Stochastic Iterative Hard Thresholding for Low-Tucker-Rank Tensor Recovery

Rachel Grotheer*, Shuang Li†, Anna Ma‡, Deanna Needell§, and Jing Qin¶
*Goucher College, Baltimore, MD 21204, rachel.grotheer@goucher.edu
†Department of Electrical Engineering, Colorado School of Mines, Golden, CO 80401, shuangli@mines.edu
‡Department of Mathematics, University of California, Irvine, CA 92697, anna.ma@uci.edu
§Department of Mathematics, University of California, Los Angeles, CA 90095, deanna@math.ucla.edu
¶Department of Mathematics, University of Kentucky, Lexington, KY 40502, jing.qin@uky.edu

***Abstract*—Low-rank tensor recovery problems have been widely studied in many signal processing and machine learning applications. Tensor rank is typically defined under certain tensor decomposition. In particular, Tucker decomposition is known as one of the most popular tensor decompositions. In recent years, researchers have developed many state-of-the-art algorithms to address the problem of low-Tucker-rank tensor recovery. Motivated by the favorable properties of the stochastic algorithms, such as stochastic gradient descent and stochastic iterative hard thresholding, we aim to extend the stochastic iterative hard thresholding algorithm from vectors to tensors in order to address the problem of recovering a low-Tucker-rank tensor from its linear measurements. We have also developed linear convergence analysis for the proposed method and conducted a series of experiments with both synthetic and real data to illustrate the performance of the proposed method.**

I. Introduction

Tensors are high-dimensional extensions of vectors and matrices. There are many different kinds of tensor decompositions, among which, the Canonical Polyadic (CP) decomposition and Tucker decomposition are the most popular [1], [2]. In recent years, low-rank tensor recovery problems have gained a great amount of attention in various applications including hyperspectral image restoration [3], video processing [4], signal processing [5], [6], and simultaneous blind deconvolution and phase retrieval [7]. Unlike low-rank matrix recovery problems, which often use nuclear norm minimization as a popular heuristic for rank minimization, the computation of the nuclear norm for high order tensors is NP-hard [8], [9].

Over the decades, the iterative hard thresholding (IHT) algorithm has been widely used in compressive sensing [10], [11], [12] and low-rank matrix recovery [13], [14], [15]. It has many extensions, such as the stochastic variant proposed in [16], which was further extended to the multiple measurement vector framework in [17]. Inspired by the idea of using the IHT algorithm in low-rank matrix recovery problems, the authors in [18] extended the IHT algorithm to the tensor framework and proposed the Tensor IHT (TIHT) algorithm as an alternative to the tensor nuclear norm minimization. The authors of [7] then combined this TIHT algorithm with higher-order singular value decomposition (HOSVD), a type of Tucker decomposition, to solve a low-rank tensor recovery problem formulated from a simultaneous blind deconvolution and phase retrieval problem. Another recent work [19] also extends the IHT algorithm to the problem of low-rank tensor recovery based on a low-Tucker-rank approximation technique named sequentially optimal modal projections.

The stochastic versions of gradient descent algorithms and IHT algorithms usually have many favorable properties. For example, these algorithms do not need to compute the full gradient, which makes it possible for them to be utilized in large scale problems where computing the full gradient is very expensive. These properties inspired us to extend stochastic IHT to the tensor framework and introduce the Stochastic Tensor IHT (StoTIHT) algorithm to recover a low-Tucker-rank tensor from its linear measurements. In this work, we provide convergence analysis for the proposed StoTIHT algorithm, based on a Tucker decomposition of the tensor, under the assumption that the linear operator used to obtain the measurements satisfies a tensor restricted isometry property (TRIP). Our simulations also indicate that the proposed StoTIHT algorithm converges much faster than the original TIHT algorithm in a large scale setting.

The remainder of this work is organized as follows. In Section II, we briefly review some fundamental concepts and definitions used in the tensor framework. We formulate our low-rank tensor recovery problem in Section III and present the proposed StoTIHT algorithm in Section IV. We then introduce the linear convergence analysis for our proposed StoTIHT algorithm in Section V and illustrate its performance with both synthetic and real data in Section VI. Finally, we conclude our work in Section VII.

II. Preliminaries

In this section, we briefly review some fundamental concepts and definitions used in the tensor framework [1], [2], [20]. We denote a d-th order tensor as $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$. Vectors and matrices can be viewed as low-dimensional tensors with $d = 1$ and 2, respectively. Denote $\mathbf{X}^{\{i\}} \in \mathbb{R}^{n_i \times (n_1 n_2 \cdots n_{i-1} n_{i+1} \cdots n_d)}$ as the *mode-i matricization* or the *i-th unfolding* of a tensor $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$. One can refer to [20] for a more detailed definition and some simple examples. Similar to the matrix case, it is possible to convert a tensor to a column vector while the ordering of all elements is not unique. In this work, we stick to the following tensor

vectorization: for any tensor $\mathbf{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$, its vectorized version $\text{vec}(\mathbf{X})$ is obtained by columnwise stacking all entries of the mode-1 matricization $\mathbf{X}^{\{1\}} \in \mathbb{R}^{n_1 \times (n_2 n_3 \cdots n_d)}$. The *inner product* of two tensors $\mathbf{X}_1, \mathbf{X}_2 \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ is then defined as

$$\langle \mathbf{X}_1, \mathbf{X}_2 \rangle \triangleq \text{vec}(\mathbf{X}_2)^\top \text{vec}(\mathbf{X}_1).$$

Based on the tensor inner product, the induced *Frobenius norm* is defined as

$$\|\mathbf{X}\|_F \triangleq \sqrt{\langle \mathbf{X}, \mathbf{X} \rangle}.$$

The Tucker decomposition is one common tensor decomposition and one can find more details in [21], [22]. As one type of Tucker decompositions, HOSVD decomposes a tensor $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ as follows

$$\mathbf{X} = \mathbf{S} \times_1 \mathbf{U}^{(1)} \cdots \times_d \mathbf{U}^{(d)}. \tag{II.1}$$

Here, $\mathbf{S} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ and $\mathbf{U}^{(i)} \in \mathbb{R}^{n_i \times r_i}$ denote the core tensor and the basis, respectively. One can refer to [1], [2] for more details about the properties of the core tensor and basis. The product $\times_i$ is the *mode-i (matrix) product* of the tensor, that is, the product of a tensor and a matrix along the i-th mode of the tensor. The *Tucker rank* of tensor $\mathbf{X}$ is defined as a tuple $\boldsymbol{r} = (r_1, \cdots, r_d)$ with $r_i = \text{rank}(\mathbf{X}^{\{i\}})$.

III. Problem Formulation

In this work, we aim to recover a rank-$\boldsymbol{r}$ tensor $\mathbf{X}^\star \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ from its linear measurements $\boldsymbol{y} = \mathcal{A}(\mathbf{X}^\star) \in \mathbb{R}^m$, where $\mathcal{A} : \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d} \to \mathbb{R}^m$ is a linear operator used to generate the measurements. More specifically, the i-th element of $\boldsymbol{y}$ is given as

$$\boldsymbol{y}(i) = \mathcal{A}_i(\mathbf{X}^\star) = \langle \mathbf{A}_i, \mathbf{X}^\star \rangle, \quad i = 1, \ldots, m, \tag{III.1}$$

where $\mathbf{A}_i \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_d}$ is a sensing tensor. We define the cost function $F(\mathbf{X})$ to be

$$\begin{aligned} F(\mathbf{X}) &\triangleq \frac{1}{2m} \|\boldsymbol{y} - \mathcal{A}(\mathbf{X})\|_2^2 = \frac{1}{2m} \sum_{i=1}^{m} (\boldsymbol{y}(i) - \langle \mathbf{A}_i, \mathbf{X} \rangle)^2 \\ &= \frac{1}{M} \sum_{i=1}^{M} \left(\frac{1}{2b} \sum_{j=(i-1)b+1}^{ib} (\boldsymbol{y}(j) - \langle \mathbf{A}_j, \mathbf{X} \rangle)^2 \right) \\ &= \frac{1}{M} \sum_{i=1}^{M} \frac{1}{2b} \|\boldsymbol{y}_{b_i} - \mathcal{A}_{b_i}(\mathbf{X})\|_2^2 \triangleq \frac{1}{M} \sum_{i=1}^{M} f_i(\mathbf{X}). \end{aligned} \tag{III.2}$$

Here the measurement vector $\boldsymbol{y} \in \mathbb{R}^m$ can be decomposed into M non-overlapping vectors $\boldsymbol{y}_{b_i} \in \mathbb{R}^b$, $i = 1, \ldots, M$. Note that b is an integer and $M = \lceil m/b \rceil$. We denote $\mathcal{A}_{b_i} : \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d} \to \mathbb{R}^b$ as a linear operator with the j-th entry of $\mathcal{A}_{b_i}(\mathbf{X})$ being $\langle \mathbf{A}_{(i-1)b+j}, \mathbf{X} \rangle$, $j = 1, \ldots b$. It can be seen that each function $f_i(\mathbf{X})$ is associated with a collection of measurements $\boldsymbol{y}_{b_i}$.

Due to the low-rank structure, $\mathbf{X}^\star$ can be recovered by solving the following rank-restricted minimization problem

$$\underset{\mathbf{X} \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_d}}{\text{minimize}} \; F(\mathbf{X}) \quad \text{subject to} \quad \text{rank}(\mathbf{X}) \leq \boldsymbol{r}, \tag{III.3}$$

where the cost function $F(\mathbf{X})$ is defined in (III.2). Many methods have been proposed to solve the above problem in existing literature. For instance, [23], [24] relax the problem by minimizing the sum of the nuclear norm of the tensor matricizations. However, this kind of convex relaxation is not optimal [25]. Inspired by the IHT algorithm for solving compressive sensing and low-rank matrix recovery problems, [18] extends the IHT algorithm [10] to the tensor framework and proposes the TIHT algorithm. In particular, to recover $\mathbf{X}^\star$, the TIHT algorithm consists of the following two steps at the t-th iteration:

$$\widetilde{\mathbf{X}}^t = \mathbf{X}^t + \mu \mathcal{A}^*(\boldsymbol{y} - \mathcal{A}(\mathbf{X}^t)), \tag{III.4}$$

$$\mathbf{X}^{t+1} = \mathcal{H}_{\boldsymbol{r}}(\widetilde{\mathbf{X}}^t). \tag{III.5}$$

Here, μ is the stepsize and $\mathcal{A}^* : \mathbb{R}^m \to \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_d}$ is the adjoint operator of $\mathcal{A}$. That is, for tensor $\mathbf{X}$ and vector $\boldsymbol{y}$, $\langle \mathcal{A}(\mathbf{X}), \boldsymbol{y} \rangle = \langle \mathbf{X}, \mathcal{A}^*(\boldsymbol{y}) \rangle$. The operator $\mathcal{H}_{\boldsymbol{r}}(\mathbf{X})$ computes a best rank-$\boldsymbol{r}$ approximation of a tensor $\mathbf{X}$ using HOSVD. Note that the second step (III.5) is not straightforward, and the assumption

$$\|\mathcal{H}_{\boldsymbol{r}}(\widetilde{\mathbf{X}}^t) - \widetilde{\mathbf{X}}^t\|_F \leq \eta \|\widetilde{\mathbf{X}}^t_{\text{best}} - \widetilde{\mathbf{X}}^t\|_F \tag{III.6}$$

is held for all $t = 1, 2, \ldots, T$ with some $\eta \in [1, \infty)$ in [18]. Here $\widetilde{\mathbf{X}}^t_{\text{best}}$ is the best rank-$\boldsymbol{r}$ approximation of $\widetilde{\mathbf{X}}^t$ with respect to the Tucker decomposition (given by the HOSVD), namely, $\widetilde{\mathbf{X}}^t_{\text{best}} = \arg\min_{\text{rank}(\mathbf{X}) \leq \boldsymbol{r}} \|\widetilde{\mathbf{X}}^t - \mathbf{X}\|_F$. We assume such an approximation exists in our convergence analysis.

IV. Proposed Algorithm

Assume that the linear measurements $\boldsymbol{y}$ can be rewritten as

$$\boldsymbol{y} = \mathbf{A}\boldsymbol{x}^\star,$$

where $\mathbf{A} \in \mathbb{R}^{m \times n_1 n_2 \cdots n_d}$ is a matrix with the i-th row being the vectorized version of $\mathbf{A}_i$, and $\boldsymbol{x}^\star$ is the vectorized version of $\mathbf{X}^\star$. Then, we can update $\widetilde{\mathbf{X}}^t$ in (III.4) with

$$\widetilde{\boldsymbol{x}}^t = \boldsymbol{x}^t + \mu \mathbf{A}^\top (\boldsymbol{y} - \mathbf{A}\boldsymbol{x}^t), \tag{IV.1}$$

where $\widetilde{\boldsymbol{x}}$ is the vectorized version of $\widetilde{\mathbf{X}}$.

It is well known that stochastic gradient descent and its variants do not require computation of the full gradient and, thus can be much more efficient in large scale settings especially when the computation and/or storage of the full gradient is very expensive. Thus, we propose a stochastic variant of the TIHT algorithm, termed as Stochastic TIHT (StoTIHT), which replaces (IV.1) with

$$\widetilde{\boldsymbol{x}}^t = \boldsymbol{x}^t + \frac{\mu}{M p(i_t)} \mathbf{A}(i_t, :)^\top (\boldsymbol{y}_{b_{i_t}} - \mathbf{A}(i_t, :)\boldsymbol{x}^t), , \tag{IV.2}$$

where i_t is an index randomly selected from $[M] = \{1, 2, \cdots, M\}$ with probability $p(i_t)$, and $\mathbf{A}(i_t, :) \in$

$\mathbb{R}^{b\times n_1 n_2\cdots n_d}$ denotes the i_t-th block of $\mathbf{A}$. This updating step is equivalent to

$$\begin{aligned}\widetilde{\mathbf{X}}^t &= \mathbf{X}^t + \frac{\mu}{Mp(i_t)}\frac{1}{b}\sum_{j=(i_t-1)b+1}^{i_t b}\mathbf{A}_j(\boldsymbol{y}(j)-\langle\mathbf{A}_j,\mathbf{X}^t\rangle)\\ &= \mathbf{X}^t - \frac{\mu}{Mp(i_t)}\nabla f_{i_t}(\mathbf{X}^t)\end{aligned} \tag{IV.3}$$

with $f_{i_t}(\mathbf{X}^t) \triangleq \frac{1}{2b}\sum_{j=(i_t-1)b+1}^{i_t b}(\boldsymbol{y}(j)-\langle\mathbf{A}_j,\mathbf{X}\rangle)^2$ as in (III.2). Based on the above analysis, we summarize the proposed algorithm in Algorithm 1.

Algorithm 1 Stochastic Tensor Iterative Hard Thresholding (StoTIHT)

1: **Input:** $\boldsymbol{r}$, μ, and $p(i)$.
2: **Output:** $\widehat{\mathbf{X}} = \mathbf{X}^T$.
3: **Initialize:** $\mathbf{X}^0 = \mathbf{0}$
4: **for** $t = 0, 1, \ldots, T-1$ **do**
5: Randomly select a batch index $i_t \in [M]$ with probability $p(i_t)$
6: Compute the gradient $\nabla f_{i_t}(\mathbf{X}^t)$ as given in (IV.3)
7: $\widetilde{\mathbf{X}}^t = \mathbf{X}^t - \frac{\mu}{Mp(i_t)}\nabla f_{i_t}(\mathbf{X}^t)$
8: $\mathbf{X}^{t+1} = \mathcal{H}_{\boldsymbol{r}}(\widetilde{\mathbf{X}}^t)$
9: If the stopping criteria are met, exit.
10: **end for**

V. Convergence Analysis

In this section, we discuss the convergence of the proposed StoTIHT algorithm. In what follows, we first introduce the tensor restricted isometry property (TRIP).

Definition 1. *(TRIP) [18] Let $\mathcal{A} : \mathbb{R}^{n_1\times n_2\times\cdots\times n_d} \to \mathbb{R}^m$ and $\mathcal{A}_{b_i} : \mathbb{R}^{n_1\times n_2\times\cdots\times n_d} \to \mathbb{R}^b$ be the two linear operators defined in Section III. For a fixed tensor Tucker decomposition and a corresponding Tucker rank $\boldsymbol{r}$, we say $\mathcal{A}$ and $\mathcal{A}_{b_i}$ satisfy the TRIP if there exists a tensor restricted isometry constant $\delta_{\boldsymbol{r}}$ such that*

$$\frac{1}{m}\|\mathcal{A}(\mathbf{X})\|_2^2 \geq (1-\delta_{\boldsymbol{r}})\|\mathbf{X}\|_F^2 \tag{V.1}$$

$$\frac{1}{b}\|\mathcal{A}_{b_i}(\mathbf{X})\|_2^2 \leq (1+\delta_{\boldsymbol{r}})\|\mathbf{X}\|_F^2 \tag{V.2}$$

hold for all tensors $\mathbf{X} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_d}$ of Tucker-rank at most $\boldsymbol{r}$.

Using TRIP and following the convergence analysis in [16], we obtain the linear convergence of the proposed algorithm.

Theorem 1. *Assume that the operators $\mathcal{A} : \mathbb{R}^{n_1\times n_2\times\cdots\times n_d} \to \mathbb{R}^m$ and $\mathcal{A}_{b_i} : \mathbb{R}^{n_1\times n_2\times\cdots\times n_d} \to \mathbb{R}^b$ used in generating the linear measurements $\boldsymbol{y}$ satisfy the TRIP defined in Definition 1. Let $\mathbf{X}^\star$ be a feasible solution of the optimization problem* (III.3)*, and $\mathbf{X}^0$ the initial tensor. We also assume that the Tucker-rank-$\boldsymbol{r}$ approximation operator $\mathcal{H}_{\boldsymbol{r}}(\cdot)$ satisfies* (III.6) *for all $t = 0, 1, \ldots, T-1$ with some $\eta \in [1, \infty)$. Then at the t-th iteration of Algorithm 1, the expectation of the recovery error is bounded by*

$$\mathbb{E}_{I_t}\|\mathbf{X}^{t+1}-\mathbf{X}^\star\|_F \leq \kappa^{t+1}\|\mathbf{X}^0-\mathbf{X}^\star\|_F + \sigma_{\mathbf{X}^\star},$$

where $I_t = \{i_1, i_2, \ldots, i_t\}$ is the set containing all indices $i_1, i_2, \ldots, i_t$ randomly selected at and before iteration t. Here κ and $\sigma_{\mathbf{X}^\star}$ are the contraction coefficient and tolerance parameter, which are defined as

$$\begin{aligned}\kappa \triangleq& 2\sqrt{1-(2-\mu\alpha_{3\boldsymbol{r}})\mu\rho^-_{3\boldsymbol{r}}}\\ &+\sqrt{\eta^2-1}\sqrt{1+\mu^2\alpha_{3\boldsymbol{r}}\rho^+_{3\boldsymbol{r}}-2\mu\rho^-_{3\boldsymbol{r}}}\\ \sigma_{\mathbf{X}^\star} \triangleq& \frac{\mu}{M\min_{i\in[M]}p(i)}\Big(2\mathbb{E}_{i_t}\left\|\mathcal{P}_{U^t}(\nabla f_{i_t}(\mathbf{X}^\star))\right\|_F\\ &+\sqrt{\eta^2-1}\mathbb{E}_{i_t}\left\|\nabla f_{i_t}(\mathbf{X}^\star)\right\|_F\Big)\end{aligned}$$

with $\rho^+_{\boldsymbol{r}} \triangleq 2(1+\delta_{\boldsymbol{r}})$, $\rho^-_{\boldsymbol{r}} \triangleq 1-\delta_{\boldsymbol{r}}$, $\alpha_{\boldsymbol{r}} \triangleq \max_i \frac{\rho^+_{\boldsymbol{r}}}{Mp(i)}$, and i_t being an index selected from $[M]$ with probability $p(i_t)$. Here, U^t is defined as a subspace of $\mathbb{R}^{n_1\times n_2\times\ldots\times n_d}$ spanned by $\mathbf{X}^\star$, $\mathbf{X}^t$ and $\mathbf{X}^{t+1}$. Then $\mathcal{P}_{U^t} : \mathbb{R}^{n_1\times n_2\times\ldots\times n_d} \to U^t$ is the orthogonal projection onto U^t.

VI. Numerical Simulations

In this section, we demonstrate the performance of the proposed StoTIHT algorithm by conducting a variety of experiments on both synthetic and real data. To quantitatively evaluate the performance, we use the relative recovery error given as $\frac{\|\mathbf{X}^\star-\widehat{\mathbf{X}}\|_F}{\|\mathbf{X}^\star\|_F}$ and consider the recovery as a success if the relative recovery error is less than 10^{-5}. The experiments are conducted in Matlab R2014a installed on a laptop with an Intel(R) Core(TM) i7-4700MQ CPU @ 2.40GHz and 64G RAM.

In the first experiment, we work on third-order tensors (i.e., $d = 3$) and set the parameters $n_1 = 5$, $n_2 = 5$, and $n_3 = 6$. The true Tucker-rank is $\boldsymbol{r} = (1, 2, 2)$. Note that we choose a relatively small tensor size and rank in this experiment to reduce the problem size and thereby computational time, which are nevertheless sufficient to show how the proposed algorithm performs. With these parameters, we generate a core tensor $\mathbf{S} \in \mathbb{R}^{r_1\times\cdots\times r_d}$ and a basis $\mathbf{U}^{(i)} \in \mathbb{R}^{n_i\times r_i}$ as random Gaussian tensors or equivalently matrices with entries following $\mathcal{N}(0, 1)$. Then we create a low-rank tensor $\mathbf{X}^\star$ as the tensor of interest according to the Tucker decomposition given in (II.1). We set $m = 360$. The sensing tensors $\mathbf{A}_i$ with $i = 1, \ldots, m$ are also generated as random Gaussian tensors with entries satisfying $\mathcal{N}(0, 1)$, but followed by a normalization. In particular, each entry of the sensing tensors is rescaled by a factor of $1/\|\mathbf{A}\|_F$, where $\mathbf{A} \in \mathbb{R}^{m\times(n_1 n_2\cdots n_d)}$ is a matrix with the i-th row being the vectorized version of $\mathbf{A}_i$. Then, we obtain the measurements $\boldsymbol{y} \in \mathbb{R}^m$ according to the measurement model (III.1). For simplicity, we set $p(i) = \frac{1}{M}$ with $i = 1, \ldots, M$. Here, $M = \frac{m}{b}$ is the number of batches and we experiment with different batch sizes b, as shown in Figures 1 and 2. Note that in the case when $b = m$, the proposed StoTIHT algorithm reduces to the regular TIHT

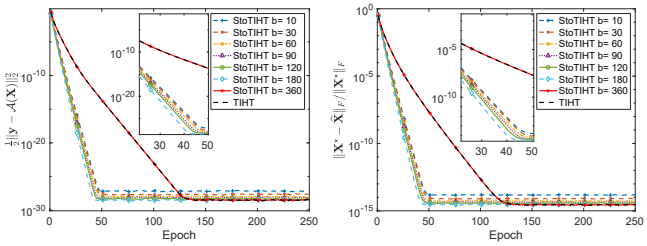
algorithm (black dashed line in Figures 1 and 2). We set the stepsize $\mu = 0.46m$ and use $\widehat{\mathbf{X}}$ to denote the recovered low-rank tensor. For the sake of analysis, we adopt the epoch, which is defined as the number of iterations needed to use m rows. For deterministic algorithms like TIHT algorithm, an epoch is just one iteration, while for our StoTIHT algorithm, an epoch is m/b iterations. We present how the (a) cost function and (b) relative recovery error behave with respect to the number of epochs in Figure 1 and running time in Figure 2. The presented results are all averaged over 100 trials. It can be seen that the proposed StoTIHT algorithm converges much faster than the original TIHT algorithm in such a high dimensional setting.

Next, we fix the batch size as $b = 0.5m$ and repeat the above experiment with several values of m and $\boldsymbol{r}$ as shown in Figure 3. We allow a maximum of 200 epochs for both the TIHT and StoTIHT algorithms in this experiment. The presented results are all averaged over 100 trials. We set the stepsize $\mu = m$. It can be seen that in this region with small m, the percentage of trials with successful recovery increases as we increase the number of measurements m and decrease the Tucker-rank. Then we repeat this experiment with a large number of measurements. We fix the batch size at $b = 0.25m$ and set the maximum number of epochs at 80. We change the stepsize back to $\mu = 0.5m$. As is shown in Figure 4, in the region with a large m and a small number of epochs, our proposed StoTIHT algorithm always successfully recovers the tensor and significantly outperforms the TIHT algorithm. This observation also coincides with Figure 1, which indicates that the proposed StoTIHT algorithm converges much faster than the TIHT algorithm in a large scale setting. Therefore, we conjecture that the TIHT algorithm needs more epochs to get a successful recovery. To verify this conjecture, we conduct another experiment and present the number of epochs needed to achieve a successful recovery for different r and m in Figure 5. We decrease the stepsize to $\mu = 0.4m$. It can be seen that the proposed StoTIHT algorithm needs fewer epochs to get a perfect recovery especially when m is large.

Finally, we test the proposed StoTIHT algorithm on a real candle video, which can be downloaded from the Dynamic Texture Toolbox in http://www.vision.jhu.edu/code/. To ensure the recovery problem in a relatively small dimension, we truncate the video frames to be of size 30×30 and only keep the first 10 frames. Thus, the tensor to be recovered is of size $30 \times 30 \times 10$. We assume this tensor has Tucker-rank $\boldsymbol{r} = (8, 8, 2)$. Then, we use the same strategy as in the synthetic experiments to obtain the linear measurements with $m = 3 \times 10^4$. We set the batch size as $b = 0.25m$ for the StoTIHT algorithm. The last two true candle frames and the recovered ones are shown in Figure 6. We also present the cost function and relative recovery error in Figure 7.

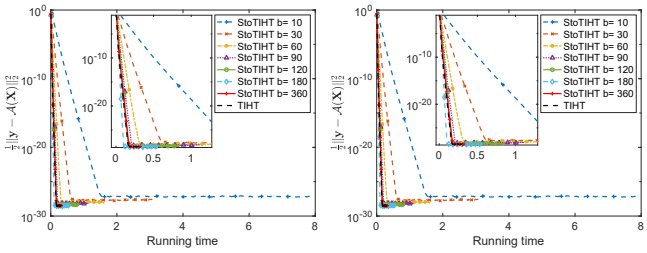
VII. Conclusion

In this paper, we propose the StoTIHT algorithm by combining StoIHT and HOSVD to address the problem of recovering a low-Tucker-rank tensor from its linear measurements. Using



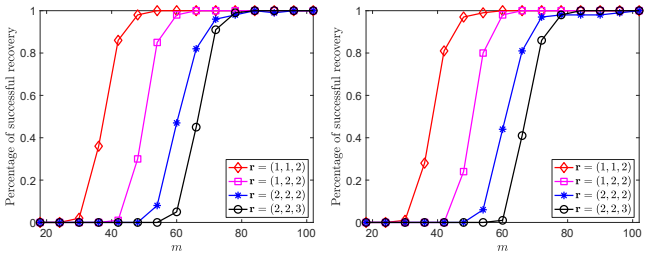
(a) (b)

Fig. 1. Low-rank tensor recovery: $n_1 = 5$, $n_2 = 5$, $n_3 = 6$, $m = 360$, $\boldsymbol{r} = (1, 2, 2)$.



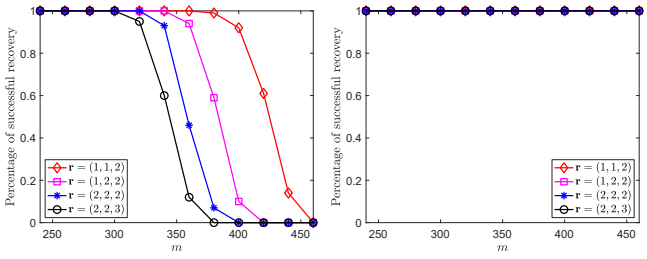
(a) (b)

Fig. 2. Low-rank tensor recovery: $n_1 = 5$, $n_2 = 5$, $n_3 = 6$, $m = 360$, $\boldsymbol{r} = (1, 2, 2)$.



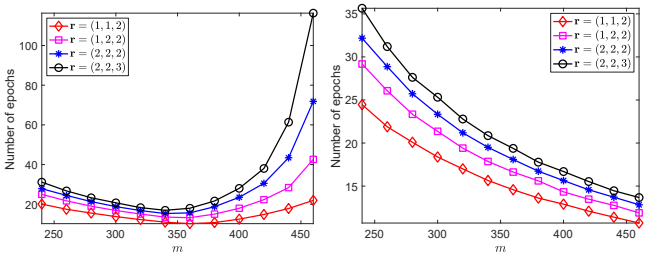
(a) TIHT (b) StoTIHT

Fig. 3. Low-rank tensor recovery: $n_1 = 5$, $n_2 = 5$, $n_3 = 6$, $b = 0.5m$.



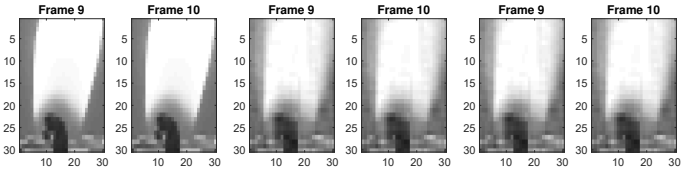
(a) TIHT (b) StoTIHT

Fig. 4. Low-rank tensor recovery: $n_1 = 5$, $n_2 = 5$, $n_3 = 6$, $b = 0.25m$.



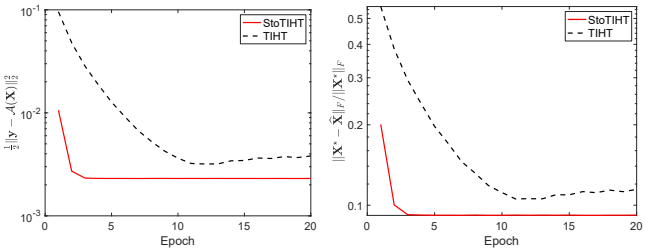
(a) TIHT (b) StoTIHT

Fig. 5. Low-rank tensor recovery: $n_1 = 5$, $n_2 = 5$, $n_3 = 6$, $b = 0.25m$.



(a) Original (b) TIHT (c) StoIHT

Fig. 6. Candle video recovery: $n_1 = 30$, $n_2 = 30$, $n_3 = 10$, $b = 0.25m$.



(a) (b)

Fig. 7. Candle video recovery: $n_1 = 30$, $n_2 = 30$, $n_3 = 10$, $b = 0.25m$. (a) Cost function. (b) Plot of relative error.

the tensor restricted isometry, we can guarantee the linear convergence of the proposed algorithm. Our simulation results also show that the proposed StoTIHT algorithm significantly outperforms the original TIHT algorithm especially for high-dimensional data sets. In particular, the proposed StoTIHT algorithm converges much faster and can achieve a lower recovery error than the original TIHT algorithm in a large-scale setting.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Security Agency under Grant No. H98230-19-1-0119, The Lyda Hill Foundation, The McGovern Foundation, and Microsoft Research, while the authors were in residence at the Mathematical Sciences Research Institute in Berkeley, California, during the summer of 2019 as part of the Summer Research for Women in Mathematics (SWiM) program. In addition, Li was supported by the NSF grants CCF-1409258, CCF-1704204, and the DARPA Lagrange Program under ONR/SPAWAR contract N660011824020. Grotheer was supported by the Goucher College Summer Research grant. Needell was supported by NSF CAREER DMS #1348721 and NSF BIGDATA DMS #1740325. Qin was supported by the NSF DMS #1941197.

REFERENCES

[1] I. S. Sokolnikoff, *Tensor analysis: Theory and applications*. Wiley, 1951.

[2] X.-D. Zhang, *Matrix analysis and applications*. Cambridge University Press, 2017.

[3] H. Fan, Y. Chen, Y. Guo, H. Zhang, and G. Kuang, "Hyperspectral image restoration using low-rank tensor recovery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 10, pp. 4589–4604, 2017.

[4] J. A. Bengua, H. N. Phien, H. D. Tuan, and M. N. Do, "Efficient tensor completion for color image and video recovery: Low-rank tensor train," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2466–2479, 2017.

[5] Q. Li, A. Prater, L. Shen, and G. Tang, "Overcomplete tensor decomposition via convex optimization," in *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, 2015, pp. 53–56.

[6] Q. Li, S. Li, H. Mansour, M. B. Wakin, D. Yang, and Z. Zhu, "Jazz: A companion to music for frequency estimation with missing data," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 3236–3240.

[7] S. Li, G. Tang, and M. B. Wakin, "Simultaneous blind deconvolution and phase retrieval with tensor iterative hard thresholding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2977–2981.

[8] C. J. Hillar and L.-H. Lim, "Most tensor problems are NP-hard," *Journal of the ACM (JACM)*, vol. 60, no. 6, p. 45, 2013.

[9] S. Friedland and L.-H. Lim, "Nuclear norm of higher-order tensors," *Mathematics of Computation*, vol. 87, no. 311, pp. 1255–1281, 2018.

[10] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[11] J. D. Blanchard, J. Tanner, and K. Wei, "CGIHT: conjugate gradient iterative hard thresholding for compressed sensing and matrix completion," *Information and Inference: A Journal of the IMA*, vol. 4, no. 4, pp. 289–327, 2015.

[12] R. E. Carrillo and K. E. Barner, "Lorentzian iterative hard thresholding: Robust compressed sensing with prior information," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4822–4833, 2013.

[13] J. Tanner and K. Wei, "Normalized iterative hard thresholding for matrix completion," *SIAM Journal on Scientific Computing*, vol. 35, no. 5, pp. S104–S125, 2013.

[14] E. Chunikhina, R. Raich, and T. Nguyen, "Performance analysis for matrix completion via iterative hard-thresholded svd," in *2014 IEEE Workshop on Statistical Signal Processing (SSP)*. IEEE, 2014, pp. 392–395.

[15] J. Geng, X. Yang, X. Wang, and L. Wang, "An accelerated iterative hard thresholding method for matrix completion," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 8, no. 7, pp. 141–150, 2015.

[16] N. Nguyen, D. Needell, and T. Woolf, "Linear convergence of stochastic iterative greedy algorithms with sparse constraints," *IEEE Transactions on Information Theory*, vol. 63, no. 11, pp. 6869–6895, 2017.

[17] J. Qin, S. Li, D. Needell, A. Ma, R. Grotheer, C. Huang, and N. Durgin, "Stochastic greedy algorithms for multiple measurement vectors," *arXiv preprint arXiv:1711.01521*, 2017.

[18] H. Rauhut, R. Schneider, and Ž. Stojanac, "Low rank tensor recovery via iterative hard thresholding," *Linear Algebra and its Applications*, vol. 523, pp. 220–262, 2017.

[19] J. H. de Morais Goulart and G. Favier, "Low-rank tensor recovery using sequentially optimal modal projections in iterative hard thresholding (SEMPIHT)," *SIAM Journal on Scientific Computing*, vol. 39, no. 3, pp. A860–A889, 2017.

[20] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[21] L. R. Tucker, "Implications of factor analysis of three-way matrices for measurement of change," *Problems in Measuring Change, University of Wisconsin Press Madison*, vol. 15, pp. 122–137, 1963.

[22] ——, "The extension of factor analysis to three-dimensional matrices," *Contributions to Mathematical Psychology, Holt, Rinehart and Winston, New York*, pp. 110–127, 1964.

[23] 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.

[24] C. Mu, B. Huang, J. Wright, and D. Goldfarb, "Square deal: Lower bounds and improved relaxations for tensor recovery," in *International conference on machine learning*, 2014, pp. 73–81.

[25] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM review*, vol. 52, no. 3, pp. 471–501, 2010.