# Orthogonalized ALS: A Theoretically Principled Tensor Decomposition Algorithm for Practical Use

**Vatsal Sharan** [1] **Gregory Valiant** [1]

## Abstract

The popular Alternating Least Squares (ALS) algorithm for tensor decomposition is efficient and easy to implement, but often converges to poor local optima—particularly when the weights of the factors are non-uniform. We propose a modification of the ALS approach that is as efficient as standard ALS, but provably recovers the true factors with random initialization under standard incoherence assumptions on the factors of the tensor. We demonstrate the significant practical superiority of our approach over traditional ALS for a variety of tasks on synthetic data—including tensor factorization on exact, noisy and over-complete tensors, as well as tensor completion—and for computing word embeddings from a third-order word tri-occurrence tensor.

## 1. Introduction

From a theoretical perspective, tensor methods have become an incredibly useful and versatile tool for learning a wide array of popular models, including topic modeling (Anandkumar et al., 2012), mixtures of Gaussians (Ge et al., 2015), community detection (Anandkumar et al., 2014a), learning graphical models with guarantees via the method of moments (Anandkumar et al., 2014b; Chaganty & Liang, 2014) and reinforcement learning (Azizzadenesheli et al., 2016). The key property of tensors that enables these applications is that tensors have a unique decomposition (decomposition here refers to the most commonly used CANDECOMP/PARAFAC or CP decomposition), under mild conditions on the factor matrices (Kruskal, 1977); for example, tensors have a unique decomposition whenever the factor matrices are full rank. As tensor methods naturally model three-way (or higher-order) relationships, it is not too optimistic to hope that their practical utility will only increase, with the rise of multi-modal measurements (e.g. measurements taken by "Internet of Things" devices) and the numerous practical applications involving high order dependencies, such as those encountered in natural language processing or genomic settings. In fact, we are already seeing exciting applications of tensor methods for analysis of high-order spatiotemporal data (Yu & Liu, 2016), health data analysis (Wang et al., 2015a) and bioinformatics (Colombo & Vlassis, 2015). Nevertheless, to truly realize the practical impact that the current theory of tensor methods portends, we require better algorithms for computing decompositions—practically efficient algorithms that are both capable of scaling to large (and possibly sparse) tensors, and are robust to noise and deviations from the idealized "low-rank" assumptions.

As tensor decomposition is NP-Hard in the worst-case (Hillar & Lim, 2013; Håstad, 1990), one cannot hope for algorithms which always produce the correct factorization. Despite this worst-case impossibility, accurate decompositions can be efficiently computed in many practical settings. Early work from the 1970's (Leurgans et al., 1993; Harshman, 1970) established a simple algorithm for computing the tensor decomposition (in the noiseless setting) provided that the factor matrices are full rank. This approach, based on an eigendecomposition, is very sensitive to noise in the tensor (as we also show in our experiments), and does not scale well for large, sparse tensors.

Since this early work, much progress has been made. Nevertheless, many of the tensor decomposition algorithms hitherto proposed and employed have strong provable success guarantees but are computationally expensive (though still polynomial time)—either requiring an expensive initialization phase, being unable to leverage the sparsity of the input tensor, or not being efficiently parallelizable. On the other hand, there are also approaches which are efficient to implement, but which fail to compute an accurate decomposition in many natural settings. The Alternating Least Squares (ALS) algorithm (either with random initialization or more complicated initializations) falls in this latter category and is, by far, the most widely employed decomposition algorithm despite its often poor performance

[1]Stanford University, USA. Correspondence to: Vatsal Sharan <vsharan@stanford.edu>.

*Proceedings of the $34^{th}$ International Conference on Machine Learning*, Sydney, Australia, PMLR 70, 2017. 

and propensity for getting stuck in local optima (which we demonstrate on both synthetic data and real NLP data).

In this paper we propose an alternative decomposition algorithm, "Orthogonalized Alternating Least Squares" (Orth-ALS) which has strong theoretical guarantees, and seems to significantly outperform the most commonly used existing approaches in practice on both real and synthetic data, for a number of tasks related to tensor decomposition. This algorithm is a simple modification of the ALS algorithm to periodically "orthogonalize" the estimates of the factors. Intuitively, this periodic orthogonalization prevents multiple recovered factors from "chasing after" the same true factors, allowing for the avoidance of local optima and more rapid convergence to the true factors.

From the practical side, our algorithm enjoys all the benefits of standard ALS, namely simplicity and computational efficiency/scalability, particularly for very large yet sparse tensors, and noise robustness. Additionally, the speed of convergence and quality of the recovered factors are *substantially* better than standard ALS, even when ALS is initialized using the more expensive SVD initialization. As we show, on synthetic low-rank tensors, our algorithm consistently recovers the true factors, while standard ALS often falters in local optima and fails both in recovering the true factors and in recovering an accurate low-rank approximation to the original tensor. We also applied Orth-ALS to a large 3-tensor of word co-occurrences to compute "word embeddings".[1] The embedding produced by our Orth-ALS algorithm is *significantly* better than that produced by standard ALS, as we quantify via a near 30% better performance of the resulting word embeddings across standard NLP datasets that test the ability of the embeddings to answer basic analogy tasks (i.e. "puppy is to dog as kitten is to __?") and semantic word-similarity tasks. Together, these results support our optimism that with better decomposition algorithms, tensor methods will become an indispensable, widely-used data analysis tool in the near future.

Beyond the practical benefits of Orth-ALS, we also consider its theoretical properties. We show that Orth-ALS provably recovers all factors under *random* initialization for worst-case tensors as long as the tensor satisfies an incoherence property (which translates to the factors of the tensors having small correlation with each other), which is satisfied by random tensors with rank $k = o(d^{0.25})$ where $d$ is the dimension of the tensor. This requirement that $k = o(d^{0.25})$ is significantly worse than the best known provable recovery guarantees for polynomial-time algorithms on random tensors—the recent work Ma et al. (2016) succeeds even in the over-complete setting with $k = o(d^{1.5})$. Nevertheless, our experiments support our belief that this shortcoming is more a property of our analysis than the algorithm itself. Additionally, for many practical settings, particularly natural language tasks, the rank of the recovered tensor is typically significantly sublinear in the dimensionality of the space, and the benefits of an extremely efficient and simple algorithm might outweigh limitations on the required rank for provable recovery.

Finally, as a consequence of our analysis technique for proving convergence of Orth-ALS, we also improve the known guarantees for another popular tensor decomposition algorithm—the tensor power method. We show that the tensor power method with *random* initialization converges to one of the factors with small residual error for rank $k = o(d)$, where $d$ is the dimension. We also show that the convergence rate is *quadratic* in the dimension. Anandkumar et al. (2014c) had previously shown local convergence of the tensor power method with a linear convergence rate (and also showed global convergence via a SVD-based initialization scheme, obtaining the first guarantees for the tensor power method in non-orthogonal settings). Our new results, particularly global convergence from random initialization, provide some deeper insights into the behavior of this popular algorithm.

The rest of the paper is organized as follows—Section 2 states the notation. In Section 3 we discuss related work. Section 4 introduces Orth-ALS, and states the convergence guarantees. We state our convergence results for the tensor power method in Section 4.2. The experimental results, on both synthetic data and the NLP tasks are discussed in Section 5. Proof details have been deferred to the Appendix.

## 2. Notation

We state our algorithm and results for 3rd order tensors, and believe the algorithm and analysis techniques should extend easily to higher dimensions. Given a 3rd order tensor $T \in \mathbb{R}^{d \times d \times d}$ our task is to decompose the tensor into its factor matrices $A, B$ and $C$: $T = \sum_{i \in [k]} w_i A_i \otimes B_i \otimes C_i$, where $A_i$ denotes the $i$th column of a matrix $A$. Here $w_i \in \mathbb{R}, A_i, B_i, C_i \in \mathbb{R}^d$ and $\otimes$ denotes the tensor product: if $a, b, c \in \mathbb{R}^d$ then $a \otimes b \otimes c \in \mathbb{R}^{d \times d \times d}$ and $(a \otimes b \otimes c)_{ijk} = a_i b_j c_k$. We will refer to $w_i$ as the weight of the factor $\{A_i, B_i, C_i\}$. This is also known as CP decomposition. We refer to the dimension of the tensor by $d$ and denote its rank by $k$. We refer to different dimensions of a tensor as the modes of the tensor.

We denote $T_{(n)}$ as the mode $n$ matricization of the tensor, which is the flattening of the tensor along the $n$th direction obtained by stacking all the matrix slices together. For example $T_{(1)}$ denotes flattening of a tensor $T \in \mathbb{R}^{n \times m \times p}$ to

[1] Word embeddings are vector representations of words, which can then be used as features for higher-level machine learning. Word embeddings have rapidly become the backbone of many downstream natural language processing tasks (see e.g. (Mikolov et al., 2013b)).

a $(n \times mp)$ matrix. We denote the Khatri-Rao product of two matrices $A$ and $B$ as $(A \odot B)_i = (A_i \otimes B_i)_{(1)}$, where $(A_i \otimes B_i)_{(1)}$ denotes the flattening of the matrix $A_i \otimes B_i$ into a row vector. For any tensor $T$ and vectors $a, b, c$, we also define $T(a, b, c) = \sum_{i,j,k} T_{ijk} a_i b_j c_k$. Throughout, we say $f(n) = \tilde{O}(g(n))$ if $f(n) = O(g(n))$ up to poly-logarithmic factors.

Though all algorithms in the paper extend to asymmetric tensors, we prove convergence results under the symmetric setting where $A = B = C$. Similar to other works (Tang & Shah, 2015; Anandkumar et al., 2014c; Ma et al., 2016), our guarantees depend on the incoherence of the factor matrices ($c_{\max}$), defined to be the maximum correlation in absolute value between any two factors, i.e. $c_{\max} = \max_{i \neq j} |A_i^T A_j|$. This serves as a natural assumption to simplify the problem as it is NP-Hard in the worst case. Also, tensors with randomly drawn factors satisfy $c_{\max} \leq \tilde{O}(1/\sqrt{d})$, and our results hold for such tensors.

## 3. Background and Related Work

We begin the section with a brief discussion of related work on tensor decomposition. We then review the ALS algorithm and the tensor power method and discuss their basic properties. Our proposed tensor decomposition algorithm, Orth-ALS, builds on these algorithms.

### 3.1. Related Work on Tensor Decomposition

Though it is not possible for us to do justice to the substantial body of work on tensor decomposition, we will review three families of algorithms which are distinct from alternating minimization approaches such as ALS and the tensor power method. Many algorithms have been proposed for guaranteed decomposition of *orthogonal* tensors, we refer the reader to Anandkumar et al. (2014b); Kolda & Mayo (2011); Comon et al. (2009); Zhang & Golub (2001). However, obtaining guaranteed recovery of non-orthogonal tensors using algorithms for orthogonal tensors requires converting the tensor into an orthogonal form (known as *whitening*) which is ill conditioned in high dimensions (Le et al., 2011; Souloumiac, 2009), and is computationally the most expensive step (Huang et al., 2013). Another very interesting line of work on tensor decompositions is to use simultaneous diagonalization and higher order SVD (Colombo & Vlassis, 2016; Kuleshov et al., 2015; De Lathauwer, 2006) but these are not as computationally efficient as alternating minimization[2]. Recently, there has been intriguing work on provably decomposing random tensors using the *sum-of-squares* approach (Ma et al., 2016; Hopkins et al., 2016; Tang & Shah, 2015; Ge & Ma, 2015). Ma et al. (2016) show that a sum-of-squares based relaxation can decompose highly overcomplete random tensors of rank up to $o(d^{1.5})$. Though these results establish the polynomial learnability of the problem, they are unfortunately not practical.

Very recently, there has been exciting work on scalable tensor decomposition algorithms using ideas such as sketching (Song et al., 2016; Wang et al., 2015b) and contraction of tensor problems to matrix problems (Shah et al., 2015). Also worth noting are recent approaches to speedup ALS via sampling and randomized least squares (Battaglino et al., 2017; Cheng et al., 2016; Papalexakis et al., 2012).

### 3.2. Alternating Least Squares (ALS)

ALS is the most widely used algorithm for tensor decomposition and has been described as the "workhorse" for tensor decomposition (Kolda & Bader, 2009). The algorithm is conceptually very simple: if the goal is to recover a rank-$k$ tensor, ALS maintains a rank-$k$ decomposition specified by three sets of $d \times k$ dimensional matrices $\{\hat{A}, \hat{B}, \hat{C}\}$ corresponding to the three modes of the tensor. ALS will iteratively fix two of the three modes, say $\hat{A}$ and $\hat{B}$, and then update $\hat{C}$ by solving a least-squared regression problem to find the best approximation to the underlying tensor $T$ having factors $\hat{A}$ and $\hat{B}$ in the first two modes, namely $\hat{C}_{new} = \arg\min_{C'} \|T - \hat{A} \otimes \hat{B} \otimes \hat{C}'\|_2$. ALS will then continue to iteratively fix two of the three modes, and update the other mode via solving the associated least-squares regression problem. These updates continue until some stopping condition is satisfied—typically when the squared error of the approximation is no longer decreasing, or when a fixed number of iterations have elapsed. The factors used in ALS are either chosen uniformly at random, or via a more expensive initialization scheme such as SVD based initialization (Anandkumar et al., 2014c). In the SVD based scheme, the factors are initialized to be the singular vectors of a random projection of the tensor onto a matrix.

The main advantages of the ALS approach, which have led to its widespread use in practice are its conceptual simplicity, noise robustness and computational efficiency given its graceful handling of sparse tensors and ease of parallelization. There are several publicly available optimized packages implementing ALS, such as Kossaifi et al. (2016); Vervliet et al.; Bader et al. (2012); Bader & Kolda (2007); Smith & Karypis; Huang et al. (2014); Kang et al. (2012).

Despite the advantages, ALS does not have any global convergence guarantees and can get stuck in local optima (Comon et al., 2009; Kolda & Bader, 2009), even under very realistic settings. For example, consider a setting where the weights $w_i$ for the factors $\{A_i, B_i, C_i\}$ decay

[2] De Lathauwer (2006) prove unique recovery under very general conditions, but their algorithm is quite complex and requires solving a linear system of size $O(d^4)$, which is prohibitive for large tensors. We ran the simultaneous diagonalization algorithm of Kuleshov et al. (2015) on a dimension 100, rank 30 tensor; and the algorithm needed around 30 minutes to run, whereas Orth-ALS converges in less than 5 seconds.

according to a power-law, hence the first few factors have much larger weight than the others. As we show in the experiments (see Fig. 2), ALS fails to recover the low-weight factors. Intuitively, this is because multiple recovered factors will be chasing after the *same* high weight factor, leading to a bad local optima.

### 3.3. Tensor Power Method

The tensor power method is a special case of ALS that only computes a rank-1 approximation. The procedure is then repeated multiple times to recover different factors. The factors recovered in different iterations of the algorithm are then clustered to determine the set of unique factors. Different initialization strategies have been proposed for the tensor power method. Anandkumar et al. (2014c) showed that the tensor power method converges locally (i.e. for a suitably chosen initialization) for random tensors with rank $o(d^{1.5})$. They also showed that a SVD based initialization strategy gives good starting points and used this to prove global convergence for random tensors with rank $O(d)$. However, the SVD based initialization strategy can be computationally expensive, and our experiments suggest that even SVD initialization fails in the setting where the weights decay according to a power-law (see Fig. 2).

In this work, we prove global convergence guarantees with random initializations for the tensor power method for random and worst-case incoherent tensors. Our results also demonstrate how, with random initialization, the tensor power method converges to the factor having the largest product of weight times the correlation of the factor with the random initialization vector. This explains the difficulty of using random initialization to recover factors with small weight. For example, if one factor has weight less than a $1/c$ fraction of the weight of, say, the heaviest $k/2$ factors, then with high probability we would require at least $k^{\Theta(c^2)}$ random initializations to recover this factor. This is because the correlation between random vectors in high dimensions is approximately distributed as a Normal random variable and if $k/2+1$ samples are drawn from the standard Normal distribution, the probability that one particular sample is at least a factor of $c$ larger than the other $k/2$ other samples scales as roughly $k^{-\Theta(c^2)}$.

## 4. The Algorithm: Orthogonalized Alternating Least Squares (Orth-ALS)

In this section we introduce Orth-ALS, which combines the computational benefits of standard ALS and the provable recovery of the tensor power method, while avoiding the difficulties faced by both when factors have different weights. Orth-ALS is a simple modification of standard ALS that adds an orthogonalization step before each set of ALS steps. We describe the algorithm below. Note that steps 4-6 are just the solution to the least squares problem expressed in compact tensor notation, for instance step 4 can be equivalently stated as $X = \arg\min_{C'} \|T - \hat{A} \otimes \hat{B} \otimes \hat{C}'\|_2$. Similarly, step 9 is the least squares estimate of the weight $w_i$ of each rank-1 component $\hat{A}_i \otimes \hat{B}_i \otimes \hat{C}_i$.

**Algorithm 1** Orthogonalized ALS (Orth-ALS)

**Input:** Tensor $T \in \mathbb{R}^{d \times d \times d}$, number of iterations $N$.

1: Initialize each column of $\hat{A}, \hat{B}$ and $\hat{C} \in \mathbb{R}^{d \times k}$ uniformly from the unit sphere
2: **for** $t = 1 : N$ **do**
3: Find $QR$ decomposition of $\hat{A}$, set $\hat{A} = Q$. Orthogonalize $\hat{B}$ and $\hat{C}$ analogously.
4: $X = T_{(1)}(\hat{C} \odot \hat{B})$
5: $Y = T_{(2)}(\hat{C} \odot \hat{A})$
6: $Z = T_{(3)}(\hat{B} \odot \hat{A})$
7: Normalize $X, Y, Z$ and store results in $\hat{A}, \hat{B}, \hat{C}$
8: **end for**
9: Estimate weights $\hat{w}_i = T(\hat{A}_i, \hat{B}_i, \hat{C}_i), \forall\, i \in [k]$.
10: **return** $\hat{A}, \hat{B}, \hat{C}, \hat{w}$

To get some intuition for why the orthogonalization makes sense, let us consider the more intuitive *matrix* factorization problem, where the goal is to compute the eigenvectors of a matrix. Subspace iteration is a straightforward extension of the matrix power method to recover all eigenvectors at once. In subspace iteration, the matrix of eigenvector estimates is orthogonalized before each power method step (by projecting the second eigenvector estimate orthogonal to the first one and so on), because otherwise all the vectors would converge to the dominant eigenvector. For the case of tensors, the vectors would not all necessarily converge to the dominant factor if the initialization is good, but with high probability a random initialization would drive many factors towards the larger weight factors. The orthogonalization step is a natural modification which forces the estimates to converge to different factors, even if some factors are much larger than the others. It is worth stressing that the orthogonalization step does *not* force the final recovered factors to be orthogonal (because the ALS step follows the orthogonalization step) and in general the factors output will not be orthogonal (which is essential for accurately recovering the factors).

From a computational perspective, adding the orthogonalization step does not add to the computational cost as the least squares updates in step 4-6 of Algorithm 1 involve an extra pseudoinverse term for standard ALS, which evaluates to identity for Orth-ALS and does not have to be computed. The cost of orthogonalization is $O(k^2 d)$, while the cost of computing the pseudoinverse is also $O(k^2 d)$. We also observe significant speedups in terms of the number of iterations required for convergence for Orth-ALS as compared to standard ALS in our simulations on random tensors (see the experiments in Section 5).

**Variants of Orthogonalized ALS.** Several other modifications to the simple orthogonalization step also seem natural. Particularly for low-dimensional settings, in practice we found that it is useful to carry out orthogonalization for a few steps and then continue with standard ALS updates until convergence (we call this variant *Hybrid-ALS*). Hybrid-ALS also gracefully reverts to standard ALS in settings where the factors are highly correlated and orthogonalization is not helpful.

### 4.1. Performance Guarantees

We now state the formal guarantees on the performance of Orthogonalized ALS. The specific variant of Orthogonalized ALS that our theorems apply to is a slight modification of Algorithm 1, and differs in that there is a periodic (every $\log k$ steps) re-randomization of the factors for which our analysis has not yet guaranteed convergence. In our practical implementations, we observe that *all* factors seem to converge within this first $\log k$ steps, and hence the subsequent re-randomization is unnecessary.

**Theorem 1.** *Consider a $d$-dimensional rank $k$ tensor $T = \sum_{i=1}^{k} w_i A_i \otimes A_i \otimes A_i$. Let $c_{\max} = \max_{i \neq j} |A_i^T A_j|$ be the incoherence between the true factors and $\gamma = \frac{w_{\max}}{w_{\min}}$ be the ratio of the largest and smallest weight. Assume $\gamma c_{\max} \leq o(k^{-2})$, and the estimates of the factors are initialized randomly from the unit sphere. Provided that, at the $i(\log k + \log\log d)$th step of the algorithm the estimates for all but the first $i$ factors are re-randomized, then with high probability the orthogonalized ALS updates converge to the true factors in $O(k(\log k + \log\log d))$ steps, and the error at convergence satisfies (up to relabelling) $\| A_i - \hat{A}_i \|_2^2 \leq O(\gamma k \max\{c_{\max}^2, 1/d^2\})$ and $|1 - \frac{\hat{w}_i}{w_i}| \leq O(\max\{c_{\max}, 1/d\})$, for all $i$.*

Theorem 1 immediately gives convergence guarantees for random low rank tensors. For random $d$ dimensional tensors, $c_{\max} = O(1/\sqrt{d})$; therefore Orth-ALS converges globally with random initialization whenever $k = o(d^{0.25})$. If the tensor has rank much smaller than the dimension, then our analysis can tolerate significantly higher correlation between the factors. In the Appendix, we also prove Theorem 1 for the special and easy case of orthogonal tensors, which nevertheless highlights the key proof ideas.

### 4.2. New Guarantees for the Tensor Power Method

As a consequence of our analysis of the orthogonalized ALS algorithm, we also prove new guarantees on the tensor power method. As these may be of independent interest because of the wide use of the tensor power method, we summarize them in this section. We show a quadratic rate of convergence (in $O(\log\log d)$ steps) with random initialization for random tensors having rank $k = o(d)$. This contrasts with the analysis of Anandkumar et al. (2014c) who showed a linear rate of convergence ($O(\log d)$ steps) for random tensors, provided an SVD based initialization is employed.

**Theorem 2.** *Consider a $d$-dimensional rank $k$ tensor $T = \sum_{i=1}^{k} w_i A_i \otimes A_i \otimes A_i$ with the factors $A_i$ sampled uniformly from the $d$-dimensional sphere. Define $\gamma = \frac{w_{\max}}{w_{\min}}$ to be the ratio of the largest and smallest weight. Assume $k \leq o(d)$ and $\gamma \leq polylog(d)$. If the initialization $x_0 \in \mathbb{R}^d$ is chosen uniformly from the unit sphere, then with high probability the tensor power method updates converge to one of the true factors (say $A_1$) in $O(\log\log d)$ steps, and the error at convergence satisfies $\| A_1 - \hat{A}_1 \|_2 \leq \tilde{O}(1/\sqrt{d})$. Also, the estimate of the weight $\hat{w}_1$ satisfies $|1 - \frac{\hat{w}_1}{w_1}| \leq \tilde{O}(1/\sqrt{d})$.*

Theorem 2 provides guarantees for random tensors, but it is natural to ask if there are deterministic conditions on the tensors which guarantee global convergence of the tensor power method. Our analysis also allows us to obtain a clean characterization for global convergence of the tensor power method updates for worst-case tensors in terms of the incoherence of the factor matrix—

**Theorem 3.** *Consider a $d$-dimensional rank $k$ tensor $T = \sum_{i=1}^{k} w_i A_i \otimes A_i \otimes A_i$. Let $c_{\max} = \max_{i \neq j} |A_i^T A_j|$ and $\gamma = \frac{w_{\max}}{w_{\min}}$ be the ratio of the largest and smallest weight, and assume $\gamma c_{\max} \leq o(k^{-2})$. If the initialization $x_0 \in \mathbb{R}^d$ is chosen uniformly from the unit sphere, then with high probability the tensor power method updates converge to one of the true factors (say $A_1$) in $O(\log k + \log\log d)$ steps, and the error at convergence satisfies $\| A_1 - \hat{A}_1 \|_2^2 \leq O(\gamma k \max\{c_{\max}^2, 1/d^2\})$ and $|1 - \frac{\hat{w}_1}{w_1}| \leq O(\max\{c_{\max}, 1/d\})$.*

## 5. Experiments

We compare the performance of Orth-ALS, standard ALS (with random and SVD initialization), the tensor power method, and the classical eigendecomposition approach, through experiments on low rank tensor recovery in a few different parameter regimes, on a overcomplete tensor decomposition task and a tensor completion task. We also compare the factorization of Orth-ALS and standard ALS on a large real-world tensor of word tri-occurrence based on the 1.5 billion word English Wikipedia corpus.[3]

### 5.1. Experiments on Random Tensors

**Recovering low rank tensors:** We explore the abilities of Orth-ALS, standard ALS, and the tensor power method (TPM), to recover a low rank (rank $k$) tensor that has been constructed by independently drawing each of the $k$ factors independently and uniformly at random from the $d$ dimensional unit spherical shell. We consider several different

[3] MATLAB, Python and C code for Orth-ALS and Hybrid-ALS is available at http://web.stanford.edu/~vsharan/orth-als.html

combinations of the dimension, $d$, and rank, $k$. We also consider both the setting where all of the factors are equally weighted, as well as the practically relevant setting where the factor weights decay geometrically, and consider the setting where independent Gaussian noise has been added to the low-rank tensor.

In addition to random initialization for standard ALS and the TPM, we also explore SVD based initialization (Anandkumar et al., 2014c) where the factors are initialized via SVD of a projection of the tensor onto a matrix. We also test the classical technique for tensor decomposition via simultaneous diagonalization (Leurgans et al., 1993; Harshman, 1970) (also known as Jennrich's algorithm, we refer to it as Sim-Diag), which first performs two random projections of the tensor, and then recovers the factors by an eigenvalue decomposition of the projected matrices. This gives guaranteed recovery when the tensors are noiseless and factors are linearly independent, but is extremely unstable to perturbations.

We evaluate the performance in two respects: 1) the ability of the algorithms to recover a low-rank tensor that is close to the input tensor, and 2) the ability of the algorithms to recover accurate approximations of many of the true factors. Fig. 1 depicts the performance via the first metric. We evaluate the performance in terms of the discrepancy between the input low-rank tensor, and the low-rank tensor recovered by the algorithms, quantified via the ratio of the Frobenius norm of the residual, to the Frobenius norm of the actual tensor: $\frac{\|T-\hat{T}\|_F}{\|T\|_F}$, where $\hat{T}$ is the recovered tensor. Since the true tensor has rank $k$, the inability of an algorithm to drive this error to zero indicates the presence of local optima. Fig. 1 depicts the performance of Orth-ALS, standard ALS with random initialization and the hybrid algorithm that performs Orth-ALS for the first five iterations before reverting to standard ALS (Hybrid-ALS). Tests are conducted in both the setting where factor weights are uniform, as well as a geometric spacing, where the ratio of the largest factor weight to the smallest is 100. Fig. 1 shows that Hybrid ALS and Orth-ALS have much faster convergence and find a significantly better fit than standard ALS.

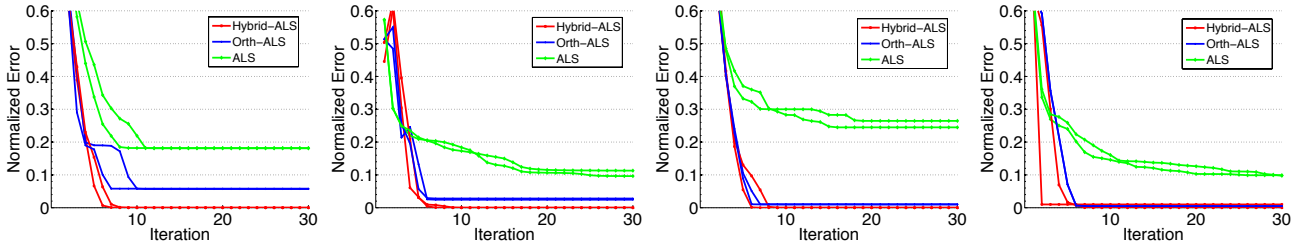
Fig. 2 quantifies the performance of the algorithms in terms of the number of the original factors that the algorithms accurately recover. We use standard ALS, Orth-ALS (Algorithm 1), Hybrid-ALS, TPM with random initialization (TPM), ALS with SVD initialization (ALS-SVD), TPM with SVD initialization (TPM-SVD) and the simultaneous diagonalization approach (Sim-Diag). We run TPM and SVD-TPM with 100 different initializations and find a rank $k = 30$ decomposition for ALS, ALS-SVD, Orth-ALS, Hybrid-ALS and Sim-Diag. We repeat the experiment (by sampling a new tensor) 10 times. We perform this evaluation in both the setting where we receive an actual low-rank tensor as input, as well as the setting where each entry $T_{ijk}$ of the low-rank tensor has been perturbed by independent Gaussian noise of standard deviation equal to $0.05T_{ijk}$. We can see that Orth-ALS and Hybrid-ALS perform significantly better than the other algorithms and are able to recover all factors in the noiseless case even when the weights are highly skewed. Note that the reason the Hybrid-ALS and Orth-ALS fail to recover all factors in the noisy case when the weights are highly skewed is that the magnitude of the noise essentially swamps the contribution from the smallest weight factors.

**Recovering over-complete tensors:** Overcomplete tensors are tensors with rank higher than the dimension, and have found numerous theoretical applications in learning latent variable models (Anandkumar et al., 2015). Even though orthogonalization cannot be directly applied to the setting where the rank is more than the dimension (as the factors can no longer be orthogonalized), we explore a deflation based approach in this setting. Given a tensor $T$ with dimension $d = 50$ and rank $r > d$, we find a rank $d$ decomposition $T_1$ of $T$, subtract $T_1$ from $T$, and then compute a rank $d$ decomposition of $T_1$ to recover the next set of $d$ factors. We repeat this process to recover subsequent factors. After every set of $d$ factors has been estimated, we also refine the factor estimates of all factors estimated so far by running an additional ALS step using the current estimates of the extracted factors as the initialization. Fig. 3a plots the number of factors recovered when this deflation based approach is applied to a dimension $d = 50$ tensor with a mild power low distribution on weights. We can see that Hybrid-ALS is successful at recovering tensors even in the overcomplete setup, and gives an improvement over ALS.

**Tensor completion:** We also test the utility of orthogonalization on a tensor completion task, where the goal is to recover a large missing fraction of the entries. Fig. 3b suggests Hybrid-ALS gives considerable improvements over standard ALS. Further examining the utility of orthogonalization in this important setting, in theory and practice, would be an interesting direction.

### 5.2. Learning Word Embeddings via Tensor Factorization

A word embedding is a vector representation of words which preserves some of the syntactic and semantic relationships in the language. Current methods for learning word embeddings implicitly (Mikolov et al., 2013b; Levy & Goldberg, 2014) or explicitly (Pennington et al., 2014) factorize some matrix derived from the matrix of word co-occurrences $M$, where $M_{ij}$ denotes how often word $i$ appears with word $j$. We explore tensor methods for learning word embeddings, and contrast the performance of standard ALS and Orthogonalized ALS on standard tasks.



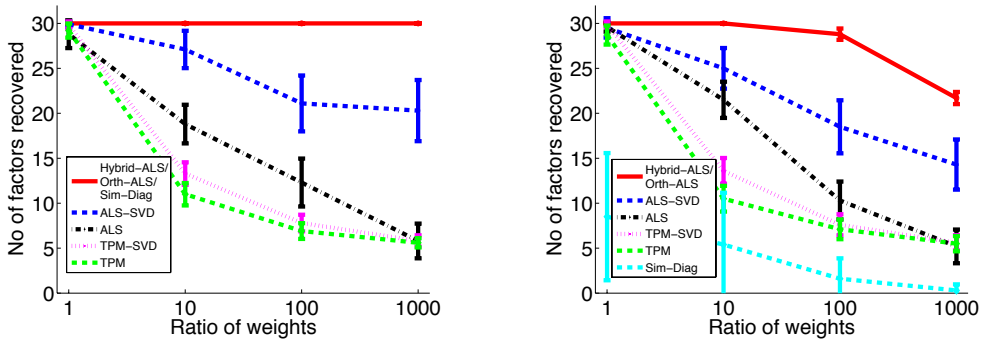

(a) $k = 30, d = 100$, uniform weights

(b) $k = 30, d = 100$, $\frac{w_{\max}}{w_{\min}} = 100$

(c) $k = 100, d = 1000$, uniform weights

(d) $k = 100, d = 1000$, $\frac{w_{\max}}{w_{\min}} = 100$

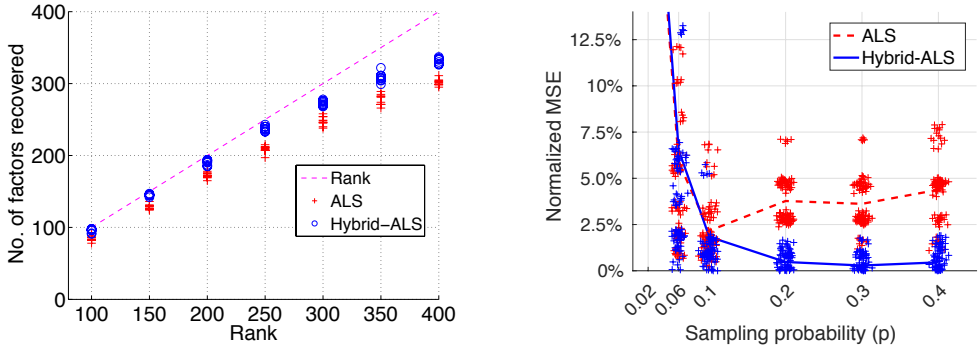
*Figure 1.* Plot of the normalized discrepancy between the recovered rank $k$ tensor $\hat{T}$ and the true tensor $T$: $\frac{\|T-\hat{T}\|_{\mathrm{F}}}{\|T\|_{\mathrm{F}}}$, as a function of the iteration. In all settings, the Orth-ALS and the hybrid algorithm drive this discrepancy nearly to zero, with the performance of Orth-ALS improving for the higher dimensional cases, whereas standard ALS algorithm has slower convergence and gets stuck in bad local optima.




(a) Noiseless case, ratio of weights equals $\frac{w_{\max}}{w_{\min}}$

(b) Noisy case, ratio of weights equals $\frac{w_{\max}}{w_{\min}}$

*Figure 2.* Average number of factors recovered by different algorithms for different values of $\frac{w_{\max}}{w_{\min}}$, the ratio of the maximum factor weight to minimum factor weight (with the weights spaced geometrically), along with error bars for the standard deviation in the number of factors recovered, across independent trials. The true rank $k = 30$, and the dimension $d = 100$. We say a factor $\{A_i, B_i, C_i\}$ of the tensor $T$ is successfully recovered if there exists at least one recovered factor $\{\hat{A}_j, \hat{B}_j, \hat{C}_j\}$ with correlation at least 0.9 in all modes. Orth-ALS and Hybrid-ALS recover all factors in almost all settings, whereas ALS and the tensor power method struggle when the weights are skewed, even with the more expensive SVD based initialization.




(a) No. of factors recovered for a dimension $d = 50$ tensor with varying rank. The weights of the factors are geometrically spaced and the ratio of the weights between every pair of consecutive factors is 1.05

(b) Average error on the missing entries for tensor completion where each entry is sampled with probability $p$, on a 100 different runs with each setting of $p$. The tensor has dimension $d = 50$ and rank $k = 10$.

*Figure 3.* Experiments on overcomplete tensors and tensor completion. Even though our theoretical guarantees do not apply to these settings, we see that orthogonalization leads to significantly better performance over standard ALS.

**Methodology.** We used the English Wikipedia as our corpus, with 1.5 billion words. We constructed a word co-occurrence tensor $T$ of the 10,000 most frequent words, where the entry $T_{ijk}$ denotes the number of times the words $i$, $j$ and $k$ appear in a sliding window of length $w$ across the corpus. We consider two different window lengths, $w = 3$ and $w = 5$. Before factoring the tensor, we apply the non-linear element-wise scaling $f(x) = \log(1 + x)$ to the tensor. This scaling is known to perform well in practice for co-occurrence matrices (Pennington et al., 2014), and makes some intuitive sense in light of the Zipfian distribution of word frequencies. Following the application of this element-wise nonlinearity, we recover a rank 100 approximation of the tensor using Orth-ALS or ALS.

We concatenate the (three) recovered factor matrices into one matrix and normalize the rows. The $i$th row of this matrix is then the embedding for the $i$th word. We test the quality of these embeddings on two tasks aimed at measuring the syntactic and semantic structure captured by these word embeddings.

We also evaluated the performance of matrix SVD based methods on the task. For this, we built the co-occurrence matrix $M$ with a sliding window of length $w$ over the corpus. We applied the same non-linear element-wise scaling and performed a rank 100 SVD, and set the word embeddings to be the singular vectors after row normalization.

It is worth highlighting some implementation details for our experiments, as they indicate the practical efficiency and scalability inherited by Orth-ALS from standard ALS. Our experiments were run on a cluster with 8 cores and 48 GB of RAM memory per core. Most of the runtime was spent in reading the tensor, the runtime for Orth-ALS was around 80 minutes, with 60 minutes spent in reading the tensor (the runtime for standard ALS was around 100 minutes because it took longer to converge). Since storing a dense representation of the 10,000×10,000×10,000 tensor is too expensive, we use an optimized ALS solver for sparse tensors (Smith & Karypis; 2015) which also has an efficient parallel implementation.

**Evaluation: Similarity and Analogy Tasks.** We evaluated the quality of the recovered word embeddings produced by the various methods via their performance on two different NLP tasks for which standard, human-labeled data exists: estimating the similarity between a pair of words, and completing word analogies.

The word similarity tasks (Bruni et al., 2012; Finkelstein et al., 2001) contain word pairs along with human assigned similarity scores, and the objective is to maximize the correlation between the similarity in the embeddings of the two words (according to a similarity metric such as the dot product) and human judged similarity.

| Algorithm | Similarity tasks | Analogy tasks |
|---|---|---|
| Standard ALS, $w = 3$ | 0.50 | 30.92% |
| Standard ALS, $w = 5$ | 0.50 | 37.38% |
| Orth-ALS, $w = 3$ | 0.59 | 40.00% |
| Orth-ALS, $w = 5$ | 0.60 | 46.37% |
| Matrix methods, $w = 3$ | 0.68 | 53.29% |
| Matrix methods, $w = 5$ | 0.67 | 57.40% |

*Table 1.* Results for word analogy and word similarity tasks for different window lengths $w$ over which the co-occurrences are counted. The embeddings recovered by Orth-ALS are significantly better than those recovered by standard ALS. Despite this, embeddings derived from word co-occurrences using matrix SVD still outperform the tensor embeddings, and we are unsure whether this is due to the relative sparsity of the tensor, sub-optimal element-wise scaling (i.e. the $f(x) = \log(1+x)$ function applied to the counts), or something more fundamental.

The word analogy tasks (Mikolov et al., 2013a;c) present questions of the form "$a$ is to $a^*$ as $b$ is to __?" (e.g. "Paris is to France as Rome is to __?"). We find the answer to "$a$ is to $a^*$ as $b$ is to $b^*$" by finding the word whose embedding is the closest to $w_{a^*} - w_a + w_b$ in cosine similarity, where $w_a$ denotes the embedding of the word $a$.

**Results.** The performances are summarized in the Table 1. The use of Orth-ALS rather than standard ALS leads to significant improvement in the quality of the embeddings as judged by the similarity and analogy tasks. However, the matrix SVD method still outperforms the tensor based methods. We believe that it is possible that better tensor based approaches (e.g. using better renormalization, additional data, or some other tensor rather than the symmetric tri-occurrence tensor) or a combination of tensor and matrix based methods can actually improve the quality of word embeddings, and is an interesting research direction. Alternatively, it is possible that natural language does not contain sufficiently rich higher-order dependencies among words that appear close together, beyond the co-occurrence structure, to truly leverage the power of tensor methods. Or, perhaps, the two tasks we evaluated on—similarity and analogy tasks—do not require this higher order. In any case, investigating these possibilities seems worthwhile.

## 6. Conclusion

Our results suggest the theoretical and practical benefits of Orthogonalized ALS, versus standard ALS. An interesting direction for future work would be to more thoroughly examine the practical and theoretical utility of orthogonalization for other tensor-related tasks, such as tensor completion. Additionally, its seems worthwhile to investigate Orthogonalized ALS or Hybrid ALS in more application-specific domains, such as natural language processing.

## References


Anandkumar, Animashree, Liu, Yi-kai, Hsu, Daniel J, Foster, Dean P, and Kakade, Sham M. A spectral algorithm for latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, pp. 917–925, 2012.

Anandkumar, Animashree, Ge, Rong, Hsu, Daniel, and Kakade, Sham M. A tensor approach to learning mixed membership community models. *The Journal of Machine Learning Research*, 15(1):2239–2312, 2014a.

Anandkumar, Animashree, Ge, Rong, Hsu, Daniel, Kakade, Sham M, and Telgarsky, Matus. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15(1):2773–2832, 2014b.

Anandkumar, Animashree, Ge, Rong, and Janzamin, Majid. Guaranteed non-orthogonal tensor decomposition via alternating rank-1 updates. *arXiv preprint arXiv:1402.5180*, 2014c.

Anandkumar, Animashree, Ge, Rong, and Janzamin, Majid. Learning overcomplete latent variable models through tensor methods. In *Proceedings of The 28th Conference on Learning Theory*, pp. 36–112, 2015.

Azizzadenesheli, Kamyar, Lazaric, Alessandro, and Anandkumar, Animashree. Reinforcement learning of POMDPs using spectral methods. In *29th Annual Conference on Learning Theory*, pp. 193–256, 2016.

Bader, Brett W. and Kolda, Tamara G. Efficient MATLAB computations with sparse and factored tensors. *SIAM Journal on Scientific Computing*, 30(1), December 2007.

Bader, Brett W., Kolda, Tamara G., et al. Matlab tensor toolbox version 2.5. Available online, January 2012.

Battaglino, Casey, Ballard, Grey, and Kolda, Tamara G. A practical randomized CP tensor decomposition. *arXiv preprint arXiv:1701.06600*, 2017.

Bruni, Elia, Boleda, Gemma, Baroni, Marco, and Tran, Nam-Khanh. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, 2012.

Chaganty, Arun Tejasvi and Liang, Percy. Estimating latent-variable graphical models using moments and likelihoods. In *ICML*, pp. 1872–1880, 2014.

Cheng, Dehua, Peng, Richard, Liu, Yan, and Perros, Ioakeim. SPALS: Fast alternating least squares via implicit leverage scores sampling. In *Advances In Neural Information Processing Systems*, pp. 721–729, 2016.

Colombo, Nicolo and Vlassis, Nikos. FastMotif: spectral sequence motif discovery. *Bioinformatics*, 31(16), 2015.

Colombo, Nicolo and Vlassis, Nikos. Tensor decomposition via joint matrix schur decomposition. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 2820–2828, 2016.

Comon, Pierre, Luciani, Xavier, and De Almeida, André LF. Tensor decompositions, alternating least squares and other tales. *Journal of chemometrics*, 23 (7-8):393–405, 2009.

De Lathauwer, Lieven. A link between the canonical decomposition in multilinear algebra and simultaneous matrix diagonalization. *SIAM journal on Matrix Analysis and Applications*, 28(3):642–666, 2006.

Duembgen, Lutz. Bounding standard gaussian tail probabilities. *arXiv preprint arXiv:1012.2063*, 2010.

Finkelstein, Lev, Gabrilovich, Evgeniy, Matias, Yossi, Rivlin, Ehud, Solan, Zach, Wolfman, Gadi, and Ruppin, Eytan. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pp. 406–414. ACM, 2001.

Ge, Rong and Ma, Tengyu. Decomposing overcomplete 3rd order tensors using sum-of-squares algorithms. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pp. 829, 2015.

Ge, Rong, Huang, Qingqing, and Kakade, Sham M. Learning mixtures of gaussians in high dimensions. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pp. 761–770. ACM, 2015.

Harshman, Richard A. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

Håstad, Johan. Tensor rank is NP-Complete. *Journal of Algorithms*, 11(4):644–654, 1990.

Hillar, Christopher J and Lim, Lek-Heng. Most tensor problems are NP-Hard. *Journal of the ACM*, 60(6), 2013.

Hopkins, Samuel B, Schramm, Tselil, Shi, Jonathan, and Steurer, David. Fast spectral algorithms from sum-of-squares proofs: tensor decomposition and planted sparse vectors. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, 2016.

Huang, Furong, Niranjan, UN, Hakeem, Mohammad Umar, and Anandkumar, Animashree. Fast detection of overlapping communities via online tensor methods. *arXiv preprint arXiv:1309.0787*, 2013.

Huang, Furong, Matusevych, Sergiy, Anandkumar, Anima, Karampatziakis, Nikos, and Mineiro, Paul. Distributed latent dirichlet allocation via tensor factorization. In *NIPS Optimization Workshop*, 2014.

Kang, U, Papalexakis, Evangelos, Harpale, Abhay, and Faloutsos, Christos. Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012.

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

Kolda, Tamara G and Mayo, Jackson R. Shifted power method for computing tensor eigenpairs. *SIAM Journal on Matrix Analysis and Applications*, 32(4), 2011.

Kossaifi, Jean, Panagakis, Yannis, and Pantic, Maja. Tensorly: Tensor learning in python. *arXiv preprint arXiv:1610.09555*, 2016.

Kruskal, Joseph B. 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.

Kuleshov, Volodymyr, Chaganty, Arun Tejasvi, and Liang, Percy. Tensor factorization via matrix factorization. In *AISTATS*, 2015.

Le, Quoc V, Karpenko, Alexandre, Ngiam, Jiquan, and Ng, Andrew Y. ICA with reconstruction cost for efficient overcomplete feature learning. In *Advances in Neural Information Processing Systems*, pp. 1017–1025, 2011.

Leurgans, SE, Ross, RT, and Abel, RB. A decomposition for three-way arrays. *SIAM Journal on Matrix Analysis and Applications*, 14(4):1064–1083, 1993.

Levy, Omer and Goldberg, Yoav. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, pp. 2177–2185, 2014.

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

Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013b.

Mikolov, Tomas, Yih, Wen-tau, and Zweig, Geoffrey. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pp. 746–751, 2013c.

Papalexakis, Evangelos E, Faloutsos, Christos, and Sidiropoulos, Nicholas D. Parcube: Sparse parallelizable tensor decompositions. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 521–536. Springer, 2012.

Pennington, Jeffrey, Socher, Richard, and Manning, Christopher D. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Shah, Parikshit, Rao, Nikhil, and Tang, Gongguo. Sparse and low-rank tensor decomposition. In *Advances in Neural Information Processing Systems*, 2015.

Smith, Shaden and Karypis, George. SPLATT: The Surprisingly ParalleL spArse Tensor Toolkit.

Smith, Shaden and Karypis, George. DMS: Distributed sparse tensor factorization with alternating least squares. Technical report, 2015.

Song, Zhao, Woodruff, David, and Zhang, Huan. Sublinear time orthogonal tensor decomposition. In *Advances in Neural Information Processing Systems*, 2016.

Souloumiac, Antoine. Joint diagonalization: Is non-orthogonal always preferable to orthogonal? In *3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2009.

Tang, Gongguo and Shah, Parikshit. Guaranteed tensor decomposition: A moment approach. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1491–1500, 2015.

Vervliet, N., Debals, O., Sorber, L., Van Barel, M., and De Lathauwer, L. Tensorlab 3.0, Mar. . Available online.

Wang, Yichen, Chen, Robert, Ghosh, Joydeep, Denny, Joshua C, Kho, Abel, Chen, You, Malin, Bradley A, and Sun, Jimeng. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1265–1274. ACM, 2015a.

Wang, Yining, Tung, Hsiao-Yu, Smola, Alexander J, and Anandkumar, Anima. Fast and guaranteed tensor decomposition via sketching. In *Advances in Neural Information Processing Systems*, pp. 991–999, 2015b.

Yu, Rose and Liu, Yan. Learning from multiway data: Simple and efficient tensor regression. In *Proceedings of the 33nd International Conference on Machine Learning (ICML-16)*, pp. 238–247, 2016.

Zhang, Tong and Golub, Gene H. Rank-one approximation to high order tensors. *SIAM Journal on Matrix Analysis and Applications*, 23(2):534–550, 2001.