Iterative Collaborative Filtering for Sparse Noisy Tensor Estimation

Devavrat Shah
Electrical Engineering and Computer Science
Massachusetts Institute of Technology, Cambridge, MA
Email: devavrat@mit.edu

Christina Lee Yu
Operations Research and Information Engineering
Cornell University, Ithaca, NY
Email: cleeyu@cornell.edu

***Abstract*—We consider the task of tensor estimation, i.e. estimating a low-rank 3-order $n \times n \times n$ tensor from noisy observations of randomly chosen entries in the sparse regime. In the context of matrix (2-order tensor) estimation, a variety of algorithms have been proposed and analyzed in the literature including the popular collaborative filtering algorithm that is extremely well utilized in practice. However, in the context of tensor estimation, there is limited progress. No natural extensions of collaborative filtering are known beyond "flattening" the tensor into a matrix and applying standard collaborative filtering.**

As the main contribution of this work, we introduce a generalization of the collaborative filtering algorithm for the setting of tensor estimation and argue that it achieves sample complexity that (nearly) matches the conjectured lower bound on the sample complexity. Interestingly, our generalization uses the matrix obtained from the "flattened" tensor to compute similarity as in the classical collaborative filtering but by defining a novel "graph" using it. The algorithm recovers the tensor with mean-squared-error (MSE) decaying to 0 as long as each entry is observed independently with probability $p = \Omega(n^{-3/2+\epsilon})$ for any arbitrarily small $\epsilon > 0$. It turns out that $p = \Omega(n^{-3/2})$ is the conjectured lower bound as well as "connectivity threshold" of graph considered to compute similarity in our algorithm.

I. Introduction

Tensor estimation involves the task of predicting underlying structure in a high-dimensional dataset given only a sparse subset of observations. We call this "tensor estimation" rather than the conventional "tensor completion" as the goal is not only to fill missing entries but also to estimate noisily observed entries. Whereas matrices represent data associated to two dimensions, rows and columns, tensors represent data associated to general d dimensions. In an e-commerce platform, a datapoint collected from an interaction may be associated to a user, product, and date/time, which could be represented in a 3-order tensor where the three dimensions would correspond to users, products, and date/time. Image data is also naturally represented in a 3-order tensor format, with two dimensions representing the location of the pixel, and the third dimension representing the RGB color components. Extending algorithms from matrix to general tensor estimation is important to allow for more flexible analysis of higher order data.

II. Related Literature

Algorithms for analyzing sparse low rank matrices (equivalent to 2-order tensors) where the observations are sampled uniformly randomly have been well-studied. The algorithms consist of spectral decompositions (e.g. USVT) [1]–[3], nuclear norm minimization [4]–[9], gradient descent [1], [2], [10]–[12], alternating minimization [13], [14], and nearest neighbor style collaborative filtering [15]–[17]. These algorithms have been shown to be provably consistent as long as the number of observations is $\Omega(rn \operatorname{polylog} n)$ (e.g. [2], [3], [5]) or $\omega(r^5 n)$ [17], where r is the rank and n is the number of rows and columns. Lower bounds show that $\Omega(rn)$ samples are necessary for consistent estimation, and $\Omega(rn \log(n))$ samples are necessary for exact recovery [5], [6], implying that the proposed algorithms are nearly sample efficient order-wise up to the information theoretic lower bounds.

There are results extending matrix estimation algorithms to general higher dimensional tensor estimation, assuming the tensor is low rank and that observations are sampled uniformly randomly. Most commonly, they flatten or unfold the tensor to a matrix and subsequently apply matrix estimation algorithms [18]–[21]. A d-order tensor where each dimension is length n would be flattened to a $n^{\lfloor d/2 \rfloor} \times n^{\lceil d/2 \rceil}$ matrix, resulting in a sample complexity of $\Omega(n^{\lceil d/2 \rceil}\text{polylog}(n))$.

There are a few algorithms that improve upon this bound when d is odd, each mirroring a different technique used originally in matrix estimation [22]–[27]. For a 3-order tensor, they provide consistent estimators requiring only a sample complexity of $\Omega(n^{3/2}\text{polylog}(n))$. [22], [23] analyzes the alternating minimization algorithm for exact recovery of the tensor given noiseless observations and finite rank $r = \Theta(1)$. [24] analyzes nuclear norm minimization with a suitable tensor nuclear norm; however, as computing tensor nuclear norm is NP-hard [28], the algorithm is not efficiently computable. [25], [26] use the sum of squares method, and [27] introduces a spectral method. Both of these latter algorithms can handle noisy observations and overcomplete tensors where the rank is larger than the dimension.

The sum of squares hierarchy conjectures a matching computational sample complexity lower bound of $\Omega(n^{3/2})$ for polynomial time estimators [25]. Information theoretic bounds imply that one needs at least $\Omega(rn)$ observations to recover the underlying tensor, consistent with the degrees of freedom or number of parameters to specify a rank r tensor. Interestingly, this implies a conjectured gap between the computational and statistically achievable sample complexities.

III. Contribution

We propose a new algorithm for symmetric tensor estimation which generalizes nearest neighbor collaborative filtering algorithms [17]. It is provably consistent with sample complexity $\Omega(n^{3/2+\epsilon})$ for any arbitrarily small $\epsilon > 0$. This shows that our simple iterative collaborative filtering algorithm nearly achieves the conjectured computational sample complexity lower bound of $\Omega(n^{3/2})$ for tensor estimation. While we present the results for symmetric tensors, our method and analysis can extend to asymmetric tensors, which we discuss in section VII. Our algorithm is a different style than the previously proposed SOS, spectral, or alternating minimization algorithms. It follows the nearest neighbor framework of similarity based collaborative filtering algorithms, which first computes similarities between coordinates, and then estimates entries by averaging datapoints from similar coordinates.

The nonobvious part of the algorithm is how to determine similarities between coordinates given sparse observations. The basic idea is to flatten the tensor into a matrix, and consider the structure in the graph constructed from the sparsity pattern of the matrix. Our algorithm computes similarities between two coordinates by comparing the local neighborhoods associated to each coordinate in the graph. The algorithm and analysis also sheds insight on the conjectured lower bound, as the threshold of $n^{3/2}$ is precisely the density of observations needed for connectivity in the associated graph. A benefit of our algorithm is that it can be implemented in a parallelized manner where the similarities between pair of indices are computed in parallel. This lends itself to distributed, scalable implementation. A naive bound on sample complexity of our algorithm for 3-order tensor is at most pn^6. By using approximate nearest neighbor indices, these can be further improved and made truly implementable.

IV. Problem Statement and Model

Consider an $n \times n \times n$ symmetric tensor F generated as follows: For each $u \in [n]$, sample $\theta_u \sim U[0,1]$ independently. Let the true underlying tensor F be described by a Lipschitz function f evaluated over the latent variables, $F(u,v,w) = f(\theta_u, \theta_v, \theta_w)$ for $u, v, w \in [n]$. Additionally assume that the function has finite spectrum,

$$f(u,v,w) = \sum_{k=1}^{r} \lambda_k q_k(\theta_u) q_k(\theta_v) q_k(\theta_w),$$

where $r = \Theta(1)$ and $q_k(.)$ denotes the orthonormal ℓ_2 eigenfunctions, satisfying $\int_0^1 q_k(\theta)^2 d\theta = 1$ and $\int_0^1 q_k(\theta) q_h(\theta) d\theta = 0$ for $k \neq h$. Assume that the eigenfunctions are bounded, i.e. $|q_k(\theta)| \leq B$ for all $k \in [r]$.

Let M denote the observed symmetric data tensor, and let $\Omega \subseteq [n]^3$ denote the set of observed indices. Due to the symmetry, it is sufficient to restrict the index set to triplets (u,v,w) such that $u \leq v \leq w$, as the datapoint is identical for all other permutations of the same triplet. The datapoint at each of these distinct triplets $\{(u,v,w) : u \leq v \leq w\}$ is observed independently with probability $p \in (0,1]$, where we assume the observation is corrupted by mean zero independent additive noise terms. For $(u,v,w) \in \Omega$,

$$M(u,v,w) = F(u,v,w) + \delta_{uvw},$$

and for $(u,v,w) \notin \Omega$, $M(u,v,w) = 0$. We allow δ_{uvw} to have different distributions for different distinct triplets (u,v,w) as long as the variance is uniformly bounded by σ^2. The goal is to recover the underlying tensor F from the incomplete noisy observation M so that the mean squared error (MSE) is small, where MSE for an estimate $\hat{F}$ is defined as

$$\text{MSE}(\hat{F}) := \mathbb{E}\left[\frac{1}{n^3} \sum_{(u,v,w)\in[n]^3} (\hat{F}(u,v,w) - F(u,v,w))^2\right].$$

Let Λ denote the diagonal $r \times r$ matrix where $\Lambda_{kk} = \lambda_k$. Let Q denote the $r \times n$ matrix where $Q_{ka} = q_k(\theta_a)$. Let $\mathcal{Q}$ denote the $r \times \binom{n}{2}$ matrix where $\mathcal{Q}_{kb} = q_k(\theta_{b_1}) q_k(\theta_{b_2})$ for some $b \in \binom{n}{2}$ that represents the pair of vertices (b_1, b_2) for $b_1 < b_2$. The finite spectrum assumption for f implies a low rank tensor representation for F,

$$F = \sum_{k=1}^{r} \lambda_k (Q^T e_k) \otimes (Q^T e_k) \otimes (Q^T e_k).$$

A limitation of our model is that we assume constant rank $r = \Theta(1)$, which does not allow for overcomplete tensors where $r > n$. In addition we assume a generative latent variable model, which imposes additional smoothness beyond the low rank assumption. The specific condition that $\theta_u \sim U[0,1]$ is not necessary, however we do need the fact that the latent variables are sampled iid from some underlying distribution. This guarantees incoherence-style conditions, as extreme outliers would correspond to regions with extremely small mass in the distribution and thus are unlikely.

The Lipschitz assumption implies that in addition to low rank, the tensor is "smooth", and thus there are sets of rows and columns that are similar to one another. As our algorithm is based on a nearest neighbor style approach we need that for any coordinate u, there is a significant mass of other coordinates a that are similar to u with respect to the function behavior. Other regularity conditions beyond Lipschitz that would also guarantee sufficiently many "nearest neighbors" would plausibly lead to similar results for our algorithm.

V. Algorithm Intuition

The algorithm is a nearest neighbor style algorithm in which the first phase is to estimate a distance function between coordinates, denoted $\texttt{dist}(u,a)$ for all $(u,a) \in [n]^2$. Given the similarities, for some threshold ξ, the algorithm estimates by averaging datapoints from coordinates (a,b,c) for which $\texttt{dist}(u,a) \leq \xi$, $\texttt{dist}(v,b) \leq \xi$, and $\texttt{dist}(w,c) \leq \xi$.

The entry $F(a,b,c)$ depends on a coordinate a through its representation in the eigenspace, given by Qe_u. Therefore $f(a,b,c) \approx f(u,v,w)$ as long as $Qe_u \approx Qe_a$, $Qe_v \approx Qe_b$, and $Qe_w \approx Qe_c$. Ideally we would like our distance function $\texttt{dist}(u,a)$ to approximate $\|Qe_u - Qe_a\|_2$, but these are hidden latent features that we do not have direct access to.

Let's start with a thought experiment supposing that the density of observations were $p = \omega(n^{-1})$. Then for a pair of coordinates u and a, the expected number of pairs (b,c)

such that both (u,b,c) and (a,b,c) are observed is on the order of $p^2n^2 = \omega(1)$. For fixed θ_a, θ_u, and for randomly sampled θ_b, θ_c, the expected squared difference between the two corresponding datapoints reflects the distance between Qe_a and Qe_u along with the overall level of noise,

$$\begin{aligned}
&\mathbb{E}[(M(a,b,c) - M(u,b,c))^2 \mid \theta_a, \theta_u] \\
&= \mathbb{E}[(F(a,b,c) - F(u,b,c))^2 \mid \theta_a, \theta_u] + \mathbb{E}[\delta_{abc}^2 + \delta_{ubc}^2] \\
&= \mathbb{E}[(\textstyle\sum_k \lambda_k (q_k(\theta_a) - q_k(\theta_u)) q_k(\theta_b) q_k(\theta_c))^2 \mid \theta_a, \theta_u] + 2\sigma^2 \\
&= \mathbb{E}[\textstyle\sum_k \lambda_k^2 (q_k(\theta_a) - q_k(\theta_u))^2 q_k(\theta_b)^2 q_k(\theta_c)^2 \mid \theta_a, \theta_u] + 2\sigma^2 \\
&= \textstyle\sum_k \lambda_k^2 (q_k(\theta_a) - q_k(\theta_u))^2 + \sigma^2 \\
&= \|\Lambda Q(e_a - e_u)\|_2^2 + 2\sigma^2,
\end{aligned}$$

where we use the fact that $q_k(\cdot)$ are orthonormal. This suggests approximating $\texttt{dist}(u,a)$ with the average squared difference between datapoints corresponding to pairs (b,c) for which both (u,b,c) and (a,b,c) are observed.

This method does not attain the $p = n^{-3/2}$ sample complexity, as the expected number of pairs (b,c) for which (a,b,c) and (u,b,c) are both observed will go to zero for $p = o(n^{-1})$. This limitation arises due to the fact that when $p = o(n^{-1})$, the observations are extremely sparse. Consider the $n \times \binom{n}{2}$ "flattened" matrix of the tensor where row u correspond to coordinates $u \in [n]$, and columns correspond to pairs of indices, e.g. $(b,c) \in [n] \times [n]$ with $b \leq c$. For any given row u, there are very few other rows that share observations along any column with the given row u, i.e. the number of 'neighbors' of any row index is few. If we wanted to exploit the intuition of the above simple calculations, we have to somehow enrich the neighborhood. We do so by constructing a graph using the non-zero pattern of the matrix as an adjacency matrix. This mirrors the idea from [17] for matrix estimation, which approximates distances by comparing expanded local neighborhoods in the graph representing the sparsity pattern of the unfolded or flattened tensor. The precise algorithm is described in the next section formally.

VI. Formal Algorithm

We will unfold the tensor to a matrix and use iterative sparse collaborative filtering from the matrix estimation setting [17].

Step 1: Sample Splitting

Let us assume for simplicity of the analysis that we obtain 3 independent fresh observation sets of the data, $\Omega_1, \Omega_2, \Omega_3$. Matrices M_1, M_2, and M_3 contain information from the subset of the data in M associated to Ω_1, Ω_2, and Ω_3 respectively. M_1 is used to define local neighborhoods of each vertex, M_2 is used to compute similarities of these neighborhoods, and M_3 is used to average over datapoints for the final estimate.

Step 2: Construct Bipartite Graph from Ω_1, M_1

Let $\mathcal{V}$ denote the set of coordinate pairs consisting of distinct coordinates, i.e. $\mathcal{V} = \{(b,c) \in [n]^2 \; s.t. \; b < c\}$, and $|\mathcal{V}| = \binom{n}{2}$. First we flatten the tensor M_1 to a n-by-$\binom{n}{2}$ matrix, where each row $a \in [n]$ corresponds to an original coordinate of the tensor, and each column corresponds to a pair of coordinates $(b,c) \in \mathcal{V}$ from the original tensor. A row-column pair of the matrix thus corresponds to a triplet of coordinates in the original tensor, and the matrix entry is given the same value as the associated tensor datapoint $M(a,b,c)$.

Next we define a bipartite graph corresponding to the flattened matrix. Construct a graph with vertex set $[n] \cup \mathcal{V}$. There is an edge between vertex $a \in [n]$ and vertex $(b,c) \in \mathcal{V}$ if $(a,b,c) \in \Omega_1$, and the corresponding weight of the edge is $M(a,b,c)$. Recall that we assumed a symmetric model such that triplets that are permutations of one another will have the same data entry and thus the same edge weight in the associated graph. Figure 1(a) provides a concrete example of a bipartite graph constructed from tensor observations.

Step 3: Expanding the Neighborhood

Consider the graph constructed from Ω_1, M_1. For each vertex $u \in [n]$, we construct a breadth first (BFS) tree rooted at vertex u such that the vertices along the BFS tree consist of strictly distinct coordinates, i.e. if vertex $a \in [n]$ is visited in an earlier layer of the BFS tree, then no vertex corresponding to (a,b) for any $b \in [n]$ can be visited subsequently. Similarly, if (a,b) is visited in the BFS tree, then no subsequent vertices including either coordinates a or b can be visited.

There may be multiple valid BFS trees due to different ordering of visiting edges at the same depth/layer. For example, if a vertex at depth s has edges to two different vertices at depth $s-1$, only one of the edges can be chosen to maintain the tree property, but either choice is equally valid. Similarly, if there are vertices in $\mathcal{V}$ at depth s which share the same coordinates, such as (a,b) and (a,c), then only one of those vertices can be chosen. Either choice is equally valid but would lead to subsequently different trees, as visiting (a,b) would prevent the tree from downstream visiting coordinates a or b but would leave coordinate c available for future vertex visits. Figure 1(c) shows valid BFS trees for a bipartite graph constructed from an example tensor.

The graph is bipartite so that each subsequent layer of the BFS tree alternates between the vertex sets $[n]$ and $\mathcal{V}$. Consider a valid BFS tree rooted at vertex $u \in [n]$ which respects the constraint that no coordinate is visited more than once. We will use $\mathcal{U}_{u,s} \subseteq \mathcal{V}$ to denote the set of vertices at depth $(2s-1)$ of the BFS tree, and we use $\mathcal{S}_{u,s} \subseteq [n]$ to denote the set of vertices at depth $2s$ of the BFS tree. Let $\mathcal{B}_{u,s}$ denote the set of vertices which are visited in the first s layers of the BFS tree, $\mathcal{B}_{u,s} = \cup_{l \in \lceil s/2 \rceil} \mathcal{U}_{u,l} \cup_{h \in \lfloor s/2 \rfloor} \mathcal{S}_{u,h}$. Let $\mathcal{G}(\mathcal{B}_{u,s})$ denote all the information corresponding to the subgraph restricted to the first s layers of the BFS tree rooted at u. This includes the vertex set $\mathcal{B}_{u,s}$, the latent variables $\{\theta_a\}_{a \in \mathcal{B}_{u,s}}$ and the edge weights $\{M_1(a,b,c)\}_{a,(b,c) \in \mathcal{B}_{u,s}}$.

We define neighborhood vectors which represent the different layers of the BFS tree. Let $N_{u,s} \in [0,1]^n$ be associated to set $\mathcal{S}_{u,s}$, where the a-th coordinate is equal to the product of weights along the path from u to a in the BFS tree for $a \in \mathcal{S}_{u,s}$. Similarly, let $W_{u,s} \in [0,1]^{\mathcal{V}}$ be associated to set $\mathcal{U}_{u,s}$, where the (b,c)-th coordinate is equal to the product of weights along the path from u to (b,c) in the BFS tree for

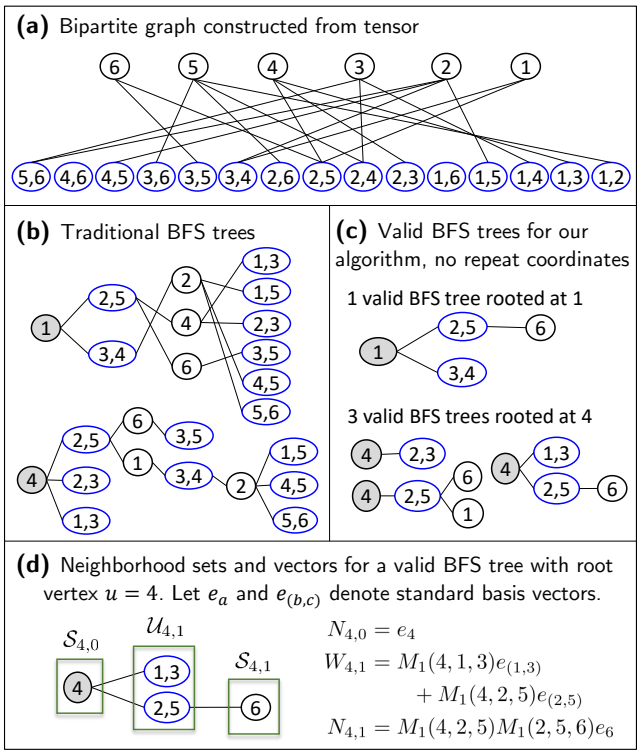

Fig. 1. Consider a symmetric 3-order tensor with $n = 6$, Ω_1 containing $\{(1,3,4)(1,2,5)(2,3,4)(2,4,5)(2,5,6)(3,5,6)\}$. Figure **(a)** depicts the bipartite graph constructed from this set of observations. Weights would be assigned to edges based on the value of the observed entry in the tensor M_1. Figure **(b)** depicts the traditional notion of the BFS tree rooted at vertices 1 and 4. Vertices at layer/depth s correspond to vertices with shortest path distance of s to the root vertex. Figure **(c)** depicts valid BFS trees for our algorithm, which imposes an additional constraint that coordinates cannot be repeated. For the BFS tree rooted at vertex 1, edges $((2,5),4)$ and $((3,4),2)$ are not valid, as coordinates 2 and 4 have both been visited in layer 2 by the vertices $(2,5)$ and $(3,4)$. There are three different valid BFS trees rooted at vertex 4 which resolve conflicting edges with different choices. Edges $(4,(2,5))$ and $(4,(2,3))$ are conflicting as they share the coordinate 2. Edges $(4,(2,3))$ and $(4,(1,3))$ are conflicting as they share the coordinate 3. Figure **(d)** depicts the sets $\mathcal{S}_{u,s}$ and $\mathcal{U}_{u,s}$ along with the neighborhood vectors $N_{u,s}$ and $W_{u,s}$ for a specific valid BFS tree rooted at vertex $u = 4$.

$(b,c) \in \mathcal{U}_{u,s}$. For $a \in [n]$, let $\pi_u(a)$ denote the parent of a in the valid BFS tree rooted at vertex u. For $(b,c) \in \mathcal{V}$, let $\pi_u(b,c)$ denote the parent of (b,c) in the BFS tree rooted at vertex u. We can define the neighborhood vectors recursively,

$$N_{u,s}(a) = M_1(a, \pi_u(a))W_{u,s}(\pi_u(a))\mathbb{I}(a \in \mathcal{S}_{u,s})$$
$$W_{u,s}(b,c) = M_1(\pi_u(b,c), b, c)N_{u,s-1}(\pi_u(b,c))\mathbb{I}_{((b,c)\in\mathcal{U}_{u,s})}$$

and $N_{u,0} = e_u$. Let $\tilde{N}_{u,r}$ denote the normalized vector $\tilde{N}_{u,r} = N_{u,r}/|\mathcal{S}_{u,r}|$ and let $\tilde{W}_{u,r}$ denote the normalized vector $\tilde{W}_{u,r} = W_{u,r}/|\mathcal{U}_{u,r}|$. Figure 1(d) illustrates the neighborhood sets and vectors for a valid BFS tree.

Step 4: Computing the distances

Let us choose neighborhood depth t proportional to $\frac{\ln(1/(pn))}{\ln(p^2n^3)}$. Compute $\texttt{dist}(u,v)$ according to

$$\texttt{dist}(u,v) = \tfrac{1}{p}\big(\tilde{N}_{u,t} - \tilde{N}_{v,t}\big)^T M_2\big(\tilde{W}_{u,t+1} - \tilde{W}_{v,t+1}\big).$$

Step 5: Averaging datapoints to produce final estimate

Let $\mathcal{E}_{uvw}$ denote the set of indices (a,b,c) such that $a \le b \le c$, $(a,b,c) \in \Omega_3$, and the estimated distances $\texttt{dist}(u,a)$, $\texttt{dist}(v,b)$, $\texttt{dist}(w,c)$ are all less than some chosen threshold parameter ξ. The final estimate averages the datapoints corresponding to indices in $\mathcal{E}_{uvw}$,

$$\hat{F}(u,v,w) = \tfrac{1}{|\mathcal{E}_{uvw}|}\textstyle\sum_{(a,b,c)\in\mathcal{E}_{uvw}} M_3(a,b,c).$$

VII. Theoretical Guarantees

We provide an upper bound on the MSE of the algorithm, which shows that the MSE converges to zero as long as $p = n^{-3/2+\epsilon}$ for some $\epsilon > 0$. We assume $\frac{1-2\epsilon}{4\epsilon}$ is not integral.

Theorem VII.1. *We assume that the function f is L-Lipschitz with respect to θ, and that $\theta \sim U[0,1]$. Assume that $p = n^{-3/2+\epsilon}$ for some $\epsilon \in (0, \frac{1}{2})$. Choose t to be the unique integer between*

$$t \in \left(\tfrac{1-2\epsilon}{4\epsilon}, \tfrac{1+2\epsilon}{4\epsilon}\right).$$

For any arbitrarily small $\rho > 0$, choose the threshold

$$\xi = \Theta\left(\left(\tfrac{n^{\max(\rho, 4\epsilon t-1)}}{pn(p^2n^3)^t}\right)^{1/2}\right).$$

The algorithm produces estimates so that,

$$MSE = \begin{cases} O\left(n^{-\frac{\epsilon}{2}+\frac{(1-4\epsilon t)}{4}+\frac{\rho}{2}}\right) & \text{if } 4\epsilon t - 1 \le \rho \\ O\left(n^{-\frac{\epsilon}{2}+\frac{(4\epsilon t-1)}{4}}\right) & \text{if } 4\epsilon t - 1 > \rho \end{cases}.$$

The computational complexity is bounded above by $O(pn^6)$.

As long as $\frac{1-2\epsilon}{4\epsilon}$ is not integral, then this result implies that our simple collaborative filtering algorithm based on estimating similarities produces a consistent estimator with MSE converging to zero with n.

The computational complexity can be estimated by analyzing steps 3-5 of the algorithm. Recall that each coordinate is visited at most one time in the process of building any valid BFS tree. Step 3 costs $O(n^2)$, as there are n trees to construct which each take a maximum of n edge traversals. Step 4 costs $O(pn^4)$ as the vectors $M_{u,t+1}$ and $N_{u,t}$ are at most n-sparse, the density of matrix M_2 is p, and there are $\binom{n}{2}$ pairs of coordinates for which we need to compute distances. Step 5 costs $O(pn^6)$ as there are $\Theta(n^3)$ triplets we need to estimate, and each involves averaging at most $O(pn^3)$ datapoints.

Although our stated results assume a symmetric tensor, the results naturally extend to asymmetric $(n_1 \times n_2 \times n_3)$ tensors as long as n_1, n_2, and n_3 are proportional to one another. Our analysis can be modified for the asymmetric setting, or one can reduce the asymmetric tensor to a $(n \times n \times n)$ symmetric tensor where $n = n_1 + n_2 + n_3$, and the coordinates of the new tensor consists of the union of the coordinates in all three dimensions of the asymmetric tensor. The results applied to this larger tensor would still hold with adjustments of the model allowing for piecewise Lipschitz functions.

In the proof sketch that follows below, we show that for the 3-order tensor, the sample complexity threshold of $p = \omega(n^{-3/2})$ directly equals the density of observations

needed to guarantee the bipartite graph is connected with high probability. Although our stated results assume a 3-order tensor, we conjecture that our algorithm and analysis can be extended to general d-order tensors. A naive calculation seems to imply that $\omega(n^{d/2})$ is the connectivity threshold for the graph corresponding to a flattening of a d-order tensor.

VIII. Proof Sketch

The main challenge of the proof is to show that with high probability,

$$\left|\texttt{dist}(u,a) - \|\Lambda^{2t+1}Q(e_a - e_u)\|_2^2\right| = o(1).$$

This would imply that if $\texttt{dist}(u,a)$ is small, then $Qe_a \approx Qe_u$. In particular, if $\|\Lambda^{2t+1}Q(e_a - e_u)\|_2^2 \leq \eta$, then

$$|f(\theta_u,\theta_b,\theta_c) - f(\theta_a,\theta_b,\theta_c)| \leq \left(\frac{B^2\sqrt{r}}{|\lambda_d|^{2t+\frac{1}{2}}}\right)\eta^{1/2}.$$

We use the conditions of Lipschitzness and the latent variable model to lower bound the fraction of vertices a that are "close" to u measured according to $\|\Lambda^{2t+1}Q(e_a - e_u)\|_2^2$. The final MSE calculation then results from a bias and variance tradeoff.

In order to show that $\texttt{dist}(u,a)$ concentrates, we argue that $e_k^T\mathcal{Q}\tilde{W}_{u,s}$ conditioned on $\mathcal{U}_{u,s}$ and all previous layers of the BFS tree (denoted $\mathcal{G}(\mathcal{B}_{u,2(s-1)})$), is a sum of $|\mathcal{U}_{u,s}|$ iid random variables with mean $e_k^T\Lambda Q\tilde{N}_{u,s-1}$. Similarly we also show that $e_k^TQ\tilde{N}_{u,s}$ conditioned on $\mathcal{S}_{u,s}$ and all previous layers of the BFS tree (denoted $\mathcal{G}(\mathcal{B}_{u,2s-1})$), is a sum of $|\mathcal{S}_{u,s}|$ iid random variables with mean $e_k^T\Lambda\mathcal{Q}\tilde{W}_{u,s}$. The two of these results together indicate that

$$e_k^TQ\tilde{N}_{u,t} \approx e_k^T\Lambda^{2t}Qe_u \text{ and } e_k^T\mathcal{Q}\tilde{W}_{u,t+1} \approx e_k^T\Lambda^{2t+1}Qe_u.$$

Conditioned on M_1, then $\texttt{dist}(u,a)$ can be written as a sum of approximately $p|\mathcal{U}_{u,t+1}||\mathcal{S}_{u,t}|$ random variables.

In order for these expressions to concentrate around their mean, we need the observations to be sufficiently dense so that $|\mathcal{U}_{u,s}| = \omega(1)$ for $s \in [t+1]$, $|\mathcal{S}_{u,s}| = \omega(1)$ for $s \in [t]$, and $p|\mathcal{U}_{u,t+1}||\mathcal{S}_{u,t}| = \omega(1)$. If we ignore the constraints that coordinates are not allowed to be repeated in valid BFS trees, we can roughly estimate the expected sizes of these neighborhoods, $\mathbb{E}[|\mathcal{S}_{u,s}| \mid \mathcal{G}(\mathcal{B}_{u,2s-1})] \approx pn|\mathcal{U}_{u,s}|$ and $\mathbb{E}[|\mathcal{U}_{u,s}| \mid \mathcal{G}(\mathcal{B}_{u,2(s-1)})] \approx \frac{pn^2}{2}|\mathcal{S}_{u,s-1}|$. This implies that

$$\mathbb{E}[|\mathcal{U}_{u,s}|] \approx \frac{pn^2}{2}\left(\frac{p^2n^3}{2}\right)^{s-1} \text{ and } \mathbb{E}[|\mathcal{S}_{u,s}|] \approx \left(\frac{p^2n^3}{2}\right)^{s},$$

so that

$$p|\mathcal{U}_{u,t+1}||\mathcal{S}_{u,t}| \approx \frac{(pn)^2}{2}\left(\frac{p^2n^3}{2}\right)^{2t}.$$

In order to guarantee that these quantities are sufficiently large to prove concentration, the density of observations must be $p = \omega(n^{-3/2})$, and the depth t must satisfy $t > \frac{\ln(1/(pn))}{\ln(p^2n^3)}$. This argument illustrates that the conjectured $n^{3/2}$ computational sample complexity lower bound directly matches the threshold of observation density that would guarantee connectivity in the graph corresponding to the flattened tensor.

References

[1] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from a few entries," *IEEE Transactions on Information Theory*, vol. 56, no. 6, pp. 2980–2998, 2010.

[2] ——, "Matrix completion from noisy entries," *Journal of Machine Learning Research*, vol. 11, no. Jul, pp. 2057–2078, 2010.

[3] S. Chatterjee, "Matrix estimation by universal singular value thresholding," *The Annals of Statistics*, vol. 43, no. 1, pp. 177–214, 2015.

[4] E. Candes and B. Recht, "Exact matrix completion via convex optimization," *Communications of the ACM*, vol. 55, no. 6, 2009.

[5] E. J. Candes and Y. Plan, "Matrix completion with noise," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 925–936, 2010.

[6] E. J. Candès and T. Tao, "The power of convex relaxation: Near-optimal matrix completion," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2053–2080, 2010.

[7] B. Recht, "A simpler approach to matrix completion," *Journal of Machine Learning Research*, vol. 12, no. Dec, pp. 3413–3430, 2011.

[8] S. Negahban and M. J. Wainwright, "Estimation of (near) low-rank matrices with noise and high-dimensional scaling," *The Annals of Statistics*, pp. 1069–1097, 2011.

[9] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *Journal of machine learning research*, vol. 11, no. Aug, pp. 2287–2322, 2010.

[10] Y. Chen and M. J. Wainwright, "Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees," *arXiv preprint arXiv:1509.03025*, 2015.

[11] R. Sun and Z.-Q. Luo, "Guaranteed matrix completion via non-convex factorization," *IEEE Transactions on Information Theory*, vol. 62, no. 11, pp. 6535–6579, 2016.

[12] R. Ge, J. D. Lee, and T. Ma, "Matrix completion has no spurious local minimum," in *Advances in Neural Information Processing Systems*, 2016, pp. 2973–2981.

[13] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM, 2013, pp. 665–674.

[14] M. Hardt, "Understanding alternating minimization for matrix completion," in *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*. IEEE, 2014, pp. 651–660.

[15] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Commun. ACM*, 1992.

[16] D. Song, C. E. Lee, Y. Li, and D. Shah, "Blind regression: Nonparametric regression for latent variable models via collaborative filtering," in *Advances in Neural Information Processing Systems*, 2016.

[17] C. Borgs, J. Chayes, C. E. Lee, and D. Shah, "Thy friend is my friend: Iterative collaborative filtering for sparse matrix estimation," in *Advances in Neural Information Processing Systems*, 2017, pp. 4715–4726.

[18] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 208–220, 2013.

[19] S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, vol. 27, no. 2, p. 025010, 2011.

[20] R. Tomioka, K. Hayashi, and H. Kashima, "Estimation of low-rank tensors via convex optimization," *arXiv preprint arXiv:1010.0789*, 2010.

[21] R. Tomioka, T. Suzuki, K. Hayashi, and H. Kashima, "Statistical performance of convex tensor decomposition," in *Advances in Neural Information Processing Systems*, 2011, pp. 972–980.

[22] P. Jain and S. Oh, "Provable tensor factorization with missing data," in *Advances in Neural Information Processing Systems*, 2014.

[23] S. Bhojanapalli and S. Sanghavi, "A new sampling technique for tensors," *arXiv preprint arXiv:1502.05023*, 2015.

[24] M. Yuan and C.-H. Zhang, "On tensor completion via nuclear norm minimization," *Foundations of Computational Mathematics*, vol. 16, no. 4, pp. 1031–1068, 2016.

[25] B. Barak and A. Moitra, "Noisy tensor completion via the sum-of-squares hierarchy," in *Conference on Learning Theory*, 2016.

[26] A. Potechin and D. Steurer, "Exact tensor completion with sum-of-squares," *arXiv preprint arXiv:1702.06237*, 2017.

[27] A. Montanari and N. Sun, "Spectral algorithms for tensor completion," *Communications on Pure and Applied Mathematics*, vol. 71, no. 11, pp. 2381–2425, 2018.

[28] S. Friedland and L.-H. Lim, "Nuclear norm of higher-order tensors," *arXiv preprint arXiv:1410.6072*, 2014.