Received: 14 April 2021 | Revised: 24 December 2021 | Accepted: 29 December 2021

DOI: 10.1002/nla.2431

RESEARCH ARTICLE

WILEY

Accelerating alternating least squares for tensor decomposition by pairwise perturbation

Linjian Ma | **Edgar Solomonik**

Department of Computer Science, University of Illinois at Urbana Champaign, Urbana, Illinois, USA

Correspondence
Linjian Ma, Department of Computer Science, University of Illinois at Urbana Champaign, Urbana, IL 61801, USA.
Email: lma16@illinois.edu

Funding information
US NSF Advanced Cyberinfrastructure, Grant/Award Number: 1931258; National Science Foundation, Grant/Award Number: ACI-1548562

Abstract

The alternating least squares (ALS) algorithm for CP and Tucker decomposition is dominated in cost by the tensor contractions necessary to set up the quadratic optimization subproblems. We introduce a novel family of algorithms that uses perturbative corrections to the subproblems rather than recomputing the tensor contractions. This approximation is accurate when the factor matrices are changing little across iterations, which occurs when ALS approaches convergence. We provide a theoretical analysis to bound the approximation error. Our numerical experiments demonstrate that the proposed pairwise perturbation algorithms are easy to control and converge to minima that are as good as ALS. The experimental results show improvements of up to 3.1× with respect to state-of-the-art ALS approaches for various model tensor problems and real datasets.

KEYWORDS

alternating least squares, CP decomposition, tensor, Tucker decomposition

1 | INTRODUCTION

Tensor decompositions provide general techniques for approximation and modeling of high dimensional data.[1-6] They are fundamental in methods for computational chemistry,[7-9] physics,[10] and quantum information.[10,11] Tensor decompositions are performed on tensors arising both in the context of numerical-PDEs (e.g., as part of preconditinioners[12]) as well as in data-driven statistical modeling.[1,13-15] The alternating least squares (ALS) method, which is most commonly used to compute many of these tensor decompositions, has become a target for parallelization,[16,17] performance optimization,[18,19] and acceleration by randomization.[20] We propose a new algorithm, pairwise perturbation (PP), that asymptotically accelerates ALS iteration complexity for CP and Tucker decomposition by leveraging an approximation that is provably accurate for well-conditioned problems and is effective when the algorithm approaches the optimization local minima.

Each iteration of ALS is a sweep over quadratic optimization subproblems for each individual factor matrix composing the decomposition. For both CP and Tucker decomposition, computational cost of each sweep is dominated by the tensor contractions needed to setup the quadratic optimization subproblem for every factor matrix. These contractions are redone at every ALS sweep since they involve the factor matrices, all of which change after each sweep. We propose to circumvent these contractions in the scenario when the factor matrices are changing only slightly at each sweep, which is expected when ALS approaches a local minima. Our method approximates the setup of each quadratic optimization subproblem by computing perturbative corrections to the right-hand side due to the change in each factor matrix since a previous ALS sweep. To do so, pairwise perturbative operators are computed that propagate the change to each factor matrix to the subproblem needed to update each other factor matrix. Computing these operators costs slightly more than

a typical ALS sweep. These operators are then reused to *approximately* perform more ALS sweeps until the changes to the factor matrices are deemed large, at which point, regular ALS sweeps are performed. Once the updates performed in these regular sweeps are again small, the pairwise operators are recomputed. Each sweep computed approximately in this way costs asymptotically less than a regular ALS sweep.

Within CP-ALS, the computational bottleneck of each sweep involves an operation called the matricized tensor-times Khatri–Rao product (MTTKRP). Similarly, the costliest operation in the ALS-based Tucker decomposition (Tucker-ALS) method is called the tensor times matrix-chain (TTMc) product. For an order N tensor with modes of dimension s, approximated computation of ALS sweeps via PP reduces the cost of that sweep from $O\left(s^N R\right)$ to $O\left(s^2 R + sR^2\right)$ for a rank-R CP decomposition and from $O\left(s^N R\right)$ to $O\left(s^2 R^{N-1}\right)$ for a rank-R Tucker decomposition.

To quantify the accuracy of the PP algorithm, in Section 4, we provide an error analysis for both MTTKRP and TTMc operations. For both operations, we first view the ALS procedure in terms of pairwise updates, pushing updates to least-squares problems of all factor matrices as soon as any one of them is updated. This reformulation is algebraically equivalent to the original ALS procedure. If the relative change to each factor matrix since PP operators were constructed is bounded by $O(\epsilon)$, we can bound the absolute error of the way PP propagates updates in MTTKRP/TTMc calculations due to changes in any one of the other factor matrices. For order three tensors, this absolute error bound yields a relative error bound that depends on a matrix condition number. For the TTMc operation in Tucker decomposition, we derive a 2-norm relative error bound for the overall TTMc calculations (as opposed to updates thereof) of $O\left(\epsilon^2\right)$ that holds when the residual of the Tucker decomposition is somewhat less than the norm of the original tensor. We also derive a Frobenius norm error bound of $O\left(\epsilon^2 (s/R)^{N/2}\right)$ for TTMc, which only assumes that higher-order singular value decomposition (HOSVD)[21,22] is performed to initialize Tucker-ALS (which is typical). In addition, in the Appendix, we show that for the CP decomposition, if the factor matrices have changed by $O(\epsilon)$ in norm, the relative error in PP for the overall MTTKRP calculation is bounded by a term that scales with $O\left(\epsilon^2\right)$ and a tensor condition number. However, we demonstrate that in the worst case scenario, for decomposition of any large tensor, this tensor condition number can be infinite.

In order to evaluate the performance benefit of PP, in Section 5, we compare per ALS sweep and full decomposition performance using a NumPy-based[23] sequential implementation. Our microbenchmark results compare the performance of one CP-ALS sweep with different input tensor sizes. We consider the initialization sweep, in which the PP operators are calculated, as well as the approximated sweep, in which the operators are not recalculated, of the PP algorithm. These results show that the approximated PP sweeps are up to 6.3× faster than one ALS sweep with the dimension tree algorithm[18,24-29] for an order three tensor with dimension size 960, and up to 33.0× faster than one ALS sweep for an order six tensor. We then study the performance and numerical behavior of PP for the decomposition of synthetic tensors and application datasets. Our experimental results show that PP achieves fitness as high as standard ALS, and achieves speed-ups of up to 3.1× for CP decomposition and up to 1.13× for Tucker decomposition with respect to state of the art ALS algorithms.

We also evaluate the performance of PP based on a distributed-memory parallel implementation on many nodes of an Intel KNL system (Stampede2) using Cyclops Tensor Framework[30] and ScaLAPACK[31] libraries. Our experimental results show that PP achieves fitness as high as standard ALS, and achieves speed-ups of up to 1.75× with respect to a standard ALS implementation on top of the Cyclops library on Stampede2.

2 | BACKGROUND

This section first outlines the notation used throughout this article, then outlines the basic alternating least square algorithms for both CP and Tucker decomposition.

2.1 | Notation and definitions

Our analysis makes use of tensor algebra in both element-wise equations and specialized notation for tensor operations.[1] For vectors, bold lowercase Roman letters are used, for example, $\mathbf{x}$. For matrices, bold uppercase Roman letters are used, for example, $\mathbf{X}$. For tensors, bold calligraphic fonts are used, for example, $\boldsymbol{\mathcal{X}}$. An order N tensor corresponds to an N-dimensional array with dimensions $s_1 \times \cdots \times s_N$. Elements of vectors, matrices, and tensors are denotes in parentheses, for example, $\mathbf{x}(i)$ for a vector $\mathbf{x}$, $\mathbf{X}(i,j)$ for a matrix $\mathbf{X}$, and $\boldsymbol{\mathcal{X}}(i,j,k,l)$ for an order 4 tensor $\boldsymbol{\mathcal{X}}$. Columns of a matrix $\mathbf{X}$ are denoted by $\mathbf{x}_i = \mathbf{X}(:,i)$. The mode-$n$ matrix product of an order N tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$ with a matrix $\mathbf{A} \in \mathbb{R}^{J \times s_n}$

is denoted by $\mathcal{X} \times_n \mathbf{A}$, with the result having dimensions $s_1 \times \cdots \times s_{n-1} \times J \times s_{n+1} \times \cdots \times s_N$. The mode-$n$ vector product of $\mathcal{X}$ with a vector $\mathbf{v} \in \mathbb{R}^{s_n}$ is denoted by $\mathcal{X} \times_n \mathbf{v}^T$, with the result having dimensions $s_1 \times \cdots \times s_{n-1} \times s_{n+1} \times \cdots \times s_N$. Matricization is the process of unfolding a tensor into a matrix. Given a tensor $\mathcal{X}$ the mode-n matricized version is denoted by $\mathbf{X}_{(n)} \in \mathbb{R}^{s_n \times K}$ where $K = \prod_{m=1, m\neq n}^{N} s_m$. We generalize this notation to define the unfoldings of a tensor $\mathcal{X}$ with dimensions $s_1 \times \cdots \times s_N$ into an order $M+1$ tensor, $\mathcal{X}_{(i_1,\ldots,i_M)} \in \mathbb{R}^{s_{i_1} \times \cdots \times s_{i_M} \times K}$, where $K = \prod_{i \in \{1,\ldots,N\} \setminus \{i_1,\ldots,i_M\}} s_i$, for example, $\mathcal{X}(j,k,l,m) = \mathcal{X}_{(1,3)}(j,l,k+(m-1)s_2)$. We use parenthesized superscripts as labels for different tensors, for example, $\mathcal{X}^{(1)}$ and $\mathcal{X}^{(2)}$ are generally unrelated tensors.

The Hadamard product of two matrices $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{I \times J}$ resulting in matrix $\mathbf{W} \in \mathbb{R}^{I \times J}$ is denoted by $\mathbf{W} = \mathbf{U} * \mathbf{V}$, where $\mathbf{W}(i,j) = \mathbf{U}(i,j)\mathbf{V}(i,j)$. We use $*$ to denote a chain of Hadamard products, for example, $*_{i=1}^{n} \mathbf{A}^{(i)} = \mathbf{A}^{(1)} * \cdots * \mathbf{A}^{(n)}$. The outer product of K vectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(K)}$ of corresponding sizes $s_1, \ldots, s_K$ is denoted by $\mathcal{X} = \mathbf{u}^{(1)} \circ \cdots \circ \mathbf{u}^{(K)}$ where $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_K}$ is an order K tensor. The Kronecker product of vectors $\mathbf{u} \in \mathbb{R}^I$ and $\mathbf{v} \in \mathbb{R}^J$ is denoted by $\mathbf{w} = \mathbf{u} \otimes \mathbf{v}$ where $\mathbf{w} \in \mathbb{R}^{IJ}$. For matrices $\mathbf{A} \in \mathbb{R}^{I \times K}$ and $\mathbf{B} \in \mathbb{R}^{J \times K}$, their Khatri–Rao product results in a matrix of size $(IJ) \times K$ defined by $\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \ldots, \mathbf{a}_K \otimes \mathbf{b}_K]$. We use $\odot$ to denote a chain of Khatri–Rao products, for example, $\odot_{i=1}^{n} \mathbf{A}^{(i)} = \mathbf{A}^{(1)} \odot \cdots \odot \mathbf{A}^{(n)}$.

2.2 | CP decomposition with ALS

The CP tensor decomposition[32,33] is a higher-order generalization of the matrix singular value decomposition (SVD). The CP decomposition is denoted by

$$\mathcal{X} \approx \left[\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\right], \quad \text{where} \quad \mathbf{A}^{(i)} = \left[\mathbf{a}_1^{(i)}, \ldots, \mathbf{a}_R^{(i)}\right],$$

and serves to approximate a tensor by a sum of R tensor products of vectors,

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(N)}.$$

The CP-ALS method alternates among quadratic optimization problems for each of the factor matrices $\mathbf{A}^{(n)}$, resulting in linear least squares problems for each row,

$$\mathbf{A}_{\text{new}}^{(n)} \mathbf{P}^{(n)T} \cong \mathbf{X}_{(n)},$$

where the matrix $\mathbf{P}^{(n)} \in \mathbb{R}^{I_n \times R}$, where $I_n = s_1 \times \cdots \times s_{n-1} \times s_{n+1} \times \cdots \times s_N$, is formed by Khatri–Rao products of the other factor matrices,

$$\mathbf{P}^{(n)} = \mathbf{A}^{(1)} \odot \cdots \odot \mathbf{A}^{(n-1)} \odot \mathbf{A}^{(n+1)} \odot \cdots \odot \mathbf{A}^{(N)}.$$

These linear least squares problems are often solved via the normal equations.[1] We also adopt this strategy here to devise the PP method. The normal equations for the nth factor matrix are

$$\mathbf{A}_{\text{new}}^{(n)} \mathbf{\Gamma}^{(n)} = \mathbf{X}_{(n)} \mathbf{P}^{(n)},$$

where $\mathbf{\Gamma} \in \mathbb{R}^{R \times R}$ can be computed via

$$\mathbf{\Gamma}^{(n)} = \mathbf{S}^{(1)} * \cdots * \mathbf{S}^{(n-1)} * \mathbf{S}^{(n+1)} * \cdots * \mathbf{S}^{(N)}, \quad \text{with each} \quad \mathbf{S}^{(i)} = \mathbf{A}^{(i)T} \mathbf{A}^{(i)}.$$

These equations also give the nth component of the optimality conditions for the unconstrained minimization of the nonlinear objective function,

$$f\left(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right) = \frac{1}{2} \left\| \mathcal{X} - \left[\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\right] \right\|_F^2,$$

for which the nth component of the gradient is

$$\frac{\partial f}{\partial \mathbf{A}^{(n)}} = \mathbf{G}^{(n)} = \mathbf{A}^{(n)}\mathbf{\Gamma}^{(n)} - \mathbf{X}_{(n)}\mathbf{P}^{(n)} = \left(\mathbf{A}^{(n)} - \mathbf{A}^{(n)}_{\text{new}}\right)\mathbf{\Gamma}^{(n)}.$$

Algorithm 1 presents the basic ALS method described above, keeping track of the Frobenius norm of the N components of the overall gradient to ascertain convergence.

Algorithm 1. CP-ALS: ALS procedure for CP decomposition

Input: Tensor $\mathcal{A} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$,stopping criteria Δ
Initialize $\left[\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\right]$ as uniformly distributed random matrices within $[0, 1]$, initialize $\mathbf{G}^{(n)} \leftarrow \mathbf{A}^{(n)}$, $\mathbf{S}^{(n)} \leftarrow \mathbf{A}^{(n)T}\mathbf{A}^{(n)}$ for $n \in \{1, \ldots, N\}$
while $\sum_{i=1}^{N} \left\|\mathbf{G}^{(i)}\right\|_F > \Delta \|\mathcal{A}\|_F$ **do**
  for $n \in \{1, \ldots, N\}$ **do**
    $\mathbf{\Gamma}^{(n)} \leftarrow \mathbf{S}^{(1)} * \cdots * \mathbf{S}^{(n-1)} * \mathbf{S}^{(n+1)} * \cdots * \mathbf{S}^{(N)}$
    Update $\mathbf{M}^{(n)}$ based on the dimension tree algorithm shown in Figure 1
    $\mathbf{A}^{(n)}_{\text{new}} \leftarrow \mathbf{M}^{(n)}\mathbf{\Gamma}^{(n)\dagger}$
    $\mathbf{G}^{(n)} \leftarrow \left(\mathbf{A}^{(n)} - \mathbf{A}^{(n)}_{\text{new}}\right)\mathbf{fb}^{(n)}$
    $\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)}_{\text{new}}$
    $\mathbf{S}^{(n)} \leftarrow \mathbf{A}^{(n)T}\mathbf{A}^{(n)}$
  end for
end whilereturn $\left[\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\right]$

The MTTKRP computation, $\mathbf{M}^{(n)} = \mathbf{X}_{(n)}\mathbf{P}^{(n)}$, is the main computational bottleneck of CP-ALS.[34] The computational cost of MTTKRP is $\Theta(s^N R)$ if $s_n = s$ for all $n \in \{1, \ldots, N\}$. With the dimension tree algorithm, which will be detailed in Section 2.4, the computational complexity for all the MTTKRP calculations in one ALS sweep is $4s^N R$ to leading order in s. The normal equations worsen the conditioning, but are advantageous for CP-ALS, since $\mathbf{\Gamma}^{(n)}$ can be computed and inverted in just $O(s^2R + R^3)$ cost and the MTTKRP can be amortized by dimension trees. If QR is used instead of the normal equations, the product of $\mathbf{Q}$ with the right-hand sides would have the cost $2s^N R$ and would need to be done for each linear least squares problem, increasing the overall leading order cost by a factor of $N/2$.

2.3 | Tucker decomposition with ALS

In this section, we review the ALS method for computing a low-rank Tucker decomposition of a tensor.[22] Tucker decomposition approximates a tensor by a core tensor contracted by matrices with orthonormal columns along each mode. The Tucker decomposition is given by

$$\mathcal{A} \approx \left[\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\right] = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)}.$$

The corresponding element-wise expression is

$$\mathcal{A}(x_1, \ldots, x_N) \approx \sum_{\{z_1, \ldots, z_N\}} \mathcal{G}(z_1, \ldots, z_N) \prod_{r \in \{1, \ldots, N\}} \mathbf{A}^{(r)}(x_r, z_r).$$

The core tensor $\mathcal{G}$ is of order N with dimensions (Tucker ranks) $R_1 \times \cdots \times R_N$ (throughout error and cost analysis we assume each $R_n = R$ for $n \in \{1, \ldots, N\}$). The matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{s_n \times R_n}$ have orthonormal columns.

The HOSVD[21,22] computes the leading left singular vectors of each one-mode unfolding of $\mathcal{A}$, providing a good starting point for the Tucker-ALS algorithm. The classical HOSVD computes the truncated SVD of $\mathbf{X}_{(n)} \approx \mathbf{U}^{(n)}\mathbf{\Sigma}^{(n)}\mathbf{V}^{(n)T}$ and sets

$\mathbf{A}^{(n)} = \mathbf{U}^{(n)}$ for $n \in \{1, \dots, N\}$. The interlaced HOSVD[35,36] instead computes the truncated SVD of

$$\mathbf{Z}^{(n)}_{(n)} = \mathbf{U}^{(n)}\mathbf{\Sigma}^{(n)}\mathbf{V}^{(n)T}, \quad \text{where} \quad \mathcal{Z}^{(1)} = \mathcal{X} \quad \text{and} \quad \mathbf{Z}^{(n+1)}_{(n)} = \mathbf{\Sigma}^{(n)}\mathbf{V}^{(n)T}.$$

The interlaced HOSVD is cheaper, since the size of each $\mathcal{Z}^{(n)}$ is $s^{N-n+1}R^{n-1}$.

The ALS method for Tucker decomposition,[1,37,38] which is also called the higher-order orthogonal iteration (HOOI), then proceeds by fixing all except one factor matrix, and computing a low-rank matrix factorization to update that factor matrix and the core tensor. To update the nth factor matrix, Tucker-ALS factorizes

$$\mathcal{Y}^{(n)} = \mathcal{X} \times_1 \mathbf{A}^{(1)T} \cdots \times_{n-1} \mathbf{A}^{(n-1)T} \times_{n+1} \mathbf{A}^{(n+1)T} \cdots \times_N \mathbf{A}^{(N)T},$$

which is called the TTMc, into a product of an matrix with orthonormal columns $\mathbf{A}^{(n)}$ and the core tensor $\mathcal{G}$, so that $\mathbf{Y}^{(n)}_{(n)} \approx \mathbf{A}^{(n)}\mathbf{G}_{(n)}$. This factorization can be done by taking $\mathbf{A}^{(n)}$ to be the R_n leading left singular vectors of $\mathbf{Y}^{(n)}_{(n)}$. This Tucker-ALS procedure is given in Algorithm 2.

Algorithm 2. Tucker-ALS: ALS procedure for Tucker decomposition

Input: Tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$,decomposition ranks $\{R_1, \dots, R_N\}$,stopping criteria Δ
Initialize $[\![\mathcal{G}; \mathbf{A}^{(1)}, \dots, \mathbf{A}^{(N)}]\!]$ using HOSVD, initialize $\mathcal{F} \leftarrow \mathcal{G}$
while $\|\mathcal{F}\|_F > \Delta\|\mathcal{X}\|_F$ **do**
  for $n \in \{1, \dots, N\}$ **do**
    Update $\mathcal{Y}^{(n)}$ based on the dimension tree algorithm
    $\mathbf{A}^{(n)} \leftarrow R_n$ leading left singular vectors of $\mathbf{Y}^{(n)}_{(n)}$
  end for
  $\mathcal{G}_{\text{new}} \leftarrow \mathcal{Y}^{(N)} \times_N \mathbf{A}^{(N)T}$
  $\mathcal{F} \leftarrow \mathcal{G}_{\text{new}} - \mathcal{G}$
  $\mathcal{G} \leftarrow \mathcal{G}_{\text{new}}$
end whilereturn $[\![\mathcal{G}; \mathbf{A}^{(1)}, \dots, \mathbf{A}^{(N)}]\!]$

As in previous work,[39,40] our implementation computes these singular vectors by finding the left eigenvectors of the Gram matrix $\mathbf{W} = \mathbf{Y}^{(n)}_{(n)}\mathbf{Y}^{(n)T}_{(n)}$. Computing the Gram matrix sacrifices some numerical stability, but avoids a large SVD and provides consistency of the signs of the singular vectors across ALS sweeps.

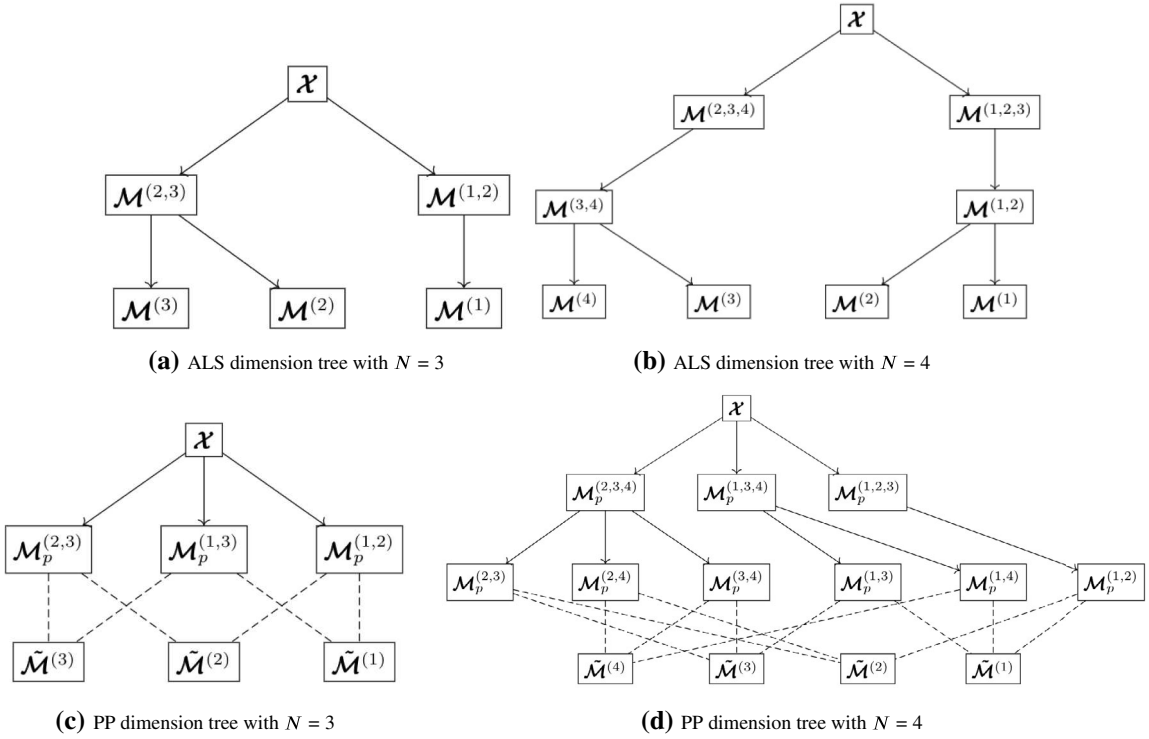
2.4 | The dimension tree algorithm

For CP-ALS, the tensor contractions for MTTKRP can be amortized across the linear least squares problems necessary for a given ALS sweep (for loop iteration in Algorithm 1). Such amortization techniques are referred to as dimension tree algorithms and a variety of dimension trees have been studied to minimize costs.[18,24-29] As our analysis focuses on leading order cost in s, simple binary dimension trees are an optimal choice. These dimension trees for $N = 3, 4$ are illustrated in Figure 1a,b. We define the partially contracted MTTKRP intermediates $\mathcal{M}^{(i_1,i_2,\dots,i_m)}$ therein as follows,

$$\mathcal{M}^{(i_1,i_2,\dots,i_m)} = \mathcal{X}_{(i_1,i_2,\dots,i_m)} \bigodot_{j \in \{1,\dots,N\} \setminus \{i_1,i_2,\dots,i_m\}} \mathbf{A}^{(j)}. \tag{1}$$

Elementwise,

$$\mathcal{M}^{(i_1,i_2,\dots,i_m)}(x_{i_1}, x_{i_2}, \dots, x_{i_m}, k) = \sum_{\{x_1,\dots,x_N\} \setminus \{x_{i_1},x_{i_2},\dots,x_{i_m}\}} \mathcal{X}(x_1, \dots, x_N) \prod_{r \in \{1,\dots,N\} \setminus \{i_1,i_2,\dots,i_m\}} \mathbf{A}^{(r)}(x_r, k),$$



(a) ALS dimension tree with $N = 3$ **(b)** ALS dimension tree with $N = 4$

(c) PP dimension tree with $N = 3$ **(d)** PP dimension tree with $N = 4$

FIGURE 1 Dimension trees for ALS and pairwise perturbation. In (c, d), the solid arrows denote the data dependencies in building pairwise perturbation operators, and is calculated in the PP initialization step. The dashed lines denote the data dependencies in the PP approximated step calculations

where $\mathcal{M}^{(1,\ldots,N)}$ is the input tensor $\mathcal{X}$. The first level contractions (contractions between the input tensor and one factor matrix) can be done via matrix multiplications between the reshaped input tensor and the factor matrix. These contractions have a cost of $O\left(s^N R\right)$ and are generally the most time-consuming part of ALS. Other contractions (transforming one intermediate into another intermediate) can be done via batched matrix-vector products, and the complexity of an ith level contraction is $O\left(s^{N+1-i}R\right)$. Because two first level contractions are necessary for the construction of tree dimension tree, as is illustrated in Figure 1a,b, to calculate all the $\mathbf{M}^{(n)}$ in one ALS sweep, to leading order in s, the computational complexity is $4s^N R$.

For Tucker-ALS, the TTMc that computes each $\mathcal{Y}^{(n)}$ is the main computational bottleneck of Tucker-ALS[41] and can also be amortized by the dimension tree. The intermediates for Tucker dimension tree are the partially contracted TTMc, $\mathcal{Y}^{(i_1,i_2,\ldots,i_m)}$, defined as follows,

$$\mathcal{Y}^{(i_1,i_2,\ldots,i_m)} = \mathcal{X} \bigtimes_{j\in\{1,\ldots,N\}\setminus\{i_1,i_2,\ldots,i_m\}} \mathbf{A}^{(j)T},$$

where $\mathcal{X}$ is contracted with all the matrices $\mathbf{A}^{(j)}$ except $\mathbf{A}^{(i_1)}, \ldots, \mathbf{A}^{(i_m)}$. Each contraction can be done via matrix multiplications, and the complexity of an ith level contraction is $O\left(s^{N+1-i}R^i\right)$. Similar to CP-ALS, to calculate all the $\mathcal{Y}^{(n)}$ in one ALS sweep, to leading order in s, the computational complexity is $4s^N R$.

3 | PP ALGORITHMS

We now introduce a PP algorithm to accelerate the ALS procedure when the iterative optimization steps are approaching a local minimum. We first derive the approximation for order three tensors, then generalize the algorithm to order N tensors. The key idea of the PP method is to compute *PP operators*, which correlate a pair of factor matrices. These tensors are then used to repeatedly update the quadratic subproblems for each tensor. As we will show, these updates are

provably accurate if the factor matrices do not change significantly since their state at the time of formation of the PP operators.

3.1 | PP for order three tensors

3.1.1 | CP-ALS

The PP procedure for CP-ALS approximates the MTTKRP outputs. Consider an order three equi-dimensional tensor with size in each mode s and CP rank R, the first mode MTTKRP can be expressed as $\mathbf{M}^{(1)} = \mathbf{X}_{(1)}\left(\mathbf{A}^{(2)} \odot \mathbf{A}^{(3)}\right)$. Let $\mathbf{A}_p^{(n)}$ denote the $\mathbf{A}^{(n)}$ calculated with regular ALS at some number of sweeps prior to the current one. Then $\mathbf{A}^{(n)}$ at the current sweep can be expressed as

$$\mathbf{A}^{(n)} = \mathbf{A}_p^{(n)} + d\mathbf{A}^{(n)},$$

and $\mathbf{M}^{(1)}$ can be expressed as

$$\mathbf{M}^{(1)} = \underbrace{\mathbf{X}_{(1)}\left(\mathbf{A}_p^{(2)} \odot \mathbf{A}_p^{(3)}\right) + \mathbf{X}_{(1)}\left(\mathbf{A}_p^{(2)} \odot d\mathbf{A}^{(3)}\right) + \mathbf{X}_{(1)}\left(d\mathbf{A}^{(2)} \odot \mathbf{A}_p^{(3)}\right)}_{\mathbf{U}^{(1)}} + \mathbf{X}_{(1)}\left(d\mathbf{A}^{(2)} \odot d\mathbf{A}^{(3)}\right). \tag{2}$$

The PP procedure for CP-ALS approximates $\mathbf{M}^{(1)}$ with $\tilde{\mathbf{M}}^{(1)} = \mathbf{U}^{(1)} + \mathbf{V}^{(1)}$, where $\mathbf{U}^{(1)}$ is the first three terms in Equation (2) and $\mathbf{V}^{(1)}$ approximates the final term through approximating the input tensor $\mathcal{X}$ by its approximate CP decomposition,

$$\mathbf{X}_{(1)}\left(d\mathbf{A}^{(2)} \odot d\mathbf{A}^{(3)}\right) \approx \mathbf{V}^{(1)} = \left(\left[\left[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}\right]\right]\right)_{(1)}\left(d\mathbf{A}^{(2)} \odot d\mathbf{A}^{(3)}\right) = \mathbf{A}^{(1)}\left(\left(\mathbf{A}^{(2)T} d\mathbf{A}^{(2)}\right) * \left(\mathbf{A}^{(3)T} d\mathbf{A}^{(3)}\right)\right),$$

which can be calculated with the cost of $O\left(sR^2\right)$. The remaining error term is

$$\left(\mathcal{X} - \left[\left[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}\right]\right]\right)_{(1)}\left(d\mathbf{A}^{(2)} \odot d\mathbf{A}^{(3)}\right).$$

Therefore, the norm of the error scales as $O\left(C\epsilon^2\right)$ if each $\left\|d\mathbf{A}^{(i)}\right\|_2 \leq \epsilon$ and the decomposition residual norm is bounded by C.

The approximated MTTKRP, $\tilde{\mathbf{M}}^{(1)}$, can be rewritten as a function of $\mathcal{M}_p^{(i_1,i_2,\ldots,i_m)}$, which is defined in the same way as $\mathcal{M}^{(i_1,i_2,\ldots,i_m)}$ in Equation (1) except that $\mathcal{X}$ is contracted with $\mathbf{A}_p^{(j)}$ for $j \in \{1,\ldots,N\} \setminus \{i_1, i_2, \ldots, i_m\}$, thus $\mathbf{M}_p^{(1)} = \mathcal{X}_{(1)}(\mathbf{A}_p^{(2)} \odot \mathbf{A}_p^{(3)})$, $\mathbf{M}_p^{(1,2)} = \mathcal{X}_{(1,2)}\mathbf{A}_p^{(3)}$, $\mathbf{M}_p^{(1,3)} = \mathcal{X}_{(1,3)}\mathbf{A}_p^{(2)}$. For each $x \in \{1,\ldots,s\}$ and $k \in \{1,\ldots,R\}$,

$$\tilde{\mathbf{M}}^{(1)}(x,k) = \mathbf{M}_p^{(1)}(x,k) + \sum_{y=1}^{s} \mathcal{M}_p^{(1,2)}(x,y,k) d\mathbf{A}^{(2)}(y,k) + \sum_{y=1}^{s} \mathcal{M}_p^{(1,3)}(x,y,k) d\mathbf{A}^{(3)}(y,k) + \mathbf{V}^{(1)}(x,k).$$

PP has two steps: the initialization step, where the terms $\mathbf{M}_p^{(1)}$ and PP operators $\mathcal{M}_p^{(1,2)}$, $\mathcal{M}_p^{(1,3)}$ are calculated, and the approximated step, where these terms are used in the equation above to calculate $\tilde{\mathbf{M}}^{(1)}$. Using the dimension tree structure shown in Figure 1c, the initialization step for all the three modes can be done with the leading order cost of $6s^3R$, 1.5× the cost of the ALS dimension tree. Each approximated step for all the modes can be done with the leading order cost of $3\left(4s^2R + 6sR^2\right)$ overall.

3.1.2 | Tucker-ALS

We derive a similar PP algorithm for order three Tucker-ALS. The first mode of TTMc can be expressed as $\mathcal{Y}^{(1)} = \mathcal{Y} \times_2 \mathbf{A}^{(2)T} \times_3 \mathbf{A}^{(3)T}$. PP approximates $\mathcal{Y}^{(1)}$ with

$$\tilde{\mathcal{Y}}^{(1)} = \mathcal{X} \times_2 \mathbf{A}_p^{(2)T} \times_3 \mathbf{A}_p^{(3)T} + \mathcal{X} \times_2 \mathbf{A}_p^{(2)T} \times_3 d\mathbf{A}^{(3)T} + \mathcal{X} \times_2 d\mathbf{A}^{(2)T} \times_3 \mathbf{A}_p^{(3)T},$$

and the error term is $\mathcal{X} \times_2 d\mathbf{A}^{(2)T} \times_3 d\mathbf{A}^{(3)T}$. The expression above can be rewritten as a function of $\mathcal{Y}_p^{(i_1,i_2,\ldots,i_m)}$, which is defined in the same way as $\mathcal{Y}^{(i_1,i_2,\ldots,i_m)}$ except that $\mathcal{X}$ is contracted with $\mathbf{A}_p^{(j)}$ for $\mathcal{Y}_p^{(i_1,i_2,\ldots,i_m)}$,

$$\tilde{\mathcal{Y}}^{(1)} = \mathcal{Y}_p^{(1)} + \mathcal{Y}_p^{(1,2)} \times_2 d\mathbf{A}^{(2)T} + \mathcal{Y}_p^{(1,3)} \times_3 d\mathbf{A}^{(3)T}.$$

Using the dimension tree structure, the initialization step for all the three modes can be done with the leading order cost of $6s^3R$, 1.5× the cost of the ALS dimension tree. Each approximated step for all the modes can be done with the leading order cost of $12s^2R^2$ overall.

3.2 | General PP algorithm

We now generalize PP to order N tensors.

3.2.1 | CP-ALS

The MTTKRP in nth mode, $\mathbf{M}^{(n)}$, can be expressed as

$$\mathbf{M}^{(n)} = \mathbf{X}_{(n)} \bigodot_{i=1,i\neq n}^{N} \left(\mathbf{A}_p^{(i)} + d\mathbf{A}^{(i)}\right).$$

$\mathbf{M}^{(n)}$ can be expressed as a function of $\mathcal{M}_p^{(i_1,i_2,\ldots,i_m)}$ as follows,

$$\begin{aligned}\mathbf{M}^{(n)}(y,k) = \mathbf{M}_p^{(n)}(y,k) &+ \sum_{i=1,i\neq n}^{N} \sum_{x=1}^{s_i} \mathcal{M}_p^{(i,n)}(x,y,k) d\mathbf{A}^{(i)}(x,k) \\ &+ \sum_{i=1,i\neq n}^{N} \sum_{j=i+1,j\neq n}^{N} \sum_{x=1}^{s_i} \sum_{z=1}^{s_j} \mathcal{M}_p^{(i,j,n)}(x,z,y,k) d\mathbf{A}^{(i)}(x,k) d\mathbf{A}^{(j)}(z,k) + \cdots .\end{aligned}$$

From the above expression, we observe that, except the first two terms, all terms include the contraction between tensor $\mathcal{M}_p^{(i_1,i_2,\ldots,i_m)}$ and at least two matrices $d\mathbf{A}^{(i)}$, so that their norm scales quadratically with the norm of the perturbative updates $d\mathbf{A}^{(i)}$. Therefore, their norm scales as $O\left(\epsilon^2\right)$ if $\left\|d\mathbf{A}^{(i)}\right\|_2 \leq \epsilon$. The PP algorithm obtains an effective approximation by keeping the first two terms (these terms are illustrated in Figure 1d for an order four tensor), and approximating the input tensor using its approximate CP decomposition in the third term to lower the error to a greater extent. For each $y \in \{1,\ldots,s_n\}$ and $k \in \{1,\ldots,R\}$,

$$\tilde{\mathbf{M}}^{(n)}(y,k) = \mathbf{M}_p^{(n)}(y,k) + \sum_{i=1,i\neq n}^{N} \sum_{x=1}^{s_i} \mathcal{M}_p^{(i,n)}(x,y,k) d\mathbf{A}^{(i)}(x,k) + \sum_{i,j=1,i,j\neq n,i\neq j}^{N} \mathbf{V}^{(n,i,j)}(y,k), \tag{3}$$

$$\text{where} \quad \mathbf{M}_p^{(n)} = \mathbf{X}_{(n)} \bigodot_{i=1,i\neq n}^{N} \mathbf{A}_p^{(i)}, \quad \mathcal{M}_p^{(i,n)} = \mathcal{X}_{(i,n)} \bigodot_{j\in\{1,\ldots,N\}\setminus\{i,n\}}^{N} \mathbf{A}_p^{(j)},$$

$$\text{and} \quad \mathbf{V}^{(n,i,j)} = \mathbf{A}^{(n)} \left(\left(\mathbf{A}^{(i)T} d\mathbf{A}^{(i)}\right) * \left(\mathbf{A}^{(j)T} d\mathbf{A}^{(j)}\right) * \overset{N}{\underset{k=1,k\neq i,j,n}{*}} \left(\mathbf{A}^{(k)T} \mathbf{A}^{(k)}\right) \right).$$

We evaluate the benefit of including the $\mathbf{V}^{(n,i,j)}$ correction in Section 5.1. Given $\mathcal{M}_p^{(i,n)}$ and $\mathbf{M}_p^{(n)}$, calculation of $\tilde{\mathbf{M}}^{(n)}$ for $n \in \{1,\ldots,N\}$ requires $2N^2\left(s^2R + sR^2\right)$ operations overall. Further, we show in Section 4.1 that the column-wise relative approximation error of $\tilde{\mathbf{M}}^{(n)}$ with respect to $\mathbf{M}^{(n)}$ is small if each $\left\|d\mathbf{a}_k^{(n)}\right\|_2 / \left\|\mathbf{a}_k^{(n)}\right\|_2$ for $n \in \{1,\ldots,N\}, k \in \{1,\ldots,R\}$ is sufficiently small. Algorithm 3 presents the PP-CP-ALS method described above.

Algorithm 3. PP-CP-ALS: Pairwise perturbation procedure for CP-ALS

Input: tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$,stopping criteria Δ, PP tolerance $\epsilon < 1$
Initialize $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ as uniformly distributed random matrices within $[0, 1]$, initialize $\mathbf{G}^{(n)}, d\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)}, \mathbf{S}^{(n)} \leftarrow \mathbf{A}^{(n)T}\mathbf{A}^{(n)}$ for $i \in \{1, \ldots, N\}$
while $\sum_{i=1}^{N} \left\|\mathbf{G}^{(i)}\right\|_F > \Delta \|\mathcal{X}\|_F$ **do**
  if $\forall\, i \in \{1, \ldots, N\}, \left\|d\mathbf{A}^{(i)}\right\|_F < \epsilon \left\|\mathbf{A}^{(i)}\right\|_F$ **then**
    Compute $\mathcal{M}_p^{(i,n)}, \mathbf{M}_p^{(n)}$ for $i, n \in \{1, \ldots, N\}$ via dimension tree in Section 3.2.3
    for $n \in \{1, \ldots, N\}$ **do**
      $\mathbf{A}_p^{(n)} \leftarrow \mathbf{A}^{(n)}, d\mathbf{A}^{(n)} \leftarrow \mathbf{O}$
    end for
    while $\sum_{i=1}^{N} \left\|\mathbf{G}^{(i)}\right\|_F > \Delta \|\mathcal{X}\|_F$ and $\forall\, i \in \{1, \ldots, N\}, \left\|d\mathbf{A}^{(i)}\right\|_F < \epsilon \left\|\mathbf{A}^{(i)}\right\|_F$ **do**
      for $n \in \{1, \ldots, N\}$ **do**
        $\mathbf{\Gamma}^{(n)} \leftarrow \mathbf{S}^{(1)} * \cdots * \mathbf{S}^{(n-1)} * \mathbf{S}^{(n+1)} * \cdots * \mathbf{S}^{(N)}$
        Update $\tilde{\mathbf{M}}^{(n)}$ based on Equation (3)
        $\mathbf{A}_{\text{new}}^{(n)} \leftarrow \tilde{\mathbf{M}}^{(n)} \mathbf{\Gamma}^{(n)\dagger}$
        $\mathbf{G}^{(n)} \leftarrow \left(\mathbf{A}^{(n)} - \mathbf{A}_{\text{new}}^{(n)}\right) \mathbf{\Gamma}^{(n)}$
        $\mathbf{A}^{(n)} \leftarrow \mathbf{A}_{\text{new}}^{(n)}$
        $\mathbf{S}^{(n)} \leftarrow \mathbf{A}^{(n)T}\mathbf{A}^{(n)}$
        $d\mathbf{A}^{(n)} = \mathbf{A}_{\text{new}}^{(n)} - \mathbf{A}_p^{(n)}$
      end for
    end while
  end if
  Perform regular ALS sweep as in Algorithm 1, taking $d\mathbf{A}^{(n)} \leftarrow \mathbf{A}_{\text{new}}^{(n)} - \mathbf{A}^{(n)}$ for each $n \in \{1, \ldots, N\}$
end whilereturn $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$

3.2.2 | Tucker-ALS

We derive a similar PP algorithm for Tucker-ALS. Similar to the expression for $\mathbf{M}^{(n)}$ in CP-ALS, $\mathcal{Y}^{(n)}$ can be expressed as

$$\mathcal{Y}^{(n)} = \mathcal{X} \bigtimes_{i=1, i\neq n}^{N} \left(\mathbf{A}_p^{(i)T} + d\mathbf{A}^{(i)T}\right).$$

The expression above can be rewritten as a function of $\mathcal{Y}_p^{(i_1, i_2, \ldots, i_m)}$,

$$\mathcal{Y}^{(n)} = \mathcal{Y}_p^{(n)} + \sum_{i=1, i\neq n}^{N} \mathcal{Y}_p^{(i,n)} \times_i d\mathbf{A}^{(i)T} + \sum_{i=1, i\neq n}^{N} \sum_{j=i+1, j\neq n}^{N} \mathcal{Y}_p^{(i,j,n)} \times_i d\mathbf{A}^{(i)T} \times_j d\mathbf{A}^{(j)T} + \cdots.$$

The PP algorithm again takes only the first order terms in $d\mathbf{A}^{(i)}$, computing

$$\tilde{\mathcal{Y}}^{(n)} = \mathcal{Y}_p^{(n)} + \sum_{i=1, i\neq n}^{N} \mathcal{Y}_p^{(i,n)} \times_i d\mathbf{A}^{(i)T}, \quad \text{where} \quad \mathcal{Y}_p^{(n)} = \mathcal{X} \bigtimes_{l=1, l\neq n}^{N} \mathbf{A}_p^{(l)T} \quad \text{and} \quad \mathcal{Y}_p^{(i,n)} = \mathcal{X} \bigtimes_{j \in \{1,\ldots,N\} \setminus \{i,n\}} \mathbf{A}_p^{(j)T}.$$

Given $\mathcal{Y}_p^{(i,n)}$ and $\mathcal{Y}_p^{(n)}$, $\tilde{\mathcal{Y}}^{(n)}$ for $n \in \{1, \ldots, N\}$ can be calculated with $2N^2 s^2 R^{N-1}$ cost overall. In Section 4.2, we show that the relative Frobenius norm approximation error of $\tilde{\mathcal{Y}}^{(n)}$ with respect to $\mathcal{Y}^{(n)}$ is small, so long as each $\left\|d\mathbf{A}^{(n)}\right\|_F / \left\|\mathbf{A}^{(n)}\right\|_F$ is sufficiently small. Algorithm 4 presents the PP-Tucker-ALS method described above.

Algorithm 4. PP-Tucker-ALS: Pairwise perturbation procedure for Tucker-ALS

Input: tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$, decomposition ranks $\{R_1, \ldots, R_N\}$, stopping criteria Δ, PP tolerance ϵ
Initialize $\left[\!\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\!\right]$ using HOSVD,initialize $d\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)}$ for $i \in \{1, \ldots, N\}$, initialize $\mathcal{F} \leftarrow \mathcal{G}$
while $\|\mathcal{F}\|_F > \Delta \|\mathcal{X}\|_F$ **do**
  if $\forall\, i \in \{1, \ldots, N\}, \left\|d\mathbf{A}^{(i)}\right\|_F < \epsilon \left\|\mathbf{A}^{(i)}\right\|_F$ **then**
    Compute $\mathcal{Y}_p^{(i,n)}, \mathcal{Y}_p^{(n)}$ for $i, n \in \{1, \ldots, N\}$ via dimension tree in Section 3.2.3
    for $n \in \{1, \ldots, N\}$ **do**
      $\mathbf{A}_p^{(n)} \leftarrow \mathbf{A}^{(n)}$, $d\mathbf{A}^{(n)} \leftarrow \mathbf{O}$
    end for
    while $\|\mathcal{F}\|_F > \Delta \|\mathcal{X}\|_F$ and $\|\mathcal{F}\|_F < \epsilon \|\mathcal{X}\|_F$ **do**
      for $n \in \{1, \ldots, N\}$ **do**
        $\mathcal{Y}^{(n)} \leftarrow \mathcal{Y}_p^{(n)} + \sum_{i=1, i \neq n}^{N} \mathcal{Y}_p^{(i,n)} \times_i d\mathbf{A}^{(i)}$
        $\mathbf{A}^{(n)} \leftarrow R_n$ leading left singular vectors of $\mathbf{Y}_{(n)}^{(n)}$
        $d\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)} - \mathbf{A}_p^{(n)}$
      end for
      $\mathcal{G}_{\text{new}} \leftarrow \mathcal{Y}^{(N)} \times_N \mathbf{A}^{(N)T}$
      $\mathcal{F} \leftarrow \mathcal{G}_{\text{new}} - \mathcal{G}$
      $\mathcal{G} \leftarrow \mathcal{G}_{\text{new}}$
    end while
  end if
  Perform regular ALS sweep as in Algorithm 2, taking $d\mathbf{A}^{(n)} \leftarrow \mathbf{A}_{\text{new}}^{(n)} - \mathbf{A}^{(n)}$ for each $n \in \{1, \ldots, N\}$
  $\mathcal{G}_{\text{new}} \leftarrow \mathcal{Y}^{(N)} \times_N \mathbf{A}^{(N)T}$
  $\mathcal{F} \leftarrow \mathcal{G}_{\text{new}} - \mathcal{G}$
  $\mathcal{G} \leftarrow \mathcal{G}_{\text{new}}$
end whilereturn $\left[\!\left[\mathcal{G}; \mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\!\right]$

3.2.3 | Dimension trees for PP operators

Computation of the PP operators $\mathcal{M}_p^{(i,n)}$ and of $\mathcal{M}_p^{(n)}$ can benefit from amortization of common tensor contraction (Khatri–Rao product or multilinear multiplication) subexpressions. In the context of ALS, this technique is known as dimension trees and has been successfully employed to accelerate TTMc and MTTKRP. The same trees can be used for both CP and Tucker, although the tensor intermediates and contraction operations are different (Khatri–Rao products for CP and multilinear multiplication for Tucker). We describe the trees for CP decomposition, computing each $\mathcal{M}_p^{(i,n)}$ and $\mathcal{M}_p^{(n)}$. Figure 1c,d describes the dimension tree for $N = 3, 4$. Our tree constructions assume that the tensors are equidimensional, if this is not the case, the largest dimensions should be contracted first.

The main goal of the dimension tree is to perform a minimal number of contractions to obtain each $\mathcal{M}_p^{(i,n)}$. Each matrix $\mathcal{M}_p^{(n)}$ can be simply obtained by a contraction with $\mathcal{M}_p^{(i,n)}$ for any $i \neq n$. Each level of the tree for $l = 1, \ldots, N-1$ should contain intermediate tensors containing $N - l + 1$ uncontracted modes belonging to the original tensor (the root is the original tensor $\mathcal{X} = \mathcal{M}^{(1,\ldots,N)}$). For any pair of the original tensor modes, each level should contain an intermediate for which these modes are uncontracted. Since the leaves at level $l = N - 1$ have two uncontracted modes, they will include each $\mathcal{M}_p^{(i,n)}$ for $i < n$ and have $\binom{N}{2}$ tensors overall. At level l it then suffices to compute $\binom{l+1}{2}$ tensors $\mathcal{M}^{(i,j,l+2,l+3,\ldots,N)}, \forall i, j \in \{1, \ldots, l+1\}, i < j$. Each $\mathcal{M}^{(i,j,l+2,l+3,\ldots,N)}$ can be computed by contraction of $\mathcal{M}^{(s,t,v,l+2,l+3,\ldots,N)}$ and $\mathbf{A}^{(w)}$ where $\{s, t, v\} = \{i, j, w\}$ with $w = \max_{w \in \{l-1, l, l+1\} \setminus \{i,j\}}(w)$ and $s < t < v$.

The construction of PP operators for CP decomposition costs

$$2R \sum_{l=2}^{N-1} \binom{l+1}{2} s^{N-l+2} = 6s^N R + 12s^{N-1} R + O\left(s^{N-2} R^2\right).$$

TABLE 1 Cost comparison between pairwise perturbation algorithm and ALS dimension tree algorithm for CP and Tucker decompositions

	DT ALS	PP initialization step	PP approximated step
CP	$4s^N R$	$6s^N R$	$2N^2(s^2R + sR^2)$
Tucker	$4s^N R$	$6s^N R$	$2N^2 s^2 R^{N-1}$

The cost to form PP operators for Tucker decomposition is

$$2\sum_{l=2}^{N-1}\binom{l+1}{2}s^{N-l+2}R^{l-1} = 6s^N R + 12s^{N-1}R^2 + O\left(s^{N-2}R^3\right).$$

We summarize the leading order computational costs for both algorithms in Table 1. The PP initialization step, which involves the PP operator construction and does one more first level contraction, is computationally 1.5× more expensive than the ALS algorithm.

As for the memory footprint, ALS with the best choice of dimension tree requires intermediate tensors of size $O\left(s^{\lceil N/2\rceil}R\right)$. As an example, for the order four case shown in Figure 1b, the first and second level contractions are combined to save memory, so that $\mathcal{M}^{(3,4)}$ and $\mathcal{M}^{(1,2)}$ are stored, both of size $O\left(s^2R\right)$. The PP dimension tree described above and in Figure 1d needs at least $O\left(s^{N-1}R\right)$ auxiliary memory to store the first level contraction results. The memory needed for PP can be reduced similar to ALS. For example, when calculating the PP operator $\mathcal{M}_p^{(1,3)}$ for an order four tensor, we can bypass the first level contraction and save its memory via directly performing a contraction between the input tensor and the Khatri–Rao product output $\mathbf{A}^{(1)} \odot \mathbf{A}^{(3)}$. Combining the first $l \leq N-2$ levels of contractions requires $O\left(s^{N-l}R + N^2s^2R\right)$ auxiliary memory, but incurs a cost of $O\left(l^2s^{N-1}R\right)$.

4 | ERROR ANALYSIS

In this section, we formally bound the approximation error of the PP algorithm relative to ALS. We show that quadratic optimization problems computed by PP differ only slightly from ALS so long as the factor matrices have not changed significantly since the construction of the PP operators.

4.1 | CP-ALS

To bound the error of PP, we view the ALS procedure for CP decomposition in terms of pairwise updates (Algorithm 5), pushing updates to least-squares problems of all tensors as soon as any one of them is updated. This reformulation is algebraically equivalent to Algorithm 1, but makes oracle-like use of $\mathcal{M}^{(m,n)}$ (Equation 1), recomputing which would increase the computational cost. We can bound the error of the way PP propagates updates to any right-hand side $\mathbf{M}^{(m)}$ due to changes in any one of the other factor matrices $\delta\mathbf{A}^{(n)}$. We define the update $\mathbf{H}^{(m,n)}$ in terms of its columns,

$$\mathbf{h}_k^{(m,n)}(x) = \sum_{y=1}^{s_n}\mathcal{M}^{(m,n)}(x,y,k)\delta\mathbf{A}^{(n)}(y,k), \quad \text{where} \quad \delta\mathbf{A}^{(n)} = \mathbf{A}^{(n)}_{\text{new}} - \mathbf{A}^{(n)}.$$

Note that $\delta\mathbf{A}^{(n)}$ denotes the update of nth factor between two neighboring sweeps, which should be distinguished from $d\mathbf{A}^{(n)}$, denoting the perturbation of nth factor in PP. Based on the definition, the update of each $\mathbf{M}^{(m)}$ after an ALS sweep is the summation of $\mathbf{H}^{(m,n)}$ expressed as $\delta\mathbf{M}^{(m)} = \sum_{n=1,n\neq m}^{N}\mathbf{H}^{(m,n)}$.

For simplicity, we first perform an error analysis for the case where the second order correction terms $\mathbf{V}^{(n,i,j)}$ are not included in PP. In Theorem 1, we prove that when the column-wise norm of $d\mathbf{A}^{(n)} = \mathbf{A}^{(n)} - \mathbf{A}_p^{(n)}$ relative to the norm of $\mathbf{A}^{(n)}$ for $n \in \{1,\ldots,N\}$ is small, the absolute error of column-wise results for $\mathbf{H}^{(m,n)}$ calculated from PP with respect to that calculated from exact ALS is also small. Corollary 1 provides a simple relative error bound for third-order tensors. Overall, these bounds demonstrate that PP should generally compute updates with small relative error with respect to the

magnitude of the perturbation of the factor matrices since the setup of the pairwise operators. However, this relative error can be amplified during other steps of ALS, which are ill-conditioned, that is, can suffer from catastrophic cancellation (the same would hold for round-off error).

Algorithm 5. CP-ALS: Reinterpreted ALS procedure for CP decomposition

Input: Tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$, stopping criteria Δ
Initialize $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ as uniformly distributed random matrices within $[0, 1]$, initialize $\mathbf{G}^{(n)}$, $\delta\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)}$, $\mathbf{S}^{(n)} \leftarrow \mathbf{A}^{(n)T}\mathbf{A}^{(n)}$ for $i \in \{1, \ldots, N\}$
for $n \in \{1, \ldots, N\}$ **do**
    Update $\mathbf{M}^{(n)}$ based on the dimension tree algorithm shown in Figure 1
end for
while $\sum_{i=1}^{N} \left\|\mathbf{G}^{(i)}\right\|_F > \Delta \|\mathcal{X}\|_F$ **do**
    for $n \in \{1, \ldots, N\}$ **do**
        $\mathbf{\Gamma}^{(n)} \leftarrow \mathbf{S}^{(1)} * \cdots * \mathbf{S}^{(n-1)} * \mathbf{S}^{(n+1)} * \cdots * \mathbf{S}^{(N)}$
        $\mathbf{A}^{(n)}_{\text{new}} \leftarrow \mathbf{M}^{(n)}\mathbf{\Gamma}^{(n)\dagger}$
        $\delta\mathbf{A}^{(n)} = \mathbf{A}^{(n)}_{\text{new}} - \mathbf{A}^{(n)}$
        $\mathbf{G}^{(n)} \leftarrow -\delta\mathbf{A}^{(n)}\mathbf{f}\mathbf{b}^{(n)}$
        $\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)}_{\text{new}}$
        $\mathbf{S}^{(n)} \leftarrow \mathbf{A}^{(n)T}\mathbf{A}^{(n)}$
        for $m \in \{1, \ldots, N\}, m \neq n$ **do**
            Update $\mathbf{M}^{(m)}$ as $\mathbf{M}^{(m)}(x, k) = \mathbf{M}^{(m)}(x, k) + \sum_{y=1}^{s_n} \mathcal{M}^{(m,n)}(x, y, k)\delta\mathbf{A}^{(n)}(y, k)$
        end for
    end for
end whilereturn $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$

We then perform an error analysis for the case where the second order correction terms $\mathbf{V}^{(n,i,j)}$ are included in PP in Theorem 2. We show that the second order corrections can tighten the leading order error by a factor related to the CP decomposition accuracy.

Theorem 1. *For $k \in \{1, \ldots, R\}$, if $\left\|d\mathbf{a}_k^{(l)}\right\|_2 / \left\|\mathbf{a}_k^{(l)}\right\|_2 \leq \epsilon < 1$ for all $l \in \{1, \ldots, N\}$, the PP algorithm without second order corrections computes the update $\tilde{\mathbf{H}}^{(1,N)}$ with columnwise error,*

$$\left\|\tilde{\mathbf{h}}_k^{(1,N)} - \mathbf{h}_k^{(1,N)}\right\|_2 = O(N\epsilon)\left\|\hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1}\left\|\mathbf{a}_k^{(j)}\right\|_2,$$

where $\mathbf{H}^{(1,N)}$ is the update to the matrix $\mathbf{M}^{(1)}$ due to the change $\delta\mathbf{A}^{(N)}$ performed by a regular ALS sweep, and $\hat{\mathcal{T}} = \mathcal{X} \times_N \delta\mathbf{a}_k^{(N)T}$. Analogous bounds hold for $\mathbf{H}^{(m,n)}$ for any $m, n \in \{1, \ldots, N\}$, $m \neq n$.

Proof. The ALS update and approximated update are

$$\mathbf{h}_k^{(1,N)} = \hat{\mathcal{T}} \bigtimes_{i \in \{2,\ldots,N-1\}} \mathbf{a}_k^{(i)T} \quad \text{and} \quad \tilde{\mathbf{h}}_k^{(1,N)} = \hat{\mathcal{T}} \bigtimes_{i \in \{2,\ldots,N-1\}} \left(\mathbf{a}_k^{(i)T} - d\mathbf{a}_k^{(i)T}\right). \tag{4}$$

We can expand the error as

$$\tilde{\mathbf{h}}_k^{(1,N)} - \mathbf{h}_k^{(1,N)} = \sum_{S \subset \{2,\ldots,N-1\}, S \neq \emptyset} \hat{\mathcal{T}} \bigtimes_{i \in \{2,\ldots,N-1\}} \mathbf{v}_k^{(i)T}, \quad \text{where} \quad \mathbf{v}_k^{(i)} = \begin{cases} -d\mathbf{a}_k^{(i)} & : i \in S, \\ \mathbf{a}_k^{(i)} & : i \notin S. \end{cases} \tag{5}$$

Consequently, we can upper-bound the error due to terms with $|S| = d$ by

$$\binom{N-2}{d} \epsilon^d \left\|\hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1} \left\|\mathbf{a}_k^{(j)}\right\|_2 = O(N\epsilon)^d \left\|\hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1} \left\|\mathbf{a}_k^{(j)}\right\|_2.$$

Therefore, the error bound when $|S| = d$ scales as $O(N\epsilon)^d$, and the leading order error is $O(N\epsilon)$. ∎

Note that this error bound involves $\hat{\mathcal{T}}$, which is small in norm due to being constructed from contraction with $\delta\mathbf{a}_k^{(N)}$. Thus, the error norm generally scales as $O(\epsilon^2)$ relative to the norm of the original tensor $\mathcal{X}$, since $O(N\epsilon)\left\|\hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1}\left\|\mathbf{a}_k^{(j)}\right\|_2 = O(N\epsilon^2)\left\|\mathcal{X}\right\|_2 \prod_{j=2}^{N-1}\left\|\mathbf{a}_k^{(j)}\right\|_2$.

Corollary 1. *For $N = 3$, using the bounds from the proof of Theorem 1, under the same assumptions, we obtain the absolute error bound,*

$$\left\|\tilde{\mathbf{h}}_k^{(1,3)} - \mathbf{h}_k^{(1,3)}\right\|_2 \leq \left\|\hat{\mathbf{T}}\right\|_2 \left\|\mathbf{a}_k^{(2)}\right\|_2 \epsilon,$$

where $\hat{\mathbf{T}} = \mathcal{X} \times_3 \delta\mathbf{a}_k^{(3)T}$. Further, since $\mathbf{h}_k^{(1,3)} = \hat{\mathbf{T}}\mathbf{a}_k^{(2)}$, the relative error is bounded by

$$\frac{\left\|\tilde{\mathbf{h}}_k^{(1,3)} - \mathbf{h}_k^{(1,3)}\right\|_2}{\left\|\mathbf{h}_k^{(1,3)}\right\|_2} \leq \kappa(\hat{\mathbf{T}})\epsilon.$$

From Theorem 1, we can conclude that the relative error in computing any column update $\mathbf{h}_k^{(i,j)}$ is $O(\epsilon)$ when $\epsilon \ll 1$ and the correct update is sufficiently large, for example, for $i = 1$ and $j = N$, $\left\|\mathbf{h}_k^{(1,N)}\right\|_2 = \Omega\left(\left\|\hat{\mathcal{T}}\right\|_2 \prod_{i=2}^{N-1}\left\|\mathbf{a}_k^{(l)}\right\|_2\right)$. When this is the case, we can also bound the error of the update to the columns of the right-hand sides $\delta\mathbf{M}^{(n)}$ formed in ALS, so long as the sum of the updates $\mathbf{H}^{(n,m)}$ for $m \neq n$ is not too small in norm relative to each update matrix.

We now perform analysis for the case where the second order corrections $\mathbf{V}^{(n,i,j)}$ are included in PP.

Theorem 2. *For $k \in \{1, \ldots, R\}$, if $\left\|d\mathbf{a}_k^{(l)}\right\|_2 / \left\|\mathbf{a}_k^{(l)}\right\|_2 \leq \epsilon < 1$ for all $l \in \{1, \ldots, N\}$, the PP algorithm with second order correction terms computes the update term $\tilde{\mathbf{H}}^{(1,N)}$ with columnwise error,*

$$\left\|\tilde{\mathbf{h}}_k^{(1,N)} - \mathbf{h}_k^{(1,N)}\right\|_2 = O(N\epsilon)\left\|\hat{\mathcal{P}} - \hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1}\left\|\mathbf{a}_k^{(j)}\right\|_2 + O\left((N\epsilon)^2\right)\left\|\hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1}\left\|\mathbf{a}_k^{(j)}\right\|_2,$$

where $\hat{\mathcal{P}} = \mathcal{Z} \times_N \delta\mathbf{a}_k^{(N)T}$, and $\mathcal{Z}$ denotes the approximate CP decomposition of $\mathcal{X}$, $\left[\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}\right]\right]$. $\mathbf{H}^{(1,N)}$ is the update to the matrix $\mathbf{M}^{(1)}$ due to the change $\delta\mathbf{A}^{(N)}$ performed by a regular ALS sweep, and $\hat{\mathcal{T}} = \mathcal{X} \times_N \delta\mathbf{a}_k^{(N)T}$. Analogous bounds hold for $\mathbf{H}^{(m,n)}$ for any $m, n \in \{1, \ldots, N\}$, $m \neq n$.

Proof. The ALS approximated update is

$$\tilde{\mathbf{h}}_k^{(1,N)} = \hat{\mathcal{T}} \bigtimes_{i\in\{2,\ldots,N-1\}} \left(\mathbf{a}_k^{(i)T} - d\mathbf{a}_k^{(i)T}\right) + \sum_{i\in\{2,\ldots,N-1\}} \hat{\mathcal{P}} \times_i d\mathbf{a}_k^{(i)T} \bigtimes_{j\in\{2,\ldots,N-1\}, j\neq i} \mathbf{a}_k^{(j)T}. \tag{6}$$

We can expand the error as

$$\tilde{\mathbf{h}}_k^{(1,N)} - \mathbf{h}_k^{(1,N)} = \sum_{S\subset\{2,\ldots,N-1\}, |S|\geq 2} \hat{\mathcal{T}} \bigtimes_{i\in\{2,\ldots,N-1\}} \mathbf{v}_k^{(i)T} + \sum_{i\in\{2,\ldots,N-1\}} \left(\hat{\mathcal{P}} - \hat{\mathcal{T}}\right) \times_i d\mathbf{a}_k^{(i)T} \bigtimes_{j\in\{2,\ldots,N-1\}, j\neq i} \mathbf{a}_k^{(j)T},$$

where $\mathbf{v}_k^{(i)} = -d\mathbf{a}_k^{(i)}$ if $i \in S$ and $\mathbf{v}_k^{(i)} = \mathbf{a}_k^{(i)}$ otherwise. By the same analysis as in Theorem 1, the error due to each term with $|S| = d$, $d \geq 2$ can be bounded as $O(N\epsilon)^d \left\|\hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1}\left\|\mathbf{a}_k^{(j)}\right\|_2$. We can then upper-bound the error due to the second term by

$$O(N\epsilon)\left\|\hat{\mathcal{P}}-\hat{\mathcal{T}}\right\|_2\prod_{j=2}^{N-1}\left\|\mathbf{a}_k^{(j)}\right\|_2, \tag{7}$$

thus completing the proof. ∎

From Theorem 2, we can conclude that when the approximate CP decomposition is close to $\mathcal{X}$, the term expressed in Equation (7) will have small magnitude, making the absolute error second order accurate in terms of ϵ.

In Appendix A.4, we also obtain relative error bounds on MTTKRPs (the right-hand sides in the linear least squares subproblems). However, this error bound is relative to the condition number of $\mathcal{X}$ (defined in Appendix A.1), which is infinite for sufficiently large tensors.

4.2 | Tucker-ALS

For Tucker decomposition, the PP approximation satisfies better bounds than for CP decomposition, due to the orthogonality of the factor matrices. We can not only obtain the similar bound as Theorem 1, but also obtain stronger results in tensor spectral norm (defined in (8)) assuming that the residual of the Tucker decomposition is bounded (it suffices that the decomposition achieves one digit of accuracy in residual), and stronger results in Frobenius norm assuming that the ratio of rank to dimension is not too large.

The spectral norm of any tensor $\mathcal{T}\in\mathbb{R}^{s_1\times\cdots s_N}$ is

$$\|\mathcal{T}\|_2=\max_{\substack{\forall i\in\{2,\ldots,N\},\mathbf{x}^{(i)}\in\mathbb{R}^{s_i}\\ \|\mathbf{x}^{(2)}\|_2=\cdots=\|\mathbf{x}^{(N)}\|_2=1}}\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N\}}\mathbf{x}^{(i)T}\right\|_2, \tag{8}$$

where $\mathcal{T}$ is contracted with $\mathbf{x}^{(i)}$ along its ith mode. The spectral tensor norm corresponds to the magnitude of the largest tensor singular value.[42] Computing the spectral norm is NP-hard,[43] but can usually be done in practice by specialized variants of ALS.[44] The spectral norm is invariant under reordering of modes of $\mathcal{T}$. Lemma 1 shows submultiplicativity of this norm for the multilinear multiplication.

Lemma 1. *Given any tensor $\mathcal{T}\in\mathbb{R}^{s_1\times\cdots\times s_N}$ and matrix $\mathbf{M}\in\mathbb{R}^{s_N\times R}$, if $\mathcal{V}=\mathcal{T}\times_N\mathbf{M}^T$ then $\|\mathcal{V}\|_2\le\|\mathcal{T}\|_2\|\mathbf{M}\|_2$.*

Proof. There exist unit vectors $\mathbf{x}^{(2)},\ldots,\mathbf{x}^{(N)}$ such that

$$\|\mathcal{V}\|_2=\left\|\mathcal{V}\bigtimes_{i\in\{2,\ldots,N\}}\mathbf{x}^{(i)T}\right\|_2=\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N-1\}}\mathbf{x}^{(i)T}\times_N\left(\mathbf{M}\mathbf{x}^{(N)}\right)^T\right\|_2.$$

Let $\mathbf{z}=\mathbf{M}\mathbf{x}^{(N)}$, so $\|\mathbf{z}\|_2\le\|\mathbf{M}\|_2$. If $\|\mathbf{z}\|_2=0$, then $\|\mathcal{V}\|_2=0$, the inequality holds. Otherwise, since

$$\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N-1\}}\mathbf{x}^{(i)T}\times_N\mathbf{z}^T\right\|_2\le\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N-1\}}\mathbf{x}^{(i)T}\times_N\mathbf{z}^T\right\|_2\frac{\|\mathbf{M}\|_2}{\|\mathbf{z}\|_2}\le\|\mathcal{T}\|_2\|\mathbf{M}\|_2,$$

the inequality still holds. ∎

Using Lemma 1, we prove in Lemma 2 that after contracting a tensor with a matrix with orthonormal columns, whose row length is higher or equal to the column length, the contracted tensor norm is the same as the original tensor norm.

Lemma 2. *Given tensor $\mathcal{G}\in\mathbb{R}^{r_1\times\cdots\times r_N}$, the mode-n product for any $n\in\{1,\ldots,N\}$, with a matrix with orthonormal columns $\mathbf{M}\in\mathbb{R}^{s\times r_n}$, $r_n\le s$, satisfies $\|\mathcal{G}\|_2=\|\mathcal{G}\times_n\mathbf{M}\|_2$.*

Proof. Based on the submultiplicative property of the tensor norm (Lemma 1),

$$\|\mathcal{G}\|_2=\left\|\mathcal{G}\times_n(\mathbf{M}^T\mathbf{M})\right\|_2=\left\|\mathcal{G}\times_n\mathbf{M}\times_n\mathbf{M}^T\right\|_2\le\|\mathcal{G}\times_n\mathbf{M}\|_2\left\|\mathbf{M}^T\right\|_2=\|\mathcal{G}\times_n\mathbf{M}\|_2,$$

and simultaneously, $\|\mathcal{G} \times_n \mathbf{M}\|_2 \leq \|\mathcal{G}\|_2 \|\mathbf{M}\|_2 = \|\mathcal{G}\|_2$. ∎

Below we demonstrate that

- similar to Algorithm 5 and Theorem 1, when we view the ALS procedure for Tucker decomposition of equidimensional tensors in terms of pairwise updates, we can bound the error of updates to any right-hand side $\mathcal{Y}^{(m)}$ due to changes in any one of the other factor matrices $\delta\mathbf{A}^{(n)}$. We define the update $\mathcal{J}^{(m,n)}$ as

$$\mathcal{J}^{(m,n)} = \mathcal{Y}^{(m,n)} \times_n \delta\mathbf{A}^{(n)T}, \quad \text{where} \quad \delta\mathbf{A}^{(n)} = \mathbf{A}^{(n)}_{\text{new}} - \mathbf{A}^{(n)}.$$

  The columnwise absolute error bound for MTTKRP holds for $\mathcal{J}^{(m,n)}$ when the column-wise 2-norm relative perturbations of the input matrices are bounded by $O(\epsilon)$ (Theorem 3),
- the relative error of $\mathcal{Y}^{(m)}$ for $m \in \{1, \ldots, N\}$ satisfies the bound of $O\left(\epsilon^2\right)$, so long as the residual of Tucker decomposition is small (Theorem 4),
- the relative error of $\mathcal{Y}^{(m)}$ for $m \in \{1, \ldots, N\}$ is bounded in Frobenius norm by $O\left(\epsilon^2\right)$ for a fixed problem size assuming that HOSVD is performed to initialize Tucker-ALS (Theorem 5).

Theorem 3. *For an order N tensor $\mathcal{X}$ with dimension sizes s, if $\left\|d\mathbf{a}_k^{(n)}\right\|_2 / \left\|\mathbf{a}_k^{(n)}\right\|_2 \leq \epsilon < 1$ for all $n \in \{1, \ldots, N\}, k \in \{1, \ldots, R\}$, the PP algorithm computes update $\mathcal{J}^{(1,N)}$ with error,*

$$\left\|\tilde{\mathbf{j}}^{(1,N)}_{i_2,\ldots,i_N} - \mathbf{j}^{(1,N)}_{i_2,\ldots,i_N}\right\|_2 = O(N\epsilon) \left\|\hat{\mathcal{T}}\right\|_2 \prod_{j=2}^{N-1} \left\|\mathbf{a}_k^{(j)}\right\|_2,$$

where $\hat{\mathcal{T}} = \mathcal{X} \times_N \delta\mathbf{a}^{(N)T}_{i_N}$ and $\mathbf{j}^{(1,N)}_{i_2,\ldots,i_N}(x) = \mathcal{J}^{(1,N)}(x, i_2, \ldots, i_N)$.

Proof. The proof is similar to that of Theorem 1. The ALS update and approximated update after a change $\delta\mathbf{A}^{(N)}$ are

$$\mathbf{j}^{(1,N)}_{i_2,\ldots,i_N} = \hat{\mathcal{T}} \bigtimes_{j=2}^{N-1} \mathbf{a}^{(j)T}_{i_j} \quad \text{and} \quad \tilde{\mathbf{j}}^{(1,N)}_{i_2,\ldots,i_N} = \hat{\mathcal{T}} \bigtimes_{j=2}^{N-1} \left(\mathbf{a}^{(j)T}_{i_j} - d\mathbf{a}^{(j)T}_{i_j}\right).$$

The error bound proceeds by analogy to the proof of Theorem 1. ∎

Using Lemma 2, we prove in Theorem 4 that when the relative error of the matrices $\mathbf{A}^{(n)}$ for $n \in \{1, \ldots, N\}$ is small and the residual of the Tucker decomposition is loosely bounded, the relative error bound for the $\mathcal{Y}^{(n)}$ is independent of the tensor condition number defined in Section A.

Theorem 4. *Given tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$, if $\left\|d\mathbf{A}^{(n)}\right\|_2 \leq \epsilon \ll 1$ for $n \in \{1, \ldots, N\}$ and $\left\|\mathcal{X} - [[\mathcal{G}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]]\right\|_2 \leq \frac{1}{3}\|\mathcal{X}\|_2$, $\tilde{\mathcal{Y}}^{(n)}$ is constructed with error,*

$$\frac{\left\|\tilde{\mathcal{Y}}^{(n)} - \mathcal{Y}^{(n)}\right\|_2}{\left\|\mathcal{Y}^{(n)}\right\|_2} = O\left(\epsilon^2\right).$$

Proof.

$$\frac{\left\|\tilde{\mathcal{Y}}^{(n)} - \mathcal{Y}^{(n)}\right\|_2}{\left\|\mathcal{Y}^{(n)}\right\|_2} \leq \binom{N}{2} \max_{i,j} \frac{\left\|\mathcal{Y}_p^{(i,j,n)} \times_i d\mathbf{A}^{(i)T} \times_j d\mathbf{A}^{(j)T}\right\|_2}{\left\|\mathcal{Y}^{(n)}\right\|_2} \leq \binom{N}{2} \max_{i,j} \frac{\left\|\mathcal{Y}_p^{(i,j,n)}\right\|_2 \left\|d\mathbf{A}^{(i)}\right\|_2 \left\|d\mathbf{A}^{(j)}\right\|_2}{\left\|\mathcal{Y}^{(n)}\right\|_2}.$$

Let $\tilde{\mathcal{X}} = \left[\left[\mathcal{G}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}\right]\right], \mathcal{R} = \mathcal{X} - \tilde{\mathcal{X}}$. Define the tensors $\mathcal{Z}^{(i,j,n)}$ by contraction of $\mathcal{R}$ with all except three factor matrices,

$$\mathcal{Z}^{(i,j,n)} = \mathcal{R} \bigtimes_{r \in \{1,\ldots,N\} \setminus \{i,j,n\}} \mathbf{A}^{(r)T}.$$

For $\left\|\mathcal{X} - \tilde{\mathcal{X}}\right\|_2 = \|\mathcal{R}\|_2 \leq \frac{1}{3}\|\mathcal{X}\|_2$, we have $\frac{2}{3}\|\mathcal{X}\|_2 \leq \left\|\tilde{\mathcal{X}}\right\|_2 \leq \frac{4}{3}\|\mathcal{X}\|_2$. Based on Lemma 2,

$$\left\|\mathcal{Y}^{(n)}\right\|_2 = \left\|\mathcal{G} \times_n \mathbf{A}^{(n)} + \mathcal{Z}^{(i,j,n)} \times_i \mathbf{A}^{(i)T} \times_j \mathbf{A}^{(j)T}\right\|_2 \geq \|\mathcal{G}\|_2 - \left\|\mathcal{Z}^{(i,j,n)}\right\|_2 \left\|\mathbf{A}^{(i)T}\right\|_2 \left\|\mathbf{A}^{(j)T}\right\|_2 \geq \|\mathcal{G}\|_2 - \|\mathcal{R}\|_2 \geq \frac{1}{3}\|\mathcal{X}\|_2.$$

Additionally,

$$\left\|\mathcal{Y}^{(i,j,n)}\right\|_2 = \left\|\mathcal{G} \times_i \mathbf{A}^{(i)} \times_j \mathbf{A}^{(j)} \times_n \mathbf{A}^{(n)} + \mathcal{Z}^{(i,j,n)}\right\|_2 \leq \|\mathcal{G}\|_2 + \|\mathcal{R}\|_2 \leq \frac{5}{3}\|\mathcal{X}\|_2.$$

Therefore,

$$\frac{\left\|\tilde{\mathcal{Y}}^{(n)} - \mathcal{Y}^{(n)}\right\|_2}{\left\|\mathcal{Y}^{(n)}\right\|_2} \leq \binom{N}{2} \max_{i,j} \frac{\left\|\mathcal{Y}_p^{(i,j,n)}\right\|_2 \left\|d\mathbf{A}^{(i)}\right\|_2 \left\|d\mathbf{A}^{(j)}\right\|_2}{\left\|\mathcal{Y}^{(n)}\right\|_2} \leq \binom{N}{2} \frac{\frac{5}{3}\|\mathcal{X}\|_2 \epsilon^2}{\frac{1}{3}\|\mathcal{X}\|_2} = O\left(\epsilon^2\right).$$

■

We now derive a Frobenius norm error bound that is independent of residual norm and tensor condition number, and is based the ratio of the tensor dimensions and the Tucker rank. We arrive at this result (Theorem 5) by obtaining a lower bound on the residual achieved by the HOSVD (Lemmas 3 and 4).

Lemma 3. *Given tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$ and matrix $\mathbf{A} \in \mathbb{R}^{R \times s_n}$, where $R < \max\left\{s_n, \prod_{i=1, i\neq n}^{N} s_i\right\}$ and $\mathbf{A}$ consists of R leading left singular vectors of $\mathbf{X}_{(n)}$. Let $\mathcal{Z} = \mathcal{X} \times_n \mathbf{A}$, $\|\mathcal{X}\|_F \geq \|\mathcal{Z}\|_F \geq \sqrt{\frac{R}{s_n}}\|\mathcal{X}\|_F$.*

Proof. The singular values of $\mathbf{A}\mathbf{X}_{(n)}$ are the first R singular values of $\mathbf{X}_{(n)}$. Since the square of the Frobenius norm of a matrix is the sum of the squares of the singular values, $\|\mathcal{Z}\|_F^2 = \left\|\mathbf{A}\mathbf{X}_{(n)}\right\|_F^2 \geq (R/s_n)\left\|\mathbf{X}_{(n)}\right\|_F^2 = (R/s_n)\|\mathcal{X}\|_F^2$ and $\|\mathcal{Z}\|_F \leq \|\mathcal{X}\|_F$. ■

Lemma 4. *For any equidimensional order N tensor $\mathcal{X}$ with size s, $\left\|\mathcal{Y}^{(n)}\right\|_F \geq \left(\frac{R}{s}\right)^{N/2}\|\mathcal{X}\|_F$ if Tucker-ALS starts from an interlaced HOSVD.*

Proof. In Tucker-ALS, $\|\mathcal{G}\|_F$ is strictly increasing after each Tucker iteration, where $\mathcal{G}$ is $\mathcal{X}$'s HOSVD core tensor. Since the interlaced SVD computes each $\mathbf{A}^{(n)}$ from the truncated SVD of the product of $\mathcal{X}$ and the first $n-1$ factor matrices, we can apply Lemma 3 N times,

$$\left\|\mathcal{X} \times_1 \mathbf{A}^{(1)T} \cdots \times_{N-1} \mathbf{A}^{(N-1)T}\right\|_F \geq \|\mathcal{G}\|_F \geq \sqrt{\frac{R}{s}}\left\|\mathcal{X} \times_1 \mathbf{A}^{(1)T} \cdots \times_{N-1} \mathbf{A}^{(N-1)T}\right\|_F,$$

$$\vdots$$

$$\|\mathcal{X}\|_F \geq \|\mathcal{G}\|_F \geq (R/s)^{N/2}\|\mathcal{X}\|_F.$$

■

Theorem 5. *Given any equidimensional order N tensor $\mathcal{X}$ with size s, if $\left\|d\mathbf{A}^{(n)}\right\|_F \leq \epsilon$ for $n \in [1, N]$, $\tilde{\mathcal{Y}}^{(n)}$ is constructed with error,*

$$\frac{\left\|\tilde{\mathcal{Y}}^{(n)} - \mathcal{Y}^{(n)}\right\|_F}{\left\|\mathcal{Y}^{(n)}\right\|_F} = O\left(\epsilon^2\left(\frac{s}{R}\right)^{N/2}\right),$$

assuming that HOSVD is used to initialize Tucker-ALS and the residual associated with factor matrices $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(n)}$ is no higher than that attained by HOSVD.

Proof.

$$\frac{\left\|\tilde{\mathcal{Y}}^{(n)}-\mathcal{Y}^{(n)}\right\|_F}{\left\|\mathcal{Y}^{(n)}\right\|_F} \leq \binom{N}{2} \max_{i,j} \frac{\left\|\mathcal{Y}_p^{(i,j,n)} \times_i d\mathbf{A}^{(i)T} \times_j d\mathbf{A}^{(j)T}\right\|_F}{\left\|\mathcal{Y}^{(n)}\right\|_F}.$$

From Lemma 4, we have

$$\frac{\left\|\mathcal{Y}_p^{(i,j,n)} \times_i d\mathbf{A}^{(i)T} \times_j d\mathbf{A}^{(j)T}\right\|_F}{\left\|\mathcal{Y}^{(n)}\right\|_F} \leq \frac{\|\mathcal{X}\|_F \left\|d\mathbf{A}^{(i)}\right\|_F \left\|d\mathbf{A}^{(j)}\right\|_F}{\left(\frac{R}{s}\right)^{N/2} \|\mathcal{X}\|_F}.$$

Consequently, we can bound the relative error by

$$\frac{\left\|\tilde{\mathcal{Y}}^{(n)}-\mathcal{Y}^{(n)}\right\|_F}{\left\|\mathcal{Y}^{(n)}\right\|_F} \leq \binom{N}{2} (s/R)^{N/2} \max_{i,j} \left\|d\mathbf{A}^{(i)}\right\|_F \left\|d\mathbf{A}^{(j)}\right\|_F = O\left(\epsilon^2 \left(\frac{s}{R}\right)^{N/2}\right).$$

■

5 | EXPERIMENTS

We evaluate the performance of the PP algorithms on both synthetic tensors and application datasets. The synthetic experiments enable us to test tensors with known factors and to measure how effectively the algorithm works across many problem instances. We also consider publicly available tensor datasets as well as tensors of interest for quantum chemistry calculations and demonstrate the effectiveness of our algorithms on practical problems. We focus on the experiments on CP decomposition, because for many cases in Tucker decomposition, HOOI converges in small number of iterations with the initialization of HOSVD.

We use the metrics *relative residual* and *fitness* to evaluate the convergence of the decomposition. Let $\tilde{\mathcal{X}}$ denote the tensor reconstructed by the factor matrices and the core tensor, the relative residual and fitness are defined as follows,

$$r = \frac{\|\mathcal{X}-\tilde{\mathcal{X}}\|_F}{\|\mathcal{X}\|_F}, \quad f = 1 - r.$$

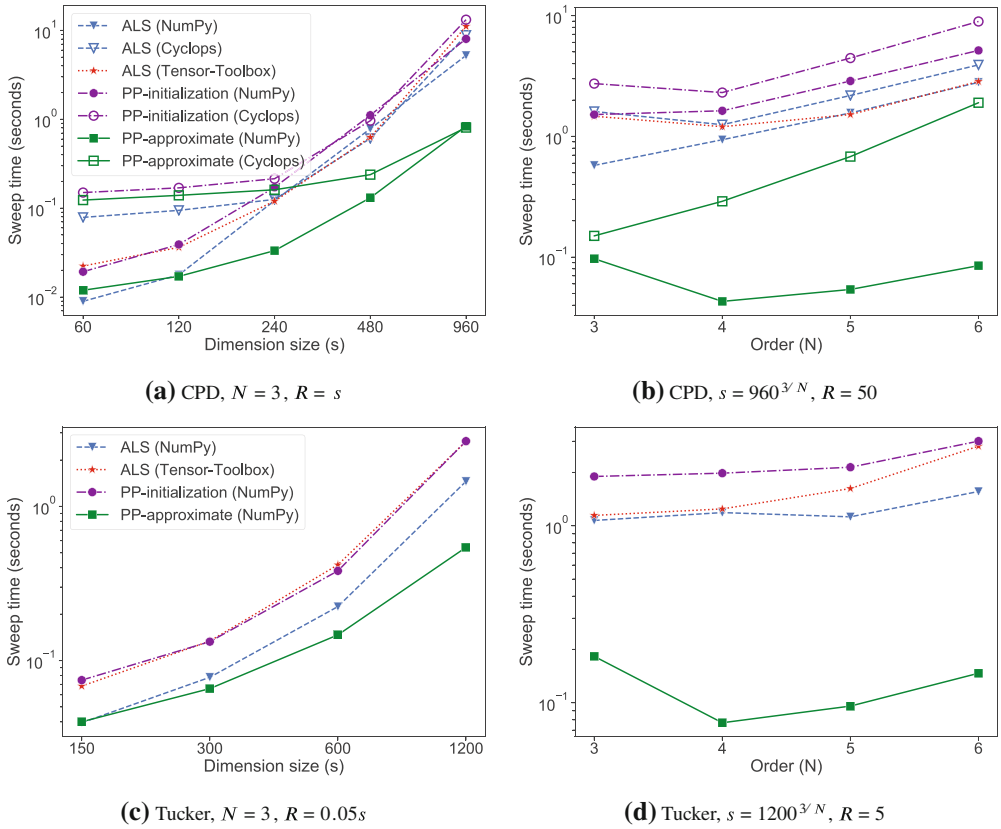
We compare the performance of our own implementations of regular ALS with dimension trees to the PP algorithm. Both algorithms are implemented in Python with NumPy for sequential calculation and with Cyclops Tensor Framework (v1.5.5),[30] which is a distributed-memory library for matrix/tensor contractions that uses MPI for interprocessor communication and OpenMP for threading. We also make use of a wrapper Cyclops provides for ScaLAPACK[31] routines to solve symmetric positive definite linear systems of equations and compute the SVD.*

The experimental results are collected on the Stampede2 supercomputer located at the University of Texas at Austin. We leverage the Knight's Landing (KNL) nodes exclusively, each of which consists of 68 cores, 96 GB of DDR RAM, and 16 GB of MCDRAM. These nodes are connected via a 100 Gb/s fat-tree Omni-Path interconnect. For both NumPy and Cyclops implementations, we use Intel compilers and the MKL library for threaded BLAS routines, including batched BLAS routines, which are efficient for Khatri–Rao products arising in MTTKRP in CP decomposition, and employ the HPTT library[45] for high-performance tensor transposition. All storage and computation assumes the tensors are dense.

5.1 | Sequential experimental results

We collect the sequential results on one KNL node on Stampede2, leveraging 64 threads for MKL and HPTT routines.

We compare the per-sweep time of the ALS dimension tree to the PP initialization and approximated sweep in Figure 2. Each initialization sweep constructs the PP operators and updates all the factor matrices, while an approximated sweep computes approximate updates to all the factor matrices using the PP operators constructed in the last initialization sweep. We also provide the reference per-sweep time of the ALS implementation from MATLAB Tensor Toolbox.[46] As can be



(a) CPD, $N = 3$, $R = s$ **(b)** CPD, $s = 960^{3/N}$, $R = 50$

(c) Tucker, $N = 3$, $R = 0.05s$ **(d)** Tucker, $s = 1200^{3/N}$, $R = 5$

FIGURE 2 Sequential ALS sweep time comparison for both CP and Tucker decompositions. Results are taken as the mean time across 5 sweeps. The line label of (b) is the same as (a), of (d) is the same as (c). In (a, c), we vary the dimension size and the decomposition rank, and fix the input tensor order. In (b, d), we vary the input tensor order, and fix the input tensor size and the decomposition rank

seen, both ALS sweep times on top of NumPy and Cyclops are comparable to the Tensor Toolbox. For both decompositions and all the configurations, the time of an PP initialization sweep is 1.5–2.0× the time of a dimension tree based ALS sweep, while the approximated steps can have up to 6.3× speed-up for an order three tensor and 33.0× speed-up for an order six tensor for CP, and up to 10.6× speed-up for an order 6 tensor for Tucker. In addition, larger speed-up can be achieved with the increase of dimension size s and the tensor order N, which is consistent with Table 1.

We use five different tensors to test the sequential performance of PP. Sequential performance results are collected using NumPy, as NumPy has better sequential performance than Cyclops, as shown in Figure 2a,b. For all the experiments, the PP tolerance is set as 0.1 for CP decomposition, and set as 0.3 for Tucker decomposition.

1. *Tensors with random collinearity*.[20] We create tensors based on known randomly-generated factor matrices $\mathbf{A}^{(n)}$. The factor matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{s \times R}$ are randomly generated so that the columns have collinearity defined based on a scalar C (selected randomly for the tensor from a given interval $[a, b)$), so that

$$\frac{\left\langle \mathbf{a}_i^{(n)}, \mathbf{a}_j^{(n)} \right\rangle}{\left\| \mathbf{a}_i^{(n)} \right\|_2 \left\| \mathbf{a}_j^{(n)} \right\|_2} = C, \quad \forall i, j \in \{1, \ldots, R\}, i \neq j.$$

   Higher collinearity corresponds to greater overlap between columns within each factor matrix, which makes the convergence of CP-ALS slower.[47]
2. *Tensors made by random matrices*. We create tensors based on known uniformly distributed randomly-generated factor matrices $\mathbf{A}^{(n)} \in [0, 1]^{s \times R}$,

$$\mathcal{X} = \left[\left[\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)} \right] \right].$$

In the experiments, we set R to be the same as the decomposition rank.

3. *Quantum chemistry tensor*. We also test on the density fitting intermediate tensor arising in quantum chemistry, which is the Cholesky factor of the two-electron integral tensor.[8,9] For an order 4 two-electron integral tensor $\boldsymbol{\mathcal{T}}$, its Cholesky factor is an order 3 tensor $\boldsymbol{\mathcal{D}}$, with their relations shown as follows:

$$\boldsymbol{\mathcal{T}}(a,b,c,d) = \sum_{s=1}^{P} \boldsymbol{\mathcal{D}}(a,b,s)\boldsymbol{\mathcal{D}}(c,d,s),$$

where P is the third mode dimension size of $\boldsymbol{\mathcal{D}}$. CP decomposition can be performed on $\boldsymbol{\mathcal{D}}$ to provide the compressed form of the density fitting intermediate and can be used to speed up post Hartree–Fock calculations.[48] We generate the density fitting tensor via the PySCF library,[49] which represents the compressed restricted Hartree–Fock wave function of an 8 water molecule chain system with a STO3G basis set. The generated tensor has size $904 \times 56 \times 56$. We set the CP rank to be 400.

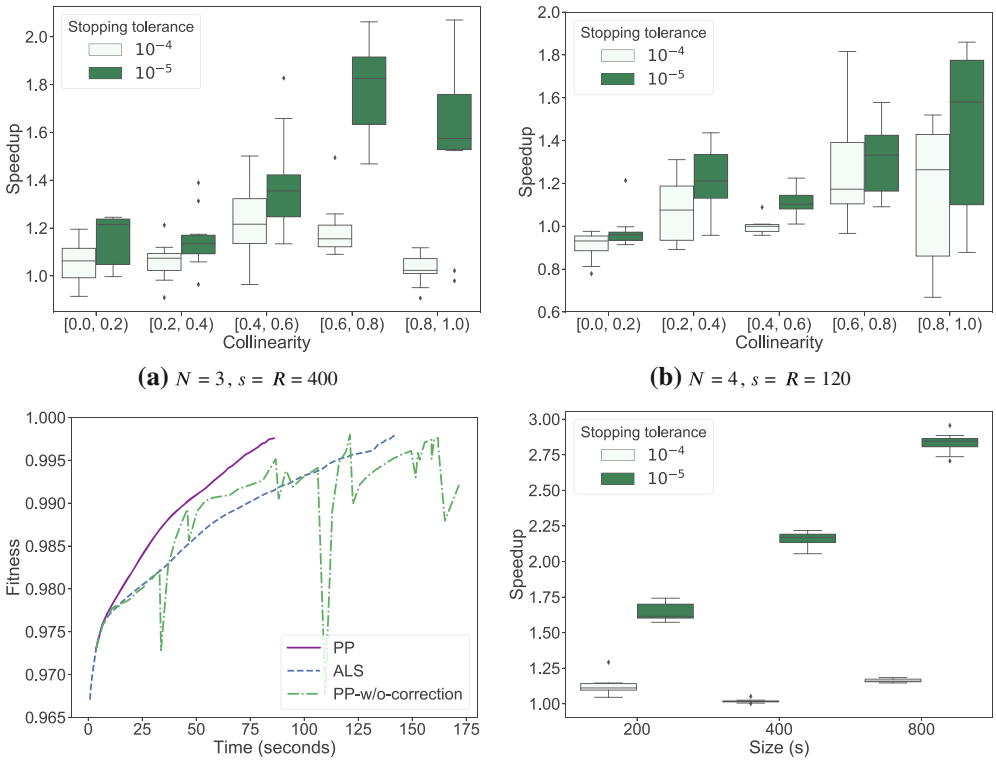
4. *COIL dataset*. COIL-100 is an image-recognition data set that contains images of objects in different poses[50] and has been used previously as a tensor decomposition benchmark.[20,51] There are 100 different object classes, each of which is imaged from 72 different angles. Each image has 128×128 pixels in three color channels. Transferring the data into tensor format, we have a $128 \times 128 \times 3 \times 7200$ tensor. We fix the CP decomposition rank to be 15 and the Tucker decomposition rank to be $10 \times 10 \times 3 \times 50$.
5. *Time-Lapse hyperspectral radiance images*. We consider the 3D hyperspectral imaging dataset called "Souto wood pile".[52] The dataset is usually used on the benchmark of nonnegative tensor decomposition.[29,53] The hyperspectral data consists of a tensor with dimensions $1024 \times 1344 \times 33 \times 9$. We fix the CP decomposition rank to be 50 and the Tucker decomposition rank to be $100 \times 100 \times 3 \times 3$.

The order three tensors are tested to justify the relative error bound shown in Section 4.1. The performance of PP on higher order CP decompositions is also considered. The input tensors are explicitly given for all cases we considered. Note that for cases arising in scientific computing where the input tensors are given in the CP decomposition format, the efficient CP-to-Tucker-to-CP decomposition technique based on reduced HOSVD (RHOSVD) introduced in Reference 54 can be used. We focus on the high rank CP decomposition, because for the cases with rank $R < s$, Tucker decomposition or HOSVD can be used to effectively compress the input tensor from dimensions of size s to R, and then CP decomposition can be performed.[55,56]

We test the synthetic tensors for CP decomposition. These tensors are all generated based on known factor matrices whose column sizes are equal to the decomposition rank, so these tensors have exact decompositions. For Tensor 1, we test on both order three tensors with both dimension sizes s and decomposition rank R equal to 400 and order four tensors with $s = R = 120$, and test the performance of PP on tensors with different collinearity for the exact input factor matrices. For Tensor 2, we test on order three tensors with $s = R$, and test the performance of PP with different dimension size and corresponding rank.

We display the speed-ups of PP compared to the dimension tree algorithm for synthetic tensors in Figure 3. Figure 3a,b shows the speed-up distribution with different exact factor matrices collinearity. We stop the algorithm when the stopping tolerance (defined as the fitness difference between two neighboring sweeps) is reached. It can be seen that for both order three and order four tensors, PP achieves up to 2.0× speed-up, and high speed-up is achieved with tighter stopping tolerance. We find that the stricter stopping tolerance of 10^{-5} is valuable, as generally it permits about one more digit of accuracy to be achieved in fitness compared to a tolerance of 10^{-4}. In addition, experiments with a 10^{-4} stopping tolerance sometimes stop at transient swamps[57] with high decomposition residual, where ALS makes small progress for a period but the residual norm decreases more rapidly afterwards. In addition, PP tends to have higher speed-ups with relatively high collinearity. This is because tensors with high collinearity will converge in more sweeps, and more PP approximated sweeps are activated as can be seen in Table 2. PP starts working early for almost all the experiments, as can be observed in Figure 3c, where PP starts to have speed-up when the fitness is around 0.975 and the experiment time is less than 20 s, and in Table 2, where almost all the PP initialization steps start within 20 sweeps. In addition, the fitness increases monotonically in Figure 3c, indicating that PP controls the approximation error well.

Figure 3c also illustrates the importance of the second-order correction term, $\mathbf{V}^{(n,i,j)}$, in Equation (3). We set the PP tolerance to be 0.02 for the PP experiment without corrections, which results in more conservative use of PP approximate steps than with the 0.1 tolerance we use for PP with the second-order correction. As can be seen, without the correction,



(a) $N = 3, s = R = 400$ **(b)** $N = 4, s = R = 120$

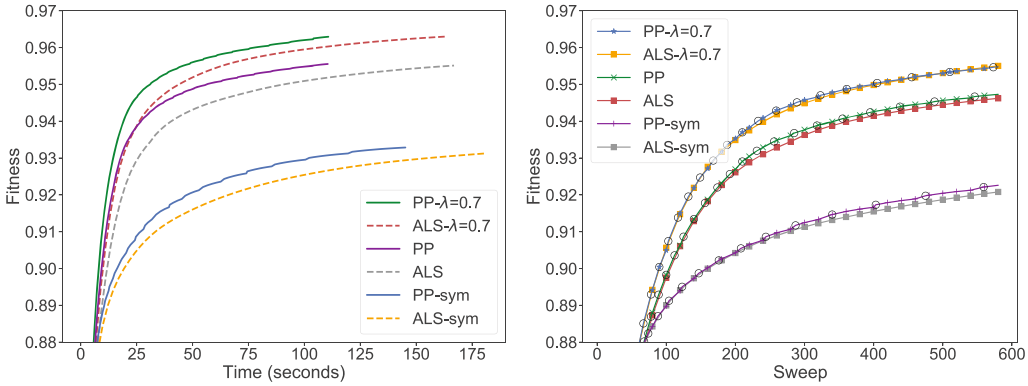
(c) $N = 3, s = R = 400$, collinearity$\in [0.6, 0.8)$ **(d)** Random tensors, $N = 3, s = R$

FIGURE 3 (a, b) Box plot of the relation between PP speed-up and input collinearity ranges for tensors with specific collinearity. (c) Fitness-time relation for the decomposition of one tensor with specific collinearity. (d) Box plot of the relation between PP speed-up and size in each mode for order 3 random tensors. For all the box plots, each box is based on 10 experiments with different random seeds. Each box shows the 25th–75th quartiles, the median is indicated by a horizontal line inside the box, and outliers are displayed as dots

TABLE 2 Detailed statistics of the results shown in Figure 3

Configuration	Num-ALS	Num-PP-init	Num-PP-approx	PP-init-sweep	PP-init-fit	Final-fit
$N = 3, \text{col} \in [0.0, 0.2)$	19.9	2.5	11.4	12.7	0.8203	0.9330
$N = 3, \text{col} \in [0.2, 0.4)$	49.1	18.4	35.3	7.7	0.7937	0.9991
$N = 3, \text{col} \in [0.4, 0.6)$	60.8	52.9	149.1	8.8	0.9345	0.9999
$N = 3, \text{col} \in [0.6, 0.8)$	54.8	50.1	252.1	5.7	0.9751	0.9962
$N = 3, \text{col} \in [0.8, 1.0)$	12.8	9.4	51.1	4.3	0.9940	0.9966
$N = 4, \text{col} \in [0.0, 0.2)$	20.1	3.3	2.4	13.7	0.6802	0.8235
$N = 4, \text{col} \in [0.2, 0.4)$	15.4	1.9	5.6	14.0	0.9525	0.9945
$N = 4, \text{col} \in [0.4, 0.6)$	34.0	7.5	13.5	22.6	0.9477	0.9935
$N = 4, \text{col} \in [0.6, 0.8)$	46.1	29.3	73.3	9.1	0.9365	0.9990
$N = 4, \text{col} \in [0.8, 1.0)$	47.5	26.4	62.4	6.2	0.9831	0.9963

Note: From left to right: The tensor configuration (col stands for collinearity), number of exact ALS sweeps within the PP algorithm, number of PP initialization sweeps, number of PP approximated sweeps, index of sweep when PP is first initialized (approximation begins), the fitness when PP is first initialized, and the final fitness of the experiment. All the data are the average statistics from 10 experiments.



(a) Fitness-time relation **(b)** Fitness-sweep relation

FIGURE 4 Comparison of PP and the dimension tree algorithm for CP decomposition on the quantum chemistry tensor with different variants. PP-sym/ALS-sym denotes the decomposition with symmetry constraint. PP-$\lambda = 0.7$/ALS-$\lambda = 0.7$ denotes the decomposition with step size chosen to be 0.7. (b) Detailed fitness-sweep relation for part of the sweeps. In (b), squares on the dimension tree lines represent the results per 20 sweeps (including all PP initialization, PP approximated, and ALS sweeps), and the black circles on pairwise perturbation lines represent the time when pairwise perturbation reinitializes

TABLE 3 Detailed statistics of different experiments

Tensor	Num-ALS	Num-PP-init	Num-PP-approx	Time-ALS	Time-PP-init	Time-PP-approx
Chemistry (Figure 4)	44	40	1416	0.1116	0.1655	0.0703
Coil (Figure 5a)	31	22	147	2.357	3.660	0.0648
TimeLapse (Figure 5b)	23	16	161	0.4087	0.9236	0.0562
Chemistry (Figure 7)	88	54	1358	5.338	9.608	2.254

Note: From left to right: The tensor type, number of ALS sweeps until PP experiments are finished, number of PP initialization sweeps, number of PP approximated sweeps, the average time of each ALS sweep, the average time of each PP initialization sweep, and average time of each PP approximated sweep.

PP suffers from more instability and no speed-up is achieved for this experiment. Therefore, for all other experiments, the correction terms are included as part of PP.

Figure 3d shows the speed-up distribution with different dimension size for order three tensors made by random factor matrices. It can be seen from the figure than PP achieves up to 3.0× speed-up, and PP has larger speed-ups on larger tensors, consistent with the cost analysis.

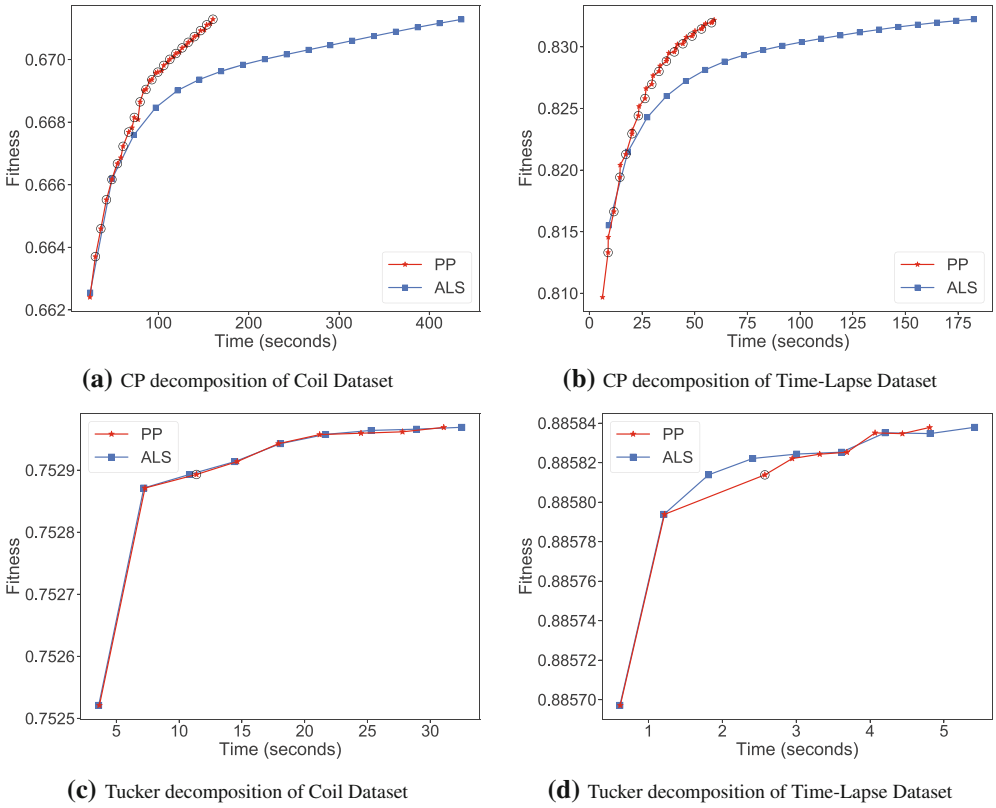
We also test the performance of PP on CP decomposition of the quantum chemistry tensor, as is shown in Figure 4, with detailed statistics shown in Table 3. In addition to the original ALS algorithm, we consider two other ALS variants for this problem: the ALS algorithm with different update step size, and the ALS algorithm with a symmetry constraint.[8] The algorithm with different update step size updates the factor matrices $\mathbf{A}^{(n)}$ based on

$$\mathbf{A}^{(n)}_{new} = (1-\lambda)\mathbf{A}^{(n)} + \lambda \mathbf{M}^{(n)} \mathbf{\Gamma}^{(n)\dagger},$$

where λ is the update step size. A good choice of λ can help achieving better convergence. The symmetry constrained algorithm considers the input tensor is symmetric in the two equidimensional modes and restricts the two factor matrices for these two modes to be the same: $\mathcal{X} = [\,[\mathbf{A}, \mathbf{B}, \mathbf{B}]\,]$. We update $\mathbf{A}$ the same as the original ALS step, and update $\mathbf{B}$ with the update step size $\lambda = 0.8$ to avoid divergence.

As is shown in Figure 4a, for all the variants of ALS algorithms, PP performs better than the dimension tree algorithm, achieving 1.25-1.52× speed-up. All the experiments are stopped after 1500 sweeps. It can also be observed in Figure 4b that PP usually restarts once approximately every 40 sweeps, and for each sweep, the fitness of both ALS and PP are almost the same, indicating that PP controls the approximation error well.

We test the performance of PP on real image datasets with NumPy in Figure 5, with detailed statistics shown in Table 3. We display the fitness and execution time for CP decomposition of the two image datasets in Figure 5a,b. We observe that



(a) CP decomposition of Coil Dataset

(b) CP decomposition of Time-Lapse Dataset

(c) Tucker decomposition of Coil Dataset

(d) Tucker decomposition of Time-Lapse Dataset

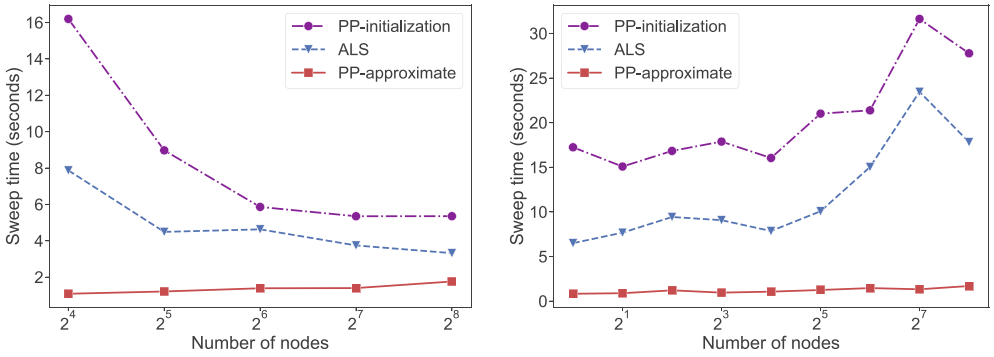
FIGURE 5 Experimental results on image datasets between pairwise perturbation and ALS for CP and Tucker decompositions. Each dot on the ALS/PP lines represents the results per 10 sweeps for CP and per sweep for Tucker decomposition (including all PP initialization, PP approximated, and ALS sweeps), and the black circles on pairwise perturbation lines represent the time when pairwise perturbation restarts

PP achieves a lower execution time for them. The speed-up for the Coil Dataset is 2.72× and for the Time-Lapse Dataset is 3.1×.

PP is also used to speedup HOOI procedure in Tucker decomposition. However, as noted in other work,[58] we observed that ALS sweeps do not significantly lower the residual beyond what is achieved by the first sweep (HOSVD). We display the fitness and the execution time for Tucker decomposition of the two real datasets in Figure 5c,d. The speed-up for the Coil Dataset is 1.05× and for the Time-Lapse Dataset is 1.13×. The reason for no obvious speed-up for the Coil Dataset is that the tensor is not equidimensional (one dimension is 7200, while others are all smaller or equal to 128). Therefore, when updating the factor matrix with a dimension of 7200, the number of operations necessary to construct the SVD input for PP are similar to that for the dimension tree Tucker algorithm. For the Time-Lapse Dataset, the tensor dimensions are more evenly distributed (two dimensions are greater than 1000), and we observe a greater speed-up. We conclude that the proposed Tucker PP algorithm performs better when used on the tensors whose dimensions are approximately equal.

5.2 | Parallel performance

We perform a parallel scaling analysis to compare the simulation time for one ALS sweep of the dimension tree algorithm to the initialization and the approximated step of the PP algorithm with Cyclops in Figure 6. Parallelism is used to accelerate the tensor contractions via calling Cyclops kernels as well as the linear system solve via calling ScaLAPACK kernels. The Cyclops library reduces each tensor contraction to a matrix multiplication. For the PP initialization step, this approach either keeps the input tensor in place, performs local multiplications, and afterwards performs a reduction on the output tensor when the rank R is small, or performs a general 3D parallel matrix multiplication when R is high. For the PP approximated step, this approach parallelizes small-sized batched matrix-vector products and result in over-parallelization. We direct readers to Reference 59 for a detailed communication cost analysis and a more communication efficient algorithm for parallel PP.



(a) Strong scaling of CP decomposition **(b)** Weak scaling of CP decomposition

FIGURE 6 Benchmark results for ALS sweeps with Cyclops, taken as the mean time across 5 sweeps

We use 8 processes per node and 8 threads per process for the benchmark experiments. The PP initialization step includes the construction of the PP operators, and is therefore much slower than the approximated steps. For strong scaling, we consider order $N = 6$ tensors with dimension $s = 50$ and rank $R = 6$ CP and Tucker decompositions. For weak scaling, on p processors, we consider order $N = 6$ tensors with dimension $s = \lfloor 32p^{1/6} \rfloor$ and rank $R = \lfloor 4p^{1/6} \rfloor$.

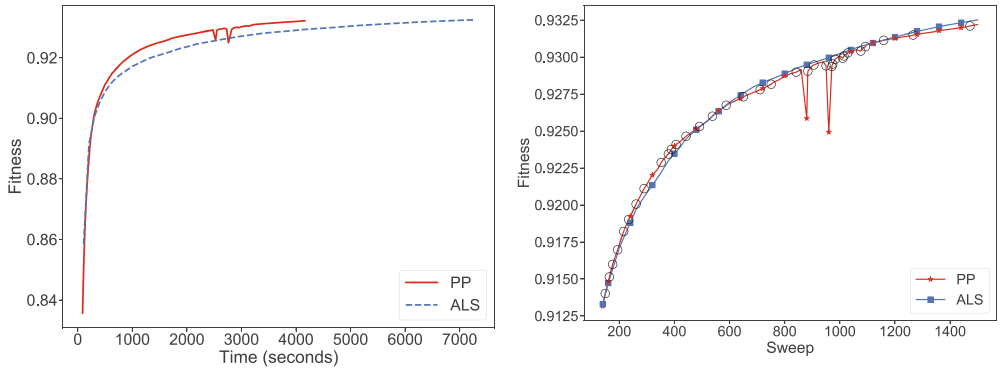
For weak scaling, Figure 6 shows that with the increase of number of nodes, the step time for all three steps increases. The approximated step time of PP is always much faster (7.8 and 10.5 times faster on 1 node and 256 nodes, respectively, compared to the dimension tree based ALS step time) than the two other steps, showing the good scalability of PP. For strong scaling, the figure shows that the approximated step time of PP increases with the number of nodes, while the two other step times decrease. The PP approximated step is much cheaper computationally and becomes dominated by communication with increasing node counts, thereby slowing down in step time. For the two other steps, the matrix calculation time will be decreased a lot with the increase of node number, thereby the step time is decreased. Overall, we observe that the potential performance benefit of PP is greater for weak scaling.

5.3 | Parallel experimental results

We test the parallel performance of PP with Cyclops on a quantum chemistry tensor. Similar to Section 5.1, we generate the order three density fitting tensor representing the compressed restricted Hartree–Fock wave function of an 40 water molecule chain system with a STO3G basis set. The generated tensor has size $4520 \times 280 \times 280$. We set the CP rank to be 1800. We show the parallel performance with Cyclops for the quantum chemistry tensor in Figure 7, with detailed statistics shown in Table 3. We perform experiments on 4 KNL nodes, leveraging 64 processors on each node. For the PP experiment, after first level contractions of the PP dimension tree, we redistribute the resulting tensor across all the processes so that it is partitioned in the rank mode, which makes the PP approximated steps faster. It can be seen that PP performs better than the dimension tree algorithm, achieving 1.75× speed-up to reach a fitness of 0.933. It can also be observed in Figure 7b that for most of the sweeps, the fitness of both the dimension tree algorithm and PP are almost the same, indicating that PP controls the approximation error well.

6 | DISCUSSIONS

One disadvantage of the standard CP-ALS algorithm is that it could be slow or has no convergence when a solution with high resolution is required, which is also called the "swamp" phenomenon.[60] Consequently, researchers have been looking at different alternatives to CP-ALS, including various regularization techniques[61,62] and line search.[47,63] These alternatives usually have higher convergence rate compared to the standard CP-ALS. We can combine some of these algorithms with PP to design algorithms with both faster convergence rate and cheaper per-sweep cost. For example, we show in Appendix B that PP can be combined with the enhanced line search (ELS-ALS) algorithm,[47] which is an effective line search algorithm on top of the standard ALS for CP decomposition, to accelerate the scheme.



(a) Fitness-time relation **(b)** Fitness-sweep relation

FIGURE 7 Comparison of PP and the dimension tree algorithm for CP decomposition on the quantum chemistry tensor with Cyclops. (b) Detailed fitness-sweep relation for part of the sweeps. In (b), squares on the dimension tree lines represent the results per 20 sweeps (including all PP initialization, PP approximated, and ALS sweeps), and the black circles on pairwise perturbation lines represent the time when pairwise perturbation reinitializes

7 | CONCLUSION

We have provided the PP algorithm for both CP and Tucker decompositions for dense tensors. The advantage of this algorithm is that it uses perturbative corrections rather than recomputing the tensor contractions to set up the quadratic optimization subproblems. Our error analysis demonstrates that it is accurate when the factor matrices change little. Specifically, our implementation of PP shows speed-ups for CP-ALS of up to 3.1× across all synthetic and application data with respect to the best known method for exact CP-ALS with the NumPy-based sequential implementation.

We leave analysis and benchmarking of PP with sparse tensors for future work. Since contraction between the input tensor and the first factor matrix requires fewer operations, there is less likely to be a benefit in using PP. Additionally, it is likely of interest to investigate more efficient adaptations of PP for non-equidimensional tensors and to experiment with alternative schemes for switching between regular ALS and PP.

ACKNOWLEDGMENTS

The authors are grateful to Daniel Kressner for pointing out the connection to the Hurwitz problem, to Fan Huang for finding the $8 \times 8 \times 8$ perfectly conditioned tensor, and to Nick Vannieuwenhoven for helpful comments. The authors were supported by the US NSF OAC SSI program, Award No. 1931258. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562. We used XSEDE to employ Stampede2 at the Texas Advanced Computing Center (TACC) through allocation TG-CCR180006.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

ENDNOTE

*All of our code is available at https://github.com/LinjianMa/tensor_decompositions.

ORCID

Linjian Ma https://orcid.org/0000-0001-7470-5415

REFERENCES

1. Kolda TG, Bader BW. Tensor decompositions and applications. SIAM Rev. 2009;51(3):455–500.
2. Grasedyck L, Kressner D, Tobler C. A literature survey of low-rank tensor approximation techniques. GAMM-Mitteilungen. 2013;36(1):53–78.

3. Cichocki A, Lee N, Oseledets I, Phan AH, Zhao Q, Mandic DP. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions. Found Trends Mach Learn. 2016;9(4-5):249–429.
4. Hao N, Horesh L, Kilmer M. Nonnegative tensor decomposition. Compressed sensing & sparse filtering. New York, NY: Springer; 2014. p. 123–48.
5. Perros I, Chen R, Vuduc R, Sun J. Sparse hierarchical Tucker factorization and its application to healthcare. Proceedings of the 2015 IEEE International Conference on Data Mining (ICDM). IEEE; 2015. p. 943–948.
6. Carroll JD, Chang JJ. Analysis of individual differences in multidimensional scaling via an N-way generalization of Eckart-Young decomposition. Psychometrika. 1970;35(3):283–319.
7. Benedikt U, Auer H, Espig M, Hackbusch W, Auer AA. Tensor representation techniques in post-Hartree–Fock methods: matrix product state tensor format. Mol Phys. 2013;111(16-17):2398–413.
8. Hummel F, Tsatsoulis T, Grüneis A. Low rank factorization of the Coulomb integrals for periodic coupled cluster theory. J Chem Phys. 2017;146(12):124105.
9. Hohenstein EG, Parrish RM, Martínez TJ. Tensor hypercontraction density fitting. I. Quartic scaling second-and third-order Møller-Plesset perturbation theory. J Chem Phys. 2012;137(4):044103.
10. Orús R. A practical introduction to tensor networks: matrix product states and projected entangled pair states. Ann Phys. 2014;349:117–58.
11. Huckle T, Waldherr K, Schulte-Herbrüggen T. Computations in quantum tensor networks. Linear Algebra Appl. 2013;438(2):750–81. Tensors and Multilinear Algebra.
12. Pazner W, Persson PO. Approximate tensor-product preconditioners for very high order discontinuous Galerkin methods. J Comput Phys. 2018;354:344–69.
13. Anandkumar A, Ge R, Hsu DJ, Kakade SM, Telgarsky M. Tensor decompositions for learning latent variable models. J Mach Learn Res. 2014;15(1):2773–832.
14. Liu J, Musialski P, Wonka P, Ye J. Tensor completion for estimating missing values in visual data. IEEE Trans Pattern Anal Mach Intell. 2013;35(1):208–20.
15. Nagy JG, Kilmer ME. Kronecker product approximation for preconditioning in three-dimensional imaging applications. IEEE Trans Image Process. 2006;15(3):604–13.
16. Karlsson L, Kressner D, Uschmajew A. Parallel algorithms for tensor completion in the CP format. Parallel Comput. 2016;57:222–34.
17. Hayashi K, Ballard G, Jiang Y, Tobia MJ. Shared-memory parallelization of MTTKRP for dense tensors. Proceedings of the ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPOPP). New York, NY: ACM; 2018. p. 393–4.
18. Chakaravarthy VT, Choi JW, Joseph DJ, Liu X, Murali P, Sabharwal Y, et al. On optimizing distributed Tucker decomposition for dense tensors. Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS). IEEE; 2017. p. 1038–47.
19. Schatz MD, Low TM, van de Geijn RA, Kolda TG. Exploiting symmetry in tensors for high performance: multiplication with symmetric tensors. SIAM J Sci Comput. 2014;36(5):C453–79.
20. Battaglino C, Ballard G, Kolda TG. A practical randomized CP tensor decomposition. SIAM J Matrix Anal Appl. 2018;39(2):876–901.
21. De Lathauwer L, De Moor B, Vandewalle J. A multilinear singular value decomposition. SIAM J Matrix Anal Appl. 2000;21(4):1253–78.
22. Tucker LR. Some mathematical notes on three-mode factor analysis. Psychometrika. 1966;31(3):279–311.
23. Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature. 2020;585(7825):357–62.
24. Li J, Choi J, Perros I, Sun J, Vuduc R. Model-driven sparse CP decomposition for higher-order tensors. Proceedings of the 2017 IEEE international parallel and distributed processing symposium (IPDPS). IEEE; 2017. p. 1048–57.
25. Kaya O, Robert Y. Computing dense tensor decompositions with optimal dimension trees. Algorithmica. 2019;81(5):2092–121.
26. Kaya O. High performance parallel algorithms for tensor decompositions. Lyon, France: Université de Lyon; 2017.
27. Phan AH, Tichavskỳ P, Cichocki A. Fast alternating LS algorithms for high order CANDECOMP/PARAFAC tensor factorizations. IEEE Trans Signal Process. 2013;61(19):4834–46.
28. Vannieuwenhoven N, Meerbergen K, Vandebril R. Computing the gradient in optimization algorithms for the CP decomposition in constant memory through tensor blocking. SIAM J Sci Comput. 2015;37(3):C415–38.
29. Ballard G, Hayashi K, Ramakrishnan K. Parallel nonnegative CP decomposition of dense tensors. Proceedings of the 2018 IEEE 25th International Conference on High Performance Computing (HiPC). IEEE; 2018. p. 22–31.
30. Solomonik E, Matthews D, Hammond JR, Stanton JF, Demmel J. A massively parallel tensor contraction framework for coupled-cluster computations. J Parallel Distrib Comput. 2014;74(12):3176–90.
31. Blackford LS, Choi J, Cleary A, D'Azeuedo E, Demmel J, Dhillon I, et al. ScaLAPACK user's guide. Philadelphia, PA: Society for Industrial and Applied Mathematics; 1997.
32. Hitchcock FL. The expression of a tensor or a polyadic as a sum of products. Stud Appl Math. 1927;6(1-4):164–89.
33. Harshman RA. Foundations of the PARAFAC procedure: models and conditions for an explanatory multimodal factor analysis. Los Angeles: University of California at Los Angeles; 1970.
34. Ballard G, Knight N, Rouse K. Communication lower bounds for matricized tensor times Khatri-Rao product. Proceedings of the 2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS). IEEE; 2018. p. 557–67.
35. Vannieuwenhoven N, Vandebril R, Meerbergen K. A new truncation strategy for the higher-order singular value decomposition. SIAM J Sci Comput. 2012;34(2):A1027–52.
36. Hackbusch W. Tensor spaces and numerical tensor calculus. Vol 42. Berlin, Germany: Springer Science & Business Media; 2012.
37. Andersson CA, Bro R. Improving the speed of multi-way algorithms: Part I. tucker3. Chemom Intell Lab Syst. 1998;42(1-2):93–103.

38. De Lathauwer L, De Moor B, Vandewalle J. On the best rank-1 and rank-(r_1, r_2, … , r_n) approximation of higher-order tensors. SIAM J Matrix Anal Appl. 2000;21(4):1324–42.
39. Oseledets IV, Tyrtyshnikov EE. Breaking the curse of dimensionality, or how to use SVD in many dimensions. SIAM J Sci Comput. 2009;31(5):3744–59.
40. Choi J, Liu X, Chakaravarthy V. High-performance dense Tucker decomposition on GPU clusters. Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis SC18. IEEE Press; 2018. p. 42.
41. Kaya O, Uçar B. High performance parallel algorithms for the Tucker decomposition of sparse tensors. Proceedings of the 2016 45th International Conference on Parallel Processing (ICPP). IEEE; 2016. p. 103–12.
42. Lim LH. Singular values and eigenvalues of tensors: a variational approach. Proceedings of the 1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, IEEE; 2005. p. 129–32.
43. Hillar CJ, Lim LH. Most tensor problems are NP-hard. J ACM. 2013;60(6):1–39.
44. Friedland S, Mehrmann V, Pajarola R, Suter SK. On best rank one approximation of tensors. Numer Linear Algebra Appl. 2013;20(6):942–55.
45. Springer P, Su T, Bientinesi P. HPTT: a high-performance tensor transposition C++ library. Proceedings of the 4th ACM SIGPLAN International Workshop on Libraries, Languages, and Compilers for Array Programming; New York, NY: ACM; 2017. p. 56–62.
46. Kolda TG, Bader BW. MATLAB tensor toolbox. Livermore, CA: Sandia National Laboratories; 2006.
47. Rajih M, Comon P, Harshman RA. Enhanced line search: a novel method to accelerate PARAFAC. SIAM J Matrix Anal Appl. 2008;30(3):1128–47.
48. Hohenstein EG, Parrish RM, Sherrill CD, Martínez TJ. Communication: tensor hypercontraction. III. least-squares tensor hypercontraction for the determination of correlated wavefunctions. J Chem Phys. 2012;137(22):221101.
49. Sun Q, Berkelbach TC, Blunt NS, Booth GH, Guo S, Li Z, et al. PySCF: the Python-based simulations of chemistry framework. Wiley Interdiscip Rev Comput Mol Sci. 2018;8(1):e1340.
50. Nene SA, Nayar SK, Murase H. Columbia object image library (coil-100). Princeton, NJ: Citeseer; 1996.
51. Zhou G, Cichocki A, Xie S. Decomposition of big tensors with low multilinear rank; 2014. arXiv preprint arXiv:14121885.
52. Nascimento SM, Amano K, Foster DH. Spatial distributions of local illumination color in natural scenes. Vis Res. 2016;120:39–44.
53. Liavas AP, Kostoulas G, Lourakis G, Huang K, Sidiropoulos ND. Nesterov-based alternating optimization for nonnegative tensor factorization: algorithm and parallel implementation. IEEE Trans Signal Process. 2017;66:944–53.
54. Khoromskij BN, Khoromskaia V. Multigrid accelerated tensor approximation of function related multidimensional arrays. SIAM J Sci Comput. 2009;31(4):3002–26.
55. Carroll JD, Pruzansky S, Kruskal JB. CANDELINC: a general approach to multidimensional analysis of many-way arrays with linear constraints on parameters. Psychometrika. 1980;45(1):3–24.
56. Khoromskij B, Khoromskaia V. Low rank tucker-type tensor approximation to classical potentials. Open Math. 2007;5(3):523–50.
57. Mohlenkamp MJ. The dynamics of swamps in the canonical tensor approximation problem. SIAM J Appl Dyn Syst. 2019;18(3):1293–333.
58. Austin W, Ballard G, Kolda TG. Parallel tensor compression for large-scale scientific data. Proceedings of the 2016 IEEE International Parallel and Distributed Processing Symposium. IEEE; 2016. p. 912–22.
59. Ma L, Solomonik E. Efficient parallel CP decomposition with pairwise perturbation and multi-sweep dimension tree. Proceedings of the 2021 IEEE International Parallel and Distributed Processing Symposium. IEEE; 2021. p. 412–21.
60. Mitchell BC, Burdick DS. Slowly converging PARAFAC sequences: swamps and two-factor degeneracies. J Chemom. 1994;8(2):155–68.
61. Navasca C, De Lathauwer L, Kindermann S. Swamp reducing technique for tensor decomposition. Proceedings of the 2008 16th European Signal Processing Conference. IEEE; 2008. p. 1–5.
62. Li N, Kindermann S, Navasca C. Some convergence results on the regularized alternating least-squares method for tensor decomposition. Linear Algebra Appl. 2013;438(2):796–812.
63. Mitchell D, Ye N, De Sterck H. Nesterov acceleration of alternating least squares for canonical tensor decomposition: momentum step size selection and restart mechanisms. Numer Linear Algebra Appl. 2020;27(4):e2297.
64. Hurwitz A. Über die Composition der quadratischen Formen von belibig vielen Variablen. Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse. 1898;1898:309–16.
65. Radon J. Lineare Scharen orthogonaler Matrizen. Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg. Volume 1. New York, NY: Springer; 1922. p. 1–14.
66. Adams JF. Vector fields on spheres. Ann Math. 1962;75(3):603–32.
67. Adams JF, Lax PD, Phillips RS. On matrices whose real linear combinations are nonsingular. Proc Am Math Soc. 1965;16(2):318–22.

How to cite this article: Ma L, Solomonik E. Accelerating alternating least squares for tensor decomposition by pairwise perturbation. Numer Linear Algebra Appl. 2022;29(4):e2431. https://doi.org/10.1002/nla.2431

APPENDIX A. ERROR BOUNDS BASED ON A TENSOR CONDITION NUMBER

We provide relative error bounds for the pairwise perturbation (PP) algorithm for both CP-ALS and Tucker-ALS for tensors that are "well-conditioned," in a sense that is defined in this appendix. However, results related to the Hurwitz problem regarding multiplicative relations of quadratic forms,[64] imply that equidimensional order three tensors can have a bounded condition number only if their dimension is $s \in \{1, 2, 4, 8\}$. We provide families of tensors with unit condition number with such dimensions. The results shed further light on the stability of MTTKRP as well as ALS, and yield nontrivial bounds for small tensors. For factorization of large tensors, the bounds proven in this section are not meaningful, since their condition number is necessarily infinite for at least one ordering of modes.

A.1 Tensor condition number

We consider a notion of tensor condition number that corresponds to a global bound on the conditioning of the multilinear vector-valued function, $g_{\mathcal{T}} : \otimes_{i=2}^{N} \mathbb{R}^{s_i} \to \mathbb{R}^{s_1}$ associated with the product of the tensor with vectors along all except the first mode,

$$g_{\mathcal{T}}\left(\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\right) = \mathcal{T} \bigtimes_{i \in \{2,\ldots,N\}} \mathbf{x}^{(i)T},$$

where $\mathcal{T}$ is contracted with $\mathbf{x}^{(i)}$ along its ith mode. The norm and condition number are given by extrema of the norm amplification of $g_{\mathcal{T}}$, which are described by the amplification function $f_{\mathcal{T}} : \otimes_{i=2}^{N} \mathbb{R}^{s_i} \to \mathbb{R}$,

$$f_{\mathcal{T}}\left(\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\right) = \frac{\left\|g_{\mathcal{T}}(\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)})\right\|_2}{\left\|\mathbf{x}^{(2)}\right\|_2 \cdots \left\|\mathbf{x}^{(N)}\right\|_2}.$$

The spectral norm of the tensor corresponds to its supremum,

$$\left\|\mathcal{T}\right\|_2 = \sup\{f_{\mathcal{T}}\}.$$

The tensor condition number can be defined as

$$\kappa(\mathcal{T}) = \sup\{f_{\mathcal{T}}\} / \inf\{f_{\mathcal{T}}\},$$

which enables quantification of the worst-case relative amplification of error with respect to input for the product of a tensor with vectors along all except the first mode. In particular, $\kappa(\mathcal{T})$ provides an upper bound on the relative norm of the perturbation of $g_{\mathcal{T}}$ with respect to the relative norm of any perturbation to any input vector.

For a matrix $\mathbf{M} \in \mathbb{R}^{s_1 \times s_2}$, if $s_1 > s_2$ the above notion of condition number gives $\kappa(\mathbf{M}) = \sigma_{\max}(\mathbf{M})/\sigma_{\min}(\mathbf{M})$ where $\sigma_{\min}(\mathbf{M})$ is the smallest singular value of $\mathbf{M}$ in the reduced SVD, while if $s_1 < s_2$, then $\kappa(\mathbf{M}) = \infty$. When tensor dimensions are unequal, the condition number is infinite if the first dimension is not the largest, so for some i, $s_i > s_1$. Aside from this condition, the ordering of modes of $\mathcal{T}$ does not affect the condition number, since for any $m > 1$, the supremum/infimum of $f_{\mathcal{T}}$ over the domain of unit vectors are for some choice of $\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m-1)}, \mathbf{x}^{(m+1)}, \ldots, \mathbf{x}^{(N)}$ the maximum/minimum singular values of

$$\mathbf{K} = \mathcal{T} \bigtimes_{i \in \{2,\ldots,m-1,m+1,\ldots,N\}} \mathbf{x}^{(i)T}.$$

A.2 Well-conditioned tensors

We provide two examples of order three tensors that have unit condition number. Other perfectly conditioned tensors can be obtained by multiplying the above tensors by an orthogonal matrix along any mode (we prove below that such transformations preserve condition number). The first example has $s_i = 2$, and yields a Givens rotation when contracted with a vector along the last mode. It is composed of two slices:

$$\begin{bmatrix} 1 & \\ & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} & 1 \\ -1 & \end{bmatrix}.$$

The second example has $s_i = 4$ and is composed of four slices:

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & -1 \end{bmatrix}, \begin{bmatrix} & 1 & & \\ -1 & & & \\ & & & 1 \\ & & 1 & \end{bmatrix}, \begin{bmatrix} & & & 1 \\ & & 1 & \\ & -1 & & \\ 1 & & & \end{bmatrix}, \begin{bmatrix} & & -1 & \\ & & & 1 \\ 1 & & & \\ & 1 & & \end{bmatrix}.$$

Finally, for $s_i = 8$, we provide an example by giving matrices $\mathbf{M}$ and $\mathbf{N}$, so that the tensor has nonzeros $\boldsymbol{\mathcal{T}}(i, j, \mathbf{M}(i, j)) = \mathbf{N}(i, j)$ for each entry in $\mathbf{M}$,

$$\mathbf{M} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 1 & 4 & 3 & 6 & 5 & 8 & 7 \\ 3 & 4 & 1 & 2 & 7 & 8 & 5 & 6 \\ 4 & 3 & 2 & 1 & 8 & 7 & 6 & 5 \\ 5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 \\ 6 & 5 & 8 & 7 & 2 & 1 & 4 & 3 \\ 7 & 8 & 5 & 6 & 3 & 4 & 1 & 2 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \end{bmatrix}, \quad \mathbf{N} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 \\ -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \end{bmatrix}.$$

The fact that the latter two tensors have unit condition number can be verified by symbolic algebraic manipulation or numerical tests.

These tensors provide solutions to special cases of the Hurwitz problem,[64] which seeks bilinear forms $z_1, \ldots, z_n$ in variables $x_1, \ldots, x_l$ and $y_1, \ldots, y_m$ such that

$$\left(x_1^2 + \cdots + x_l^2\right)\left(y_1^2 + \cdots + y_m^2\right) = z_1^2 + \cdots + z_n^2.$$

Consequently, if for $\boldsymbol{\mathcal{T}}$ and any vectors $\mathbf{x}, \mathbf{y}$,

$$\frac{\left\|\boldsymbol{\mathcal{T}} \times_2 \mathbf{x}^T \times_3 \mathbf{y}^T\right\|_2}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} = 1 \quad \Rightarrow \quad \left\|\boldsymbol{\mathcal{T}} \times_2 \mathbf{x}^T \times_3 \mathbf{y}^T\right\|_2^2 = \|\mathbf{x}\|_2^2 \|\mathbf{y}\|_2^2,$$

so we can define bilinear forms,

$$z_i = \sum_j \sum_k \boldsymbol{\mathcal{T}}(i, j, k) x_j y_k,$$

that provide a solution to the Hurwitz problem. Such equidimensional tensors with unit condition number exist for dimension $s \in \{1, 2, 4, 8\}$,[65] corresponding to the Hurwitz problem with $l = m = n = s$. However, solutions to the Hurwitz problem with $l = m = n$ cannot exist for any other dimension. Furthermore, tight bounds exist on the dimension s_3 for a tensor of dimensions $s \times s \times s_3$ to have bounded condition number ($\inf\{f_{\boldsymbol{\mathcal{T}}}\} > 0$). This problem is equivalent to finding s_3 matrices of dimension $s \times s$, such that any nonzero linear combination thereof is invertible. Factorizing $s = 2^{4a+b}c$, where $b \in \{0, 1, 2, 3\}$ and c is odd, $s_3 \leq 8a + 2^b$.[66,67]

A.3 Properties of the tensor condition number

In our analysis, we make use of the following submultiplicativity property of the tensor condition number with respect to multilinear multiplication (the property also generalizes to pairs of arbitrary order tensors contracted over one mode).

Lemma 5. *For any $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$ and matrix $\mathbf{M}$, if $\boldsymbol{\mathcal{V}} = \boldsymbol{\mathcal{T}} \times_N \mathbf{M}^T$ then $\kappa(\boldsymbol{\mathcal{V}}) \leq \kappa(\boldsymbol{\mathcal{T}})\kappa(\mathbf{M})$.*

Proof. Assume $\kappa(\boldsymbol{\mathcal{V}}) > \kappa(\boldsymbol{\mathcal{T}})\kappa(\mathbf{M})$, then there exist unit vectors $\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}$ and $\mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(N)}$ such that

$$\kappa(\mathcal{T})\kappa(\mathbf{M}) < \kappa(\mathcal{V}) = \frac{\left\|\mathcal{V}\bigtimes_{i\in\{2,\ldots,N\}}\mathbf{x}^{(i)T}\right\|_2}{\left\|\mathcal{V}\bigtimes_{i\in\{2,\ldots,N\}}\mathbf{y}^{(i)T}\right\|_2} = \frac{\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N-1\}}\mathbf{x}^{(i)T}\times_N\left(\mathbf{M}\mathbf{x}^{(N)}\right)^T\right\|_2}{\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N-1\}}\mathbf{y}^{(i)T}\times_N\left(\mathbf{M}\mathbf{y}^{(N)}\right)^T\right\|_2}.$$

Let $\mathbf{u} = \mathbf{M}\mathbf{x}^{(N)}$ and $\mathbf{v} = \mathbf{M}\mathbf{y}^{(N)}$, so $\|\mathbf{u}\|_2 / \|\mathbf{v}\|_2 \leq \kappa(\mathbf{M})$, yielding a contradiction,

$$\kappa(\mathcal{V}) \leq \frac{\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N-1\}}\mathbf{x}^{(i)T}\times_N(\mathbf{u}/\|\mathbf{u}\|_2)^T\right\|_2}{\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N-1\}}\mathbf{y}^{(i)T}\times_N(\mathbf{v}/\|\mathbf{v}\|_2)^T\right\|_2}\kappa(\mathbf{M}) \leq \kappa(\mathcal{T})\kappa(\mathbf{M}).$$

■

Applying Lemma 5 with a vector, that is, when $\mathbf{M} \in \mathbb{R}^{s_N\times 1}$ and so has condition number $\kappa(\mathbf{M}) = 1$, implies $\kappa\left(\mathcal{T}\times_N\mathbf{M}^T\right) \leq \kappa(\mathcal{T})$. By an analogous argument to the proof of Lemma 5, we can also conclude that the norm and infimum of such a product of $\mathcal{T}$ with unit vectors are bounded by those of $\mathcal{T}$, giving the following corollary.

Corollary 2. *For any $\mathcal{T} \in \mathbb{R}^{s_1\times\cdots\times s_N}$, vector $\mathbf{u} \in \mathbb{R}^{s_n}$, and any $n \in \{1,\ldots,N\}$ such that $\exists m \in \{1,\ldots,N\}$ with $s_m \geq s_n$ and $m \neq n$, if $\mathcal{V} = \mathcal{T}\times_n\mathbf{u}^T$, then $\|\mathcal{V}\|_2 \leq \|\mathbf{u}\|_2\|\mathcal{T}\|_2$, $\inf\{f_{\mathcal{V}}\} \geq \|\mathbf{u}\|_2\inf\{f_{\mathcal{T}}\}$, and $\kappa(\mathcal{V}) \leq \kappa(\mathcal{T})$.*

For an orthogonal matrix $\mathbf{M}$, Lemma 5 can be applied in both directions, namely for $\mathcal{V} = \mathcal{T}\times_N\mathbf{M}^T$ and $\mathcal{T} = \mathcal{V}\times_N\mathbf{M}$, so we observe that $\kappa(\mathcal{V}) = \kappa(\mathcal{T})$. Using this fact, we demonstrate in the following theorem that any tensor $\mathcal{T}$ can be transformed by orthogonal matrices along each mode, so that one of its fibers has norm $\|\mathcal{T}\|_2/\kappa(\mathcal{T})$.

Theorem 6. *For any $\mathcal{T} \in \mathbb{R}^{s_1\times\cdots\times s_N}$, there exist orthogonal matrices $\mathbf{Q}^{(2)}\ldots\mathbf{Q}^{(N)}$, with $\mathbf{Q}^{(i)} \in \mathbb{R}^{s_i\times s_i}$, such that $\mathcal{V} = \mathcal{T}\times_2\mathbf{Q}^{(2)}\cdots\times_N\mathbf{Q}^{(N)}$ satisfies $\kappa(\mathcal{V}) = \kappa(\mathcal{T})$, $\|\mathcal{V}\|_2 = \|\mathcal{T}\|_2$, and the first fiber of $\mathcal{V}$, that is, the vector $\mathbf{v}$ with $\mathbf{v}(i) = \mathcal{V}(i,0,\ldots,0)$, satisfies $\|\mathbf{v}\|_2 = \|\mathcal{T}\|_2/\kappa(\mathcal{T})$.*

Proof. Given a tensor $\mathcal{T}$ with infinite condition number, there must exist $N-1$ unit vectors $\mathbf{x}^{(2)},\ldots,\mathbf{x}^{(N)}$, such that $\left\|\mathcal{T}\bigtimes_{i\in\{2,\ldots,N\}}\mathbf{x}^{(i)T}\right\|_2 = \|\mathcal{T}\|_2/\kappa(\mathcal{T})$. We define $N-1$ orthogonal matrices $\mathbf{Q}^{(2)},\ldots,\mathbf{Q}^{(N)}$ such that $\mathbf{Q}^{(i)T}\mathbf{x}^{(i)} = \mathbf{e}_i$. We can then contract $\mathcal{T}$ with these matrices along the last $N-1$ modes, resulting in $\mathcal{V}$, with the same condition number as $\mathcal{T}$ (by Lemma 5) and the same norm (by a similar argument). Then, we have that the first fiber of $\mathcal{V}$ is

$$\mathbf{v} = \mathcal{V}\bigtimes_{i\in\{2,\ldots,N\}}\mathbf{e}_i^T = \mathcal{T}\bigtimes_{i\in\{2,\ldots,N\}}\mathbf{x}^{(i)T},$$

and consequently $\|\mathbf{v}\|_2 = \|\mathcal{T}\|_2/\kappa(\mathcal{T})$. ■

By Theorem 6, the condition number of a tensor is infinity if and only if it can be transformed by products with orthogonal matrices along the last $N-1$ modes into a tensor with a zero fiber. Further, any tensor $\mathcal{T}$ may be perturbed to have infinite condition number by adding to it some $\delta\mathcal{T}$ with relative norm $\|\delta\mathcal{T}\|_2 / \|\mathcal{T}\|_2 = 1/\kappa(\mathcal{T})$.

A.4 PP-CP-ALS error bound using tensor condition number

For CP decomposition, we obtain condition-number-dependent column-wise error bounds on $\mathbf{M}^{(n)}$ (the right-hand sides in the linear least squares subproblems), based on the magnitude of the relative perturbation to $\mathbf{A}^{(n)}$ since the formation of the PP operators.

Theorem 7. *If $\frac{\left\|d\mathbf{a}_k^{(n)}\right\|_2}{\left\|\mathbf{a}_k^{(n)}\right\|_2} \leq \epsilon \ll 1$ for $n \in \{1,\ldots,N\}, k \in \{1,\ldots,R\}$ and $s_m \leq s_n$ for any $m \in \{1,\ldots,N\}$, the PP algorithm without second order corrections computes $\tilde{\mathbf{M}}^{(n)}$ with column-wise error,*

$$\frac{\left\|\tilde{\mathbf{m}}_k^{(n)} - \mathbf{m}_k^{(n)}\right\|_2}{\left\|\mathbf{m}_k^{(n)}\right\|_2} = O\left(\epsilon^2\kappa(\mathcal{X})\right),$$

where $\mathbf{M}^{(n)}$ is the matrix given by a regular ALS sweep.

Proof. We bound the error due to second order perturbations in $d\mathbf{A}^{(1)}, \ldots, d\mathbf{A}^{(n)}$, by similar analysis, higher-order perturbations would lead to errors smaller by a factor of $O(\text{poly}(N)\epsilon)$ and are consequently negligible if $\epsilon \ll 1$. Consider the order four tensors $\mathcal{M}^{(i,j,n)}$ (Equation 1) based on the current factor matrices $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}$ and the PP operators $\mathcal{M}_p^{(i,j,n)}$ based on past factor matrices $\mathbf{A}_p^{(1)}, \ldots, \mathbf{A}_p^{(N)}$. The contribution of second order terms to the error is

$$\tilde{\mathbf{m}}_k^{(n)}(x) - \mathbf{m}_k^{(n)}(x) \approx \sum_{\substack{i,j \in \{1,\ldots,n-1,n+1,\ldots,N\} \\ i \neq j}} \sum_{y=1}^{s} \sum_{z=1}^{s} \mathcal{M}_p^{(i,j,n)}(x,y,z,k) d\mathbf{a}_k^{(i)}(y) d\mathbf{a}_k^{(j)}(z).$$

This absolute error has magnitude,

$$\left\| \tilde{\mathbf{m}}_k^{(n)} - \mathbf{m}_k^{(n)} \right\|_2 \leq \binom{N}{2} \max_{i,j} \left\| \mathcal{M}_p^{(i,j,n)}(:,:,:,k) \right\|_2 \left\| d\mathbf{a}_k^{(i)} \right\|_2 \left\| d\mathbf{a}_k^{(j)} \right\|_2.$$

Using the fact that for any i, j we can express $\mathbf{m}_k^{(n)}$ as

$$\mathbf{m}_k^{(n)}(x) = \sum_{y=1}^{s} \sum_{z=1}^{s} \mathcal{M}^{(i,j,n)}(x,y,z,k) \mathbf{a}_k^{(i)}(y) \mathbf{a}_k^{(j)}(z),$$

we can lower bound the magnitude of the answer with respect to any $\mathcal{M}^{(i,j,n)}$,

$$\left\| \mathbf{m}_k^{(n)} \right\|_2 \geq \inf \left\{ \left\| f_{\mathcal{M}^{(i,j,n)}(:,:,:,k)} \right\|_2 \right\} \left\| \mathbf{a}_k^{(i)} \right\|_2 \left\| \mathbf{a}_k^{(j)} \right\|_2.$$

Combining the upper bound on the absolute error with the lower bound on norm,

$$\frac{\left\| \tilde{\mathbf{m}}_k^{(n)} - \mathbf{m}_k^{(n)} \right\|_2}{\left\| \mathbf{m}_k^{(n)} \right\|_2} \leq \binom{N}{2} \max_{i,j} \frac{\left\| \mathcal{M}_p^{(i,j,n)}(:,:,:,k) \right\|_2 \left\| d\mathbf{a}_k^{(i)} \right\|_2 \left\| d\mathbf{a}_k^{(j)} \right\|_2}{\inf \left\{ \left\| f_{\mathcal{M}^{(i,j,n)}(:,:,:,k)} \right\|_2 \right\} \left\| \mathbf{a}_k^{(i)} \right\|_2 \left\| \mathbf{a}_k^{(j)} \right\|_2}.$$

Lemma 5 implies that for any i, j, k,

$$\left\| \mathcal{M}_p^{(i,j,n)}(:,:,:,k) \right\|_2 \leq \|\mathcal{X}\|_2 \prod_{l \in \{1,\ldots,N\} \setminus \{i,j,n\}} \left\| \mathbf{A}_p^{(l)}(:,k) \right\|_2$$

and that

$$\inf \left\{ \left\| f_{\mathcal{M}^{(i,j,n)}(:,:,:,k)} \right\|_2 \right\} \geq \inf \left\{ \|f_{\mathcal{X}}\|_2 \right\} \prod_{l \in \{1,\ldots,N\} \setminus \{i,j,n\}} \left\| \mathbf{A}^{(l)}(:,k) \right\|_2.$$

Since, $\left\| \mathbf{A}_p^{(l)}(:,k) \right\|_2 \leq (1+\epsilon) \left\| \mathbf{A}^{(l)}(:,k) \right\|_2$, we obtain the bound,

$$\frac{\left\| \tilde{\mathbf{m}}_k^{(n)} - \mathbf{m}_k^{(n)} \right\|_2}{\left\| \mathbf{m}_k^{(n)} \right\|_2} \leq \binom{N}{2} \kappa(\mathcal{X})(1+\epsilon)^{N-3} \epsilon^2 \approx \binom{N}{2} \kappa(\mathcal{X}) \epsilon^2.$$

■

This error bound is relative to the condition number of $\mathcal{X}$, which means the bound is sensitive to the input tensor and that the error may be unbounded if $\mathcal{X}$ has an exact CP decomposition of rank at most $\min_i s_i$.

A.5 PP-Tucker-ALS error bound using tensor condition number

For Tucker decomposition, we again obtain bounds based on the perturbation to $\mathbf{A}^{(n)}$, this time for $\mathcal{Y}^{(n)}$ (the tensors on whose matricizations a truncated SVD is performed). Using Lemma 5, we prove in Theorem 8 that when the tensor has same length in each mode and the relative error of the matrices $\mathbf{A}^{(n)}$ for $n \in \{1, \ldots, N\}$ is small, the relative error for the $\tilde{\mathcal{Y}}^{(n)}$ is also small.

Algorithm 6. ELS-ALS: Enhanced line search for CP decomposition

1: **Input:** Tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots s_N}$
2: Initialize $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ as uniformly distributed random matrices within $[0, 1]$
3: **while** not converge **do**
4: Update $\mathbf{A}^{(n)}_{\text{new}}$ based on Line 7 in Algorithm 1 for $n \in \{1, \ldots, N\}$
5: Get the ELS step size α_{ELS} based on Equation (B1)
6: $\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)} + \alpha_{\text{ELS}} \left(\mathbf{A}^{(n)}_{\text{new}} - \mathbf{A}^{(n)} \right)$ for $n \in \{1, \ldots, N\}$
7: **end whilereturn** $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$

Theorem 8. *Given an order N equidimensional tensor $\mathcal{X}$ with size s, if $\left\| d\mathbf{A}^{(n)} \right\|_2 \leq \epsilon \ll 1$ for $n \in \{1, \ldots, N\}$, the PP algorithm computes $\tilde{\mathcal{Y}}^{(n)}$ with error,*

$$\frac{\left\| \tilde{\mathcal{Y}}^{(n)} - \mathcal{Y}^{(n)} \right\|_2}{\left\| \mathcal{Y}^{(n)} \right\|_2} = O\left(\epsilon^2 \kappa(\mathcal{X})\right).$$

Proof. As in Theorem 7, we bound the error due to second-order terms,

$$\frac{\left\| \tilde{\mathcal{Y}}^{(n)} - \mathcal{Y}^{(n)} \right\|_2}{\left\| \mathcal{Y}^{(n)} \right\|_2} = \binom{N}{2} \max_{i,j} \frac{\left\| \mathcal{Y}_p^{(i,j,n)} \times_i d\mathbf{A}^{(i)T} \times_j d\mathbf{A}^{(j)T} \right\|_2}{\left\| \mathcal{Y}^{(i,j,n)} \times_i \mathbf{A}^{(i)T} \times_j \mathbf{A}^{(j)T} \right\|_2}.$$

From Lemma 5, we have

$$\frac{\left\| \mathcal{Y}_p^{(i,j,n)} \times_i d\mathbf{A}^{(i)T} \times_j d\mathbf{A}^{(j)T} \right\|_2}{\left\| \mathcal{Y}^{(i,j,n)} \times_i \mathbf{A}^{(i)T} \times_j \mathbf{A}^{(j)T} \right\|_2} \leq \frac{\left\| \mathcal{Y}_p^{(i,j,n)} \right\|_2 \left\| d\mathbf{A}^{(i)} \right\|_2 \left\| d\mathbf{A}^{(j)} \right\|_2}{\inf \left\{ \left\| f_{\mathcal{Y}^{(i,j,n)}} \right\|_2 \right\} \left\| \mathbf{A}^{(i)} \right\|_2 \left\| \mathbf{A}^{(j)} \right\|_2}.$$

Since $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$ are both matrices with orthonormal columns,

$$\frac{\left\| \tilde{\mathcal{Y}}^{(n)} - \mathcal{Y}^{(n)} \right\|_2}{\left\| \mathcal{Y}^{(n)} \right\|_2} \leq \binom{N}{2} \max_{i,j} \frac{\left\| \mathcal{Y}_p^{(i,j,n)} \right\|_2 \left\| d\mathbf{A}^{(i)} \right\|_2 \left\| d\mathbf{A}^{(j)} \right\|_2}{\inf \left\{ \left\| f_{\mathcal{Y}^{(i,j,n)}} \right\|_2 \right\}} = O\left(\epsilon^2 \kappa(\mathcal{X})\right).$$

■

APPENDIX B. COMBINING PAIRWISE PERTURBATION WITH ENHANCED LINE SEARCH

In this section, we compare the enhanced line search (ELS-ALS)[47] algorithm with an algorithm that combines pairwise perturbation with ELS (ELS-PP) for CP decomposition.

The ELS-ALS algorithm is presented in Algorithm 6. For an order N input tensor $\mathcal{X}$, the step size α_{ELS} is chosen based on

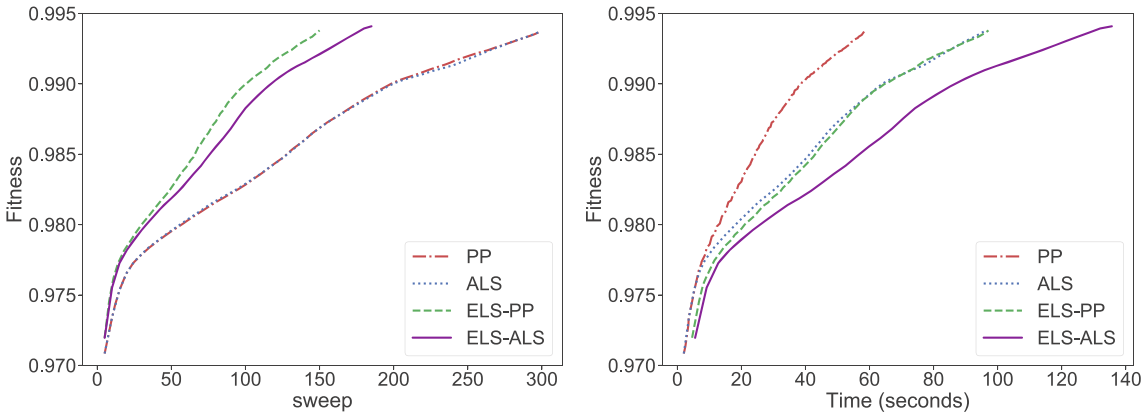
$$\alpha_{\text{ELS}} = \arg\min_{\alpha} \left\| \mathbf{X}_{(1)} - \left[\mathbf{A}^{(1)} + \alpha \left(\mathbf{A}^{(1)}_{\text{new}} - \mathbf{A}^{(1)} \right) \right] \left[\bigodot_{i=2}^{N} \mathbf{A}^{(i)} + \alpha \left(\mathbf{A}^{(i)}_{\text{new}} - \mathbf{A}^{(i)} \right) \right]^T \right\|_F^2, \tag{B1}$$

which minimizes an order $2N$ polynomial. When $N = 3$, Equation (B1) can be simplified to

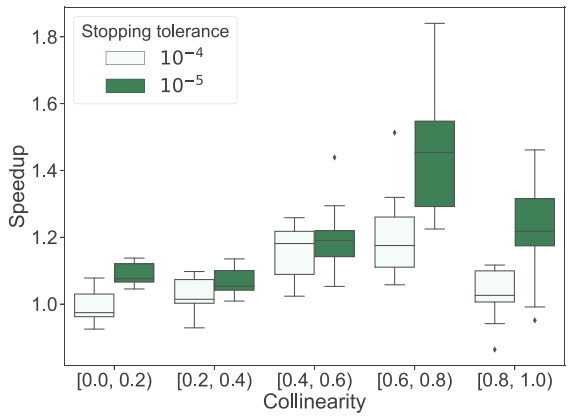
$$\alpha_{\text{ELS}} = \arg\min_{\alpha} \left\| \mathbf{X}_{(1)} - \left[\mathbf{A}^{(1)} + \alpha \left(\mathbf{A}^{(1)}_{\text{new}} - \mathbf{A}^{(1)} \right) \right] \left\{ \left[\mathbf{A}^{(2)} + \alpha \left(\mathbf{A}^{(2)}_{\text{new}} - \mathbf{A}^{(2)} \right) \right] \odot \left[\mathbf{A}^{(3)} + \alpha \left(\mathbf{A}^{(3)}_{\text{new}} - \mathbf{A}^{(3)} \right) \right] \right\}^T \right\|_F^2.$$

Algorithm 7. PP-CP-ALS: Pairwise perturbation procedure for CP-ALS

Input: tensor $\mathcal{X} \in \mathbb{R}^{s_1 \times \cdots \times s_N}$, PP tolerance $\epsilon < 1$
Initialize $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$ as uniformly distributed random matrices within $[0, 1]$, initialize $d\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)}$
while not converge **do**
  if $\forall\, i \in \{1, \ldots, N\}, \|d\mathbf{A}^{(i)}\|_F < \epsilon \|\mathbf{A}^{(i)}\|_F$ **then**
    $\mathbf{A}_p^{(n)} \leftarrow \mathbf{A}^{(n)}$, $d\mathbf{A}^{(n)} \leftarrow \mathbf{O}$ for $n \in \{1, \ldots, N\}$
    while not converge and $\forall\, i \in \{1, \ldots, N\}, \|d\mathbf{A}^{(i)}\|_F < \epsilon \|\mathbf{A}^{(i)}\|_F$ **do**
      Update $\mathbf{A}_{\text{new}}^{(n)}$ based on Line 13 in Algorithm 3 for $n \in \{1, \ldots, N\}$
      $d\mathbf{A}^{(n)} = \mathbf{A}_{\text{new}}^{(n)} - \mathbf{A}_p^{(n)}$ for $n \in \{1, \ldots, N\}$
      Get the ELS step size α based on Equation (B2)
      $\mathbf{A}^{(n)} \leftarrow \mathbf{A}_p^{(n)} + \alpha_{\text{ELS}} \left(d\mathbf{A}^{(n)}\right)$ for $n \in \{1, \ldots, N\}$
    end while
  end if
  Perform an ELS-ALS sweep as in Algorithm∼6, taking $d\mathbf{A}^{(n)} \leftarrow \alpha_{\text{ELS}} \left(\mathbf{A}_{\text{new}}^{(n)} - \mathbf{A}^{(n)}\right)$ for each $n \in \{1, \ldots, N\}$
end whilereturn $[\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}]\!]$



(a) $N = 3$, $s = R = 400$, collinearity$\in [0.6, 0.8)$ **(b)** $N = 3$, $s = R = 400$, collinearity$\in [0.6, 0.8)$



(c) $N = 3$, $s = R = 400$

FIGURE B1 (a) Fitness-sweep relation for the decomposition of one tensor with specific collinearity. (b) Fitness-time relation for the decomposition of one tensor with specific collinearity. (c) Box plot showing the speed-up of ELS-PP compared to ELS-ALS. Each box shows the 25th–75th quartiles, the median is indicated by a horizontal line inside the box, and outliers are displayed as dots

We direct readers to the reference[47] for an a detailed computational cost analysis of Equation (B1).

The ALS-PP algorithm is presented in Algorithm 7. When the algorithm is in the PP phase (satisfying the if condition in Line 4), the step size is chosen based on

$$\alpha_{\mathrm{ELS}} = \arg\min_{\alpha} \left\| \mathbf{X}_{(1)} - \left[\mathbf{A}_p^{(1)} + \alpha \left(\mathbf{A}_{\mathrm{new}}^{(1)} - \mathbf{A}_p^{(1)} \right) \right] \left[\bigodot_{i=2}^{N} \mathbf{A}_p^{(i)} + \alpha \left(\mathbf{A}_{\mathrm{new}}^{(i)} - \mathbf{A}_p^{(i)} \right) \right]^T \right\|_F^2. \tag{B2}$$

Note that the search directions are $\left(\mathbf{A}_{\mathrm{new}}^{(i)} - \mathbf{A}_p^{(i)} \right)$ for $i \in \{1, \ldots, N\}$ and w.r.t. $\mathbf{A}_p^{(i)}$ rather than $\mathbf{A}^{(i)}$, which is different from Equation (B1). Forming the explicit polynomial expression in Equation (B2) can leverage pre-computed PP operators and is cheaper.

We display the speed-ups of ELS-PP compared to ELS-ALS for synthetic tensors in Figure B1. Figure B1c shows the speed-up distribution with different exact factor matrices collinearity. We stop the algorithm when the stopping tolerance (defined as the fitness difference between two neighboring sweeps) is reached. It can be seen that ELS-PP achieves up to 1.8× speed-up, and high speed-up is achieved with tighter stopping tolerance.

Figure B1a shows that both ELS-PP and ELS-ALS have faster convergence rate compared to PP and ALS algorithms. In addition, ELS-PP converges a bit faster than ELS-ALS. Figure B1b shows that ELS-PP takes less time than ELS-ALS to reach the same final accuracy. Note that both ELS-PP and ELS-ALS take longer time compared to the standard ALS. This is consistent with the findings in Reference 47, where forming the line search polynomial could take a relatively long time.