FastRank: Fast Tensor Rank Approximation Based on Spectral Energy

Konstantinos Bougiatiotis[1, 2] **Georgios Paliouras**[1]
[1]Institute of Informatics and Telecommunications, National Center Scientific Research "Demokritos", Greece
[2]Department of Informatics and Telecommunications, National and Kapodistrian University, Greece

Abstract

Complex multi-dimensional data are often represented as tensors, analyzed through tensor decompositions. A central challenge is selecting the right number of components for the decomposition. In the Canonical Polyadic Decomposition (CPD), this means determining the canonical rank, which directly impacts decomposition quality. Existing methods typically estimate rank by repeatedly computing CPDs, an expensive process. We introduce *FastRank*, a theoretically grounded method that estimates rank without CPD computation. By applying Singular Value Decomposition (SVD) to a sum-reduced matrix of the tensor and analyzing its eigenspectrum, FastRank achieves over 1000× speedup and surpasses state-of-the-art accuracy. We validate it using both synthetic and real data, including noisy settings, and highlight its scalability in knowledge graph completion, where prior methods fail due to computational limitations.

1 INTRODUCTION

Tensors, or multi-dimensional arrays, are a natural and powerful abstraction for representing complex, structured data. They are increasingly used across a broad spectrum of machine learning and data mining applications [Ji et al., 2019], including recommender systems [Frolov and Oseledets, 2017], signal processing [Sidiropoulos et al., 2017], neuroscience [Williams et al., 2018], and graph analysis [Balažević et al., 2019]. Among the many operations that can be performed on tensors, *tensor decompositions* have become foundational tools for uncovering latent structure in such data. This is largely because many real-world datasets exhibit substantial redundancy and are often well-approximated by low-rank models, as they lie near low-dimensional subspaces. This empirical observation underpins the success of low-rank tensor decompositions [Kolda and Bader, 2009, Sidiropoulos et al., 2017].

A critical parameter in most tensor decomposition methods is the *rank* of the tensor, which informally refers to the minimum number of components needed to accurately reconstruct the original data. Determining the correct rank is essential: a rank that is too low may lead to underfitting and the loss of important structure, while an excessively high rank risks overfitting and increased computational cost. The rank corresponding to the commonly-used Canonical Polyadic Decomposition (CPD), known as the *canonical rank*, is especially important due to its interpretability and widespread applicability [Bro and Kiers, 2003, Kolda and Bader, 2009]. However, estimating the rank of a tensor is a computationally challenging task. It has been shown that determining the exact rank of a tensor (a similar problem) is NP-hard [Hillar and Lim, 2013]. As a result, the literature features a variety of heuristic methods for approximating tensor rank, including CORCONDIA-based diagnostics [Bro and Kiers, 2003], AutoTen [Papalexakis, 2016] for sparse tensors, NORMO [Tsitsikas and Papalexakis, 2020] which aims to identify and remove redundancy, and several Bayesian approaches [Zhao et al., 2015]. However, most of these techniques rely on repeatedly computing the expensive CPD, making them prohibitively slow or impractical for large-scale tensors. Furthermore, many require either prior knowledge of an upper bound on the rank or considerable human intervention. These limitations underscore the need for faster

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

and more scalable alternatives.

In this paper, we propose *FastRank*[1], a novel and highly efficient method for tensor rank estimation that circumvents tensor decomposition entirely. Our approach is grounded in a mathematically principled idea: we apply a sum-reduction operation to the tensor, transforming it into a two-dimensional matrix, and then estimate the rank of the original tensor using the spectral energy of this matrix. Building on this insight, we design a simple algorithm capable of handling noisy and real-world data by thresholding the singular values, based on the spectral energy of the matrix. We evaluate FastRank across multiple tensor datasets and also demonstrate its utility in guiding a knowledge graph completion model. FastRank is exceptionally efficient, achieving over 1000× speedup compared to the current state-of-the-art, while also providing more accurate estimates. Notably, in the knowledge graph completion use case, FastRank is the only method capable of producing an estimate within seconds, whereas other existing methods fail to scale to such data.

To summarize, the main contributions of this work are as follows:

1. *Novel theoretical formulation*: We show that our sum-collapse operation yields a matrix whose rank is upper-bounded by the rank of the original tensor, and we leverage its spectral energy to estimate the tensor's rank.

2. *FastRank algorithm*: Building on our theoretical insights, we develop a simple yet effective algorithm to handle noisy and real-world data within this framework. The method requires no tensor decomposition and does not depend on an upper-bound estimate.

3. *Extensive evaluation*: We benchmark FastRank against state-of-the-art methods using synthetically generated tensors (with and without noise), real tensors with known rank, and a knowledge graph completion task. We demonstrate that FastRank is over 1000× faster than the best competing method, while also achieving higher accuracy.

2 BACKGROUND

Here, we introduce the notation used throughout the paper, provide background on tensor decomposition, and formally state the problem we address. We denote tensors, matrices, vectors, and scalars by $\mathcal{X}$, $\mathbf{X}$, $\mathbf{x}$, and x, respectively. For a third-order tensor $\mathcal{X}$, we use $\mathcal{X}[i,j,k]$ or $\mathcal{X}_{ijk}$ to denote the entry at position (i,j,k) along the first, second, and third modes, respectively. In this work, our focus will be on third-order tensors mainly.

Canonical Polyadic or CANDECOMP/PARAFAC decomposition (CPD) [Harshman et al., 1970] is one of the most well-studied tensor decomposition methods. Using CPD, a third-order tensor $\mathcal{T}$ of shape $I \times J \times K$ (we will assume throughout the paper that $I \geq J \geq K$, unless stated otherwise) can be expressed as the sum of $R_{\mathcal{T}}$ rank-one tensors:

$$\begin{aligned} \mathcal{T} &= \sum_{r=1}^{R_{\mathcal{T}}} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r, \\ \mathcal{T}[i,j,k] &= \sum_{r=1}^{R_{\mathcal{T}}} \mathbf{a}_r[i]\, \mathbf{b}_r[j]\, \mathbf{c}_r[k], \end{aligned} \tag{1}$$

where $\mathbf{a}_r \in \mathbb{R}^I$, $\mathbf{b}_r \in \mathbb{R}^J$, and $\mathbf{c}_r \in \mathbb{R}^K$ are the rank-one component vectors, and the symbol $\circ$ denotes the outer product across all modes of the tensor, where each vector is indexed by the respective dimensions of the tensor. The matrices $\mathbf{A} \in \mathbb{R}^{I \times R_{\mathcal{T}}}$, $\mathbf{B} \in \mathbb{R}^{J \times R_{\mathcal{T}}}$, and $\mathbf{C} \in \mathbb{R}^{K \times R_{\mathcal{T}}}$ consist of these vectors stacked as columns, and are commonly referred to as the factor matrices. Many methods aim to recover the factor matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ by minimizing the reconstruction error (typically expressed using the Frobenius norm):

$$\left\| \mathcal{T} - \sum_{r=1}^{R_{\mathcal{T}}} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \right\|_F$$

often using alternating least squares or similar optimization techniques. However, this task is out of scope for this work.

The quantity of interest in this work is the canonical rank $R_{\mathcal{T}}$ of a latent low-rank tensor $\mathcal{T}$, which plays a fundamental role in CP decomposition. Estimating this rank is notoriously difficult [Hillar and Lim, 2013], particularly when only a noisy observation is available. In many machine learning applications, one observes an empirical tensor $\widehat{\mathcal{T}} = \mathcal{T} + \mathcal{N}$, and the goal is to recover structural properties of the underlying signal rather than decompose the noisy tensor exactly.

Problem. *Given an observed tensor $\widehat{\mathcal{T}} = \mathcal{T} + \mathcal{N}$, where $\mathcal{T}$ is an unknown low-rank tensor, estimate the canonical rank $R_{\mathcal{T}}$ of $\mathcal{T}$, namely the smallest R such that*

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r.$$

[1]Corresponding code and data are available in `https://github.com/kbogas/FastRank/`

From this perspective, our task can be viewed as rank estimation for an approximate low-rank reconstruction of the observed tensor. The emphasis is not on computing the exact CP rank of the noisy tensor $\widehat{\mathcal{T}}$, but on estimating the effective rank of the latent tensor that generates it.

3 RELATED WORK

Determining the rank of a tensor is a fundamental yet computationally challenging problem in tensor analysis. One of the earliest heuristic methods is the Core Consistency Diagnostic (CORCONDIA) [Bro and Kiers, 2003], which assesses the validity of a CPD model by measuring the discrepancy between the original tensor and its decomposition. Building on this concept, AutoTen [Papalexakis, 2016] automates rank selection by iteratively computing CORCONDIA scores and selecting the optimal number of components. Normalized SVD (NSVD) [Tsitsikas and Papalexakis, 2020] extends this idea using singular value decomposition, but it requires manual intervention, limiting its practicality.

The NORMO method [Fernandes et al., 2020] seeks the highest rank at which extracted components remain non-redundant, using correlation coefficients to identify important factors. While effective, it requires multiple CPD computations, making it computationally expensive. Some Bayesian methods aim to avoid exhaustive CPD by incorporating probabilistic priors. For instance, Bayesian inference is used in [Zhao et al., 2015] to estimate rank, while Automatic Relevance Determination (ARD) [Minka, 2001] is adapted in [Mørup and Hansen, 2009], using it iteratively to prune low-norm components and refine the number of factors. Although these methods are adaptive, they remain computationally intensive due to their reliance on iterative optimization schemes.

Matricization-based methods, which first transform a tensor into a matrix by stacking its slices row- or column-wise, are more closely related to our work. The minimum description length (MDL) approach in [da Costa et al., 2007] utilizes the eigenvalue spectrum obtained from the singular value decomposition (SVD) of a matricized tensor. Its generalized extension [Liu et al., 2016] considers eigenvalues from all multiple matricizations to estimate the tensor rank. These methods are considerably faster than CPD-based approaches, as they avoid the expensive decomposition step. However, they often suffer from numerical instability, especially in noisy environments, so some efforts have aimed to improve their robustness, for instance by employing modified eigenvalue estimators [Yokota et al., 2016]. Nevertheless, these techniques seem unsuitable for real-world data, as they tend to produce inaccurate rank estimates in the presence of noise [Rošt'áková and Rosipal, 2022]. A notable recent advancement that combines efficiency with improved performance is FRAPPE [Shiao and Papalexakis, 2024], which introduces a self-supervised learning approach to approximate rank without explicitly computing the CPD.

To summarize, a common limitation of most existing methods is their reliance on computing the CPD, often multiple times, which incurs substantial computational costs. Moreover, most methods, including the current strongest baseline FRAPPE, require an explicit upper bound on the rank to function effectively. In contrast, *FastRank* avoids both of these limitations, preserving the efficiency of matricization-based techniques while offering greater accuracy than the current state-of-the-art, both in synthetic experiments and real-world scenarios.

4 PROPOSED METHOD

The foundation of FastRank lies in the observation that the rank of a 3-mode tensor can be estimated using the spectral energy (i.e., singular values) of a sum-reduced 2D matrix obtained by collapsing the tensor along any mode. Formally:

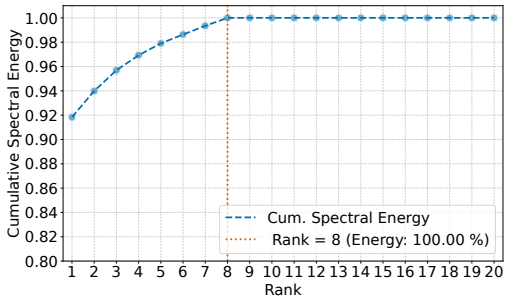
Claim 1. *Let $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ be a tensor of canonical rank $R_{\mathcal{T}}$, and let $\mathbf{M} \in \mathbb{R}^{I \times J}$ be the matrix obtained by summing over the third mode: $\mathbf{M}[i,j] = \sum_{k=1}^{K} \mathcal{T}[i,j,k]$. Then the matrix rank R_M satisfies: $R_M \leq R_{\mathcal{T}}$.*

Regarding Claim 1, the choice of mode over which the summation is performed is without loss of generality and does not affect the result. For concreteness, we state the claim for the third mode, which is the smallest; this choice is made purely for practical reasons in the proposed methodology, as explained in the following section. A formal proof of this claim is provided in the technical appendix. The appendix also includes a straightforward extension of the claim to tensors of arbitrary order N, along with additional experiments validating the theoretical results on N-D tensors (i.e., tensors with N dimensions).

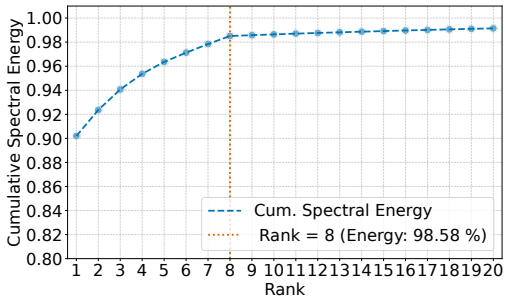
Based on the previous claim and motivated by other matricization-based methodologies [Liu et al., 2016], we propose using the singular values σ_r of the sum-reduced matrix $\mathbf{M}$ to estimate the original rank of the tensor. Drawing from ideas in spectral matrix theory [Girko, 1985], we leverage the Cumulative Spectral Energy (CSE), which is the cumulative sum of the nor-

malized squares of the singular values of matrix $\mathbf{M}$, to estimate the tensor rank.

We will motivate this procedure through an example. Let us first generate a synthetic tensor with shape $100\times50\times10$ and rank $R_{\mathcal{T}} = 8$. The factors $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are sampled from a uniform random distribution $\mathcal{U}[0, 1)$, and the tensor $\mathcal{T}$ is created according to Eq. (1). We then sum-reduce the tensor along the third dimension to obtain a 100×50 matrix $\mathbf{M}$, for which we calculate the SVD, square its singular values, normalize them, and compute their cumulative sum. In Fig. 1a, we observe the resulting CSE plot (cut off at rank 20), which reaches a value of 1 at rank 8 and remains constant thereafter. This indicates that the remaining singular values are zero ($\sigma_9 = \cdots = \sigma_{50} = 0$), showing that the matrix rank is $R_M = 8$, which matches the tensor rank $R_{\mathcal{T}}$ in this scenario. Therefore, in such noise-free scenarios, we will be using the minimum rank at which the CSE of $\mathbf{M}$ reaches its maximum value of 1, as the estimate for the tensor rank.



(a) Cumulative spectral energy for the tensor without noise.



(b) Cumulative spectral energy for the tensor with noise.

Figure 1: These subfigures illustrate the differences in the cumulative spectral energy plots for a synthetic tensor with shape $100\times50\times10$ and $R_{\mathcal{T}} = 8$ (a) without noise and (b) with 20% noise.

However, we know that synthetic data do not always exhibit the same properties as real-world data, which often contain noise and minor deviations due to measurement errors, inconsistencies, etc. Moreover, SVD is susceptible to noise, especially when estimating the tail of the singular values [Stewart, 1998]. We can observe this phenomenon in our synthetic tensor as well. To simulate noise, we generate a noise tensor $\mathcal{N} \sim \mathcal{U}[0, 1)$ and add 20% of this scaled noise to the original synthetic tensor $\mathcal{T}$. Formally, this is expressed as:

$$\mathcal{T}_{noise} = \mathcal{T} + \alpha \frac{||\mathcal{T}||_F}{||\mathcal{N}||_F} \mathcal{N}, \tag{2}$$

where $\alpha = 0.2$ represents the noise level. This noise addition procedure is similar to the one used in FRAPPE [Shiao and Papalexakis, 2024] and follows the default procedure for adding noise in tensors as in [Bader and Kolda, 2006].

Repeating the procedure described above, we generate the CSE for the noisy tensor $\mathcal{T}_{noise}$, and the result is plotted in Fig. 1b. In this case, the CSE at rank 8 has not yet reached 1, indicating that the remaining singular values are nonzero and are overestimated in importance due to noise in the tensor. However, we can also observe that the true rank of the tensor corresponds to the elbow of the curve, as before, and the CSE increases very slowly afterwards.

Based on this observation, we propose a simple heuristic for identifying the elbow point of the CSE curve. Specifically, we seek the earliest rank at which the CSE exceeds a user-defined threshold t, and the curve shows signs of plateauing beyond that point. The FastRank method is presented in Algorithm 1, with steps 3-5 implementing the simple "elbow-finding" heuristic.

Algorithm 1 FastRank

Input: Tensor $\mathcal{T}$, threshold t
Output: Estimated rank r
1: Sum-reduce $\mathcal{T}$ over its smallest mode to form matrix $\mathbf{M}$
2: $\sigma \leftarrow \text{SVD}(\mathbf{M})$
3: $\sigma_{normal} \leftarrow \sigma^2 / \sum \sigma^2$
4: $CSE \leftarrow \text{cumsum}(\sigma_{normal})$
5: $r \leftarrow \min \{i : \text{CSE}_i \geq t, |\text{CSE}_i - \text{CSE}_{i-1}| \leq 0.01\}$
6: **return** r

One important detail is that in the first step of the Algo. 1, we sum-reduce the tensor $\mathcal{T}$ along its smallest dimension. The reasoning behind this choice is that the matrix rank of $\mathbf{M}$ is also upper-bounded by the matrix's dimensions: $R_M \leq \min(I, J)$. Therefore, to capture as large a tensor rank as possible, we need to ensure that the remaining dimensions of $\mathcal{T}$ in $\mathbf{M}$ are as large as possible. Hence, we sum-reduce along the smallest dimension. Moreover, extending FastRank to N-D tensors is trivially done by simply iterating the

sum-reduction process (i.e., step 1 of Algo. 1) over the smallest modes of the tensor, until we have a 2D matrix $\mathbf{M}$.

The complexity of FastRank is mainly driven by the cost of the SVD, which for a fully dense matrix $\mathbf{M}$ is $\mathcal{O}(n^3)$, where $n = \max(I, J, K)$. In comparison, FRAPPE [Shiao and Papalexakis, 2024], which is one of the most competitive method in terms of performance, has a complexity of $\mathcal{O}(n^4)$. A detailed analysis on the scalability and sensitivity on the threshold t for FastRank is included in the technical appendix to provide a more comprehensive evaluation of the method.

5 EXPERIMENTS

Assessing the performance of rank estimation methods is challenging due to the limited availability of tensors with known ranks, making it impractical to rely exclusively on real-world datasets. To address this limitation, we utilize a combination of different scenarios for evaluation: rank estimation on synthetically generated tensors with known ranks (with and without noise), rank estimation on real-world tensors whose ranks have been established in prior literature, and a practical scenario where the rank estimation methodology is applied to guide a knowledge graph completion model. The evaluation of rank estimation, in cases where the tensor rank is known, is performed using the Mean Absolute Error (MAE), formally defined as:

$$MAE = \frac{1}{N}\sum_{i}^{N} |R^i_{\mathcal{T}} - \tilde{R}^i_{\mathcal{T}}|$$

where i iterates over the N tensors in the data, and $\tilde{R}^i_{\mathcal{T}}$ corresponds to the estimated rank for tensor i. All experiments were run on an Ubuntu Server with an AMD Ryzen processor @ 3.9GHz. We allocated 8 threads and a maximum of 50GB of RAM for all experiments.

5.1 Synthetic Tensors

To generate synthetic data, we first construct the CPD factor matrices by sampling from a uniform random distribution $\mathcal{U}[0, 1)$ for randomly selected tuples of I, J, K, and $R_{\mathcal{T}}$, following a generation procedure similar to that in [Shiao and Papalexakis, 2024]. The generated tensors are a mix of sparse and dense ones to simulate different real-world scenarios. The sparsity factor is randomly sampled in the range $[1\%, 50\%]$. In this first experiment, we limit the dimensions to $I, J, K \leq 100$ and the rank to $R_{\mathcal{T}} \leq 50$ (as we are in the low-rank regime) following common practice [Shiao and Papalexakis, 2024]. Furthermore, we also make sure that the rank of each tensor can be approximated by the SVD of $\mathbf{M}$, that is, $R_{\mathcal{T}} \leq \min(I, J)$. We compare FastRank against another fast matricization/eigenvalue-based method N-D MDL (MDL) [Liu et al., 2016], a CPD-based method AutoTen [Papalexakis, 2016], and FRAPPE [Shiao and Papalexakis, 2024], which is one of the latest and most accurate methods available. An important note is that for AutoTen and FRAPPE, we must specify the maximum possible rank for each tensor. In our experiments, we provide the maximum possible rank of 50. This practically bounds the maximum error these models can make per prediction, artificially boosting their performance, as will be shown in Table 4.

5.1.1 No-noise setup

First, we generate 300 tensors without noise following the previous procedure. For FastRank, we have two variants: the default version, where we sum-reduce each tensor using its smallest dimension to generate the matrix $\mathbf{M}$ of shape $(I \times J)$, and a second version where we sum-reduce along the middle dimension J, resulting in a matrix of shape $(I \times K)$. The second variant will be able to predict ranks only up to K (since $R_M \leq \min(I, K) = K$), so we expect to see a higher error rate for this version. We use this variant to validate the theoretical claim that the method works regardless of the mode along which we sum-reduce the tensor. For both variants, the FastRank threshold t is set to 1 as we are in the noiseless scenario.

Table 1: Performance results for different methods on 300 synthetic tensors without noise. The first group of results ($R_{\mathcal{T}} \leq K$) corresponds to a subset with tensors that have rank smaller than the smallest tensor dimension. The values reported are the means (± std for ranks).

Model	**MAE**	**Time** (s/ten)
Tensors with $R_{\mathcal{T}} \leq K$ (94/300 cases)		
AutoTen	8.47 ± 15.02	11.44
FRAPPE	3.96 ± 5.21	59.23
MDL	3.44 ± 13.96	0.0125
FastRank on $(I \times K)$	**0.00 ± 0.00**	**0.0002**
FastRank on $(I \times J)$	**0.00 ± 0.00**	**0.0004**
Tensors with $R_{\mathcal{T}} \leq J$ (300/300 cases)		
AutoTen	14.58 ± 16.33	10.83
FRAPPE	6.82 ± 8.14	48.68
MDL	4.68 ± 12.77	0.0100
FastRank on $(I \times K)$	9.89 ± 13.12	**0.0002**
FastRank on $(I \times J)$	**0.00 ± 0.00**	**0.0004**

In Table 1, we report first the performance on a sub-

set of tensors for which the second variant of FastRank has the capacity to estimate the rank (i.e., the cases where $R_{\mathcal{T}} \leq K$), and next on the full dataset. As observed, in the first scenario, both variants achieve a zero error rate, indicating that the sum-reduced matrix rank is the same as the original tensor rank for all tensors in the dataset. This also validates that the mode on which we sum-reduce does not influence the estimation, as expected from a theoretical perspective. Moving on to the full dataset, we see that the FastRank variant indeed shows a drop in performance, as expected, while the default model retains a zero error rate, outperforming all other baselines. Specifically, we see that FRAPPE and MDL perform better than AutoTen, with MDL being much faster than the other two. This is expected as MDL is a matricization-based method utilizing the SVD of the unfolded tensor. Nonetheless, FastRank is still 100× faster than MDL for these synthetic tensors.

5.1.2 Noisy setup

Real-world data usually contain noisy measurements and perturbations. For this reason, we created noisy tensors according to the following real-world scenarios, as in the work of [Shiao and Papalexakis, 2024]:

- *Sparse tensors with sparse noise*: Factors are sparse, and noise is added only to non-zero entries, similar to a weighted, time-evolving graph.
- *Sparse tensors with dense noise*: Sparse factors with noise added to all tensor entries, as in dynamic sensor grid data.
- *Dense tensors with dense noise*: Fully dense factors and noise across all entries that mimic fluorescence data streams.

The noise is added per Eq. (2), and the noise parameter α ranges from 2% to 10%. For this setup, we use $t = 99.99\%$ for FastRank, based on visual inspection of the CSE curve on a few noisy examples. The results are shown in Table 2, where we again observe significant speed improvements, as well as better accuracy in terms of rank estimation over all competing methods, under noisy conditions. Specifically for MDL, we see a great drop in performance, as already documented in [Rošťáková and Rosipal, 2022], where it is shown that most matricization-based methods suffer in noisy environments. In contrast, FastRank handles noise much better, through the eigenspectrum thresholding process, while still being very fast.

This demonstrates that the idea of using cumulative spectral energy for tensor rank estimation is feasible even under noise and that the FastRank algorithm for handling noisy setups is valid. In the technical appendix, results on noisy and noiseless synthetic 4D tensors using FastRank are also provided.

Table 2: Performance results for different methods on 300 synthetic tensors with noise.

Model	**MAE**	**Time** (sec/tensor)
AutoTen	17.54 ± 22.47	10.27 ±7.98
FRAPPE	9.37 ± 9.64	52.86 ± 90.82
MDL	20.90 ± 9.64	0.01 ± 0.01
FastRank	**7.08 ± 8.23**	**0.0004 ± 0.00**

5.2 Known-Rank Tensors

In this section, we evaluate the performance of FastRank alongside competing methods on datasets with known ranks. These include three widely studied low-rank fluorescence datasets, namely *amino* ($5 \times 61 \times 201$, $R_{\mathcal{T}} = 3$) [Bro, 1997], *dorrit* ($24 \times 27 \times 121$, $R_{\mathcal{T}} = 4$) [Riu and Bro, 2003], and *sugar* ($7 \times 268 \times 571$, $R_{\mathcal{T}} = 3$) [Bro, 1999], which exhibit naturally trilinear structures, and each CPD component corresponds to a distinct chemical. We also include the *tongue* dataset ($5 \times 10 \times 13$, $R_{\mathcal{T}} = 3$) [Harshman et al., 1977], which consists of X-ray recordings from five speakers articulating various vowels.

We compare FastRank against multiple established rank estimation methods: AutoTen, FRAPPE and MDL as before, and also a CPD-based method NORMO [Fernandes et al., 2020]. For AutoTen, NORMO, and MDL, we report their results as published in the respective literature. For FRAPPE, we reproduce its predictions by running the model with the recommended parameters for 10 trials and report the average estimated rank. Since all these models (except MDL) require an upper bound on the tensor rank, we follow common practice [Shiao and Papalexakis, 2024] and provide $2 \times R_{\mathcal{T}}$ as the maximum rank input. For FastRank, we use $t = 95\%$ across all datasets.

Table 3: Predictions of different methods on 4 known-rank tensors and their MAE across all of them.

Method	**amino**	**dorrit**	**sugar**	**tongue**	**MAE**
True Rank	3	4	3	3	–
AutoTen	**3**	**4**	2	5	0.75
NORMO	4	**4**	5	1	1.25
MDL	11	120	556	6	173
FRAPPE	**3**	3	4	4	1.46
FastRank	**3**	5	**3**	4	**0.5**

From the results shown in Table 3, we observe that FastRank achieves the lowest MAE across the four datasets, correctly estimating the exact rank in 2 out of 4 cases, and deviating by only one rank in the remaining two. In contrast, while the other methods succeed in estimating the correct rank for one or two datasets, their errors are typically larger when incorrect. Finally, the other eigenvalue-based method, MDL, overestimates the actual rank, particularly in the *sugar* dataset. This is mainly due to its tendency to overvalue the importance of near-zero eigenvalues, a phenomenon commonly reported for eigen-based methods [Yokota et al., 2016].

In Table 4, we also report the runtime of FastRank and FRAPPE on these datasets. For FRAPPE, we provide two variants: the default configuration (FRAPPE◇), which uses a maximum rank of $2 \times R_{\mathcal{T}}$, and an alternative (FRAPPE□) that sets a fixed maximum rank of 10. Both variants were executed 10 times, and we report the average predictions along with their standard deviations. As in the synthetic data, FastRank consistently outperforms FRAPPE in terms of runtime, achieving a $\sim 1000\times$ speedup across all datasets. Additionally, the FRAPPE□ variant demonstrates a noticeable decline in both speed and accuracy. For example, on the *sugar* and *tongue* datasets, the estimated ranks nearly double, and so does the average runtime. This highlights a critical limitation for most of the existing methods: the requirement to predefine a maximum possible rank, which strongly affects both the prediction accuracy and computational efficiency.

Table 4: Time (in seconds) needed and the predicted ranks for FastRank and two FRAPPE variants with different maximum possible ranks.

Method	**amino**	**dorrit**	**sugar**	**tongue**
True Rank	3	4	3	3
FastRank (time)	0.001	0.001	0.012	0.001
FRAPPE◇ (time)	8.65	7.81	120.36	0.41
FRAPPE□ (time)	12.52	11.66	**277.88**	0.94
FastRank (rank)	3	5	3	4
FRAPPE◇ (rank)	3.09	2.50	3.63	3.61
FRAPPE□ (rank)	3.40	2.58	**6.60**	**6.40**

◇ Max Rank: $2 * R_{\mathcal{T}}$, □ Max Rank: 10

Finally, Fig. 2 presents the CSE curves for all four datasets, plotted up to rank 10. Similar to the synthetic examples, these curves demonstrate that the CSE of the sum-reduced matrix captures meaningful information about the underlying tensor rank. Notably, the true rank consistently appears at or near the "elbow" of the CSE curve. This further validates FastRank's automatic procedure and also provides an avenue for user validation or manual override of the prediction through visual inspection.

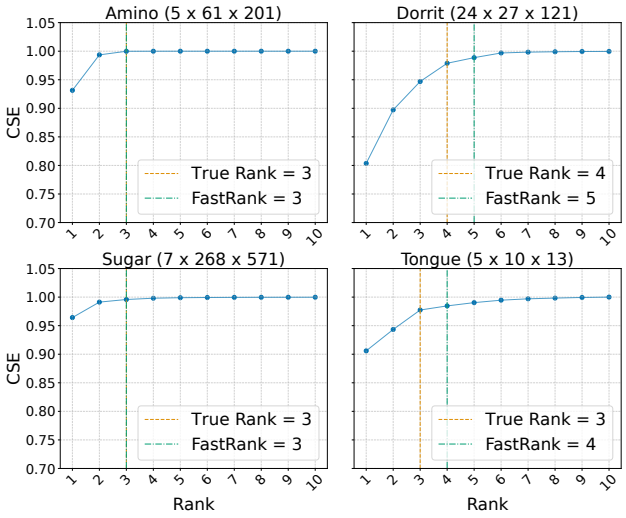


Figure 2: Cumsum curves of the spectral energy for four known-rank datasets, their true ranks, and the predicted values with FastRank. We can see that in all cases, both the true and the estimated rank are near or on the "elbow" of the curve.

5.3 Knowledge Graph Rank Estimation

Finally, we assess the effectiveness of our model in selecting the appropriate number of components for a tensor decomposition model within the context of a knowledge graph completion task. Knowledge Graphs (*KGs*) are large, graph-structured databases that store facts in the form of triples (e_s, r, e_o), where e_s and e_o represent subject and object entities, and r denotes a relation. However, most real-world KGs are incomplete, motivating the need for the automated inference of missing facts. KGs can be represented as third-order binary tensors, where each entry indicates the presence (1) or absence (0) of a specific triple. The KG completion task aims to infer which of the zero entries correspond to true but missing facts. Given a query of the form $(e_s, r, ?)$, the goal is to rank all possible object entities e_o and identify those most likely to complete the triple.

For our experiments, we use *codex-s*, a variant of the CoDeX benchmark [Safavi and Koutra, 2020], which includes entities and relations sourced from Wikipedia. When represented as a tensor, the KG has shape $2034 \times 2034 \times 42$, corresponding to 2034 entities and 42 relations. Our setup proceeds as follows: we first transform the KG into a binary tensor and use FastRank to estimate its rank. This estimated rank is then used as the number of components in a tensor decomposition framework that embeds entities and

relations using this rank as the embedding dimension [Balažević et al., 2019]. Using the learned embeddings, we then perform KG completion. Specifically, for each test triple of the form $(e_s, r, ?)$, we score all possible object entities and evaluate the model using Hits@10 (*H@10*), which measures the proportion of test queries for which the correct entity e_o is ranked among the top-10 candidate entities.

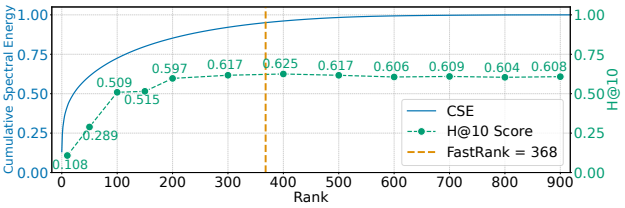


Figure 3: Cumsum curve for the spectral energy (blue) of the CoDex-s knowledge graph, the H@10 performance of the decomposition model in KG completion over different ranks (green), and the FastRank prediction (orange). The predicted rank is near the plateau of the model's performance, providing a good estimate for rank selection.

The objective of this experiment was to select the appropriate number of components (i.e., the rank) that balances predictive performance and computational efficiency. Ideally, this rank should lie near the "elbow" of the rank-performance curve, where additional components yield diminishing returns. In Fig. 3, the green, dashed curve illustrates the performance of the decomposition model across different rank values. As shown, the predictive performance saturates around ranks $[300, 500]$ and even slightly declines thereafter, confirming the presence of an "elbow region".

In the same figure, we also present the (blue) CSE curve for the sum-reduced matrix, and the estimated rank from FastRank marked as a vertical (orange) line at 368. Notably, the shape of the CSE curve closely mirrors that of the model performance curve, suggesting that CSE is a reliable proxy for identifying the appropriate rank. Furthermore, FastRank's estimate of 368 falls within the optimal range, validating its effectiveness as a fast and accurate estimator. While FastRank needed a few seconds to compute an accurate estimate of the rank for the tensor decomposition task, generating the full performance curve for empirical selection would take several hours due to the need to train multiple decomposition models at varying ranks. Thus, FastRank provides a reasonable rank estimate that is a good starting point (performance-wise) for the task, at a fraction of the time cost.

As a point of comparison, alternative methods such as AutoTen, NORMO, and FRAPPE did not complete within 24 hours for this KG tensor. This is primarily due to their iterative decomposition procedures and their inefficiency at high ranks, both of which manifest at the scale of knowledge graphs. For this experiment, we used the official decomposition implementation and trained all models for 200 epochs, by which point validation scores had converged, using the default hyperparameter settings. For FastRank, we use $t = 95\%$, as in the previous experiment with real-world data.

6 DISCUSSION

The experimental results presented above validate the core premise of FastRank: enabling accurate and computationally efficient tensor rank estimation. In this section, we further elaborate on design choices, discuss the relation between sum-reduction and unfolding-based matricizations, and highlight several methodological nuances and directions for future work.

A key theoretical question is under what conditions the rank R_M of the sum-reduced matrix equals the original tensor rank $R_{\mathcal{T}}$. As shown in the noiseless case (Table 1), FastRank achieves zero estimation error, indicating that $R_M = R_{\mathcal{T}}$ across all tested synthetic tensors. This suggests that, under ideal conditions, sum-reduction preserves the tensor rank. At the same time, there exist theoretical cases where $R_M < R_{\mathcal{T}}$. Two such edge cases are discussed in the technical appendix, although they did not arise in our synthetic experiments, suggesting that they may be uncommon in practice. A more systematic characterization of these pathological configurations remains an interesting direction for future theoretical work.

A related practical question is why sum-reduction can outperform unfolding-based procedures in the low-rank, noisy regime, even though unfolding generally produces a larger matrix and therefore a larger maximum detectable rank. The key point is that these two matricization strategies induce different admissible rank ranges and different levels of spectral contamination from noise. Consider an observed tensor $T_{\text{obs}} = T + \mathcal{N}$ of size $I \times J \times K$ with $I \geq J \geq K$, where T is low-rank and $\mathcal{N}$ is a noise tensor. After summation along the third mode, one obtains an observed matrix (easy to show) of the form:

$$\mathbf{M}_{obs} = \mathbf{M} + \mathbf{M}_{noise}$$

of size $I \times J$. By subadditivity of matrix rank,

$$\text{rank}(\mathbf{M}_{obs}) \leq \text{rank}(\mathbf{M}) + \text{rank}(\mathbf{M}_{noise})$$

while the matrix dimensions impose:

$$\text{rank}(\mathbf{M}_{\text{obs}}) \leq \min(I, J) = J.$$

Hence, in the practically relevant regime considered throughout this paper, the estimated rank is constrained to a relatively narrow interval determined by the preserved matrix dimensions. By contrast, unfolding along the third mode yields a matrix of size $I \times (JK)$, whose rank may range up to $\min(I, JK)$. This enlarged feasible range increases the capacity to represent both signal and noise, which is beneficial when the tensor rank is genuinely large, but it also makes spectral heuristics more sensitive to noisy directions. In other words, sum-reduction compresses the admissible rank range and thereby limits the extent to which high-rank noise can dominate the observed eigenspectrum. This effect is particularly favorable in low-rank settings, where the true tensor rank already lies well within the detectable range of the sum-reduced matrix.

This interpretation is also consistent with the runtime advantage of FastRank. Operating on the sum-reduced matrix requires estimating the spectrum of an $I \times J$ matrix, whereas common unfolding-based alternatives operate on matrices such as $I \times (JK)$, or more generally on the most square unfolding available. Since the cost of spectral decompositions grows rapidly with matrix dimensions, the reduction in matrix size translates directly into substantial computational gains. To illustrate this effect, Table 5 compares rank estimation on a sum-reduced matrix of size $K \times K$ against rank estimation on an unfolded matrix of size $K \times K^2$. Specifically, we generate 300 tensors for each K, both dense and sparse following the procedure presented in the main paper, with rank $R_T \leq K$ and report the average runtime for estimating the reduced/unfolded matrix rank (no noise in this setup). We can see that even for third-order tensors, the gap becomes large quickly as K increases: for $K = 100$, the unfolded approach is more than an order of magnitude slower. A more detailed comparison of sum-reduction to unfolding is presented in Appendix C.

Table 5: Average runtime (milliseconds) for sum-reduced vs unfolded rank estimation, over different sizes K.

Matrix \ K	**10**	**50**	**100**	**500**	**1000**
Sum-reduced	0.1	0.3	1.3	257	5,289
Unfolded	0.1	2.3	11.4	8,635	90,892

Importantly, our claim is not that unfolding is universally inferior. On the contrary, unfolding increases the maximum detectable rank and can therefore be advantageous outside the low-rank regime studied here. Rather, our contribution is to show that when the tensor rank is within the detectable range of the sum-reduced matrix, sum-reduction preserves the essential spectral structure needed for rank estimation while providing a markedly more favorable trade-off between robustness and computational cost. This is precisely the setting targeted by FastRank, and it explains why the method performs particularly well on both synthetic and real-world low-rank tensors.

Moreover, when using FastRank in noisy data scenarios, it is important to consider how different types of noise affect the rank estimation. In particular, the impact of perturbations localized on either the collapsed or the preserved modes of the tensor could lead to varying distortions in the singular value spectrum of the sum-reduced matrix. Studying these effects systematically would complement the theoretical foundation of FastRank under noise and offer practical guidance for tailoring the method to specific application domains (a preliminary study as this is presented in Appendix C). More broadly, extending the present analysis to alternative reduction operators, structured noise models, and adaptive threshold-selection mechanisms constitutes a promising direction for future work.

Finally, tensor decomposition methods could be extended to utilize the left- and right-singular vectors of the SVD, which are currently treated as simple by-products. For instance, instead of randomly initializing the factors $\mathbf{A}$ and $\mathbf{B}$, one could initialize them using the SVD singular vectors in their place. Initial experiments with this approach, indicate speedups for methods based on alternating least squares approximations.

7 CONCLUSIONS

In this work, we introduced *FastRank*, a theoretically grounded method for estimating the canonical rank of low-rank tensors. Our approach leverages the singular values of a sum-reduced matrix to infer rank and applies a spectral energy thresholding scheme to ensure robustness to noise. A novel contribution of this work is showing that sum-reduction provides an effective matricization method for reliable rank estimation. We evaluated FastRank on synthetic tensors, real-world datasets with known ranks, and a knowledge graph completion task. In all cases, it achieved lower errors and was over 1000× faster than the current state-of-the-art. Importantly, FastRank was the only approach able to scale to large knowledge graphs, providing accurate rank estimates aligned with downstream task performance. We believe this formulation opens promising directions for studying the relation between the tensor and its sum-reduced ranks in noisy settings.

References

[Bader and Kolda, 2006] Bader, B. W. and Kolda, T. G. (2006). Algorithm 862: Matlab tensor classes for fast algorithm prototyping. *ACM Transactions on Mathematical Software (TOMS)*, 32(4):635–653.

[Balažević et al., 2019] Balažević, I., Allen, C., and Hospedales, T. M. (2019). Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*.

[Bro, 1997] Bro, R. (1997). Parafac. tutorial and applications. *Chemometrics and intelligent laboratory systems*, 38(2):149–171.

[Bro, 1999] Bro, R. (1999). Exploratory study of sugar production using fluorescence spectroscopy and multi-way analysis. *Chemometrics and Intelligent Laboratory Systems*, 46(2):133–147.

[Bro and Kiers, 2003] Bro, R. and Kiers, H. A. (2003). A new efficient method for determining the number of components in parafac models. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 17(5):274–286.

[Carroll and Chang, 1970] Carroll, J. D. and Chang, J.-J. (1970). Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319.

[da Costa et al., 2007] da Costa, J. P. C., Haardt, M., Romer, F., and Del Galdo, G. (2007). Enhanced model order estimation using higher-order arrays. In *2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, pages 412–416. IEEE.

[Eckart and Young, 1936] Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

[Fernandes et al., 2020] Fernandes, S., Fanaee-T, H., and Gama, J. (2020). Normo: A new method for estimating the number of components in cp tensor decomposition. *Engineering Applications of Artificial Intelligence*, 96:103926.

[Frolov and Oseledets, 2017] Frolov, E. and Oseledets, I. (2017). Tensor methods and recommender systems. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(3):e1201.

[Girko, 1985] Girko, V. L. (1985). Spectral theory of random matrices. *Russian Mathematical Surveys*, 40(1):77.

[Harshman et al., 1977] Harshman, R., Ladefoged, P., and Goldstein, L. (1977). Factor analysis of tongue shapes. *The Journal of the Acoustical Society of America*, 62(3):693–707.

[Harshman et al., 1970] Harshman, R. A. et al. (1970). Foundations of the parafac procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA working papers in phonetics*, 16(1):84.

[Hillar and Lim, 2013] Hillar, C. J. and Lim, L.-H. (2013). Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):1–39.

[Ji et al., 2019] Ji, Y., Wang, Q., Li, X., and Liu, J. (2019). A survey on tensor techniques and applications in machine learning. *IEEE Access*, 7:162950–162990.

[Kolda and Bader, 2009] Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM review*, 51(3):455–500.

[Liu et al., 2016] Liu, K., Da Costa, J. P. C., So, H. C., Huang, L., and Ye, J. (2016). Detection of number of components in candecomp/parafac models via minimum description length. *Digital Signal Processing*, 51:110–123.

[Minka, 2001] Minka, T. P. (2001). *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology.

[Mørup and Hansen, 2009] Mørup, M. and Hansen, L. K. (2009). Automatic relevance determination for multi-way models. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 23(7-8):352–363.

[Papalexakis, 2016] Papalexakis, E. E. (2016). Automatic unsupervised tensor mining with quality assessment. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 711–719. SIAM.

[Riu and Bro, 2003] Riu, J. and Bro, R. (2003). Jackknife technique for outlier detection and estimation of standard errors in parafac models. *Chemometrics and Intelligent Laboratory Systems*, 65(1):35–49.

[Rošťáková and Rosipal, 2022] Rošťáková, Z. and Rosipal, R. (2022). Determination of the number of components in the parafac model with a nonnegative tensor structure: A simulated eeg data study. *Neural Computing and Applications*, 34(17):14793–14805.

[Safavi and Koutra, 2020] Safavi, T. and Koutra, D. (2020). Codex: A comprehensive knowledge graph completion benchmark. *arXiv preprint arXiv:2009.07810*.

[Shiao and Papalexakis, 2024] Shiao, W. and Papalexakis, E. E. (2024). Frappe: Fast rank approximation with explainable features for tensors. *Data Mining and Knowledge Discovery*, 38(6):4217–4232.

[Sidiropoulos et al., 2017] Sidiropoulos, N. D., De Lathauwer, L., Fu, X., Huang, K., Papalexakis, E. E., and Faloutsos, C. (2017). Tensor decomposition for signal processing and machine learning. *IEEE Transactions on signal processing*, 65(13):3551–3582.

[Simon and Zha, 2000] Simon, H. D. and Zha, H. (2000). Low-rank matrix approximation using the lanczos bidiagonalization process with applications. *SIAM Journal on Scientific Computing*, 21(6):2257–2274.

[Stewart, 1998] Stewart, G. W. (1998). *Perturbation theory for the singular value decomposition*. Citeseer.

[Tsitsikas and Papalexakis, 2020] Tsitsikas, Y. and Papalexakis, E. E. (2020). Nsvd: normalized singular value deviation reveals number of latent factors in tensor decomposition. *Big Data*, 8(5):412–430.

[Williams et al., 2018] Williams, A. H., Kim, T. H., Wang, F., Vyas, S., Ryu, S. I., Shenoy, K. V., Schnitzer, M., Kolda, T. G., and Ganguli, S. (2018). Unsupervised discovery of demixed, low-dimensional neural dynamics across multiple timescales through tensor component analysis. *Neuron*, 98(6):1099–1115.

[Yokota et al., 2016] Yokota, T., Lee, N., and Cichocki, A. (2016). Robust multilinear tensor rank estimation using higher order singular value decomposition and information criteria. *IEEE Transactions on Signal Processing*, 65(5):1196–1206.

[Zhao et al., 2015] Zhao, Q., Zhang, L., and Cichocki, A. (2015). Bayesian cp factorization of incomplete tensors with automatic rank determination. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1751–1763.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes. Please see Section 2 for the mathematical setting and Section 4 for the proposed method.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Yes. This is presented at the end of Section 4.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. Yes. Please see the provided supplementary material that contains scripts and data for implementing the proposed method and reproducing all experimental results.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. Yes. Please see the Proof Sections in the Supplementary Material.
   (b) Complete proofs of all theoretical results. Yes. Please see the Proof Sections in the Supplementary Material.
   (c) Clear explanations of any assumptions. Yes. Please see the Proof Sections in the Supplementary Material.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). Yes. Please see the included README.md and scripts in the attached supplementary material.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes. Please see the included README.md and scripts in the attached supplementary material.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes. Please see the included README.md and scripts in the attached supplementary material.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes. Please see the included README.md and scripts in the attached supplementary material.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. Yes. We provide citations to the existing used datasets and the competing methods.
   (b) The license information of the assets, if applicable. Not applicable.
   (c) New assets either in the supplemental material or as a URL, if applicable. Yes. Please see the included README.md and scripts in the attached supplementary material.
   (d) Information about consent from data providers/curators. Not Applicable.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. Not Applicable.
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable.
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable.

Supplementary Material

A PROOF FOR 3-D TENSORS

In this section, we provide the formal proof for Claim 1, alongside a discussion on possible theoretical cases where the matrix rank R_M is strictly lower than the tensor rank $R_{\mathcal{T}}$.

First, as a reminder from the main text, using the Canonical Polyadic or CANDECOMP/PARAFAC decomposition (CPD) [Harshman et al., 1970, Carroll and Chang, 1970] a third-order tensor $\mathcal{T}$ of shape $I \times J \times K$, can be expressed as the sum of $R_{\mathcal{T}}$ rank-one tensors:

$$\begin{aligned} \mathcal{T} &\approx \sum_{r=1}^{R_{\mathcal{T}}} T_r = \sum_{r=1}^{R_{\mathcal{T}}} \mathbf{i_r} \circ \mathbf{j_r} \circ \mathbf{k_r} \\ \mathcal{T}[i,j,k] &\approx \sum_{r=1}^{R_{\mathcal{T}}} \mathbf{A}[i,r] \cdot \mathbf{B}[j,r] \cdot \mathbf{C}[k,r] \end{aligned} \tag{3}$$

, where $\circ$ denotes the outer product.

The CPD can be seen as generalizing the Singular Value Decomposition (SVD) [Eckart and Young, 1936] of a matrix to a tensor. As a reminder, the SVD of a matrix $\mathbf{M} \in \mathbb{R}^{I \times J}$ is given by:

$$\begin{aligned} \mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T &= \sum_{r=1}^{R_M} \sigma_r \mathbf{u}_r \circ \mathbf{v}_r^T \\ \mathbf{M}[i,j] &= \sum_{r=1}^{R_M} \sigma_r U_{ir} V_{jr} \end{aligned} \tag{4}$$

, where $R_M \leq \min(I, J)$ is the rank of $\mathbf{M}$. The matrix $\mathbf{\Sigma}$ is a diagonal matrix of size $R_M \times R_M$ containing the non-zero singular values σ_r on its diagonal. These singular values are non-negative and can be arranged in decreasing order: $\sigma_1 > \sigma_2 > \cdots > \sigma_{R_M} > 0 = \sigma_{R_M+1} = \cdots = \sigma_J$. The matrices $\mathbf{U} \in \mathbb{R}^{I \times R_M}$ and $\mathbf{V} \in \mathbb{R}^{J \times R_M}$ contain the orthonormal left- and right-singular vectors $\mathbf{u}_r$ and $\mathbf{v}_r$, respectively.

In the main paper, we posit that:

Claim 2. *Let $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ be a tensor of canonical rank $R_{\mathcal{T}}$, and let $\mathbf{M} \in \mathbb{R}^{I \times J}$ be the matrix obtained by summing over the third mode:*

$$\mathbf{M}[i,j] = \sum_{k=1}^{K} \mathcal{T}[i,j,k] \tag{5}$$

Then the matrix rank $R_M = rank(\mathbf{M})$ satisfies:

$$R_M \leq R_{\mathcal{T}}.$$

Proof. Assuming that the tensor $\mathcal{T}$ has CP rank $R_{\mathcal{T}}$, it can be written as:

$$\mathcal{T}[i,j,k] = \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r[i]\mathbf{b}_r[j]\mathbf{c}_r[k], \tag{6}$$

where $\mathbf{a}_r \in \mathbb{R}^I$, $\mathbf{b}_r \in \mathbb{R}^J$, and $\mathbf{c}_r \in \mathbb{R}^K$ are the factor vectors for component r. We now substitute Eq. (6) into Eq. (5) and expand:

$$\begin{aligned}
\mathbf{M}[i,j] &= \mathcal{T}[i,j,1] + \mathcal{T}[i,j,2] + \cdots + \mathcal{T}[i,j,K] \\
&= \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r[i]\mathbf{b}_r[j]\mathbf{c}_r[1] + \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r[i]\mathbf{b}_r[j]\mathbf{c}_r[2] + \cdots \\
&\quad \cdots + \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r[i]\mathbf{b}_r[j]\mathbf{c}_r[K] \\
&= \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r[i]\mathbf{b}_r[j]\big(\mathbf{c}_r[1] + \mathbf{c}_r[2] + \cdots + \mathbf{c}_r[K]\big) \\
&= \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r[i]\mathbf{b}_r[j]\big(\sum_{k=1}^{K} \mathbf{c}_r[k]\big).
\end{aligned} \tag{7}$$

Let us now define:

$$\gamma_r = \sum_{k=1}^{K} \mathbf{c}_r[k]. \tag{8}$$

Then substituting Eq. (8) in Eq. (7):

$$\mathbf{M}[i,j] = \sum_{r=1}^{R_\mathcal{T}} \mathbf{a}_r[i]\mathbf{b}_r[j]\gamma_r. \tag{9}$$

In matrix form, we can write Eq. (9) as:

$$\mathbf{M} = \sum_{r=1}^{R_\mathcal{T}} \gamma_r \mathbf{a}_r \mathbf{b}_r^T = \sum_{r=1}^{R_\mathcal{T}} \mathbf{M}_r. \tag{10}$$

Each term $\mathbf{M}_r = \gamma_r \mathbf{a}_r \mathbf{b}_r^T$ is at most rank one, because a matrix written as the (scaled) outer product of two non-zero vectors has always rank one. Therefore $\mathbf{M}$ is a sum of at most $R_\mathcal{T}$ rank-one matrices $\mathbf{M}_r$, leading to:

$$\text{rank}(\mathbf{M}) = R_M \leq R_\mathcal{T}.$$

□

A.1 Discussion

Some remarks regarding the procedure above. First, as seen from the proof, the selection of the third mode for sum-reducing was done without loss of generality, meaning that we can reduce whichever mode we want. This is validated empirically from the results in the noiseless synthetic setup in Table 1 of the main paper.

Secondly, a central theoretical consideration is the condition under which the matrix rank R_M of the sum-reduced matrix equals the original tensor rank $R_\mathcal{T}$. As demonstrated in the noiseless scenario in Table 1, FastRank achieves zero estimation error, indicating that it can retrieve $R_\mathcal{T}$, when it is retrievable, i.e. $R_M \leq R_\mathcal{T}$. This suggests that, sum-reduction preserves the tensor's rank information. However, there are cases where R_M is strictly less than $R_\mathcal{T}$. Focusing on Eq. (10) we can identify two such cases :

- **Linear dependence:** If any of the outer products $\mathbf{a}_r\mathbf{b}_r^T$ (i.e., the rank-one matrices) are linearly dependent on other outer products (i.e., matrices in the sum), the rank of their sum does not increase. Thus, the resulting matrix $\mathbf{M}$ may have rank strictly less than $R_{\mathcal{T}}$.
- **Zero scalar weights:** If $\gamma_r = \sum_k \mathbf{c}_r[k] = 0$ for any r, the corresponding term vanishes and contributes nothing to the matrix $\mathbf{M}$, again reducing the rank.

The fact that FastRank achieves a zero error rate, despite the above, suggests that such cases were not encountered in the 300 randomly generated tensors, implying that either they are relatively uncommon in practice or that the synthetic data differ from real-world scenarios. Nevertheless, a deeper exploration of the conditions that could lead to the strict inequality $R_M < R_{\mathcal{T}}$ could provide valuable insights into the limitations of FastRank.

B EXTENSION TO N-D TENSORS

In this section, we extend the claim to N-D tensors and provide the accompanying proof.

Claim 3. *Let $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ be an order-N tensor of canonical rank $R_{\mathcal{T}}$. Define a matrix $\mathbf{M} \in \mathbb{R}^{I_m \times I_n}$ as the result of sequential sum-reduction over all modes except two: modes m and n. Then:*

$$\text{rank}(\mathbf{M}) \leq R_{\mathcal{T}}.$$

Proof. Assume that $\mathcal{T}$ admits a CP decomposition of rank $R_{\mathcal{T}}$:

$$\mathcal{T}[i_1, i_2, \ldots, i_N] = \sum_{r=1}^{R_{\mathcal{T}}} \prod_{d=1}^{N} \mathbf{A}^{(d)}[i_d, r],$$

where $\mathbf{A}^{(d)} \in \mathbb{R}^{I_d \times R_{\mathcal{T}}}$ is the factor matrix for mode d, and $\mathbf{a}_r^{(d)}$ denotes its r-th column.

Now, construct a matrix $\mathbf{M} \in \mathbb{R}^{I_m \times I_n}$ by summing over all other modes $d \in \{1, \ldots, N\} \setminus \{m, n\}$. That is,

$$\mathbf{M}[i_m, i_n] = \sum_{i_{d_1}=1}^{I_{d_1}} \cdots \sum_{i_{d_{N-2}}=1}^{I_{d_{N-2}}} \mathcal{T}[i_1, i_2, \cdots, i_m, i_n, \cdots, i_N],$$

where $\{d_1, \ldots, d_{N-2}\} = \{1, \ldots, N\} \setminus \{m, n\}$.

Substituting the CP expression:

$$\mathbf{M}[i_m, i_n] = \sum_{i_{d_1}=1}^{I_{d_1}} \cdots \sum_{i_{d_{N-2}}=1}^{I_{d_{N-2}}} \left(\sum_{r=1}^{R_{\mathcal{T}}} \prod_{d=1}^{N} \mathbf{A}^{(d)}[i_d, r] \right).$$

Interchanging the summation order as in Eq. (7):

$$\mathbf{M}[i_m, i_n] = \sum_{r=1}^{R_{\mathcal{T}}} \mathbf{A}^{(m)}[i_m, r]\mathbf{A}^{(n)}[i_n, r] \prod_{d \notin \{m,n\}} \left(\sum_{i_d=1}^{I_d} \mathbf{A}^{(d)}[i_d, r] \right).$$

Define the scalar:

$$\gamma_r = \prod_{d \notin \{m,n\}} \left(\sum_{i_d=1}^{I_d} \mathbf{A}^{(d)}[i_d, r] \right),$$

so we get:

$$\mathbf{M}[i_m, i_n] = \sum_{r=1}^{R_{\mathcal{T}}} \gamma_r \mathbf{A}^{(m)}[i_m, r]\mathbf{A}^{(n)}[i_n, r].$$

Which in matrix form is written as:

$$\mathbf{M} = \sum_{r=1}^{R_{\mathcal{T}}} \gamma_r \mathbf{a}_r^{(m)} \left(\mathbf{a}_r^{(n)}\right)^\top.$$

As before, each term is a rank-one matrix (outer product of two vectors scaled by a scalar). Therefore, $\mathbf{M}$ is a linear combination of at most $R_{\mathcal{T}}$ rank-one matrices, implying again:

$$\text{rank}(\mathbf{M}) \leq R_{\mathcal{T}}.$$

□

B.1 Validation on 4D Tensors

In order to validate the extended claim we first need to modify the FastRank algorithm (main paper Algo. 1), to accommodate for N-D tensors. This is trivially done by repeatedly sum-reducing the tensor along its smallest dimensions until we are left with a 2D Matrix. The extended algorithm is provided in Algo.2.

Algorithm 2 Extended FastRank for N-D Tensors

Input: Tensor $\mathcal{T}$, threshold t
Output: Estimated rank r
$\mathbf{M} \leftarrow \mathcal{T}$
while $order(\mathbf{M}) \geq 2$ **do**
    $a \leftarrow argmin(\mathbf{M}.shape)$
    $\mathbf{M} \leftarrow sum(\mathbf{M}, axis = a)$
end while
$\sigma \leftarrow \text{SVD}(\mathbf{M})$
$\sigma_{normal} \leftarrow \sigma^2 / \sum \sigma^2$
$CSE \leftarrow \text{cumsum}(\sigma_{normal})$
$r \leftarrow \min\{i : \text{CSE}[i] \geq t, |\text{CSE}[i] - \text{CSE}[i-1]| \leq 0.01\}$
return r

With the extended algorithm in place, we move on to adapt the experimental procedure from 3D synthetic tensors, as described in the main paper, to 4D tensors as well. This way, we can experimentally validate Claim 2 and also showcase the applicability of FastRank on tensors of order $N \geq 3$.

As before, we first generate synthetic data, by sampling the CPD factor matrices from the same uniform random distribution $\mathcal{U}[0, 1)$. Once again, we limit the dimensions of the 4D tensors $I, J, K, L \leq 100$ with $I \geq J \geq K \geq L$ and the rank to $R_{\mathcal{T}} \leq 50$, ensuring that the rank of each tensor can be approximated by the SVD of $\mathbf{M}$, that is, $R_{\mathcal{T}} \leq \min(I, J)$. As in the 3D case, we first sample 300 4D matrices without noise while also varying the sparsity factor of each one in the range $[1\%, 50\%]$. Next, we follow the same procedure but also add noise, as described in the main paper, on another set of 300 4D tensors. Then we run the FastRank Algorithm on both noiseless and noisy tensors.

Table 6: Performance results of FastRank on synthetic 4D tensors, with and without noise. The values are reported as (mean ± std).

Model	**MAE**	**Time** (sec/tensor)
	300 *4D Tensors without noise*	
FastRank	**0.00 ± 0.00**	**0.0003 ± 0.00**
	300 *4D Tensors with noise*	
FastRank	4.86 ± 5.70	**0.0004 ± 0.00**

The results are presented in Table 6. In the absence of noise, FastRank exhibits perfect accuracy, achieving a zero error rate. This result empirically confirms that, for the noiseless case, the rank of the sum-reduced matrix

consistently matches the true rank of the original 4D tensor across the entire dataset, as was the case with the 3D tensors. When transitioning to the noisy setting and employing the same threshold of $t = 99.99\%$, as utilized in the main paper in this scenario, the performance of FastRank on higher-order (N-D) tensors remains consistent with the results observed for third-order tensors (i.e., MAE of 6.98 as seen in Table 2 of the main paper). Crucially, the computational overhead remains minimal. This efficiency, stemming from the low cost of the iterative sum-reduction operation, underscores the scalability and practical applicability of FastRank for analyzing higher-dimensional tensor data as well.

C FASTRANK DETAILS

This section presents a comprehensive analysis of the FastRank algorithm, offering both quantitative and qualitative insights into its operational characteristics. We first conduct a detailed examination of the hyperparameter t, which thresholds the Cumulative Spectral Energy (CSE), to understand its influence on rank estimation accuracy under additive noise on both synthetic and real-world tensors. We then evaluate the robustness of FastRank under a missing-value corruption model, demonstrating that the method remains effective beyond the standard additive-noise setting. The investigation also addresses the method's adaptability, extending the analysis to higher-order (N-D) tensors to validate its generalizability beyond the 3D case. Finally, we assess the scalability of FastRank by measuring its computational performance as the tensor dimensions increase, thereby establishing its practical viability for large-scale data analysis.

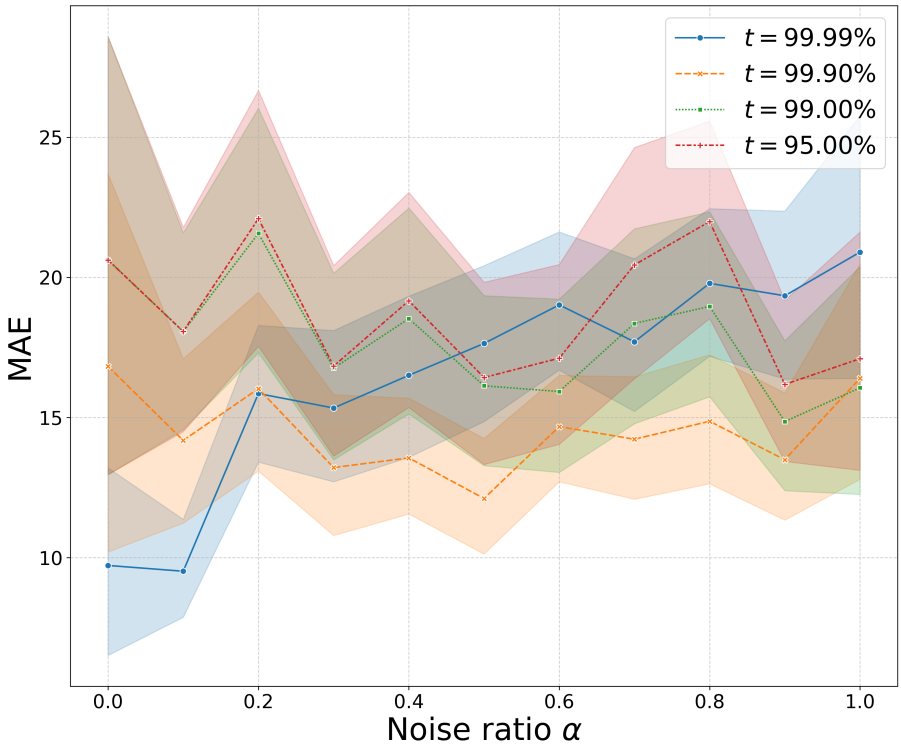


Figure 4: Performance of FastRank with different threshold parameter t on 3D noisy synthetic tensors, with varying noise ratio α. We see that as the noise ratio increases, higher values of t lead to better performance.

C.1 Sensitivity Analysis of t

To evaluate the robustness of our proposed FastRank algorithm, we perform a sensitivity analysis with respect to the Cumulative Spectral Energy (CSE) threshold, denoted by t, under varying levels of noise. As a reminder, the core of FastRank involves reducing a third-order tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ to a matrix representation via sum-reduction across one of its modes. We then apply Singular Value Decomposition (SVD) to this matrix to obtain its singular values, $\sigma_1, \sigma_2, \ldots, \sigma_n$. The rank is estimated by identifying the smallest integer r such that the CSE,

defined as:

$$\frac{\sum_{i=1}^{r} \sigma_i^2}{\sum_{i=1}^{n} \sigma_i^2} > t$$

The experiment utilizes synthetically generated 3D tensors with dimensions up to 100. Noise is introduced additively to a ground-truth tensor $\mathcal{T}$ according to the model:

$$T_{\text{noisy}} = T + \alpha \frac{\|T\|_F}{\|N\|_F} N$$

where N is a noise tensor of identical dimensions, $\|\cdot\|_F$ is the Frobenius norm, and $\alpha \in [0, 1]$ is the noise ratio controlling the noise intensity. We generate 1000 such tensors following the procedure described in the main paper.

The results of this analysis are presented in Figure 4, which plots the Mean Absolute Error (MAE) of the rank estimation against the noise ratio α for four distinct threshold values: $t \in \{99.99\%, 99.90\%, 99.00\%, 95.00\%\}$. We plot the mean of the performance for each data point and the 95% confidence interval around it as translucent bands. A clear dependency between the optimal threshold and the noise level is observable. In low-noise regimes ($\alpha < 0.2$), a high threshold of $t = 99.99\%$ yields the lowest MAE, accurately capturing the underlying rank by retaining a large portion of the spectral energy. However, as the noise ratio increases, the performance of this high threshold degrades significantly, as it erroneously incorporates spectral energy from the noise components. Conversely, lower thresholds demonstrate greater robustness to noise. For instance, $t = 99.90\%$ provides superior performance for a wide range of noise ratios ($\alpha > 0.2$), suggesting it strikes a balance between signal fidelity and noise rejection. The key takeaway is that the selection of t is a critical, data-dependent parameter. While a higher threshold is optimal for clean data, a moderately lower threshold is necessary to maintain accuracy in the presence of significant noise, preventing the model from overfitting to the noisy structure.

To complement the synthetic analysis, we also evaluate the effect of t on real-world datasets with known rank. Specifically, we consider the `amino`, `dorrit`, `sugar`, and `tongue` datasets, and report in Table 7 the estimated ranks obtained by FastRank for different threshold values. This experiment allows us to assess whether the trends observed under controlled synthetic noise persist in practical settings.

Table 7: Rank estimates for known-rank real-world datasets under varying t.

	amino	**dorrit**	**sugar**	**tongue**	**MAE**
True Rank	3	4	3	3	–
$t = 99.99$	4	16	14	1	6.5
$t = 99$	3	6	3	5	1
$t = 95$ (**Default**)	3	5	3	4	**0.5**
$t = 90$	3	5	3	2	**0.5**
$t = 80$	3	5	3	2	**0.5**

The results indicate that FastRank is robust across a broad range of threshold values on these real-world tensors. In particular, the default choice $t = 95\%$ performs consistently well, and the thresholds $t = 90\%$ and $t = 80\%$ achieve the same lowest MAE. By contrast, the very high threshold $t = 99.99\%$ leads to clear overestimation on several datasets, especially `dorrit` and `sugar`, suggesting that such an aggressive retention of spectral energy is not well suited to noisier real-world data. This observation is consistent with the synthetic results above. That is, high values of t are beneficial in low-noise regimes, whereas more moderate thresholds provide improved robustness when the data contain substantial noise or modeling imperfections.

Overall, these results support the use of a moderate threshold as a reliable default in practice. They also reinforce the interpretation that t governs the balance between preserving informative spectral structure and avoiding spurious components induced by noise. Finally, as a practical heuristic for selecting t without extensive cross-validation, we have empirically found that visual inspection of the CSE curve is effective. By plotting the cumulative spectral energy against the number of singular values for a few samples from the data, one can identify the "elbow" of the curve, which corresponds to the point of diminishing returns in energy capture. Selecting the threshold t at this elbow provides a robust estimate that balances signal preservation with noise filtering for the data at hand.

C.2 FastRank under Missing-Value Noise

We additionally examine the applicability of FastRank in the presence of missing-value corruption, which is relevant for incomplete tensor data such as partially observed images or videos. Instead of considering additive perturbations, we model missingness through elementwise masking:

$$T_{\text{obs}} = T \odot \mathcal{N}, \text{ with } \mathcal{N}[i,j,k] \sim \text{Bernoulli}(1-\alpha),$$

where $\odot$ denotes elementwise multiplication and α is the missingness probability. Under this model, the masking tensor contains zeros with probability α and ones with probability $1-\alpha$.

To evaluate FastRank in this setting, we replicate the synthetic experimental setup used in the main paper, generating 300 tensors with maximum dimension at most 100 and true rank bounded by $\min\{I, J\}$. We then apply FastRank using different threshold levels t across several missingness levels α. The resulting MAE values are reported in Table 8.

Table 8: MAE for Missing-Value Noise, for different levels of noise α and thresholds $t\%$.

$t\%\backslash\alpha$	$\alpha = 0.01$	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.5$
$t = 99.99$	**8.55**	**15.10**	**13.38**	19.73
$t = 99$	23.57	21.22	24.97	**15.61**
$t = 95$	26.43	26.28	30.45	26.47

The results reveal a trend that is consistent with the additive-noise setting. In particular, the high threshold $t = 99.99\%$ performs best up to 20% missingness, while under severe corruption (50% missingness) the lower threshold $t = 99\%$ yields the lowest error. This behavior qualitatively mirrors the one observed in Figure 4, indicating that the effect of missing-value corruption on the singular spectrum is similar to that of additive noise. Overall, these findings suggest that FastRank remains robust across different noise models, and that the choice of threshold t continues to govern the trade-off between signal preservation and robustness to corruption.

C.3 Comparison with Unfolding-Based Matricization

A natural question is how FastRank compares to rank-estimation procedures based on tensor unfolding. Unfolding generally produces a larger matrix and therefore increases the maximum detectable matrix rank. At first sight, this may appear advantageous. However, our focus throughout the paper is the practically relevant low-rank regime, where the latent tensor rank already lies within the detectable range of the sum-reduced matrix. In this setting, the main distinction between the two approaches is not detectability, but rather the trade-off between robustness to noise and computational efficiency.

Why sum-reduction can be preferable in the low-rank, noisy regime. Assume that we observe a noisy tensor

$$T_{\text{obs}} = T + \mathcal{N},$$

of size $I \times J \times K$ with $I \geq J \geq K$, where T is low-rank and $\mathcal{N}$ is typically high-rank. Summing along the third mode yields an observed matrix

$$\mathbf{M}_{\text{obs}} = \mathbf{M} + \mathbf{M}_{\text{noise}},$$

of size $I \times J$, where $\mathbf{M}$ and $\mathbf{M}_{\text{noise}}$ are the sum-reduced matrices associated with T and $\mathcal{N}$, respectively. By subadditivity of matrix rank,

$$\text{rank}(\mathbf{M}_{\text{obs}}) \leq \text{rank}(\mathbf{M}) + \text{rank}(\mathbf{M}_{\text{noise}}),$$

while the matrix dimensions imply

$$\text{rank}(\mathbf{M}_{\text{obs}}) \leq \min(I, J) = J.$$

Hence,

$$\text{rank}(\mathbf{M}_{\text{obs}}) \leq \min\big(J,\ \text{rank}(\mathbf{M}) + \text{rank}(\mathbf{M}_{\text{noise}})\big).$$

Since the noise component is typically high-rank, one is often in the regime where the dimension bound dominates, so that the effective observed rank remains capped by J. This has two important consequences. First, the

admissible rank range of the observed matrix is tightly controlled. Second, even when the noise tensor is high-rank, its sum-reduced matrix cannot exceed rank J, making it less dominant in the final eigenspectrum.

By contrast, unfolding along mode three produces

$$\mathbf{U}_{\text{obs}} = \mathbf{U} + \mathbf{U}_{\text{noise}},$$

where $\mathbf{U}_{\text{obs}}$ has size $I \times (JK)$. In this case,

$$\text{rank}(\mathbf{U}_{\text{obs}}) \leq \min\big(I, JK,\ \text{rank}(\mathbf{U}) + \text{rank}(\mathbf{U}_{\text{noise}})\big),$$

so the admissible rank range is much larger. This enlarged range can be beneficial when the underlying rank is high, but it also allows the noise contribution to occupy a substantially larger portion of the spectral space. As a result, eigenspectrum-based heuristics may become more prone to overestimation under unfolding than under sum-reduction. In this sense, sum-reduction acts as a structural compression of the estimation problem: it restricts the set of admissible rank values and attenuates the effective contribution of high-rank noise in the observed matrix.

Noiseless comparison. To make this distinction concrete, we compare sum-reduction and unfolding on third-order tensors of size $100 \times 10 \times 10$, where sum-reduction yields matrices of size 100×10, while unfolding along mode three yields matrices of size 100×100. We vary the CP rank across $k \in \{1, 5, 10, 15, 20\}$ and generate batches of 100 tensors for each value of k. In the noiseless case, matrix rank is computed directly, without thresholding.

Table 9 reports the MAE. As expected, both approaches recover the correct rank whenever it lies within the detectable range of the sum-reduced matrix, namely for $k \leq 10$. For larger values of k, the sum-reduced matrix cannot represent ranks above 10, and the corresponding error simply reflects the gap $k - 10$. Unfolding, in contrast, continues to recover the exact rank because its larger matrix dimensions allow for a wider admissible range.

Table 9: MAE versus rank k without noise, comparing sum-reduction and unfolding on tensors of size $100 \times 10 \times 10$.

Method \ Rank	$k = 1$	$k = 5$	$k = 10$	$k = 15$	$k = 20$
Sum-Reduction + Matrix Rank	0.00	0.00	0.00	5.00	10.00
Unfolding + Matrix Rank	0.00	0.00	0.00	0.00	0.00

The corresponding runtimes are reported in Table 10. Even in this relatively small-dimensional setting, sum-reduction is substantially faster, with speedups ranging from roughly one to two orders of magnitude depending on k. This already illustrates the practical gain of working on the smaller sum-reduced matrix.

Table 10: Runtime (milliseconds) versus k without noise, comparing sum-reduction and unfolding on tensors of size $100 \times 10 \times 10$.

Method \ Rank	$k = 1$	$k = 5$	$k = 10$	$k = 15$	$k = 20$
Sum-Reduction + Matrix Rank	0.330	0.301	0.372	0.278	0.125
Unfolding + Matrix Rank	22.299	22.014	15.631	5.637	0.794

Noisy comparison. We repeat the same experiment under the default additive-noise model used throughout the paper. Rank estimation is then performed via thresholding. For the sum-reduced version, we use the default threshold from the main text $t = 95\%$. For unfolding, we evaluate thresholds in $\{99.99, 99.9, 99, 95, 90\}$ and report the best-performing value for each case.

The results are shown in Table 11. In the low-rank regime $k \leq 10$, sum-reduction consistently yields lower MAE than unfolding. For $k > 10$, unfolding becomes preferable, as expected, because the tensor rank exceeds the maximum detectable rank of the sum-reduced matrix. Thus, the comparison confirms the intended scope of FastRank: when the tensor is genuinely low-rank relative to its dimensions, sum-reduction offers a better accuracy–efficiency trade-off than unfolding-based alternatives.

Table 11: MAE versus rank k with additive noise, comparing sum-reduction and unfolding on tensors of size $100 \times 10 \times 10$.

Method \ Rank	$k = 1$	$k = 5$	$k = 10$	$k = 15$	$k = 20$
Sum-Reduction + Thresholding	2.35	2.22	2.89	11.62	17.40
Unfolding + Thresholding	2.70	2.41	3.09	6.54	10.83

Overall, the comparison with unfolding-based matricization highlights three points. First, unfolding indeed enlarges the maximum detectable rank, which can be advantageous outside the low-rank regime. Second, when the tensor rank lies within the detectable range of the sum-reduced matrix, FastRank preserves the spectral information needed for accurate estimation while being substantially faster. Third, in noisy low-rank settings, restricting the admissible rank range through sum-reduction appears to improve robustness by limiting the influence of high-rank noise on the observed spectrum.

C.4 Scalability Analysis

We also perform a scalability analysis on synthetic tensors to showcase FastRank's applicability.

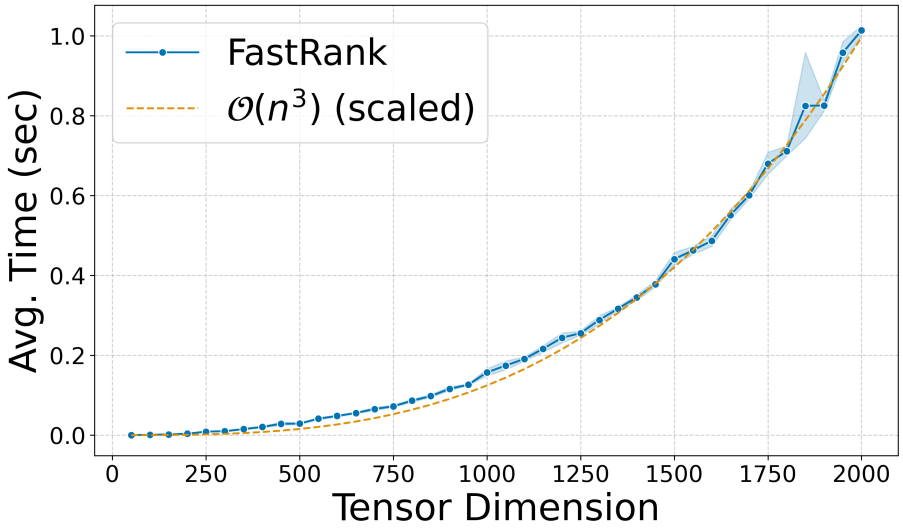


Figure 5: Average runtime of FastRank on 3D dense tensors, as we increase the tensors' maximum dimension and the corresponding $\mathcal{O}(n^3)$ curve. Notably, for a dense tensor of shape $2000 \times 2000 \times 2000$, the runtime is around 1 second.

In Fig. 5, we plot the average runtime of FastRank across a range of tensor dimensions to evaluate its empirical scalability. We synthetically generate dense, third-order tensors of shape $I \times I \times I$, where I ranges from 50 to 2000 in increments of 50. At each dimension, 50 independent tensors are sampled, and FastRank is executed on each instance; the average runtime (in seconds) is then recorded. The resulting curve is juxtaposed with a scaled $\mathcal{O}(n^3)$ reference line, representing the dominant asymptotic complexity incurred by the core singular value decomposition (SVD) step within FastRank. As the plot demonstrates, the observed runtimes closely follow the cubic scaling trend, especially as I increases, thereby validating the expected theoretical behavior.

Importantly, FastRank demonstrates favorable runtime characteristics for practical problem sizes. For instance, even for a dense tensor of shape $2000 \times 2000 \times 2000$, the average execution time remains under one second on standard hardware. This highlights the method's efficiency and makes it a feasible solution for medium-scale tensor problems where exact SVD remains computationally viable.

It is important to emphasize that these results pertain to dense tensor inputs. In scenarios involving sparse tensors (and thus sparse matrices), state-of-the-art sparse SVD algorithms, such as those based on Lanczos bidiagonalization, attain significantly better asymptotic behavior [Simon and Zha, 2000]. In particular, sparse

SVD methods typically achieve complexity on the order of $\mathcal{O}(T \cdot (nnz(A)\, k + k^3))$, where $nnz(A)$ is the number of non-zeros in the input, k is the target rank, and T is the number of iterations to convergence. Because $nnz(A) \ll n^2$ for large sparse inputs, the effective scaling can be far below $\mathcal{O}(n^3)$.