 3, \ldots k$ implies that the subsequence $\left\{ C_{[p_j]}^{(1)} \right\}$ converges. By continuity, $\left\{ \widehat{\mathbf{u}}_{[p_j+1]}^{(1)} \right\}$ converges. But then by definition, $\left\{ C_{[p_j]}^{(2)} \right\}$ converges and, thus, so does $\left\{ \mathbf{u}_{[p_j+1]}^{(2)} \right\}$. Cycling through the ℓ-loop in Algorithm 4.1, the assertion is proved. ∎

THEOREM 8.4. *For almost all symmetric tensors* T*, the sequences* $\{\mathbf{u}_{[p]}^{(\ell)}\}$ *generated in Algorithm 4.1 converge to the same limit point.*

Proof. Let $\left\{ \mathbf{u}_{[p_j]}^{(\ell)} \right\}$ be any simultaneously convergent subsequences. By Corollary 8.3, the corresponding subsequences $\left\{ C_{[p_j]}^{(1)} \right\}$ converge simultaneously. Using the same argument in the proof of Lemma 8.1, we see that subsequences $\left\{ \mathbf{u}_{[p_j]}^{(\ell)} \right\}$ and $\left\{ \mathbf{u}_{[p_j+1]}^{(\ell)} \right\}$ converge to the same limit point for all $\ell \in [\![k]\!]$. The limit point must satisfy the polynomial system (5.3), whence we assume is geometrically isolated. By Lemma 2.7, we obtain the convergence. ∎

REFERENCES

[1] S. BANACH, *Ãber homogene polynome in (l{2})*, Studia Mathematica, 7 (1938), pp. 36–44.

[2] K. BATSELIER, H. LIU, AND N. WONG, *A constructive algorithm for decomposing a tensor into a finite sum of orthonormal rank-1 terms*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 1315–1337.

[3] J. BRACHAT, P. COMON, B. MOURRAIN, AND E. TSIGARIDAS, *Symmetric tensor decomposition*, Linear Algebra Appl., 433 (2010), pp. 1851–1872.

[4] P. COMON, G. GOLUB, L.-H. LIM, AND B. MOURRAIN, *Symmetric tensors and symmetric tensor rank*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 1254–1279.

[5] A. P. DA SILVA, P. COMON, AND A. L. F. DE ALMEIDA, *A finite algorithm to compute rank-1 tensor approximations*, IEEE Signal Processing Letters, 23 (2016), pp. 959–963.

[6] M. DANA AND K. D. IKRAMOV, *On the codimension of the variety of symmetric matrices with multiple eigenvalues*, Zap. Nauchn. Sem. S.-Peterburg. Otdel. Mat. Inst. Steklov. (POMI), 323 (2005), pp. 34–46, 224.

[7] S. FRIEDLAND, *Best rank one approximation of real symmetric tensors can be chosen symmetric*, Front. Math. China, 8 (2013), pp. 19–40.
[8] S. FRIEDLAND, V. MEHRMANN, R. PAJAROLA, AND S. K. SUTER, *On best rank one approximation of tensors*, Numer. Linear Algebra Appl., 20 (2013), pp. 942–955.
[9] M. ISHTEVA, P.-A. ABSIL, AND P. VAN DOOREN, *Jacobi algorithm for the best low multilinear rank approximation of symmetric tensors*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 651–672.
[10] Y.-L. JIANG AND X. KONG, *On the uniqueness and perturbation to the best rank-one approximation of a tensor*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 775–792.
[11] E. KOFIDIS AND P. A. REGALIA, *On the best rank-1 approximation of higher-order supersymmetric tensors*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 863–884.
[12] T. G. KOLDA, *Numerical optimization for symmetric tensor decomposition*, Math. Program., 151 (2015), pp. 225–248.
[13] T. G. KOLDA AND B. W. BADER, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500.
[14] J. J. MORÉ AND D. C. SORENSEN, *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 553–572.
[15] G. NI AND Y. WANG, *On the best rank-1 approximation to higher-order symmetric tensors*, Math. Comput. Modelling, 46 (2007), pp. 1345–1352.
[16] L. QI, *The best rank-one approximation ratio of a tensor space*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 430–442.
[17] A. J. SOMMESE AND C. W. WAMPLER, II, *The numerical solution of systems of polynomials arising in engineering and science*, World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2005.
[18] N. VANNIEUWENHOVEN, R. VANDEBRIL, AND K. MEERBERGEN, *A new truncation strategy for the higher-order singular value decomposition*, SIAM J. Sci. Comput., 34 (2012), pp. A1027–A1052.
[19] L. WANG AND M. T. CHU, *On the global convergence of the alternating least squares method for rank-one approximation to generic tensors*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 1058–1072.
[20] T. ZHANG AND G. H. GOLUB, *Rank-one approximation to high order tensors*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 534–550 (electronic).
[21] X. ZHANG, C. LING, AND L. QI, *The best rank-1 approximation of a symmetric tensor and related spherical optimization problems*, SIAM J. Matrix Anal. Appl., 33 (2012), pp. 806–821.