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 this 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

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

1. Introduction. A real-valued tensor of order $k$ can be represented by a $k$-way array

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

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

$$\bigotimes_{\ell=1}^{k} u^{(\ell)} = u^{(1)} \otimes \cdots \otimes u^{(k)} := \begin{bmatrix} u^{(1)}_{i_1} \cdots u^{(k)}_{i_k} \end{bmatrix},$$

where elements are the products of entries from vectors $u^{(\ell)} \in \mathbb{R}^{I_j}$, $\ell = 1, \ldots, k$, 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

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

(1.1)

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

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

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

(1.2)

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

$$\lambda = \lambda (u^{(1)}, \ldots, u^{(k)}) = \left< T, \bigotimes_{\ell=1}^{k} u^{(\ell)} \right>.$$  

(1.3)

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
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 \(u^{(i)}\), 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, 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 to symmetric tensors 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. In the first part, 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. In the second part, we turn that argument can be turned into iterative SVD-based algorithms for computing the symmetric best rank-1 approximation. Our main focus is on the second part of this paper where we offer a convergence analysis that is new in the literature.

We organize this paper 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, but 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 \(\llbracket m \rrbracket\) denote henceforth the set of integers \(\{1, \ldots, m\}\) for a given positive integer \(m\). Suppose that the set \(\llbracket k \rrbracket\) is partitioned as the union of two disjoint nonempty subsets \(\alpha = \{\alpha_1, \ldots, \alpha_s\}\) and \(\beta = \{\beta_1, \ldots, \beta_t\}\), where \(s + t = k\). An element in the tensor \(T \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_k}\) will be marked as \(\tau_{[I,J]}^{(\alpha,\beta)}\) where \(I := (i_1, \ldots, i_s)\) and \(J := (j_1, \ldots, j_t)\) contain those indices at locations \(\alpha\) and \(\beta\), respectively. Each index in the arrays \(I\) and \(J\) 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 \((\alpha, \beta)\) is clear, then without causing ambiguity we abbreviate the element as \(\tau_{[I,J]}\). For example, if \(\alpha = \{2, 4\}\) and \(\beta = \{1, 3, 5, 6\}\), then \(\tau_{2,3,5,6,4} = \tau_{3,5,2,1,6,4}\). Note that the representation \(\tau_{[I,J]}\) is \((\alpha, \beta)\) specific, but there is no preference of \(\alpha\) over \(\beta\). The partition \((\alpha, \beta)\) may be regarded as generalizing the notion of row and column which we are familiar with in matrices.

Given a fixed partitioning \(\llbracket k \rrbracket = \alpha \cup \beta\), we shall regard an order-\(k\) tensor \(T \in \mathbb{R}^{I_1 \times \cdots \times I_k}\) 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

\[
\mathcal{J}_\beta : \mathbb{R}^{I_{\alpha_1} \times \cdots \times I_{\alpha_s}} \to \mathbb{R}^{I_{\beta_1} \times \cdots \times I_{\beta_t}},
\]  

(2.1)
such that for any $S \in \mathbb{R}^{I_{\alpha_1} \times \cdots \times I_{\alpha_s}}$, we have

$$\mathcal{T}_\beta(S) := T\otimes_\beta S = \left[\langle\tau_{[\ell_1, \ldots, \ell_\epsilon]}, S\rangle\right] \in \mathbb{R}^{I_{\beta_1} \times \cdots \times I_{\beta_s}},$$

(2.2)

where

$$\langle\tau_{[\ell_1, \ldots, \ell_\epsilon]}, S\rangle := \sum_{i_1 = 1}^{I_{\alpha_1}} \cdots \sum_{i_s = 1}^{I_{\alpha_s}} \tau_{[i_1, \ldots, i_s]}^{[\ell_1, \ldots, \ell_\epsilon]} S_{i_1, \ldots, i_s}$$

(2.3)

is the Frobenius inner product generalized to multi-dimensional arrays. In other words, the product $\otimes_\beta$ generalizes the conventional matrix-to-vector multiplication by combining the $(\ell_1, \ldots, \ell_\epsilon)$-th “slice” in the $\beta$ direction of the tensor $T$, i.e., $\tau_{[\ell_1, \ldots, \ell_\epsilon]}$, 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 $T \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_k}$, a partitioning $[k] = \alpha \cup \beta$, and vectors $u^{(\ell)} \in \mathbb{R}^{I_\ell}$, $\ell = 1, \ldots, k$, then it holds that

$$\begin{aligned}
\left\langle T, \bigotimes_{\ell = 1}^k u^{(\ell)} \rightangle &= \left\langle T\otimes_\beta \bigotimes_{i=1}^s u^{(\alpha_i)}, \bigotimes_{j=1}^t v^{(\beta_j)} \right\rangle.
\end{aligned}$$

(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. \(\square\)

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

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

(2.5)

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

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

$$\begin{aligned}
&= \sum_{\mu = 1}^{I_{\beta_2}} \left( \sum_{i_1 = 1}^{I_{\alpha_1}} \cdots \sum_{i_{k-2} = 1}^{I_{\alpha_{k-2}}} \tau_{[i_1, \ldots, i_{k-2}]}^{[\alpha,\beta]} \alpha_1 \cdots u^{(\alpha_{k-2})} \right) v_{\mu}
\end{aligned}$$

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. \(\square\)

**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 $v^{(\ell)} \in \mathbb{R}^n$, $\ell = 1, \ldots, s$, be arbitrary vectors. Then the product $T\otimes_\beta \bigotimes_{i=1}^s v^{(\rho)}$ is a symmetric tensor of order $t$ and is independent of any permutation $\rho$ of $[s]$ and any subset $\beta \subset [k]$ with cardinality $t$. 3
Proof. Let \([k] = \alpha \cup \beta\) be a partitioning where, without loss of generality, indices in \(\beta\) are arranged in ascending order. Then for \(\ell_j \in [n], j = 1, \ldots, t\), we have

\[
\left( T_{\otimes \beta} \otimes T \right)_{i=1}^{s} v^{(\rho_i)}_{\ell_1, \ldots, \ell_t} = \sum_{i_1=1}^{n} \cdots \sum_{i_s=1}^{n} \tau_{[i_1, \ldots, i_s]}(\alpha, \beta) v^{(\rho_1)}_{i_1} \cdots v^{(\rho_s)}_{i_s} = \sum_{i_1=1}^{n} \cdots \sum_{i_s=1}^{n} \tau_{1(1), \ldots, 1s} v^{(\rho_1)}_{i_1} \cdots v^{(\rho_s)}_{i_s} = \sum_{i_s=1}^{n} \tau_{1(1), \ldots, 1s} v^{(\rho_1)}_{i_1} \cdots v^{(\rho_s)}_{i_s}.
\]

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

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

\[
T_{\otimes \beta} \otimes \left( T_{\otimes \beta} \otimes v^{(\rho_1)} \right) = \left( T_{\otimes \beta} \otimes v^{(\rho_1)} \right)_{i=1}^{s} \otimes v^{(\rho_i)}_{i=j+1} \tag{2.6}
\]

holds for any \(j \in [s]\) and any permutation \(\rho\) of \([s]\).

When the subset \(\beta \subset [k]\) 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 \(T \in \mathbb{R}^{n \times \ldots \times n}\) of order \(k\) is symmetric and a subset \(\beta \subset [k]\) with cardinality 2, then, with respect to any tensor \(S \in \mathbb{R}^{n \times \ldots \times n}\) of order \(k - 2\), the product \(T_{\otimes \beta} S\) is a symmetric matrix.

Proof. For convenience, write \(M := T_{\otimes \beta} S\). 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 real number \(r\), let \(B_r(z)\) denote the neighborhood \([x-r, x+r]\) of \(x\).

For any \(0 < \epsilon < \frac{1}{2} \min\{|x-z|, |y-z|\}\), there exists a large enough integer \(K = K(\epsilon)\) such that \(a_{\alpha_k} \in B_\epsilon(z), a_{\beta_k} \in B_\epsilon(y), |a_{k+1} - a_k| < \epsilon\) for all \(k \geq K\). Infinitely many elements of \(\{a_k\}\) must leave \(B_\epsilon(z)\) to enter \(B_\epsilon(y)\) and vice versa. By doing so, there is an infinite subsequence of \(\{a_k\}\) contained in \(B_\epsilon(z)\). This shows that \(z\) is also an accumulation point. Since \(x\) 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 sequence \(\{a_{k_{j_i}}\}\) 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 $N_{\epsilon}(a^*) := \{ x \in \mathbb{R} | |x - a^*| \leq \epsilon \}$ such that $a^*$ is the only accumulation point of the sequence $\{a_k\}$. Let $\{a_k\}$ 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.,

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

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

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

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

Being bounded, a subsequence $\{a_{\ell_j}\}$ of $\{a_j\}$ must converge. Being contained in $N_{\epsilon}(a^*)$, the limit point must be $a^*$. Therefore $|a_{\ell_j} - a^*| < \frac{\epsilon}{2}$ 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

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

This is a contradiction. $\square$

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

$$\max_{y \in \mathbb{R}^m, \|y\| = 1} \frac{y^T A y}{z^T A z}$$  

is precisely the largest singular value $\sigma_1$ of $A$, where the global maximizer $(y_1, z_1)$ consists of precisely the corresponding left and right singular vectors. The best rank-1 approximation to $A$ is given by $\sigma_1 y_1 z_1^T$. In the event that $A \in \mathbb{R}^{m \times m}$ is symmetric and that the largest singular value of $A$ is simple, then $y = \pm z$ 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 $\lambda \bigotimes_{\ell=1}^k \mathbf{u}^{(\ell)}$ is the best rank-1 approximation to a given order-$k$ symmetric tensor $T$. By (1.3), the generalized Rayleigh quotient $\lambda = \langle T, \bigotimes_{\ell=1}^k \mathbf{u}^{(\ell)} \rangle$ is positive and maximal. Consider the case $\beta = \{1, 2\}$. By Lemma 2.1, we can write

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

By Corollary 2.5, the matrix $C := T^\mathbf{\beta} \bigotimes_{\ell=3}^k \mathbf{u}^{(\ell)}$ is symmetric. Assume that the largest singular value, which is $\lambda$, of $C$ is simple. Then, by Lemma 3.1, we conclude that $\mathbf{u}^{(1)} = \pm \mathbf{u}^{(2)}$. Moving to the choice $\beta = \{2, 3\}$

---

$^1$ 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.
and assuming again that $\lambda$ is simple for the newly defined matrix $C$, we then have $\mathbf{u}^{(2)} = \pm \mathbf{u}^{(3)}$. Continuing this process, we conclude that $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}$ differ from each other by at most a negative sign. At the end, we may write $\lambda \odot_{\ell=1}^k \mathbf{u}^{(\ell)} = \pm \lambda \mathbf{u}^{(1)k}$. 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 1. For efficiency, we also propose a modification by random permutations in Algorithm 2 followed by a more simplified Algorithm 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.

To convey the idea, it is convenient to adopt the subscript $\vert_p$ in Algorithm 1 to indicate the quantity at the $p$-th iteration. Each sweep of $p$ at Line 1 in Algorithm 1 involves $k$ pairs of $\beta$ ranging circularly from $(1, 2), (2, 3), \ldots, (k, 1)$. Thus, each $\mathbf{u}^{(\ell)}_{\vert_{p+1}}$ is updated twice. The first updates for $\ell = 2, \ldots, k$, denoted by $\mathbf{\hat{u}}^{(\ell)}_{\vert_{p+1}}$ at Line 10, are not essential and can be completely removed from the algorithm without affecting the calculation, but its presence helps bridge the monotonicity in theory. The update $\mathbf{\hat{u}}^{(1)}_{\vert_{p+1}}$ is temporarily overwritten as $\mathbf{u}^{(1)}_{\vert_{p+1}}$ at Line 9 for the computation of $C^{(\ell)}_{\vert_p}$ 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 \( \beta_t \) in the multiplication by \( \otimes \beta_t \) at Line 4 is entirely unnecessary. We include it in the description to help keep track of the procedure. We register the intermediate values \( \lambda_{\ell}^{( \ell )} \) as well, even though only \( \lambda_{\ell+1}^{( \ell )} \) 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 \( u_{\ell+1}^{( \ell )} \) of the matrix \( C_{\ell}^{( \ell )} \) gives rise to the absolute maximal value \( \lambda_{\ell+1}^{( \ell )} \) for the functional

\[
g(x, y) := \left\langle T, \otimes_{i=1}^{\ell-1} u_{[\ell+1]}^{(\ell)} \otimes x \otimes \otimes_{i=\ell+2}^{k} u_{[\ell]}^{(\ell)} \right\rangle
\]

among all possible vectors \( x \) and \( y \), the mechanism of updating \( x \) and \( y \) simultaneously in Algorithm 1 is going to increase the generalized Rayleigh quotient faster than the combination of two applications of ALS approach to \( x \) followed by \( y \). The gain is also better than the maximum of updating \( x \) or \( y \) 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 \( \ell \)-loop in Algorithm 1 is to shuffle the columns \( u^{(1)}, \ldots, u^{(k)} \) by a random permutation \( \sigma \) and generate a matrix \( C \) for updating. This randomized procedure is modified at Line 7 in Algorithm 2. To avoid confusion with data generated from Algorithm 1, we employ a slightly different notation when describing this algorithm. Again, by Lemma 2.3, the reference to \( \beta_t \) is irrelevant. For simplicity, we always choose to update the last two vectors \( u^{(\sigma_{k-1})}, u^{(\sigma_k)} \) after the permutation. A classical result in probability theory asserts that the expected number of trials for a permutation to recur is \( \frac{k(k-1)}{2} \). But by the time that a repetition occurs, the vectors \( u^{(1)}, \ldots, u^{(k)} \) 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 \( u^{(\sigma_t)} \) at Line 7 in Algorithm 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 3. It can be argued that Algorithm 2 would be equivalent to Algorithm 3 in the sense that, if one could foresee the future permutation at Line 7 and prearrange the columns in the order \( \{ u^{(\sigma_1)}, \ldots, u^{(\sigma_k)} \} \) before Line 5 in Algorithm 3, then both


Algorithm 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 $u^{(1)}, \ldots, u^{(k)} \in \mathbb{R}^n$

Ensure: A local best rank-1 approximation to $T$

1: $t \leftarrow 0$
2: $\mu_0 \leftarrow \left\langle T, \bigotimes_{i=1}^{k} u^{(i)} \right\rangle$
3: repeat
4: $t \leftarrow t + 1$
5: $C_t \leftarrow T \bigotimes_{i=1}^{k-2} u^{(i)}$
6: $[u_t, s_t, v_t] = \text{svds}(C_t, 1)$ \{Dominant singular value triplet via Matlab routine \text{svds}\}
7: $\sigma \leftarrow \text{random permutation of } \{1, \ldots, k\}$
8: if $(\sigma_1) < 0$ then
9: $u_t = -u_t$
10: end if
11: $\mu_t \leftarrow s_t$
12: $u^{(\sigma_1-1)}, u^{(\sigma_k)} \leftarrow u_t$ \{Randomly replace two factors\}
13: until $\mu_t$ meets convergence criteria

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 $\mu_t$. Though $\mu_0 = \lambda_0$ to begin with, this $\mu_t$ in general is not the same as the $\lambda_t$ generated by Algorithm 2 when $t \geq 1$. Note the simplification at Line 5 in Algorithm 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 2 and Algorithm 3 deserves noting. In Algorithm 2, the replacement at Line 12 does not interfere with the vectors $u^{(\sigma_i)}$, $i \in [k-2]$, 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 $\sigma$ will ask to replace 2 such vectors, which is high when $k$ is large. We may thus consider a subclass of Algorithm 3 by requiring the update at Line 12 be limited to $[k-2]$. The limit points of $\{u_t\}$ by this subclass iteration form a subset of those by the unmodified Algorithm 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. The contrast is at Line 5 where $C_t$ is calculated based on one single factor $u_t$. 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 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.

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 3 for its practicality in implementation. We also include the behavior of Algorithm 1 in the Appendix for its elegance in theory. The analysis of Algorithm 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.
**Algorithm 4** (Best rank-1 approximation via symmetric SVD.)

Require: An order-\( k \), \( n \)-dimensional, symmetric tensor \( T \) and a starting unit vector \( u_0 \in \mathbb{R}^n \)

Ensure: A local best rank-1 approximation to \( T \)

1. \( t \leftarrow 0 \)
2. \( \mu_0 \leftarrow \langle T, \bigotimes_{i=1}^k u_0 \rangle \)
3. **repeat**
4. \( t \leftarrow t + 1 \)
5. \( C_t \leftarrow T \bigotimes \bigotimes_{i=1}^{k-2} u_0 \) \quad \{Using the same factor for all\}
6. \([u_t, s_t, v_t] = \text{svds}(C_t, 1)\) \quad \{Dominant singular value triplet via Matlab routine svds\}
7. **if** \( (u_t)_1 < 0 \) **then**
8. \( u_t = -u_t \)
9. **end if**
10. \( \mu_t \leftarrow s_t \)
11. \( u_0 \leftarrow u_t \)
12. **until** \( \mu_t \) meets convergence criteria

**Lemma 5.1.** The scalars \( \{\lambda_{[p]}^{(\ell)}\} \) generated in Algorithm 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 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

\[
\lambda_{[p]}^{(k)} = \left| \langle T, \bigotimes_{\ell=1}^k u_{[p]}^{(\ell)} \rangle \right| = \left| \langle T \bigotimes \beta_1 \bigotimes_{\ell=3}^k u_{[p]}^{(\ell)}, u_{[p]}^{(1)} \otimes u_{[p]}^{(2)} \rangle \right| \leq \left| \langle T \bigotimes \beta_1 \bigotimes_{\ell=3}^k u_{[p]}^{(\ell)}, \tilde{u}_{[p+1]}^{(1)} \otimes \tilde{u}_{[p+1]}^{(2)} \rangle \right|
\]

\[
= \lambda_{[p+1]}^{(1)} = \left| \langle T \bigotimes \beta_2 \bigotimes_{\ell=4}^k u_{[p+1]}^{(\ell)}, u_{[p+1]}^{(1)} \otimes u_{[p+1]}^{(2)} \rangle \right| \leq \left| \langle T \bigotimes \beta_2 \bigotimes_{\ell=4}^k u_{[p+1]}^{(\ell)}, \tilde{u}_{[p+1]}^{(1)} \otimes \tilde{u}_{[p+1]}^{(2)} \rangle \right|
\]

\[
= \lambda_{[p+1]}^{(2)} \leq \cdots
\]

\[
\leq \lambda_{[p+1]}^{(k-1)} = \left| \langle T, \tilde{u}_{[p+1]}^{(1)} \otimes \bigotimes_{\ell=2}^{k-1} u_{[p+1]}^{(\ell)} \otimes \tilde{u}_{[p+1]}^{(k)} \rangle \right| = \left| \langle T \bigotimes \beta_2 \bigotimes_{\ell=2}^{k-1} u_{[p+1]}^{(\ell)}, \tilde{u}_{[p+1]}^{(1)} \otimes \tilde{u}_{[p+1]}^{(k)} \rangle \right|
\]

\[
\leq \left| \langle T \bigotimes \beta_2 \bigotimes_{\ell=2}^{k-1} u_{[p+1]}^{(\ell)}, u_{[p+1]}^{(1)} \otimes u_{[p+1]}^{(k)} \rangle \right| = \lambda_{[p+1]}^{(k)}
\]

are always maintained. The monotone sequence \( \{\lambda_{[p]}^{(k)}\} \) 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 \( \{\lambda_{\ell}\} \) generated in Algorithm 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

\[
\lambda_t = \left| \langle T, \bigotimes_{\ell=1}^k u_{[t]}^{(\ell)} \rangle \right| = \left| \langle T \bigotimes \beta_i \bigotimes_{i=1}^{k-2} u_{[t]}^{(\sigma_i)}, u_{[t]}^{(\sigma_{i-1})} \otimes u_{[t]}^{(\sigma_k)} \rangle \right| \leq \left| \langle T \bigotimes \beta_i \bigotimes_{i=1}^{k-2} u_{[t]}^{(\sigma_i)}, u_{t+1} \otimes u_{t+1} \rangle \right| = \lambda_{t+1},
\]

since \( u_{t+1} \) is the dominant singular vector of the matrix \( T \bigotimes \beta_i \bigotimes_{i=1}^{k-2} u_{[t]}^{(\sigma_i)} \). Again, the monotone sequence \( \{\lambda_{\ell}\} \) is bounded above by \( \|T\|_F \), so it must converge.
Lemma 5.3. The scalars \( \{ \mu_t \} \) generated in Algorithm 3 form a monotone convergent sequence.

Proof. Suppose that \( C_t = T \otimes_{i=1}^{k-2} u_i^{(i)} \) has been defined in terms of vectors \( u_1^{(1)}, \ldots, u_i^{(k-2)} \) from the previous step. Suppose also that its dominant singular vector \( u_i \) has been calculated. Then

\[
\mu_t = \left| T \otimes_{i=1}^{k-2} u_i^{(i)} \right|.
\]

(5.1)

To proceed to the next step, some of these columns \( u_1^{(1)}, \ldots, u_i^{(k-2)} \) might be replaced by \( u_i \). Let \( \Theta^o \) and \( \Theta^+ \) denote subsets of \([k - 2]\) containing indices of those vectors that will not be changed and are to be updated, respectively. Depending on the current permutation \( \sigma_t \), there are three possible scenarios – the cardinality \( |\Theta^+| \) of the set \( \Theta^+ \) can be 0, 1, or 2. By Lemmas 2.1 and 2.3, we may write

\[
\mu_t = \left| T \otimes_{i \in \Theta^o} u_i^{(i)} \otimes \bigotimes_{j \in \Theta^+} u_j^{(j)} \right| = \left| T \otimes_{i \in \Theta^o} u_i^{(i)} \otimes u_i^{[\Theta^+]}, \bigotimes_{j \in \Theta^+} u_j^{(j)} \otimes u_t^{2-|\Theta^+|} \right|
\]

(5.2)

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

5.2. Convergence of Iterates. The above argument about the monotonicity of values \( \lambda_{[p]}^{(k)}, \lambda_t, \) or \( \mu_t \) 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 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 1 is given in the Appendix.

For clarity, enumerate the column vectors at the end of each \( t \)-loop by \( \{ u_i^{(1)}, \ldots, u_i^{(k)} \} \) 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 \( \sigma \) which varies in \( t \) takes place. Toward this end, we establish a sequence of results.

Since the dominant singular vector \( u_i \) are always normalized to unit length, the sequence \( \{ u_i \} \) must have a convergent subsequence. We first argue that its limit point must propagate to all elements of \( \{ u_i^{(1)}, \ldots, u_i^{(k)} \} \) in the following sense.

Lemma 5.4. If \( \{ u_i \} \) is a convergent subsequence and \( \lim_{i \to \infty} u_i = u \), then \( \lim_{i \to \infty} u_i^{(\ell)} = u \) for all \( \ell \in [k] \).

Proof. Let \( \{ \sigma_i \} \) denote the corresponding sequence of permutations used in the algorithm for generating \( \{ u_i \} \). Note that each time at least two vectors in the set \( \{ u_i^{(1)}, \ldots, u_i^{(k)} \} \) are identical with \( u_i \). Specifically, by Line 12 in the algorithm, we have \( u_i^{(\sigma_i)} = u_i^{(\sigma_i^{(1)})} = u_i \). As \( u_i \) gets close to \( u \), so do these two vectors. However, since \( \{ \sigma_i \} \) 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 \( \{ u_i^{(1)}, \ldots, u_i^{(k)} \} \) are close to \( u \) when \( i \) is large enough. \( \square \)

Lemma 5.5. For almost all symmetric tensors \( T \), the accumulation points of the sequence \( \{ u_i \} \) generated by Algorithm 3 are geometrically isolated.
Proof. Suppose that \( \varpi \) is an accumulation point. By Corollary 2.4, we may perform the following operation, and by Lemma 5.4, \( \varpi \) is a solution to the nonlinear equation
\[
T \otimes u^{k-2} = \langle T, u^k \rangle u^2.
\]
(5.3)
The equation (5.3) is a polynomial system of degree \( k + 2 \) in the unknown \( u \in \mathbb{R}^n \) 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. \( \square \)

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 \( \langle T, u^k \rangle \). The monotone behavior of \( \{\lambda_i\} \) or \( \{\mu_i\} \) 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 \( \{u_i\} \) of dominant singular vectors generated in Algorithm 3 converges.

**Proof.** Suppose that \( \{u_i\} \) is any subsequence converging to \( \varpi \). Suppose also by Lemma 5.5 that \( \varpi \) is isolated. By Lemma 5.4, the corresponding subsequences \( \{u^{(i)}_t\} \) converge to \( \varpi \) for all \( \ell \in [k] \). By construction (Line 5 in Algorithm 3), the subsequence \( \{C_t^{(i)}\} \) of matrices converges. By continuity, the subsequence \( \{u_{t+1}^{(i)}\} \) must also converge to \( \varpi \). In particular, \( \|u_{t+1}^{(i)} - u_t^{(i)}\| \to 0 \). The condition in Lemma 2.7 therefore is satisfied. It follows that the whole sequence \( \{u_t^{(i)}\} \) converges to \( \varpi \). \( \square \)

We conclude this section with a simple illustration. Given a symmetric tensor \( T \) of order 3 and an arbitrary unit vector \( u_0 \), Algorithm 3 can be cast as fixed-point iteration defined by
\[
u_{t+1} = F(u_t),
\]
(5.4)
where \( F(u) \) represents the dominant singular vector of \( T \otimes u \) (with consistent sign at the first entry). Then, by Theorem 5.6, the sequence \( \{u_t\} \) 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 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 \( \bigotimes_{i=1}^{k-2} u^{(i)} \).
- \( T_C \) = the time needed to perform the tensor multiplication \( \otimes \) 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 \( \otimes \) 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 \( T_C \).
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 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 $\beta$-product $T \odot \beta S$ is the most expensive part of the calculation for our Algorithm 3. Since the the conventional ALS method involves a similar calculation, we make a rough complexity comparison between Algorithm 3 and the conventional ALS method. In the ALS, forming $\bigotimes_{i=1}^{k-1} u^{(i)}$ requires $n^{k-1}$ entry-to-entry multiplications and there are $n$ layers to form the the inner product $T \odot \bigotimes_{i=1}^{k-1} u^{(i)}$. In total, the ALS requires $n^k + n^{k-1}$ scalar multiplications. In contrast, the SVD-based method requires $n^k + n^{k-2}$ scalar multiplication per $\beta$-product if $k > 3$, and $n^3$ if $k = 3$. We then have to take into account the number of iterations needed for reaching convergence which is difficult to estimate. The svds involved in the SVD-based method requires more overhead than the ALS method at the price of $O(n^2)$, but overall it might take fewer iterations and, hence, cost must less CPU time as is indicated in Figure 6.2.

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 $x_i \in \mathbb{R}^{10}$ and combine them into $T = \sum_{i=1}^{15} x_i^T$. As these vector $\{x_i\}$ 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 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 seems 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.

Example 4. To experiment the sensitivity of a rank-1 tensor subject to perturbations, we randomly generate six vectors $x_0, x_1, \ldots, x_5 \in \mathbb{R}^{10}$ from the identical and independent standard normal distribution. Define $T_0 = x_0^7$. This rank-1, order-7, dimension-10, and symmetric tensor $T_0$ will be fixed as our target. Define the unit tensor $B := \sum_{i=1}^{5} x_i^7 \| \sum_{i=1}^{5} x_i^7 \|_F^2$ 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 $T_\sigma$ by Algorithm 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 $T_0$ 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 linear in $\sigma$.

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 1 is somewhat too conservative for computation in practice. However, its fundamental structure is the basis of Algorithm 3. In this appendix, we prove its convergence which of mathematical interest in its own right.

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

![Convergence and Multiple Solutions](image_url)
LEMMA 8.1. There is a subsequence \( \{u^{(\ell)}_{[p_j]}\} \) generated by Algorithm 1 that converges to the same limit point for all \( \ell = [k] \).

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

For clarity, we accomplish the proof in two steps. First, we argue that the sequences \( \{u^{(\ell)}_{[p_j]}\} \) 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 \( \{\tilde{u}^{(\ell)}_{[p_j]}\} \) and \( \{u^{(\ell)}_{[p_j+1]}\} \) of \( \{C^{(\ell)}_{[p_j]}\} \) converge simultaneously for all \( \ell \in [k] \). Denote \( \lim_{j \to \infty} C^{(\ell)}_{[p_j]} = C^{(\ell)} \), \( \lim_{j \to \infty} \tilde{u}^{(1)}_{[p_j+1]} = u^{(1)}_{[p_2]} \) and \( \lim_{j \to \infty} u^{(\ell)}_{[p_j+1]} = u^{(\ell)}_{[p_2]} \) for \( \ell = 2, \ldots, k \). The subscript \( \ell \) 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 \( \ell \).

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

\[
C^{(\eta)}_{[p_j]} u^{(\eta)}_{[p_j+1]} = \left( T \otimes \beta_{\eta} \left( \tilde{u}^{(1)}_{[p_j+1]} \otimes \cdots \otimes u^{(\eta-1)}_{[p_j+1]} \otimes u^{(\eta+2)}_{[p_j]} \otimes \cdots \otimes u^{(k)}_{[p_j]} \right) \right) u^{(\eta)}_{[p_j+1]} \\
= \left( T \otimes \beta_{\eta+1} \left( \tilde{u}^{(1)}_{[p_j+1]} \otimes \cdots \otimes u^{(\eta-1)}_{[p_j+1]} \otimes u^{(\eta)}_{[p_j+1]} \otimes \cdots \otimes u^{(k)}_{[p_j]} \right) \right) u^{(\eta+2)}_{[p_j]} \\
= C^{(\eta+1)} u^{(\eta+2)}_{[p_j]}.
\] (8.1)

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

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

\[
C^{(1)}_{[p_j]} \tilde{u}^{(1)}_{[p_j+1]} = \left( T \otimes \beta_{1} \left( u^{(3)}_{[p_j]} \otimes u^{(4)}_{[p_j]} \otimes \cdots \otimes u^{(k)}_{[p_j]} \right) \right) \tilde{u}^{(1)}_{[p_j+1]} \\
= \left( T \otimes \beta_{2} \left( \tilde{u}^{(1)}_{[p_j+1]} \otimes u^{(4)}_{[p_j]} \otimes \cdots \otimes u^{(k)}_{[p_j]} \right) \right) u^{(3)}_{[p_j]} = C^{(2)}_{[p_j]} u^{(3)}_{[p_j]}.
\] (8.2)
Taking the limits at both ends of (8.2), together with \( C^{(2)} \) being nonsingular, we see that \( \lim_{j \to \infty} u_{[p_j]}^{(3)} \) exists.

Case 3. For \( \ell = 2 \), observe the relationship
\[
C^{(k)}_{[p_j-1]} u^{(k)}_{[p_j]} = \pm \lambda^{(k)}_{[p_j]} u^{(k)}_{[p_j]}
= \left( T \otimes \beta_k \left( u^{(2)}_{[p_j]} \otimes \cdots \otimes u^{(k-1)}_{[p_j]} \right) \right) u^{(k)}_{[p_j]}
= \left( T \otimes \beta_1 \left( u^{(3)}_{[p_j]} \otimes \cdots \otimes u^{(k)}_{[p_j]} \right) \right) u^{(2)}_{[p_j]} = C^{(1)}_{[p_j]} u^{(2)}_{[p_j]}. \tag{8.3}
\]

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

Now we prove that sequences converge to the same limit point. Denote \( \lim_{j \to \infty} u^{(\ell)}_{[p_j]} = u^{(\ell)}_{\infty} \) for \( \ell = 1, \ldots, k \). It thus becomes clear that
\[
C^{(\ell)} = T \otimes \beta_1 \left( u^{(1)}_{\infty} \otimes \cdots \otimes u^{(\ell-1)}_{\infty} \otimes u^{(\ell+2)}_{\infty} \otimes u^{(\ell+3)}_{\infty} \otimes \cdots \otimes u^{(k)}_{\infty} \right). \tag{8.4}
\]

Analogous to (8.1), we mention also the identity
\[
C^{(k-1)}_{[p_j]} u^{(k-1)}_{[p_j+1]} = \pm \lambda^{(k-1)}_{[p_j+1]} u^{(k-1)}_{[p_j+1]}
= \left( T \otimes \beta_{k-1} \left( u^{(1)}_{[p_j+1]} \otimes u^{(2)}_{[p_j+1]} \otimes \cdots \otimes u^{(k-2)}_{[p_j+1]} \right) \right) u^{(k-1)}_{[p_j+1]}
= \left( T \otimes \beta_{k} \left( u^{(2)}_{[p_j+1]} \otimes \cdots \otimes u^{(k-2)}_{[p_j+1]} \otimes u^{(k-1)}_{[p_j+1]} \right) \right) u^{(1)}_{[p_j+1]} = C^{(k)}_{[p_j+1]} u^{(1)}_{[p_j+1]}. \tag{8.5}
\]

It follows, by construction and continuity, that we have the relationships
\[
\pm \bar{\lambda} u^{(1)}_{\infty} = \pm \bar{\lambda} u^{(k)}_{\infty} = C^{(1)} u^{(2)}_{\infty}, \tag{8.6}
\]
\[
C^{(\ell)} u^{(\ell)}_{\infty} = \pm \bar{\lambda} u^{(\ell+2)}_{\infty}, \quad \ell = 1, \ldots, k-2, \tag{8.7}
\]
\[
C^{(k-1)} u^{(k-1)}_{\infty} = \pm \bar{\lambda} u^{(k-1)}_{\infty} = C^{(k)} u^{(1)}_{\infty}, \tag{8.8}
\]
\[
C^{(k)} u^{(k)}_{\infty} = \pm \bar{\lambda} u^{(k)}_{\infty}. \tag{8.9}
\]

By continuity again, note that \( \bar{\lambda} \) is the dominant singular value of all matrices \( C^{(\ell)} \), \( \ell = 1, \ldots, k \). Under the assumption that \( \bar{\lambda} \) is simple, the corresponding singular vector is unique up to a sign change. However, because in Lines 6 to 8 of Algorithm 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
\[
\begin{cases}
  u^{(1)}_{\infty} = u^{(1)}_1 = u^{(k)}_{\infty} = u^{(2)}_{\infty},
  \\
  u^{(\ell)}_{\infty} = u^{(\ell+2)}_{\infty}, \quad \ell = 1, \ldots, k-2,
  \\
  u^{(1)}_{\infty} = u^{(k-1)}_{\infty} = u^{(k)}_{\infty}.
\end{cases} \tag{8.10}
\]

These relationships allow us to write
\[
\bar{\lambda} = \left| \left( C^{(1)} u^{(1)}_{\infty} \right) \right| = \left| \left( T \otimes \beta_{1} \left( u^{(3)}_{\infty} \otimes u^{(4)}_{\infty} \otimes \cdots u^{(k)}_{\infty} \right), u^{(1)}_{\infty} \otimes u^{(2)}_{\infty} \right) \right|
= \left| \left( T \otimes \beta_{2} \left( u^{(1)}_{\infty} \otimes u^{(4)}_{\infty} \otimes \cdots u^{(k)}_{\infty} \right), u^{(2)}_{\infty} \otimes u^{(3)}_{\infty} \right) \right|.
\]
But we also have
\[ \tilde{\lambda} = \left\langle C^{(2)}, u_2^{(2)} \otimes u_2^{(2)} \right\rangle = \left\langle T \otimes \beta_2 \left( u_2^{(4)} \otimes \ldots \otimes u_2^{(k)} \right), u_2^{(2)} \otimes u_2^{(2)} \right\rangle. \]

By the uniqueness of the dominant singular vector, since the first entry is kept positive, we conclude that
\[ u_2^{(2)} = u_2^{(2)}. \quad (8.11) \]

Repeating this process, we can prove that \( u_2^{(\ell)} = u_2^{(\ell)} \) for \( \ell = 2, \ldots, k - 1 \). Together with (8.10), we finally prove that the sequence \( \left\{ u_2^{(\ell)} \right\} \) converges to the same limit point for all \( \ell = 1, \ldots, k \).

**Corollary 8.2.** If the sequences \( \left\{ C^{(\ell)}_{[p_j]} \right\} \) 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 \( u_2^{(\ell)} = u_2^{(\ell)} \) 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 \( \left\{ C^{(\ell)}_{[p_j]} \right\} \) to argue the simultaneously convergent subsequences \( \left\{ u_2^{(\ell)}_{[p_j]} \right\} \). We can also reverse the argument.

**Corollary 8.3.** If subsequences \( \left\{ u_2^{(\ell)}_{[p_j]} \right\} \) converge simultaneously for all \( \ell \in [k] \), then so do subsequences \( \left\{ C^{(\ell)}_{[p_j]} \right\} \) and \( \left\{ u_2^{(\ell)}_{[p_j+1]} \right\} \).

**Proof.** The simultaneous convergence of \( \left\{ u_2^{(\ell)}_{[p_j]} \right\} \) for \( \ell = 3, \ldots, k \) implies that the subsequence \( \left\{ C^{(1)}_{[p_j]} \right\} \) converges. By continuity, \( \left\{ u_2^{(1)}_{[p_j+1]} \right\} \) converges. But then by definition, \( \left\{ C^{(2)}_{[p_j]} \right\} \) converges and, thus, so does \( \left\{ u_2^{(2)}_{[p_j+1]} \right\} \). Cycling through the \( \ell \)-loop in Algorithm 1, the assertion is proved.

**Theorem 8.4.** For almost all symmetric tensors \( T \), the sequences \( \left\{ u_2^{(\ell)} \right\} \) generated in Algorithm 1 converge to the same limit point.

**Proof.** Let \( \left\{ u_2^{(\ell)}_{[p_j]} \right\} \) be any simultaneously convergent subsequences. By Corollary 8.3, the corresponding subsequences \( \left\{ C^{(1)}_{[p_j]} \right\} \) converge simultaneously. Using the same argument in the proof of Lemma 8.1, we see that subsequences \( \left\{ u_2^{(\ell)}_{[p_j]} \right\} \) and \( \left\{ u_2^{(\ell)}_{[p_j+1]} \right\} \) 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**