Electronic Colloquium on Computational Complexity, Report No. 73 (2026)

An Algorithmic Proof of Kruskal's Tensor Decomposition Theorem

Vishwas Bhargava* Leonard J. Schulman† Shiri Sivan‡

Abstract

A famous theorem of Kruskal gives the simplest and arguably most fundamental criterion under which a tensor is guaranteed a unique minimum-rank decomposition. Kruskal's condition requires that the sum of the Kruskal ranks $\{k_i\}_{i=1}^m$ of the components satisfies $\sum_{i\in[m]} k_i \geq 2r + m - 1$, where r denotes the rank and m the order of the tensor. However, Kruskal's original proof and subsequent simplifications/generalizations have remained non-constructive. With the sole exception of the case $(k_1 = r, k_2 = r, k_3 = 2)$, attributed to Jennrich—no algorithm has been established for decomposing tensors under the Kruskal condition without additional assumptions. In fact, whether there exists an efficient algorithm for decomposing a tensor under the Kruskal condition was explicitly posed as an open problem in the work of Bhaskara et al. (COLT 2014). Even slight variations of the Jennrich special case, such as the $(r, r-1, 3)$ case, have remained algorithmically open; specifically, no sub-exponential time bound was known.

In this work, we make progress on this problem by giving an elementary, constructive proof of Kruskal's Theorem for general m-way tensors. Concretely, we present a randomized algorithm that decomposes any tensor satisfying the Kruskal condition by utilizing random projections to map the problem into a geometry of intersecting hyperplanes via a MinRank instance. Specifically for 3-way tensors satisfying $k_1+k_2+k_3 = 2r+2$, the algorithm achieves a runtime of $n^{O(k)}$ where $k = \min(k_1, k_2, k_3)$. Thus, we extend smoothly beyond the Jennrich special case, achieving polynomial-time complexity for any family of tensors that satisfies the Kruskal condition, provided the least Kruskal rank is bounded.

1 Introduction

Tensor decomposition is a fundamental tool for modern science, with applications ranging from machine learning and statistics to signal processing and computational complexity. In a nutshell, tensors are multi-dimensional arrays with entries from a field $\mathbb{F}$. For instance, a 3-dimensional (or 3-way or 3-mode) tensor can be written as $\mathcal{T} = (\alpha_{i,j,k}) \in \mathbb{F}^{n_1\times n_2\times n_3}$.

A tensor $T \in \mathbb{F}^{n_1\times\cdots\times n_m}$ can always be written as the sum of rank 1 tensors; such an expression is known as a *tensor decomposition*.[1] Here, a rank 1 tensor over a field $\mathbb{F}$ is the outer product of m nonzero vectors, written as

$$v_1 \otimes v_2 \otimes ... \otimes v_m \in \mathbb{F}^{n_1\times n_2\times\cdots\times n_m}.$$

*Department of Computer Science and Engineering, IIT Bombay, India. Email: vishwas@cse.iitb.ac.in. Part of this work was done when the first author was a postdoc at Caltech.

†Department of Computing and Mathematical Sciences, Caltech, USA. Email: schulman@caltech.edu. Supported in part by NSF CCF-2321079.

‡Department of Computing and Mathematical Sciences, Caltech, USA. Email: ssivan@caltech.edu. Supported in part by NSF CCF-2321079

[1] Also known as the CANDECOMP/PARAFAC (CP) decomposition.

ISSN 1433-8092

If the decomposition consists of r summands, it is called an r-decomposition. The minimal r for which there exists an r-*decomposition* is defined as the *tensor rank* of T, and such a decomposition is then referred to as a *rank decomposition*.

When T has a decomposition

$$T = \sum_{t=1}^{r} v_t^{(1)} \otimes v_t^{(2)} \otimes \cdots \otimes v_t^{(m)},$$

we define the *components* as, for each $i \in [m]$, the matrix with columns $v_t^{(i)}$ $(t = 1, \ldots, r)$:

$$V^{(i)} = [v_1^{(i)}, v_2^{(i)}, \ldots, v_r^{(i)}] \in \mathbb{R}^{n_i \times r}$$

The vector space containing $v_t^{(i)}$ will be referred to as the i'th *mode* of the tensor. We also employ the shorthand notation

$$T = [\![V^{(1)}, V^{(2)}, \ldots, V^{(m)}]\!].$$

A primary reason for the wide applicability of tensors is that their rank decomposition can be unique (also known as *identifiable*), a property that matrices (unless of rank 1) lack. The motivations for identifiability in data analysis (beginning at least with work of the psychologist Spearman on tests of intelligence [Spe61]) are abundantly discussed in many references e.g., [CC70, Har70, RSG17]. We refer the interested reader to these works and the references therein.

The most celebrated criterion for uniqueness of tensor decomposition was given by Kruskal in 1977 [Kru77] (and see [Kru76, Kru89]). The *Kruskal rank* (or k-rank) of a matrix is the maximum integer k such that every subset of k columns is linearly independent. Clearly, k-rank $\leq$ rank. Kruskal's uniqueness theorem states that if a tensor T has an r-decomposition $[\![V^{(1)}, \ldots, V^{(m)}]\!]$, and the k-ranks of the components satisfy

$$\sum_{i=1}^{m} k_{V^{(i)}} \geq 2r + m - 1, \tag{1}$$

then $\text{rank}(T) = r$ and the decomposition is unique up to permutation and scaling of the columns (going forward we omit the "up to scaling and permutation"). The inequality in (1) is the *Kruskal Condition*.

While the theorem guarantees that a unique decomposition exists, the original proof and subsequent simplifications (e.g., [SS07, Rho10, Lan11]) are existential rather than constructive. This limitation is widely recognized. The question of whether there is a hardness result for tensor decomposition, or conversely, whether there exists an efficient (polynomial-time) algorithm for decomposing a tensor under the general conditions of Kruskal's uniqueness theorem—and whether there is an algorithmic proof of the theorem itself—was explicitly posed as a major open problem in COLT 2014 by Bhaskara et al. [BCMV14], carrying a $100 reward.

It is important to note that while the tensor rank problem (and therefore also the problem of finding a decomposition of specified rank) is NP-hard, it remains unknown whether this hardness persists when the input tensor is promised to satisfy the Kruskal condition. The only significant special case of the Kruskal condition in which an efficient algorithm is known is the case that is due (prior to Kruskal's work) to Jennrich, in which two of the components are invertible ($k_1 = k_2 = r$) while the third has Kruskal rank at least 2. Even this restricted case has found surprisingly many applications in computer science and statistics. However, even a slight variation, such as the $(r, r-1, 3)$ case, has, to the best of our knowledge, remained algorithmically open. This was the original motivation for our work.

1.1 Our contribution

We give the first algorithmic proof of the Kruskal theorem. Rather than relying on an existential argument, we use structural implications of the Kruskal condition to recover the components directly. For ease of presentation, we state our results over the real field $\mathbb{R}$. Our algorithm and its runtime guaranty are stated in the Real RAM model, where arithmetic over the reals is exact and of unit cost. However, the algorithm can be easily implemented over the rationals using standard Turing machines. That is, if the unknown decomposition is in fact rational, then our algorithm can work entirely over the rationals. There is a known polynomial-time reduction (in the input size $\prod n_i$) that converts an m-way tensor satisfying the Kruskal condition into a 3-way tensor that also satisfies the Kruskal condition (see Theorem 3 in [SB00] or Lemma A.1 in the Appendix).[2] Consequently, we focus our analysis on the 3-way case.

As our objective is to design a decomposition algorithm for Kruskal tensors, we assume T satisfies the Kruskal condition:

$$k_1 + k_2 + k_3 \geq 2r + 2.$$

We set $k := 2r+2-\max\{k_1+k_2, k_1+k_3, k_2+k_3\}$. This value governs both the time complexity and the core technical arguments of our algorithm.

Theorem 1.1 (Algorithmic Kruskal for 3-tensors)**.** *Let $\mathcal{T} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ be a tensor with a decomposition $\mathcal{T} = \sum_{i=1}^{r} u_i \otimes v_i \otimes w_i$ such that the components have Kruskal ranks respectively k_1, k_2, k_3 satisfying the Kruskal condition $k_1 + k_2 + k_3 \geq 2r + 2$. Then there exists a randomized algorithm that recovers the components in $(n_1 + n_2 + n_3)^{O(k)}$ time with high probability.*

By definition, $k \leq \min\{k_1, k_2, k_3\}$. If the Kruskal condition holds with some slack (i.e., $\sum k_i > 2r + 2$), we can effectively replace the smallest Kruskal rank with this smaller value k that saturates the bound. For example, if all Kruskal ranks are r, then $k = 2$.

Our work settles the request of Bhaskara et al. [BCMV14] for an algorithmic proof of the Kruskal theorem, and makes substantial progress on the efficiency question without fully settling it: we provide a time complexity upper bound for decomposing Kruskal tensors that scales exponentially with respect to the minimum Kruskal rank of the modes. Notably, for (previously open) instances "close" to the Jennrich case—such as the $(r, r-p, p+2)$ regime for fixed p—our algorithm is polynomial in the input size. By contrast, the best existing method, the brute-force approach of reducing the decomposition to an algebraic system, will typically require $\exp(nr)$ time for $n = \max n_i$ due to the nr unknowns involved.

Built-into the Kruskal setting is the fact that the Kruskal condition is NP-hard to verify; see [Kha95]. Consequently, there is no efficient way to certify that the resulting output components truly constitute a rank decomposition without already knowing that the input tensor satisfies the Kruskal condition.

We should mention that in the m-to-3 reduction alluded to above, the initial m-way tensor does not strictly need to satisfy the Kruskal condition; it is sufficient that it should reduce to a 3-way instance which does. The reduction is achieved by "clubbing" various modes together through Khatri-Rao products—a standard reshaping technique in the literature. (For the m-to-3 reduction see [SB00, LP23, COV17], or Sections 5 and A.) By partitioning the modes into sets $I, J,$ and L

[2] Lemma A.1 proves that every Kruskal m-mode tensor has a Kruskal 3-reshaping. Finding this reshaping can be accomplished in polynomial-time since the number of ways to reshape the tensor is polynomial in the input size.

such that the resulting reshaped tensor satisfies the 3-way Kruskal criteria, we effectively provide a constructive proof for the more general *Reshaped Kruskal Theorem*.

Theorem 1.2 (Reshaped Kruskal Theorem)**.** *Consider an m-way tensor $\mathcal{T}$ with an r-rank decomposition $[\![V^{(1)}, \ldots, V^{(m)}]\!]$. If there exists a partition of the modes $I \sqcup J \sqcup L = [m]$ such that the Kruskal ranks of the partitioned (Khatri-Rao) matrices satisfy:*

$$k_I + k_J + k_L \geq 2r + 2$$

then $\operatorname{rank}(\mathcal{T}) = r$ *and the decomposition is unique. Furthermore, there exists a randomized algorithm that recovers the components in $N^{O(k)}$ time with high probability, where N is the input size, and $k := 2r + 2 - \max\{k_I + k_J, k_I + k_L, k_J + k_L\}$.*

1.2 Related Work

Tensor decomposition is a highly active area of research with many fundamental open problems and landmark results. Here, we mention some of these results with a focus on the identifiability and computational complexity of the problem (and omitting entirely the extensive literature in algebraic complexity theory).

On identifiability, the work of Lovitz and Petrov [LP23] provides an elegant generalization of Kruskal's theorem using ideas from matroid theory. It is not clear whether our method can be extended to their generalization.

On the complexity and algorithmic side, determining tensor rank is famously NP-hard [Hås90, HL13]. Consequently, one cannot expect efficient algorithms to determine rank or find decompositions for general input tensors. The Kruskal framework is the least restrictive uniqueness condition in which hardness is not known. The Kruskal condition is also tight [Der13]. From this point of view our result is a *worst-case* tensor decomposition algorithm, unlike "generic" uniqueness or decomposition theorems. Indeed, most research on tensor decomposition algorithms has focused on tensors with additional non-degeneracy conditions; tensors drawn from specific distributions; or heuristic approaches. We discuss some of these results now, moving from no assumptions to progressively adding more structure; as we will see, the assumptions have a tremendous effect on the rank parameters we can efficiently decompose.

In the worst-case setting for 3-dimensional tensors, one essentially cannot outperform brute force, which involves solving systems over the base field. Formally, Schaefer and Stefankovic [SŠ18] showed that for *any* field $\mathbb{F}$, given a system S of algebraic equations over $\mathbb{F}$, we can in polynomial time construct a 3-dimensional tensor $\mathcal{T}_S$ (with a linear blowup) and an integer k such that S has a solution in $\mathbb{F}$ *if and only if* $\mathcal{T}_S$ has rank at most k over $\mathbb{F}$. However, for higher m-dimensional tensors, there exists an FPT-style worst-case algorithm with $2^{r^{O(1)}}\operatorname{poly}(n, m)$ runtime [BS25]. The next level is the generic setting, where the input tensor can be anything barring a (result-specific) lower-dimensional algebraic variety. For generic tensors, Chiantini and Ottaviani [CO12] prove the uniqueness of decomposition up to rank $\frac{(n_1-1)(n_2-1)(n_3-1)+1}{n_1+n_2+n_3-2}$. Again, in the generic setting, Kothari-Moitra-Wein [KMW24] recently provided an efficient decomposition algorithm for families of tensors with rank approaching $2n$. Adding even more structure (and moving towards applications), one has the average-case setting; for instance, Ma et al. [MSS16] deployed techniques from the Sum-of-Squares hierarchy to reach rank $\tilde{O}(n^{1.5})$ for tensors drawn from Gaussian decomposition.

2 Proof Overview

We focus our analysis on 3-way tensors, as decomposition of m-way tensors can be reduced in a standard fashion to the 3-way case through the reshaping procedure alluded to in Section 1.1. Let $T \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ be a tensor that satisfies the Kruskal condition $k_1 + k_2 + k_3 \geq 2r + 2$, and has a decomposition $T = \sum_{j=1}^{r} u_j \otimes v_j \otimes w_j$. Since part of our goal is to reprove the Kruskal theorem, we shall not assume that this decomposition is unique. As per our notation, we define components $U = [u_1, \ldots, u_r] \in \mathbb{R}^{n_1 \times r}$, $V = [v_1, \ldots, v_r] \in \mathbb{R}^{n_2 \times r}$, and $W = [w_1, \ldots, w_r] \in \mathbb{R}^{n_3 \times r}$.

We start by recovering a single component, say V. Towards this end, we turn to a geometric point of view. Namely, we complete an equivalent task: recovering the r hyperplanes orthogonal to the columns of V. This task is completed in two steps. First we *identify* all the intersection points of the hyperplanes (collectively these form a zero-dimensional variety); then we *reconstruct* the hyperplane arrangement, given the intersection points. We refer to the combination of these two steps as the *identifiability test*, as this test establishes the identifiability of the (projected) component V.

Both steps require that the hyperplane arrangement lie in general position, meaning the intersection of any t hyperplanes has dimension $\max\{d - t, 0\}$. Conveniently, this can be accomplished with probability 1 via a random projection (see Lemma 4.1).

The identifiability test

To identify the intersection points, we construct a MinRank instance[3] whose solutions correspond precisely to the aforementioned intersection points. Let $T' = T \times_2 A = [\![U, AV, W]\!]$ be the projected tensor, where $A \in \mathbb{R}^{k_2 \times n_2}$ is a random matrix ($\times_i A$ means that the i-th mode of the tensor is contracted against the rows of matrix A). For variables $z = (z_1, \ldots, z_d)$ where $d = k_2 - 1$, and $[z, 1]$ is the vector $(z_1, \ldots, z_d, 1)$, we define the V-mode contraction:

$$M(z) = T' \times_2 [z, 1] = U \mathrm{Diag}([z, 1]AV) W^T. \tag{2}$$

In words, $M(z)$ is the linear combination of 2nd mode slices associated with coefficients $[z, 1]$. Under the Kruskal condition, we show that for all $z \in \mathbb{R}^d$, $\mathrm{rank}(M(z)) \geq r - d$. Crucially, we prove that the matrix $M(z)$ achieves this minimum rank $r - d$ if and only if $[z, 1]AV$ has exactly d zeros.

We only look at slice combinations of the form $[z, 1]$ because fixing the last entry ensures that d-intersections are 0-dimensional (points). A more subtle reason for this choice is the projective nature of tensor decomposition; since we can scale components freely, we have the freedom to look for one fewer variable in the search for these hyperplanes.

Geometrically, this implies that the minimizers of rank $M(z)$ are precisely the d-wise intersection points of the hyperplanes defined by the columns of the projected component AV (see Theorem 4.2). This already is a significant step toward identifiability, as the matrix $M(z)$ depends solely on the observable tensor rather than the hidden decomposition itself. We are essentially reducing tensor decomposition to the search problem associated with MinRank. We denote

$$\mathrm{MinRank}(M(z), \ell) = \min_{z \in \mathbb{R}^{\ell - 1}} \mathrm{rank}(M(z)), \tag{3}$$

[3]The MinRank search problem is the task of finding a linear combination of given matrices $M_1, \ldots, M_n$ that minimizes the rank of the resulting matrix. See e.g., [FSEDS10, BBC$^+$20]

where $M(y) := W\mathrm{Diag}([y,1]AV)W^T$ for a random $A \in \mathbb{R}^{\ell \times n_v}$.[4]

It is a geometric fact that the set of d-wise intersection points of an arrangement of hyperplanes in general position in $\mathbb{R}^d$ determines those hyperplanes; and in addition, the hyperplanes can be efficiently computed from the intersection points. This allows us to reconstruct the matrix AV (see Algorithm HP-reconstruction and Lemma 4.5).

The identifiability test is the heart of our algorithmic proof of Kruskal's Theorem. Establishing the uniqueness of the projected component AV via this test effectively reduces the problem to the Jennrich setting, where uniqueness is well-understood and efficient decomposition applies (see Lemma 4.6). With U and W at our disposal, the identifiability of the original V follows.

The tensor decomposition algorithm

Our decomposition algorithm, sketched below, is directly based on the ideas we discussed above for identifiability. For convenience, at each step we link to the text for more details.

Algorithm 1 Rank decomposition algorithm for 3-way tensors (outline)

Input: $T \in \mathbb{R}^{n_1 \times n_2 \times n_3}$.

For $k' = 2, 3, ..., \min\{n_1, n_2, n_3\}$:

For $i = 1, 2, 3$:

1. Apply a random projection $A \in \mathbb{R}^{k' \times n_i}$ on the i-th mode (Step 1).
2. Generate $M(z)$ = the formal linear combination of mode-i slices with coefficients $(z_1, ..., z_{k'-1}, 1)$ (Step 2).
3. Solve MinRank$(M(z), k')$. (Step 3)
   (a) Denote the solution m. (see Equation 3 for the definition of MinRank)
   (b) Calculate the minimizers $\mathcal{S} = \{z \in \mathbb{R}^{k'-1} |\ \mathrm{rank}\, M(z) = m\}$.
   (c) Set $r' = m + k' - 1$.
4. Reconstruct $AV^{(i)}$ (Step 4).
5. Restore the two other components $V^{(j)}, V^{(\ell)}$ (Step 5).
6. Recover $V^{(i)}$ (Step 6).
7. If $T = [\![V^{(1)}, V^{(2)}, V^{(3)}]\!]$, output the decomposition.

Output **fail**.

We employ randomness in Steps 1, 3 and 4. The successful event in Step 1 occurs with probability 1 (see Lemma A.2), whereas the randomness in Steps 3 and 4 is utilized solely for the sake of computational efficiency and does not affect the uniqueness argument. Concretely, each randomized step admits a deterministic alternative: the random projection can be replaced by going over a sufficiently large grid (by Schwartz-Zippel); the system solving step (and thus MinRank) can be performed deterministically (for instance using the algorithm of Ierardi [Ier89]); and the hyperplane

[4]The parameter ℓ is the number of matrices in the MinRank instance, since $M(z)$ is a sum of the matrices $\{W\mathrm{Diag}(a_i^* V)W^T\}_{i\in[\ell]}$, where a_i^* denotes the i-th row of A.

reconstruction can be performed via the deterministic brute-force strategy described in Section 4.2 (specifically, Lemma 4.4 and the subsequent discussion).[5] Thus the algorithm can in principle be made fully deterministic, and its output depends only on the input tensor — not on any random choices.

Thus, the algorithm's output is uniquely determined for each i and k'. It only remains to show that different parameter values cannot result in distinct decompositions. Towards this end, we prove the existence of fixed parameters k and i, determined solely by T, for which the recovery algorithm is guaranteed to succeed. Specifically, if T admits a rank decomposition satisfying the Kruskal condition, the algorithm—called on these parameters—is guaranteed to recover the constituent components. Because the algorithm is deterministic, it maps the tensor T to a unique set of factors (up to scaling and permutation). If two distinct decompositions were to exist, both would necessarily be recovered as the same output by this deterministic procedure, which is a contradiction. Consequently, the decomposition is unique.

This parameter k is defined by

$$k := 2r + 2 - \max\{k_1 + k_2, k_1 + k_3, k_2 + k_3\},$$

and the corresponding i is the mode of minimal Kruskal rank. That is, $k \leq k_i = \min\{k_1, k_2, k_3\}$. [6]

Since we do not know the tensor rank and Kruskal ranks we iterate over different k guesses in the outermost loop of the algorithm. The correctness of the algorithm, as well as the uniqueness of the decomposition, rely on the following facts:

1. If an r-decomposition satisfies the Kruskal condition, then the tensor has rank r and every rank decomposition has the same Kruskal ranks and thus k (see Lemma 4.7).

2. At iterations $k' < k$, the decomposition algorithm does not output a 'redundant' decomposition, i.e., an r'-decomposition with $r' > r$ (see Lemma 4.8).

3. At iteration k, the algorithm outputs a (unique) r-decomposition (see Theorem 4.9).

The proofs of facts 1 and 2 are provided in Section 4, while an overview of the 3rd fact follows below. For technical details regarding the subroutines, time complexity, and success probability, see Section 4.

For fact 3, we now "walk" through iteration (k, i) of the algorithm, assuming the success event occurs at every step of the algorithm.

Step 1. Apply a random projection.

A random projection will ensure that $AV^{(i)}$ (for $i \in [3]$) has full Kruskal rank with high probability. The success event implies that the hyperplanes perpendicular to the columns of $Q := AV^{(i)}$ lie in general position. More formally, denote the columns of Q by $q_1, ..., q_r$. For all $i \in [r]$ we define the hyperplane

$$H_i = \{z \in \mathbb{R}^d | \ \langle [z, 1], q_i \rangle = 0\}.$$

We denote the full set of hyperplanes by $\mathcal{H} = \{H_1, ..., H_r\}$. By Lemma 4.1, $\mathcal{H}$ lies in general position with probability 1.

[5] The deterministic version can be implemented in $r^{O(d^2)}$ time, compared to $r^{O(d)}$ for the randomized version.

[6] (k, i) are not the only parameters on which the algorithm is guaranteed to succeed. However, for uniqueness we only need one set of parameters that succeed on all decompositions. Computationally, the choice (k, i) delivers the best time complexity.

Step 2. Generate a MinRank instance

The MinRank instance generated at this step is the linear combination of i-mode slices associated with coefficients $(z_1, ..., z_d, 1)$, defined by

$$M(z) = T \times_i [z, 1]A = V^{(j)}\text{Diag}([z, 1]AV^{(i)})(V^{(\ell)})^T.$$

Step 3. Solve the MinRank instance

The above specific factorization of $M(z)$ coupled with the Kruskal structure of the factor matrices $V^{(j)}$ and $V^{(\ell)}$ ensures that the rank drops precisely when z lies on the intersection of the hyperplanes (see Theorem 4.2). Thus we have,

$$\textsc{MinRank}(M(z), k) = r - d$$

so at Step 3c we set $r' = r$, and then at 3b we find all the minimizers

$$\mathcal{S} = \{z \in \mathbb{R}^d | \text{ rank } M(z) = r - d\}.$$

By Theorem 4.2 this collection of points is exactly the set of d-wise intersection points of hyperplanes in $\mathcal{H}$.

Step 4. Reconstruct the projected component

Since $\mathcal{H}$ is in general position in $\mathbb{R}^d$, all d of the hyperplanes intersect at a unique point, and each hyperplane contains exactly $\binom{r-1}{d-1}$ points from $\mathcal{S}$. Moreover, any hyperplane in $\mathbb{R}^d \backslash \mathcal{H}$ contains fewer points from $\mathcal{S}$ (see Lemma A.4) so we can reconstruct $\mathcal{H}$ by iterating over subsets of d points, generating a list of candidate hyperplanes, and checking which ones contain the characterizing number of points (see Algorithm HP-reconstruction for a randomized version of this algorithm).

Recovering $\mathcal{H}$ is equivalent to the recovery of $Q = AV^{(i)}$ (up to scaling and permutation of the columns), because by definition, the columns of Q are the normals to the hyperplanes.

Step 5. Restore the remaining components $V^{(j)}$ and $V^{(\ell)}$

With the projected component $Q := AV^{(i)}$ in hand we recover the components $V^{(j)}$ and $V^{(\ell)}$ by essentially reducing the problem to the "Jennrich setting," and applying the Jennrich decomposition algorithm (see Lemma 3.8).

Specifically, we find a matrix $P \in \mathbb{R}^{2\times k}$ perpendicular to the first $k - 2$ columns of Q, and effectively "zero out" those summands in the decomposition. The resulting tensor now has a reduced number of components, $r' = r - k + 2$. Furthermore, this reduction transforms the tensor from an initial (k_1, k_2, k_3) configuration to a "Jennrich state" where the Kruskal ranks of the j and ℓ modes are r', while the projected i-mode's Kruskal rank becomes 2. The fact that the truncated components have full rank r' follows from the Kruskal condition ($\sum k_i \geq 2r + 2$ implies that $k_j, k_\ell \geq r' = r - k + 2$). This ensures that the truncated tensor satisfies the requirement for Jennrich's algorithm. The remaining component can be recovered directly by applying p_j with $\langle p_j, q_i \rangle = \delta_{i,j}$ (see Algorithm 5). Thus, we can uniquely recover $V^{(j)}$ and $V^{(\ell)}$ (in polynomial time).

Step 6. Recover $V^{(i)}$

We recover $V^{(i)}$ by solving the linear system obtained from the entries of T (which become linear in the entries of $V^{(i)}$ once we plug in the values of $V^{(j)}$ and $V^{(\ell)}$).

Generalization to m-way tensors

The transition from 3-way to general m-way tensors is handled via a 3-reshaping process. Any m-dimensional tensor satisfying the Kruskal condition can be partitioned into a 3-way tensor through Khatri-Rao products of its components.

We prove that if the original tensor satisfies the Kruskal condition, there exists a partition $I \sqcup J \sqcup L = [m]$ such that the resulting 3-way tensor also satisfies the Kruskal condition. After decomposing this reshaped 3-way tensor to find the partitioned components (e.g., V_I, V_J, V_L), we recover the original components $\{V^{(i)}\}_{i\in[m]}$ by exploiting the 'injectivity' (up to scaling) of the Khatri-Rao product. This extends our "worst-case" guarantee to any m-dimensional tensor satisfying the Kruskal bound. In terms of time complexity, the search for a 3-partition satisfying the Kruskal condition contributes a multiplicative factor of $O(3^m)$ which is polynomial in the input size.

2.1 Outline of the paper

Section 3 collects definitions and notation (including some given previously). In Section 4 we formally present the decomposition algorithm, analyze it, and provide auxiliary lemmas and subroutines. Section 5 contains the proof of the main theorem. Finally, the Appendix contains more technical details, some of which may be well-known.

3 Preliminaries

For matrix $M \in \mathbb{R}^{m\times n}$, row indices $R \subseteq [m]$, and column indices $C \subseteq [n]$, we denote $M(R, C)$ the submatrix of M induced by R and C. We also denote $M(C) := M([m], C)$. We denote the rows of M by m_i^* for $i \in [m]$ and the columns by m_j for $j \in [n]$ (generally, rows/columns are denoted by the lower case letter of the matrix notation).

Definition 3.1 (Kruskal rank)**.** *The* Kruskal rank *(or k-*rank*) of a matrix $M \in \mathbb{R}^{m\times n}$ is the largest integer k such that every subset of k columns is linearly independent.*

Definition 3.2 (mode-i slices)**.** *We define the mode-i slices of a m-way tensor T as the $(m-1)$-way tensors obtained from T by fixing the i-th index.*

Definition 3.3 (mode-i product)**.** *The mode-i product of a tensor $T \in \mathbb{R}^{n_1\times\cdots\times n_m}$ with a matrix $A \in \mathbb{R}^{J\times n_i}$, denoted by $T \times_i A$, is a tensor of dimensions $n_1 \times \cdots \times n_{i-1} \times J \times n_{i+1} \times \cdots \times n_m$. For all $j \in J$, the j-th slice of $T \times_i A$ is a linear combination of the slices of T along the same mode:*

$$(T \times_i A)_{:,\ldots,:,j,:,\ldots,:} = \sum_{k=1}^{n_i} A_{jk} T_{:,\ldots,:,k,:,\ldots,:},$$

Note that if T has decomposition

$$T = \sum_{j=1}^{r} v_j^{(1)} \otimes v_j^{(2)} \otimes \cdots \otimes v_j^{(m)}$$

then

$$T \times_i A = \sum_{j=1}^{r} v_j^{(1)} \otimes v_j^{(2)} \otimes \cdots \otimes A v_j^{(i)} \otimes \cdots \otimes v_j^{(m)}.$$

Definition 3.4 (mode-V contractions)**.** *Let $T \in \mathbb{R}^{n_u \times n_v \times n_w}$ be a 3-way tensor with decomposition $T = [\![U, V, W]\!]$. Define for all $z \in \mathbb{R}^{n_v - 1}$,*

$$M(z) = UD([z, 1]V)W^T = T \times_v [z, 1].$$

In other words, $M(z) \in \mathbb{R}^{n_u \times n_w}$ is the mode-v product of T with $[z, 1] \in \mathbb{R}^{n_v}$.

Definition 3.5 (Khatri-Rao product)**.** *Let matrices $A = [a_1, a_2, ..., a_r] \in \mathbb{R}^{n_a \times r}$ and $B = [b_1, b_2, ..., b_r] \in \mathbb{R}^{n_b \times r}$. The Khatri-Rao product $A \odot B$ is the column-wise Kronecker product:*

$$A \odot B = [a_1 \otimes b_1, a_2 \otimes b_2, ..., a_r \otimes b_r] \in \mathbb{R}^{(n_a n_b) \times r}.$$

Note that the rows of $A \odot B$ are the Hadamard products $a_i^* \odot b_j^* \in \mathbb{R}^r$, for all $i \in [n_a]$ and $j \in [n_b]$ (the Hadamard product is the element-wise product of two vectors of the same length).

Definition 3.6 (d-reshaping)**.** *Let $T \in \mathbb{R}^{n_1 \times \cdots \times n_m}$ be an m-way tensor and let $\mathcal{S} = \{S_1, \ldots, S_d\}$ be a partition of the index set $[m]$. The d-reshaping of T with respect to $\mathcal{S}$ is a d-way tensor $T_{\mathcal{S}} \in \mathbb{R}^{N_1 \times \cdots \times N_d}$, where the dimension of the i-th mode is given by $N_i = \prod_{j \in S_i} n_j$.*

If T has a CP decomposition $T = \sum_{j=1}^{r} v_j^{(1)} \otimes \cdots \otimes v_j^{(m)}$, then $T_{\mathcal{S}}$ admits a d-way decomposition:

$$T_{\mathcal{S}} = \sum_{j=1}^{r} u_j^{(1)} \otimes u_j^{(2)} \otimes \cdots \otimes u_j^{(d)}, \tag{4}$$

where each factor vector $u_j^{(i)}$ is the vectorization of the Kronecker product of the original factor vectors in the i-th partition set:

$$u_j^{(i)} = vec\left(\bigotimes_{k \in S_i} v_j^{(k)} \right). \tag{5}$$

Equivalently, the component $U^{(i)} = [u_1^{(i)}, \ldots, u_r^{(i)}]$ is the Khatri-Rao product of the matrices $\{V^{(k)}\}_{k \in S_i}$ following the order of indices in S_i.

Definition 3.7. *(mode-i unfolding) The mode-i unfolding of a tensor $T \in \mathbb{R}^{n_1 \times \cdots \times n_m}$, denoted $T_{(i)}$, is the 2-reshaping associated with the partition of modes $S_1 = \{i\}$ and $S_2 = [m] \setminus \{i\}$. This results in a matrix $T_{(i)} \in \mathbb{R}^{n_i \times \prod_{j \neq i} n_j}$, where the i-th mode is mapped to the rows and all other modes are flattened into the columns.*

Lemma 3.8 (Jennrich)**.** *Let $T = [\![U, V, W]\!]$ be a 3-way tensor of rank r, with components $U \in \mathbb{R}^{n_u \times r}$, $V \in \mathbb{R}^{n_v \times r}$, and $W \in \mathbb{R}^{n_w \times r}$. Suppose that:*

1. *U and W have full column rank r.*
2. *Every pair of columns of V is linearly independent (i.e., $k_v \geq 2$).*

Then the rank decomposition of T is unique. Furthermore, there exists an algorithm to recover the components in time $O(n_u n_v n_w + r^3)$.

Lemma 3.9 (Making a Tensor Concise)**.** *[BSV21, Lemma 5.1] Let $T \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_m}$ be an m-th order tensor of rank r. There exists a randomized algorithm with runtime $\mathrm{poly}(\prod n_i, r)$ that outputs a compressed tensor $T' \in \mathbb{R}^{n_1' \times n_2' \times \cdots \times n_m'}$ and transformation matrices $A_i \in \mathbb{R}^{n_i' \times n_i}$ of full row rank such that, with probability 1:*

1. *$T' = (A_1 \otimes A_2 \otimes \cdots \otimes A_m) \cdot T$.*
2. *The dimensions of T' satisfy $n_i' = \mathrm{rank}(T_{(i)}) \leq r$ for each mode $i \in \{1, \ldots, m\}$, where $T_{(i)}$ denotes the mode-i unfolding of T.*
3. *The rank r and the Kruskal ranks $\{k_i\}_{i=1}^m$ are preserved; consequently, T satisfies the Kruskal condition if and only if T' does.*

The factor matrices $\{U_i\}_{i=1}^m$ of the original tensor T are recovered from the factor matrices $\{U_i'\}_{i=1}^m$ of T' via an efficiently computable linear mapping in poly$(\prod n_i, r)$ time.

Theorem 3.10 (Lakshman–Lazard [LL91])**.** *Let $I \subset K[x_1, \ldots, x_n]$ be a zero-dimensional ideal over a field K of characteristic zero, with generators of degree at most d. Under any admissible monomial ordering, the Gröbner basis of radical of I can be computed in $d^{O(n)}$ time, with probability at least $1 - 1/2^{d^n}$. Moreover, zero-dimensionality of I can be decided at the same cost and with the same probability of success.*

Lemma 3.11 (Shape Lemma [CLO98, Ch. 2, Exercise 16])**.** *Let $I \subset K[x_1, \ldots, x_n]$ be a zero-dimensional radical ideal such that the x_n-coordinates of the points in $V(I)$ are distinct. Let G be the reduced Gröbner basis of I with respect to a lex monomial order with x_n as the last variable. If $|V(I)| = m$, then G consists of n polynomials*

$$g_1 = x_1 + h_1(x_n), \quad \ldots \quad , g_{n-1} = x_{n-1} + h_{n-1}(x_n), \quad g_n = x_n^m + h_n(x_n),$$

where $h_1, \ldots, h_n$ are polynomials in x_n of degree at most $m-1$. In particular, all m points of $V(I)$ are recovered by finding the roots of g_n and back-substituting into $g_1, \ldots, g_{n-1}$.

4 The decomposition algorithm

In this section, we formally present the decomposition algorithm for 3-way tensors. For tensors with $m > 3$ modes, we first reshape the tensor to a 3-way tensor, apply our decomposition algorithm, and finally recover the components of the original m-way tensor. We defer the discussion of this process to Section 5.

Let T be a 3-way tensor with an r-decomposition $[\![U, V, W]\!]$, and corresponding Kruskal ranks k_u, k_v, k_w. Recall that

$$k := 2r + 2 - \max\{k_u + k_v, k_u + k_w, k_v + k_w\}. \tag{6}$$

Under the Kruskal condition, k is independent of the specific decomposition (see Lemma 4.7).

Because k and its associated mode are unknown a priori, the full decomposition procedure (Algorithm 1) iteratively tests candidate values $k' \geq 2$ and modes $i \in \{u, v, w\}$ until it reaches k (and its corresponding mode).[7] We restate a single round of this process as Algorithm DECOMPOSE,

[7] To optimize performance, the search over k' is the outermost loop, as time complexity is exponential in k.

which takes parameters (T, k', i). We show that Algorithm DECOMPOSE(T, k, i) returns the unique decomposition with high probability, provided that T satisfies the Kruskal condition and $k_i = \min\{k_u, k_v, k_w\}$.

Here, we will assume that our tensor is concise: A concise tensor is an m-th order tensor where each mode's dimension n_i is equal to the rank of its mode-i flattening. Consequently, $n_i \leq r$ for all $i \in \{1, \ldots, m\}$. Due to Lemma 3.9, this 'compression' and the consequent 'decompression' can be performed efficiently, so we may assume WLOG that the input tensor is given in concise form.

Algorithm DECOMPOSE single iteration of Algorithm 1

Input: (T, k', i), where T is concise. Denote $d' = k' - 1$.

1. Apply a random matrix $A \in \mathbb{R}^{k' \times n_i}$ on the i-th mode of T to obtain $T' = T \times_i A$.
2. For variables $z = (z_1, ..., z_{d'})$ form the matrix
$$M(z) := T' \times_i [z, 1].$$
3. Run MinRank(M, k').
   (a) if the algorithm fails, continue to the next iteration of Algorithm 1. Otherwise,
   (b) denote $m :=$ MinRank(M, k') and
$$\mathcal{S} = \{z \in \mathbb{R}^{d'} \mid \operatorname{rank} M(z) = m\}$$
   (c) Set $r' = m + d'$.
4. Apply HP-reconstruction$(\mathcal{S}, r', k')$ to recover the i-th component of T'.
   (a) If the algorithm fails, continue to the next iteration of Algorithm 1. Otherwise,
   (b) denote the output Q.
5. To recover the two other components U' and W', apply Algorithm 5 on input (T', Q).
6. Solve the system obtained by the v-mode reshaping
$$T_{\{v\},\{u,w\}} = V'(U' \odot W').$$
7. Verify that $T = [\![U', V', W']\!]$ (if not, continue to the next iteration).

Output: (U', V', W').

The algorithm follows the exact procedure described in the proof overview (Section 2). We begin by elaborating on each step, followed by the proof of correctness of DECOMPOSE in Theorem 4.9.

As discussed in the proof overview (Section 2), a key initial step to recover the factor matrices is to ensure that the hyperplanes perpendicular to the columns of the projected matrix V lie in *general position*. In Step 1 of the algorithm, a projection matrix $A \in \mathbb{R}^{k \times n_v}$ is drawn from an absolutely continuous distribution (e.g., Gaussian) and applied to the v-mode. The resulting tensor $\mathcal{T}' \in \mathbb{R}^{n_u \times k \times n_w}$ possesses an r-rank decomposition $[\![U, Q, W]\!]$, where $Q := AV \in \mathbb{R}^{k \times r}$ represents

the projected factor matrix.

This ensures that Q inherits the necessary structural properties for recovery, as formalized in the following lemma:

Lemma 4.1 (General Position of Projected Hyperplanes)**.** *Let $V \in \mathbb{R}^{n_v \times r}$ be a matrix with Kruskal rank $k_v \geq k$. Let $A \in \mathbb{R}^{k \times n_v}$ be drawn from an absolutely continuous distribution (e.g., Gaussian), and let $Q = AV$. Then, with probability 1:*

1. *Q has full Kruskal rank k.*
2. *The first $d = k - 1$ rows of Q have full Kruskal rank d.*
3. *The set of hyperplanes $\mathcal{H} = \{H_j\}_{j=1}^{r}$ defined by $H_j = \{z \in \mathbb{R}^d \mid \langle [z, 1], q_j \rangle = 0\}$ lies in general position in $\mathbb{R}^d$.*

Proof. The "bad" matrices A that fail to produce a full Kruskal rank Q belong to an algebraic variety defined by the vanishing of specific determinants. Specifically, for any subset $I \subseteq [r]$ with $|I| = k$, the set $\mathcal{A}_I = \{A \mid \det(AV(I)) = 0\}$ is the zero set of a non-trivial polynomial (since V has full Kruskal rank, $V(I)$ has a left inverse).

Because the Lebesgue measure of any such variety is zero, a random A avoids the union $\bigcup \mathcal{A}_I$ with probability 1. The same argument applies to the first d rows of Q. By Lemma A.2, Items 1 and 2 directly translate to the hyperplanes being in general position. This ensures that every d-wise intersection of hyperplanes in $\mathcal{H}$ corresponds to a unique point in $\mathbb{R}^d$, which is the key to our recovery algorithm. □

If A is a "good" projection, then the set $\mathcal{H}$ lies in general position in $\mathbb{R}^d$, meaning that the intersection of any $t \leq d$ hyperplanes forms a $(d - t)$-dimensional flat in $\mathbb{R}^d$ (and the intersection of $t > d$ hyperplanes is empty). In particular, d-wise intersections are points, and every point is unique. There are exactly $\binom{r}{d}$ such points (see Lemma A.4).

4.1 Step 2: finding the d-wise intersection points

In Step 2 of the decomposition algorithm, we list all the points z, for which

$$\operatorname{rank} M(z) \leq r - d.$$

We need to show that:

1. These points are precisely the d-wise intersection points of hyperplanes in $\mathcal{H}$. Namely,

$$\{z : \operatorname{rank} M(z) \leq r - d\} = \bigcup_{I:|I|=d} \bigcap_{i \in I} H_i,$$

   and

2. there are exactly $\binom{r}{d}$ points in this set.

The first fact is proved in the following lemma, while the second fact follows from a standard property of general position; we defer its statement and proof to Appendix Lemma A.2.

Theorem 4.2. *Let $T = [\![U, Q, W]\!]$ be an r-decomposition satisfying the Kruskal condition. Suppose that $Q \in \mathbb{R}^{k\times r}$ has full Kruskal rank k, and let $M(z) = T \times_q [z, 1]$ denote the Q-mode contraction for $z \in \mathbb{R}^d$ where $d =: k - 1$.*

Then,

1. *for all $z \in \mathbb{R}^d$,*
$$\operatorname{rank} M(z) \geq r - d,$$
2. $\operatorname{rank} M(z) = r - d$ *if and only if* $\operatorname{Wt}([z, 1]Q) = r - d$, *and*
3. *The minimizers of* $\operatorname{rank} M$ *are precisely the d-wise intersection points:*
$$\{z : \operatorname{rank} M(z) = r - d\} = \bigcup_{I:|I|=d} \bigcap_{i\in I} H_i.$$

Proof. For all $z \in \mathbb{R}^d$, denote $I_z \subseteq [r]$ the indices where $[z, 1]Q$ is zero. Thus, $[z, 1]Q(I_z) = 0$ for all z. Since Q has Kruskal rank k, the subsystem $Q(I)$ has full rank k whenever $|I| \geq k$, so $|I_z| \leq d = k - 1$ for all z. That is, $[z, 1]Q$ has at most d zeros,

$$\implies \operatorname{Wt}([z, 1]Q) \geq r - d \tag{7}$$

for all z.

For Items 1 and 2 we prove the following:

1. $\operatorname{Wt}([z, 1]Q) = r - d \implies \operatorname{rank} M(z) = r - d$,
2. $\operatorname{Wt}([z, 1]Q) > r - d \implies \operatorname{rank} M(z) > r - d$.

Fix $z \in \mathbb{R}^d$, and denote $t := |I_z|$. Let A, D and B be the subsystems of $U, D([z, 1]Q)$ and W^T (respectively) induced by $[r]\backslash I_z$. We have $A \in \mathbb{R}^{n_u\times(r-t)}$, $D \in \mathbb{R}^{(r-t)\times(r-t)}$, $B \in \mathbb{R}^{(r-t)\times n_w}$, and

$$\operatorname{rank}(M(z)) = \operatorname{rank}(ADB).$$

By Sylvester's rank inequality, for pair of matrices $M_1 \in \mathbb{R}^{n_1\times r}$ and $M_2 \in \mathbb{R}^{r\times n_2}$,

$$\operatorname{rank}(M_1M_2) \geq \operatorname{rank} M_1 + \operatorname{rank} M_2 - r.$$

Therefore,

$$\begin{gathered}\operatorname{rank}(ADB) \geq \\ \operatorname{rank}(A) + \operatorname{rank}(DB) - (r - t) = \\ \operatorname{rank}(A) + \operatorname{rank}(B) - (r - t) \geq \\ \min\{r - t, k_u\} + \min\{r - t, k_w\} - (r - t) \\ \implies \operatorname{rank}(M(z)) \geq \min\{r - t, k_u\} + \min\{r - t, k_w\} - (r - t).\end{gathered} \tag{8}$$

1. Suppose that $t = d$.[8] The Kruskal condition implies that $r - d < k_u, k_w$.[9]

   Equation 8 then implies that

   $$\operatorname{rank}(M(z)) \geq r - d.$$

   On the other hand,

   $$\operatorname{rank}(M) = \operatorname{rank}(ADB) \leq \min\{n_u, r - d, n_w\} = r - d.$$

   We have shown that

   $$\mathrm{Wt}([z, 1]Q) = r - d \implies \operatorname{rank}(M(z)) = r - d.$$

2. Let $t < d$. By symmetry, we may assume without loss of generality that $k_u \geq k_w$. The proof proceeds in two cases:

   Case 1: $r - t \leq k_u$

   Equation 8 implies that

   $$\operatorname{rank}(M) \geq \min\{r - t, k_w\} > r - d.$$

   Case 2: $r - t > k_u$

   $$\operatorname{rank}(M) \geq k_u + k_w - (r - t) \geq (2r + 2 - k) - (r - t) = (r - d) + (t + 1) > r - d.$$

   In both cases, $\operatorname{rank}(M) > r - d$.

This concludes the proof of Items 1 and 2.
Finally, we prove Item 3:

$$\{z : \operatorname{rank} M(z) \leq r - d\} = \bigcup_{I:|I|=d} \bigcap_{i \in I} H_i.$$

Recall the definition of the hyperplanes,

$$H_i = \{z \in \mathbb{R}^d | \ \langle [z, 1], q_i \rangle = 0\}, \quad \forall i \in [r].$$

For every z it holds that $z \in \bigcap_{i \in I_z} H_i$.

- If $\operatorname{rank} M(z) \leq r - d$, then by Items 1 and 2, $\mathrm{Wt}([z, 1]Q) = r - d$

  $$\implies |I_z| = d$$

  $$\implies z \in \bigcup_{I:|I|=d} \bigcap_{i \in I} H_i.$$

[8] Recall that $t := |I_z|$ is the number of zeros in $[z, 1]V$.
[9] If $k_u \leq r - d = r - k + 1$ then $k_u + k + k_w \leq 2r + 1$ which violates the Kruskal condition.

- Conversely, if $z \in \bigcup_{I:|I|=d} \bigcap_{i\in I} H_i$ then there exists I of size d s.t. $z \in \bigcap_{i\in I} H_i$, so $|I_z| \geq d$. By Equation 7, $|I_z| = d$. We apply again Item 2 and obtain

$$\operatorname{rank} M(z) = r - d.$$

□

Algorithm MinRank Find minimizers of $\operatorname{rank} M(z)$

Input: $(M(z), k')$ where

- $M(z) = UD([z, 1]V)W^T$ (this is an affine linear combination of matrices with coefficients $\{z_1, ..., z_{k'-1}, 1\}$),
- $U \in \mathbb{R}^{n_u \times r}, V \in \mathbb{R}^{k' \times r}$, $W \in \mathbb{R}^{n_w \times r}$, and
- $n_u, n_w \leq r$.

Denote $n = \min\{n_u, n_w\}$.
Termination Condition: If at any point the total number of operations performed in the steps below exceeds $n^{ck'}$, for a large fixed determinable constant c, the algorithm aborts and outputs **fail**.

1. For $t = n, n-1, \ldots, 1$:
   (a) Generate the system $\mathcal{E}_t$ by setting all t-minors of $M(z)$ to zero.
   (b) If $\mathcal{E}_t$ is zero-dimensional, compute the lexicographical Gröbner basis of the radical of $\mathcal{E}_t$ using Theorem 3.10. Then, determine whether a solution exists by applying Lemma 3.11.
   (c) If the solution set is empty:
      i. Find $\mathcal{S} = \mathbb{V}(\mathcal{E}_{t+1})$, the solution set of the previous system.
      ii. Output $(t, \mathcal{S})$.
2. Output **fail**.

Before proving the correctness of Algorithm MinRank, we emphasize that Theorem 3.10 determines dimensionality over $\mathbb{C}$. It is possible in principle, that $\mathcal{E}_t$ has only solutions in $\mathbb{C}\backslash\mathbb{R}$. In that case, MinRank may output a complex set of minimizers. However, by Theorem 4.2

$$\text{MinRank}(M(z), k) = r - d,$$

and all the minimizers are solutions of a real linear system, and are therefore real. Thus, when Algorithm MinRank is called with $k' = k$, the solution set is real.

Lemma 4.3. *Let $d := k - 1$ as in Theorem 4.2, with k as defined by Equation 6, and suppose that $r > 9$. If V has full Kruskal rank k', then with probability at least $1 - 2^{-r}$,*

1. *if $k' < k$, then MinRank outputs $\textsc{MinRank}(M(z), k')$ and a list of all the minimizers, or **fail**, and*

2. if $k' = k$, then MinRank outputs $\textsc{MinRank}(M(z), k)$ and a list of all the minimizers.

Either way, the time complexity is at most $n^{ck'} = r^{O(d)}$.

Proof. **Failure probability:** The randomness, and thus failure probability of the algorithm is dominated by the applications of Theorem 3.10. Given the polynomial system $\mathcal{E}_t$ defined in Step 1a, when $k' > 2$, we apply Theorem 3.10 to determine the dimensionality of the system, and in the zero-dimensional case, compute the Gröbner basis in time $t^{O(k')} \leq r^{O(k')}$ with failure probability at most $2^{-t^{(k'-1)}} \leq 2^{-t^2}$. Otherwise, when $k' \leq 2$, the system $\mathcal{E}_t$ has at most one variable and can be solved deterministically in $\text{poly}(r)$ time with no failure probability, via standard univariate polynomial root-finding. Thus, iteration t has failure probability at most 2^{-t^2}. There are $n - \text{MinRank}(M(z), k') + 1$ iterations in the case of success at $t = \text{MinRank}(M(z), k')$, or otherwise, a failure occurs at $t = \text{MinRank}(M(z), k')$ with probability at most 2^{-t^2}. Since $\text{MinRank}(M(z), k')$ is monotone decreasing in k',[10] and $\text{MinRank}(M(z), k) = r - d$ (see Theorem 4.2),

$$\textsc{MinRank}(M(z), k') \geq r - d, \tag{9}$$

for all $k' \leq k$. We conclude that the probability that Step 1b fails for any t in the range n down to $\textsc{MinRank}(M(z), k')$ is then at most

$$\sum_{t=r-d}^{n} 2^{-t^2} \leq (n - r + d + 1)2^{-(r-d)^2} < 2^{-r}.$$

The second inequality holds for all $r > 9$ because $(n - r + d + 1)2^{-(r-d)^2}$ is monotone increasing in d and the maximum is attained at $d = \frac{2r-1}{3}$ (this upper bound on d follows from Equation 6).

Correctness: Assuming that no failures occur (i.e., Theorem 3.10 holds, the termination condition is not met, and step 2 is not reached), we show that Algorithm MinRank outputs $(t, \mathcal{S})$ where $t = \textsc{MinRank}(M(z), k')$ and $\mathcal{S}$ is the set of all minimizers of $\operatorname{rank} M(z)$. By definition, for any $z \in \mathbb{C}^{k'-1}$, $\operatorname{rank} M(z) \leq t \iff z$ is a solution to $\mathcal{E}_{t+1}$. The algorithm halts upon finding the first t for which $\mathcal{E}_t$ has no solution. That is, $\mathcal{E}_{t+1}$ has a solution z so $\textsc{MinRank}(M, k') \leq \operatorname{rank} M(z) \leq t$, and $\mathcal{E}_t$ has no solutions, so for every z, $\operatorname{rank} M(z) \geq t$. Therefore $\textsc{MinRank}(M, k') = t$, and the solutions to $\mathcal{E}_{t+1}$ are exactly the minimizers of $\operatorname{rank} M(z)$.[11] If $k' < k$ the algorithm either outputs $\textsc{MinRank}(M, k')$ or **fail** if the termination condition is met. For the case $k' = k$, we show that the termination condition allows for the computation of $\text{MinRank}(M(z), k)$ and the complete solution set, as detailed in the time complexity analysis.

Time complexity: In Step 1a we produce the input for Theorem 3.10. Namely, we calculate the t-minors from the matrix $M(z)$. The number of minors is at most

$$\binom{n_u}{t}\binom{n_w}{t} \leq \binom{r}{t}^2 \leq \binom{r}{r-d}^2 = \binom{r}{d}^2 = r^{O(d)}.$$

The second inequality assumes that $d < \frac{r}{2}$. Otherwise, the maximal number of minors is upper bounded by $\binom{r}{r/2}^2$ which in this case is also $r^{O(d)}$. For each minor, we calculate the coefficients via

[10] As k' grows, we are essentially increasing the search space.

[11] If $t = n$, $\mathbb{V}(\mathcal{E}_{n+1}) := \mathbb{C}^{k'-1}$.

interpolation. The minors have d variables and degree t so there are at most $\binom{t+d}{d} = r^{O(d)}$ monomials. Evaluation costs poly(r) per point. So the overall cost for calculating the dense representation is $r^{O(d)}$.

Step 1b invokes Theorem 3.10 and Lemma 3.11 to determine whether a solution exists. At the last iteration, when no solution exists, Step 1c produces a Gröbner basis (also via Theorem 3.10 and Lemma 3.11). The cost of either of these calls is $t^{O(k)} \leq r^{O(d)}$.

Given a Gröbner basis, calculating the solutions (all z such that rank $M(z) = \textsc{MinRank}(M(z), k)$) can be completed in poly(number of solutions) time. For the correct choice of input ($k' = k$ and V has full Kruskal rank), the algorithm terminates in poly(number of solutions) $= r^{O(d)}$ time;[12] otherwise, by the design of the algorithm, it is guaranteed to terminate within $r^{O(d)}$ steps anyway. □

4.2 Step 3: Reconstructing $\mathcal{H}$

Suppose that Step 2 of the decomposition algorithm was successfully executed. Then we have obtained the set of points

$$\mathcal{S} = \{z \in \mathbb{R}^d | \text{ rank } M(z) \leq r - d\}.$$

By Theorem 4.23 this is also the set of d-wise intersection points of hyperplanes in $\mathcal{H}$, i.e.,

$$\mathcal{S} = \bigcup_{I:|I|=d} \bigcap_{i \in I} H_i.$$

Our goal at Step 4 is to reconstruct $\mathcal{H}$. We use the following criterion for identifying hyperplanes in $\mathcal{H}$, the proof of which is deferred to the Appendix: Lemma A.4.

Lemma 4.4 (Hyperplane Identification Criterion)**.** *Let $\mathcal{H}$ be a set of r hyperplanes in $\mathbb{R}^d$ in general position, and let $\mathcal{S}$ be the set of all d-wise intersections of hyperplanes in $\mathcal{H}$. If H is a $(d-1)$-dimensional hyperplane such that*

$$|H \cap \mathcal{S}| > \frac{r}{d}\binom{r-2}{d-2},$$

then $H \in \mathcal{H}$.

Thus, to reconstruct $\mathcal{H}$ we can iterate through all the d-size subsets of $\mathcal{S}$. For each subset, we calculate its affine rank. If it is $d-1$, we find the normal to the hyperplane through the points. Then we can count the number of intersection points on the hyperplane to determine whether or not the hyperplane belongs to $\mathcal{H}$. However, there are $\binom{|\mathcal{S}|}{d} = r^{O(d^2)}$ subsets to explore, which dominate any other step of the decomposition algorithm, so we again apply randomness to reduce time complexity to $r^{O(d)}$. We describe our algorithm for faster sampling of the points to recover these hyperplanes, followed by an analysis of its time complexity and success probability.

[12] By Corollary A.3, the system has exactly $\binom{r}{d}$ solutions.

Algorithm HP-reconstruction Reconstruct $\mathcal{H}$ given $\mathcal{S}$, the set of d-wise intersection points

Input: $(\mathcal{S}, r, t)$
Initialize $\mathcal{H} = \emptyset$.
Repeat r^{2d+t} times:

1. Choose d points $D \subset \mathcal{S}$ uniformly at random.
2. If D has affine rank $d-1$, find the unique hyperplane H through D (up to scaling).
3. If $|H \cap \mathcal{S}| > \frac{r}{d}\binom{r-2}{d-2}$, and $H \notin \mathcal{H}$, add H to $\mathcal{H}$.

If $|\mathcal{H}| \neq r$, output fail. Otherwise, output $\mathcal{H}$.

Lemma 4.5. *Let $\mathcal{H} = \{H_1, ..., H_r\}$ be a set of hyperplanes in general position in $\mathbb{R}^d$,[13] and let $\mathcal{S}$ be the set of d-wise intersection points of hyperplanes in $\mathcal{H}$, then when invoked with parameters $(\mathcal{S}, r, t)$, Algorithm HP-reconstruction outputs $\mathcal{H}$ with probability at least $1 - r \cdot 2^{-r^t}$ and runtime $r^{O(d)+t}$. In particular, when $t = 1$ Algorithm HP-reconstruction runs in time $r^{O(d)}$ and has success probability at least $1 - 2^{-r}$.*

Proof. For correctness, we note that a hyperplane satisfies $|H \cap \mathcal{S}| > \frac{r}{d}\binom{r-2}{d-2}$ if and only if $H \in \mathcal{H}$ (see Corollary A.3 and Lemma A.4).

Now we analyze the success probability. By Lemma A.6, d points chosen uniformly at random from $\mathcal{S}$ span any hyperplane in $\mathcal{H}$ with probability at least $\frac{1}{r^{2d}}$. Thus, if we draw r^{2d+t} sets of size d, then for any $H_\ell \in \mathcal{H}$,

$$\Pr[H_\ell \text{ not recovered}] \leq (1 - \frac{1}{r^{2d}})^{r^{2d+t}} < e^{-r^t} < 2^{-r^t}$$

$$\implies \Pr[\mathcal{H} \text{ not reconstructed}] < r \cdot 2^{-r^t}.$$

We can therefore reconstruct $\mathcal{H}$ with probability at least $1 - r \cdot 2^{-r^t}$.
The time complexity of calculating the affine rank and calculating the normal is poly(d). Counting the number of intersection points on the hyperplane costs $O(d|\mathcal{S}|) = O(dr^d)$ (this involves counting the number of points in $\mathcal{S}$ that are orthogonal to the normal defining H). We can verify that $H \notin \mathcal{H}$ in time $O(dr)$. Therefore, we complete Steps 1-3 in time $r^{O(d)}$. We repeat these steps r^{2d+t} times, which leads to an overall time complexity of $r^{O(d)+t}$. □

4.3 Step 4: restoring U and W given Q

Once Step 3 outputs Q, which is a random projection of V, Step 4 aims to learn U and W using Q. That is, we are given a tensor $T = \sum_{i=1}^r u_i \otimes v_i \otimes q_i \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, one of its components Q, and we want to recover U and V.

We assume that Q has full Kruskal rank $k_q = n_q \leq \min\{k_u, k_v\}$, and that the factor matrices satisfy the Kruskal condition. Our approach is to use Q to eliminate $k_q - 2$ terms from the decomposition while maintaining a Kruskal rank of 2 in the third mode. This reduction transforms

[13] Meaning that the affine dimension of $\bigcap_{i \in I} H_i$ is $\max\{d - |I|, 0\}$ for all $I \subseteq [r]$.

the problem into a Jennrich instance, which can then be solved directly. We formally present the algorithm for this recovery process, followed by its proof of correctness.

Algorithm 5 Decompose T given a component Q.

Input: (T, Q) where

- $T = \sum_{i=1}^{r} u_i \otimes v_i \otimes q_i$.
- $k_q = n_q$.
- **Kruskal's Condition:** $k_u + k_v + k_q \geq 2r + 2$

For any matrix M with r columns, denote M_1 the sub-matrix induced by columns $\{1, ..., k_q - 2\}$, and M_2 the sub-matrix induced by columns $\{k_q - 1, ..., r\}$.

1. If $k_q = 2$ apply Jennrich's Algorithm and output the decomposition.
2. Else, find full rank $P \in \mathbb{R}^{2 \times k_q}$ perpendicular to $Q_1 \in \mathbb{R}^{k_q \times (k_q - 2)}$ (i.e. the rows of P form a basis for the left kernel of Q_1).
3. Apply Jennrich's algorithm to $T \times_q P = \sum_{i=1}^{r} u_i \otimes v_i \otimes Pq_i = \sum_{i=k_q-1}^{r} u_i \otimes v_i \otimes Pq_i$ to obtain a decomposition
$$T \times_q P = [\![\tilde{U}_2, \tilde{V}_2, PQ_2]\!].$$
Matching U, V, Q: Resolve permutation and scaling ambiguities by aligning the output with the ground-truth matrix PQ_2. Apply the resulting permutation and inverse scaling to $\tilde{U}_2$ and $\tilde{V}_2$ to maintain consistency with Q.
4. Compute $T_2 := \sum_{i=k_q-1}^{r} \tilde{u}_i \otimes \tilde{v}_i \otimes q_i = \sum_{i=k_q-1}^{r} u_i \otimes v_i \otimes q_i$ and
$$T_1 := T - T_2 = \sum_{i=1}^{k_q-2} u_i \otimes v_i \otimes q_i.$$
5. For each $\ell \in [k_q - 2]$, find $p_\ell \in \mathbb{R}^{k_q}$ such that $\langle p_\ell, q_j \rangle = \mathbb{1}_{\ell=j}$ for all $j \in [k_q - 2]$ (for example, take p_ℓ to be the ℓ-th row of $Q_1^\dagger$). Decompose the rank-1 matrices
$$T_1 \times_q p_\ell = u_\ell \otimes v_\ell$$
via SVD to find $\tilde{U}_1$ and $\tilde{V}_1$.

Output: $\tilde{U}$ and $\tilde{V}$.

Lemma 4.6. *Let $T = \sum_{i=1}^{r} u_i \otimes v_i \otimes q_i$ be a rank-r tensor. Given the tensor T and the factor matrix $Q = [q_1, \ldots, q_r]$ as input, if $k_q = n_q$ and $k_u + k_v + k_q \geq 2r + 2$, then Algorithm 5 outputs a decomposition of T in time* $\mathrm{poly}(n_u n_v n_q)$. *Furthermore, under these conditions, the decomposition $\{u_i, v_i, q_i\}_{i=1}^{r}$ is the only possible rank-r representation of T up to permutation and scaling.*

Proof. The algorithm proceeds by reducing the decomposition of T to sub-problems solvable via Jennrich's algorithm or direct projection.

Correctness: If $k_q = 2$, the result follows trivially from the Kruskal condition, which implies $k_u = k_v = r$. In this case, Jennrich's algorithm directly recovers the decomposition (and proves uniqueness).

Otherwise, we consider the transformed tensor $T \times_q P = \sum_{i=k_q-1}^{r} u_i \otimes v_i \otimes Pq_i$, which possesses a decomposition of rank $r' = r - k_q + 2$. The Kruskal condition $k_u + k_v + k_q \geq 2r + 2$ ensures that $k_u, k_v \geq r'$, implying that the factor matrices U_2 and V_2 have full column rank. Furthermore, the Kruskal rank of PQ_2 is at least 2; if it were not, a linear combination of its columns would lie in the kernel of P, contradicting the linear independence of $\{q_1, \ldots, q_{k_q}\}$. Consequently, $T \times_q P$ satisfies the requirements for Jennrich's algorithm, which will successfully recover a decomposition equal to $[\![U_2, V_2, PQ_2]\!]$ up to permutation and scaling.

In Step 3, we utilize the ground-truth matrix PQ_2 to perform the Matching U, V, W procedure. Since PQ_2 has distinct columns, this matching step ensures we resolve all permutation ambiguities and determine the specific scaling factors required to exactly recover $u_j \otimes v_j$ for all $j \in \{k_q - 1, \ldots, r\}$.

With T_2 recovered, we define $T_1 = T - T_2 = \sum_{i=1}^{k_q-2} u_i \otimes v_i \otimes q_i$. Since we have access to the remaining components of Q, we can directly access each $u_\ell \otimes v_\ell$ by projecting T_1 onto the dual basis vectors p_ℓ. These vectors p_ℓ are linearly independent by design, forming a dual basis for the first $k_q - 2$ columns of Q, ensuring that each projection uniquely isolates a single rank-1 component without interference. Due to the full column rank condition of the first $k_q - 2$ columns of Q, this projection uniquely isolates each rank-1 component, which is then decomposed via SVD.

Complexity: The time complexity is dominated by Jennrich's algorithm and the SVD computations, both of which are polynomial in the dimensions n_u, n_v, n_q. Specifically, the matching and projection steps involve standard linear algebraic operations within $\text{poly}(n_u n_v n_q)$ time. □

4.4 Analysis of DECOMPOSE

As mentioned in Section 2 (Proof Overview), the correctness of DECOMPOSE follows from these 3 facts:

1. Lemma 4.7: If an r-decomposition satisfies the Kruskal condition, then the tensor has rank r and every rank-decomposition has the same k.
2. Lemma 4.8: At iterations $k' \leq k$, the decomposition algorithm does not output a 'redundant' decomposition. i.e. an r'-decompositions with $r' > r$.
3. Theorem 4.9: At iteration k, the algorithm outputs a (unique) r-decomposition (with high probability).

We now prove these 3 statements by order.

Lemma 4.7 (Uniqueness of 'good' parameters)**.** *Let T be a tensor with r-decomposition $[\![V^{(1)}, V^{(2)}, V^{(3)}]\!]$ and Kruskal ranks (k_1, k_2, k_3). If (k_1, k_2, k_3) satisfies the Kruskal condition then:*

1. $\operatorname{rank} T = r$*, and*
2. *all rank decompositions have Kruskal ranks (k_1, k_2, k_3) (and therefore satisfy the Kruskal condition).*

Proof. Let $[\![U^{(1)}, U^{(2)}, U^{(3)}]\!]$ be an r'-decomposition. To prove that $\operatorname{rank} T = r$ we show that $r' \geq r$. For any $\ell \in [3]$ we apply a random projection $A \in \mathbb{R}^{k_\ell \times n_\ell}$ on the ℓ-mode. We obtain the tensor

$$T' = T \times_\ell A$$

with r-decomposition $[\![V^{(i)}, V^{(j)}, AV^{(\ell)}]\!]$. With probability 1, $AV^{(\ell)}$ has Kruskal rank k_ℓ (see Lemma 4.1), so the decomposition $(V^{(i)}, V^{(j)}, AV^{(\ell)})$ satisfies the Kruskal condition. Denote $d = k_\ell - 1$, and let $M(\cdot)$ be the ℓ-mode contractions of T' (see Definition 3.4). By Theorem 4.2(1),

$$\operatorname{rank}(M(z)) \geq r - d, \tag{10}$$

for all $z \in \mathbb{R}^d$.

On the other hand, $AU^{(\ell)} \in \mathbb{R}^{k_\ell \times r'}$ so for any $I \subseteq [r']$ of size d, the left kernel of $AU^{(\ell)}(I)$ is non-trivial and there exists z_0 s.t. $\operatorname{Wt}([z_0, 1]AU^{(\ell)}) \leq r' - d$.[14] Therefore,

$$\operatorname{rank}(M(z_0)) \leq r' - d. \tag{11}$$

Putting together Equations (10) and (11) we obtain

$$r \leq r'.$$

We proceed to prove Item 2. Suppose that $(U^{(1)}, U^{(2)}, U^{(3)})$ is a rank decomposition with Kruskal ranks k'_1, k'_2 and k'_3. Our goal is to prove that $k'_i = k_i$ for all $i \in [3]$. Assume towards contradiction that $k'_\ell < k_\ell$ for some $\ell \in [3]$. The Kruskal rank of $AU^{(\ell)} \in \mathbb{R}^{k_\ell \times r}$ is k'_ℓ at most. Since $k'_\ell < k_\ell$, there exists $I \subseteq [r]$ of size k_ℓ s.t. $AU^{(\ell)}(I)$ has a non-trivial left kernel. Suppose that $[z_0, 1]$ is in this kernel, then $\operatorname{Wt}([z_0, 1]AU^{(\ell)}) \leq r - k_\ell < r - d$. Therefore,

$$\operatorname{rank}(M(z_0)) < r - d,$$

which contradicts Equation (10).

We conclude that $k'_i \geq k_i$ for all $i \in [3]$. Therefore, $(U^{(1)}, U^{(2)}, U^{(3)})$ satisfies the Kruskal inequality. Now we apply the same logic, flipping the roles of the two decompositions and obtain the inequalities $k'_i \leq k_i$ for all $i \in [3]$.

We conclude that $k'_i = k_i$ for all $i \in [3]$. □

Lemma 4.8. *If $k' \leq k$, then the candidate rank r' computed by the algorithm at Step 3c satisfies $r' \leq r$, where r is the true rank of the tensor.*

Proof. By the design of the algorithm, we set r' such that $r' - d' = m$, where $m = \textsc{MinRank}(M(z), k')$ is the minimum rank achievable for the given k'-dimensional projection.

We claim that $m \leq r - d'$. Recall that $M(z)$ has the following form:

$$M(z) = UD([z, 1]Q)W^T$$

where $Q \in \mathbb{R}^{k' \times r}$. Thus, for any d' columns $I \subseteq [r]$, the subsystem $Q(I)$ has a left kernel. In particular, there exists z s.t.

$$\operatorname{rank} M(z) \leq \operatorname{Wt}([z, 1])Q \leq r - d'$$

[14] If the last row of $AU^{(\ell)}(I)$ is not in the span of its first $k - 1$ rows, we apply a row permutation so there is a vector of the form $[z, 1]$ in the left kernel. Note that permuting the rows of A has no effect on the Kruskal rank of $AV^{(\ell)}$.

$$\implies r' - d' = m \leq \operatorname{rank} M(z) \leq r - d'.$$

We conclude that $r' \leq r$. □

With the ingredients from the previous sections established, we are now ready to analyze the decomposition algorithm. The full decomposition algorithm includes loops iterating over guesses of k and corresponding mode.

In order to simplify the probability estimates, we assume that $r > 9$. Indeed, for cases where $r \leq 9$, one can simply apply a worst-case tensor decomposition algorithm [BSV21, BS25], which remains functional even without assuming the Kruskal condition.

Theorem 4.9 (Decomposition algorithm)**.** *Let $T \in \mathbb{R}^{n_u \times n_v \times n_w}$ be a 3-way tensor with an r-decomposition $[\![U, V, W]\!]$ with U, V, W over $\mathbb{R}$ of appropriate dimensions. Suppose that the decomposition satisfies the Kruskal condition and $r > 9$. Let k be as in Eq.6. Then, we have the following.*

1. ***Completeness:*** *By iteration k of the outer-loop, the algorithm outputs the components U, V, W (up to scaling and permutation) with probability at least $1 - 2^{-0.1r}$.*
2. ***Soundness:*** *With probability at least $1 - 2^{-0.1r}$, the algorithm does not output a redundant decomposition (r'-decomposition with $r' > r$) at any iteration $k' \in \{2, ..., k\}$ of the outer-loop.*
3. ***Time Complexity:*** *The overall time complexity is*

$$\max(\text{poly}(N), (r)^{O(k)})$$

where $N = n_u n_v n_w$.

Proof. **Soundness.** The algorithm utilizes a brute-force search over the projection dimension k'. For each k', we compute $m = \textsc{MinRank}(M, k')$ and set the candidate rank $r' = m + d'$. In the success event, Lemma 4.8 ensures that for any $k' \leq k$, the computed rank r' will never exceed the true rank r. The reconstruction algorithm (HP-reconstruction) is called with the parameter r' and outputs a set of r' hyperplanes. Therefore, the matrix Q obtained by this step has dimensions $k' \times r'$. Needless to say, the algorithm never outputs a deficient decomposition with $r' < r$ due to the verification step (and the established minimality of r by Lemma 4.7).

The failure probability is at most $3k \cdot 2^{-r}$, the additive failure probability across iterations $k' \in \{2, ..., k\}$ and $i \in \{u, v, w\}$ (see Lemma 4.3 for the failure rate of MinRank). The total failure probability is bounded by $2^{-0.1r}$ (for all $r > 3$).

Completeness. For iteration $k' = k$, Lemma 4.1 ensures that the hyperplanes $\mathcal{H}$ perpendicular to the i-th mode columns of T' are in general position. By Theorem 4.2, the d-wise intersection points of these hyperplanes correspond exactly to the rank minimizers of $M(z)$. Thus, the call $\textsc{MinRank}(M, k)$ identifies the minimal rank $r - d$ and the set of minimizers:

$$\mathcal{S} = \bigcup_{I:|I|=d} \bigcap_{i \in I} H_i$$

Given $\mathcal{S}$, Lemma 4.5 ensures that HP-reconstruction$(\mathcal{S}, r, d + 1)$ outputs $\mathcal{H}$. This recovery of $\mathcal{H}$ determines the component Q up to scaling and permutation, which is sufficient for Algorithm 5 to recover the unique factor matrices U and V (see Lemma 4.6).

We calculate the failure probability at iteration k.

1. **MinRank and System Solving:** The MinRank call (which includes generating determinantal equations and solving the system via Lakshman-Lazard) fails with probability at most 2^{-r} across the brute-force search (by Lemma 4.3).

2. **Hyperplane Reconstruction:** Reconstructing the collection $\mathcal{H}$ from the intersection points $\mathcal{S}$ fails with probability at most $r2^{-r}$ as per Lemma 4.5.

The algorithm fails to decompose by iteration k only if at some iteration $k' < k$ the algorithm produced a redundant (yet correct) decomposition, or if at iteration k either Algorithm MinRank or HP-reconstruction failed. The first event happens w.p. at most $3k \cdot 2^{-r}$, and the second w.p. at most $(r+1)2^{-r}$. By the union bound the total failure probability is at most

$$(3k + r + 1)2^{-r} < 2^{-0.1r}$$

(for any $r > 9$).

Time Complexity. The runtime is dominated by the algebraic solving and geometric reconstruction steps:

- **MinRank:** Finding the minimizers $\mathcal{S}$ takes $r^{O(k')}$ time.
- **Reconstruction:** Reconstructing $\mathcal{H}$ from $\mathcal{S}$ is performed in $r^{O(d)}$ time.
- **Decomposition:** Recovering U and W via Algorithm 5 takes $\text{poly}(N)$ time.

Combining these, the total complexity for any candidate k' is $\max(\text{poly}(N), r^{O(k')})$. A successful application is completed within $3k$ iterations, which leads to overall $\max(\text{poly}(N), r^{O(k)})$ time complexity. □

5 Decomposing high dimensional tensors

In the previous section we presented a decomposition algorithm for 3-way tensors. To deal with m-way tensors with $m > 3$, we reshape the tensor and apply the decomposition algorithm on the reshaped tensor. We now introduce some notation to formalize this process. Let $T = [\![V^{(1)}, V^{(2)}, ..., V^{(m)}]\!]$. For all $J \subseteq [m]$ with $|J| \geq 2$ we denote V^J the Khatri-Rao product of $\{V^{(j)}\}_{j \in J}$ (see Definition 3.5), and k_J the Kruskal rank of V^J. For any partition $I \sqcup J \sqcup L = [m]$, the 3-reshaping of T (see Definition 3.6) has components V^I, V^J and V^L. If these matrices satisfy the Kruskal condition, then the decomposition algorithm succeeds (w.h.p.) and outputs V^I, V^J, V^L up to permutation and scaling of the columns. The columns of these matrices are vectorized rank one tensors, so we can directly recover the components of the m-way tensor (via SVD).

Theorem 5.1 (Reshaped Kruskal Theorem)**.** *Let $r \geq 2$, and $m \geq 3$. Consider an m-way tensor $T \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_m}$ with decomposition:*

$$T = \sum_{j=1}^{r} v_j^{(1)} \otimes v_j^{(2)} \otimes \cdots \otimes v_j^{(m)} \tag{12}$$

For each subset $\emptyset \neq I \subseteq [m]$, define V_I as the Khatri-Rao product of matrices $\{V^{(i)}\}_{i \in I}$. If there exists a partition $I \sqcup J \sqcup L = [m]$ and the Kruskal ranks of these partitioned matrices satisfy the Kruskal condition:

$$k_I + k_J + k_L \geq 2r + 2 \tag{13}$$

then,

1. $\operatorname{rank} T = r$, *and*

2. *the decomposition is unique up to scaling and permutation.*

Furthermore, for $r > 9$*, there exists a randomized algorithm that recovers the unique components with probability at least* $1 - 2^{-0.1r}$ *and runtime* $N^{O(k)}$*, where* $k := 2r + 2 - \max\{k_I + k_J, k_I + k_L, k_J + k_L\}$ *and* $N := \prod_{i \in [m]} n_i$ *is the input size.*

Proof. We first prove Item 1. Let $\tilde{T}$ be the 3-way array associated with the partition $I \sqcup J \sqcup L = [m]$. Any r'-decomposition of T can be reshaped into an r'- decomposition of $\tilde{T}$. By Lemma 4.7(1), $\operatorname{rank} \tilde{T} = r$, so $r' \geq r$.

We proceed to the 2nd item. By Lemma 4.7(2), all rank decompositions of $\tilde{T}$ have the same Kruskal ranks k_I, k_J, k_L. Since the decomposition algorithm works uniquely (up to scaling and permutation), $\tilde{T} = [\![V_I, V_J, V_L]\!]$ is a unique rank decomposition. Finally, the Khatri-Rao product is injective (up to scaling), so

$$T = [\![V_1, V_2, ..., V_m]\!]$$

is a unique rank decomposition.

We proceed to analyze the algorithm. We search for k at the outermost loop, because the time complexity of the decomposition is exponential in k. This allows us to cap the overall complexity. Then we search for the corresponding mode and for good partitions. In worst case, we have $O(k)$ iterations for the k-loop, $O(3^m)$ iterations for the partition-search loop (and 3 iterations for the mode loop).

By Theorem 4.9, the decomposition algorithm has runtime $N^{O(k)}$, which absorbs the complexity of all repetitions (including the partition search loop).

The success probability follows directly from Theorem 4.9. □

If the input m-dimensional tensor satisfies the Kruskal condition, then there is an efficient way to perform the 3-reshaping. As a direct consequence, we obtain the following corollary. We defer the reshaping procedure and its correctness to the Appendix, see Lemma A.1.

Corollary 5.2 (Kruskal's Uniqueness Theorem)**.** *Consider an* m*-way tensor* $T \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_m}$ *with decomposition:*

$$T = \sum_{j=1}^{r} V_j^{(1)} \otimes V_j^{(2)} \otimes \cdots \otimes V_j^{(m)} \tag{14}$$

If the Kruskal condition holds:

$$\sum_{t=1}^{m} k_t \geq 2r + m - 1 \tag{15}$$

then,

1. $\operatorname{rank} T = r$, *and*

2. *the decomposition is unique up to scaling and permutation.*

Furthermore, for $r > 9$*, there exists a randomized algorithm that outputs the unique components with probability at least* $1 - 2^{-0.1r}$ *and runtime* $N^{O(k)}$*, where* $k := 2r + 2 - \max\{k_I + k_J, k_I + k_L, k_J + k_L\}$ *and* $N := \prod_{i \in [m]} n_i$*.*

Again, for cases where $r \leq 9$, one can simply apply a worst-case tensor decomposition algorithm [BSV21, BS25], which will decompose these m-tensors even without the Kruskal condition in $\text{poly}(m, \max n_i)$ time.

References

[BBC+20] M. Bardet, M. Bros, D. Cabarcas, P. Gaborit, R. A. Perlner, D. Smith-Tone, J.-P. Tillich, and J. A. Verbel. Improvements of algebraic attacks for solving the rank decoding and minrank problems. In *ASIACRYPT*, 2020.

[BCMV14] Aditya Bhaskara, Moses Charikar, Ankur Moitra, and Aravindan Vijayaraghavan. Open problem: Tensor decompositions: Algorithms up to the uniqueness threshold? In *Conference on Learning Theory*, pages 1280–1282. PMLR, 2014.

[BS25] Vishwas Bhargava and Devansh Shringi. Faster & deterministic FPT algorithm for worst-case tensor decomposition. In *52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*, pages 28–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025.

[BSV21] Vishwas Bhargava, Shubhangi Saraf, and Ilya Volkovich. Reconstruction algorithms for low-rank tensors and depth-3 multilinear circuits. Technical Report 045, Electronic Colloquium on Computational Complexity (ECCC), 2021. Version with full proofs; subsequently appeared in STOC 2021. URL: `https://eccc.weizmann.ac.il/report/2021/045/`.

[CC70] J.D. Carroll and J.J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35:283–319, 1970. `doi:10.1007/BF02310791`.

[CLO98] David A. Cox, John Little, and Donal O'Shea. *Using Algebraic Geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer, New York, 1998.

[CO12] Luca Chiantini and Giorgio Ottaviani. On generic identifiability of 3-tensors of small rank. *SIAM Journal on Matrix Analysis and Applications*, 33(3):1018–1037, 2012. `doi:10.1137/110829180`.

[COV17] Luca Chiantini, Giorgio Ottaviani, and Nick Vannieuwenhoven. Effective criteria for specific identifiability of tensors and forms. *SIAM Journal on Matrix Analysis and Applications*, 38(2):656–681, 2017.

[Der13] Harm Derksen. Kruskal's uniqueness inequality is sharp. *Linear Algebra and its Applications*, 438(2):708–712, 2013. Tensors and Multilinear Algebra. `doi:10.1016/j.laa.2011.05.041`.

[FSEDS10] J. Faugère, M. Safey El Din, and P. Spaenlehauer. Computing loci of rank defects of linear matrices using Gröbner bases and applications to cryptology. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 257–264, 2010. `doi:10.1145/1837934.1837984`.

[Har70] Richard A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-model factor analysis. In *UCLA working papers in phonetics, 16(1):84*, 1970. URL: https://api.semanticscholar.org/CorpusID:6816804.

[Hås90] Johan Håstad. Tensor rank is NP-complete. *Journal of Algorithms*, 11(4):644–654, 1990. doi:10.1016/0196-6774(90)90014-6.

[HL13] Christopher J Hillar and Lek-Heng Lim. Most tensor problems are NP-hard. *Journal of the ACM (JACM)*, 60(6):1–39, 2013.

[Ier89] Doug Ierardi. Quantifier elimination in the theory of an algebraically-closed field. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 138–147, 1989.

[Kha95] Leonid Khachiyan. On the complexity of approximating extremal determinants in matrices. *Journal of Complexity*, 11(1):138–153, 1995.

[KMW24] Pravesh K Kothari, Ankur Moitra, and Alexander S Wein. Overcomplete tensor decomposition via Koszul-Young flattenings. *arXiv preprint arXiv:2411.14344*, 2024.

[Kru76] J. B. Kruskal. More factors than subjects, tests and treatments: an indeterminacy theorem for canonical decomposition and individual differences scaling. *Psychometrica*, 41(3):281–293, 1976.

[Kru77] Joseph B. Kruskal. Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95–138, 1977. doi:10.1016/0024-3795(77)90069-6.

[Kru89] J. B. Kruskal. Rank, decomposition and uniqueness for 3-way and n-way arrays. In *Multiway data analysis*, pages 7–18. Elsevier Science Publishers B.B. (North-Holland, 1989.

[Lan11] Joseph M Landsberg. *Tensors: geometry and applications: geometry and applications*, volume 128. American Mathematical Soc., 2011.

[LL91] Yagati N Lakshman and Daniel Lazard. On the complexity of zero-dimensional algebraic systems. In *Effective methods in algebraic geometry*, pages 217–225. Springer, 1991.

[LP23] Benjamin Lovitz and Fedor Petrov. A generalization of Kruskal's theorem on tensor decomposition. In *Forum of Mathematics, Sigma*, volume 11, page e27. Cambridge University Press, 2023.

[MSS16] Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 438–446. IEEE, 2016.

[Rho10] John A. Rhodes. A concise proof of Kruskal's theorem on tensor decomposition. *Linear Algebra and its Applications*, 432(7):1818–1824, 2010. doi:10.1016/j.laa.2009.11.033.

[RSG17] S. Rabanser, O. Shchur, and S. Günnemann. Introduction to tensor decompositions and their applications in machine learning, 2017. URL: arXiv:1711.10781.

[SB00] Nicholas D Sidiropoulos and Rasmus Bro. On the uniqueness of multilinear decomposition of n-way arrays. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 14(3):229–239, 2000.

[Spe61] Charles Spearman. "General intelligence" objectively determined and measured. In J. J. Jenkins & D. G. Paterson, editor, *Studies in individual differences: The search for intelligence*, page 59–73. Appleton-Century-Crofts, 1961. Reprint from American Journal of Psychology, 1904, Vol 15[2], 201-293. doi:10.1037/11491-006.

[SS07] Alwin Stegeman and Nicholas D Sidiropoulos. On Kruskal's uniqueness condition for the Candecomp/Parafac decomposition. *Linear Algebra and its applications*, 420(2-3):540–552, 2007.

[SŠ18] Marcus Schaefer and Daniel Štefankovič. The complexity of tensor rank. *Theory of Computing Systems*, 62(5):1161–1174, 2018.

A Reshaping into 3-tensors

Let $T = [\![V_1, V_2, ..., V_m]\!]$ be an m-mode tensor. For the sake of completeness, we prove that if T satisfies the Kruskal condition, then T admits a 3rd-order reshaping (see Definition 3.6) that also satisfies the Kruskal condition. This result is well-established in the literature (see, e.g., [SB00]). For any $J \subseteq [m]$, V^J is the Khatri-Rao product of the components $\{V_j\}_{j\in J}$ (see Definition 3.5).

Lemma A.1. *If T is an m-way tensor satisfying the Kruskal condition, then there exists a partition of $[m]$ into disjoint subsets $J \sqcup K \sqcup L = [m]$ s.t. $[\![V^J, V^K, V^L]\!]$ satisfies the Kruskal condition, namely,*

$$k_J + k_K + k_L \geq 2r + 2.$$

Proof. We use the following facts:

1. For any matrices A, B of dimensions $n_A \times r$ and $n_B \times r$ respectively, if $k_A, k_B \geq 1$ then

   $$k_{A\odot B} \geq \min\{r, k_A + k_B - 1\}$$

   where $A \odot B$ denotes the Khatri-Rao product of A and B (see Lemma 1 in [SB00]).

2. The Kruskal rank is monotone non-decreasing. That is, if $J \subseteq L \subseteq [m]$, then

   $$k_J \leq k_L,$$

   as long as $k_i \geq 1$ for all $i \in L$. This is a direct result of Fact 1.[15]

[15] $$k_L \geq \min\{r, k_J + k_{L\setminus J} - 1\} \geq k_J.$$

We use induction on m. The base case is trivial. Now assume that the claim holds for all $(m-1)$-way tensors satisfying the Kruskal condition, for $m-1 \geq 3$, and suppose that

$$\sum_{i=1}^{m} k_i \geq 2r + m - 1.$$

We analyze 2 cases:

- If for all $i \neq j \in [m]$, $k_i + k_j \geq r+2$ then[16]

  1. $k_1 + k_2 \geq r+2$, and
  2. $k_{\{3,4\}} \underset{\text{Fact 1}}{\geq} \min\{r, k_3 + k_4 - 1\} \geq \min\{r, (r+2)-1\} = r$,

  so

  $$k_1 + k_2 + k_{\{3,4,\ldots,m\}} \underset{\text{Fact 2}}{\geq} k_1 + k_2 + k_{\{3,4\}} \geq 2r+2.$$

- Otherwise, there exist $i \neq j \in [m]$ for which $k_i + k_j \leq r+1$. Assume WLOG that $k_1 + k_2 \leq r+1$.

  $$k_{\{1,2\}} \underset{\text{Fact 1}}{\geq} \min\{r, k_1 + k_2 - 1\} = k_1 + k_2 - 1.$$

  $$\implies k_{\{1,2\}} + \sum_{i=3}^{m} k_i \geq \sum_{i=1}^{m} k_i - 1 \geq 2r + m - 2.$$

  By the hypothesis, there exists a partition $J \sqcup K \sqcup L = \{\{1,2\}, 3, ..., m\}$ s.t.

  $$k_J + k_K + k_L \geq 2r + 2.$$

□

A.1 General position statements

In the first step of the decomposition algorithm (Algorithm DECOMPOSE) we apply a random matrix on one of the components of the tensor we are trying to decompose. We obtain a component $Q \in \mathbb{R}^{k \times r}$ and define, for every column, the orthogonal hyperplane: $H_i := \{z \in \mathbb{R}^d | \ \langle [z,1], q_i^{|} \rangle = 0\}$. We denote the full set of hyperplanes by

$$\mathcal{H} = \{H_1, ..., H_r\}.$$

Lemma A.2. *If $Q \in \mathbb{R}^{k \times r}$ has Kruskal rank k, and $Q([d],[r])$ has Kruskal rank $d = k-1$, then $\mathcal{H}$ is in general position in $\mathbb{R}^d$. That is, for all $I \subseteq [r]$, the affine dimension of $\bigcap_{i \in I} H_i$ is $\max\{d - |I|, 0\}$.*

Proof. For any $I \subseteq [r]$ denote $q_{k_2}^I$, the restriction of the last row of Q to the indices I. Then,

$$\bigcap_{i \in I} H_i = \{z \in \mathbb{R}^d | \ [z,1] Q([k_2], I) = 0\} = \{z \in \mathbb{R}^d | \ z Q([d], I) = -q_{k_2}^I\}.$$

[16] Note that this case together with $m \geq 5$, allows decomposition using Jennrich's algorithm.

- If $|I| \leq d$, the matrix $Q([d], I)$ has rank $|I|$ so $\{z \mid zQ([d], I) = -q^I_{k_2}\}$ is an affine space with dimension $d - |I|$. In particular, when $|I| = d$, $\bigcap_{i \in I} H_i$ has exactly one point.
- If $|I| > d$ then the matrix $Q(I)$ has rank k, so it has no left kernel and $\bigcap_{i \in I} H_i$ is empty.

□

When $\mathcal{H}$ is in general position, we can count the number of d-wise intersection points on the flats defined by $\mathcal{H}$:

Corollary A.3. *If $\mathcal{H}$ is in general position in $\mathbb{R}^d$, then*

$$\mathcal{S} := \{\bigcap_{i \in I} H_i \mid I \subseteq [r], |I| = d\}$$

has size $\binom{r}{d}$. More generally, for any $I \subseteq [r]$, $|I| = t \leq d$,

$$|\bigcap_{i \in I} H_i \cap \mathcal{S}| = \binom{r-t}{d-t}.$$

Proof. Let $I \subseteq [r]$ be a subset of size $t \leq d$. By Lemma A.2, every choice of $d - t$ hyperplanes in $\mathcal{H} \setminus \{H_i\}_{i \in I}$ yields a point in $\mathcal{S}$, so

$$|\bigcap_{i \in I} H_i \cap \mathcal{S}| \leq \binom{r-t}{d-t}$$

On the other hand, any choice of $d-t$ hyperplanes in $\mathcal{H} \setminus \{H_i\}_{i \in I}$ corresponds to a different point, because every two d-wise intersections intersect trivially (by Lemma A.2), so

$$|\bigcap_{i \in I} H_i \cap \mathcal{S}| = \binom{r-t}{d-t}.$$

In particular,

$$|\mathcal{S}| = \binom{r}{d}$$

and

$$|H_i \cap \mathcal{S}| = \binom{r-1}{d-1}$$

for any $i \in [r]$. □

Lemma A.4. *Let $r > d$, $d \geq 2$. We have*

- *$\mathcal{H}$, a set of r hyperplanes of dimension $d-1$ in general position in $\mathbb{R}^d$.*
- *$\mathcal{S}$, the set of d-wise intersections of hyperplanes in $\mathcal{H}$.*

If H is a $(d-1)$-dimensional hyperplane, and

$$|H \cap \mathcal{S}| > \frac{r}{d}\binom{r-2}{d-2},$$

then $H \in \mathcal{H}$.

Proof. We prove the claim using induction on d. When $d = 2$, $\mathcal{S}$ consists of 2-wise intersection points, and the hyperplanes are lines. Let H be a line with

$$|H \cap \mathcal{S}| > \frac{r}{d}\binom{r-2}{d-2} = r/2$$

points from $\mathcal{S}$. Counting with multiplicity, H has at least $r+1$ intersection points in $\mathcal{S}$, so by the Pigeonhole Principle there exists $H_\ell \in \mathcal{H}$ that intersects H in (at least) two $\mathcal{S}$-points and therefore $H = H_\ell \in \mathcal{H}$.

Now assume that the claim holds for dimension $d-1$ ($d \geq 3$) and assume that there exists a hyperplane $H \notin \mathcal{H}$ with $|H \cap \mathcal{S}| > \frac{r}{d}\binom{r-2}{d-2}$. For every $H_i \in \mathcal{H}$ we have $|H \cap H_i \cap \mathcal{S}| \leq \binom{r-2}{d-2}$:

- If $H \cap H_i = H_\ell \cap H_i$ for some other $H_\ell \in \mathcal{H}$, then by Corollary A.3

$$|H \cap H_i \cap \mathcal{S}| = \binom{r-2}{d-2}.$$

- Otherwise, by the hypothesis,[17]

$$|H \cap H_i \cap \mathcal{S}| < \frac{r-1}{d-1}\binom{r-3}{d-3} < \binom{r-2}{d-2}.$$

Therefore,

$$|H \cap \mathcal{S}| = \frac{1}{d}\sum_{i\in[r]} |H \cap H_i \cap \mathcal{S}| \leq \frac{r}{d}\binom{r-2}{d-2}$$

which is a contradiction to $|H \cap \mathcal{S}| > \frac{r}{d}\binom{r-2}{d-2}$.

□

Corollary A.5. *For any hyperplane H,*

$$|H \cap \mathcal{S}| \leq \binom{r-1}{d-1}$$

Proof. If $H \in \mathcal{H}$, then

$$|H \cap \mathcal{S}| = \binom{r-1}{d-1}.$$

If $H \notin \mathcal{H}$, then by Lemma A.4,

$$|H \cap \mathcal{S}| \leq \frac{r}{d}\binom{r-2}{d-2} \leq \binom{r-1}{d-1}$$

□

Lemma A.6. *Let $\mathcal{P} = \{s_1, ..., s_d\}$ be a set of points chosen uniformly at random from $\mathcal{S}$. Then, for any $H_\ell \in \mathcal{H}$, $\mathcal{P}$ spans H_ℓ with probability at least $\frac{1}{r^{2d}}$.*

[17] We apply the hypothesis with "universe" H_i, hyperplanes $\mathcal{H}_i = \{H_i \cap H_\ell\}_{\ell \neq i}$, and intersection points $\mathcal{S}_i = H_i \cap \mathcal{S}$. General position in H_i holds due to Lemma A.2.

Proof. For all $t \in [d]$ denote r_t the affine rank of the points $\{s_1, ..., s_t\}$, and $\mathcal{L}_t$ the r_t-flat through these points. Then, for any $\ell \in [r]$,

$$\Pr[\mathcal{L}_d = H_\ell] =$$

$$\Pr[\mathcal{P} \subseteq H_\ell \wedge r_d = d-1] =$$

$$\Pr[r_d = d-1 | \mathcal{P} \subseteq H_\ell] \Pr[\mathcal{P} \subseteq H_\ell]$$

Denote E the event $\mathcal{P} \subseteq H_\ell$. We want to lower-bound $\Pr[r_d = d-1|E] \Pr[E]$. We have

$$\Pr[E] = (\frac{d}{r})^d > \frac{1}{r^d} \tag{16}$$

so we only need to lower-bound $\Pr[r_d = d-1|E]$. We have

$$\Pr[r_d = d-1|E] = \Pr[\forall t \in [d], r_t = t-1|E] =$$

$$\Pr[r_1 = 0|E] \prod_{t=2}^{d} \Pr[r_t = t-1 | r_{t-1} = t-2, E]$$

By Corollary A.5, a hyperplane inside H_ℓ can have at most $\binom{r-2}{d-2}$ points, so for all $t \in [d]$

$$\Pr[r_t = t-1 | r_{t-1} = t-2, E] = \Pr[s_t \notin \mathcal{L}_{t-1} | r_{t-1} = t-2, E] \geq$$

$$\frac{\binom{r-1}{d-1} - \binom{r-2}{d-2}}{\binom{r-1}{d-1}} = \frac{r-d}{r-1}.$$

Therefore,

$$\Pr[r_d = d-1|E] \geq (\frac{r-d}{r-1})^{d-1} > \frac{1}{r^d}. \tag{17}$$

We conclude that

$$\Pr[\mathcal{L}_d = H_\ell] > \frac{1}{r^{2d}}$$

□

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il