Near-Linear Time and Fixed-Parameter Tractable Algorithms for Tensor Decompositions

Arvind V. Mahankali[1]
Stanford University, CA, USA

David P. Woodruff
Carnegie Mellon University, Pittsburgh, PA, USA

Ziyu Zhang[1]
MIT CSAIL, Cambridge, MA, USA

Abstract

We study low rank approximation of tensors, focusing on the Tensor Train and Tucker decompositions, as well as approximations with tree tensor networks and general tensor networks. As suggested by hardness results also shown in this work, obtaining $(1+\varepsilon)-$approximation algorithms for rank k tensor train and Tucker decompositions efficiently may be computationally hard for these problems. Therefore, we propose different algorithms that respectively satisfy some of the objectives above while violating some others within a bound, known as bicriteria algorithms. On the one hand, for rank-k tensor train decomposition for tensors with q modes, we give a $(1+\varepsilon)$-approximation algorithm with a small bicriteria rank ($O(qk/\varepsilon)$ up to logarithmic factors) and $O(q \cdot \mathrm{nnz}(A))$ running time, up to lower order terms. Here $\mathrm{nnz}(A)$ denotes the number of non-zero entries in the input tensor A. We also show how to convert the algorithm of [28] into a relative error approximation algorithm, but their algorithm necessarily has a running time of $O(qr^2 \cdot \mathrm{nnz}(A)) + n \cdot \mathrm{poly}(qk/\varepsilon)$ when converted to a $(1+\varepsilon)$-approximation algorithm with bicriteria rank r. Thus, the running time of our algorithm is better by at least a k^2 factor. To the best of our knowledge, our work is the first to achieve a near-input-sparsity time relative error approximation algorithm for tensor train decomposition. Our key technique is a method for efficiently obtaining subspace embeddings for a matrix which is the flattening of a Tensor Train of q tensors - the number of rows in the subspace embeddings is polynomial in q, thus avoiding the curse of dimensionality. We extend our algorithm to tree tensor networks and tensor networks on arbitrary graphs. Another way of coping with intractability is by looking at fixed-parameter tractable (FPT) algorithms. We give FPT algorithms for the tensor train, Tucker, and Canonical Polyadic (CP) decompositions, which are simpler than the FPT algorithms of [52], since our algorithms do not make use of polynomial system solvers. Our technique of using an exponential number of Gaussian subspace embeddings with exactly k rows (and thus exponentially small success probability) may be of independent interest.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Low rank approximation, Sketching algorithms, Tensor decomposition

Digital Object Identifier 10.4230/LIPIcs.ITCS.2024.79

Related Version *Full Version*: https://arxiv.org/abs/2207.07417

Funding *David P. Woodruff*: Simons Investigator Award, NIH Grant 5R01 HG 10798-2, NSF Grant CCF-1815840, and ONR Grant N00014-18-1-2562.

Acknowledgements D. Woodruff would like to thank partial support from a Simons Investigator Award.

[1] This work was done while at Carnegie Mellon University.

1 Introduction

Data dimensionality reduction has played an important role in numerical linear algebra tasks, such as regression and low rank approximation. Typically in such problems, one has as input a matrix $A \in \mathbb{R}^{n\times d}$, and applies a random linear map $S \in \mathbb{R}^{k\times n}$ for A, obtaining $S \cdot A$, where the dimension k is much smaller than n. This provides a compression of A, and often still retains many of its useful properties, e.g., the row span of SA includes a good low rank approximation to A. We refer the reader to [55] for an extensive overview of the applications of sketching for dimensionality reduction.

We investigate the role of randomized dimensionality reduction for tasks involving *tensors* – in many settings, it is more effective to represent data as a multidimensional tensor $T \in \mathbb{R}^{n\times n\times\ldots\times n}$, rather than as a matrix. This work proposes new algorithms for *tensor low-rank approximation*. In cases where data is represented as a very large matrix $A \in \mathbb{R}^{n\times d}$, significant space savings can be achieved using a low-rank approximation of A – concretely, if we find matrices $U \in \mathbb{R}^{n\times k}$ and $V \in \mathbb{R}^{k\times d}$ such that $UV \approx A$ and $k \ll n, d$, the space required to represent the information in A is $nk + kd$, which is much less than the space required for A itself. Similarly, significant space savings can be achieved with low rank tensor decompositions of a q-mode tensor $A \in \mathbb{R}^{n\times\ldots\times n}$ with n and q large. For instance, a Tensor Train decomposition of A with rank k requires $O(qnk^2)$ parameters. If k is small, this can be much fewer than the n^q parameters required to store A.

There are multiple generalizations of the concept of rank to tensors with more modes. Here, we use *mode* to refer to the number of indices required to access an entry of a tensor – thus, a matrix $A \in \mathbb{R}^{n\times n}$ has 2 modes, and a tensor $A \in \mathbb{R}^{n\times n\times n}$ has 3 modes. We frequently use the letter q to denote the number of modes that an input tensor A has. We investigate Tensor Train, Tucker, and CP decompositions, as well as general tensor networks in this work. These decompositions are introduced shortly below.

Tensor low rank approximations have the potential to allow us to handle high-dimensional data in many applications. Tucker and Tensor Train decompositions have been applied in various fields such as simulation data and robotics ([21]), machine learning model compression ([59]), scientific computing ([54]), and machine learning theory ([56]). Tensor networks are especially suited for solving high-dimensional scientific computing problems [31, 18, 30]. In particular, they have found extensive use in quantum computing ([37]). They have also been used recently to solve parametric PDEs ([19, 16]), Hamilton-Jacobi-Bellman PDE ([26, 22]), and others ([31, 33, 36]).

The CP decomposition is one notion of low-rank approximation of tensors that we study in this paper. To define the CP decomposition, we first define the outer product of two tensors: [2]

▶ **Definition 1** (Outer Product). *Let $v_1, v_2, \ldots, v_q \in \mathbb{R}^n$. Then, the **outer product** of $v_1, \ldots, v_q$, denoted $v_1 \otimes \ldots \otimes v_q \in \mathbb{R}^{n\times n\ldots\times n}$, is the q-mode tensor whose entry in the index $(i_1, \ldots, i_q)$ is $v_1(i_1)\ldots v_q(i_q)$. More generally, given two tensors $A \in \mathbb{R}^{n_1\times\cdots\times n_p}, B \in \mathbb{R}^{m_1\times\cdots m_q}$, the **tensor outer product** of A and B, denoted $A\otimes B$, is a tensor with dimensions $(n_1, \cdots, n_p, m_1, \cdots, m_q)$. The $(i_1, \cdots, i_p, j_1, \cdots, j_q)^{th}$ entry of $A \otimes B$ is $A_{i_1,\cdots,i_p}B_{j_1,\cdots,j_q}$.*

[2] We refer the interested reader to [51] for an overview of tensor decompositions.

We can now define the rank-k CP decomposition:

▶ **Definition 2** (Tensor Rank and CP Decomposition). *Let $A \in \mathbb{R}^{n\times n\times\ldots\times n}$ be a q-mode tensor. We say A has **CP rank** k if there matrices $U^1,\cdots,U^q \in \mathbb{R}^{n\times k}$ such that $A = \sum_{i=1}^k U_i^1 \otimes \cdots \otimes U_i^q$.*

For a general tensor $A \in \mathbb{R}^{n\times\ldots\times n}$, rank-$k$ CP decomposition is the problem of finding a tensor $B \in \mathbb{R}^{n\times\ldots\times n}$ with CP rank at most k such that $\|A-B\|_F$ is minimized. In other words, rank-k CP decomposition is the problem of finding

$$U_*^1, U_*^2, \ldots, U_*^q = \text{argmin}_{U^1,U^2,\ldots,U^q\in\mathbb{R}^{n\times k}} \left\| \sum_{i=1}^k U_i^1 \otimes U_i^2 \otimes \cdots \otimes U_i^q - A \right\|_F. \tag{1}$$

Here, given a tensor $A \in \mathbb{R}^{n\times\ldots\times n}$ with q modes, its Frobenius norm is defined as $\|A\|_F = \sqrt{\sum_{i_1,i_2,\ldots,i_q=1}^n A_{i_1,i_2,\ldots,i_q}^2}$.

The work of [52] previously made a significant advance on sketching for low CP rank decomposition – this work gave bicriteria and fixed parameter tractable algorithms for low CP rank factorizations under a variety of loss functions, such as the squared Frobenius norm, sum of Frobenius norms of faces, and so on. Here, by *bicriteria*, we mean that the rank of the output can be slightly larger than k, though the approximation quality is compared with the best CP rank-k approximation. When considering other notions of tensor rank, e.g. Tensor Train rank, we use the term bicriteria to mean that the Tensor Train rank of the output can be larger than k, while we compare to the best approximation of Tensor Train rank at most k.

Unfortunately, the best CP low rank approximation is difficult to define, given border rank issues [3], and even computing the CP rank is NP-hard, which rules out any relative error low rank approximation in polynomial time. Due to these issues, practical work on tensor decomposition has often studied other notions of rank, such as the Tensor Train rank [28, 46] (Matrix Product State rank) and the Tucker rank [14, 32]. We first define the Tucker rank, and the corresponding problem of low Tucker rank decomposition, using the concepts of Kronecker product and matricization of tensors:

▶ **Definition 3** (Kronecker Product). *Let $A \in \mathbb{R}^{a\times b}$ and $B \in \mathbb{R}^{c\times d}$. Then, their **Kronecker product** is the matrix $A \otimes B \in \mathbb{R}^{ac\times bd}$ whose entry in row (i,j) and column (k,l) is $A_{ik}B_{jl}$. We also occasionally denote the Kronecker product by $A \times B$.*

▶ **Definition 4** (Vectorization and Matricization of Tensors). *Let $A \in \mathbb{R}^{n\times n\times\ldots\times n}$ be a q-mode tensor. Then the **vectorization** of A, denoted $vec(A) \in \mathbb{R}^{n^q}$, is the vector whose $(1+\sum_{j=1}^q (i_j-1)n^{q-j})^{th}$ entry is $A(i_1,\ldots,i_q)$. The **mode-t matricization** of A, which we denote by $M_t(A) \in \mathbb{R}^{n\times n^{q-1}}$, is the matrix whose j^{th} row is $vec(A(:,\ldots,:,j,:,\ldots,:))$, where $A(:,\ldots,:,j,:,\ldots,:)$ denotes the slice of A whose index in the t^{th} mode is j.*

More generally, for $m < q$, the matricization $M_{i_1,\cdots,i_m}(A)$ of a q-mode tensor $A \in \mathbb{R}^{n\times n\times\ldots\times n}$ is the $n^m \times n^{q-m}$ matrix whose $(\sum_{t=1}^m j_t n^{m-t})^{th}$ row is the vectorization of the slice of A whose index in the i_t^{th} mode is j_t. For instance, given a q-mode tensor $A \in \mathbb{R}^{n\times\ldots\times n}$ for $q \geq 3$, the matricization $M_{1,2}(A)$ is the $n^2 \times n^{q-2}$ matrix whose $(j_1 n + j_2)^{th}$ row is $vec(A(j_1,j_2,\ldots))$. Recall that $A(j_1,j_2,\ldots)$ denotes the slice of A containing the entries whose entries in the first and second modes are j_1 and j_2 respectively.

[3] The border rank of a tensor A is defined as the minimum $k \in \mathbb{N}$ such that $\forall\varepsilon > 0$, there exists a tensor $A' = \sum_{i=1}^k \otimes_{j=1}^q U_i^j$ such that $\|A - A'\| < \varepsilon$. Border rank issues refer to that the CP rank of A is not necessarily equal to the border rank of A.

Thus, we can define the Tucker rank of a tensor:

▶ **Definition 5** (Tucker Rank). *Let $A \in \mathbb{R}^{n\times n\times\ldots\times n}$ be a q-mode tensor. We say A has Tucker rank at most $(k_1,\ldots,k_q)$ if there exist $U^1 \in \mathbb{R}^{n\times k_1}, U^2 \in \mathbb{R}^{n\times k_2},\ldots,U^q \in \mathbb{R}^{n\times k_q}$, and a q-mode tensor $G \in \mathbb{R}^{k_1\times k_2\times\ldots\times k_q}$, such that $A = \sum_{i_1\in[k_1],i_2\in[k_2],\ldots,i_q\in[k_q]} G(i_1,\ldots,i_q)U^1(:,i_1)\otimes\ldots\otimes U^q(:,i_q) = G\times_1 U^1\times_2\cdots\times_q U^q$, with $\times_j$ being the j^{th}-mode product. As noted in [51], this can be rewritten as $vec(A) = (U^1\times\ldots\times U^q)vec(G)$, or as $M_t(A) = U^t M_t(G)(U^1\times\ldots U^{t-1}\times U^{t+1}\times\ldots U^q)^T$. This notion of rank is also referred to in the literature as a **multilinear rank**.*

One of the main problems that we study in this paper is that of approximating an arbitrary tensor A by another tensor B which has a low Tucker rank. For simplicity, we consider a special case for the tuple $(k_1,\ldots,k_q)$ in the definition above:

▶ **Problem 6** (Tucker-(p,q) Decomposition). *Let $A \in \mathbb{R}^{n\times n\ldots\times n}$ be a q-mode tensor, and $k\in\mathbb{N}$. Then, we wish to find a q-mode tensor $B\in\mathbb{R}^{n\times n\times\ldots\times n}$ for which $\|B-A\|_F^2$ is as small as possible, subject to the constraint that B has multilinear rank at most $(k,\ldots,k,n,\ldots,n)$ (where this tuple has k in the first p coordinates and n in the last $q-p$ coordinates). In other words, the first p factors in the Tucker decomposition of B have k columns, and the last $q-p$ factors have n columns. Note that without loss of generality, this means that the last $q-p$ factors can be taken to be the $n\times n$ identity matrix. We refer to the Tucker-(q,q) decomposition problem (i.e., the case where the tensor B has multilinear rank at most $(k,k,\ldots,k)$) as the Tucker-q decomposition problem.*

We next define the Tensor Train rank, based on the concept of tensor contractions.

▶ **Definition 7** (Tensor Contraction $\circ_{i,j}$). *Let $A\in\mathbb{R}^{n_1\times\cdots\times n_p}$ be a $p-$mode tensor and $B\in\mathbb{R}^{m_1\times\cdots\times m_q}$ a $q-$mode tensor. Assuming that $n_i = m_j$, their **tensor inner product** or **tensor contraction** $A\circ_{i,j}B$ is defined as $\sum_{u=1}^{u=n_i=m_j} A_{:,\cdots,u,\cdots,:}\otimes B_{:,\cdots,u,\cdots,:}$, where u is an index for the i^{th} mode of A and j^{th} mode of B. The operation $\circ_i$ is short for $A\circ_{p,i}B$ and the operation $\circ$ is short for $A\circ_{p,1}B$. For example, when $p=q=2$, $A\circ B$ corresponds to matrix multiplication.*

Additionally, we can take the tensor contraction for several modes at once, as follows. Let $A\in\mathbb{R}^{n_1\times\ldots\times n_p}$ be a p-mode tensor and $B\in\mathbb{R}^{m_1\times\ldots\times m_q}$ a q-mode tensor. Then, given two tuples $(i_1,\ldots,i_d)$ and $(j_1,\ldots,j_d)$ of modes (and assuming $n_{i_r} = m_{j_r}$ for all $r\in[d]$), we can define the contraction $A\circ_{(i_1,\ldots,i_d),(j_1,\ldots,j_d)}B$ as $\sum_{k_1=1}^{n_{i_1}}\sum_{k_2=1}^{n_{i_2}}\cdots\sum_{k_d=1}^{n_{i_d}} A(i_1=k_1,\ldots,i_d=k_d)\otimes B(j_1=k_1,\ldots,j_d=k_d)$. Here we have used $A(i_1=k_1,\ldots,i_d=k_d)$ to denote the slice of A in which the index for mode i_1 is k_1, and so on.

▶ Remark 8. This operation is referred to in the literature as tensor contraction because it is equivalent to the self-contraction at modes $(i,p+j)$ of $A\otimes B$.

We now define Tensor Train rank, and the corresponding problem of rank-k Tensor Train decomposition, which is another of the main problems that we study in this paper.

▶ **Definition 9** (Tensor Train Rank and Tensor Train Decomposition (introduced by [46])). *Let $A\in\mathbb{R}^{n\times\ldots\times n}$ be a q-mode tensor for some $q\in\mathbb{N}$. We say A has **Tensor Train rank** $(k_1,\ldots,k_{q-1})$ for some $k_1,\ldots,k_{q-1}\in\mathbb{N}$ if there exist $U_1\in\mathbb{R}^{n\times k_1}$, $U_q\in\mathbb{R}^{k_{q-1}\times n}$, and $U_i\in\mathbb{R}^{k_{i-1}\times n\times k_i}$ for $i\in[q]\setminus\{1,q\}$, such that $A = U_1\circ\cdots\circ U_q$. We define the problem of **Tensor Train decomposition** as follows. Let $A\in\mathbb{R}^{n\times\ldots\times n}$ be a q-mode tensor for some $q\in\mathbb{N}$, and let $k\in\mathbb{N}$. Then, we wish to solve*

$$\min_{U_1\in\mathbb{R}^{n\times k},U_q\in\mathbb{R}^{k\times n},U_i\in\mathbb{R}^{k\times n\times k}} \|U_1\circ\cdots\circ U_q - A\|_F$$

In other words, we wish to find a tensor B of Tensor Train rank $(k,\ldots,k)$ such that $\|A-B\|_F$ is minimized.

Although the formulation of the Tensor Train Decomposition approximation problem in this work uses the original $q-$mode tensor A as the input, our subspace embedding and algorithm could be adapted to perform dimensionality reduction when the input is given in the Tensor Train format with a greater rank.

2 Our Contributions

As mentioned in the last section, even computing the CP rank is NP-hard. However, for both the Tensor Train and Tucker decompositions, an $O(\sqrt{q})$-relative error low rank approximation in Frobenius norm is computable in polynomial time (see, e.g., the discussion and references in [28]), thus suggesting that there are more efficient low rank approximation algorithms for these notions of rank than for the CP rank. However, as suggested by hardness results in this work, it is difficult to achieve both $(1+\varepsilon)$ relative error and rank exactly k in polynomial time. Thus, we present bicriteria algorithms in both directions.

We obtain the first polynomial-time $(1+\varepsilon)$-approximation algorithm for the Tensor Train decomposition with any non-trivial bicriteria rank. In particular, our bicriteria rank is $O(\frac{qk}{\epsilon}\log(\frac{q}{\delta}))$, where δ denotes the failure probability of the algorithm. (In the rest of the paper, unless otherwise noted, δ denotes a failure probability.)

▶ **Theorem 10** (Special Case of Theorem 26 for Tensor Train Decomposition). *Let $A \in \mathbb{R}^{n\times\ldots\times n}$ be a q-mode tensor. Then, with probability at least $1-\delta$, Algorithm 1 outputs the factors $U^1,\ldots,U^q$ of a tensor M with Tensor Train rank at most $O(\frac{qk}{\epsilon}\log(\frac{q}{\delta}))$ such that*

$$\|M-A\|_F \leq (1+\epsilon)\min_T \|T-A\|_F$$

where the minimum on the right-hand side is taken over all tensors T with Tensor Train rank at most k. The running time of our algorithm is $O(q\cdot nnz(A))$ up to lower-order terms.

Our analysis is novel compared to [28] (which proposed an $O(\sqrt{q})$-approximation algorithm for Tensor Train decomposition) as we introduce new techniques for efficiently computing subspace embeddings for matrices which are the sequential contraction of several tensors. We further discuss our analysis in Subsection 2. Compared to a tensor with Tensor Train rank k, which requires $O(qnk^2)$ parameters to express, the tensor output by our algorithm requires only $\widetilde{O}(\frac{q^3nk^2}{\epsilon^2})$ parameters. Thus, the number of parameters we obtain has an optimal dependence on n and k.

The other direction where we make contributions for these same decompositions is simpler fixed-parameter tractable algorithms for exact rank k approximations under the Tensor Train and Tucker decompositions. The work of [52] gives $(1+\varepsilon)$-approximation algorithms for the Tensor Train and Tucker decompositions, with output rank exactly k and running time $2^{\text{poly}(k/\epsilon)}$ (ignoring the dependence on q and $\text{nnz}(A)$). However, the algorithms of [52] have the following drawbacks: (1) the $\text{poly}(k/\varepsilon)$ factors in the $2^{\text{poly}(k/\varepsilon)}$ time are large and perhaps suboptimal, and (2) their algorithms run polynomial system solvers for deciding the existential theory of the reals (see, e.g., [6]), which was shown to be equivalent to deciding CP-rank [49]. Note that generic polynomial system solvers are highly complex, raising the question of whether simpler fixed-parameter tractable algorithms for the Tucker and Tensor Train decompositions can be obtained. For (1), our fixed-parameter tractable algorithm for Tensor Train decomposition improves the dependence on the $\text{poly}(k/\epsilon)$ factor in the exponent by an $O(k^2)$ factor. For (2), our algorithms for both the Tensor Train and Tucker decompositions only make use of dense Gaussian sketching matrices rather than polynomial system solvers. We note that in addition to [52], there are also works, see, e.g.,

[20, 34, 14, 15, 39, 43, 46, 28] that compute low rank approximations in polynomial time, but only provide additive error guarantees or take a prohibitive amount of time. We state the guarantees that our fixed-parameter tractable algorithms achieve:

▶ **Theorem 11.** *There is an algorithm for Tucker-(p, q) decomposition (Algorithm 2) which, given a q-mode tensor $A \in \mathbb{R}^{n\times n\times\ldots\times n}$, and $k \in \mathbb{N}$, outputs $U^1, \ldots, U^p \in \mathbb{R}^{n\times k}$ for which, with probability at least $\frac{4}{5}$,*

$$\min_G \|(U^1 \times \ldots \times U^p \times I_n \times \ldots \times I_n)vec(G) - vec(A)\|_F \\ \leq (1+\varepsilon) \min_{U^1,\ldots,U^p,G} \|(U^1 \times \ldots \times U^p \times I_n \times \ldots \times I_n)vec(G) - vec(A)\|_F \qquad (2)$$

with running time $O(p \cdot nnz(A)) + n \cdot (\frac{pk}{\varepsilon})^{O(\frac{p^2k^2\log p}{\varepsilon})}$.

▶ **Theorem 12.** *There is an algorithm for Tensor Train decomposition (Algorithm 6 presented in Subsection D.3 of the full version of this work) which, given a q-mode tensor $A \in \mathbb{R}^{n\times\ldots\times n}$, and $k \in \mathbb{N}$, outputs $U^1 \in \mathbb{R}^{n\times k}$, $U^2, \ldots, U^{q-1} \in \mathbb{R}^{k\times n\times k}$, and $U^q \in \mathbb{R}^{k\times n}$, such that with probability at least $\frac{2}{3}$,*

$$\|U^1 \circ \cdots \circ U^q - A\|_F \leq (1+\varepsilon) \min_{U^1,\ldots,U^q} \|U^1 \circ \cdots \circ U^q - A\|_F$$

with running time $O(q \cdot nnz(A)) + n \cdot poly(\frac{qk}{\varepsilon}) \cdot e^{\Theta(\frac{q^2k^2}{\varepsilon}\log(\frac{qk}{\varepsilon}))}$ and polynomial space.

▶ Remark 13. Throughout we assume our input tensor has all modes of the same dimension n. This is for presentation purposes only and our techniques can straightforwardly handle tensors where modes have differing dimensions with minor modifications.

To summarize, we propose a polynomial-time bicriteria algorithm, using a novel sketch for tensor contractions, for $(1+\varepsilon)$-approximate Tensor Train decomposition, obtaining a bicriteria rank of $O(\frac{qk}{\epsilon}\log(\frac{q}{\delta}))$. In addition, we give fixed-parameter tractable $(1+\varepsilon)$-approximation algorithms for the Tucker, Tensor Train and CP decompositions which do not depend on polynomial system solvers.

We also include fine-grained hardness results for rank-1 Tucker-$(2, 3)$ decomposition, rank-1 CP decomposition, and Tensor Train decomposition with $q = 3$, which lower-bound the optimal dependency in terms of $1/\varepsilon$ which any algorithm with output rank k can achieve. Additionally, we consider further generalizations of our bicriteria algorithm for Tensor Train decomposition, to other notions of tensor rank based on *tensor networks*. Lastly, we obtain bicriteria algorithms for Tucker-(p, q) decomposition with a robust loss function.

2.1 Bicriteria Algorithm for Tensor Train Decomposition

We state known theoretical guarantees for polynomial time Tensor Train decomposition in Table 1, along with our result. We stress that no polynomial time relative error $(1+\varepsilon)$-approximations were known – even for the case of Tensor Train decomposition, previous work either obtained additive error or $O(\sqrt{q})$-approximation.

Subspace Embeddings for Tensor Contractions

The key component in our bicriteria algorithms is a *subspace embedding for matrices which are implicitly defined in terms of tensor contractions* – in other words, we give a technique for obtaining a subspace embedding of matrices of the form $M_{\{1,\ldots,i-1\}}(U^1 \circ \cdots \circ U^{i-1}) \in \mathbb{R}^{n^{i-1}\times k}$,

Algorithm 1 $(1+\varepsilon)$-approximation algorithm for Tensor Train decomposition with output rank $t = O(\frac{qk}{\varepsilon}\log(\frac{q}{\delta}))$.

Require: $A \in \mathbb{R}^{n\times\ldots\times n}$ with q modes.
Ensure: $U^1 \in \mathbb{R}^{n\times k}$, $U^2, \ldots, U^{q-1} \in \mathbb{R}^{k\times n\times k}$, $U^q \in \mathbb{R}^{k\times n}$

// Compute U^1, processing the first mode, first
$T_1 \leftarrow$ An $O(\frac{q^3k^2}{\varepsilon^2\delta}) \times n^{q-1}$ Countsketch matrix
$R_1 \leftarrow$ A $t \times O(\frac{q^3k^2}{\varepsilon^2\delta})$ matrix whose entries are drawn i.i.d. from $\{-\frac{1}{\sqrt{t}}, \frac{1}{\sqrt{t}}\}$
$U^1 \leftarrow M_1(A)T_1^T R_1^T \in \mathbb{R}^{n\times t}$

// Store a sketch of U^1, which we denote M_1, for future use.
$S_1 \leftarrow$ An $s \times n$ Countsketch matrix, where $s = O(\frac{q^4t^2d^3}{\varepsilon^2\delta}) = O(\frac{q^6k^2d^3}{\varepsilon^4\delta}\log^2(\frac{q}{\delta}))$.
$M_1 \leftarrow S_1U^1 \in \mathbb{R}^{s\times t}$
$A \leftarrow S_1 \circ_1 A \in \mathbb{R}^{s\times n\times\ldots\times n}$, i.e. a q-mode tensor where the last $q-1$ modes have dimension n.

// Now process modes 2 through q. At the beginning of the i^{th} iteration, A has $q-i+2$ modes,
// where the first has dimension s and the rest have dimension n.
for $i = 2, \ldots, q$ **do**
  // First compute U^i using M_{i-1} which is a sketch of $U^1 \circ \cdots \circ U^{i-1}$.
  $U^i \leftarrow M_{i-1}^\dagger \circ_1 A \in \mathbb{R}^{t\times n\times\ldots\times n}$, with $q-i+2$ modes.
  if i is not q **then**
    $T_i \leftarrow$ An $O(\frac{q^3k^2}{\epsilon^2\delta}) \times n^{q-i}$ Countsketch matrix (i.e. an $O(\frac{q^3k^2}{\epsilon^2\delta}) \times n \times \ldots \times n$ tensor)
    $R_i \leftarrow$ A $t \times O(\frac{q^3k^2}{\epsilon^2\delta})$ matrix whose entries are drawn i.i.d. from $\{-\frac{1}{\sqrt{t}}, \frac{1}{\sqrt{t}}\}$
    // Note that in the following we represent R_iT_i as a $t \times n \times \ldots \times n$ tensor.
    $U^i \leftarrow U^i \circ_{(3,\ldots,q-i+2),(2,\ldots,q-i+1)} (R_iT_i) \in \mathbb{R}^{t\times n\times t}$

    // Next compute M_i
    $M_i \leftarrow M_{i-1} \circ U_i \in \mathbb{R}^{s\times n\times t}$
    $S_i \leftarrow$ An $s \times sn$ Countsketch matrix, represented as $s \times s \times n$ tensor
    $M_i \leftarrow S_i \circ_{(2,3),(1,2)} M_i \in \mathbb{R}^{s\times t}$
    $A \leftarrow S_i \circ_{(2,3),(1,2)} A \in \mathbb{R}^{s\times n\times\ldots\times n}$
  end if
end for

return $U^1, \ldots, U^q$

Table 1 Known algorithms for Tensor Train decomposition. In the algorithm of [28], p is a parameter for oversampling on the output bicriteria rank. We can extend their algorithm to a $(1+\varepsilon)$-approximation algorithm with running time $qr^2 \cdot \mathrm{nnz}(A) + n \cdot \mathrm{poly}(qk/\varepsilon)$, where r is the desired bicriteria rank. The leading order term in the running time is significantly slower than that of our algorithm.

Work	Running Time	Approximation Factor	Rank
[46] (TT-SVD)	$n^{O(q)}$	$\sqrt{q-1}$	k
[28]	$O(q((k+p)^2\mathrm{nnz}(A) + (k+p)^3n))$	$\sqrt{q-1}\left(1 + O(\sqrt{\frac{12k}{p}}) + O(\frac{e\sqrt{k+p}}{p+1})\right)$	$r = k+p$
This work	$O(q \cdot \mathrm{nnz}(A)) + n \cdot \mathrm{poly}(qk/\varepsilon)$	$(1+\varepsilon)$	$O(qk/\varepsilon \log(q/\delta))$

with $n \cdot \text{poly}(ik/\varepsilon)$ running time, without computing all $k \cdot n^{i-1}$ entries of this matrix. Recall that, as in [55], given a matrix $A \in \mathbb{R}^{n \times d}$, a $(1 \pm \varepsilon)$ subspace embedding of A is a matrix $S \in \mathbb{R}^{s \times n}$ such that $\|SAx\|_2 = (1 \pm \varepsilon)\|Ax\|_2$ for all $x \in \mathbb{R}^d$.

We can obtain such a subspace embedding for $M_{\{1,\ldots,i-1\}}(U^1 \circ \cdots \circ U^{i-1})$ as follows. We wish to construct a linear map $\mathcal{L} : \mathbb{R}^{n^{i-1}} \to \mathbb{R}^s$, for $s = \text{poly}(ik/\varepsilon)$, such that for all $x \in \mathbb{R}^k$,

$$\|\mathcal{L}M_{\{1,\ldots,i-1\}}(U^1 \circ \cdots \circ U^{i-1})x\|_2 = (1 \pm \varepsilon)\|M_{\{1,\ldots,i-1\}}(U^1 \circ \cdots \circ U^{i-1})x\|_2$$

However, for $x \in \mathbb{R}^k$, the entries of $M_{\{1,\ldots,i-1\}}(U^1 \circ \cdots \circ U^{i-1})x$ are in one-to-one correspondence with those of the $(i-1)$-mode tensor $U^1 \circ \cdots \circ U^{i-1} \circ x \in \mathbb{R}^{n \times \ldots \times n}$. We can thus significantly reduce the dimension of $U^1 \circ \cdots \circ U^{i-1} \circ x$, while preserving its norm, for all $x \in \mathbb{R}^k$, as follows. First let $S_1 \in \mathbb{R}^{s_1 \times n}$ be a $(1 \pm \varepsilon/q)$ subspace embedding for U^1 – then, for all $x \in \mathbb{R}^k$, we have the equality

$$\begin{aligned} \|S_1 \circ U^1 \circ U^2 \circ \cdots \circ U^{i-1} \circ x\|_2 &= \|S_1 U^1 M_1(U^2 \circ \cdots \circ U^{i-1} \circ x)\|_2 \\ &= (1 \pm \varepsilon/q)\|U^1 M_1(U^2 \circ \cdots \circ U^{i-1} \circ x)\|_2 \\ &= (1 \pm \varepsilon/q)\|U^1 \circ U^2 \circ \cdots \circ U^{i-1} \circ x\|_2 \end{aligned} \tag{3}$$

In addition, note that

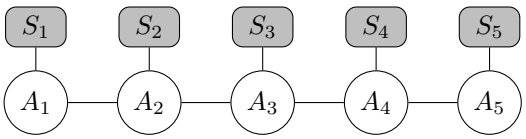


Figure 1 Illustration for Subspace Embeddings for Tensor Contractions.

$$S_1 \circ U^1 \circ \cdots \circ U^{i-1} \circ x = (S_1 U^1) \circ U^2 \circ U^3 \circ \cdots \circ U^{i-1} \circ x$$

and $S_1 U^1 \in \mathbb{R}^{s_1 \times k}$ can be computed in $O(s_1 nk)$ time, while $(S_1 U^1) \circ U^2 \in \mathbb{R}^{s_1 \times n \times k}$ can be computed in $O(s_1 nk^2)$ time. Now, define $V^2 \in \mathbb{R}^{s_1 n \times k}$ by $V^2 = M_{\{1,2\}}((S_1 U^1) \circ U^2)$ – then,

$$\|V^2 \circ U^3 \circ \cdots \circ U^{i-1} \circ x\|_2 = \|(S_1 U^1) \circ U^2 \circ U^3 \circ \cdots \circ U^{i-1} \circ x\|_2 = (1 \pm \varepsilon/q)\|U^1 \circ \cdots \circ U^{i-1} \circ x\|_2$$

and thus the next step is to apply a subspace embedding $S_2 \in \mathbb{R}^{s_2 \times n}$ to V^2. In general, we proceed iteratively – if we have computed $V^j \in \mathbb{R}^{s_{j-1} n \times k}$, such that for all $x \in \mathbb{R}^k$,

$$\|V^j \circ U^{j+1} \circ \cdots \circ U^{i-1} \circ x\|_2 = (1 \pm \varepsilon/q)^{j-1}\|U^1 \circ \cdots \circ U^{i-1} \circ x\|_2$$

then we can first compute $S_j V^j$ where $S_j \in \mathbb{R}^{s_j \times s_{j-1} n}$ is a subspace embedding for V^j, followed by computing $V^{j+1} := M_{\{1,2\}}((S_j V^j) \circ U^{j+1}) \in \mathbb{R}^{s_j \times k}$ – we then find that

$$\|V^{j+1} \circ U^{j+2} \circ \cdots \circ U^{i-1} \circ x\|_2 = (1 \pm \varepsilon/q)^{j}\|U^1 \circ \cdots \circ U^{i-1} \circ x\|_2$$

At the end of this procedure, we will have computed $V^{i-1} \in \mathbb{R}^{s_{i-2} n \times k}$ such that for all $x \in \mathbb{R}^k$,

$$\|V^{i-1} x\|_2 = (1 \pm \varepsilon/q)^{i-1}\|U^1 \circ \cdots \circ U^{i-1} \circ x\|_2$$

and if we let $W_{i-1} = S_{i-1} V^{i-1} \in \mathbb{R}^{s_{i-1} \times k}$ where $S_{i-1} \in \mathbb{R}^{s_{i-1} \times s_{i-2} n}$ is a subspace embedding matrix, we find that

$$\|W_{i-1} x\|_2 = (1 \pm O(\varepsilon))\|U^1 \circ \cdots \circ U^{i-1} \circ x\|_2$$

Table 2 Known algorithms for fixed parameter tractable $(1+\varepsilon)-$approximation for different decompositions – here TT refers to Tensor Train. For each of these decompositions, we remove the use of generic polynomial system solvers that were used in [52]. Here $s = \text{poly}(\frac{qk}{\varepsilon}) \cdot 2^{(\text{poly}(qk/\varepsilon))}$ in our Tensor Train algorithm. We note that in row 2 of this table, the n^δ factor in the running time is not related to the failure probability, but rather is due to an assumption that [52] make on the norms of the factors of the CP decomposition.

Work	Running Time	Decomposition	Rank
[52]	$q \cdot \text{nnz}(A) + n \cdot \text{poly}(qk/\varepsilon)$	CP	$O((k/\varepsilon)^{q-1})$
[52]	$(q \cdot \text{nnz}(A) + n\text{poly}(qk/\varepsilon) + 2^{O(qk^2/\varepsilon)}) \cdot n^\delta$	CP	k
[52]	$\text{nnz}(A) + n \cdot \text{poly}(k, 1/\varepsilon) + 2^{O(k^2/\varepsilon + k^3)}$	Tucker-$(3,3)$	k
[52]	$\text{nnz}(A) + n \cdot \text{poly}(k, 1/\varepsilon) + 2^{O(k^4/\varepsilon)}$	TT $(q = 3)$	k
This work	$O(p \cdot \text{nnz}(A)) + n \cdot \left(\frac{pk}{\varepsilon}\right)^{O(p^2k^2 \log p/\varepsilon)}$	Tucker-(p,q)	k
This work	$O(q \cdot \text{nnz}(A)) + n \cdot \text{poly}(\frac{qk}{\varepsilon}) \cdot 2^{\Theta(\frac{q^2k^2}{\varepsilon} \log(\frac{qk}{\varepsilon}))}$	TT	k
This work	$(q \cdot \text{nnz}(A) + n \cdot \text{poly}(k, q/\varepsilon))^{\text{poly}(kq/\varepsilon)}$	CP	k^{q-1}

and the overall computation is $O(qsnk \cdot (s+k))$. We can choose $s_j = \text{poly}(qk/\varepsilon)$, meaning the final matrix W_{i-1} has dimensions $\text{poly}(qk/\varepsilon) \times k$. We note that our subspace embedding for tensor contractions may be of independent interest, and is part of a growing body of work on obtaining subspace embeddings for matrices that are only represented implicitly [3, 50, 29, 42].

In Section B of the full version of this work, we discuss how to apply this technique to improve on previous work of [28] and obtain a bicriteria Tensor Train decomposition algorithm. We show that the algorithm of [28] can be recast as a $(1+\varepsilon)$-approximation algorithm for Tensor Train decomposition, and show how our subspace embedding can be used to reduce their running time to $O(q \cdot \text{nnz}(A))$, together with lower-order terms. We note that our analysis is significantly different from [28], since we give a technique for obtaining a subspace embedding of a matricization of a Tensor Train. In addition, our subspace embedding does not follow directly from the subspace embedding of [1] – the subspace embedding of [1] applies only to tensors with low CP rank, and would require a sketch size of at least k^q if applied to Tensor Train decomposition.

The way we have described the above subspace embedding for a tensor contraction is sequential. One could instead matricize each mode in parallel, obtaining a matrix with n rows and rank k^2 for each internal mode matricization. One could then build a binary tree of sketches, fusing and sketching two modes at a time for the internal nodes of the tree. While this could help with multiple processors, the concrete ε, k, and q factors in the bicriteria rank in our Tensor Train application in Appendix Section B of the full version of this work are a bit worse.

We extend our techniques to obtain a $(1+\epsilon)$-approximation algorithms for decomposing a tensor according to general tree networks as well. (See [10] for a survey of more general tensor networks, which generalize the Tucker and Tensor Train ranks.) We do this using a dynamic programming approach, by processing the tensors from the leaves to the root (see [23] for a prior application of this approach to tree tensor networks).

2.2 Fixed Parameter Tractable Algorithms

We give fixed parameter tractable $(1+\varepsilon)$-approximation algorithms for CP decomposition, Tucker decomposition, and Tensor Train decompositions.

The idea behind our algorithms is to generate an exponential (in k, q, and $1/\varepsilon$) number of guesses for the factor matrices and take the best solution found by sketching. Analyzing such algorithms requires understanding common primitives, such as ℓ_2-regression, subspace embeddings, and approximate matrix product, but in a regime that has not been studied before, namely, where the success probability is exponentially small. The reason the success probability is so small is that we need to choose sketching matrices with exactly k rows (as opposed to say, $\text{poly}(k/\varepsilon)$) in order to ensure that our output has rank exactly k. Since the success probability of our primitives is $2^{-\text{poly}(kq/\varepsilon)}$, by repeating $2^{\text{poly}(kq/\varepsilon)}$ times independently, one of the solutions found by sketching will provide a $(1+\varepsilon)$-approximation.

Since we just enumerate over our guesses, we avoid the polynomial system solvers used in [52]. Our algorithms are conceptually simpler and easier to implement (e.g., the algorithm proposed in [6] relies on real algebraic geometry). Our techniques may be useful for other linear algebra problems where known algorithms use polynomial system solvers, such as weighted low rank approximation [4], and non-negative matrix factorization [2, 44].

Perhaps of interest independent of our FPT algorithms is the following Theorem which shows that for the same optimal sketching dimension [11] considered in previous work for approximate regression, one can solve for an *approximate* minimizer in the sketch space, and argue this is a $(1+\varepsilon)$-approximate solution in the original space.

▶ **Theorem 14** (Strenghtened Version of Theorem 3.1 of [11]). *Let $\varepsilon, \delta, \tau \in (0,1)$. Let $A \in \mathbb{R}^{n\times d_1}$, $B \in \mathbb{R}^{n\times d_2}$, with A of rank at most k. Let $S \in \mathbb{R}^{m\times n}$ be a random matrix such that*

- *S is a $(1 \pm \frac{1}{3})$ ℓ_2 subspace embedding for the column span of A with probability at least $1-\delta$.*
- *S has the $(\sqrt{\varepsilon/k}, \delta)$-approximate matrix product property.*
- *$E[S_i^T S_j]$ is 1 if $i = j$ and 0 otherwise, where S_i denotes the i^{th} column of S.*

Then, with probability $1 - O(\delta)$, if $X^* = \text{argmin}_X \|AX - B\|_F$, then for all $\widehat{X} \in \mathbb{R}^{d_1\times d_2}$, such that $\|SA\widehat{X} - SB\|_F \leq (1+\tau)\min_X \|SAX - SB\|_F$, it holds that $\|A\widehat{X} - B\|_F \leq (1 + O(\varepsilon) + O(\tau/\delta))\|AX^* - B\|_F$. In particular, this holds if $S \in \mathbb{R}^{m\times n}$ is a matrix whose entries are each chosen from $\{-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}\}$ uniformly at random, with $m = O(k\log(1/\delta)/\varepsilon)$.

Proof. Proof is included in the full version of this work. ◀

Previous work surprisingly could only show this for solving for the exact solution in the sketch space. We use this in our Tensor Train FPT algorithm for finding candidate guesses. More details on the techniques and results for each decomposition are summarized below.

Tucker FPT Algorithm

A key fact that guarantees the success probability of our FPT Algorithm (shown in Algorithm 2) for $(1 \pm \varepsilon)$ rank-k approximation under the Tucker-(p, q) decomposition is the following lemma

▶ **Lemma 15** ($e^{-\text{poly}(k/\varepsilon)}$ Success Probability Subspace Embedding). *Let $n, k, s, t \in \mathbb{N}, \varepsilon > 0$, and $k, t < n$. Suppose $A \in \mathbb{R}^{n\times k}$ has rank k and $B \in \mathbb{R}^{n\times s}$ has rank t. Let $R \in \mathbb{R}^{k\times n}$ have i.i.d. $\mathcal{N}(0, 1/k)$ entries. If $X^* = \text{argmin}_X \|AX - B\|_F^2$ and $\widehat{X} = \text{argmin}_X \|RAX - RB\|_F^2$, then $\|A\widehat{X} - B\|_F^2 \leq (1+\varepsilon)\|AX^* - B\|_F^2$ with probability at least $e^{-\Theta(k^2\log k)}(\frac{\varepsilon}{k})^{O(kt)}$.*

Proof. Proof is included in the full version of this work. ◀

As we range over the guesses of U produced by our algorithm, at least one guess is a *good* guess, meaning it provides a $(1 \pm \frac{\varepsilon}{p^2})$ approximation. We then use an ℓ_2 Kronecker Product Regression result of [17] together with projection cost-preserving sketches [12], to efficiently evaluate the cost of each guess.

▶ **Theorem 16** (ℓ_2 Kronecker Product Regression – Theorem 3.1 and Algorithm 1 of [17])**.** *Let $A_1, A_2, \ldots, A_q$, where $A_i \in \mathbb{R}^{n_i \times d_i}$. Let $n = \prod_i n_i$ and $d = \prod_i d_i$. Let $b \in \mathbb{R}^n$. Then, there is an algorithm which, in running time $\sum_{i=1}^q nnz(A_i) + poly(d/(\varepsilon\delta))$ and with success probability $1-\delta$, returns $\widehat{x} \in \mathbb{R}^d$ such that $\|(A_1 \times \ldots \times A_q)\widehat{x} - b\|_2 \leq (1+\varepsilon)\min_x \|(A_1 \times \ldots \times A_q)x - b\|_2$, and also returns $\widehat{e} = (1 \pm \varepsilon)\|(A_1 \times \ldots \times A_q)\widehat{x} - b\|_2 - \|b\|_2$ (where we use the notation $a = (1 \pm \varepsilon)b$ to indicate that $a \in [(1-\varepsilon)b, (1+\varepsilon)b]$).*

Proof. Proof is included in the full version of this work. ◀

Tensor Train FPT Algorithm

Our FPT algorithm for Tensor Train decompositions uses similar tools as that for Tucker decompositions, but notably it uses our new subspace embedding for tensor contractions that we introduced above in the context of obtaining efficient bicriteria algorithms.

For Tensor Train and Tucker-(p, q) decompositions, our algorithm achieves output rank exactly k and $(1+\varepsilon)$ relative approximation error, in fixed-parameter tractable time. Previous work, such as [53] which uses High Order SVD, generally produces either higher than $(1+\varepsilon)$ approximation error, or higher output rank, or both. A summary of our results and previous fixed parameter tractable algorithms that achieve $(1+\varepsilon)$-approximation is presented in Table 2.

▶ Remark 17. Note that for CP rank, the best rank k approximation may not exist in general. To deal with the case where the best rank k approximation does not exist, we can add an arbitrarily small additive error term γ to the approximation guarantees, as done in [52].

CP-Rank FPT Algorithm

Finally, our toolbox allows us to obtain new FPT algorithms for CP decompositions. Our approach here is inspired by that of [52], but instead of using polynomial system solvers, uses our low probability sketching primitives. For CP decomposition, our algorithm gives a tradeoff compared to [52], as we obtain bicriteria rank k^{q-1}, which is better than their bicriteria algorithm but worse than their fixed-parameter tractable algorithm (which obtains output rank k), but our running time is fixed-parameter tractable while their bicriteria algorithm is polynomial time. As mentioned above, our algorithm does not use polynomial system solvers unlike the fixed-parameter tractable algorithm of [52]. Note that by Theorem 1.1 of [49], computing the CP rank of a tensor is equivalent to solving polynomial systems, meaning obtaining output rank k is at least as hard as solving polynomial systems, which may not be true for the Tucker and Tensor Train decompositions.

2.3 Hardness

We show new fine-grained hardness results for rank-1 Tucker-$(2, 3)$ decomposition, rank-1 CP decomposition, and Tensor Train decomposition with $q = 3$. These three are in fact equivalent decompositions. To see why, first note that rank-1 CP decomposition is equivalent to rank-1 Tensor Train decomposition since a tensor $A \in \mathbb{R}^{n \times n \times n}$ with Tensor Train rank 1 can be written as $u_1 \circ u_2 \circ u_3$ where $u_1 \in \mathbb{R}^{n \times 1}$, $u_2 \in \mathbb{R}^{1 \times n \times 1}$, and $u_3 \in \mathbb{R}^{1 \times n}$, and the

Algorithm 2 Fixed-parameter tractable algorithm for obtaining a $(1+\varepsilon)$-approximate rank k solution for Tucker-(p,q) decomposition. Here, ErrorEstimate denotes the algorithm referred to in Theorem 16 (which is the algorithm of [17] with certain parameters modified) – ErrorEstimate$(U^1,\ldots,U^p,M_j,\delta)$ returns an estimate of the error $\min_x \|(U^1\times\ldots\times U^p)x-M_j\|_2$, with failure probability δ. The variable NewError is referred to as $\widehat{e}$ in the analysis. Note that in order to output the core tensor, we would incur a n^{q-p} term in the running time. In the case $p=q$, we can also output a near-optimal core tensor, by using the algorithm of [17], while still achieving the desired running time. In addition, in the case $p=q$, it is not necessary to multiply A_p by a PCP before performing Kronecker product regression.

Require: A q-mode tensor $A\in\mathbb{R}^{n\times\ldots\times n}$, $p\le q$, $k\in\mathbb{N}$, $\varepsilon\in(0,1)$
Ensure: $U^1,U^2,\ldots,U^p\in\mathbb{R}^{n\times k}$

// Generating guesses for U^m for $m\in[p]$
$T\leftarrow(\frac{pk}{\varepsilon})^{O(\frac{pk^2\log p}{\varepsilon})}$

for $m=1\to p$ **do**
  $S_m\leftarrow$ An $O(\frac{p^3k^2}{\varepsilon^2})\times n^{q-1}$ Countsketch matrix
  $T_m\leftarrow$ An $s\times O(\frac{p^3k^2}{\varepsilon^2})$ matrix with i.i.d. $\pm\frac{1}{\sqrt{s}}$ entries, where $s=O(\frac{pk\log p}{\varepsilon})$.
  $\mathcal{S}_m\leftarrow\varnothing$
  $\widehat{A_m}\leftarrow M_m(A)S_m^TT_m^T$

  for $t=1\to T$ **do**
    $R_m\leftarrow$ A $k\times s$ matrix with i.i.d. $N(0,1/k)$ entries.
    $U_t^m\leftarrow\widehat{A_m}R_m^T$
    $\mathcal{S}_m\leftarrow\mathcal{S}_m\cup\{U_t^m\}$
  end for
end for

// Evaluating error of each tuple in $\mathcal{S}_1\times\ldots\times\mathcal{S}_p$
$A_p\leftarrow$ An $n^p\times n^{q-p}$ matrix whose $((i_1,\ldots,i_p),(i_{p+1},\ldots,i_q))^{th}$ entry is $A(i_1,\ldots,i_q)$
$S_{\text{PROJ}}\leftarrow$ An $O(k^{2p}/\varepsilon^2)\times n^{q-p}$ Countsketch matrix.
$M\leftarrow A_pS_{\text{PROJ}}^T$ ▷ Used as PCP for A_p in Kronecker product regression
$\widehat{U^1},\widehat{U^2},\ldots,\widehat{U^p}\leftarrow 0$
MinError $\leftarrow\|A\|_F^2$

for $U^1\in\mathcal{S}_1,\ldots,U^p\in\mathcal{S}_p$ **do**
  NewError $\leftarrow 0$

  for $j=1\to O(k^{2p}/\varepsilon^2)$ **do**
    $r\leftarrow O(k^{2p}/\varepsilon^2)$, the number of columns in M
    NewError $\leftarrow$ NewError + ErrorEstimate$(U^1,\ldots,U^p,M_j,\delta=\frac{1}{1000rT^p})$
  end for
  NewError $\leftarrow$ NewError $+\|M\|_F^2$

  if NewError $\le$ MinError **then**
    MinError $\leftarrow$ NewError
    $\widehat{U^1},\widehat{U^2},\ldots,\widehat{U^p}\leftarrow U^1,U^2,\ldots,U^p$
end if
  end for
  return $\widehat{U^1},\widehat{U^2},\ldots,\widehat{U^p}$

$(i,j,k)^{th}$ entry of $u_1 \circ u_2 \circ u_3$ can be written as $u_{1,i}u_{2,j}u_{3,k}$. Thus, every tensor with CP rank 1 has Tensor Train rank 1, and vice versa. Additionally, suppose a tensor $A \in \mathbb{R}^{n\times n\times n}$ can be written as a tensor with multilinear rank $(1,1,n)$ (corresponding to Tucker-$(2,3)$ decomposition). Then, we can write $A = (u_1 \times u_2 \times I)G$ for some tensor $G \in \mathbb{R}^{1\times 1\times n}$ and $u_1, u_2 \in \mathbb{R}^{n\times 1}$. Thus, $A(i,j,k) = (e_i^\top u_1 \times e_j^\top u_2 \times e_k^\top I)G = u_{1,i}u_{2,j}G(1,1,k)$ – in other words, Tucker-$(2,3)$ decomposition is equivalent to rank-1 CP decomposition. It can also be seen that the rank-1 Tucker-3 decomposition is equivalent to rank-1 CP decomposition using the same argument. Thus, using for instance that rank-1 CP decomposition is NP-hard [27, 25], all of these problems are NP-hard for $\varepsilon = 0$.

Moreover, as shown in [52], under the Exponential Time Hypothesis, there is a $2^{\Omega(\varepsilon^{1/4})}$ time lower bound for rank-1 CP decomposition and thus for all of these problems. We make a stronger assumption based on the 2-to-4 norm defined as follows.

▶ **Definition 18.** *Let $B \in \mathbb{R}^{n\times n}$. Then, we define $\|B\|_{2,4} = \left(\sum_{i=1}^n \|B_{i,:}\|_2^4\right)^{1/4}$, where $B_{i,:}$ is the i^{th} row of B, and $\|B\|_{2\to 4} = \sup_{\|x\|_2=1} \|Bx\|_4$.*

▶ **Conjecture 19.** *Any algorithm which, given a matrix $A \in \mathbb{R}^{n\times n}$, approximates $\|A\|_{2\to 4}$ to a multiplicative $O(1)$ factor, requires $2^{\Omega(n)}$ time.*

We include some justification and discussion of this conjecture below after giving our hardness result.

Running Time Lower Bound for Tucker-$(2,3)$ Decomposition

We show that under Conjecture 19, any **$(1+\varepsilon)$-approximation** algorithm for rank-1 CP, Tucker and Tensor Train decomposition requires $2^{\Omega(1/\varepsilon)}$ time.

▶ **Theorem 20.** *For convenience, given a tensor $A \in \mathbb{R}^{n\times n\times n}$, let $A^k := A(:,:,k)$. If Conjecture 19 is true, then any algorithm which, given $A \in \mathbb{R}^{n\times n\times n}$, finds unit vectors $u, v \in \mathbb{R}^n$, such that*

$$\sum_{k=1}^n \|uu^T A^k vv^T - A^k\|_F^2 \le \left(1+\Theta\left(\frac{1}{n}\right)\right) \min_{\|u_*\|_2=\|v_*\|_2=1} \sum_{k=1}^n \|u_* u_*^T A^k v_* v_*^T - A^k\|_F^2$$

requires at least $2^{\Omega(n)}$ time. Thus, any algorithm which, given $A \in \mathbb{R}^{n\times n\times n}$ and $\varepsilon \in (0,1)$, finds unit vectors $u, v \in \mathbb{R}^n$, such that

$$\sum_{k=1}^n \|uu^T A^k vv^T - A^k\|_F^2 \le (1+\epsilon) \min_{\|u_*\|_2=\|v_*\|_2=1} \sum_{k=1}^n \|u_* u_*^T A^k v_* v_*^T - A^k\|_F^2$$

requires at least $2^{\Omega(1/\varepsilon)}$ time.

This theorem implies that the rank-1 Tucker-$(2,3)$ decomposition problem requires $2^{\Omega(1/\epsilon)}$ time if $1+\epsilon$ relative error is desired. To see why, recall that (see Definition 5) the rank-1 Tucker-$(2,3)$ decomposition problem corresponds to finding $u_1 \in \mathbb{R}^n, u_2 \in \mathbb{R}^n$ and $G \in \mathbb{R}^{1\times 1\times n}$ such that $\|M_3(G)(u_1^\top \times u_2^\top) - M_3(A)\|_F^2$ is minimized. We can thus rewrite the objective as

$$\min_{g,u_1,u_2} \sum_{k=1}^n \|g_k(u_1^\top \times u_2^\top) - \mathrm{vec}(A^k)\|_F^2 = \min_{g,u_1,u_2} \sum_{k=1}^n \|g_k u_1 u_2^\top - A^k\|_F^2 \tag{4}$$

using the notation $g_k = G(1,1,k)$ and $A^k = A(:,:,k)$. Therefore, given a fixed u_1 and u_2, we have $g_k := \mathrm{argmin}_t \|tu_1u_2^\top - A^k\|_F^2$. Additionally, by the Pythagorean theorem, assuming

that u_1 and u_2 have unit norm (as otherwise, we can scale all the g_k's appropriately), we have

$$\begin{aligned}\|tu_1u_2^\top - A^k\|_F^2 &= \|tu_1u_2^\top - u_1u_1^\top A^k\|_F^2 + \|u_1u_1^\top A^k - A^k\|_F^2 \\ &= \|tu_1u_2^\top - u_1u_1^\top A^k u_2u_2^\top\|_F^2 + \|u_1u_1^\top A^k u_2u_2^\top - u_1u_1^\top A^k\|_F^2 + \|u_1u_1^\top A^k - A^k\|_F^2 \quad (5)\end{aligned}$$

where we have applied the Pythagorean theorem twice. Thus, it is optimal to have $g_k = u_1^\top A^k u_2$, and therefore, for unit vectors $u, v \in \mathbb{R}^n$, the objective

$$\sum_{k=1}^{n} \|uu^T A^k vv^T - A^k\|_F^2$$

is equivalent to the Tucker-$(2,3)$ decomposition objective. Thus, under this conjecture, our fixed-parameter tractable algorithms for the Tucker and Tensor Train decompositions are optimal in terms of their dependency on $1/\varepsilon$ in the running time.

Techniques

Our reduction from $2 \to 4$ norm uses the specific form of a Tucker-$(2,3)$ decomposition, expanding the objective function, together with the fact that for symmetric matrices A_k, it holds that $\max_{\|u\|_2=1} \sum_{k=1}^m \langle A_k u, u\rangle^2 = \max_{\|x\|_2=\|y\|_2=1} \sum_{k=1}^m \langle A_k x, y\rangle^2$. We use an inequality relating $\|A\|_{2\to 4}$ to the sum of 4-th powers of Euclidean norms of slices of A, which in turn is related to the Tucker-$(2,3)$ objective after applying this fact.

Justification for Conjecture 19

Conjecture 19 is a new conjecture which we introduce in this work. We essentially assume that the algorithm of [5] for computing the $2 \to 4$ norm of a matrix (which is a well-studied, important problem in complexity theory) has optimal running time – this is a style of argument similar to that used in other fine-grained complexity reductions.

To our knowledge, existing upper bounds for approximating $\|A\|_{2\to 4}$ up to a multiplicative $O(1)$ factor do not contradict our conjecture. Indeed, the work of [5] shows how in $2^{O(n^{1/2})}$ time one can distinguish between the cases $\|A\|_{2\to 4} \le c\sigma_{min}(A)$ and $\|A\|_{2\to 4} \ge C\sigma_{min}(A)$ if $1 < c < C$ are fixed constants, where $\sigma_{min}(A)$ is the smallest singular value of A. However, for super-constant c and C, one can show their algorithm takes $2^{\Omega(n)}$ time, and thus is no faster than enumerating over a net. [4] Thus, either our lower bounds hold, or a major breakthrough will be needed for approximating the $\|A\|_{2\to 4}$ norm. We note that such reductions are common in fine-grained complexity theory (see [7]).

[4] On page 64 of `https://arxiv.org/pdf/1205.4484.pdf` it is mentioned that if $\dim(V) \le C^2 n^{2/q}$ which is $C^2 n^{1/2}$ for $q = 4$, then brute force enumeration can be used and this would take time exponential in $\dim(V)$. Note that the goal is to determine whether $\|A\|_{2\to 4} \ge C\sigma$ where σ is the least singular value of A. If no restrictions are placed on C, then it could be larger than $n^{1/4}$, since for a fixed v, $\frac{\|Av\|_4}{\|Av\|_2}$ could be as large as $n^{1/4}$, and moreover, $\frac{\|Av\|_2}{\|v\|_2}$ could be arbitrarily larger than σ, meaning $\frac{\|Av\|_4}{\|v\|_2}$ could be larger than $n^{1/4}\sigma$. Thus, brute-force enumeration would take $2^{O(n)}$ time, in this case. If brute-force enumeration is not used, then Corollary 10.2 of that work can be used to obtain the bound $\|V\|_{2->4} \ge \sqrt{\dim(V)}/n^{1/4}$. Since $\dim(V) \le n$, this bound is at best $n^{1/4}$, while C could be larger. Thus, the algorithm of [5] can take exponential time if no restrictions are placed on C and c.

2.4 Robust Loss for Tucker Decomposition

As studied in [52], another question is whether it is possible to obtain low rank decompositions with good relative error for loss functions which are more robust to outliers than the Frobenius norm. The fact that the Frobenius norm is not robust to outliers can be seen from the following simplified example in the setting of ℓ_2 regression. Suppose we are given 4 pairs (x_1, y_1), (x_2, y_2), (x_3, y_3) and (x_4, y_4), where the x_i are in $\mathbb{R}^2$ and $y_1 = 1$, $y_2 = -1$, $y_3 = 10$ and $y_4 = 2$, and we wish to find a weight vector $w \in \mathbb{R}^2$ such that $\sum_{i=1}^4 (w^\top x_i - y_i)^2$. Then, for example when $w = 0$, the contribution of the data point (x_3, y_3) to the loss will be much larger than that of the other data points. In general, this pattern holds: if a tensor has certain very large entries, then performing low rank approximation using the Frobenius norm can lead to those entries being fit at the expense of other entries. A number of works attempt to address this question, see, e.g., [52, 8] for work which looks at the ℓ_1-norm and sum of Euclidean norm losses. We propose a new algorithm for Tucker decomposition where the loss function is the sum of the Frobenius norms of the faces, where a face is a subtensor obtained by fixing the index along the last mode. We note that [52] only consider robust loss functions for the CP decomposition.

We consider the following norm $\|\cdot\|_R$, which is less sensitive to outliers than the Frobenius norm:

▶ **Definition 21** (Sum of Frobenius Norms of Mode-q Faces). *Let $A \in \mathbb{R}^{n \times \cdots \times n}$ be a q-mode tensor. Then, $\|A\|_R = \sum_{i \in [n]} \|A(:, \ldots, :, i)\|_F$, where $A(:, \ldots, :, i)$ is the slice of A whose index in the last mode is i. In other words, $\|A\|_R$ is the sum of the Frobenius norms of the faces along the last mode.*

$\|\cdot\|_R$ is a high dimensional generalization of the $\ell_{2,1}$ norm for matrices. $\ell_{2,1}$ norm has been frequently adopted over the Frobenius norm for many optimization and maching learning tasks. For example, [57] uses $\ell_{2,1}$ norm for discrimitive feature selection.

▶ **Problem 22** (Robust Tucker-(p, q) Decomposition). *Let $A \in \mathbb{R}^{n \times \cdots \times n}$ be a q-mode tensor, with $p < q$, and let $k \in \mathbb{N}$. Then, we wish to find a q-mode tensor $X \in \mathbb{R}^{n \times \cdots \times n}$ of multilinear rank at most $(k, \ldots, k, n, \ldots, n)$ (where the first p entries of the tuple are k and the last $q - p$ entries are n), such that $\|X - A\|_R$ is minimized.*

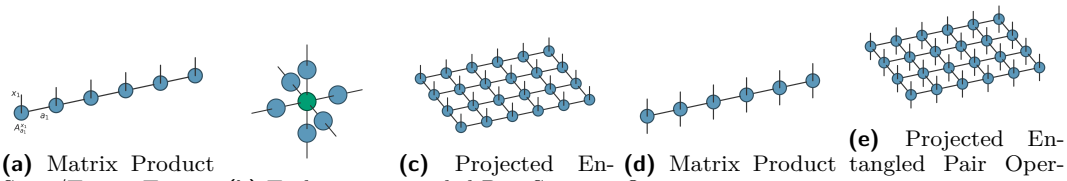
We give an $O(p \cdot (\sqrt{k} \log k + \sqrt{\log(np)}))$-approximation algorithm with bicriteria rank $O(k \log^2 k)$ – we formally state our result in Appendix Section F of the full version of this work. We make the assumption that $p < q$ only for this robust loss function – all of our results for Frobenius norm Tucker decomposition hold even if $p = q$. We make this assumption since it allows us to apply techniques for $\ell_{1,2}$-norm matrix low rank approximation - different techniques might be required if $p = q$.

▶ Remark 23. In all of our results, unidentifiability does not affect our guarantees, since we are not recovering the ground truth. Instead, we compare the error to the best tensor which can be represented according to the given network with a specified rank.

2.5 Generalization: Tensor Network Approximation

We extend our results summarized in Subsubsection 2.1 to general tensor networks, which we define informally below. See Appendix Subsection A.1 of the full version of this work for a formal definition.

▶ **Definition 24** (General Tensor Network Contraction – Informal). *Let $\mathcal{G} = (V, E)$ be a graph. For each $v \in V$, suppose U_v is a tensor with $\deg(v)$ modes of dimension k (corresponding to each edge incident to v) and one mode of dimension n. We define the contraction of $\mathcal{G}$ to be*



(a) Matrix Product State/Tensor Train. **(b)** Tucker. **(c)** Projected Entangled Pair State. **(d)** Matrix Product Operator. **(e)** Projected Entangled Pair Operator.

Figure 2 Popular Tensor Networks.

the tensor obtained by contracting U_u and U_v for every edge $(u, v) \in \mathcal{G}$, until we are left with a single vertex. For convenience, we denote this contraction by $\mathcal{G}(\{U_v \mid v \in V\})$. We will informally refer to k as the rank of the network according to $\mathcal{G}$.

The Tensor Train decomposition is a special case of tensor networks with $\mathcal{G}$ being a line graph. We first extend our Tensor Train decomposition algorithm to obtain a bicriteria $(1+\varepsilon)-$approximation algorithm for the special case of tree network decomposition.

▶ **Theorem 25** (Theorem 26 – Informal)**.** *Let $A \in \mathbb{R}^{n \times \ldots \times n}$ be a q-mode tensor and $\mathcal{T}$ be a tree. Then, with probability at least $1-\delta$, our algorithm outputs a tree network M of rank $O(\frac{qk}{\epsilon}\log(\frac{q}{\delta}))$, such that*

$$\|M - A\|_F \leq (1+\epsilon)\min_T \|T - A\|_F$$

where the minimum on the right-hand side is taken over all tensors T with rank at most k according to $\mathcal{T}$. The running time of our algorithm is $O(q \cdot nnz(A)) + n \cdot (\frac{qk}{\epsilon\delta})^{O(d)}$.

We also show how to apply the tree network decomposition algorithm to approximate tensors with low ranks under a particular graph $\mathcal{G}$ which may have cycles. Many such network-based decompositions have been used in different applications.

▶ **Theorem 26.** *Let $k, q, n \in \mathbb{N}$, and let A and $\mathcal{T}$ be as specified in Algorithm 3. Then, Algorithm 3 finds $\{U_v \mid v \in V\}$ with bicriteria rank $t = O(\frac{qk}{\varepsilon}\log(\frac{q}{\delta}))$ such that*

$$\|\mathcal{T}(\{U_v \mid v \in V\}) - A\|_F \leq (1+\varepsilon)\min_{U_v} \|\mathcal{T}(\{U_v \mid v \in V\}) - A\|_F$$

with probability at least $1-\delta$. The running time of Algorithm 3 is $O(q \cdot nnz(A)) + n \cdot (\frac{qk}{\varepsilon\delta})^{O(d)}$.

Tensors with Low General Tensor Network Rank

We show how to find a binary tree $\mathcal{T}$ on a vertex set V' containing $O(q)$ nodes, and corresponding tensors U_v, again with a mode corresponding to each edge $e \in T$ of dimension $\text{poly}(k^{\deg(G)\text{tw}(G)}q/\varepsilon)$, for which

$$\|\mathcal{T}(\{U_v \mid v \in V'\}) - A\|_F^2 \leq (1+\varepsilon) \cdot \|\mathcal{G}(\{U_v^* \mid v \in V\}) - A\|_F^2,$$

in $O(\text{nnz}(A)q + n \cdot \text{poly}(k^{\deg(G)\text{tw}(G)}q/\varepsilon)$ time, where $\text{tw}(G)$ is the treewidth of G, $\deg(G)$ is the maximum degree of a vertex, and $\text{nnz}(A)$ is the number of non-zero entries of A. Although our output tensor $\mathcal{T}(\{U_v \mid v \in V'\})$ does not have the same topology as the original network $\mathcal{G}$, it performs as well, up to a $(1+\varepsilon)$-factor, as the best tensor with network topology $\mathcal{G}$, and moreover, the number of parameters in the tree network $\mathcal{T}$ is at most a factor $k^{O(\deg(G)\text{tw}(G))}$ larger than that in $\mathcal{G}$. The overall tree network will have $q \cdot n \cdot k^{O(\deg(G)^2\text{tw}(G)^2)}$ parameters, instead of n^q.

Algorithm 3 $(1+\varepsilon)$-approximation algorithm for tree network decomposition with output rank $O(\frac{qk}{\varepsilon}\log(\frac{q}{\delta}))$. Note that we can assume each of the leaves in $\mathcal{T}$ has one mode of dimension n – if a leaf v has no such mode, then U_v is simply a k-dimensional vector, and can be contracted with its parent and ignored for the purposes of approximating A.

Require: $A \in \mathbb{R}^{n\times\ldots\times n}$ with q modes, and a tree $\mathcal{T} = (V, E)$ with $O(q)$ vertices, such that for any $u, v \in V$ with $(u, v) \in E$, the edge between u and v has rank k, and each vertex in V has at most one mode of dimension n. Without loss of generality, each leaf in V has one mode of dimension n. All vertices of $\mathcal{T}$ have degree at most $d \in \mathbb{N}$.

Ensure: $\{U_v \mid v \in V\}$ such that for each $v \in V$, the dimension of U_v corresponding to each of its outgoing edges in $\mathcal{T}$ is at most $t = O(\frac{qk}{\varepsilon}\log(q/\delta))$

// Process the leaves first
$L \leftarrow$ A list of all the leaves of $\mathcal{T}$
for $v \in L$ **do**
  $T_v \leftarrow$ An $O(\frac{q^3k^2}{\varepsilon^2\delta}) \times n^{q-1}$ Countsketch matrix
  $R_v \leftarrow$ A $t \times O(\frac{q^3k^2}{\varepsilon^2\delta})$ matrix whose entries are drawn i.i.d. from $\{-\frac{1}{\sqrt{t}}, \frac{1}{\sqrt{t}}\}$
  $i \leftarrow$ the mode of U_v of dimension n
  $\widetilde{U_v} \leftarrow M_i(A)T_v^T R_v^T \in \mathbb{R}^{n\times t}$

  // Store a sketch of $\widetilde{U_v}$, which we denote M_v, for future use.
  $S_v \leftarrow$ An $s \times n$ Countsketch matrix, where $s = O(\frac{q^4t^2d^3}{\varepsilon^2\delta}) = O(\frac{q^6k^2d^3}{\varepsilon^4\delta}\log^2(\frac{q}{\delta}))$.
  $M_v \leftarrow S_v\widetilde{U_v} \in \mathbb{R}^{s\times t}$
  $A \leftarrow S_v \circ_i A$
end for

// Now process each vertex $v \in V$ after processing its entire subtree.
$\mathcal{I} \leftarrow$ A list of the vertices in $V \setminus L$ where each vertex appears after the rest in its subtree.
for $v \in \mathcal{I}$ **do**
  // First compute $\widetilde{U_v}$ using the sketches of the subtrees of the children of v.
  $M_{v-subtree} \leftarrow \bigotimes_u M_u \in \mathbb{R}^{s^{\deg(u)-1}\times t^{\deg(u)-1}}$, where u ranges over the children of v
  $\mathcal{G}_1 \leftarrow$ The modes of A corresponding to row dimensions of M_u for the children u of v
  $\mathcal{G}_2 \leftarrow$ The remaining modes of A
  $\widetilde{U_v} \leftarrow M_{v-subtree}^{\dagger} \circ_{\mathcal{G}_1} A$
  if v is not the root of $\mathcal{T}$ **then**
    $T_v \leftarrow$ An $O(\frac{q^3k^2}{\varepsilon^2\delta}) \times n^{|\mathcal{G}_2|}$ Countsketch matrix
    $R_v \leftarrow$ A $t \times O(\frac{q^3k^2}{\varepsilon^2\delta})$ matrix whose entries are drawn i.i.d. from $\{-\frac{1}{\sqrt{t}}, \frac{1}{\sqrt{t}}\}$
    $\widetilde{U_v} \leftarrow R_vT_v \circ_{\mathcal{G}_2} \widetilde{U_v} \in \mathbb{R}^{t\times\ldots\times t\times t}$
  end if

  // Next, compute M_v.
  $M_v \leftarrow M_{v-subtree} \circ \widetilde{U_v} \in \mathbb{R}^{s^{\deg(u)-1}\times t}$ where the contraction is along the modes of $\widetilde{U_v}$ corresponding to the edges between u and its children
  $S_v \leftarrow$ An $s \times s^{\deg(u)-1}$ Countsketch matrix.
  $M_v \leftarrow S_vM_v$
  $A \leftarrow S_v \circ_{\mathcal{G}_1} A$
end for

return $\{\widetilde{U_v} \mid v \in V\}$

The main idea of our algorithm is that, given a graph $\mathcal{G} = (V, E)$ and a corresponding set of factors $\{U_v \mid v \in V\}$ where the ranks on the edges in E are all of rank k, there exists a binary tree network $\mathcal{T}$ with corresponding factors $\{W_v \mid v \text{ is a vertex of } \mathcal{T}\}$ which is equivalent, i.e.,

$$\mathcal{T}(\{W_v \mid v \text{ is a vertex of } \mathcal{T}\}) = \mathcal{G}(\{U_v \mid v \in V\})$$

Crucially, $\mathcal{T}$ also has low rank. Specifically, the ranks on the edges of $\mathcal{T}$ are at most $k^{\deg(\mathcal{G})\mathrm{tw}(\mathcal{G})}$. This is shown in Appendix Subsection C.1 in the full version of this work.

Using the observation above, we construct the binary tree $\mathcal{T}$ by contracting the edges of $\mathcal{G}$ in a particular order, and each vertex w resulting from the contraction of an edge (u, v) has two children u' and v' corresponding to u and v. In order for the edges of $\mathcal{T}$ to have low rank, it is crucial for the degree of the vertices w obtained from contraction to be small. For this to hold, we make use of the following result from [41]: there is an order in which we can contract the edges of $\mathcal{G}$, such that the largest degree of any vertex at any point during the contraction is at most $O(\deg(\mathcal{G})\mathrm{tw}(\mathcal{G}))$. Moreover, this order can be computed in $\mathrm{poly}(|V|)e^{O(\deg(\mathcal{G})\mathrm{tw}(\mathcal{G}))}$ time. The technical details of the algorithm are presented in Appendix Section C of the full version of this work.

3 Comparison to Prior Work

The study of fast algorithms for tensor decompositions is a vast subject with work in numerical linear algebra, scientific computing, and theoretical computer science. We discuss several other approaches to these problems below, but we emphasize the main differences with our work here:

1. There is no previous work for general tensor network low rank approximation, despite work that motivates studying this question [58].
2. We obtain relative error $(1 + \varepsilon)$-approximations, whereas, with the exception of [52], previous work either gets an additive error or at best a fixed constant approximation factor (e.g. [28] and [23] which obtain $O(\sqrt{q})$ approximation algorithms).
3. Previous works do not run in $\mathrm{nnz}(A) \cdot q$ time, but rather at least $\mathrm{nnz}(A)\mathrm{poly}(kq)$ time (such as [28] which takes at least $O(qs^2\mathrm{nnz}(A))$ time where s is their output rank, and [23] for Hierarchical Tucker decomposition which requires $O(n^q)$ time due to the use of SVD).

In addition to the above, a major contribution of our work is to introduce many of the advanced and recent techniques from randomized numerical linear algebra to the study of more advanced tensor decompositions. We note that some such tools have been developed, such as TensorSketch [47] and its optimizations [1], but they do not directly apply to Tensor Train decompositions, for example. In this work, we develop new algorithmic tools, such as sparse affine and subspace embeddings for tensors represented implicitly, such as those represented as a sequence of $\circ$ operations, as well as new analytical tools such as the propagation of JL-moment guarantees across the nodes of a tree network. We note that the idea of implicit tensor maps is seen in prior works such as [35]. However, the context and goal are vastly different, the map in [35] is specific to a subspace, where ours is an embedding. [38] that has a similar goal in developing tensor embeddings does not present an end-to-end algorithm and guarantees for approximate tensor decomposition.

Besides [52], there are some prior works on tensor decompositions using the same family of techniques as ours, such as [53] with Tucker-factor-wise sketches, [48] with randomized projections, [39] with subspace embeddings for tensors, and [40] with leverage score sampling.

Table 3 Comparison of algorithms for the Tucker decomposition. ω is the matrix multiplication complexity exponent.

Work	Running Time	Approximation Factor
[23]	$O(n^{\frac{3}{2}q})$	$\sqrt{q}$
[53]	$O\left(\left(\frac{s(1-s/n)^q}{1-s/n}+qk\right)n^q\right)$	$2q$
[43]	$O(\sum_{j=1}^{q}(k^j n^{q-j+1}+k^j n^{q-j}))$	$O(\sqrt{q})$
This work	$O(q\cdot \text{nnz}(A))+n\cdot(qk/\varepsilon)^{O(q^2k^2\log q\frac{1}{\varepsilon})}$	$1+\varepsilon$

Several algorithms using more standard regression techniques have been developed for tensor decompositions. This includes Alternative Least Squares algorithms such as [20], and SVD-like algorithms such as [43], [46], [28], and [34]. These works are generally significantly more computationally expensive, or have worse guarantees, or both. In particular, they do not achieve $(1+\varepsilon)-$approximation within a practical time complexity.

In [13] which studies the related problem of Tensor Train rounding, a Tensor Train embedding similar to ours is proposed – the embedding consists of a Tensor Train network where each factor has i.i.d. Gaussian entries. This work also proposes the "right-to-left partial $\circ$" operation to quickly apply this embedding to another Tensor Train. The work [13] does not analyze the error incurred using this embedding in their algorithms, and the embedding is not used as a subspace embedding or affine embedding. Instead, it is used to approximately perform the QR decomposition of matrices of the form $T_{X,n}T_{Y,n+1:N}$, where Y is a Tensor Train of length N given as the input to the Tensor Train rounding algorithm, and $T_{Y,n+1:N}$ is a matrix formed from cores $n+1$ through N of Y. Cores $n+1$ through N of the Gaussian Tensor Train network are applied to $T_{Y,n+1:N}$, and the column span of the resulting matrix is computed.

The concurrent and independent [9] follows a different line of techniques towards $\circ$ based decompositions. Compared to ours, their Tensor Ring decomposition result makes several additional assumptions, notably $\forall i,j,k\in[q]|T_{i,j,k}-Tr(U_i^*U_jU_k^*)|<=\eta$ and the analogous second order constraint for the Sum-of-Squares technique. Their error guarantee is $\text{poly}(n,k,\max_{i\in q}||U_i^*||_F,1/m)\eta^c$ over component-wise parameter distance (m is a lower bound on the condition number of the fused matrix of an arbitrary combination of two modes in the optimal solution), but each output component is of rank k. To remove the n dependency in the error guarantee, they need additional smoothed analysis assumptions. Our work takes a different approach to achieve a bicriteria algorithm and gets $(1+\varepsilon)$ relative error guarantees with a comparable runtime.

We note that whether one can efficiently contract tensor networks with cycles is a question that has been studied in quantum physics and quantum computation (e.g., [41, 24, 45], to name a few examples). To our knowledge, the problem of approximating a given input tensor A by a general tensor network of low rank has not been extensively studied. However, [58] gives examples where the rank of a tensor A with respect to one tensor network can be significantly lower than the rank with respect to another. We also note that hardness results for contracting PEPS networks (e.g., [24]) suggest that either converting PEPS networks to equivalent tree networks is hard to do efficiently, or that an equivalent tree network would require a much larger rank.

References

1 Thomas D. Ahle, Michael Kapralov, Jakob Bæk Tejs Knudsen, Rasmus Pagh, Ameya Velingker, David P. Woodruff, and Amir Zandieh. Oblivious sketching of high-degree polynomial kernels. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 141–160. SIAM, 2020. doi:10.1137/1.9781611975994.9.

2 Sanjeev Arora, Rong Ge, Ravi Kannan, and Ankur Moitra. Computing a nonnegative matrix factorization - provably. *SIAM J. Comput.*, 45(4):1582–1611, 2016. doi:10.1137/130913869.

3 Haim Avron, Vikas Sindhwani, and David P. Woodruff. Sketching structured matrices for faster nonlinear regression. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2994–3002, 2013.

4 Frank Ban, David P. Woodruff, and Qiuyi (Richard) Zhang. Regularized weighted low rank approximation. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4061–4071, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/90aef91f0d9e7c3be322bd7bae41617d-Abstract.html.

5 Boaz Barak, Fernando GSL Brandao, Aram W Harrow, Jonathan Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 307–326, 2012.

6 Saugata Basu, Richard Pollack, and Marie-Françoise Roy. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of the ACM*, 43(6):1002–1045, 1996.

7 Karl Bringmann. Fine-grained complexity theory (tutorial). In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, volume 126 of *LIPIcs*, pages 4:1–4:7. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.STACS.2019.4.

8 Dimitris G. Chachlakis, Ashley Prater-Bennette, and Panos P. Markopoulos. L1-norm tucker tensor decomposition. *CoRR*, abs/1904.06455, 2019. arXiv:1904.06455.

9 Sitan Chen, Jerry Li, Yuanzhi Li, and Anru R. Zhang. Learning polynomial transformations, 2022. doi:10.48550/arXiv.2204.04209.

10 Andrzej Cichocki, Namgil Lee, Ivan V. Oseledets, Anh Huy Phan, Qibin Zhao, and Danilo P. Mandic. Low-rank tensor networks for dimensionality reduction and large-scale optimization problems: Perspectives and challenges PART 1. *CoRR*, abs/1609.00893, 2016. arXiv:1609.00893.

11 Kenneth L. Clarkson and David P. Woodruff. Numerical linear algebra in the streaming model. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 205–214. ACM, 2009. doi:10.1145/1536414.1536445.

12 Michael B. Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 163–172, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2746539.2746569.

13 Hussam Al Daas, Grey Ballard, Paul Cazeaux, Eric Hallman, Agnieszka Miedlar, Mirjeta Pasha, Tim W. Reid, and Arvind K. Saibaba. Randomized algorithms for rounding in the tensor-train format, 2021. doi:10.48550/arXiv.2110.04393.

14 Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000. doi:10.1137/S0895479896305696.

15 Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-(r1 ,r2 ,. . .,rn) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000. doi:10.1137/S0895479898346995.

16 A Dektor, A Rodgers, and D Venturi. Rank-adaptive tensor methods for high-dimensional nonlinear pdes. *J. Sci. Comput.*, 88:36, 2021.

17 Huaian Diao, Rajesh Jayaram, Zhao Song, Wen Sun, and David P. Woodruff. Optimal sketching for kronecker product regression and low rank approximation. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4739–4750, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/0c215f194276000be6a6df6528067151-Abstract.html.

18 S. V. Dolgov, B. N. Khoromskij, and I. V. Oseledets. Fast solution of parabolic problems in the tensor train/quantized tensor train format with initial application to the fokker–planck equation. *SIAM Journal on Scientific Computing*, 34(6):A3016–A3038, 2012.

19 Sergey Dolgov, Boris N. Khoromskij, Alexander Litvinenko, and Hermann G. Matthies. Polynomial chaos expansion of random coefficients and the solution of stochastic partial differential equations in the tensor train format. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1):1109–1135, 2015.

20 Matthew Fahrbach, Mehrdad Ghadiri, and Thomas Fu. Fast low-rank tensor decomposition by ridge leverage score sampling, 2021. arXiv:2107.10654.

21 Alex Gorodetsky, Sertac Karaman, and Youssef Marzouk. High-dimensional stochastic optimal control using continuous tensor decompositions. *The International journal of robotics research*, 37(2-3), 2018.

22 Alex Gorodetsky, Sertac Karaman, and Youssef Marzouk. High-dimensional stochastic optimal control using continuous tensor decompositions. *The International Journal of Robotics Research*, 37(2-3):340–377, 2018.

23 Lars Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. Appl.*, 31(4):2029–2054, 2010. doi:10.1137/090764189.

24 Jonas Haferkamp, Dominik Hangleiter, Jens Eisert, and Marek Gluza. Contracting projected entangled pair states is average-case hard. *Physical Review Research*, 2(1), January 2020. doi:10.1103/physrevresearch.2.013010.

25 C. J. Hillar and L.-H. Lim. Most tensor problems are np-hard. *ACM 60*, 6(45), 2013. doi:10.1145/2512329.

26 M K Horowitz, A Damle, and J W Burdick. Linear Hamilton Jacobi Bellman equations in high dimensions. *In 53rd IEEE Conference on Decision and Control*, pages 5880–5887, 2014.

27 Johan Håstad. Tensor rank is np-complete. *Journal of Algorithms*, 11(4):644–654, 1990. doi:10.1016/0196-6774(90)90014-6.

28 Benjamin Huber, Reinhold Schneider, and Sebastian Wolf. *A Randomized Tensor Train Singular Value Decomposition*, pages 261–290. Springer International Publishing, Cham, 2017. doi:10.1007/978-3-319-69802-1_9.

29 Rajesh Jayaram, Alireza Samadian, David P. Woodruff, and Peng Ye. In-database regression in input sparsity time. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 4797–4806. PMLR, 2021.

30 Vladimir A. Kazeev and Boris N. Khoromskij. Low-rank explicit qtt representation of the laplace operator and its inverse. *SIAM Journal on Matrix Analysis and Applications*, 33(3):742–758, 2012.

31 Boris N. Khoromskij. Tensors-structured numerical methods in scientific computing: Survey on recent advances. *Chemometrics and Intelligent Laboratory Systems*, 110(1):1–19, 2012. doi:10.1016/j.chemolab.2011.09.001.

32 Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

33 Katharina Kormann. A semi-lagrangian vlasov solver in tensor train format. *SIAM Journal on Scientific Computing*, 37(4):B613–B632, 2015.

34 Lingjie Li, Wenjian Yu, and Kim Batselier. Faster tensor train decomposition for sparse data. *CoRR*, abs/1908.02721, 2019. arXiv:1908.02721.

35 Allen Liu and Jerry Li. Clustering mixtures with almost optimal separation in polynomial time. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, pages 1248–1261, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519935.3520012.

36 Michael Lubasch, Pierre Moinier, and Dieter Jaksch. Multigrid renormalization. *Journal of Computational Physics*, 372:587–602, 2018. doi:10.1016/j.jcp.2018.06.065.

37 Ningyi Lyu, Micheline B. Soley, and Victor S. Batista. Tensor-train split-operator ksl (tt-soksl) method for quantum dynamics simulations. *Journal of Chemical Theory and Computation*, 18(6):3327–3346, 2022. PMID: 35649210. doi:10.1021/acs.jctc.2c00209.

38 Linjian Ma and Edgar Solomonik. Cost-efficient gaussian tensor network embeddings for tensor-structured inputs, 2022. doi:10.48550/arXiv.2205.13163.

39 Osman Asif Malik and Stephen Becker. Low-rank tucker decomposition of large tensors using tensorsketch. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/file/45a766fa266ea2ebeb6680fa139d2a3d-Paper.pdf.

40 Osman Asif Malik and Stephen Becker. A sampling-based method for tensor ring decomposition. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 7400–7411. PMLR, 18–24 July 2021. URL: https://proceedings.mlr.press/v139/malik21b.html.

41 Igor L. Markov and Yaoyun Shi. Simulating quantum computation by contracting tensor networks. *SIAM J. Comput.*, 38(3):963–981, 2008. doi:10.1137/050644756.

42 Raphael A. Meyer, Cameron Musco, Christopher Musco, David P. Woodruff, and Samson Zhou. Fast regression for structured inputs. *CoRR*, abs/2203.07557, 2022.

43 Rachel Minster, Arvind K Saibaba, and Misha E Kilmer. Randomized algorithms for low-rank tensor decompositions in the tucker format. *SIAM journal on mathematics of data science*, 2(1):189–215, 2020.

44 Ankur Moitra. An almost optimal algorithm for computing nonnegative rank. *SIAM J. Comput.*, 45(1):156–173, 2016. doi:10.1137/140990139.

45 Bryan O'Gorman. Parameterization of tensor network contraction. *arXiv preprint arXiv:1906.00013*, 2019.

46 Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

47 Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–247, 2013.

48 Beheshteh Rakhshan and Guillaume Rabusseau. Tensorized random projections. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3306–3316. PMLR, 26–28 August 2020. URL: https://proceedings.mlr.press/v108/rakhshan20a.html.

49 Marcus Schaefer and Daniel Stefankovic. The complexity of tensor rank. *CoRR*, abs/1612.04338, 2016. arXiv:1612.04338.

50 Xiaofei Shi and David P. Woodruff. Sublinear time numerical linear algebra for structured matrices. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4918–4925. AAAI Press, 2019.

51 Nicholas D. Sidiropoulos, Lieven De Lathauwer, Xiao Fu, Kejun Huang, Evangelos E. Papalexakis, and Christos Faloutsos. Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, 65(13):3551–3582, July 2017. doi:10.1109/tsp.2017.2690524.

52 Zhao Song, David P. Woodruff, and Peilin Zhong. Relative error tensor low rank approximation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '19, pages 2772–2789, USA, 2019. Society for Industrial and Applied Mathematics.

53 Yiming Sun, Yang Guo, Charlene Luo, Joel Tropp, and Madeleine Udell. Low-rank tucker approximation of a tensor from streaming data. *SIAM Journal on Mathematics of Data Science*, 2(4):1123–1150, 2020. doi:10.1137/19M1257718.

54 Jeheon Woo, Woo Youn Kim, and Sunghwan Choi. System-specific separable basis based on tucker decomposition: Application to density functional calculations. *Journal of Chemical Theory and Computation*, 18(5):2875–2884, 2022. PMID: 35437014. doi:10.1021/acs.jctc.1c01263.

55 David P. Woodruff. Sketching as a tool for numerical linear algebra. *Found. Trends Theor. Comput. Sci.*, 10(1-2):1–157, 2014. doi:10.1561/0400000060.

56 Wenbin Yang, Zijia Wang, Jiacheng Ni, Qiang Chen, and Zhen Jia. A low-rank tensor bayesian filter framework for multi-modal analysis. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 3738–3742, 2022. doi:10.1109/ICIP46576.2022.9897228.

57 Yi Yang, Heng Tao Shen, Zhigang Ma, Zi Huang, and Xiaofang Zhou. L2,1-norm regularized discriminative feature selection for unsupervised learning. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1589–1594. AAAI Press, 2011.

58 Ke Ye and Lek-Heng Lim. Tensor network ranks, 2018. doi:10.48550/arXiv.1801.02662.

59 Miao Yin, Yang Sui, Siyu Liao, and Bo Yuan. Towards efficient tensor decomposition-based dnn model compression with optimization framework. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10669–10678, 2021.