Grassmannian Optimization for Online Tensor Completion and Tracking with the t-SVD

Kyle Gilman, *Student Member, IEEE,* Davoud Ataee Tarzanagh, *Member, IEEE*
and Laura Balzano, *Senior Member, IEEE*

***Abstract*—We propose a new fast streaming algorithm for the tensor completion problem of imputing missing entries of a low-tubal-rank tensor using the tensor singular value decomposition (t-SVD) algebraic framework. We show the t-SVD is a specialization of the well-studied block-term decomposition for third-order tensors, and we present an algorithm under this model that can track changing free submodules from incomplete streaming 2-D data. The proposed algorithm uses principles from incremental gradient descent on the Grassmann manifold of subspaces to solve the tensor completion problem with linear complexity and constant memory in the number of time samples. We provide a local expected linear convergence result for our algorithm. Our empirical results are competitive in accuracy but much faster in compute time than state-of-the-art tensor completion algorithms on real applications to recover temporal chemo-sensing and MRI data under limited sampling.**

***Index Terms*—t-SVD, Grassmannian optimization, online tensor completion, block-term decomposition**

I. Introduction

Modern data are increasingly high-dimensional and multi-way, increasing the storage and computational burden of signal processing algorithms. Many practical applications collect data over multiple modalities and can be approximated by a linear spectral mixture model, such as hyperspectral imaging (HSI), which captures dozens or even hundreds of images in narrow, adjacent spectral bands for each frame [52], or time-sequential HSI, i.e., hyperspectral video (HSV), with hundreds of spectral bands and megapixel spatial resolution which requires images to be recorded at the order of 10 G pixels per second. It is currently infeasible to process this type of high-rate data in real time applications [19]. Similarly, chemo-sensing experiments record sensor readings from dozens of channels in hundreds of experiments over thousands of time series. Batch processing of large-scale tensor data quickly becomes computationally intractable, and even storing these tensors is problematic as the memory requirements grow rapidly with the number and size of the tensor modes. Additional challenges include large numbers of missing tensor entries, streaming multiway data that needs to be processed on the fly, and data that may evolve in time with model dynamics.

To address these concerns, there is extensive recent literature studying low-dimensional tensor decompositions and fast algorithms for computing them. These decompositions provide a low-memory model approximation to tensor data that can be used for compression and interpolation of missing entries. Several algebraic frameworks exist for the analysis and decomposition of tensors, each with their own notion of tensor rank. In this paper, we consider sampling and recovery of three-way tensors using the algebraic framework of the tensor singular value decomposition (t-SVD) [15], [29], [39]. Three-way tensors are treated as linear operators over the space of oriented matrices and group rings of fibers under the tensor-product (t-product) multiplicative operator. Using this framework, one obtains an SVD-like factorization referred to as the tensor-SVD (t-SVD) with a defined notion of rank referred to as the tubal-rank. A key property of the t-SVD is the optimality of the truncated t-SVD for data approximation under the Frobenius norm measure [58]. The t-SVD has found wide utility in computer vision [11], [60], image and signal processing [59], [62], [34], geophysics, HSI/HSV [17], [18], and other applications because of its ability to capture signal shifts and scaling due to the model's circulant algebra. However, most existing t-SVD based methods are batch methods that require all of the data to be stored in memory at computation time and/or require the computation of multiple SVDs. This is very time-consuming and inefficient for large-scale data. Current t-SVD algorithms also do not model dynamically changing data.

Despite much development of the t-SVD, little work has shown its connections to standard multilinear algebra models, which are more mathematically interpretable. In this paper, we show the t-SVD can be equivalently expressed in standard multilinear algebra as a certain block-term decomposition (BTD) problem. The BTD model is a generalization of both the CANDECOMP/PARAFAC decomposition (CPD) and Tucker tensor decompositions with important applications in linear spectral mixture models, decoupling multivariate polynomials, and audio signal separation [44]. To the best of our knowledge, we are the first to show this equivalence. We show the t-SVD is an efficient factorization of each block in the BTD by utilizing a fixed unitary factor—the discrete Fourier transform matrix—in the third mode.

The impetus of this paper is to propose a fast, efficient algorithm for recovering low-tubal-rank tensor data from streaming, highly-incomplete multiway data with incremen-

K. Gilman, D. Tarzanagh, & L. Balzano are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, 48109 USA (e-mail: kgilman@umich.edu; tarzanaq@umich.edu; girasole@umich.edu).

The authors were supported in part by AFOSR YIP award FA9550-19-1-0026, NSF CAREER award CCF-1845076, and NSF BIGDATA award IIS-1838179.

tal gradient descent on the product manifold of low-rank matrices. Our methods are online by nature and can handle dynamically changing data, avoid computing SVDs, maintain orthonormality on the product of Grassmann manifolds, scale linearly in computation with the number of samples, and are highly parallelizable. We compare our method to batch t-SVD methods and online tensor decompositions. We show our method's ability to track dynamically time-varying low-rank free submodules in real data settings.

A. Organization of this paper

- Section II introduces our notation and the mathematical representations for the CP, Tucker, and BTD decompositions. Since our tensor factorization algorithm is based on the t-product [15], we briefly cover the background for this decomposition and leave the details for the appendix. At a high level, the t-product is convolutional and so can be performed by a product in the Fourier domain. We also discuss the properties of the t-SVD as it relates to the block-term decomposition.
- Section III details related work in tensor decompositions and completion.
- Section IV proposes our tensor completion method, summarized in Algorithm 1. This section also provides a local convergence result for the proposed algorithm, showing that in a local region we achieve a linear convergence rate in expectation.
- Section V gives experimental results for synthetic data, chemo-sensing experiments, and MRI completion.

II. Preliminaries

A. Notation

We shall denote all scalar quantities as s, vectors as $\boldsymbol{v}$, matrices as $\boldsymbol{A}$, and tensors as $\boldsymbol{\mathcal{X}}$. The i^{th} lateral slice of a three-way tensor $\boldsymbol{\mathcal{X}}$ is a matrix and is denoted as $\overrightarrow{\boldsymbol{\mathcal{X}}}_i$; in MATLAB notation this object refers to $\boldsymbol{\mathcal{X}}(:,i,:)$. The (frontal) faces of a tensor $\boldsymbol{\mathcal{X}}$, denoted as $\boldsymbol{X}_i$, are $\boldsymbol{\mathcal{X}}(:,:,i)$. Any $1\times 1\times d_3$ tube along the third-dimension is denoted as $\overrightarrow{\boldsymbol{v}}$. The n-mode unfolding of a tensor $\boldsymbol{\mathcal{X}} \in \boldsymbol{F}^{d_1\times\cdots\times d_N}$ into a $d_n \times \Pi_{i\neq n}^N d_i$ matrix is written as $\boldsymbol{X}_{(n)}$.

We write the Kronecker product as $\otimes$, the Khatri-Rao product as $\odot$, and the outer product as $\circ$. The mode-n product of a tensor $\boldsymbol{\mathcal{X}}$ with matrix $\boldsymbol{C}$ is denoted as $\boldsymbol{\mathcal{X}} \times_n \boldsymbol{C}$ and its mode-n matricization is defined as $(\boldsymbol{\mathcal{X}} \times_n \boldsymbol{C})_{(n)} = \boldsymbol{C}\boldsymbol{X}_{(n)}$. Refer to [30] for more on these products and their properties and identities. For the purposes of t-SVD and BTD models, we will often need to write $\boldsymbol{\mathcal{X}} \times_3 \boldsymbol{C}$, for some tensor $\boldsymbol{\mathcal{X}}$ and matrix $\boldsymbol{C}$, which we will denote as $\overline{\boldsymbol{\mathcal{X}}}$. The faces of $\overline{\boldsymbol{\mathcal{X}}}$ are then written as $\overline{\boldsymbol{X}}_i$.

We denote the Frobenius norm as $\|\boldsymbol{\mathcal{A}}\|_F = \sqrt{\sum_{ijk} |\boldsymbol{\mathcal{A}}_{ijk}|^2}$. The complex conjugate of a quantity $\texttt{conj}(\cdot)$ takes the complex conjugate of each entry. The complex conjugate transpose of a matrix $\boldsymbol{A}$ is denoted as $\boldsymbol{A}'$ and the psuedo-inverse as $\boldsymbol{A}^\dagger$.

B. Multilinear tensor decompositions

A rank-r CP decomposition is a sum of r rank-1 outer products [24]. For a three-way tensor $\boldsymbol{\mathcal{X}} \in \boldsymbol{F}^{d_1\times d_2\times d_3}$ with scalar weights $\boldsymbol{\lambda} = [\lambda_1 \cdots \lambda_r]' \in \mathbb{R}^r$ and factor matrices (assumed to be normalized to have unit column norms) $\boldsymbol{A} = [\boldsymbol{a}_1 \cdots \boldsymbol{a}_r] \in \boldsymbol{F}^{d_1\times r}$, $\boldsymbol{B} = [\boldsymbol{b}_1 \cdots \boldsymbol{b}_r] \in \boldsymbol{F}^{d_2\times r}$, and $\boldsymbol{C} = [\boldsymbol{c}_1 \cdots \boldsymbol{c}_r] \in \boldsymbol{F}^{d_3\times r}$ the decomposition is expressed as

$$\text{(CP)} \qquad \boldsymbol{\mathcal{X}} \approx \sum_{i=1}^{r} \lambda_i \boldsymbol{a}_i \circ \boldsymbol{b}_i \circ \boldsymbol{c}_i := [\![\boldsymbol{\lambda}; \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]\!]. \quad (1)$$

A multirank-(m,n,p) Tucker decomposition [49] permits a different rank in each mode unfolding of the tensor and represents each unfolding's columns in the span of an orthonormal basis. The Tucker decomposition for orthonormal factor matrices $\boldsymbol{A} = [\boldsymbol{a}_1 \cdots \boldsymbol{a}_m] \in \boldsymbol{F}^{d_1\times m}$, $\boldsymbol{B} = [\boldsymbol{b}_1 \cdots \boldsymbol{b}_n] \in \boldsymbol{F}^{d_2\times n}$, $\boldsymbol{C} = [\boldsymbol{c}_1 \cdots \boldsymbol{c}_p] \in \boldsymbol{F}^{d_3\times p}$ and core tensor $\boldsymbol{\mathcal{G}} \in \boldsymbol{F}^{m\times n\times p}$ is

$$\text{(Tucker)} \qquad \boldsymbol{\mathcal{X}} \approx \boldsymbol{\mathcal{G}} \times_1 \boldsymbol{A} \times_2 \boldsymbol{B} \times_3 \boldsymbol{C} := [\![\boldsymbol{\mathcal{G}}; \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]\!]. \quad (2)$$

The core tensor is a smaller tensor whose entries show the level of interaction between the different components $\boldsymbol{A}, \boldsymbol{B}$, and $\boldsymbol{C}$. A Tucker tensor is decomposed as a core multiplied by the corresponding factor matrix along each mode [30]. Observe that the CPD is a Tucker tensor whose factors are non-orthogonal with a core tensor having all ones along the super-diagonal and zeros everywhere else.

The block-term decomposition (BTD) model [12] is a useful generalization of both the CP and Tucker decompositions. The model expresses a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{d_1\times d_2\times d_3}$ as a sum of low-multirank tensors:

$$\text{(BTD)} \qquad \boldsymbol{\mathcal{X}} \approx \sum_{k=1}^{K} [\![\boldsymbol{\mathcal{G}}_k; \boldsymbol{A}_k, \boldsymbol{B}_k, \boldsymbol{C}_k]\!], \quad (3)$$

where $\boldsymbol{\mathcal{G}}_k \in \mathbb{R}^{M_k\times N_k\times P_k}$ is each multirank-(M_k, N_k, P_k) core tensor, and $\boldsymbol{A}_k \in \mathbb{R}^{d_1\times M_k}$, $\boldsymbol{B}_k \in \mathbb{R}^{d_2\times N_k}$, and $\boldsymbol{C}_k \in \mathbb{R}^{d_3\times P_k}$ for $k = 1,\ldots,K$ are the factor matrices. From (3), it is easy to see the $K = 1$ BTD with orthogonal factors specializes to the Tucker decomposition while a multirank-$(1,1,1)$ BTD is simply the CPD.

C. t-SVD tensors

1) Discrete Fourier Transform: Denote the normalized Discrete Fourier Transform (DFT) matrix for operation on a length-n signal as the unitary matrix $\boldsymbol{F}_n \in \mathbb{C}^{n\times n}$ and the DFT of some vector $\boldsymbol{v} \in \mathbb{R}^n$ as $\bar{\boldsymbol{v}} = \boldsymbol{F}_n\boldsymbol{v} \in \mathbb{C}^n$. The DFT is computed in $\mathcal{O}(n\log n)$ time by the fast Fourier transform (FFT) as $\bar{\boldsymbol{v}} = \texttt{fft}(\boldsymbol{v})$. Similarly, $\boldsymbol{v} = \boldsymbol{F}_n'\bar{\boldsymbol{v}}$ computes the inverse DFT (IDFT).

We denote $\overline{\boldsymbol{\mathcal{X}}} \in \mathbb{C}^{d_1\times d_2\times d_3}$ as the result of computing the DFT along the 3rd dimension, i.e. performing the DFT on the tubes of $\boldsymbol{\mathcal{X}}$, or equivalently $\overline{\boldsymbol{\mathcal{X}}} := \boldsymbol{\mathcal{X}} \times_3 \boldsymbol{F}_{d_3}$. Using the FFT (with indexing in MATLAB notation) we have $\overline{\boldsymbol{\mathcal{X}}} = \texttt{fft}(\boldsymbol{\mathcal{X}}, [], 3)$ and similarly by the inverse DFT, we have $\boldsymbol{\mathcal{X}} = \texttt{ifft}(\overline{\boldsymbol{\mathcal{X}}}, [], 3)$.

2) Tensor-tensor product: Define the block-diagonal matrix $\overline{\boldsymbol{X}} \in \mathbb{C}^{d_1d_3\times d_2d_3}$ to be the matrix with d_3 frontal faces of $\overline{\boldsymbol{\mathcal{X}}}$ along the diagonal, i.e. denote each frontal face of size $d_1\times d_2$ as $\overline{\boldsymbol{X}}_k$ and we have

$$\overline{\boldsymbol{X}} = \texttt{bdiag}(\overline{\boldsymbol{\mathcal{X}}}) = \begin{bmatrix} \overline{\boldsymbol{X}}_1 & & \\ & \ddots & \\ & & \overline{\boldsymbol{X}}_{d_3} \end{bmatrix}. \quad (4)$$

We define the block-circulant matrix of the frontal faces of $\boldsymbol{\mathcal{X}}$ as $\texttt{bcirc}(\boldsymbol{\mathcal{X}})$, where

$$\texttt{bcirc}(\boldsymbol{\mathcal{X}}) = \begin{bmatrix} \boldsymbol{X}_1 & \boldsymbol{X}_{d_3} & \dots & \boldsymbol{X}_2 \\ \boldsymbol{X}_2 & \boldsymbol{X}_1 & \dots & \boldsymbol{X}_3 \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{X}_{d_3} & \boldsymbol{X}_{d_3-1} & \dots & \boldsymbol{X}_1 \end{bmatrix} \in \mathbb{R}^{d_1 d_3 \times d_2 d_3}.$$

From properties of block-circulant matrices, $\texttt{bcirc}(\boldsymbol{\mathcal{X}})$ can be block-diagonalized by the DFT:

$$\overline{\boldsymbol{X}} = (\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1}) \cdot \texttt{bcirc}(\boldsymbol{\mathcal{X}}) \cdot (\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_2}), \tag{5}$$

where $(\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_2})$ is unitary [11]. For $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ we define the fold and unfold operators [15]:

$$\begin{aligned} \texttt{unfold}(\boldsymbol{\mathcal{X}}) &= \begin{bmatrix} \boldsymbol{X}_1' & \boldsymbol{X}_2' & \cdots & \boldsymbol{X}_{d_3}' \end{bmatrix}', \\ \texttt{fold}(\texttt{unfold}(\boldsymbol{\mathcal{X}})) &= \boldsymbol{\mathcal{X}}, \end{aligned}$$

where the $\texttt{unfold}(\cdot)$ operator maps $\boldsymbol{\mathcal{X}}$ to a matrix of size $d_1 d_3 \times d_2$ and $\texttt{fold}(\cdot)$ is its inverse operator.

Definition II.1. *Tensor-product (t-product)[15]: Let $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{d_2 \times l \times d_3}$. Then the t-product $\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$ is defined to be a tensor of size $d_1 \times l \times d_3$,*

$$\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}} = \texttt{fold}(\texttt{bcirc}(\boldsymbol{\mathcal{A}}) \cdot \texttt{unfold}(\boldsymbol{\mathcal{B}})). \tag{6}$$

The t-product can be understood from several perspectives. First, in the canonical domain, a three-way tensor of size $d_1 \times d_2 \times d_3$ can be thought of as an $d_1 \times d_2$ matrix whose entries are tubes lying in the third dimension. The t-product is then analogous to matrix-matrix multiplication but where circular convolution replaces scalar multiplication between the matrix elements. Second, the t-product is equivalent to matrix-matrix multiplication in the Fourier domain, or $\boldsymbol{\mathcal{C}} = \boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}}$ is equivalent to $\overline{\boldsymbol{C}} = \overline{\boldsymbol{A}}\,\overline{\boldsymbol{B}}$ from (5). This is shown as follows:

$$\begin{aligned} &\texttt{unfold}(\boldsymbol{\mathcal{C}}) = \texttt{bcirc}(\boldsymbol{\mathcal{A}}) \cdot \texttt{unfold}(\boldsymbol{\mathcal{B}}) \\ &= (\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1}) \cdot ((\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1})\texttt{bcirc}(\boldsymbol{\mathcal{A}})(\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_2})) \\ &\qquad \cdot ((\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_2})\texttt{unfold}(\boldsymbol{\mathcal{B}})) \\ &= (\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1})\overline{\boldsymbol{A}} \cdot \texttt{unfold}(\overline{\boldsymbol{\mathcal{B}}}). \end{aligned} \tag{7}$$

Therefore,

$$\texttt{unfold}(\overline{\boldsymbol{\mathcal{C}}}) = \overline{\boldsymbol{A}} \cdot \texttt{unfold}(\overline{\boldsymbol{\mathcal{B}}})$$

and for each front slice of $\overline{\boldsymbol{C}}$, $\overline{\boldsymbol{C}}_k = \overline{\boldsymbol{A}}_k \overline{\boldsymbol{B}}_k \quad \forall k = 1, \dots, d_3$. Eq. (7) and Lemma D.1 in the Appendix admit an efficient algorithm to compute the t-product using FFTs, as shown in Algorithm 3 in the Appendix. Like matrix multiplication, the t-product is associative and linear [15]. In the case where $d_3 = 1$, it is easy to see that the t-product becomes regular matrix multiplication.

With the definition of this product between tensors, we can define analogous definitions of conjugate transpose ($\boldsymbol{\mathcal{X}}'$), the identity tensor ($\boldsymbol{\mathcal{I}}_{nnd} \in \mathbb{R}^{n \times n \times d}$), orthogonal tensors, and a type of diagonal tensor called the F-diagonal tensor. We leave these details for the reader in Appendix D. Next we briefly discuss an SVD-like factorization of tensors under the t-product, and a definition of tubal-rank under the t-product and t-SVD.

Theorem II.2 (Tensor Singular Value Decomposition (t-SVD))**.** *[15] Any tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ can be factorized as $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{S}} * \boldsymbol{\mathcal{V}}'$, where $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{d_1 \times d_1 \times d_3}$,$\boldsymbol{\mathcal{V}} \in \mathbb{R}^{d_2 \times d_2 \times d_3}$ are orthogonal tensors, and $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ is an F-diagonal tensor.*

We state Theorem II.2 without proof here and refer the reader to [11] for a detailed proof. The t-SVD can be computed efficiently by Algorithm 4 in the appendix.

Definition II.3 (Tensor multi-rank and tubal-rank)**.** *[62] For any $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$, its multi-rank is a vector defined as $\boldsymbol{r} = (\mathrm{rank}(\overline{\boldsymbol{X}}_1), \dots, \mathrm{rank}(\overline{\boldsymbol{X}}_{d_3})) \in \mathbb{R}^{d_3}$. The tensor tubal-rank, $\mathrm{rank}_t(\boldsymbol{\mathcal{X}})$, is defined as the number of nonzero singular tubes of $\boldsymbol{\mathcal{S}}$ from the t-SVD, i.e.,*

$$\mathrm{rank}_t(\boldsymbol{\mathcal{X}}) = \#\{i : \boldsymbol{\mathcal{S}}(i,i,:) \neq \boldsymbol{0}\} = \max\{r_1, \dots, r_{d_3}\},$$

where $r_k = \mathrm{rank}(\overline{\boldsymbol{X}}_k)$.

Definition II.4 (Module over the commutative ring)**.** *[60] It can be shown the set of tubes $\mathbb{C}^{1 \times 1 \times d_3}$ equipped with the t-product forms a ring with unity $\mathbb{R}(\mathbb{G}_{d_3})$ [48]. Define $\mathbb{M}_{d_3}^{d_1}$ to be a module, or the set of all 2-D lateral slices of size $d_1 \times 1 \times d_3$, over the ring of tubes. Since for any element $\overrightarrow{\boldsymbol{\mathcal{X}}} \in \mathbb{M}_{d_3}^{d_1}$ and coefficient tube $\overrightarrow{\boldsymbol{v}} \in \mathbb{R}^{1 \times 1 \times d_3}$, the lateral slice $\overrightarrow{\boldsymbol{\mathcal{Y}}} = \overrightarrow{\boldsymbol{\mathcal{X}}} * \overrightarrow{\boldsymbol{v}}$ is also an element of the module, so $\mathbb{M}_{d_3}^{d_1}$ is closed under tubal-scalar multiplication.*

Definition II.5 (Free submodule (FSM))**.** *$\mathbb{M}_{d_3}^{d_1}$ is called a free submodule of dimension $r < d_1$ over the commutative ring $\mathbb{R}(\mathbb{G}_{d_3})$ [60], where one can construct a spanning basis of orthonormal lateral slices $\{\overrightarrow{\boldsymbol{\mathcal{U}}}_1, \overrightarrow{\boldsymbol{\mathcal{U}}}_2, \dots, \overrightarrow{\boldsymbol{\mathcal{U}}}_r\}$ for which we we can uniquely represent any element $\overrightarrow{\boldsymbol{\mathcal{X}}} \in \mathbb{M}_{d_3}^{d_1}$ as a t-linear combination of the spanning basis with some tubal coefficients $\overrightarrow{\boldsymbol{w}}_k$:*

$$\overrightarrow{\boldsymbol{\mathcal{X}}} = \sum_{k=1}^{r} \overrightarrow{\boldsymbol{\mathcal{U}}}_k * \overrightarrow{\boldsymbol{w}}_k = \boldsymbol{\mathcal{U}} * \overrightarrow{\boldsymbol{\mathcal{W}}}. \tag{8}$$

Together, $\{\overrightarrow{\boldsymbol{\mathcal{U}}}_1, \overrightarrow{\boldsymbol{\mathcal{U}}}_2, \dots, \overrightarrow{\boldsymbol{\mathcal{U}}}_r\}$ form the orthonormal tensor $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{d_1 \times r \times d_3}$, and the arranged tubes $\overrightarrow{\boldsymbol{w}}_k$ form the lateral slice $\overrightarrow{\boldsymbol{\mathcal{W}}} \in \mathbb{R}^{r \times 1 \times d_3}$.

The definitions of free submodule over a ring generalize the notions of vector subspaces over a field of scalars and a spanning basis for a vector subspace, where the scalars of the field are the elements of the ring. Our algorithm uses the notions of free submodule to learn a spanning basis for the observed 2-D lateral slices of data in $\mathbb{M}_{d_3}^{d_1}$.

Before defining the manifolds and orthogonal groups used in our tensor problem, we first denote the complex orthogonal group, complex Stiefel manifold, and complex Grassmann manifold, respectively, from matrix linear algebra [1], [16]:

$$\begin{aligned} \overline{\mathcal{O}}(r) &:= \{\overline{\boldsymbol{R}} \in \mathbb{C}^{r \times r},\ \overline{\boldsymbol{R}}'\,\overline{\boldsymbol{R}} = \overline{\boldsymbol{R}}\,\overline{\boldsymbol{R}}' = \mathbf{I}_r\}, \\ \overline{\mathcal{S}}(r, d_1) &:= \{\overline{\boldsymbol{U}}'\overline{\boldsymbol{U}} = \mathbf{I} : \overline{\boldsymbol{U}} \in \mathbb{C}^{d_1 \times r}\}, \\ [\overline{\boldsymbol{U}}] &:= \big\{\overline{\boldsymbol{U}}\,\overline{\boldsymbol{R}} :\ \overline{\boldsymbol{R}} \in \overline{\mathcal{O}}(r)\big\} \in \overline{\mathcal{G}}(r, d_1),\ \text{for } \overline{\boldsymbol{U}} \in \overline{\mathcal{S}}(r, d_1). \end{aligned} \tag{9}$$

Next we provide extensions of these definitions to third order tensors under the t-product.

Definition II.6 (t-orthogonal group)**.** *Define $\mathcal{O}(r, r, d_3)$ as the t-orthogonal group of tubal rank-r:*

$$\mathcal{O}(r, r, d_3) := \left\{\boldsymbol{\mathcal{R}} \in \mathbb{R}^{r\times r\times d_3} :\ \boldsymbol{\mathcal{R}}' * \boldsymbol{\mathcal{R}} = \boldsymbol{\mathcal{R}} * \boldsymbol{\mathcal{R}}' = \boldsymbol{\mathcal{I}}_{rrd_3}\right\}. \tag{10}$$

Definition II.7 (t-Stiefel manifold)**.** *The t-Stiefel manifold consisting of all tubal-rank-r tensors with orthonormal lateral slices defined as*

$$\mathcal{S}(r, d_1, d_3) := \{\boldsymbol{\mathcal{U}} \in \mathbb{R}^{d_1\times r\times d_3},\ \boldsymbol{\mathcal{U}}' * \boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{I}}_{rrd_3}\}. \tag{11}$$

We note that the t-Stiefel manifold is indeed a product of Stiefel manifolds in the Fourier domain, since each frontal slice of $\overline{\boldsymbol{\mathcal{U}}} = \boldsymbol{\mathcal{U}} \times_3 \boldsymbol{F}_{d_3}$ is orthonormal and is a point on a Stiefel manifold, making $\overline{\boldsymbol{\mathcal{U}}}$ a point in the product space of Stiefel manifolds. We also note that $\boldsymbol{\mathcal{U}} = \overline{\boldsymbol{\mathcal{U}}} \times_3 \boldsymbol{F}'_{d_3}$, where $\boldsymbol{F}'_{d_3}$ is an invertible linear mapping of the frontal slices of $\overline{\boldsymbol{\mathcal{U}}}$. This together with the smoothness of $\overline{\boldsymbol{\mathcal{U}}}$ (as a product of smooth manifolds) implies that $\mathcal{S}(r, d_1, d_3)$ is also a smooth manifold; see, e.g., [56, Lemma 1].

Definition II.8 (t-Grassmann manifold)**.** *Let $\sim_t$ denote an equivalence relation on the t-Stiefel manifold $\mathcal{S}(r, d_1, d_3)$ in the sense that for any $\boldsymbol{\mathcal{U}}_1, \boldsymbol{\mathcal{U}}_2 \in \mathcal{S}(r, d_1, d_3)$, $\boldsymbol{\mathcal{U}}_1 \sim_t \boldsymbol{\mathcal{U}}_2$ means that there exists a $\boldsymbol{\mathcal{R}} \in \mathcal{O}(r, r, d_3)$ such that $\boldsymbol{\mathcal{U}}_1 = \boldsymbol{\mathcal{U}}_2 * \boldsymbol{\mathcal{R}}$. The quotient space of $\mathcal{S}(r, d_1, d_3)$ under this equivalence relation, $\mathcal{S}(r, d_1, d_3)/\mathcal{O}(r, r, d_3)$, is called t-Grassmann manifold, i.e.,*

$$\mathcal{G}(r, d_1, d_3) := \left\{[\boldsymbol{\mathcal{U}}] :\ \boldsymbol{\mathcal{U}} \in \mathcal{S}(r, d_1, d_3)\right\}, \tag{12}$$

where $[\boldsymbol{\mathcal{U}}]$ denotes the equivalence class under $\sim_t$:

$$[\boldsymbol{\mathcal{U}}] := \left\{\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{R}} :\ \boldsymbol{\mathcal{R}} \in \mathcal{O}(r, r, d_3)\right\},$$

which is the r-dimensional free sub-module in $\mathbb{M}^{d_1}_{d_3}$ spanned under the t-product by the lateral slices of $\boldsymbol{\mathcal{U}}$.

Proposition II.9. *$\mathcal{G}(r, d_1, d_3)$ is a smooth compact manifold of dimension $d_3 r(d_1 - r)$.*

Proof. Let

$$\begin{aligned}&\overline{\mathcal{G}}(r, d_1, d_3) := \\ &\quad \left\{[\overline{\boldsymbol{\mathcal{U}}}] :\ \overline{\boldsymbol{\mathcal{U}}} \in \mathbb{C}^{d_1\times r\times d_3},\ \overline{\boldsymbol{U}}_k \in \overline{\mathcal{S}}(r, d_1),\ \ \forall k \in [d_3]\right\},\end{aligned} \tag{13}$$

where

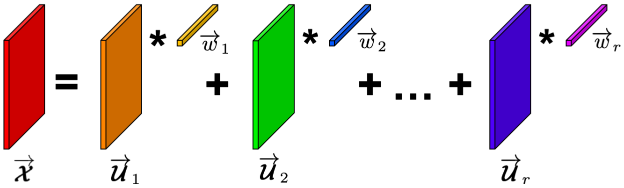
$$[\overline{\boldsymbol{\mathcal{U}}}] := \left\{\texttt{fold}\left(\overline{\boldsymbol{U}}_1\overline{\boldsymbol{R}}_1; \dots; \overline{\boldsymbol{U}}_{d_3}\overline{\boldsymbol{R}}_{d_3}\right), \overline{\boldsymbol{R}}_k \in \overline{\mathcal{O}}(r),\ \forall k \in [d_3]\right\},$$

recalling $\texttt{fold}(\cdot)$ is defined in Section II and stacks the slices in its argument into a tensor. Then, we have

$$\begin{aligned}[\overline{\boldsymbol{\mathcal{U}}}] &= \{\overline{\boldsymbol{U}}_1\overline{\boldsymbol{R}} :\ \overline{\boldsymbol{R}} \in \overline{\mathcal{O}}(r)\} \times \dots \times \{\overline{\boldsymbol{U}}_{d_3}\overline{\boldsymbol{R}} :\ \overline{\boldsymbol{R}} \in \overline{\mathcal{O}}(r)\} \\ &= [\overline{\boldsymbol{U}}_1] \times [\overline{\boldsymbol{U}}_2] \times \dots \times [\overline{\boldsymbol{U}}_{d_3}].\end{aligned}$$

This implies that $[\overline{\boldsymbol{\mathcal{U}}}] \in \overline{\mathcal{G}}(r, d_1) \times \dots \times \overline{\mathcal{G}}(r, d_1)$, where $\overline{\mathcal{G}}(r, d_1)$ is a smooth compact manifold of dimension $r(d_1 - r)$ [16]. Therefore, $\overline{\mathcal{G}}(r, d_1, d_3)$ is a smooth compact (product) manifold of dimension $d_3 r(d_1 - r)$, and since the Fourier transform is invertible, using the properties of the t-product in Definition II.1, for any $\boldsymbol{\mathcal{U}} \in \mathcal{S}(r, d_1, d_3)$, we have $[\boldsymbol{\mathcal{U}} \times_3 \boldsymbol{F}_{d_3}] = [\overline{\boldsymbol{\mathcal{U}}}]$ and $[\overline{\boldsymbol{\mathcal{U}}} \times_3 \boldsymbol{F}'_{d_3}] = [\boldsymbol{\mathcal{U}}]$. This implies that the t-Grassmannian $\mathcal{G}(r, d_1, d_3)$ from Definition II.8 is indeed a smooth and compact manifold.

□



$\vec{\boldsymbol{\mathcal{X}}}$
=
$\vec{\boldsymbol{\mathcal{U}}}_1$
*
$\vec{w}_1$
+
$\vec{\boldsymbol{\mathcal{U}}}_2$
*
$\vec{w}_2$
+ ... +
$\vec{\boldsymbol{\mathcal{U}}}_r$
*
$\vec{w}_r$

Fig. 1: An element of a free module generated by t-linear combination of spanning basis and coefficients.

D. t-SVD and BTD Equivalence

Definition II.10 ($\{(r_k, r_k, 1)\}_{k=1}^K$-multi-rank BTD)**.** *For multi-rank vector $\boldsymbol{r} = [r_1 \cdots r_K]' \in \mathbb{N}_+^K$, define the BTD tensor with the following decomposition for factors $\boldsymbol{A}_k \in \mathbb{R}^{d_1\times r_k}$, $\boldsymbol{B}_k \in \mathbb{R}^{d_2\times r_k}$, $\boldsymbol{c}_k \in \mathbb{R}^{d_3}$:*

$$\boldsymbol{\mathcal{X}} = \sum_{k=1}^{K} \boldsymbol{A}_k\boldsymbol{B}'_k \circ \boldsymbol{c}_k. \tag{14}$$

Proposition II.11. *Let $\boldsymbol{F}_{d_3}^{-1} = [\bar{\boldsymbol{f}}_1 \cdots \bar{\boldsymbol{f}}_{d_3}]$, where $\boldsymbol{F}_{d_3}^{-1}$ denotes the inverse-DFT matrix. For any tensor decomposition $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{W}}$ with factor tensors $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{d_1\times r\times d_3}$, $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{r\times d_2\times d_3}$, multi-rank $(r_1, \dots, r_{d_3})$, and tubal rank $r = \max_i\{r_i\}$, we have*

$$\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{W}} = \sum_{k=1}^{d_3} \overline{\boldsymbol{U}}_k\overline{\boldsymbol{W}}'_k \circ \bar{\boldsymbol{f}}_k. \tag{15}$$

Here, $\overline{\boldsymbol{U}}_k \in \mathbb{C}^{d_1\times r_k}$, $\overline{\boldsymbol{U}}'_k\overline{\boldsymbol{U}}_k = \mathbf{I}_{r_k}$, $\overline{\boldsymbol{W}}'_k \in \mathbb{C}^{r_k\times d_2}$ $\forall k = 1, \dots, d_3$, and $\overline{\boldsymbol{U}}_k$ and $\overline{\boldsymbol{W}}'_k$ are the rank-r_k faces of $\overline{\boldsymbol{\mathcal{U}}} = \boldsymbol{\mathcal{U}} \times_3 \boldsymbol{F}_{d_3}$ and $\overline{\boldsymbol{\mathcal{W}}} = \boldsymbol{\mathcal{W}} \times_3 \boldsymbol{F}_{d_3}$ respectively.

Proof. Let $\boldsymbol{F}_{d_3}^{-1}$ be the $d_3 \times d_3$ IDFT matrix. The identity in (15) is clear after writing the definition of the t-product:

$$\begin{aligned}&\texttt{unfold}(\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{W}}) = (\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1})\cdot \\ &(\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1})\texttt{bcirc}(\boldsymbol{\mathcal{U}})(\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_2}) \cdot (\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_2})\texttt{unfold}(\boldsymbol{\mathcal{W}}) \\ &= (\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1}) \cdot \overline{\boldsymbol{U}} \cdot \texttt{unfold}(\overline{\boldsymbol{W}}) \\ &= \texttt{unfold}\left(\sum_{k=1}^{d_3} (\overline{\boldsymbol{U}}_k\overline{\boldsymbol{W}}'_k) \circ \bar{\boldsymbol{f}}_k\right).\end{aligned}$$

Note that $\overline{\boldsymbol{\mathcal{U}}} = \boldsymbol{\mathcal{U}} \times_3 \boldsymbol{F}_{d_3}$ and $\overline{\boldsymbol{\mathcal{W}}} = \boldsymbol{\mathcal{W}} \times_3 \boldsymbol{F}_{d_3}$ respectively; these mode-products apply the Fourier transform to the third-mode fibers. □

In particular, Equation (15) links tubal and BTD decompositions and shows how a tensor factorization with multi-rank $(r_1, \dots, r_{d_3})$ can equivalently be represented as a BTD factorization with multi-rank $\{(r_k, r_k, 1)\}_{k=1}^{d_3}$. The equivalence reveals the t-SVD as a specialization of the BTD model with the third-mode fixed as the columns of the inverse DFT matrix. Each term in the t-SVD/BTD is itself a $(r_k, r_k, 1)$-multi-rank tensor with the identity core. In the linear spectral mixture model, each $\overline{\boldsymbol{U}}_k\overline{\boldsymbol{W}}'_k$ is the rank-r_k spectral map corresponding to a frequency component $\bar{\boldsymbol{f}}_k$. For a tubal-rank-1 t-SVD decomposition, (15) is a rank-d_3 CPD $[\![\overline{\boldsymbol{U}}; \overline{\boldsymbol{W}}; \boldsymbol{C}]\!]$, where $\boldsymbol{C} = \boldsymbol{F}_{d_3}^{-1}$ is the IDFT matrix.

Choosing all d_3 of the multi-ranks would be especially challenging. Since the tubal-rank r is much smaller than either of the tensor dimensions, we will slightly overparameterize the problem to a $(r, r, 1)$-*tubal BTD*:

Definition II.12 ($(r, r, 1)$-tubal BTD)**.** *Tensor $\boldsymbol{\mathcal{X}}$ has a $(r, r, 1)$-tubal BTD if $\boldsymbol{\mathcal{X}}$ has the decomposition in* (15) *for*

$$\boldsymbol{\mathcal{X}} = \sum_{k=1}^{d_3} \overline{\boldsymbol{U}}_k\overline{\boldsymbol{W}}'_k \circ \bar{\boldsymbol{f}}_k,$$

where $\overline{\boldsymbol{U}}_k \in \mathbb{C}^{d_1 \times r}$, $\overline{\boldsymbol{U}}'_k\overline{\boldsymbol{U}}_k = \mathbf{I}_r$, $\overline{\boldsymbol{W}}'_k \in \mathbb{C}^{r \times d_2}$ $\forall k = 1, \dots, d_3$.

In the strict sense, the t-SVD is not a true tensor decomposition since it lacks the trilinearity in the third mode. Rather, it is a collection of matrix factorizations that describe a tensor structure. In the light of its relationship to the BTD and linear spectral mixture models, the t-SVD, along with its variants using the DCT or other invertible linear transforms [28], becomes appropriate for applications like HSI and video compression when the tensor faces are shifted and scaled versions of one another; i.e. the model describes spectral correlations by the embedded circular convolution [61]. In the BTD form, the t-SVD decomposes the data into the frequency makeup of each pixel.

It is well-established the t-SVD is powerful in capturing the ubiquitous spatial-shifting and scaling correlations in real-world multiway data. We show tensors whose third modes exhibit these correlations lie in the t-linear span of the same free submodule, and formalize this notion in Proposition II.13 using the multilinear algebra interpretation of the t-SVD.

Proposition II.13. *An r-dimensional free submodule spanned by $\boldsymbol{\mathcal{U}}$ over the t-product is closed under circular-shifting and scaling.*

Proof. Let $\boldsymbol{\mathcal{X}}_i = \boldsymbol{\mathcal{U}} * \overrightarrow{\boldsymbol{\mathcal{W}}}_i = \sum_{k=1}^{d_3} \overline{\boldsymbol{U}}_k \bar{\boldsymbol{w}}_{i,k} \circ \bar{\boldsymbol{f}}_k$, where $\bar{\boldsymbol{w}}_{i,k} \in \mathbb{C}^r$ are the frontal faces of $\overrightarrow{\boldsymbol{\mathcal{W}}}_t \times_3 \boldsymbol{F}_{d_3}$.

Let $\boldsymbol{\mathcal{X}}_{i,\text{shift}} := \alpha_i \cdot \texttt{circshift}(\boldsymbol{\mathcal{X}}_i, s_i, \texttt{dims} = 3)$ for some real numbers α_i and integers s_i that scale and circularly shift the faces of each $\boldsymbol{\mathcal{X}}_i$. Then for n linear combinations of slices,

$$\begin{aligned}\sum_{i=1}^{n} \boldsymbol{\mathcal{X}}_{i,\text{shift}} &= \sum_{i=1}^{n}\sum_{k=1}^{d_3} \alpha_i e^{\frac{-j2\pi s_i k}{d_3}} \cdot \overline{\boldsymbol{U}}_k \bar{\boldsymbol{w}}_{i,k} \circ \bar{\boldsymbol{f}}_k \\ &= \sum_{k=1}^{d_3} \overline{\boldsymbol{U}}_k \left(\sum_{i=1}^{n} \alpha_i e^{\frac{-j2\pi s_i k}{d_3}} \bar{\boldsymbol{w}}_{i,k} \right) \circ \bar{\boldsymbol{f}}_k.\end{aligned}$$

Thus, $\sum_{i=1}^{n} \boldsymbol{\mathcal{X}}_{i,\text{shift}}$ shares the same $\boldsymbol{\mathcal{U}}$ in its t-SVD as each $\boldsymbol{\mathcal{X}}_i$. □

III. Related Work

It is well known that low-rank decompositions of highly undersampled matrix data, with certain assumptions of incoherent left and right singular vectors from the SVD and random sampling patterns, can be exploited to recover missing data by solving a convex optimization program [10]. This setting treats matrix data (a 2-way tensor) as a linear operator over a vector space and defines the rank of the matrix via its minimal decomposition into a sum of rank-1 matrices [58]. However, multiway data often contains correlations or interactions between modes of the tensor that would be destroyed if the tensor is flattened into a matrix [32]. More sophisticated algebraic techniques are required to analyze these special structures.

A. CANDECOMP/PARAFAC decomposition

One of the most widely used tensor decompositions is the CPD factorization, which finds a sum of rank-1 outer products that best compose the tensor, where the minimal number of such factors required is referred to as the CP rank. CP is powerful for imputing missing tensor data and also recovering latent factors that describe the tensor along each mode [30]. CP methods often use alternating least squares to update the factor matrices in a nonconvex optimization problem. Several varieties of CP algorithms exist for batch tensor completion [2], [26], [31]. However there are known computational and ill-posedness issues with the CP model, the foremost issue being that it is NP-hard to compute the CP rank of a tensor or the best low-rank CP approximation of a tensor in the Frobenius norm sense [23]. Furthermore, the alternating least squares algorithm is prone to getting stuck in local minima, so it may be sensitive to initialization or may require a special initialization step. CP models may also not be expressive enough to represent certain physical systems with block term decompositions.

Newer work in tensor completion has seen the development of several streaming CP tensor completion methods. A prominent streaming version of CP tensor completion was proposed by Mardani et al. [38] using stochastic gradient descent. Kasai [27] proposed another streaming CP tensor completion algorithm with a second-order stochastic gradient descent procedure based on the CP decomposition exploiting recursive least squares for faster convergence than the SGD method in Mardani et al., but at a higher computational cost. The main disadvantage to these streaming CP methods is that they require several hyperparameters that may be difficult to tune or know beforehand. These include a forgetting factor and the regularization parameters that penalize the Frobenius norm of the factor matrices [46]. While the forgetting factor must be hand-tuned, it does allow for the benefit of varying the algorithm's tracking ability from online mode to purely batch mode. Setting the appropriate CP rank of the model may also be challenging. Other streaming CP algorithms include [37], [22], [36], [50], [41].

B. Tucker decomposition

Another approach is to use the Tucker tensor decomposition in (2) and Tucker multilinear rank (or multirank) and its convex relaxation. The multirank formulation allows each tensor mode to be expressed in a subspace of different dimension. Tucker decompositions are typically computed using the Higher Order SVD (HOSVD) [13]. However, Tucker-based convex relaxation is not a tight relaxation of the Tucker rank and cannot give optimal recovery for tensor completion [48], [58].

The work in [47] proposes using randomized linear algebra in the fully-observed data setting to sketch the Tucker decomposition, which naturally permits their algorithm to handle streaming data. The authors in [40] propose a multi-aspect streaming Tucker-tensor algorithm for completing missing entries where one or more modes of the tensor grows in dimension length with time.

The online algorithm for tensor completion in [54] can also be thought of as an incremental Tucker algorithm with identity core tensor, which is the tensor of ones along the super-diagonal and zero elsewhere. Similar to our method, their algorithm tracks a low-dimensional subspace on the Grassmannian in each mode of the tensor using geodesic steps like the GROUSE algorithm [5].

C. *Block-term decomposition*

De Lathauwer et al. [12] explores a special class of third-order tensor decompositions called the block-term decomposition (BTD) model and its theoretical properties, including the specialization to the multirank-$(r_k, r_k, 1)$ model which represents a sum of matrix-vector outer products. In many applications, it is natural to represent data in the BTD model in modes of space-space-frequency [57]. The works in [44], [57] explicitly link the $(r_k, r_k, 1)$-BTD to applications with strong physical interpretation like linear spectral mixture models and spectrum cartography that could not be well-represented by CP or Tucker tensors. BTD permits a richer, more expressive representation of data with more than one tensor component, like in the Tucker model, or without restriction to rank-1 components like CP [44].

The work in [44] proposes a block-coordinate descent batch algorithm to compute the decomposition under full sampling. The authors of [57] propose algorithms for the case where tensor entries are missing in various patterns, and they prove the uniqueness and completion guarantees of the $(r_k, r_k, 1)$ BTD factors under mild conditions.

D. *t-SVD*

The t-SVD, a factorization originally posed by Kilmer et al. [29], enjoys many similar properties as matrix factorization problems, is solved by the SVD, and gives optimal recovery results under the Frobenius norm whenever the tensor data reveals a low-tubal rank structure [58], [11]. In many applications, for example time series or other ordered data, the corresponding tensor has a distinguishing orientation that exhibits a low tubal-rank structure [48]. Several works have proposed t-SVD factorization algorithms for tensors with missing entries. Zhang and Aeron [58] solve the exact tensor completion problem under the t-SVD algebra in a batch way using the tensor nuclear norm, a convex relaxation of tensor tubal-rank. The algorithm involves solving a convex program on each frontal slice of the tensor in the Fourier domain, which provably recovers the missing tensor entries given certain incoherence conditions. Zhou et al. [62] propose a different algorithm using a tensor factorization model under the t-product for rapid, efficient optimization, and Tarzanagh and Michailidis [48] employ randomized linear algebra to compute fast sketches of factorizations under the t-product. Each of these algorithms can only complete batch tensor data and cannot handle streaming multiway data.

Little work has been done to extend online matrix completion methods to the case of multiway tensor data using the t-SVD framework, apart from the work in [60] which proposed an online tensor robust principal component analysis algorithm. However, this method cannot predict missing tensor values and does not utilize orthonormal factorization. The work in [43] proposed an online tensor completion algorithm using the tensor nuclear norm for low-tubal-rank tensors, but it must compute multiple SVDs for each update.

The algorithm proposed in this paper differs from all of these methods in that it can operate incrementally over a tensor in batch mode or stream in online mode, even with dynamically changing data. Our proposed algorithm TOUCAN seeks the optimal low-rank approximation of a tensor in the Frobenius norm sense under the t-SVD and the BTD models when the data reveals a low tubal-rank structure, and is empirically robust to initialization. TOUCAN requires only a tolerance threshold and the model rank—which can be more easily determined empirically by inspecting the tubal singular value decomposition of the t-SVD of small batches of data or over the entire batch if feasible. This paper builds off the work in [20], giving a full derivation of the algorithm, exploring connections to the BTD model, adding new algorithms and theory, and including new and more extensive experiments.

IV. Proposed Method

A. *Model*

In the t-SVD framework, using Proposition II.11 we model the three-way tensor data $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ as

$$\boldsymbol{\mathcal{X}} \approx \sum_{k=1}^{d_3} \overline{\boldsymbol{U}}_k \overline{\boldsymbol{W}}_k' \circ \bar{\boldsymbol{f}}_k + \boldsymbol{\mathcal{N}} = \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{W}} + \boldsymbol{\mathcal{N}}, \tag{16}$$

where $\boldsymbol{\mathcal{N}}_{ijk} \sim \mathcal{N}(0, \sigma^2)$ represents white-Gaussian noise, and $\boldsymbol{\mathcal{U}} \in R^{d_1 \times r \times d_3}$ is an orthonormal tensor under the tensor-product, and $\overline{\boldsymbol{U}}_k \in \mathbb{R}^{d_1 \times r}, \overline{\boldsymbol{U}}_k' \overline{\boldsymbol{U}}_k = \mathbf{I}_r \quad \forall k = 1, \dots, d_3$.

Given d_2 2-D data samples $\overrightarrow{\boldsymbol{\mathcal{X}}}_1, \dots, \overrightarrow{\boldsymbol{\mathcal{X}}}_{d_2}$ of size $d_1 \times d_3$, we arrange them as lateral slices to make a three-way tensor $\boldsymbol{\mathcal{X}}$ of size $d_1 \times d_2 \times d_3$ [60]. In most circumstances, the t-SVD method would be used to compute $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{W}}$ [29]. For the purposes of this work, we consider the case of three-way tensor data where each lateral slice arrives sequentially in time and may contain missing entries, i.e. at every time t, we observe an incomplete lateral slice $\overrightarrow{\boldsymbol{\mathcal{X}}}_t \in \mathbb{M}_{d_3}^{d_1}$ on the indices $\Omega_t \subset \{1, \dots, d_1\} \times \{1, \dots, d_3\}$. Like the work in [60], we wish to compute the spanning low-dimensional free submodule of this multiway streaming data in an online way without storing the full tensor in memory or computing the t-SVD – both which may be prohibitive in large data settings.

We can learn the spanning free submodule using stochastic gradient techniques, similar to what the GROUSE algorithm

[5] does for matrices with streaming columns. We aim to track a r-dimensional free submodule of $\mathbb{M}_{d_3}^{d_1}$ that may evolve over time. Let $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{d_1 \times r \times d_3}$ be an orthonormal tensor whose r lateral slices span the free submodule of $\mathbb{M}_{d_3}^{d_1}$.

B. Deriving the objective function

We begin by writing the problem we wish to solve as a tubal-rank-r problem in t-SVD notation, and then we will express it as a $(r, r, 1)$-tubal BTD. In the scenario where the underlying free submodule does not change over time, a natural optimization problem with squared ℓ_2 error loss is given as

$$\min_{[\boldsymbol{\mathcal{U}}] \in \mathcal{G}(r,d_1,d_3)} \frac{1}{T} \sum_{t=1}^{T} \min_{\overrightarrow{\boldsymbol{\mathcal{W}}}_t \in \mathbb{R}^{r \times 1 \times d_3}} \frac{1}{2} \left\| \mathcal{A}_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t - \boldsymbol{\mathcal{U}} * \overrightarrow{\boldsymbol{\mathcal{W}}}_t) \right\|_F^2 . \tag{17}$$

Here, $\mathcal{A}_{\Omega_t}(\cdot)$ is the linear operator that extracts the observed samples in the set Ω_t from each lateral slice in $\boldsymbol{\mathcal{X}} = [\overrightarrow{\boldsymbol{\mathcal{X}}}_1, \ldots, \overrightarrow{\boldsymbol{\mathcal{X}}}_{d_2}]$, and $\mathcal{G}(r, d_1, d_3)$ denotes the t-Grassmannian from Definition II.8. We let $\mathcal{L}(\boldsymbol{\mathcal{U}}) := \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_t(\boldsymbol{\mathcal{U}})$ where

$$\mathcal{L}_t(\boldsymbol{\mathcal{U}}) := \frac{1}{2} \left\| \mathcal{A}_{\Omega_t}\left(\overrightarrow{\boldsymbol{\mathcal{X}}}_t - \boldsymbol{\mathcal{U}} * \overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})\right) \right\|_F^2, \quad \text{and} \tag{18a}$$

$$\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}) := \underset{\overrightarrow{\boldsymbol{\mathcal{W}}}_t \in \mathbb{R}^{r \times 1 \times d_3}}{\operatorname{argmin}} \quad \frac{1}{2} \left\| \mathcal{A}_{\Omega_t}\left(\overrightarrow{\boldsymbol{\mathcal{X}}}_t - \boldsymbol{\mathcal{U}} * \overrightarrow{\boldsymbol{\mathcal{W}}}_t\right) \right\|_F^2 . \tag{18b}$$

Since we have concatenated the time slices on the second dimension, let $d_2 = T$. We see it is possible to solve this problem incrementally, as described in [8], in terms of the orthonormal free-submodule $\boldsymbol{\mathcal{U}}$ and the optimal weights $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}) \in \mathbb{R}^{r \times 1 \times d_3}$ for all $t = 1, \ldots, T$. We solve the nested optimization problem in (17) for each slice $\overrightarrow{\boldsymbol{\mathcal{X}}}_t$ with stochastic gradient descent. From the results in Proposition II.11 and Proposition IV.1, we express each $\mathcal{L}_t(\boldsymbol{\mathcal{U}})$ in each slice at time t as a $(r, r, 1)$-tubal BTD. Let $\overline{\mathcal{L}}_t$ denote $\mathcal{L}_t$ in terms of the Fourier variables, and recall $\overline{\boldsymbol{U}}$ denotes the block-diagonal matrix representation of $\overline{\boldsymbol{\mathcal{U}}} = \boldsymbol{\mathcal{U}} \times_3 \boldsymbol{F}_{d_3}$, where $\overline{\boldsymbol{U}}_k$ is the k^{th} block on its diagonal of sizes $d_1 \times r$. Let $\overline{\boldsymbol{U}}_k$ and $\bar{\boldsymbol{w}}_{t,k}(\overline{\boldsymbol{U}}) \in \mathbb{C}^r$ for all $k \in [d_3]$ be the frontal faces of the tensors $\overline{\boldsymbol{\mathcal{U}}}$ and the optimal $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}) \times_3 \boldsymbol{F}_{d_3}$, respectively. Then denoting $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}}) = [\bar{\boldsymbol{w}}'_{t,1}(\overline{\boldsymbol{U}}) \cdots \bar{\boldsymbol{w}}'_{t,d_3}(\overline{\boldsymbol{U}})]'$, we can write $\overline{\mathcal{L}}_t$ as

$$\begin{aligned}
&\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}}) = \frac{1}{2} \left\| \boldsymbol{P}_{\Omega_t} \mathtt{vec}\Big(\overrightarrow{\boldsymbol{\mathcal{X}}}_t - \sum_{k=1}^{d_3} (\overline{\boldsymbol{U}}_k \bar{\boldsymbol{w}}_{t,k}(\overline{\boldsymbol{U}})) \circ \bar{\boldsymbol{f}}_r\Big) \right\|_2^2 \\
&= \frac{1}{2} \Big\| \boldsymbol{P}_{\Omega_t} \mathtt{vec}(\Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t)) \\
&-\boldsymbol{P}_{\Omega_t} \underbrace{\left[(\bar{\boldsymbol{f}}_1 \otimes \mathbf{I}_{d_1}) \quad \ldots \quad (\bar{\boldsymbol{f}}_{d_3} \otimes \mathbf{I}_{d_1})\right]}_{(\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1})} \begin{bmatrix} \overline{\boldsymbol{U}}_1 \bar{\boldsymbol{w}}_{t,1}(\overline{\boldsymbol{U}}) \\ \vdots \\ \overline{\boldsymbol{U}}_{d_3} \bar{\boldsymbol{w}}_{t,d_3}(\overline{\boldsymbol{U}}) \end{bmatrix} \Big\|_F^2 \\
&= \frac{1}{2} \Big\| \boldsymbol{\mathcal{F}}_{\Omega_t}\Big(\begin{bmatrix} \bar{\boldsymbol{x}}_{\Omega_t,1} \\ \vdots \\ \bar{\boldsymbol{x}}_{\Omega_t,d_3} \end{bmatrix} - \begin{bmatrix} \overline{\boldsymbol{U}}_1 & & 0 \\ & \ddots & \\ 0 & & \overline{\boldsymbol{U}}_{d_3} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{w}}_{t,1}(\overline{\boldsymbol{U}}) \\ \vdots \\ \bar{\boldsymbol{w}}_{t,d_3}(\overline{\boldsymbol{U}}) \end{bmatrix} \Big) \Big\|_2^2 .
\end{aligned}$$

Above, each $\bar{\boldsymbol{x}}_{\Omega_t,k} \in \mathbb{C}^{d_1}$ denotes the k^{th} frontal face of $\overline{\boldsymbol{\mathcal{X}}}_{\Omega_t} = \Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t) \times_3 \boldsymbol{F}_{d_3}$, where $\Delta_{\Omega_t}(\cdot)$ imputes zeros on the missing coordinates. $\boldsymbol{P}_{\Omega_t}$ is a subsampled identity matrix of size $|\Omega_t| \times d_1 d_3$, and $\boldsymbol{\mathcal{F}}_{\Omega_t} := \boldsymbol{P}_{\Omega_t}(\boldsymbol{F}'_{d_3} \otimes \mathbf{I}_{d_1}) \in \mathbb{C}^{|\Omega_t| \times d_1 d_3}$, which in the t-SVD framework is the subsampled inverse Fourier transform. The derivation of this relation using the t-SVD algebra is also shown in Appendix F.

Let us denote $\bar{\boldsymbol{x}}_t := \mathtt{vec}(\boldsymbol{\mathcal{X}}_{\Omega_t}) \in \mathbb{C}^{d_1 d_3}$. Using the result above, the objective (17) in t-product form then has the equivalent nested optimization problem in the Fourier domain:

$$\begin{aligned}
&\min_{[\overline{\boldsymbol{U}}_k] \in \overline{\mathcal{G}}(r,d_1) \ \forall k \in [d_3]} \frac{1}{T} \sum_{t=1}^{T} \overline{\mathcal{L}}_t(\overline{\boldsymbol{U}}), \\
&\text{where} \quad \overline{\mathcal{L}}_t(\overline{\boldsymbol{U}}) := \frac{1}{2} \| \boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}} \bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})) \|_2^2 \\
&\text{and} \quad \bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}}) = \underset{\bar{\boldsymbol{w}}_t \in \mathbb{C}^{d_3 r}}{\operatorname{argmin}} \frac{1}{2} \| \boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}} \bar{\boldsymbol{w}}_t) \|_2^2 .
\end{aligned} \tag{19}$$

The following proposition characterizes the smoothness of $\mathcal{L}(\boldsymbol{\mathcal{U}})$.

Proposition IV.1. *Suppose $|\Omega_t|$ is sufficiently large such that $(\boldsymbol{\mathcal{F}}_{\Omega_t} \overline{\boldsymbol{U}})'(\boldsymbol{\mathcal{F}}_{\Omega_t} \overline{\boldsymbol{U}})$ remains full rank. Then,*

(P1) *the inner least-squares problem* (18b) *has a unique solution.*

(P2) *$\mathcal{L}$ is a well-defined smooth function over $\mathcal{G}(r, d_1, d_3)$ and it admits a global optimizer.*

We leave the proof of Proposition IV.1 to Appendix E. At a high level, the proof shows that the (outer) optimization problem constrained to $[\boldsymbol{\mathcal{U}}] \in \mathcal{G}(r, d_1, d_3)$ is a well-defined problem on a product manifold of Grassmannians in the Fourier domain. Note that our assumption on $|\Omega_t|$ is identical to [5], [6].

The problem in (19) is nonconvex from the coupling of $\overline{\boldsymbol{U}}$ and $\bar{\boldsymbol{w}}_{\overline{\boldsymbol{U}}}$ and the orthonormality constraints $\overline{\boldsymbol{U}}_k' \overline{\boldsymbol{U}}_k = \mathbf{I}_r$ for all $k \in [d_3]$. We will minimize a problem over a product of d_3 Grassmannians $\mathcal{G}_1(r, d_1) \times \ldots \times \mathcal{G}_{d_3}(r, d_1)$ represented by $\overline{\boldsymbol{U}}$. For a single data observation, we first compute the unique minimizer $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ (equivalently $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$) to the inner least-squares problem, and then take a stochastic gradient descent step in the negative gradient direction of $\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}})$ with respect to each block $\overline{\boldsymbol{U}}_k$ on the diagonal of $\overline{\boldsymbol{U}}$ for estimating $\overline{\boldsymbol{\mathcal{U}}}$ (equivalently $\boldsymbol{\mathcal{U}}$).

C. Updating $\boldsymbol{\mathcal{U}}$

To update our estimate of the free submodule $\boldsymbol{\mathcal{U}}$, we perform a gradient descent step on each Grassmannian in the Fourier domain. We compute the gradient of the objective function $\overline{\mathcal{L}}_t$ with respect to each $\overline{\boldsymbol{U}}_k$ and then follow this gradient along a short geodesic curve on the Grassmannian [5]. Substituting the expression for the unique closed-form solution $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ into the objective, where $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}}) = \operatorname{argmin}_{\bar{\boldsymbol{w}}_t \in \mathbb{C}^{d_3 r}} \frac{1}{2} \| \boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}} \bar{\boldsymbol{w}}_t) \|_2^2$, we find the partial derivatives of $\overline{\mathcal{L}}_t$ with respect to $\overline{\boldsymbol{U}}$:

$$\begin{aligned}
\frac{\partial \overline{\mathcal{L}}_t}{\partial \overline{\boldsymbol{U}}} &= -\boldsymbol{\mathcal{F}}'_{\Omega_t} \boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}} \bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})) \bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})' \\
&:= -\boldsymbol{\mathcal{F}}'_{\Omega_t} \boldsymbol{\mathcal{F}}_{\Omega_t} \bar{\boldsymbol{r}}_t \bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})'.
\end{aligned} \tag{20}$$

See Appendix C for the derivation of the gradient. When computing the gradient, $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ is solved for as detailed in the next subsection.

Remark IV.2. *Note that the partial derivative* (20) *derived for the nested problem* (19) *should not be confused with the case where* $\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}}) = \frac{1}{2}\|\boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}}\hat{\boldsymbol{w}}_t)\|_2^2$ *for some fixed* $\hat{\boldsymbol{w}}_t$*. In that case, the gradient with respect to* $\overline{\boldsymbol{U}}$ *is also* $\frac{\partial \overline{\mathcal{L}}_t}{\partial \overline{\boldsymbol{U}}} = -\boldsymbol{\mathcal{F}}'_{\Omega_t}\boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}}\hat{\boldsymbol{w}}_t)\hat{\boldsymbol{w}}'_t$*. However, the difference in our case is that, under the first assumption of Proposition IV.1, the weights* $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}}) = (\overline{\boldsymbol{U}}'\boldsymbol{\mathcal{F}}'_{\Omega_t}\boldsymbol{\mathcal{F}}_{\Omega_t}\overline{\boldsymbol{U}})^{-1}\overline{\boldsymbol{U}}'\boldsymbol{\mathcal{F}}'_{\Omega_t}$ *are a function of* $\overline{\boldsymbol{U}}$*; See Appendix C for further details.*

Using the work in [16], the gradient on the product of Grassmannians in Fourier space is given by

$$\nabla\overline{\mathcal{L}}_t = \mathcal{P}_{\mathcal{D}}\left((\mathbf{I} - \overline{\boldsymbol{U}}\,\overline{\boldsymbol{U}}')\frac{\partial \overline{\mathcal{L}}_t}{\partial \overline{\boldsymbol{U}}}\right), \tag{21}$$

where $\mathcal{P}_{\mathcal{D}}(\cdot)$ sets the non-block-diagonal entries of the gradient to zero. The gradient of the objective on the product of Grassmannians then has the form (using $\bar{\boldsymbol{w}}$ instead of $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ for ease of notation and indexing the d_3 blocks of $\bar{\boldsymbol{w}}$)

$$\nabla\overline{\mathcal{L}}_t = \begin{bmatrix} -\bar{\gamma}_1\bar{\boldsymbol{w}}'_1 & & 0 \\ & \ddots & \\ 0 & & -\bar{\gamma}_{d_3}\bar{\boldsymbol{w}}'_{d_3} \end{bmatrix} \in \mathbb{C}^{d_1 d_3 \times d_3 r}, \tag{22}$$

where

$$\nabla\overline{\mathcal{L}}_{t,k} = -\bar{\gamma}_k\bar{\boldsymbol{w}}'_k \in \mathbb{C}^{d_1\times r} \tag{23}$$

$$\bar{\gamma}_k = \left(\mathbf{I} - \overline{\boldsymbol{U}}_k\overline{\boldsymbol{U}}'_k\right)\bar{\boldsymbol{r}}_{\Omega_t,k} \tag{24}$$

$$\bar{\boldsymbol{r}}_{\Omega_t} = \boldsymbol{\mathcal{F}}'_{\Omega_t}\boldsymbol{\mathcal{F}}_{\Omega_t}\bar{\boldsymbol{r}}_t = \texttt{unfold}(\texttt{fft}(\Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{R}}}_t), [], 3)). \tag{25}$$

Here, $\overrightarrow{\boldsymbol{\mathcal{R}}}_t = \Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t) - \overrightarrow{\boldsymbol{\mathcal{P}}}_t$, $\overrightarrow{\boldsymbol{\mathcal{P}}}_t = \boldsymbol{\mathcal{U}} * \overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$, $\Delta_{\Omega_t}(\cdot)$ imputes zeros on the unobserved tensor entries, and $\texttt{fft}(\cdot, [], 3)$ takes the Fourier transform along the third-mode tubes.

A gradient step along each geodesic in the product manifold with tangent vector $-\nabla\overline{\mathcal{L}}_{t,k}$ is given by Equation (2.65) in [16] and is a function of the singular values and vectors of $\nabla\overline{\mathcal{L}}_{t,k}$ [5]. Each $\nabla\overline{\mathcal{L}}_{t,k}$ has the rank-one SVD:

$$\nabla\overline{\mathcal{L}}_{t,k} = \begin{cases} \boldsymbol{u}_k\sigma_k\boldsymbol{v}'_k, & k = 1, \ldots, \lceil\frac{d_3+1}{2}\rceil \\ \texttt{conj}(\nabla\overline{\mathcal{L}}_{t,(d_3-k+2)}), & k = \lceil\frac{d_3+1}{2}\rceil + 1, \ldots, d_3 \end{cases} \tag{26}$$

$$\boldsymbol{u}_k = \frac{-\bar{\gamma}_k}{\|\bar{\gamma}_k\|}, \quad \boldsymbol{v}'_k = \frac{\bar{\boldsymbol{w}}'_k}{\|\bar{\boldsymbol{w}}_k\|}, \quad \sigma_k := \|\bar{\gamma}_k\|\|\bar{\boldsymbol{w}}_k\|.$$

From [16], a rank-one step of length $\eta > 0$ in the direction $-\nabla\overline{\mathcal{L}}_{t,k}$ is given by

$$\overline{\boldsymbol{U}}_{t+1,k} = \tag{27}$$
$$\overline{\boldsymbol{U}}_{t,k} + \left(\sin(\sigma_k\eta_k)\frac{\bar{\gamma}_k}{\|\bar{\gamma}_k\|} + (\cos(\sigma_k\eta_k) - 1)\frac{\bar{\boldsymbol{p}}_k}{\|\bar{\boldsymbol{p}}_k\|}\right)\frac{\bar{\boldsymbol{w}}'_k}{\|\bar{\boldsymbol{w}}_k\|},$$

where $\bar{\boldsymbol{p}}_k = \overline{\boldsymbol{U}}_k\bar{\boldsymbol{w}}_k$ is the k^{th} frontal face of $\overline{\boldsymbol{\mathcal{P}}}_t = \overrightarrow{\boldsymbol{\mathcal{P}}}_t \times_3 \boldsymbol{F}_{d_3}$ (equivalently, in block diagonal matrix form, the k^{th} block element of $\overline{\boldsymbol{P}}_t = \overline{\boldsymbol{U}}_t\overline{\boldsymbol{W}}_t(\overline{\boldsymbol{U}}_t)$, where $\overline{\boldsymbol{W}}_t(\overline{\boldsymbol{U}}_t)$ is the block-diagonal matrix formed from the $\bar{\boldsymbol{w}}_k$). Using conjugate symmetry of the Fourier transform, $\overline{\boldsymbol{U}}_k = \texttt{conj}(\overline{\boldsymbol{U}}_{(d_3-k+2)})$, $k = \lceil\frac{d_3+1}{2}\rceil + 1, \ldots, d_3$.

Following from the result in [55], we use a greedy step size $\eta_k = \arctan(\|\bar{\gamma}_k\|/\|\bar{\boldsymbol{w}}_k\|)$ on each Grassmannian. This choice of step size adaptively changes based on the FSM fit to the data, growing proportionally based on the angle between the projection and its residual. Using principles of conjugate symmetry of the FFT, we can save time by only computing the matrix-vector multiplications on half of the frontal slices in the Fourier domain and using the complex conjugate to find the others.

D. Computing the weights $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$

For the gradient computations in $\overline{\boldsymbol{U}}$, we first require computing

$$\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}}) = \underset{\bar{\boldsymbol{w}}_t \in \mathbb{C}^{d_3 r}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}}\bar{\boldsymbol{w}}_t)\|_2^2. \tag{28}$$

If we were to solve for the optimal $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ in closed form, this would require forming and inverting the $d_3 r \times d_3 r$ matrix $\overline{\boldsymbol{U}}'\boldsymbol{\mathcal{F}}'_{\Omega_t}\boldsymbol{\mathcal{F}}_{\Omega_t}\overline{\boldsymbol{U}}$, which can be very large. Instead, the blockwise separable structure of this quadratic problem suggests we use conjugate gradient descent (CGD) to estimate $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ for a fixed $\overline{\boldsymbol{U}}$. The structure permits fast, efficient computations by matrix-vector products $\overline{\boldsymbol{U}}'\boldsymbol{\mathcal{F}}'_{\Omega_t}\boldsymbol{\mathcal{F}}_{\Omega_t}\overline{\boldsymbol{U}}\boldsymbol{v}$ for some vector $\boldsymbol{v} \in \boldsymbol{F}^{d_3 r}$. Within the t-SVD, this involves FFTs, separable matrix-vector products in each slice (from the block diagonal structure of $\overline{\boldsymbol{U}}$), and zero-padding.

We observe faster overall convergence of our algorithm when solving the problem in $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ at each time step with higher accuracy. CGD is guaranteed to converge in as many iterations as the dimension of the optimized vector [45], but since $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ is $d_3 r$-dimensional, the number of maximum iterations could be rather large. As the number of missing entries increases, the matrix $\boldsymbol{\mathcal{F}}_{\Omega_t}\overline{\boldsymbol{U}}$ in the least squares problem of Eq. (28) becomes more poorly conditioned, and since the convergence rate of CGD is dependent on the condition number of this matrix, denoted $\kappa(\boldsymbol{\mathcal{F}}_{\Omega_t}\overline{\boldsymbol{U}})$, the algorithm will require more iterations to solve the problem to within some $\epsilon > 0$ accuracy, slowing the run-time of our algorithm. However, as noted above, it is impractical to form and store the large matrix, much less compute its SVD to find κ. We prove a practical upper bound on the number of CGD iterations as a function of the sampling rate and show CGD converges in far fewer iterations than the maximum for most subsampling rates. The proof, along with accompanying lemmas, is left to Appendix A. Empirical studies of our algorithm show the number of maximum CGD iterations is tightly bounded by our Theorem IV.4 for most subsampling rates.

For the following theorem, we will require a notion of tensor coherence, given in [58]:

Definition IV.3 (Tensor coherence)**.** *Let* $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{d_1\times r\times d_3}$ *be an orthonormal tensor whose* r *lateral slices span the free submodule of* $\mathbb{M}_{d_3}^{d_1}$*. Then, the* μ*-coherence of* $\boldsymbol{\mathcal{U}}$ *is given by*

$$\mu(\boldsymbol{\mathcal{U}}) := \max_{i=1,\ldots,d_1} \|\boldsymbol{\mathcal{U}}^T * \mathring{\boldsymbol{e}}_i\|_2^2, \tag{29}$$

where $\mathring{\mathbf{e}}_i \in \mathbb{R}^{d_1 \times 1 \times d_3}$ is the column basis with $\mathring{\mathbf{e}}_{i11} = 1$ and the rest of the entries are zero. Note that $\frac{r}{d_1 d_3} \leq \mu(\boldsymbol{\mathcal{U}}) \leq 1$.

It is standard practice in matrix and tensor completion literature to make some assumption that the coherence is not too large to guarantee recovery (see Candes & Recht [10] for matrix completion and Zhang & Aeron [58] who consider tensor completion under the t-product.) We will impose a coherence upper bound assumption on all of the iterates of $\boldsymbol{\mathcal{U}}_t$ as well as a sampling condition for the number of entries per slice that must be observed.

Theorem IV.4. *Let $\boldsymbol{P}_{\Omega_t}$ sample $|\Omega_t|$ rows from $(\boldsymbol{F}'_{d_3} \otimes \mathbf{I}_{d_1})\overline{\boldsymbol{U}}$ uniformly at random such that $|\Omega_t|/\log(|\Omega_t|) > C^2 \mu_0 d_3 r$, where C is a universal constant and $\mu_0 > 1/d_3$ is small. Assume coherence of the $\boldsymbol{\mathcal{U}}_t$ iterates remains bounded as $\mu(\boldsymbol{\mathcal{U}}_t) \leq \frac{\mu_0 r}{d_1}$. Then with probability at least $1 - \delta$, where $\delta \in [0, 1]$, the maximum number of conjugate gradient descent iterations, J, required to solve* (28) *to within ϵ-precision for $\epsilon > 0$ is upper bounded as:*

$$J \leq \frac{1}{2}\sqrt{\frac{1 + \delta^{-1}\tau}{1 - \delta^{-1}\tau}} \log(2/\epsilon), \tag{30}$$

$$\textit{where } \tau = C\sqrt{\mu_0 r d_3 \frac{\log(|\Omega_t|)}{|\Omega_t|}}.$$

The proof is found in Appendix A.

E. Algorithm

The preceding updates give an efficient algorithm we call TOUCAN (Tensor rank-One Update on the Complex grassmanniAN) for computing each variable in the Fourier domain with simple, efficient linear algebra operations and fast Fourier transforms. TOUCAN is numerically stable by maintaining orthonormality on the product of Grassmannians and is constant in memory use, scaling linearly with the number of observed data samples instead of in polynomial-time like batch t-SVD methods. In addition, like other t-SVD algorithms, independent computations in each slice (equivalently the blocks of the block-matrix terms) in the Fourier domain can be carried out in parallel. TOUCAN is summarized in Algorithm 1.

TOUCAN can handle two cases of online and streaming data. The first is incremental batch completion where the batch tensor is too large to read into local memory, but can be stored elsewhere. Our algorithm reads only slice $\overrightarrow{\boldsymbol{\mathcal{X}}}_t$ into local memory, updates its estimate of $\boldsymbol{\mathcal{U}}$ and weights $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$, discards this local copy of $\overrightarrow{\boldsymbol{\mathcal{X}}}_t$, and passes over each data slice like this in sequence. In this setting, it is possible to make multiple passes over the full batch while only reading parts into memory. This is a sensible approach when the underlying low-rank model is believed to be static or stationary throughout the batch. The second use case of TOUCAN is for purely streaming data where we seek to learn $\boldsymbol{\mathcal{U}}$ from each new observation and discard each observation completely after processing. The algorithm then tracks any changes in $\boldsymbol{\mathcal{U}}$ only from new observations and is able to track a time-varying low-rank model.

Algorithm 1 Tensor rank-One Update on the Complex grassmanniAN (TOUCAN): Arbitrary Missing Tensor Entries

```
Require: Data: $\overrightarrow{\boldsymbol{\mathcal{X}}}_t \in \mathbb{R}^{d_1 \times 1 \times d_3} \quad \forall i = 1, \dots, T$ observed
    on $\Omega_t$; tubal-rank $r$, tolerance $\epsilon > 0$.
Initalize Fourier transformed orthonormal tensor $\overline{\boldsymbol{\mathcal{U}}}_0 \in \mathbb{C}^{d_1 \times r \times d_3}$.
for $t = 1$ to $T$ do
    Compute $\overline{\boldsymbol{\mathcal{X}}}_{\Omega_t} = \texttt{fft}(\Delta_{\Omega_t}(\boldsymbol{\mathcal{X}}_{\Omega_t}), [], 3)$.
    Estimate optimal weights $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ by solving Eq. (28)
        with CGD to within tolerance $\epsilon > 0$.
    Predict full slice in the Fourier domain:
        $\overline{\boldsymbol{P}}_t = \overline{\boldsymbol{U}}_t \overline{\boldsymbol{W}}_t(\overline{\boldsymbol{U}}_t)$.
    Shape into tensor and transform: $\overrightarrow{\boldsymbol{\mathcal{P}}}_t = \texttt{ifft}(\overline{\boldsymbol{\mathcal{P}}}_t, [], 3)$.
    Compute residual: $\overrightarrow{\boldsymbol{\mathcal{R}}}_t = \Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t) - \overrightarrow{\boldsymbol{\mathcal{P}}}_t$.
    Compute gradient terms from Eqs. (24) and (25).
    Update subspace: $\overline{\boldsymbol{\mathcal{U}}}_{t+1}$ from (27).
    Transform: $\boldsymbol{\mathcal{U}}_{t+1} = \texttt{ifft}(\overline{\boldsymbol{\mathcal{U}}}_{t+1}, [], 3)$.
    Transform: $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}_t) = \texttt{ifft}(\overline{\boldsymbol{\mathcal{W}}}_t(\overline{\boldsymbol{\mathcal{U}}}_t), [], 3)$.
end for
return $\boldsymbol{\mathcal{U}}, \overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}_t), \quad \forall t = 1, \dots, T$
```

F. Memory and computational analysis

TOUCAN processes a tensor incrementally and thus only needs to store an orthonormal tensor $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{d_1 \times r \times d_3}$, the weights $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}) \in \mathbb{R}^{r \times 1 \times d_3}$ per $t = 1, \dots, T$, requiring $d_1 d_3 r + d_3 r$ memory elements per iteration at time t. Upon updating $\boldsymbol{\mathcal{U}}$, the new $\overrightarrow{\boldsymbol{\mathcal{W}}}_{t+1}(\boldsymbol{\mathcal{U}})$ is computed at the next iteration using the same memory. At each time instance, this is far less than storing the entire tensor in memory which would require $d_1 T d_3$ memory elements, especially when any of the dimensions is very large.

Implemented efficiently, the main loop of our algorithm requires 4 fast inverse Fourier transforms and one fast Fourier Transform. The CGD update takes $O(J(N d_1 r + d_1 d_3 \log(d_3))$ flops where $N = \lceil (d_3 + 1)/2 \rceil$ and J is the number of CGD iterations. Computing $\overrightarrow{\boldsymbol{\mathcal{R}}}$ takes $O(N d_1 r + d_1 d_3 \log(d_3) + d_1 d_3)$ flops. The update in (25) takes $O(d_1 d_3 \log(d_3))$ flops, and (24) takes $O(N d_1 r)$ flops. Computing the subspace update requires $O(N d_1 r)$ flops. Table I summarizes the memory and computational requirements of our algorithm compared to other t-SVD algorithms.

G. Convergence

Here, we prove expected linear local convergence of our algorithm. Our analysis follows naturally from the work in [6], which proved a similar result for the GROUSE algorithm, for which TOUCAN is a related extension to t-product tensors. The problem of proving convergence for this class of algorithms is complicated by the setting of streaming data, missing entries, and the optimization problem being constrained to a nonlinear manifold with rank-one updates. Other analyses for related problems may exist like in [25], but often these are limited to the cases of batch data, fully observed entries, and a specific type of retraction operator on the manifold. However, little work in the literature exists for finite-sample

Algorithm	Memory per iteration	Total batch computational complexity	Computation per iteration
TOUCAN	$O(d_1 r d_3 + r d_3)$	$O(B(J(d_1 T d_3 \log(d_3) + d_1 r T N)))$	$O(J(d_1 d_3 \log(d_3) + d_1 r N))$
TCTF	$O(d_1 r d_3 + r T d_3)$	$O(A(d_1 T d_3 \log(d_3) + d_1 r T N))$	$O(d_1 T d_3 \log(d_3) + d_1 r T N)$
TNN-ADMM	$O(d_1 T d_3)$	$O(A(d_1 T d_3 \log(d_3) + d_1 T N \min(d_1, T)))$	$O(d_1 T d_3 \log(d_3) + d_1 T N \min(d_1, T))$

TABLE I: Algorithm memory and computational complexities. A here denotes the number of algorithm iterations for batch methods, and B denotes the number of batch passes for TOUCAN. Here usually $B \ll A$.

analysis with missing data, and the only results for streaming subspace estimation with missing data in the matrix case are local convergence results for the GROUSE algorithm in [6] and the work in [14]; see [4] for a recent survey of the area.

Our problem and analysis are tailored to these specific settings, and we apply similar assumptions as made in [6] and other standard assumptions made in the literature. For simplicity of analysis, we focus on the case of tubes sampled uniformly at random since it allows us to extend the results provided in [6] to the tensor case.

Our theory provides expected linear local convergence under (i) the randomness of the observed tensor and (ii) the randomness of the subset of elements observed at each iteration. More specifically, we have the following assumptions:

A1. *Each $\overrightarrow{\boldsymbol{\mathcal{X}}}_t = \boldsymbol{\mathcal{U}}^* * \overrightarrow{\boldsymbol{\mathcal{S}}}_t$ for planted model $[\boldsymbol{\mathcal{U}}^*] \in \mathcal{G}(r, d_1, d_3)$ and $(\overrightarrow{\boldsymbol{\mathcal{S}}}_t)_{ijk} \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$.*

A2. *Let $\Omega_t \subset \{1, \ldots, d_1\}$ denote the tube indices, and assume the tube indices are chosen uniformly at random. In other words, the data $\overrightarrow{\boldsymbol{\mathcal{X}}}_t$ are sampled tubal-wise, where a tube consists of d_3 entries along the third mode dimension.*

It is worth mentioning that A1 is a generalization of the assumption made in [6] to the tensor problem. We also note that conditions analogous to A2 have been used in [35] for t-product based tensor completion tasks.

Before providing the main result, we give some additional notations and definitions. Let $\mu(\overline{\boldsymbol{U}})$ denote the matrix coherence of complex orthonormal matrix $\overline{\boldsymbol{U}} \in \mathbb{C}^{d \times r}$, i.e. $\mu(\overline{\boldsymbol{U}}) := \frac{d}{r} \max_{i=1,\ldots,d} \|\overline{\boldsymbol{U}}' \boldsymbol{e}_i\|_2^2$, where $\boldsymbol{e}_i$ is the i^{th} standard basis unit vector in $\mathbb{C}^d$. We note this is consistent with Definition IV.3. For a vector argument $\bar{\boldsymbol{x}} \in \mathbb{C}^d$, this further specializes to $\mu(\bar{\boldsymbol{x}}) = d\|\bar{\boldsymbol{x}}\|_\infty^2 / \|\bar{\boldsymbol{x}}\|_2^2$. We use $\mu_{\max}(\boldsymbol{\mathcal{U}}^*) := \max_{k=1,\ldots,d_3} \mu(\overline{\boldsymbol{U}}_k^*)$ to denote the maximum coherence of the frontal slices of $\overline{\boldsymbol{\mathcal{U}}}$.

Denote

$$\epsilon_{t,k} := r - \|\overline{\boldsymbol{U}}_k^{*\prime} \overline{\boldsymbol{U}}_{t,k}\|_F^2, \quad \forall k \in [d_3]. \tag{31}$$

We will analyze the sequence ϵ_t measuring the error between the planted model $\boldsymbol{\mathcal{U}}^*$ and the algorithm's estimate $\boldsymbol{\mathcal{U}}_t$:

$$\begin{aligned} \epsilon_t &:= d_3 r - \|\boldsymbol{\mathcal{U}}^{*\prime} * \boldsymbol{\mathcal{U}}_t\|_F^2 \\ &= d_3 r - \|\overline{\boldsymbol{U}}^{*\prime} \overline{\boldsymbol{U}}_t\|_F^2 = \sum_{k=1}^{d_3} \epsilon_{t,k}, \end{aligned} \tag{32}$$

where $\overline{\boldsymbol{U}}^*$ (resp. $\overline{\boldsymbol{U}}_t$) denotes the current estimate of the block-diagonal Fourier representation of $\boldsymbol{\mathcal{U}}^*$(resp. $\boldsymbol{\mathcal{U}}_t$). Again, here we use the normalized DFT matrix when taking the Fourier transform. The second equality follows from the definition of the Frobenius norm under the t-product. If $\boldsymbol{\mathcal{U}}_t$ perfectly estimates the free submodule spanned by $\boldsymbol{\mathcal{U}}^*$, it's easy to see from (32) that $\epsilon_t = 0$; on the other hand, if $\boldsymbol{\mathcal{U}}_t$ is orthogonal to $\boldsymbol{\mathcal{U}}^*$ in t-product, $\epsilon_t = d_3 r$.

Theorem IV.5. *Let $\{(\overrightarrow{\boldsymbol{\mathcal{W}}}_{t-1}(\boldsymbol{\mathcal{U}}_{t-1}), \boldsymbol{\mathcal{U}}_t)\}_{t \geq 1}$ denote the sequence generated by Algorithm 2. Suppose A1 and A2 hold, and the number of sampled tubes $|\Omega_t| \geq q$ for all t such that*

$$q \geq C_1 \log(d_1)^2 r \mu_{\max}(\boldsymbol{\mathcal{U}}^*) \log(20 r d_3) \tag{33}$$

for some $C_1 \geq 64/3$. Suppose there exists $\bar{\delta} \in (0, 0.6/d_3)$ such that the residual vector $\bar{\boldsymbol{v}}_{t,k} := \bar{\boldsymbol{x}}_{t,k} - \overline{\boldsymbol{U}}_{t,k} \overline{\boldsymbol{U}}_{t,k}' \bar{\boldsymbol{x}}_{t,k}$ satisfies

$$\mu(\bar{\boldsymbol{v}}_{t,k}) \leq \log d_1 \left(\frac{0.045}{\log(10 d_3)} C_1 r \mu(\overline{\boldsymbol{U}}_{t,k}) \log(20 r d_3) \right)^{\frac{1}{2}}, \tag{34a}$$

$$\mu(\bar{\boldsymbol{v}}_{t,k}) \leq (\log d_1)^2 \frac{0.05}{8 \log(10 d_3)} C_1 \log(20 r d_3), \tag{34b}$$

for all $k = 1, \ldots, d_3$ each with probability at least $1 - \bar{\delta}$. Assume further that

$$\epsilon_{t,k} \leq \min \left\{ \frac{q^2}{128 d_1^2 r}, \frac{r}{16 d_1} \mu(\overline{\boldsymbol{U}}_k^*) \right\}, \ \forall k \in [d_3], \tag{35a}$$

$$\epsilon_t \leq (8 \cdot 10^{-6})(0.6 - d_3 \bar{\delta})^2 \frac{q^3}{d_1^3 r^2}. \tag{35b}$$

Then,

$$\mathbb{E}[\epsilon_{t+1} | \epsilon_t] \leq \left(1 - 0.16(0.6 - d_3 \bar{\delta}) \frac{q}{d_1 r} \right) \epsilon_t. \tag{36}$$

Remark IV.6. *We note that in the matrix case, i.e., $d_3 = 1$, our result recovers [6, Corollary 2.15]. We also note that the failure probability increases as d_3 grows which is similar to the results provided in the t-SVD literature under the tubal sampling assumption; see, e.g., [58, Theorem 3.2].*

Remark IV.7. *The analogous supposition that Equations* (34a) *and* (34b) *hold was also made in [6] for the matrix case, where they pointed out that empirical evidence supports this assumption. This is essentially assuming that the residual vectors are roughly as incoherent as the subspaces themselves. The assumptions in* (35a) *and* (35b) *define the local region in which the expected linear rate of convergence is achieved. As discussed in [6], this local region is conservative according to empirical evidence, which is also supported by the experiments in our own work.*

Remark IV.8. *In* (36), *the expectation is with respect to the randomness in the data drawn from $\boldsymbol{\mathcal{U}}^*$ with normally distributed coefficients, and the high probability result is with respect to the tubes observed at random. Supposing that we are within the radius of local convergence, the rate* (36) *suggests our algorithm converges faster the closer $\frac{q}{d_1 r}$ is to 1, and the* *fastest when we observe fully sampled data. Conversely, with fewer tube observations q, the rate of convergence slows.*

V. Experimental Results

A. Numerical experiments

1) Incremental Tensor Completion: We first verify the validity and efficiency of TOUCAN in recovering large-scale missing tensor data from synthetically generated isotropic Gaussian distributions with low-tubal-rank. We compute the t-product of two low-tubal-rank tensors $\boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{W}}$ to yield a third order tensor of tubal-rank $r = 3$ and sample 20% of tensor entries/tubes randomly according to a Bernoulli distribution. TOUCAN observes one lateral slice sequentially, solves the inner CGD step to within a set tolerance (10^{-6}), and is allowed to process over the entire batch twice. Our simulations compare against t-SVD batch tensor completion algorithms: 1) an algorithm that optimizes tensor nuclear norm via alternating direction method of multipliers (TNN-ADMM) [58], and 2) Tensor Completion by Tensor Factorization (TCTF) of [62], which factorizes the tensor for the t-product of two low-tubal-rank tensors. For TCTF, we omit the rank-reduction steps and set the multi-ranks equal to the planted tubal-rank since our synthetic examples are generated in this manner, and the steps only add computation; this also makes TCTF more comparable to TOUCAN since they both seek a similar nonconvex factorization under the t-product. Fig. 2(a)-(d) plot the normalized root-mean-squared error (NRMSE) $\|\boldsymbol{\mathcal{X}}_{\text{est.}} - \boldsymbol{\mathcal{X}}_{\text{true}}\|_F / \|\boldsymbol{\mathcal{X}}_{\text{true}}\|_F$ of the recovered tensor to the true tensor by median elapsed wall clock time in seconds over 10 trials. In addition, we also examine cases with additive white Gaussian noise. All algorithms are coded in Python with our optimized implementations of TNN-ADMM, TCTF, and STC; for OLSTEC and TeCPSGD, we use the implementations from [27] converted to Python. Experiments were run on a Intel(R) Core(TM) i7-6850K CPU @ 3.60GHz. Our implementation can be found at https://github.com/kgilman/TOUCAN.

For tensors with large d_2 dimension, TOUCAN can rapidly complete the data in substantially less time than either batch algorithm while using only 0.3% of the memory per iteration compared to storing the entire tensor for our synthetic example. Fig. 2 shows the algorithm scales up well with the tensor dimensions, and can achieve batch completion for large-scale tensors in orders of magnitude less computation time. With additive Gaussian noise, our stochastic gradient algorithm achieves accuracy to within a noise floor proportional to the noise variance. For smaller size tensors, the batch algorithms succeed in less wall clock time, so TOUCAN only becomes advantageous when the dimensions of the tensor scale to be very large. With larger amounts of additive noise, the algorithm's advantage diminishes since the batch algorithms are able to more quickly average the noise out than our stochastic algorithm. Since the first observations in the initial phase of the algorithm will be revisited in later passes over the batch, the NRMSE curve with respect to the entire tensor shows slower progress for TOUCAN in the first iterations, but this graph under-reports the accuracy of the estimate of $\boldsymbol{\mathcal{U}}$.

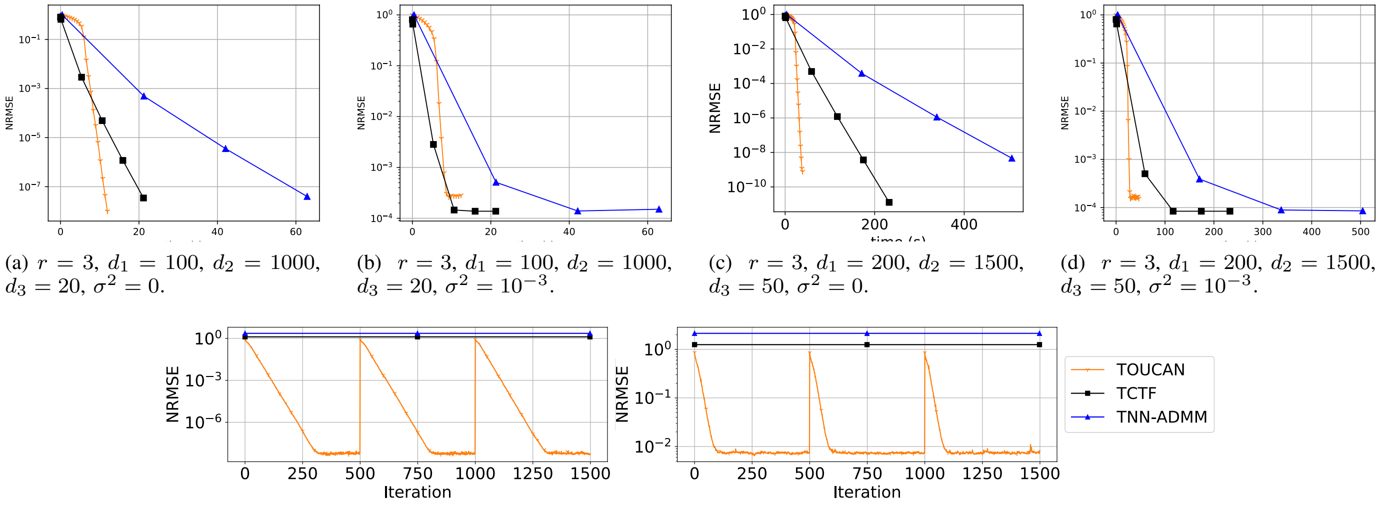
2) Dynamic FSM Tracking: We demonstrate TOUCAN's ability to track a dynamically changing FSM from streaming multiway data with missing entries. We generate a random orthonormal basis $\boldsymbol{\mathcal{U}}$ for various tubal-ranks from an i.i.d. Gaussian distribution and draw 2-D lateral slices by t-product with i.i.d. Gaussian weights. 50% of the tensor entries are sampled at random, and we record the NRMSE of the completed tensor slice $\|\overrightarrow{\boldsymbol{\mathcal{X}}}_{t,\text{est.}} - \overrightarrow{\boldsymbol{\mathcal{X}}}_{t,\text{true}}\|_F / \|\overrightarrow{\boldsymbol{\mathcal{X}}}_{t,\text{true}}\|_F$. The experiment simulates abrupt system dynamics by randomly reinitializing the underlying FSM every 500 slices. The results in Fig. 2(e),(f) illustrate TOUCAN's ability to adaptively re-estimate each new FSM and capture system dynamics that the batch algorithms cannot, as they compute estimates based on the entire batch of data collected over time.

B. Real data experiments

1) Application to Gas Measurements Tensor: We deploy our algorithm to track a dynamically changing free submodule from streaming 2-D lateral slice data with missing entries in chemo-sensing data collected by Vergara et al. [51]. The dataset consists of measurements as a gas is blown over an array of conductometric metal-oxide sensors in a wind tunnel [31]. The data is made up of six arrays each with 72 sensors, 260 seconds of data points collected at ~ 100 Hz, and 300 experiments for each of 11 gases. The sensor values vary in time as a gas permeates throughout a wind tunnel and then dissipates [31]. We chose to fix the array and gas, using the fourth sensor array and Toulene gas for our experiments, downsample to 10 Hz, and remove sensor 33 (out of 72) and time samples 1103 and 2012, which seemed to have erratic measurements, resulting in a tensor of size $300 \times 2600 \times 71$. We subtract the sample mean from the columns of each time slice – a column referring to 300 experiment samples per sensor – normalize each time slice by its Frobenius norm, and subsample only 25% of the data to simulate missing entries.

TOUCAN is compared to the batch t-SVD algorithms, the two online CP algorithms TeCPSGD and OLSTEC (we use the source code from [27] for our implementations of OLSTEC and TeCPSGD), and the online tensor completion algorithm Sequential Tensor Completion [54], which estimates an orthonormal rank-r unfolding for each mode, to recover undersampled chemo-sensing data. The online algorithms process each time slice sequentially, observing a 300×71 matrix of experiments versus sensor channels, and pass over the entire data once. We empirically found tracking a 1-dimensional FSM with TOUCAN to have the best performance. The algorithm updates its estimate of $\boldsymbol{\mathcal{U}}_{t+1}$, and weights $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}_t)$.

For competing algorithms, we tuned parameters by grid search (see Appendix I) to find the best performance in NRMSE on the first 300 time samples. The online CP algorithms learn a rank-50 decomposition, with their factors initialized with the left-singular subspaces of the mode unfoldings of the first 300 samples. We set $\lambda = 10^{-5}$ and the initial step size to be 10^3 for TeCPSGD, and $\lambda = 0.9$ and $\mu = 10^{-8}$ for OLSTEC. STC learns a multirank (15, 15, 1) model. TCTF learns a tubal-rank 10 factorization, and the ADMM algorithm penalty is set to be $\rho = 1.5$ for TNN-ADMM. The batch



(a) $r = 3$, $d_1 = 100$, $d_2 = 1000$, $d_3 = 20$, $\sigma^2 = 0$.

(b) $r = 3$, $d_1 = 100$, $d_2 = 1000$, $d_3 = 20$, $\sigma^2 = 10^{-3}$.

(c) $r = 3$, $d_1 = 200$, $d_2 = 1500$, $d_3 = 50$, $\sigma^2 = 0$.

(d) $r = 3$, $d_1 = 200$, $d_2 = 1500$, $d_3 = 50$, $\sigma^2 = 10^{-3}$.

(e) $r = 3, d_1 = d_3 = 100$, $\sigma^2 = 0$.

(f) $r = 3, d_1 = d_3 = 100$, $\sigma^2 = 10^{-3}$.

Fig. 2: (a)-(d): Batch completion of t-SVD synthetic tensors with 20% entries observed and median wall-clock time over 10 trials on the x-axis. Markers are plotted every 100 TOUCAN iterations and 50 batch algorithm iterations. (e) & (f): TOUCAN completing a tensor from a dynamically changing FSM over time compared to batch completion t-SVD methods with 50% of the entries observed. Markers are plotted for every 50 TOUCAN iterations. The second dimension of the tensor passed to the batch algorithms TCTF and TNN-ADMM is equal to the number of iterations.

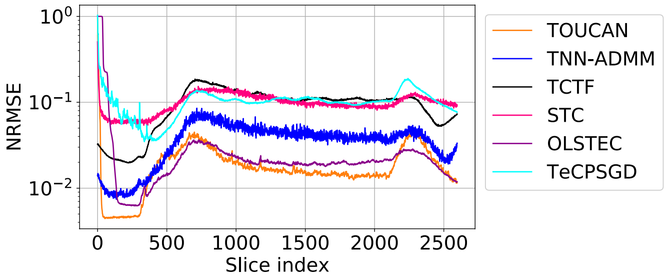


Fig. 3: NRMSE of each recovered time slice for Toluene gas dataset from 25% samples.

TABLE II: Total wall clock times in seconds for Toluene gas dataset

Algorithm	Time (s)
TOUCAN	30.81
TeCPSGD	137.31
STC	363.54
OLSTEC	552.53
TCTF	725.55
TNN-ADMM	879.78

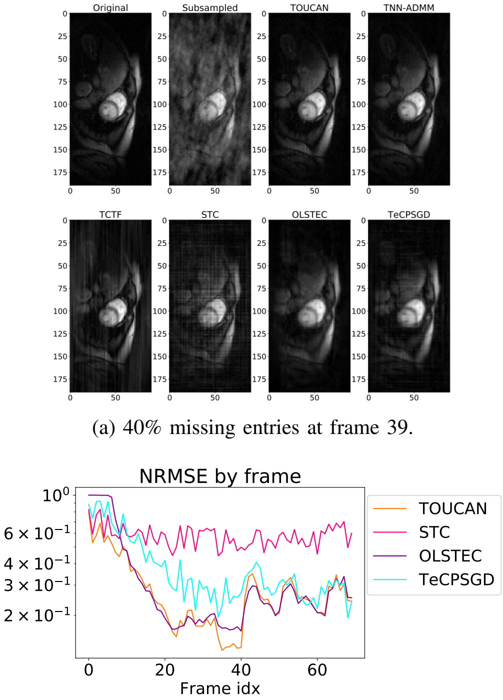
algorithms iterate until the difference NRMSE between iterates is less than 10^{-4} or a maximum number of 75 iterations is reached.

Fig. 3 compares the NRMSE of each recovered 2D slice to the true data at each time instance for the algorithms, which shows TOUCAN tracking the sensor readings with comparable error to OLSTEC. Due to the non-stationary behavior of the data, the tracking errors fluctuate as the data changes in time. While the batch methods achieve the best overall NRMSE error computed for the entire tensor, the online methods show the best reconstruction error on each sample after the initial start-up iterations. We also give the total computation time for each algorithm in Table II, emphasizing the significant speedup TOUCAN attains over the baseline algorithms, particularly the batch algorithms that are computationally prohibitive with large tensor data.

2) Streaming dynamic MRI reconstruction: Magnetic resonance imaging (MRI) collects a high-dimensional tensor that is often undersampled due to computational limitations exacerbated by large volumetric and dynamic acquisitions. One successful solution to image reconstruction from limited sampling is low-rank tensor completion [7], [38]. A t-SVD factorization of the spatial frequency-by-time (or k-t space) tensor reveals low-tubal-rank structure in the real and complex components [7], and t-SVD algorithms have been shown to be proficient at completing the k-t space tensor for image reconstruction. MRI data can also contain significant motion content and time-varying dynamics such as breathing motion. We employ TOUCAN's ability to track streaming time-dynamic multiway day to recover the k-t space tensor.

We test the completion abilities of each algorithm on the invivo myocardial perfusion dataset data from [33] with both varying levels of uniformly random entry sampling and tube sampling along the k_y direction. The dimensions of the data are $k_x = 190, k_y = 90$ and $k_t = 70$, and the data contains many dynamic motions such as heartbeats, breathing motion, and image intensity changes.

The streaming algorithms pass over the data once with the k-space rows oriented along the third tensor mode ($k_y = d_3$). TOUCAN learns a free submodule of tubal-rank 5, and two streaming CP algorithms learn a rank-50 CP decomposition. After exhaustive search for hyperparameters, we set $\lambda = 0.5$ and $\mu = 10^{-4}$ for OLSTEC, and $\lambda = 10^{-4}$ and the initial step size to be 10^5 for TeCPSGD. We set the ranks to be $r_1 = r_2 =$



(a) 40% missing entries at frame 39.

(b) 40% sampled entries uniformly at random.

Fig. 4: Reconstructed myocardial perfusion images and NRMSE of recovered real component by frame index.

$25, r_3 = 5$ for STC. STC cannot handle tube-sampled data since an entire column of one of the tensor unfoldings will be missing, so we only test it in the case where arbitrarily random entries are missing. The batch t-SVD algorithms are allowed to compute over the data until the difference in NRMSE between iterates is less than 10^{-4} or the algorithm exceeds a specified maximum number of iterations.

We record the NRMSE, mean structural similarity index measures (SSIM) [53] of the reconstructed images, and total algorithm wall-clock times in Table III. Fig. 4 displays a sample of the reconstruction results, along with plots of the NRMSE of each frame's recovered real k-t space as the online algorithms pass over the data.

When deployed on the highly dynamic invivo cardiac perfusion data, our algorithm achieves competitive reconstruction error in less wall clock time. In the tubal-sampling case, which is most practical in real fMRI collection, our method can more rapidly update its subspace estimate during initialization. Beginning at frame 41, strong breathing motion occurs, and the three algorithms are comparable in their subspace tracking abilities. Adjusting the streaming CP algorithms' hyperparameters and STC's choice of multirank also requires exhaustive trial and error, and the results are often sensitive to these choices.

VI. Discussion & Future Work

In this paper we presented a novel algorithm for low-tubal-rank tensor completion with stochastic gradient descent on the product of Grassmann manifolds under the t-SVD algebraic framework. Our method avoids computing any SVDs, and only needs to update and store a smaller orthonormal tensor and the lateral slice of weights per iteration, leading to a powerful and efficient online algorithm that scales linearly in memory use and computation. TOUCAN naturally extends well-known concepts from matrix algebra to the tensor domain for streaming data under the t-SVD model, making it practical in big data settings where batch methods would become intractable.

As long as the input tubal-rank to our algorithm is an upper bound for the tubal-rank of the data generated, our method should find a good approximation to the tensor, in which case some of the tensor factors may have small coefficients, showing that the rank could be smaller. Establishing good techniques for determining tubal-rank in an online way from missing data is a very interesting direction for future work.

TOUCAN is practical in many big data problems where the tensor data is inherently oriented, such as time series data, and contains modes with periodic data best captured by the FFT in the t-SVD framework. Fixing the third mode factor matrix to be the DFT matrix is a strong model assumption, and other works have extended the t-SVD to use other fast orthogonal transforms in the third mode [28]. Future work could consider learning an orthogonal factor matrix in the third mode that best fits the data.

Choosing the best tensor orientation is not always apparent and requires trial and error. While t-SVD algorithms can leverage periodic structure in the data, CP and Tucker models are compatible with any tensor orientation. These methods may also be preferable when the CP or multilinear ranks are much smaller than either tensor mode dimension; TOUCAN's memory requirement will grow multiplicatively between d_1, d_3, and k to store the orthonormal basis, whereas CP and Tucker methods require only storing three small factor matrices and a small core tensor in the case of Tucker tensors. Lastly, t-SVD methods are only useful for imputing missing entries when the data reveals a low-tubal-rank structure, but not for recovering interpretable latent factors that may be useful for data analysis. An interesting line of future work would be to develop a novel tensor decomposition agnostic to tensor orientation and enjoys the low memory footprint and latent factor interpretability of CP decompositions.

Acknowledgments

We would like to thank the anonymous reviewers for their detailed and thorough feedback that greatly helped improve the precision of our paper. We would like to also acknowledge an anonymous reviewer for pointing out the relationship of the t-SVD to the BTD, and Shuchin Aeron for his helpful advice during the course of this project. This work was supported in part by AFOSR YIP award FA9550-19-1-0026, NSF CAREER award CCF-1845076, NSF BIGDATA award IIS-1838179, and the IAS Charles Simonyi endowment.

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, 2009.

Sample %		Subsampled/Zero-filled	TOUCAN	TNN-ADMM	TCTF	OLSTEC	TeCPSGD	STC
50 - Random	SSIM	0.4735	*0.8637*	**0.9518**	0.7129	0.7663	0.7709	0.6474
	Time	–	*1.798*	31.29	11.05	14.94	**1.289**	7.456
50 - Tube	SSIM	0.5678	*0.8507*	**0.9350**	0.7118	0.7403	0.7171	–
	Time	–	**0.8240**	29.22	10.51	13.75	*1.171*	–
40 - Random	SSIM	0.3984	*0.8102*	**0.9266**	0.6384	0.7260	0.7272	0.6412
	Time	–	*2.110*	32.55	11.13	14.17	**1.276**	6.928
40 - Tube	SSIM	0.5094	*0.7800*	**0.8984**	0.6665	0.6905	0.6547	–
	Time	–	**0.7325**	29.06	10.35	13.56	*1.046*	–
20 - Random	SSIM	0.2508	0.5623	**0.8214**	0.4021	0.5609	*0.5719*	0.5524
	Time	–	*3.132*	29.88	9.997	14.29	**1.116**	5.130
20 - Tube	SSIM	0.3744	*0.4913*	**0.7315**	0.4316	0.4507	0.2817	–
	Time	–	**0.6613**	16.49	10.11	13.10	*0.9145*	–

TABLE III: Invivo myocardial perfusion experiment statistics. Bold indicates best, and italicized indicates second-best.

[2] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, *Scalable tensor factorizations for incomplete data*, Chemometrics and Intelligent Laboratory Systems, 106 (2011), pp. 41 – 56. Multiway and Multiset Data Analysis.

[3] H. Avron, *Advanced Algorithmic Techniques in Numerical Linear Algebra: Hybridization and Randomization*, PhD thesis, Tel Aviv University, 2010.

[4] L. Balzano, Y. Chi, and Y. M. Lu, *Streaming PCA and subspace tracking: The missing data case*, Proceedings of the IEEE, 106 (2018), pp. 1293–1310.

[5] L. Balzano, R. D. Nowak, and B. Recht, *Online identification and tracking of subspaces from highly incomplete information*, 2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton), (2010), pp. 704–711.

[6] L. Balzano and S. J. Wright, *Local convergence of an algorithm for subspace identification from partial data*, Foundations of Computational Mathematics, 15 (2015), pp. 1279–1314, https://doi.org/10.1007/s10208-014-9227-7, https://doi.org/10.1007/s10208-014-9227-7.

[7] D. Banco, S. Aeron, and W. S. Hoge, *Sampling and recovery of MRI data using low rank tensor models*, in 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Aug 2016, pp. 448–452, https://doi.org/10.1109/EMBC.2016.7590736.

[8] D. Bertsekas, *Incremental gradient, subgradient, and proximal methods for convex optimization: A survey*, Optimization, 2010 (2015).

[9] N. Boumal and P.-A. Absil, *Low-rank matrix completion via preconditioned optimization on the Grassmann manifold*, Linear Algebra and its Applications, 475 (2015), pp. 200–239.

[10] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Foundations of Computational Mathematics, 9 (2009), p. 717, https://doi.org/10.1007/s10208-009-9045-5.

[11] L. Canyi, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, *Tensor robust principal component analysis with a new tensor nuclear norm*, IEEE Transactions on Pattern Analysis and Machine Intelligence, PP (2018), https://doi.org/10.1109/TPAMI.2019.2891760.

[12] L. De Lathauwer, *Decompositions of a higher-order tensor in block terms—part II: Definitions and uniqueness*, SIAM Journal on Matrix Analysis and Applications, 30 (2008), pp. 1033–1066, https://doi.org/10.1137/070690729.

[13] L. De Lathauwer, B. De Moor, and J. Vandewalle, *A multilinear singular value decomposition*, SIAM journal on Matrix Analysis and Applications, 21 (2000), pp. 1253–1278.

[14] C. De Sa, C. Re, and K. Olukotun, *Global convergence of stochastic gradient descent for some non-convex matrix problems*, in International Conference on Machine Learning, PMLR, 2015, pp. 2332–2341.

[15] M. E. Kilmer and C. D. Martin, *Factorization strategies for third-order tensors*, Linear Algebra and Its Applications, 435 (2011), https://doi.org/10.1016/j.laa.2010.09.020.

[16] A. Edelman, T. A. Arias, and S. T. Smith, *The geometry of algorithms with orthogonality constraints*, SIAM journal on Matrix Analysis and Applications, 20 (1998), pp. 303–353.

[17] 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, 10 (2017), pp. 4589–4604, https://doi.org/10.1109/JSTARS.2017.2714338.

[18] H. Fan, C. Li, Y. Guo, G. Kuang, and J. Ma, *Spatial–spectral total variation regularized low-rank tensor decomposition for hyperspectral image denoising*, IEEE Transactions on Geoscience and Remote Sensing, 56 (2018), pp. 6196–6213, https://doi.org/10.1109/TGRS.2018.2833473.

[19] D. W. Fletcher-Holmes and A. R. Harvey, *Real-time imaging with a hyperspectral fovea*, Journal of Optics A: Pure and Applied Optics, 7 (2005), pp. S298–S302, https://doi.org/10.1088/1464-4258/7/6/007, https://doi.org/10.1088%2F1464-4258%2F7%2F6%2F007.

[20] K. Gilman and L. Balzano, *Online tensor completion and tracking of free submodules with the t-SVD*, in 2020 IEEE 45th International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2020.

[21] J. Girson and S. Aeron, *Tensor completion via optimization on the product of matrix manifolds*, in 2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), Dec 2015, pp. 177–180, https://doi.org/10.1109/CAMSAP.2015.7383765.

[22] E. Gujral, R. Pasricha, and E. E. Papalexakis, *SamBaTen: Sampling-based Batch Incremental Tensor Decomposition*, pp. 387–395, https://doi.org/10.1137/1.9781611975321.44.

[23] C. J. Hillar and L.-H. Lim, *Most tensor problems are np-hard*, J. ACM, 60 (2013), https://doi.org/10.1145/2512329, https://doi.org/10.1145/2512329.

[24] F. L. Hitchcock, *The expression of a tensor or a polyadic as a sum of products*, Journal of Mathematics and Physics, 6 (1927), pp. 164–189, https://doi.org/https://doi.org/10.1002/sapm192761164, https://onlinelibrary.wiley.com/doi/abs/10.1002/sapm192761164, https://arxiv.org/abs/https://onlinelibrary.wiley.com/doi/pdf/10.1002/sapm192761164.

[25] M. Huang, S. Ma, and L. Lai, *Robust low-rank matrix completion via an alternating manifold proximal gradient continuation method*, IEEE Transactions on Signal Processing, 69 (2021), pp. 2639–2652.

[26] P. Jain and S. Oh, *Provable tensor factorization with missing data*, in Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1, NIPS'14, Cambridge, MA, USA, 2014, MIT Press, pp. 1431–1439.

[27] H. Kasai, *Online low-rank tensor subspace tracking from incomplete data by CP decomposition using recursive least squares*, in 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), March 2016, pp. 2519–2523, https://doi.org/10.1109/ICASSP.2016.7472131.

[28] E. Kernfeld, M. Kilmer, and S. Aeron, *Tensor–tensor products with invertible linear transforms*, Linear Algebra and its Applications, 485 (2015), pp. 545 – 570, https://doi.org/https://doi.org/10.1016/j.laa.2015.07.021.

[29] M. Kilmer, K. Braman, N. Hao, and R. Hoover, *Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging*, SIAM Journal on Matrix Analysis and Applications, 34 (2013), pp. 148–172, https://doi.org/10.1137/110837711.

[30] T. Kolda and B. Bader, *Tensor decompositions and applications*, SIAM Review, 51 (2009), pp. 455–500, https://doi.org/10.1137/07070111X.

[31] T. G. Kolda and D. Hong, *Stochastic gradients for large-scale tensor decomposition.* arXiv, June 2019, https://arxiv.org/abs/1906.01687. submitted for publication.

[32] P. Kroonenberg, *Applied Multiway Data Analysis*, vol. 702, 01 2008, https://doi.org/10.1002/9780470238004.

[33] S. G. Lingala, Y. Hu, E. DiBella, and M. Jacob, *Accelerated dynamic MRI exploiting sparsity and low-rank structure: k-t SLR*, IEEE

Transactions on Medical Imaging, 30 (2011), pp. 1042–1054, https://doi.org/10.1109/TMI.2010.2100850.

[34] 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, 35 (2013), pp. 208–220, https://doi.org/10.1109/TPAMI.2012.39.

[35] X.-Y. Liu, S. Aeron, V. Aggarwal, X. Wang, and M.-Y. Wu, *Adaptive sampling of RF fingerprints for fine-grained indoor localization*, IEEE Transactions on Mobile Computing, 15 (2015), pp. 2411–2423.

[36] C. Ma, X. Yang, and H. Wang, *Randomized online CP decomposition*, in 2018 Tenth International Conference on Advanced Computational Intelligence (ICACI), March 2018, pp. 414–419, https://doi.org/10.1109/ICACI.2018.8377495.

[37] T. Maehara, K. Hayashi, and K.-i. Kawarabayashi, *Expected tensor decomposition with stochastic gradient descent*, in Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI'16, AAAI Press, 2016, p. 1919–1925.

[38] M. Mardani, G. Mateos, and G. B. Giannakis, *Subspace learning and imputation for streaming big data matrices and tensors*, IEEE Transactions on Signal Processing, 63 (2015), pp. 2663–2677, https://doi.org/10.1109/TSP.2015.2417491.

[39] C. Martin, R. Shafer, and B. LaRue, *An order-p tensor factorization with applications in imaging*, SIAM Journal on Scientific Computing, 35 (2013), pp. A474–A490, https://doi.org/10.1137/110841229.

[40] M. Nimishakavi, B. Mishra, M. Gupta, and P. Talukdar, *Inductive framework for multi-aspect streaming tensor completion with side information*, New York, NY, USA, 2018, Association for Computing Machinery, https://doi.org/10.1145/3269206.3271713.

[41] D. Nion and N. D. Sidiropoulos, *Adaptive algorithms to track the PARAFAC decomposition of a third-order tensor*, IEEE Transactions on Signal Processing, 57 (2009), pp. 2299–2310, https://doi.org/10.1109/TSP.2009.2016885.

[42] K. B. Petersen and M. S. Pedersen, *The matrix cookbook*, 2012, https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf.

[43] J. Pothier, J. Girson, and S. Aeron, *An algorithm for online tensor prediction*, 2015, https://arxiv.org/abs/1507.07974.

[44] Y. Qian, F. Xiong, S. Zeng, J. Zhou, and Y. Y. Tang, *Matrix-vector nonnegative tensor factorization for blind unmixing of hyperspectral imagery*, IEEE Transactions on Geoscience and Remote Sensing, 55 (2017), pp. 1776–1792.

[45] J. R. Shewchuk et al., *An introduction to the conjugate gradient method without the agonizing pain*, 1994.

[46] Q. Song, H. Ge, J. Caverlee, and X. Hu, *Tensor completion algorithms in big data analytics*, ACM Transactions on Knowledge Discovery from Data, 13 (2017), https://doi.org/10.1145/3278607.

[47] Y. Sun, Y. Guo, C. Luo, J. Tropp, and M. Udell, *Low-rank Tucker approximation of a tensor from streaming data*, SIAM Journal on Mathematics of Data Science, 2 (2020), pp. 1123–1150, https://doi.org/10.1137/19M1257718.

[48] D. A. Tarzanagh and G. Michailidis, *Fast randomized algorithms for t-product based tensor operations and decompositions with applications to imaging data*, SIAM Journal on Imaging Sciences, 11 (2018), pp. 2629–2664, https://doi.org/10.1137/17M1159932.

[49] L. R. Tucker, *Some mathematical notes on three-mode factor analysis*, Psychometrika, 31 (1966), pp. 279–311.

[50] M. Vandecappelle, N. Vervliet, and L. De Lathauwer, *Nonlinear least squares updating of the canonical polyadic decomposition*, in 2017 25th European Signal Processing Conference (EUSIPCO), Aug 2017, pp. 663–667, https://doi.org/10.23919/EUSIPCO.2017.8081290.

[51] A. Vergara, J. Fonollosa, J. Mahiques, M. Trincavelli, N. Rulkov, and R. Huerta, *On the performance of gas sensor arrays in open sampling systems using inhibitory support vector machines*, Sensors and Actuators B: Chemical, 185 (2013), pp. 462 – 477, https://doi.org/https://doi.org/10.1016/j.snb.2013.05.027.

[52] T. Wang, Z. Zhu, and E. Blasch, *Bio-inspired adaptive hyperspectral imaging for real-time target tracking*, IEEE Sensors Journal, 10 (2010), pp. 647–654, https://doi.org/10.1109/JSEN.2009.2038657.

[53] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, *Image quality assessment: From error visibility to structural similarity*, Trans. Img. Proc., 13 (2004), pp. 600–612, https://doi.org/10.1109/TIP.2003.819861.

[54] K. Xie, L. Wang, X. Wang, G. Xie, J. Wen, G. Zhang, J. Cao, and D. Zhang, *Accurate recovery of internet traffic data: A sequential tensor completion approach*, IEEE/ACM Transactions on Networking, 26 (2018), pp. 793–806, https://doi.org/10.1109/TNET.2018.2797094.

[55] D. Zhang and L. Balzano, *Global convergence of a Grassmannian gradient descent algorithm for subspace estimation*, in AISTATS, 2015.

[56] F. Zhang, J. Hou, J. Wang, and W. Wang, *Uniqueness guarantee of solutions of tensor tubal-rank minimization problem*, IEEE Signal Processing Letters, 27 (2020), pp. 540–544, https://doi.org/10.1109/LSP.2020.2983305.

[57] G. Zhang, X. Fu, J. Wang, X. Zhao, and M. Hong, *Spectrum cartography via coupled block-term tensor decomposition*, IEEE Transactions on Signal Processing, 68 (2020), pp. 3660–3675.

[58] Z. Zhang and S. Aeron, *Exact tensor completion using t-SVD*, IEEE Transactions on Signal Processing, 65 (2017), pp. 1511–1526, https://doi.org/10.1109/TSP.2016.2639466.

[59] Z. Zhang, G. Ely, S. Aeron, N. Hao, and M. Kilmer, *Novel methods for multilinear data completion and de-noising based on tensor-SVD*, in 2014 IEEE Conference on Computer Vision and Pattern Recognition, June 2014, pp. 3842–3849, https://doi.org/10.1109/CVPR.2014.485.

[60] Z. Zhang, D. Liu, S. Aeron, and A. Vetro, *An online tensor robust PCA algorithm for sequential 2D data*, in 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), March 2016, pp. 2434–2438, https://doi.org/10.1109/ICASSP.2016.7472114.

[61] Y.-B. Zheng, T.-Z. Huang, X.-L. Zhao, T.-X. Jiang, T.-H. Ma, and T.-Y. Ji, *Mixed noise removal in hyperspectral image via low-fibered-rank regularization*, IEEE Transactions on Geoscience and Remote Sensing, 58 (2020), pp. 734–749, https://doi.org/10.1109/TGRS.2019.2940534.

[62] P. Zhou, C. Lu, Z. Lin, and C. Zhang, *Tensor factorization for low-rank tensor completion*, IEEE Transactions on Image Processing, 27 (2018), pp. 1152–1163, https://doi.org/10.1109/TIP.2017.2762595.

Kyle Gilman Kyle Gilman is a Ph.D. candidate in Electrical and Computer Engineering working with Professor Laura Balzano at the University of Michigan, Ann Arbor, MI. Kyle received his B.S. in Electrical Engineering from the University of Wyoming 2017. His main research focus is on low-rank modeling and optimization for matrix and tensor factorizations, online learning, missing data completion, and applications to signal processing and data science problems.

Davoud Ataee Tarzanagh is currently a Postdoctoral Research Fellow in the EECS Department at the University of Michigan. He completed his Ph.D. studies in Mathematics at the University of Florida in 2020. His current research interests include mathematical optimization, analysis of high dimensional data with network structure, and tensor data analysis.

Laura Balzano Laura Balzano is an associate professor of Electrical Engineering and Computer Science at the University of Michigan. She has a PhD from the University of Wisconsin in ECE. She is currently serving as associate editor of the IEEE Open Journal of Signal Processing and the SIAM Journal of the Mathematics of Data Science. She is recipient of the NSF Career Award, ARO Young Investigator Award, AFOSR Young Investigator Award, and faculty fellowships from Intel and 3M. She received the Vulcans Education Excellence Award at the University of Michigan. Her main research focus is on modeling and optimization with big, messy data — highly incomplete or corrupted data, uncalibrated data, and heterogeneous data — and its applications in a wide range of scientific problems.

Appendix A
Proof of Theorem IV.4

Proof. The proof follows from Lemma G.1, and the fact $\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}}$ is a $d_1 d_3 \times d_3 r$ matrix with orthonormal columns.

From Lemma G.1, we have that

$$J \leq \frac{1}{2}\sqrt{\kappa}\log(2/\epsilon) \leq \frac{1}{2}\left(\frac{1+\delta^{-1}\tau}{1-\delta^{-1}\tau}\right)^{\frac{1}{2}}\log(2/\epsilon), \tag{37}$$

where

$$\kappa := \kappa(\boldsymbol{\mathcal{F}}_{\Omega_t}\overline{\boldsymbol{U}})^2, \quad \text{and } \tau := C\sqrt{d_1 d_3 \mu(\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}})\log(|\Omega|)/(|\Omega|}.$$

Now, from our assumption bounding the coherence of the iterates $\boldsymbol{\mathcal{U}}_t$, the result from Lemma G.2 gives $\mu(\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}}) \leq \mu_0 r/d_1$. To ensure the bound is not vacuous, we must ensure

$$1 \geq \delta > \tau \geq C\sqrt{d_1 d_3 \mu(\boldsymbol{U})\log(|\Omega|)/|\Omega|}. \tag{38}$$

In other words, we must sample sufficiently many rows, i.e.,

$$|\Omega_t|/\log(|\Omega_t|) > C^2 \mu_0 r d_3. \tag{39}$$

for some constant C independent of μ_0, r, d_1, and d_3.

We can simply verify Equation (39) for low rank tensors as follows. Without loss of generality, assume $d_1 \geq d_3$. Then, it follows that $\log(|\Omega_t|) \leq 2\log(d_1) \ll d_1$. Further, if the tensor is low rank and incoherent, both μ_0 and the tubal rank are small, which implies that the right side of Equation (39) is sufficiently small. Finally (39) ensures that (38) holds, and we can ensure the number of CGD iterations is low. □

Appendix B
Missing Tensor Tubes

Again, let $\boldsymbol{\mathcal{X}} = [\overrightarrow{\boldsymbol{\mathcal{X}}}_1 \ldots, \overrightarrow{\boldsymbol{\mathcal{X}}}_T] \in \mathbb{R}^{d_1 \times T \times d_3}$ be a set of lateral slices for each time instance. At every time t, we observe an incomplete lateral slice $\boldsymbol{\mathcal{X}}_t \in \mathbb{M}^{d_1}_{d_3}$ on the indices $\Omega_t \subset \{1, \ldots, d_1\}$ where not all tubes of the slice are observed. Denote $\boldsymbol{\mathcal{P}}_{\Omega_t} \in \mathbb{R}^{|\Omega_t| \times d_1 \times d_3}$ as the tensor that selects the coordinate axes of $\mathbb{R}^{d_1}$ indexed by Ω_t. $\boldsymbol{\mathcal{P}}_{\Omega_t}$ is a tensor whose first frontal slice is a subsampled identity matrix on the rows indexed by Ω_t; all other frontal slices are zeros. We then observe the lateral slice $\boldsymbol{\mathcal{P}}_{\Omega_t} * \overrightarrow{\boldsymbol{\mathcal{X}}}_t$ at time t. Let $\boldsymbol{\mathcal{U}}_{\Omega_t}$ denote the subtensor of $\boldsymbol{\mathcal{U}}$ consisting of the tubes indexed by Ω_t, and $\boldsymbol{\mathcal{X}}_{\Omega_t} = \boldsymbol{\mathcal{P}}_{\Omega_t} * \overrightarrow{\boldsymbol{\mathcal{X}}}_t$ denote a lateral slice in $\mathbb{R}^{|\Omega_t| \times 1 \times d_3}$ observed on the tubes indexed by Ω_t. It can be shown that the objective function can be rewritten as

$$\min_{[\boldsymbol{\mathcal{U}}] \in \mathcal{G}(r,d_1,d_3)} \frac{1}{T}\sum_{t=1}^{T} \min_{\overrightarrow{\boldsymbol{\mathcal{W}}}_t \in \mathbb{R}^{r \times 1 \times d_3}} \frac{1}{2}\left\| \overrightarrow{\boldsymbol{\mathcal{X}}}_{\Omega_t} - \boldsymbol{\mathcal{U}}_{\Omega_t} * \overrightarrow{\boldsymbol{\mathcal{W}}}_t \right\|_F^2. \tag{40}$$

In the Fourier domain, $\overline{\mathcal{L}}_t$ becomes becomes

$$\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}}) = \min_{\overline{\boldsymbol{W}}_t} \frac{1}{2}\|\overline{\boldsymbol{X}}_{\Omega_t} - \overline{\boldsymbol{U}}_{\Omega_t}\overline{\boldsymbol{W}}_t\|_F^2. \tag{41}$$

The notation $\overline{\boldsymbol{U}}_{\Omega_t} \in \mathbb{C}^{|\Omega_t| d_3 \times d_3 r}$ denotes the block-diagonal matrix of $\overline{\boldsymbol{U}}$ consisting of the rows indexed by Ω_t. Similarly, $\overline{\boldsymbol{X}}_{\Omega_t}$ is a block-diagonal matrix in $\mathbb{R}^{|\Omega_t| d_3 \times d_3}$ observed on the rows indexed by Ω_t. The problem is block-diagonal, and as the work in [21] showed, it is separable in each frontal slice in the Fourier domain. The algorithm is similar to that in Alg. 1, except the optimal weights $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$ can be solved exactly in closed form using pseudo-inverses in the Fourier domain, and $\bar{\boldsymbol{\rho}}_k$ is replaced by $\bar{\boldsymbol{r}}_k$ in Eq. 27. Likewise, our step size is $\eta_k = \arctan(\|\bar{\boldsymbol{r}}_k\|/\|\overline{\boldsymbol{W}}_{t,k}\|)$. We give the full algorithm in Algorithm 2.

Algorithm 2 Tensor rank-One Update on the Complex grassmanniAN (TOUCAN): Missing Tensor Tubes

```
Require: Data: $\overrightarrow{\boldsymbol{\mathcal{X}}}_t \in \mathbb{R}^{d_1 \times 1 \times d_3}$  $\forall t = 1, \ldots, T$ observed on $\Omega_t$; tubal-rank $r$.
Initialize Fourier transformed orthonormal tensor $\overline{\boldsymbol{\mathcal{U}}}_0 \in \mathbb{C}^{d_1 \times r \times d_3}$.
for $t = 1$ to $T$ do
    Compute $\overline{\boldsymbol{\mathcal{X}}}_{\Omega_t} = \texttt{fft}(\Delta_{\Omega_t}(\boldsymbol{\mathcal{X}}_t), [], 3)$.
    Estimate optimal weights: $\overline{\boldsymbol{W}}_{t,k}(\overline{\boldsymbol{U}}_t) = \overline{\boldsymbol{U}}^{\dagger}_{\Omega_t,k}\overline{\boldsymbol{X}}_{\Omega_t,k}$.
    Predict full vector: $\overline{\boldsymbol{P}}_t = \overline{\boldsymbol{U}}_t\overline{\boldsymbol{W}}_t(\overline{\boldsymbol{U}}_t)$.
    Shape into tensor and transform: $\overrightarrow{\boldsymbol{\mathcal{P}}}_t = \texttt{ifft}(\overline{\boldsymbol{\mathcal{P}}}_t, [], 3)$.
    Compute residual: $\overrightarrow{\boldsymbol{\mathcal{R}}}_t = \Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t) - \overrightarrow{\boldsymbol{\mathcal{P}}}_t$.
    Update subspace: $\overline{\boldsymbol{\mathcal{U}}}_{t+1}$ from (27).
    Transform: $\boldsymbol{\mathcal{U}}_{t+1} = \texttt{ifft}(\overline{\boldsymbol{\mathcal{U}}}_{t+1}, [], 3)$.
    Transform: $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}_t) = \texttt{ifft}(\overline{\boldsymbol{\mathcal{W}}}_t(\overline{\boldsymbol{\mathcal{U}}}_t), [], 3)$.
end for
return  $\boldsymbol{\mathcal{U}}, \overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}_t)$,  $\forall t = 1, \ldots, T$
```

Appendix C
Gradient derivation

Our algorithm substitutes $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}}) = (\overline{\boldsymbol{U}}'\boldsymbol{\mathcal{F}}'_{\Omega_t}\boldsymbol{\mathcal{F}}_{\Omega_t}\overline{\boldsymbol{U}})^{-1}\overline{\boldsymbol{U}}'\boldsymbol{\mathcal{F}}'_{\underline{\Omega_t}}$ into $\overline{\mathcal{L}}(\overline{\boldsymbol{U}})$ and computes the gradient with respect to $\overline{\boldsymbol{U}}$ directly. First rewrite $\overline{\mathcal{L}}_t$ as

$$\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}}) = \frac{1}{2}\|\boldsymbol{\mathcal{F}}_{\Omega_t}\bar{\boldsymbol{x}}_t\|_2^2 - \frac{1}{2}\operatorname{tr}((\overline{\boldsymbol{U}}'\boldsymbol{C}\overline{\boldsymbol{U}})^{-1}\overline{\boldsymbol{U}}'\boldsymbol{B}\overline{\boldsymbol{U}}),$$

where $\boldsymbol{C} := \boldsymbol{\mathcal{F}}'_{\Omega_t}\boldsymbol{\mathcal{F}}_{\Omega_t}$ and $\boldsymbol{B} := \boldsymbol{C}\bar{\boldsymbol{x}}_t\bar{\boldsymbol{x}}'_t\boldsymbol{C}$ for ease of notation.

Now, we can take the gradient with respect to $\overline{\boldsymbol{U}}$ for this form of trace function via [42, Equation (126)] and obtain

$$\begin{aligned}\frac{\partial\overline{\mathcal{L}}_t}{\partial\overline{\boldsymbol{U}}} &= -\boldsymbol{B}\overline{\boldsymbol{U}}(\overline{\boldsymbol{U}}'\boldsymbol{C}\overline{\boldsymbol{U}})^{-1} \\ &\quad + \boldsymbol{C}\overline{\boldsymbol{U}}(\overline{\boldsymbol{U}}'\boldsymbol{C}\overline{\boldsymbol{U}})^{-1}\overline{\boldsymbol{U}}'\boldsymbol{B}\overline{\boldsymbol{U}}(\overline{\boldsymbol{U}}'\boldsymbol{C}\overline{\boldsymbol{U}})^{-1}.\end{aligned} \tag{42}$$

It is then straight-forward to see Eq. (20) is equivalent to the above gradient by substituting the expression for $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ and simplifying.

Appendix D
t-Product and t-SVD

Using properties of the Fourier Transform, we give Lemma Lemma D.1, which describes conjugate symmetry of a real-valued signal transformed into the Fourier domain:

Lemma D.1. *[11] Given* $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}, \overline{\boldsymbol{A}}_1 \in \mathbb{R}^{d_1 \times d_2}$ *and* $\mathit{conj}(\overline{\boldsymbol{A}}_k) = \overline{\boldsymbol{A}}_{d_3-k+2}, \quad k = 2, \ldots, \lceil \frac{d_3+1}{2} \rceil$.

Algorithm 4 t-SVD [11]

```
Inputs: $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$
Output: t-SVD components $\boldsymbol{\mathcal{U}}, \boldsymbol{\mathcal{S}}$, and $\boldsymbol{\mathcal{V}}$ of $\boldsymbol{\mathcal{A}}$.
Compute $\overline{\boldsymbol{\mathcal{A}}} = \texttt{fft}(\boldsymbol{\mathcal{A}}, [], 3)$
Compute each frontal slice of $\overline{\boldsymbol{\mathcal{U}}}, \overline{\boldsymbol{\mathcal{S}}}, \overline{\boldsymbol{\mathcal{V}}}$ by
for $k = 1, \dots, \lceil \frac{d_3+1}{2} \rceil$ do
    $[\overline{\boldsymbol{U}}_k, \overline{\boldsymbol{S}}_k, \overline{\boldsymbol{V}}_k] = \text{SVD}(\overline{\boldsymbol{A}}_k)$;
end for
for $k = \lceil \frac{d_3+1}{2} \rceil + 1, \dots, d_3$ do
    $\overline{\boldsymbol{U}}_k = \texttt{conj}(\overline{\boldsymbol{U}}_{d_3-k+2}))$
    $\overline{\boldsymbol{S}}_k = \texttt{conj}(\overline{\boldsymbol{S}}_{d_3-k+2}))$
    $\overline{\boldsymbol{V}}_k = \texttt{conj}(\overline{\boldsymbol{V}}_{d_3-k+2}))$
end for
Compute $\boldsymbol{\mathcal{U}} = \texttt{ifft}(\overline{\boldsymbol{\mathcal{U}}}, [], 3)$, $\boldsymbol{\mathcal{S}} = \texttt{ifft}(\overline{\boldsymbol{\mathcal{S}}}, [], 3)$, $\boldsymbol{\mathcal{V}} = \texttt{ifft}(\overline{\boldsymbol{\mathcal{V}}}, [], 3)$
```

Lemma D.1 states the conjugate symmetry property for a real-valued signal in the frequency domain using properties from the Fourier transform; this will be useful later for avoiding redundant computations.

Algorithm 3 Tensor-Tensor Product [11]

```
Inputs: $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{d_2 \times l \times d_3}$
Output: $\boldsymbol{\mathcal{C}} = \boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{B}} \in \mathbb{R}^{d_1 \times l \times d_3}$
Compute $\overline{\boldsymbol{\mathcal{A}}} = \texttt{fft}(\boldsymbol{\mathcal{A}}, [], 3)$ and $\overline{\boldsymbol{\mathcal{B}}} = \texttt{fft}(\boldsymbol{\mathcal{B}}, [], 3)$
Compute each frontal slice of $\overline{\boldsymbol{\mathcal{C}}}$ by
    $$\overline{\boldsymbol{C}}_k = \begin{cases} \overline{\boldsymbol{A}}_k \overline{\boldsymbol{B}}_k, & k = 1, \dots, \lceil \frac{d_3+1}{2} \rceil \\ \texttt{conj}(\overline{\boldsymbol{C}}^{(d_3-k+2)}), & k = \lceil \frac{d_3+1}{2} \rceil + 1, \dots, d_3 \end{cases}$$
Compute $\boldsymbol{\mathcal{C}} = \texttt{ifft}(\overline{\boldsymbol{\mathcal{C}}}, [], 3)$
```

Definition D.2. *Conjugate transpose [15] The conjugate transpose of a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{d_1 \times d_2 \times d_3}$ is the tensor $\boldsymbol{\mathcal{A}}' \in \mathbb{C}^{d_2 \times d_1 \times d_3}$ obtained by conjugate transposing each frontal slice of $\boldsymbol{\mathcal{A}}$ and then reversing the order of transposed slices 2 through d_3:*

$$\boldsymbol{\mathcal{A}}' = \texttt{fold}\Big(\begin{bmatrix} \boldsymbol{A}_1{}' & \boldsymbol{A}_{d_3}{}' & \cdots & \boldsymbol{A}_2{}' \end{bmatrix}' \Big).$$

Definition D.3. *Identity tensor[15] The identity tensor $\boldsymbol{\mathcal{I}}_{nnd_3} \in \mathbb{R}^{n \times n \times d_3}$ is the tensor whose first frontal slice being the $n \times n$ identity matrix, and all other frontal slices being all zeros. Property: $\boldsymbol{\mathcal{A}} * \boldsymbol{\mathcal{I}} = \boldsymbol{\mathcal{I}} * \boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{A}}$.*

Definition D.4. *Orthogonal tensor [15] A tensor $\boldsymbol{\mathcal{Q}} \in \mathbb{R}^{n \times n \times d_3}$ is orthogonal if it satisfies $\boldsymbol{\mathcal{Q}}' * \boldsymbol{\mathcal{Q}} = \boldsymbol{\mathcal{Q}} * \boldsymbol{\mathcal{Q}}' = \boldsymbol{\mathcal{I}}$.*

Definition D.5. *F-diagonal tensor [15] A tensor is called F-diagonal if each of its frontal slices is a diagonal matrix.*

Appendix E
Proof of Proposition IV.1

Proof. (P1) Assuming we sample enough entries of the data such that $(\boldsymbol{\mathcal{F}}_{\Omega_t} \overline{\boldsymbol{U}})'(\boldsymbol{\mathcal{F}}_{\Omega_t} \overline{\boldsymbol{U}})$ remains full rank, then $\bar{\boldsymbol{w}}_t(\overline{\boldsymbol{U}})$ is the unique minimizer of the inner least-squares problem.

(P2) For a fixed $\boldsymbol{\mathcal{U}}$, let $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$ be the unique minimizer in (18b) as shown in (P1), and say we choose a different basis for $[\boldsymbol{\mathcal{U}}]$, i.e. $\boldsymbol{\mathcal{U}}^R := \boldsymbol{\mathcal{U}} * \boldsymbol{\mathcal{R}}$ for any t-orthogonal tensor $\boldsymbol{\mathcal{R}} \in \mathcal{O}(r, r, d_3)$. Now we see that $\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}^R) = \boldsymbol{\mathcal{R}}' * \overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$. Since $\mathcal{L}_t(\boldsymbol{\mathcal{U}})$ defined in (18a) merely depends on the product $\boldsymbol{\mathcal{U}} * \overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}})$, we have $\mathcal{L}_t(\boldsymbol{\mathcal{U}}^R) = \mathcal{L}_t(\boldsymbol{\mathcal{U}})$ which implies that the outer objectives of $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{U}}^R$ are identical, i.e., $\mathcal{L}(\boldsymbol{\mathcal{U}}^R) = \mathcal{L}(\boldsymbol{\mathcal{U}})$. Hence, the objective is constant over sets of full tubal-rank tensors $\boldsymbol{\mathcal{U}}$ spanning the same free submodule. Now, considering these sets as an equivalence class $[\boldsymbol{\mathcal{U}}]$, the problem is well-defined and smooth on the t-Grassmannian. This type of argument was also provided in [9, Section 3] for the offline matrix completion problem on the Grassmannian.

By Proposition II.9, $\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}})$ in (19) is a smooth function over the Cartesian product of complex (matrix) Grassmann manifolds in the Fourier domain. This, together with the fact that the Fourier transform operator $\boldsymbol{F}_{d_3}$ is invertible, also implies that $\mathcal{F} : \mathcal{G}(r, d_1, d_3) \to \mathbb{R}$ is a well-defined smooth function over the product manifold. Further, the solutions of the original problem in (17) can be obtained as follows:

$$\boldsymbol{\mathcal{U}} = \overline{\boldsymbol{\mathcal{U}}} \times_3 \boldsymbol{F}'_{d_3} = \texttt{fold}(\overline{\boldsymbol{U}}_1; \overline{\boldsymbol{U}}_2; \dots; \overline{\boldsymbol{U}}_{d_3}) \times_3 \boldsymbol{F}'_{d_3},$$
$$\overrightarrow{\boldsymbol{\mathcal{W}}}_t(\boldsymbol{\mathcal{U}}) = \texttt{fold}(\bar{\boldsymbol{w}}_{t,1}; \bar{\boldsymbol{w}}_{t,2}; \dots; \bar{\boldsymbol{w}}_{t,d_3}) \times_3 \boldsymbol{F}'_{d_3}.$$

This completes the proof. □

Appendix F
t-SVD interpretation of TOUCAN

We note here that $\boldsymbol{\mathcal{U}}$ is one choice of representation for a point on the product Grassmannian where the Fourier transform along its tubes $\overline{\boldsymbol{\mathcal{U}}}$ has as each frontal face a matrix with orthonormal columns. However, we can equivalently represent this point using an $d_1 d_3 \times d_3 r$ block-diagonal matrix in the Fourier domain, with the frontal faces of $\overline{\boldsymbol{\mathcal{U}}}$ on the diagonal. We will revisit this representation below.

We can rewrite the objective function using the block-circulant matrix definition of the t-product:

$$\mathcal{L}_t(\boldsymbol{\mathcal{U}}) = \min_{\overrightarrow{\boldsymbol{\mathcal{W}}}_t} \frac{1}{2} \| \boldsymbol{P}_{\Omega_t} \texttt{unfold}(\Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t)) \\ - \boldsymbol{P}_{\Omega_t}(\texttt{bcirc}(\boldsymbol{\mathcal{U}}) \cdot \texttt{unfold}(\overrightarrow{\boldsymbol{\mathcal{W}}}_t)) \|_F^2.$$

Here $\boldsymbol{P}_{\Omega_t}$ is a subsampled identity matrix of size $|\Omega_t| \times d_1 d_3$, $\texttt{unfold}(\Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t)) \in \mathbb{R}^{d_1 d_3}$, $\texttt{bcirc}(\boldsymbol{\mathcal{U}}) \in \mathbb{R}^{d_1 d_3 \times d_3 r}$, and $\texttt{unfold}(\overrightarrow{\boldsymbol{\mathcal{W}}}_t) \in \mathbb{R}^{d_3 r}$. Using block-circulant diagonalization and the fact $d_2 = 1$ when processing a single slice, we can rewrite the product $\boldsymbol{P}_{\Omega_t} \cdot (\texttt{bcirc}(\boldsymbol{\mathcal{U}}) \cdot \texttt{unfold}(\overrightarrow{\boldsymbol{\mathcal{W}}}_t))$ as

$$\boldsymbol{P}_{\Omega_t}(\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1})(\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1}) \texttt{bcirc}(\boldsymbol{\mathcal{U}}) \boldsymbol{F}_{d_3}^{-1} \boldsymbol{F}_{d_3} \texttt{unfold}(\overrightarrow{\boldsymbol{\mathcal{W}}}_t)$$
$$= \boldsymbol{P}_{\Omega_t}(\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1}) \overline{\boldsymbol{U}} \bar{\boldsymbol{w}}_t,$$

where $\bar{\boldsymbol{w}}_t := \texttt{unfold}(\overrightarrow{\boldsymbol{\mathcal{W}}}_t)$ and $\overline{\boldsymbol{U}} = (\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1}) \cdot \texttt{bcirc}(\boldsymbol{\mathcal{U}}) \cdot \boldsymbol{F}_{d_3}^{-1}$. $\overline{\boldsymbol{U}}$ is of size $d_1 d_3 \times d_3 r$ and gives us another representation of $\overline{\boldsymbol{\mathcal{U}}}$, with the frontal slices of $\overline{\boldsymbol{\mathcal{U}}}$ on the diagonal, with d_3 blocks of size $d_1 \times r$. We therefore have the following equivalent form for $\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}})$:

$$\overline{\mathcal{L}}_t(\overline{\boldsymbol{U}}) = \min_{\bar{\boldsymbol{w}}_t} \frac{1}{2} \| \boldsymbol{\mathcal{F}}_{\Omega_t}(\bar{\boldsymbol{x}}_t - \overline{\boldsymbol{U}} \bar{\boldsymbol{w}}_t) \|_2^2,$$

where $\bar{\boldsymbol{x}}_t \in \mathbb{R}^{d_1 d_3} := \texttt{vec}(\Delta_{\Omega_t}(\overrightarrow{\boldsymbol{\mathcal{X}}}_t) \times_3 \boldsymbol{F}_{d_3}) \in \mathbb{C}^{d_3 r}$ for convenient notation. Finally, $\boldsymbol{\mathcal{F}}_{\Omega_t} = \boldsymbol{P}_{\Omega_t}(\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1}) \in \mathbb{C}^{|\Omega_t| \times d_1 d_3}$ is the subsampled inverse Fourier transform.

Appendix G
Supporting lemmas of Theorem IV.4

The following lemma utilizes the notion of coherence of an $m \times r$ subspace basis $\boldsymbol{U}$, defined as $\mu(\boldsymbol{U}) = \max_{1 \leq i \leq m} \|\mathbf{P}_{\boldsymbol{U}} \boldsymbol{e}_i\|_2^2$, where $\mathbf{P}_{\boldsymbol{U}}$ is the orthogonal projection onto $\boldsymbol{U}$ and $\boldsymbol{e}_i$ is the i^{th} standard basis vector [10].

Lemma G.1. *[3, Lemma 8.3.3] Let $\boldsymbol{U}$ be an $m \times r$ orthonormal matrix and $\boldsymbol{S}$ be a random subsampling operator that samples $|\Omega|$ rows from $\boldsymbol{U}$ uniformly such that $|\Omega|/\log(|\Omega|) \geq C^2 m\mu(\boldsymbol{U})$. Let C be a universal constant, and $\delta \in [0,1]$. Then, with probability at least $1-\delta$ $\mathbb{E}\{\|\mathbf{I}_r - \frac{m}{|\Omega|}\boldsymbol{U}'\boldsymbol{S}'\boldsymbol{S}\boldsymbol{U}\|\} \leq C\sqrt{m\mu(\boldsymbol{U})\log(|\Omega|)/|\Omega|} := \tau,$ and*

$$\kappa(\boldsymbol{S}\boldsymbol{U}) \leq \sqrt{\frac{1+\delta^{-1}\tau}{1-\delta^{-1}\tau}}. \tag{43}$$

Lemma G.2. *Let $\boldsymbol{F}_{d_3} = [\boldsymbol{f}_1 \dots \boldsymbol{f}_{d_3}]$ denote the normalized $d_3 \times d_3$ DFT matrix. Let $\overline{\boldsymbol{U}}$ be the block-diagonal form of $\boldsymbol{\mathcal{U}}$ in the Fourier domain. For a tensor $\boldsymbol{\mathcal{U}}$, define*

$$\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}} := (\boldsymbol{F}_{d_3}^{-1} \otimes \mathbf{I}_{d_1})\overline{\boldsymbol{U}}. \tag{44}$$

Then, we have $\mu(\boldsymbol{\mathcal{U}}) = \mu(\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}})$, where the function μ is given in Definition IV.3.

Proof. It follows from Definition IV.3 and (4) that

$$\begin{aligned}
\mu(\boldsymbol{\mathcal{U}}) &= \max_{i=1,\dots,d_1} \left\| \begin{bmatrix} \overline{\boldsymbol{U}}_1' & & 0 \\ & \ddots & \\ 0 & & \overline{\boldsymbol{U}}_{d_3}' \end{bmatrix} \begin{bmatrix} \boldsymbol{e}_i \\ \vdots \\ \boldsymbol{e}_i \end{bmatrix} \right\|_2^2 \\
&= \max_{i=1,\dots,d_1} \sum_{j=1}^{d_3} \|\overline{\boldsymbol{U}}_j' \boldsymbol{e}_i\|_2^2, \qquad (45)
\end{aligned}$$

where $\boldsymbol{e}_i$ is the i^{th} standard basis vector in $\mathbb{R}^{d_1}$. Further, from the definition of $\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}}$ in (44), we have

$$\mu(\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}}) = \max_{i=1,\dots,d_1 d_3} \|\overline{\boldsymbol{U}}'(\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1})\boldsymbol{e}_i\|_2^2. \tag{46}$$

Denote the (i,j)-th entry of the normalized DFT matrix by f_{ij}. Through simple algebra, we can see that $(\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1})\boldsymbol{e}_i = \boldsymbol{f}_m \otimes \boldsymbol{e}_n$ for $i \in \{d_1 d_3\}$, $m \in \{d_3\}$, and $n \in \{d_1\}$. This together with (46) implies that

$$\begin{aligned}
\mu(\boldsymbol{\mathcal{F}}\overline{\boldsymbol{U}}) &= \max_{i=1,\dots,d_1 d_3} \|\overline{\boldsymbol{U}}'(\boldsymbol{F}_{d_3} \otimes \mathbf{I}_{d_1})\boldsymbol{e}_i\|_2^2 \\
&= \max_{\substack{n=1,\dots,d_1 \\ m=1,\dots,d_3}} \left\| \begin{bmatrix} \overline{\boldsymbol{U}}_1' & & 0 \\ & \ddots & \\ 0 & & \overline{\boldsymbol{U}}_{d_3}' \end{bmatrix} \begin{bmatrix} f_{1m}\boldsymbol{e}_n \\ \vdots \\ f_{d_3 m}\boldsymbol{e}_n \end{bmatrix} \right\|_2^2 \\
&= \max_{\substack{n=1,\dots,d_1 \\ m=1,\dots,d_3}} \sum_{j=1}^{d_3} |f_{jm}|^2 \|\overline{\boldsymbol{U}}_j' \boldsymbol{e}_n\|_2^2 \\
&= \max_{n=1,\dots,d_1} \sum_{j=1}^{d_3} \|\overline{\boldsymbol{U}}_j' \boldsymbol{e}_n\|_2^2.
\end{aligned}$$

Here, the second equality uses (45) and the last equality follows from the maximum element of the normalized Fourier transform's vectors being $e^0 = 1$. □

Appendix H
Proof of Theorem IV.5

Proof. The result follows by noting the following steps:

(i) Under Assumption A2, the tensor problem in (17) which can be re-written as d_3 separable optimization problems in the Fourier domain:

$$\min_{[\overline{\boldsymbol{U}}_k] \in \overline{\mathcal{G}}(r,d_1)} \frac{1}{T} \sum_{t=1}^{T} \min_{\bar{\boldsymbol{w}}_{t,k} \in \mathbb{C}^r} \frac{1}{2} \|\mathcal{P}_{\Omega_t}(\bar{\boldsymbol{x}}_{t,k}) - \mathcal{P}_{\Omega_t}(\overline{\boldsymbol{U}}_{t,k})\bar{\boldsymbol{w}}_{t,k}\|_F^2.$$

(ii) Under assumption A1, each $\overrightarrow{\boldsymbol{\mathcal{X}}}_t$ is generated as

$$\overrightarrow{\boldsymbol{\mathcal{X}}}_t = \texttt{fold}(\boldsymbol{\mathcal{F}}\bar{\boldsymbol{x}}_t), \quad \bar{\boldsymbol{x}}_t = \overline{\boldsymbol{U}}^* \bar{\boldsymbol{s}}_t, \quad \bar{\boldsymbol{s}}_t \overset{i.i.d}{\sim} \mathcal{CN}(0, \mathbf{I}_{d_3 r}),$$

where $\overline{\boldsymbol{U}}^* \in \mathbb{C}^{d_1 d_3 \times d_3 r}$ is block-diagonal. Hence, we have that each slice $\bar{\boldsymbol{x}}_{t,k} = \overline{\boldsymbol{U}}_k^* \bar{\boldsymbol{s}}_{t,k}$ where $\bar{\boldsymbol{s}}_{t,k} \overset{i.i.d}{\sim} \mathcal{CN}(0, \mathbf{I}_r)$. This follows from the fact that if $\boldsymbol{s}_t = \texttt{unfold}(\overrightarrow{\boldsymbol{\mathcal{S}}}_t) \sim \mathcal{N}(0, \mathbf{I}_{d_3 r})$, then $\bar{\boldsymbol{s}}_t$ is just a linear transform of a Gaussian-distributed random variable under the (normalized) Fourier transform matrix.

(iii) Let $\delta := 0.1/d_3$. Let $\overline{\boldsymbol{U}}_{t,k}$ denote the k^{th} block of $\overline{\boldsymbol{\mathcal{U}}}$ at iteration t, and let $[\overline{\boldsymbol{U}}_{t,k}]_{\Omega_t}$ denote the restriction of $\overline{\boldsymbol{U}}_{t,k}$ to the rows indexed in Ω_t. Since $|\Omega_t| \geq q$ for all t and q satisfies (33), for all $k \in [d_3]$, we obtain

$$\begin{aligned}
q &\geq C_1 \log(d_1)^2 r\mu(\overline{\boldsymbol{U}}_k^*) \log(20rd_3) \\
&\geq \frac{C_1}{2} \log(d_1)^2 r\mu(\overline{\boldsymbol{U}}_{t,k}) \log(20rd_3) \qquad (47)
\end{aligned}$$

for some $C_1 \geq 64/3$, where the last inequality follows from (31) and our assumption that $\epsilon_{t,k} \leq \frac{r}{16d_1}\mu(\overline{\boldsymbol{U}}_k^*)$; see, [6, Lemma 2.5] for more details. Now, it follows from [6, Lemma 2.8], for each $k \in [d_3]$,

$$\begin{aligned}
\|[\bar{\boldsymbol{x}}_{t,k} - \bar{\boldsymbol{p}}_{t,k}]_{\Omega_t}\|_2^2 \geq{}& \frac{|\Omega_t|(1-\xi_{t,k}) - r\mu(\overline{\boldsymbol{U}}_{t,k})\frac{(1+\beta_{t,k})^2}{1-\gamma_{t,k}}}{d_1} \\
&\cdot \|\bar{\boldsymbol{x}}_{t,k} - \overline{\boldsymbol{U}}_{t,k}\overline{\boldsymbol{U}}_{t,k}'\bar{\boldsymbol{x}}_{t,k}\|_2^2,
\end{aligned} \tag{48}$$

with probability at least $1-3\delta$. Here, $\bar{\boldsymbol{p}}_{t,k} = \overline{\boldsymbol{U}}_{t,k}\bar{\boldsymbol{w}}_{t,k}$ where $\bar{\boldsymbol{w}}_{t,k}$ is the optimal weights,

$$\xi_{t,k} := \sqrt{\frac{2\mu(\bar{\boldsymbol{v}}_{t,k})^2}{|\Omega_t|}\log\left(\frac{1}{\delta}\right)},$$

$$\beta_{t,k} := \sqrt{2\mu(\bar{\boldsymbol{v}}_{t,k})\log\left(\frac{1}{\delta}\right)},$$

$$\gamma_{t,k} := \sqrt{\frac{8r\mu(\overline{\boldsymbol{U}}_{t,k})}{3|\Omega_t|}\log\left(\frac{2r}{\delta}\right)}.$$

Since $\delta = 0.1/d_3$, and $|\Omega_t| \geq q$ and q satisfies (47), we get $\gamma_{t,k}^2 \leq 1/4$ for all $k \in [d_3]$. This together with [6, Theorem 2.6] implies that with probability at least $1-\delta$,

$$\lambda_i([\overline{\boldsymbol{U}}_{t,k}]_{\Omega_t}'[\overline{\boldsymbol{U}}_{t,k}]_{\Omega_t}) \in \left[0.5\frac{|\Omega_t|}{d_1}, 1.5\frac{|\Omega_t|}{d_1}\right] \tag{49}$$

for all $i = 1, \dots, r$, where λ_i stands for the i^{th} eigenvalue.

We note that (34) holds with probability at least $1 - \bar{\delta}$. Hence, using the union bound, we have that the bounds (34), (48), (49) all hold with probability at least $1 - (4\delta + \bar{\delta}) = 1 - (\frac{0.4}{d_3} + \bar{\delta})$.

For all $k \in [d_3]$, let $\theta_{t,k}$ denote the angle between $\mathcal{R}(\overline{\boldsymbol{U}}_{t,k})$ and the random observation vector $\bar{\boldsymbol{x}}_{t,k}$, where $\mathcal{R}(\cdot)$ stands for the range. Let $\bar{\boldsymbol{r}}_{t,k} := \bar{\boldsymbol{x}}_{t,k} - \bar{\boldsymbol{p}}_{t,k}$ denote the residual vector for k^{th} block in the Fourier domain. We note that $[\bar{\boldsymbol{r}}_{t,k}]_{\Omega_t^c} = 0$. Now, using bounds (34), (48), (49), it follows from [6, Lemmas 2.9 and 2.10] that for each $k \in [d_3]$,

$$\frac{\|\bar{\boldsymbol{r}}_{t,k}\|^2}{\|\bar{\boldsymbol{p}}_{t,k}\|^2} \geq (0.32)\frac{q}{d_1}\sin^2\theta_{t,k} \tag{50}$$

with probability at least $1 - (\frac{0.4}{d_3} + \bar{\delta})$.

(iv) Following [6, Section 2.5], for all $k \in [d_3]$ we obtain $\epsilon_{t+1,k} \leq \epsilon_{t,k} - \frac{\|\bar{\boldsymbol{r}}_{t,k}\|^2}{\|\bar{\boldsymbol{p}}_{t,k}\|^2} + 55\sqrt{\frac{d_1}{q}}\epsilon_{t,k}^{3/2}$. This together with (50) yields

$$\epsilon_{t+1,k} \leq \epsilon_{t,k} - 0.32\frac{q}{d_1}\sin^2\theta_{t,k} + 55\sqrt{\frac{d_1}{q}}\epsilon_{t,k}^{3/2} \tag{51a}$$
$$\text{with prob. at least } 1 - \left(\frac{0.4}{d_3} + \bar{\delta}\right),$$
$$\epsilon_{t+1,k} \leq \epsilon_{t,k} + 55\sqrt{\frac{d_1}{q}}\epsilon_{t,k}^{3/2} \qquad \text{otherwise.} \tag{51b}$$

Let $\pi_t := \sum_{k=1}^{d_3}\sin^2\theta_{t,k}$. From Step (ii), we have $\bar{\boldsymbol{s}}_{t,k} \overset{i.i.d}{\sim} \mathcal{CN}(0, \mathbf{I}_r)$ for each $k \in [d_3]$ which together with [6, Lemma 2.13] gives

$$\mathbb{E}[\pi_t] = \sum_{k=1}^{d_3}\mathbb{E}[\sin^2\theta_{t,k}] = \sum_{k=1}^{d_3}\frac{\epsilon_{t,k}}{r} = \frac{\epsilon_t}{r}. \tag{52}$$

where the expectation is with respect to the entries of $\bar{\boldsymbol{s}}_{t,k}$ that generate the data, and the last equality follows from (32).

(v) We now put together the theory derived in Steps (i)–(iv) to demonstrate the expected decrease in ϵ_t over a single iteration. Taking a union bound across all d_3 blocks in (51), we obtain

$$\begin{aligned}
&\epsilon_{t+1} = \sum_{k=1}^{d_3}\epsilon_{t+1,k} \\
&\leq \sum_{k=1}^{d_3}\epsilon_{t,k} - 0.32\frac{q}{d_1}\sum_{k=1}^{d_3}\sin^2\theta_{t,k} + 55\sqrt{\frac{d_1}{q}}\sum_{k=1}^{d_3}\epsilon_{t,k}^{3/2} \\
&\leq \sum_{k=1}^{d_3}\epsilon_{t,k} - 0.32\frac{q}{d_1}\sum_{k=1}^{d_3}\sin^2\theta_{t,k} + 55\sqrt{\frac{d_1}{q}}\left(\sum_{k=1}^{d_3}\epsilon_{t,k}\right)^{3/2} \\
&= \epsilon_t - 0.32\frac{q}{d_1}\pi_t + 55\sqrt{\frac{d_1}{q}}\epsilon_t^{3/2}
\end{aligned}$$

with probability at least $0.6 - d_3\bar{\delta}$ while $\epsilon_{t+1} \leq \epsilon_t + 55\sqrt{\frac{d_1}{q}}\epsilon_t^{3/2}$ otherwise. Taking the expectation with respect to the randomness of the data and using (52), we obtain

$$\begin{aligned}
\mathbb{E}[\epsilon_{t+1} \mid \epsilon_t] &\leq \epsilon_t - (0.32)(0.6 - d_3\bar{\delta})\frac{q}{d_1 r}\epsilon_t + 55\sqrt{\frac{d_1}{q}}\epsilon_t^{3/2} \\
&\leq \left(1 - 0.16(0.6 - d_3\bar{\delta})\frac{q}{d_1 r}\right)\epsilon_t.
\end{aligned}$$

Here, the last inequality follows since

$$\begin{aligned}
55\sqrt{\frac{d_1}{q}}\epsilon_t^{1/2} &\leq 55\sqrt{\frac{d_1}{q}}(0.0029)(0.6 - d_3\bar{\delta})\frac{q^{3/2}}{d_1^{3/2}r} \\
&\leq (0.16)(0.6 - d_3\bar{\delta})\frac{q}{d_1 r},
\end{aligned}$$

where the second inequality uses (35b).

□

Appendix I
Experiment hyperparameters

For the algorithms listed in Section V, we used the following grids of values for the parameter search:

- **TeCPSGD grids**: rank $= [1, 5, 10, 15, 20, 25, 30, 50]$, $\lambda = [10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 0.5, 1]$, step size = $[1, 10, 10^2, 10^3, 10^4, 10^5, 10^6]$.
- **OLSTEC grids**: rank $= [1, 5, 10, 15, 20, 25, 30, 50]$, $\lambda = [10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 0.5, 1]$, $\mu = [10^{-8}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1]$.
- **STC grid**: multi-ranks (r_1, r_2, r_3) where each r_i ranges from $[1, 5, 10, 15, 20, 25, 30]$.
- **TCTF grid**: tubal rank ranging from $[1, 5, 7, 10, 15, 20, 25, 30]$.