Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

Mode Coresets for Efficient, Interpretable Tensor Decompositions: An Application to Feature Selection in fMRI Analysis

BEN GABRIELSON[†1], HANLU YANG[†1],(Graduate Student Member, IEEE), TRUNG VU[1], VINCE CALHOUN[2],(Fellow, IEEE), and TÜLAY ADALI[1], (Fellow, IEEE)

[†] 1st Authors.
[1]Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, Baltimore MD.
[2]Tri-Institutional Center for Translational Research in Neuroimaging and Data Science (TReNDS), Georgia State University, Georgia Institute of Technology, Emory University, Atlanta, Georgia.

Corresponding authors: Ben Gabrielson (bengabr1@umbc.edu), Hanlu Yang (hyang3@umbc.edu).

This work was supported in part by grants NSF 2316420, NIH R01MH118695, NIH R01MH123610, and NIH R01AG073949. The computational hardware used is part of the UMBC High Performance Computing Facility (HPCF), supported by the U.S. NSF through the MRI and SCREMS programs (grants CNS-0821258, CNS-1228778, OAC-1726023, DMS-0821311), with additional substantial support from the University of Maryland, Baltimore County (UMBC).

ABSTRACT Generalizations of matrix decompositions to multidimensional arrays, called *tensor decompositions*, are simple yet powerful methods for analyzing datasets in the form of tensors. These decompositions model a data tensor as a sum of rank-1 tensors, whose *factors* provide uses for a myriad of applications. Given the massive sizes of modern datasets, an important challenge is how well computational complexity scales with the data, balanced with how well decompositions approximate the data. Many efficient methods exploit a small subset of the tensor's elements, representing most of the tensor's variation via a basis over the subset. These methods' efficiencies are often due to their randomized natures; however, deterministic methods can provide better approximations, and can perform *feature selection*, highlighting a meaningful subset that well-represents the entire tensor. In this paper, we introduce an efficient subset-based form of the Tucker decomposition, by selecting *coresets* from the tensor modes such that the resulting core tensor can well-approximate the full tensor. Furthermore, our method enables a novel feature selection scheme unlike other methods for tensor data. We introduce methods for random and deterministic coresets, minimizing error via a measure of discrepancy between the coreset and full tensor. We perform the decompositions on simulated data, and perform on real-world fMRI data to demonstrate our method's feature selection ability. We demonstrate that compared with other similar decomposition methods, our methods can typically better approximate the tensor with comparably low computational complexities.

INDEX TERMS Tensor Decomposition, Tucker Decomposition, Higher Order Singular Value Decomposition, Coresets, Tensor CUR Decomposition, Subset Selection, Feature Selection, fMRI.

I. INTRODUCTION

Datasets in the modern era often take the form of large multidimensional arrays called *tensors*. A tensor can be understood as a collection of values (e.g. measurements) that are each associated with a corresponding list of N array indices, where N denotes the order of the tensor. Whereas a *vector* is a first order tensor and a *matrix* is a second order tensor, the analysis of third or higher order tensors is the focus of those methods formally called tensor decompositions. Tensor decompositions generalize matrix decompositions to higher order tensors, approximating a tensor dataset as a tensor product of several factor matrices that have various use cases. These generalizations notably endow tensor decompositions with the ability to model *multilinear* relationships within the data, concisely modeling the relationships across different modes of the tensor. Furthermore, tensor decompositions provide a low-rank model of the tensor that typically is orders of magnitude smaller in memory than the original tensor. A tensor decomposition's factors are typically useful for describing the latent characteristics of the tensor, and are often used for providing a generative model of the data. All in all, tensor decompositions provide tools for a wide range of uses, such as dimension reduction [1]–[5], feature extraction [6]–[9], denoising [10]–[15], missing data completion [15]–[20], dictionary learning [21]–[25], signal processing [26]–[32], and various others. Applications of tensor decompositions are widespread and include chemometrics [1], [33], [34], psychometrics [35], [36], econometrics [37], [38], analysis of medical imaging modalities [7], [39]–[48], radar and communication applications [30], [49], [50], applications to machine learning [26], [51], [51]–[56], and many others.

Perhaps one of the simplest tensor decompositions is what is often called the canonical polyadic decomposition (CPD) [57], [58], which approximates a tensor as the sum of R rank-1 tensors where R is a user-defined positive integer. CPD can be understood as a higher-order generalization of matrix low-rank decompositions, which decompose a matrix into a sum of rank-1 matrices that best approximates the original matrix. However, whereas matrix rank decompositions are typically not unique unless additional constraints are imposed, the CPD is often unique under much milder conditions. This results in unique factors that reveal the latent structure of the data under fewer required assumptions [1], [59]–[61].

Another useful form of tensor decomposition is the Tucker Decomposition [62], [63]. The Tucker decomposition is a general form of tensor decomposition that represents an Nth order tensor as the tensor product of N factor matrices with an Nth order "core" tensor: a small tensor that can be considered a compressed version of the original tensor. A notable specific type of Tucker decomposition is the higher-order singular value decomposition (HOSVD) [63], [64], the direct generalization of the matrix singular value decomposition (SVD) to tensors. HOSVD is analytically represented by its factor matrices being the singular vectors of each "unfolding" (matricization) of the original tensor, in which case the core tensor can be interpreted as a tensorial form of principal components. While the CPD and Tucker are perhaps the most popularly used tensor decompositions, since their introduction a wide variety of other decompositions have been introduced and used successfully. These include the Tensor Train decomposition [65], [66], hierarchical Tucker decompositions [67], [68], tensor block-term decompositions [69], [70], coupled matrix-tensor factorizations [44], [71], [72], and online tensor decompositions [73]–[76].

Most tensor decompositions perform their optimization routines by breaking the problem of estimating all N factor matrices into N simpler subproblems. This typically involves solving for each factor matrix one at a time, by unfolding the tensor with respect to each of the N modes and subsequently solving for (or updating) a corresponding mode's factor matrix. While these routines simplify optimization by allowing decompositions to be solved with matrix-based methods, tensor decompositions nevertheless rely on multiplying high-dimensional matrix representations of the tensor data, which can become computationally expensive with exceedingly massive tensors. These challenges have greatly motivated computationally efficient methods for tensor decompositions, especially those that retain simple models with excellent approximation and explainability.

Many efficient tensor decompositions are direct generalizations of matrix decompositions. With matrices, a particularly useful strategy has been to approximate a matrix via projecting onto the span of only a small subset of columns. These are referred to as column subset selection (CSS) methods, of which include the matrix CUR decomposition [77], [78] which approximates a given matrix by both a subset of columns $\mathbf{C}$ and a subset of rows $\mathbf{R}$. Subset selection methods are distinguished by those that select a subset randomly, with a focus on faster decompositions, or those that select a subset deterministically, with a focus on better approximation and for performing *feature selection*: identifying particularly representative elements of the data that well-describe the rest of the data. Extensions of these matrix decompositions to tensors exist as types of Tucker decompositions that are called tensor CUR decompositions [79]–[84], which use subsets of elements from multiple modes of a tensor to provide a multilinear basis for the entire tensor. Due to their simple procedures, tensor CUR decompositions are among the fastest tensor decompositions, and can also provide good approximations of tensors with reasonably large subset sizes, yet may suffer with smaller subset sizes. These methods exclusively select subsets randomly, rather than deterministically. Extensions of deterministic subset-based methods may also be desirable for tensors, especially in the interest of determining well-representative subsets of the data.

Tensorial feature selection has been accomplished in [85]–[87] but only in the context of supervised learning for classification, where tensors are accompanied by labels and feature selection is a function of the labels. An unsupervised feature selection method for third order tensors was proposed in [88], which takes subsets from a single mode of the tensor. However, features in these subsets differ depending on what elements they correspond to in another "view" mode, and thus may be harder to interpret. Furthermore, these subsets are acquired after performing a CPD, whereas the methods we consider in this paper actually use the subsets to perform efficient tensor decompositions. To our knowledge there have been no other extensions of deterministic subset-based methods to tensor data in the general unsupervised setting, and none for multiple modes of the tensor.

In this paper, we introduce an efficient weighted subset-based type of Tucker decomposition, similar in form to the tensor CUR decompositions and the sequentially truncated higher-order singular value decomposition (ST-HOSVD) [89]. Notably, the deterministic variation of our method provides a novel unsupervised feature selection algorithm for tensor data, selecting subsets from one or more modes that are reasonably best able to summarize the structure of the tensor. Sequentially across a tensor's N modes, we select from each mode a *coreset*, i.e. a weighted subset of elements [90]–[94], that reasonably minimizes a measure of discrepancy between the coreset and the entire mode, which in turn minimizes the mean squared error cost between the tensor and its approximation. We connect the discrepancy to the cost function of HOSVD, showing that use of weighed subsets provides a better minimization to the cost than unweighted subsets used in tensor CUR decompositions. We consider two methods: one based on random coreset selection, sampling according to a weighted probability distribution, and one based on deterministic coreset selection, utilizing an efficient weighted kernel herding (WKH) [95] procedure. For a given coreset, we select the corresponding coreset weights via an efficient nonnegative least squares (NNLS) minimizing the discrepancy between the coreset and the entire mode. We analyze performance of our two methods on large datasets, testing with both simulated data and real functional magnetic resonance imaging (fMRI) data via functional connectivity matrices (FNCs) arranged as a large tensor. Comparing with similar Tucker-type methods, such as variations of the Tensor CUR decomposition and randomized HOSVD methods, we demonstrate that our methods are highly efficient, provide good approximation performance, and can be converted to a HOSVD decomposition with strong estimation quality.

The paper is organized as follows. Section II introduces preliminary concepts regarding matrices and tensors, including several basic methods for matrix and tensor decompositions. Section III explains efficient generalizations of subset-based methods for matrix decompositions to tensor decompositions, such as the tensor CUR decompositions. Section IV introduces our proposed sequential coreset-based tensor decomposition methods, which we refer to as tensor coreset decompositions (TCD). Section V provides results of our methods, compared with various other methods, on both simulated tensor datasets and a real fMRI FNC tensor dataset. Section VI concludes the paper and overviews the contributions.

II. PRELIMINARIES

A. NOTATION

Throughout the paper, we use notation that is summarized in Table 1, and is consistent with notation of other works that discuss tensor decompositions (e.g., [1]).

We denote scalars by lowercase unbolded letters (e.g., x), vectors by lowercase bolded letters (e.g., $\mathbf{x}$), matrices by uppercase bolded letters (e.g. $\mathbf{X}$), and higher order tensors (order three or higher) by calligraphic bolded letters (e.g. $\boldsymbol{\mathcal{X}}$).

The *order* of a tensor, N, also referred to as the number of *modes*, can be loosely thought of as the number of dimensions in the tensor, but more precisely it is the number of indices needed to index an entry in the tensor. For instance, a third order tensor $\boldsymbol{\mathcal{X}}$ has a corresponding (i_1, i_2, i_3) element denoted by $(\boldsymbol{\mathcal{X}})_{(i_1 i_2 i_3)}$. Each index corresponds to a different mode of the tensor, and is bounded by the *dimensionality* of that mode. For example, given a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{D_1\times D_2\times D_3}$, the dimensionality of the first mode is D_1. In general, when dealing with Nth-order tensors, we refer to the dimensionality of the nth node by D_n, and a particular index from that mode by i_n, for $i_n = 1, \dots, D_n$, and $n = 1, \dots, N$.

As our paper utilizes subsets of the tensor, we define a *subtensor* as a subset of elements in the tensor corresponding to some set of indices. We define index sets over a given nth mode of a tensor by unbolded calligraphic letters $\mathcal{I}_n$, and use a colon to otherwise indicate all elements of a mode. For example, $(\boldsymbol{\mathcal{X}})_{(i,:,:)}$ denotes the ith element of the first mode, and $(\boldsymbol{\mathcal{X}})_{(\mathcal{I}_1,:,:)}$ denotes a subset of elements in the first mode corresponding to the index set $\mathcal{I}_1$.

An important operation in tensor decompositions is the matricization of a tensor, also called the unfolding. We denote the nth mode unfolding of a tensor $\mathcal{X} \in \mathbb{R}^{D_1\times D_2\times\dots\times D_N}$ by the matrix $\mathbf{X}_{(n)} \in \mathbb{R}^{D_n\times\tilde{D}_n}$, where $\tilde{D}_n = \prod_{m=1, m\neq n}^{N} D_m$ is the product of all other mode's dimensionalities. The ith row of the nth mode unfolding is the vectorization of the ith element in the nth mode, e.g. $(\mathbf{X}_{(1)})_{(3,:)}$ denotes the third row in the first mode unfolding of $\mathbf{X}$ and is equal to vec$(\boldsymbol{\mathcal{X}}_{(3,:,:,\dots,:)})^\top$, the third element of the first mode.

The *rank* of a tensor $\boldsymbol{\mathcal{X}}$ is defined as the smallest number of rank-1 tensors that exactly sum to $\boldsymbol{\mathcal{X}}$. Unlike with matrices, determining the tensor rank is difficult for most real-world tensors. A more well-defined notion of a tensor's rank structure are the *n-ranks*, the ranks of each unfolding $\mathbf{X}_{(n)}$.

If the nth mode unfolding of a tensor $\mathbf{X}_{(n)} \in \mathbb{R}^{D_n\times\tilde{D}_n}$ is left multiplied by a matrix $\mathbf{U} \in \mathbb{R}^{J_n\times D_n}$, the resulting product $\mathbf{G} = \mathbf{U}\,\mathbf{X}_{(n)} \in \mathbb{R}^{J_n\times\tilde{D}_n}$ is equivalently represented in the tensor domain by the nth mode tensor product $\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{X}} \times_n \mathbf{U} \in \mathbb{R}^{D_1\times\dots\times D_{n-1}\times J_n\times D_{n+1}\times\dots\times D_N}$.

The norm of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{D_1\times\dots\times D_N}$ is defined by:

$$\|\boldsymbol{\mathcal{X}}\|_{\mathrm{F}} = \left(\sum_{i_1=1}^{D_1}\dots\sum_{i_N=1}^{D_N}(\boldsymbol{\mathcal{X}})^2_{(i_1,\dots,i_N)}\right)^{\frac{1}{2}}$$

Notation	Definition
x	scalar
$\mathbf{x}$	vector
$\mathbf{X}$	matrix
$\boldsymbol{\mathcal{X}}$	tensor
N	number of modes
D_n	dimensionality of nth mode, for $n = 1, \dots, N$
R_n	n-rank: rank of $\mathbf{X}_{(n)}$
$\hat{R}_n$	number of factors in decomposition of nth mode
$\tilde{D}_n$	$\prod_{m=1, m\neq n}^{N} D_m$
$\tilde{D}_n^{(n)}$	$(\prod_{m=1}^{n-1}\hat{R}_m)\,(\prod_{m=n+1}^{N} D_m)$
$\tilde{T}_n$	$\prod_{m=1, m\neq n}^{N}\hat{R}_m$
i_n	ith index of the nth mode, for $i_n = 1, \dots, D_n$
$\mathcal{I}_n$	index set of nth mode (with cardinality $\hat{R}_n$)
$\mathcal{I} = \{\mathcal{I}_1, \dots, \mathcal{I}_N\}$	All modes' index sets
$(\boldsymbol{\mathcal{X}})_{(:,\dots,i_n,\dots,:)}$	element i_n of the nth mode
$(\boldsymbol{\mathcal{X}})_{(:,\dots,\mathcal{I}_n,\dots,:)}$	subset of nth mode corresponding to indices $\mathcal{I}_n$
$\mathbf{X}_{(n)}$	nth mode unfolding of $\boldsymbol{\mathcal{X}}$
$\boldsymbol{\mathcal{X}} \times_n \mathbf{A}_n$	nth mode tensor product of $\boldsymbol{\mathcal{X}}$ with $\mathbf{A}_n$
$\mathbf{X}^\top$	matrix transpose
$\mathbf{X}^\dagger$	matrix pseudoinverse
$\mathbf{X}\,\mathbf{X}^\dagger$	projection matrix
$\mathbf{M}_{\mathcal{I}_n}$	nth mode "mapping" matrix corresponding to $\mathcal{I}_n$

Table 1. Notation used in this paper.

Algorithm 1: **HOSVD**

```
Input: $\mathcal{X} \in \mathbb{R}^{D_1 \times \ldots \times D_N}$ ($N$-mode tensor),
       $[\hat{R}_1, \ldots, \hat{R}_N]$ (number of factors per mode)
Output: $\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A}_1 \times \ldots \times_M \mathbf{A}_N$ , where
        $\mathcal{G} \in \mathbb{R}^{\hat{R}_1 \times \ldots \times \hat{R}_N}$ (core tensor),
        $\{\mathbf{A}_1, \ldots, \mathbf{A}_N\}$ (factor matrices)

  for each mode $n = 1 : N$
      unfold (matricize) tensor w.r.t. $n$th mode
          $\mathcal{X} \rightarrow \mathbf{X}_{(n)} \in \mathbb{R}^{D_n \times \tilde{D}_n}$, with $\tilde{D}_n = \prod_{\substack{m=1 \\ m \neq n}}^{N} D_m$
      compute $\mathbf{A}_n \in \mathbb{R}^{D_n \times \hat{R}_n}$,
          the $\hat{R}_n$ left singular vectors of $\mathbf{X}_{(n)}$
  end for
  $\mathcal{G} = \mathcal{X} \times_1 \mathbf{A}_1^\top \times_2 \ldots \times_N \mathbf{A}_N^\top$
```

In the next subsections, we first discuss Tucker and HOSVD decompositions for tensors, and note their complexities. We then discuss column subset selection (CSS) methods for reducing complexities of matrix decompositions, and then discuss generalizations of these methods to tensors.

B. TUCKER AND HOSVD DECOMPOSITIONS

The Tucker decomposition [62], [63] is a general type of tensor decomposition that approximates an Nth order tensor $\mathcal{X} \in \mathbb{R}^{D_1 \times \ldots \times D_N}$ by the tensor product of N factor matrices $\mathbf{A}_n$ $(n = 1, \ldots N)$, with a smaller core tensor $\mathcal{G}$. The general cost function for Tucker decompositions takes the form:

$$\mathcal{J}(\mathcal{G}, \mathbf{A}_1, \ldots, \mathbf{A}_N) = \| \mathcal{X} - \mathcal{G} \times_1 \mathbf{A}_1 \times_2 \ldots \times_N \mathbf{A}_N \|_F^2 \tag{1}$$

where $\mathbf{A}_n \in \mathbb{R}^{D_n \times \hat{R}_n}$ is the nth mode's factor matrix, $\mathcal{G} \in \mathbb{R}^{\hat{R}_1 \times \ldots \times \hat{R}_N}$ is the core tensor, and $\hat{R}_n$ are the number of factors chosen for the nth mode, which are often closely related to the tensor's n-ranks R_n, for $n = 1, \ldots, N$.

The Tucker decomposition is not unique without any further constraints. There are a variety of ways to achieve a unique Tucker decomposition over a tensor, including several subset-based approaches such as the tensor CUR decomposition and the method that we later propose in this paper.

A useful Tucker decomposition is the HOSVD [63], [64], a natural generalization of SVD to tensors. HOSVD's factor matrices of a tensor $\mathcal{X}$ are analytically given as the left singular vectors of each unfolding of $\mathcal{X}$, and the core tensor is obtained from a tensor product of these factor matrices with $\mathcal{X}$. The HOSVD procedure is described in Algorithm 1.

Several variations of HOSVD have been introduced since its inception to improve its efficiency, with one of the most used variations being the sequentially truncated HOSVD (ST-HOSVD) [89]. Across each nth mode of the tensor, ST-HOSVD first estimates a mode's factor matrix from the left singular vectors of the nth mode unfolding $\mathbf{X}_{(n)}$ (just as done with HOSVD), and then replaces $\mathcal{X}$ with the core tensor formed by the tensor product of this factor matrix with $\mathcal{X}$. Over calculation of the N mode factor matrices, the current tensor progressively reduces in size until it becomes the final core tensor and all N factor matrices are obtained. The ST-HOSVD procedure is described in Algorithm 2.

If we denote the SVD of each $\mathbf{X}_{(n)}$ in the for loop of Algorithm 2 by $\mathbf{X}_{(n)} = \mathbf{U}_{\mathbf{X}_{(n)}} \Sigma_{\mathbf{X}_{(n)}} \mathbf{V}_{\mathbf{X}_{(n)}}^\top$, such that $\mathbf{A}_n = \mathbf{U}_{\mathbf{X}_{(n)}}$, it follows that ST-HOSVD's truncation strategy sequentially replaces $\mathbf{X}_{(n)}$ with its top $\hat{R}_n$ right principal components (PCs) $\Sigma_{\mathbf{X}_{(n)}} \mathbf{V}_{\mathbf{X}_{(n)}}^\top$, thus best preserving the approximation of the original tensor while reducing the dimensionality of operations across all remaining modes.

We now compare the computational complexities of HOSVD and ST-HOSVD. For simplicity, we assume that the order of modes truncated with ST-HOSVD is $n = 1, \ldots, N$. HOSVD's computational complexity is $\mathcal{O}\left(\sum_{n-1}^{N} \min\left(D_n^2 \tilde{D}_n, \tilde{D}_n^2 D_n\right)\right)$, dominated by the N SVD-unfoldings for large tensors. ST-HOSVD considerably reduces this complexity to $\mathcal{O}\left(\sum_{n-1}^{N} \min\left(D_n^2 \tilde{D}_n^{(n)}, (\tilde{D}_n^{(n)})^2 D_n\right)\right)$, where $\tilde{D}_n^{(n)} = (\prod_{m=1}^{n-1} \hat{R}_m)\,(\prod_{m=n+1}^{N} D_m)$, here $\hat{R}_m$ is the number of factors in the mth mode. However with ST-HOSVD, the first few modes' SVDs are similar in complexity to those calculated with HOSVD. This leads ST-HOSVD to still be computationally expensive when dealing with large tensors, motivating more scalable decomposition methods.

In the next section, we overview subset-based methods for reducing complexity of matrix decompositions, from which we then overview their various generalizations to tensors.

C. MATRIX DECOMPOSITIONS BY COLUMN SUBSET SELECTION (CSS)

This subsection gives a general overview of column subset selection methods for matrices. For a more detailed discussion of the topic, we refer the reader to [96]–[98].

CSS methods approximate a matrix $\mathbf{X} \in \mathbb{R}^{M \times N}$ by selecting a subset of columns of the matrix, selecting either

Algorithm 2: **ST-HOSVD**

```
Input: $\mathcal{X} \in \mathbb{R}^{D_1 \times \ldots \times D_N}$ ($N$-mode tensor),
       $[\hat{R}_1, \ldots, \hat{R}_N]$ (number of factors per mode)
Output: $\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A}_1 \times \ldots \times_M \mathbf{A}_N$ , where
        $\mathcal{G} \in \mathbb{R}^{\hat{R}_1 \times \ldots \times \hat{R}_N}$ (core tensor),
        $\{\mathbf{A}_1, \ldots, \mathbf{A}_N\}$ (factor matrices)

  for each mode $n = 1 : N$
      unfold (matricize) tensor w.r.t. $n$th mode
          $\mathcal{X} \rightarrow \mathbf{X}_{(n)} \in \mathbb{R}^{D_n \times (\tilde{D}_n)^{(n)}}$,
          with $(\tilde{D}_n)^{(n)} = (\prod_{m=1}^{n-1} \hat{R}_m)\,(\prod_{m=n+1}^{N} D_m)$
      compute $\mathbf{A}_n \in \mathbb{R}^{D_n \times \hat{R}_n}$,
          the $\hat{R}_n$ left singular vectors of $\mathbf{X}_{(n)}$
      truncate the unfolded tensor
          $\mathbf{X}_{(n)} \rightarrow \mathbf{A}_n^\top \mathbf{X}_{(n)} \in \mathbb{R}^{\hat{R}_n \times (\tilde{D}_n)^{(n)}}$
      un-matricize the tensor
          $\mathbf{X}_{(n)} \rightarrow \mathcal{X} \in \mathbb{R}^{\hat{R}_1 \times \ldots \times \hat{R}_n \times D_{n+1} \times \ldots \times D_n}$
  end for
  $\mathcal{G} \rightarrow \mathcal{X}$
```

randomly or deterministically, and then approximating $\mathbf{X}$ by projecting onto the span of the subset. If we denote $\mathbf{X}_{\mathcal{I}_s} \triangleq (\mathbf{X})_{(:,\mathcal{I}_s)} \in \mathbb{R}^{M \times N_s}$ as the matrix formed by a N_s subset of columns, corresponding to some index set $\mathcal{I}_s$, the approximation error for some choice of $\mathbf{X}_{\mathcal{I}_s}$ is given by:

$$\begin{aligned}\mathcal{J}(\mathcal{I}_s) &= \| \mathbf{X} - \mathbf{X}_{\mathcal{I}_s}(\mathbf{X}_{\mathcal{I}_s}^\top \mathbf{X}_{\mathcal{I}_s})^{-1}\mathbf{X}_{\mathcal{I}_s}^\top \mathbf{X} \|_{\mathrm{F}}^2 \\ &= \| \mathbf{X} - \mathbf{X}_{\mathcal{I}_s}\mathbf{X}_{\mathcal{I}_s}{}^{\dagger} \mathbf{X} \|_{\mathrm{F}}^2 \\ &= \| \mathbf{X} - \mathbf{P}_{\mathbf{X}_{\mathcal{I}_s}} \mathbf{X} \|_{\mathrm{F}}^2 \qquad (2)\end{aligned}$$

where $\mathbf{X}_{\mathcal{I}_s}{}^{\dagger} = (\mathbf{X}_{\mathcal{I}_s}^\top \mathbf{X}_{\mathcal{I}_s})^{-1}\mathbf{X}_{\mathcal{I}_s}^\top$ is the pseudoinverse of $\mathbf{X}_{\mathcal{I}_s}$ (such that $\mathbf{X}_{\mathcal{I}_s}{}^{\dagger}\, \mathbf{X}_{\mathcal{I}_s} = \mathbf{I} \in \mathbb{R}^{N_s \times N_s}$, and $\mathbf{P}_{\mathbf{X}_{\mathcal{I}_s}} = \mathbf{X}_{\mathcal{I}_s}\mathbf{X}_{\mathcal{I}_s}{}^{\dagger} \in \mathbb{R}^{M \times M}$ is the projection matrix corresponding to the column space of $\mathbf{X}_{\mathcal{I}_s}$. This is equivalently given by:

$$\mathcal{J}(\mathcal{I}_s) = \| \mathbf{X} - \mathbf{X}_{\mathcal{I}_s} \mathbf{M}_{\mathcal{I}_s} \|_{\mathrm{F}}^2 \qquad (3)$$

where $\mathbf{M}_{\mathcal{I}_s} = (\mathbf{X}_{\mathcal{I}_s}^\top \mathbf{X}_{\mathcal{I}_s})^{-1}\mathbf{X}_{\mathcal{I}_s}^\top \mathbf{X} \in \mathbb{R}^{N_s \times N}$ is a matrix mapping columns of $\mathbf{X}$ onto the span of $\mathbf{X}_{\mathcal{I}_s}$, which in later sections we refer to as a "mapping" matrix.

1) Randomized CSS

Randomized CSS methods operate by assigning a weighted probability distribution to the columns and then sampling according to this distribution. Uniform sampling of the columns (giving equal sampling probability to each column) generally produces bad approximations of a matrix, especially if the columns are heterogeneous. Instead, sampling distributions are often based on probabilities weighted by the squared norm of columns, i.e. "norm sampling" [77], [99], or approximated statistical leverage scores [100]. In our paper, we focus on norm sampling, which is the most computationally efficient of the sampling-based methods, and we note that norm sampling is also conventional in tensor-based methods [79], [82], [101]. It has been proven in [99] that norm sampling provides the following error guarantees: given a matrix $\mathbf{X} \in \mathbb{R}^{M \times N}$ and values for ϵ, δ, and a defined upper limit to the rank k of $\mathbf{P}_{\mathbf{X}_{\mathcal{I}_s}} = \mathbf{X}_{\mathcal{I}_s}\mathbf{X}_{\mathcal{I}_s}{}^{\dagger}$, then a norm sampled selection for $\mathbf{X}_{\mathcal{I}_s}$ satisfies the following error probability:

$$\Pr\left\{ \|\mathbf{X} - \mathbf{P}_{\mathbf{X}_{\mathcal{I}_s}} \mathbf{X}\|_{\mathrm{F}}^2 \le \|\mathbf{X} - \mathbf{X}_k\|_{\mathrm{F}}^2 + \epsilon \|\mathbf{X}\|_{\mathrm{F}}^2 \right\} \ge 1 - \delta$$

where $\mathbf{X}_k$ is the best rank-k approximation to $\mathbf{X}$, and $0 \le \delta \le 1$ is the probability of failure.

Furthermore, it has been proven in [77] that given a norm sampled subset of columns $\mathbf{X}_{\mathcal{I}_s}$, after rescaling the columns of $\mathbf{X}_{\mathcal{I}_s}$ to be the same norm:

$$(\mathbf{X}_{\mathcal{I}_s})_{(:,i)} \to \frac{1}{\sqrt{N_s}} \frac{\| \mathbf{X} \|_{\mathrm{F}}}{\| (\mathbf{X}_{\mathcal{I}_s})_{(:,i)} \|_{\mathrm{F}}} (\mathbf{X}_{\mathcal{I}_s})_{(:,i)}\,, \qquad (4)$$

that the following error probability is satisfied $\forall\, \epsilon \ge 0$:

$$\Pr\left\{ \|\mathbf{X}\mathbf{X}^\top - \mathbf{X}_{\mathcal{I}_s}\mathbf{X}_{\mathcal{I}_s}^\top\|_{\mathrm{F}}^2 \le \frac{\eta}{(N_s)^{\frac{1}{2}}} \| \mathbf{X} \|_{\mathrm{F}}^2 \right\} \ge 1 - \delta$$

where $\eta = 1 + (8\log(\delta^{-1}))^{\frac{1}{2}}$.

If we denote the SVD of $\mathbf{X}$ by $\mathbf{X} = \mathbf{U}_{\mathbf{X}}\, \Sigma_{\mathbf{X}}\, \mathbf{V}_{\mathbf{X}}^\top$, this particular result suggests that with a high enough sample size N_s, a norm sampled $\mathbf{X}_{\mathcal{I}_s}$ can adequately approximate the left PCs $\mathbf{U}_{\mathbf{X}}\, \Sigma_{\mathbf{X}}$ of $\mathbf{X}$ with a high probability, by re-scaling the columns according to (4). We later will refer to this result when introducing our coreset-based method.

2) Deterministic CSS

Deterministic CSS methods are combinatorial methods for selecting a "best" representative subset of columns, where "best" is relative to the method used. The problem of finding a subset that exactly minimizes the approximation cost over all possible subsets has been acknowledged as being UG-hard (where "UG" refers to the unique games conjecture) [102], in which case deterministic algorithms mainly focus on obtaining a reasonably "best" subset in a reasonable amount of time. These methods can also effectively serve as *feature selection* methods, and thus there is a large overlap between methods that can be used for feature selection and those used for deterministic CSS. However, the design of CSS methods typically puts a greater emphasis on the scalability of methods, especially with the high-dimensional combinatorial problems posed by large matrices or tensors.

Perhaps the most popular method for deterministic CSS is to use the greedy algorithm, which consecutively searches for a new column to add onto a subset such that the resulting new subset best approximates the full matrix. The greedy CSS algorithm was first studied in [103], and has been demonstrated to be both scalable to large numbers of columns and provide high-quality representative subsets [97], [104]–[108].

As one may expect, deterministic CSS methods provide better approximation than randomized methods and incur better error guarantees. The tightest bounds for deterministic CSS depend on the singular values of $\mathbf{X}$; intuitively, those matrices whose singular values have higher rate of decay are simpler matrices which require much fewer columns to well-approximate the matrix. In [98], the following bound was proven on greedy CSS:

$$\| \mathbf{X} - \mathbf{P}_{\mathbf{X}_{\mathcal{I}_s}} \mathbf{X} \|_{\mathrm{F}}^2 \ge (1 - \epsilon) \| \mathbf{X} - \mathbf{X}_k \|_{\mathrm{F}}^2$$

where $\mathbf{X}_k$ is the best rank-k approximation to $\mathbf{X}$, $r \ge 16k\,(\epsilon\,\sigma_{\min}(\mathbf{X}_k))^{-1}$ is the number of steps taken by the greedy algorithm, and $\sigma_{\min}(\mathbf{X}_k)$ is the smallest singular value of $\mathbf{X}_k$. Similar results have been proven in Theorem 3 of [109]. A shared result amongst these works is that by only taking slightly more than k columns with greedy CS, the approximated matrix is less than a $1 - \epsilon$ factor from the optimal choice of k columns.

III. SUBSET METHODS GENERALIZED TO TENSORS (METHODS TO APPROXIMATE THE HOSVD)

As tensor decompositions frequently invoke matrix operations with the tensor unfolding, matrix approximation techniques have found great use for accelerating tensor decompositions [79]–[83], [101]. As our proposed method is most analogous to the HOSVD, we focus only on those subset-based methods for performing a Tucker decomposition in the form of an approximated HOSVD.

These methods generally estimate a form of Tucker decomposition that is not a HOSVD decomposition, but can be used to approximate one. In order to provide an approximate HOSVD decomposition, we may convert any method's corresponding Tucker decomposition to a HOSVD decomposition via the procedure [82] outlined in Algorithm 3.

Algorithm 3: **Convert Tucker decomposition to HOSVD**

```
Input: $\tilde{\mathcal{G}} \in \mathbb{R}^{\hat{R}_1 \times \dots \times \hat{R}_N}$ (core tensor)
       $\{\tilde{\mathbf{A}}_1, \dots, \tilde{\mathbf{A}}_N\}$ (factor matrices)
Output: $\mathcal{G} \in \mathbb{R}^{\hat{R}_1 \times \dots \times \hat{R}_N}$ (HOSVD core tensor),
        $\{\mathbf{A}_1, \dots, \mathbf{A}_N\}$ (HOSVD factor matrices)

  for each mode $n = 1 : N$
      factorize $\tilde{\mathbf{A}}_n$ using the QR decomposition:
        $[\mathbf{Q}_n, \mathbf{R}_n] = \text{qr}(\tilde{\mathbf{A}}_n)$
      replace $\tilde{\mathcal{G}} \to \tilde{\mathcal{G}} \times_n \mathbf{R}_n$
  end for
  perform HOSVD on the new core tensor:
        $[\mathcal{G}, \mathbf{A}_1, \dots, \mathbf{A}_N] = \text{HOSVD}(\tilde{\mathcal{G}})$

  for each mode $n = 1 : N$
      replace $= \mathbf{A}_n \to \mathbf{Q}_n \mathbf{A}_n$
  end for
```

There are various different strategies to provide a Tucker decomposition over a tensor $\mathcal{X}$ via exploiting the previously discussed matrix approximation techniques over the tensor unfoldings $\mathbf{X}_{(n)}$. These strategies can generally be separated into two distinct camps with differing decompositions:

- *column-based subsets*: approximate $\mathbf{X}_{(n)}$ by a subset of its columns, e.g. randomized sampling tucker CUR [80]
- *row-based subsets*: approximate $\mathbf{X}_{(n)}$ by a subset of its rows, e.g. Chidori CUR [79], [82], Fiber CUR [81], [82], and randomized-block HOSVD (RB-HOSVD) [101]

We briefly overview and contrast these two strategies in the following subsections.

A. COLUMN-BASED SUBSET METHODS FOR TENSOR UNFOLDINGS

Column-based subset methods approximate a tensor unfolding $\mathbf{X}_{(n)}$ using a subset of its columns. These columns are referred to as "fibers" in the tensor literature, and represent a fixed index in all modes of the tensor except for the nth mode. As an example, $(\mathbf{X}_{(1)})_{(:,z)}$ is a fiber of the first mode which represents $(\mathcal{X})_{(:,i_2,i_3,\dots,i_N)}$ for some indices of the $N-1$ other modes $i_2, i_3, \dots, i_N$ that correspond to some fiber index z.

For the nth mode unfolding $\mathbf{X}_{(n)}$ of a tensor $\mathcal{X}$, if we denote $\mathcal{I}_n$ as an index set for some subset of $\hat{R}_n$ columns, and denote $(\mathbf{X}_{(n)})_{\mathcal{I}_n} \triangleq (\mathbf{X}_{(n)})_{(:,\mathcal{I}_n)} \in \mathbb{R}^{D_n \times \hat{R}_n}$ as the matrix formed by these $\hat{R}_n$ columns, then (2) is restated as:

$$\begin{aligned} \mathcal{J}(\mathcal{I}_n) &= \left\| \mathbf{X}_{(n)} - \mathbf{P}_{(\mathbf{X}_{(n)})_{\mathcal{I}_n}} \mathbf{X}_{(n)} \right\|_F^2 \quad (5) \\ &= \left\| \mathbf{X}_{(n)} - (\mathbf{X}_{(n)})_{\mathcal{I}_n} \mathbf{M}_{\mathcal{I}_n} \right\|_F^2 \quad (6) \end{aligned}$$

where $\mathbf{P}_{(\mathbf{X}_{(n)})_{\mathcal{I}_n}} = (\mathbf{X}_{(n)})_{\mathcal{I}_n} (\mathbf{X}_{(n)})_{\mathcal{I}_n}{}^\dagger \in \mathbb{R}^{D_n \times D_n}$ is the projection matrix corresponding to the column space of $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$, $(\mathbf{X}_{(n)})_{\mathcal{I}_n}{}^\dagger \in \mathbb{R}^{\hat{R}_n \times D_n}$ is the pseudoinverse of $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$, and $\mathbf{M}_{\mathcal{I}_n} = (\mathbf{X}_{(n)})_{\mathcal{I}_n}{}^\dagger \mathbf{X}_{(n)} \in \mathbb{R}^{\hat{R}_n \times \tilde{D}_n}$ is the matrix mapping columns of $\mathbf{X}_{(n)}$ onto $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$.

By denoting $\mathcal{I} = \{\mathcal{I}_1, \dots, \mathcal{I}_N\}$ as the set of all N mode's column index sets $\mathcal{I}_n$, for $n = 1, \dots, N$, then we can represent the resulting decomposition's cost in a manner similar to (1):

$$\mathcal{J}(\mathcal{I}) = \left\| \mathcal{X} - \mathcal{M} \times_1 (\mathbf{X}_{(1)})_{\mathcal{I}_1} \times_2 \dots \times_N (\mathbf{X}_{(N)})_{\mathcal{I}_N} \right\|_F^2 \quad (7)$$

where the core tensor is given by $\mathcal{M} = \mathcal{X} \times_1 (\mathbf{X}_{(1)})_{\mathcal{I}_1}{}^\dagger \times_2 \dots \times_N (\mathbf{X}_{(N)})_{\mathcal{I}_N}{}^\dagger \in \mathbb{R}^{\hat{R}_1 \times \dots \times \hat{R}_N}$, and the factor matrices are given by the column subsets $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$.

This decomposition in (7) was first introduced in [80], referred to as "ApproxTensorSVD" in that paper. Later publications such as [83] refer to the algorithm as randomized sampling tucker CUR (RST-CUR). This decomposition is perhaps the most direct generalization of the matrix CUR to the tensor domain, as the decomposition takes the exact form of the matrix CUR when $N = 2$. We refer to this decomposition as RST-CUR for the remainder of the paper.

The same advantages gained by matrix CUR for matrices carries over to RST-CUR for tensors, notably a low complexity way to approximate a tensor's HOSVD. Additionally, as the factor matrices $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$ are fibers of the full tensor $\mathcal{X}$, the factor matrices retain properties held by the original tensor, which can include sparsity, nonnegativity, etc.. These qualities in $\mathcal{X}$ being retained in factor matrices $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$ may aid with the interpretability of the decomposition.

A key difference between column-based subset methods and row-based subset methods over $\mathbf{X}_{(n)}$ is how differences in dimensions affect the subset selection process. As $\mathbf{X}_{(n)} \in \mathbb{R}^{D_n \times \tilde{D}_n}$ is in general a very wide matrix with $\tilde{D}_n \gg D_n$, the massive number of columns leads deterministic column subset selection methods to be intractable, as their complexities are typically in the order of $\mathcal{O}(\tilde{D}_n^2)$ or more. Furthermore, even randomized methods typically only use a uniform distribution for sampling the columns, e.g. with norm sampling it may also be intractable to calculate the norm of all $\tilde{D}_n$ columns of $\mathbf{X}_{(n)}$. This is a significant comparative disadvantage of the column-based methods such as RST-CUR, as uniform sampling of the columns may lead to significantly worse approximations for a given choice of $\hat{R}_n$. While column subset methods over $\mathbf{X}_{(n)}$ are expensive, on the other hand, row-based subset methods are typically tractable due to the much smaller number of rows D_n, as we discuss in the next subsection.

B. ROW-BASED SUBSET METHODS FOR TENSOR UNFOLDINGS

Row-based subset methods approximate a tensor unfolding $\mathbf{X}_{(n)}$ using a subset of its rows. Aside from more advanced sampling methods being tractable over the rows than the columns of $\mathbf{X}_{(n)}$, another advantage of row-based methods is the interpretability of their subsets. Because rows in $\mathbf{X}_{(n)}$ are simply the elements of the nth mode, rows of $\mathbf{X}_{(n)}$ are easier to interpret than the fiber columns of $\mathbf{X}_{(n)}$.

For the nth mode unfolding $\mathbf{X}_{(n)}$ of a tensor $\boldsymbol{\mathcal{X}}$, if we now denote $\mathcal{I}_n$ as an index set for some subset of $\hat{R}_n$ rows, and denote $(\mathbf{X}_{(n)})_{\mathcal{I}_n} \triangleq (\mathbf{X}_{(n)})_{(\mathcal{I}_n,:)} \in \mathbb{R}^{\hat{R}_n \times \tilde{D}_n}$ as the matrix formed by these $\hat{R}_n$ rows, then (2) is restated as:

$$\mathcal{J}(\mathcal{I}_n) = \left\| \mathbf{X}_{(n)} - \mathbf{X}_{(n)} \mathbf{P}_{(\mathbf{X}_{(n)})_{\mathcal{I}_n}} \right\|_F^2 \quad (8)$$

$$= \| \mathbf{X}_{(n)} - \mathbf{M}_{\mathcal{I}_n} (\mathbf{X}_{(n)})_{\mathcal{I}_n} \|_F^2 \quad (9)$$

where $\mathbf{P}_{(\mathbf{X}_{(n)})_{\mathcal{I}_n}} \in \mathbb{R}^{T_n \times \tilde{D}_n}$ is the projection matrix corresponding for the row space of $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$, and $\mathbf{M}_{\mathcal{I}_n} \in \mathbb{R}^{D_n \times \hat{R}_n}$ is the nth mode's mapping matrix, which maps rows of $\mathbf{X}$ onto the span of $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$, and is given by:

$$\mathbf{M}_{\mathcal{I}_n} = \mathbf{X}_{(n)} (\mathbf{X}_{(n)})_{\mathcal{I}_n}^\top ((\mathbf{X}_{(n)})_{\mathcal{I}_n} (\mathbf{X}_{(n)})_{\mathcal{I}_n}^\top)^{-1} \quad (10)$$

By denoting $\mathcal{I} = \{\mathcal{I}_1, \ldots, \mathcal{I}_N\}$ as the set of all N modes' row index sets $\mathcal{I}_n$, for $n = 1, \ldots, N$, then we can represent the resulting decomposition's cost in a form similar to (1):

$$\mathcal{J}(\mathcal{I}) = \| \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}_{\mathcal{I}} \times_1 \mathbf{M}_{\mathcal{I}_1} \times_2 \ldots \times_N \mathbf{M}_{\mathcal{I}_N} \|_F^2 \quad (11)$$

where the core tensor $\boldsymbol{\mathcal{X}}_{\mathcal{I}} = (\boldsymbol{\mathcal{X}})_{(\mathcal{I}_1, \ldots, \mathcal{I}_N)} \in \mathbb{R}^{\hat{R}_1 \times \ldots \times \hat{R}_N}$ is a subtensor of $\boldsymbol{\mathcal{X}}$ over the index sets $\mathcal{I}_n$, and the factor matrices are the N mapping matrices $\mathbf{M}_{\mathcal{I}_n}$, for $n = 1, \ldots, N$.

The characteristic difference between the decomposition $\boldsymbol{\mathcal{X}}_{\mathcal{I}} \times_1 \mathbf{M}_{\mathcal{I}_1} \times_2 \ldots \times_N \mathbf{M}_{\mathcal{I}_N}$ in (11), and the decomposition $\boldsymbol{\mathcal{M}} \times_1 (\mathbf{X}_{(1)})_{\mathcal{I}_1} \times_2 \ldots \times_N (\mathbf{X}_{(N)})_{\mathcal{I}_N}$ in (7), is how elements of the tensor $\boldsymbol{\mathcal{X}}$ manifest as elements in the decomposition, relative to a tensor generalization of (3). In (7), elements of $\boldsymbol{\mathcal{X}}$ manifest as fibers in the *factor matrices* $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$, and the core tensor $\boldsymbol{\mathcal{M}}$ can be considered a tensor generalization of the *mapping matrix*. Where in (11), the opposite occurs: elements of $\boldsymbol{\mathcal{X}}$ manifest as the *core tensor* $\boldsymbol{\mathcal{X}}_{\mathcal{I}}$, and the *factor matrices* $\mathbf{M}_{\mathcal{I}_n}$ are the N modes' mapping matrices. Thus with (11), the core tensor is the element of the decomposition that retains properties of the original tensor, which may yield more useful decompositions depending on the application.

Various tensor decompositions take the form of the decomposition $\boldsymbol{\mathcal{X}}_{\mathcal{I}} \times_1 \mathbf{M}_{\mathcal{I}_1} \times_2 \ldots \times_N \mathbf{M}_{\mathcal{I}_N}$ in (11). This decomposition was first introduced in [79] shortly before the introduction of the RST-CUR decomposition. Later works such as [82] have provided significant understandings to the error guarantees of this decomposition, and have referred to it by the name "Chidori CUR" decomposition.

A key feature of the Chidori CUR is that the subset indices $\mathcal{I}_n$ are chosen prior to the decomposition, and that the mapping matrices $\mathbf{M}_{\mathcal{I}_n}$ are calculated only over those fibers of $\boldsymbol{\mathcal{X}}_{(n)}$ that correspond to the subset indices $\mathcal{I}_n$ of all $N-1$ other modes. In other words, in calculation of $\mathbf{M}_{\mathcal{I}_n}$ in (10), the matrix $\mathbf{X}_{(n)}$ is the unfolding of the nth mode "Chodiri Beam" $(\boldsymbol{\mathcal{X}})_{(\mathcal{I}_1, \ldots, \mathcal{I}_{n-1}, :, \mathcal{I}_{n+1}, \ldots, \mathcal{I}_N)} \in \mathbb{R}^{\hat{R}_1 \times \ldots \times \hat{R}_{n-1} \times D_1 \times \hat{R}_{n+1} \times \ldots \times \hat{R}_n}$, and $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$ is the unfolding of the core tensor $\boldsymbol{\mathcal{X}}_{\mathcal{I}}$ (a subtensor of the nth Chidori Beam). Because the $\mathbf{M}_{\mathcal{I}_n}$ are calculated over only the Chidori Beams, the decomposition only requires access to the Chidori beams and is thus independent from all other elements in the tensor. This reliance on only a small subset of the tensor to perform the decomposition results in one of the most computationally efficient tensor decompositions. At the same time, however, independence of the decomposition from elements outside the Chidori beams may result in a worse factorization than other decompositions, particularly when the subsets of the core tensor $\boldsymbol{\mathcal{X}}_{\mathcal{I}}$ are not well-representative of the rest of $\boldsymbol{\mathcal{X}}$, or if $\boldsymbol{\mathcal{X}}$ is otherwise heavily heterogeneous in nature.

A similar decomposition was later introduced in [81], and can be considered a generalization of the Chidori CUR where the unfolding fibers in $\mathbf{X}_{(n)}$ and $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$ are not restricted to those $\mathcal{I}_n$ of the $N-1$ other modes, but can be any random corresponding subset of fibers from $\mathbf{X}_{(n)}$ and $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$ over the entire tensor. This decomposition was also later studied in [82] and has been called the "Fiber CUR" decomposition. As the Fiber CUR allows access to any random subset of fibers of $\mathbf{X}_{(n)}$ and $(\mathbf{X}_{(n)})_{\mathcal{I}_n}$ for calculating mapping matrices $\mathbf{M}_{\mathcal{I}_n}$, its decomposition may be more robust to poorly chosen subsets of the data. However, as column fibers in Fiber CUR are typically uniformly sampled, this may also lead the Fiber CUR to exhibit considerably higher variation in the quality of the estimated $\mathbf{M}_{\mathcal{I}_n}$, which often leads to worse decompositions than those provided by Chidori CUR.

As described in Section III.A, the massive numbers of columns in $\mathbf{X}_{(n)}$ make column selection methods intractable, and thus typically only rely on uniform sampling to select the columns. However for row-based methods such as Chidori CUR and Fiber CUR, the much smaller number of rows $D_n \ll \tilde{D}_n$ allow for more sophisticated sampling methods such as norm sampling. When norm sampling is applied, index sets $\mathcal{I}_n$ are selected according to the norms of elements in the original tensor, e.g. $(\mathbf{X}_{(n)})_{(i,:)}$ which when vectorized is of dimension $\tilde{D}_n$. These sampling schemes require N passes over the tensor to construct the N index sets, and thus can still be of considerable expense. Perhaps as a result of this, row-based subset methods for tensors have exclusively used random subset methods such as uniform and norm sampling to obtain subsets of the tensor, and thus deterministic subset methods have not been explored.

Building on the ideas presented in previous sections, in the next section we introduce a new way of performing a subset-based Tucker decomposition that provides a good balance between efficiency and approximation quality, by exploiting weighted subsets of the data called *coresets*.

IV. TENSOR CORESET DECOMPOSITION

In this section, we introduce a method for the Tucker decomposition that operates by selecting *coresets*: weighted subsets of the data. As we later explain, these weighted subsets can provide a better approximation to the tensor $\boldsymbol{\mathcal{X}}$ by effectively better approximating the HOSVD's principal component tensor $\boldsymbol{\mathcal{G}}$. We later motivate additional differences vs. the previously discussed methods, such as a sequentially truncated coresets approach analogous to ST-HOSVD, and the ability to represent symmetry in the tensor over multiple modes. Furthermore, instead of exclusively selecting subsets randomly for greater efficiency, we also motivate ability to select subsets deterministically, for better approximation quality and for feature selection.

A. SUBSET DISCREPANCY – A MEASURE OF "REPRESENTATIVENESS"

To motivate weighted subsets within a tensor, we first refer back to the per-mode approximation error provided for row-based subsets in (9).

$$\mathcal{J}\left(\mathcal{I}_n\right) \;=\; \left\| \, \mathbf{X}_{(n)} - \mathbf{M}_{\mathcal{I}_n}\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n} \right\|_{\mathrm{F}}^2$$

Denoting the SVD of $\mathbf{X}_{(n)}$ by $\mathbf{X}_{(n)} = \mathbf{U}_{\mathbf{X}_{(n)}}\,\Sigma_{\mathbf{X}_{(n)}}\,\mathbf{V}^{\top}_{\mathbf{X}_{(n)}}$, the approximation error of $\mathbf{M}_{\mathcal{I}_n}\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n} \in \mathbb{R}^{D_n \times \tilde{D}_n}$ depends on how well the subset $\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n}$ can approximate the row-space of $\mathbf{X}_{(n)}$, specifically in terms of approximating its right principal components $\Sigma_{\mathbf{X}_{(n)}}\,\mathbf{V}^{\top}_{\mathbf{X}_{(n)}}$, which in the tensor domain is represented by the HOSVD core tensor $\boldsymbol{\mathcal{G}}$. These PCs are analytically given by the eigenvectors $\mathbf{V}_{\mathbf{X}_{(n)}}$ and corresponding eigenvalues $\Sigma_{\mathbf{X}_{(n)}}$ of the quadratic form $\mathbf{X}_{(n)}^{\top}\mathbf{X}_{(n)} \in \mathbb{R}^{T_n \times \tilde{D}_n}$, and thus can be approximated from $\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n}$ via the corresponding form $\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n}^{\top}\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n} \in \mathbb{R}^{T_n \times \tilde{D}_n}$. As a result, an implicit distance between the PCs of $\mathbf{X}_{(n)}$ and $\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n}$ is given by the distance between the quadratic forms:

$$\mathcal{R}\left(\mathcal{I}_n\right) \;=\; \left\| \left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_s}^{\top}\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_s} - \mathbf{X}_{(n)}^{\top}\mathbf{X}_{(n)} \right\|_{\mathrm{F}}^2 \qquad (12)$$

This can be understood as a nonparametric measure of *discrepancy* [90]–[95] between the full set $\mathbf{X}_{(n)}$ and the subset $\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n}$, analogous to the maximum mean discrepancy [92]–[94] for a particular realization of distributional "embeddings" of the elements in the set. Specifically for some ith element in the nth node, given by $\left(\mathbf{X}_{(n)}\right)_{(i,:)}$, its corresponding embedding in this discrepancy is given by $\left(\mathbf{X}_{(n)}\right)_{(i,:)}^{\top}\left(\mathbf{X}_{(n)}\right)_{(i,:)} \in \mathbb{R}^{\tilde{D}_n \times \tilde{D}_n}$, and the nth mode's "full mode embedding" $\mathbf{X}_{(n)}^{\top}\mathbf{X}_{(n)}$ is given by the sum $\mathbf{X}_{(n)}^{\top}\mathbf{X}_{(n)} = \sum_{i=1}^{D_n}\left(\mathbf{X}_{(n)}\right)_{(i,:)}^{\top}\left(\mathbf{X}_{(n)}\right)_{(i,:)}$, which we seek to best approximate via the subset's embedding $\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n}^{\top}\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n}$.

B. CORESETS – WEIGHTED SUBSETS

A subset's discrepancy can be further decreased by *weighting* the subset: assigning individual weights $w_n^{[i]}$ to each ith element in the subset. Utilizing these weighted subsets, called *coresets*, the discrepancy measure is given by:

$$\mathcal{R}\left(\mathcal{I}_n, \mathbf{w}_n\right) \;=\; \left\| \sum_{i \in \mathcal{I}_n} w_n^{[i]}\left(\mathbf{X}_{(n)}\right)_{(i,:)}^{\top}\left(\mathbf{X}_{(n)}\right)_{(i,:)} - \mathbf{B}_n \right\|_{\mathrm{F}}^2 \qquad (13)$$

where $\mathbf{w}_n = \left[w_n^{[1]}, \ldots, w_n^{[\hat{R}_n]}\right] \in \mathbb{R}^{\hat{R}_n}$ is the set of $\hat{R}_n$ coreset weights corresponding to each element in $\left(\mathbf{X}_{(n)}\right)_{\mathcal{I}_n} \in \mathbb{R}^{\hat{R}_n \times \tilde{D}_n}$, and $\mathbf{B}_n \triangleq \mathbf{X}_{(n)}^{\top}\mathbf{X}_{(n)} \in \mathbb{R}^{\tilde{D}_n \times \tilde{D}_n}$ is the nth modes full mode embedding (a fixed quantity).

An important point to acknowledge here is that in (13), the weights $w_n^{[i]}$ are applied to the element *embeddings* $\left(\mathbf{X}_{(n)}\right)_{(i,:)}^{\top}\left(\mathbf{X}_{(n)}\right)_{(i,:)}$, and not the elements themselves $\left(\mathbf{X}_{(n)}\right)_{(i,:)}$. Instead, it follows that the elements receive the square root of the weights $(w_n^{[i]})^{\frac{1}{2}}$:

$$w_n^{[i]}\left(\left(\mathbf{X}_{(n)}\right)_{(i,:)}^{\top}\left(\mathbf{X}_{(n)}\right)_{(i,:)}\right) = \left((w_n^{[i]})^{\frac{1}{2}}\left(\mathbf{X}_{(n)}\right)_{(i,:)}\right)^{\top}\left((w_n^{[i]})^{\frac{1}{2}}\left(\mathbf{X}_{(n)}\right)_{(i,:)}\right)$$

This necessitates us to later specify nonnegative weights $w_n^{[i]} \geq 0$ in order for $(w_n^{[i]})^{\frac{1}{2}}$ to be real.

We now discuss the procedure for selecting weights $\mathbf{w}_n$ that minimize the discrepancy (13). For simplicity, for now we assume that we have a particular realization of the subset $\mathcal{I}_n$, which is selected either randomly or deterministically (as we explain in the next subsection). With $\mathcal{I}_n$ fixed and discrepancy as only a function of the weights $\mathbf{w}_n$, we can equivalently write (13) in a form where all $\tilde{D}_n \times \tilde{D}_n$ matrices are instead given as $\tilde{D}_n^2 \times 1$ vectors:

$$\begin{aligned}\mathcal{R}\left(\mathbf{w}_n\right) \;&=\; \left\| \sum_{i \in \mathcal{I}_n} w_n^{[i]}\mathbf{a}_n^{[i]} - \mathbf{b}_n \right\|_2^2 \\ &=\; \left\| \, \mathbf{A}_n\mathbf{w}_n - \mathbf{b}_n \right\|_2^2\end{aligned} \qquad (14)$$

where we define $\mathbf{a}_n^{[i]} = \mathrm{vec}(\left(\mathbf{X}_{(n)}\right)_{(i,:)}^{\top}\left(\mathbf{X}_{(n)}\right)_{(i,:)}) \in \mathbb{R}^{T^2}$ as the vectorization of the ith element's embedding, $\mathbf{A}_n = \left[\mathbf{a}_n^{[1]}, \ldots, \mathbf{a}_n^{[\hat{R}_n]}\right] \in \mathbb{R}^{\tilde{D}_n^2 \times \hat{R}_n}$ as the horizontal concatenation of the $\mathbf{a}_n^{[i]}$, and $\mathbf{b}_n = \mathrm{vec}(\mathbf{B}_n) \in \mathbb{R}^{\tilde{D}_n^2}$ as the vectorization of the full mode embedding.

This is a least squares problem $\underset{\mathbf{w}_n}{\arg\min}\|\mathbf{A}_n\mathbf{w}_n - \mathbf{b}_n\|_2^2$ for which the ordinary least squares (OLS) solution is $\mathbf{w}_n = (\mathbf{A}_n^{\top}\mathbf{A}_n)^{-1}\mathbf{A}_n^{\top}\mathbf{b}_n$. However as noted previously, we also require that the weights $w_n^{[i]}$ be nonnegative in order for the square root of weights (applied to the elements themselves) to be real. Therefore, we use a NNLS algorithm [110] to solve for $\mathbf{w}_n$. This is an efficient algorithm that does not explicitly form the $\tilde{D}_n \times \tilde{D}_n$ embeddings, instead only requiring the kernels between embeddings which are significantly easier to calculate. We define the kernel between the embeddings of $\left(\mathbf{X}_{(n)}\right)_{(i,:)}$ and $\left(\mathbf{X}_{(n)}\right)_{(j,:)}$ as:

$$\mathrm{k}(i_n, j_n) = < \mathbf{a}_n^{[i]}, \mathbf{a}_n^{[j]} > \;=\; \left(\left(\mathbf{X}_{(n)}\right)_{(i,:)}\left(\mathbf{X}_{(n)}\right)_{(j,:)}^{\top}\right)^2 \qquad (15)$$

The two quantities required by the algorithm are kernels $(\mathbf{A}_n^\top \mathbf{A}_n) \in \mathbb{R}^{\hat{R}_n \times \hat{R}_n}$ and $\mathbf{A}_n^\top \mathbf{b}_n \in \mathbb{R}^{\hat{R}_n}$. The matrix $(\mathbf{A}_n^\top \mathbf{A}_n)$ provides all pairwise kernels within the $\mathcal{I}_n$ subset, and is equal to $\left((\mathbf{X}_{(n)})_{(\mathcal{I}_n,:)}(\mathbf{X}_{(n)})_{(\mathcal{I}_n,:)}^\top\right)^{\circ 2} \in \mathbb{R}^{\hat{R}_n \times \hat{R}_n}$, where $(.)^{\circ 2}$ denotes the Hadamard power (here, elementwise squaring). The vector $\mathbf{A}_n^\top \mathbf{b}_n$ provides kernels with each element in the subset with the full mode embedding, equal to $\left((\mathbf{X}_{(n)})_{(\mathcal{I}_n,:)}\mathbf{X}_{(n)}^\top\right)^{\circ 2}\mathbf{1}$ $\in \mathbb{R}^{\hat{R}_n}$, where $\mathbf{1} \in \mathbb{R}^{D_n}$ is the vector of 1s. For further efficiency, we initialize the NNLS algorithm with the mapping of the OLS solution $(\mathbf{A}_n^\top \mathbf{A}_n)^{-1}\mathbf{A}_n^\top \mathbf{b}_n$ to its nearest nonnegative vector. In our experience, we often observe that the OLS solution is already nonnegative and thus exactly minimizes (14) without requiring the NNLS algorithm.

Having provided the means to optimize the weights $\mathbf{w}_n$, in the next section we discuss ways of selecting the subsets.

C. SUBSET SELECTION – RANDOMIZED OR DETERMINISTIC

As our method is a row-based subset method over $\mathbf{X}_{(n)}$, we can consider more advanced means of selecting subsets $(\mathbf{X}_{(n)})_{(\mathcal{I}_n,:)}$ than uniform sampling of rows. We use different strategies if we seek random subsets, prioritizing computational efficiency over approximation quality, or deterministic subsets, prioritizing approximation quality in addition to the utility of feature selection.

For random subsets, we use norm sampling as done with previously mentioned methods. While we unfortunately do not provide an approximation bound for random subsets using the NNLS weights discussed previously, intuitively these weights should yield a discrepancy that is less than or equal to that provided by the normalized weights discussed in (4), as those weights are not explicitly optimizing over the discrepancy whereas the NNLS weights are. Therefore, we expect an error superior or equal to that of (4)'s weights. As we show in the next section, it is inexpensive to calculate the NNLS weights since the kernel quantities are required anyways to calculate the nth mode's mapping matrix $\mathbf{M}_{\mathcal{I}_n}$. Selection of the random subset along with calculating the weights has complexity of $\mathcal{O}((\hat{R}_n+1)\tilde{D}_n D_n + \hat{R}_n^3)$, which is linear in D_n.

For deterministic subsets, we retain the use of greedy methods in the interest of balancing approximation quality with computational efficiency. As discussed in Section II.C, greedy methods significantly outperform the error bounds of randomized methods and lead to subsets that rapidly converge to the properties of the full set. We specifically utilize the weighted kernel herding (WKH) method [95] which allows us to simultaneously and efficiently solve for the subset indices $\mathcal{I}_n$ and weights $\mathbf{w}_n$. Like the NNLS algorithm, the WKH algorithm is made more efficient by only requiring kernels to operate. It uses $\mathbf{X}_{(n)}\ \mathbf{X}_{(n)}^\top \in \mathbb{R}^{D_n \times D_n}$, the matrix of pairwise kernels between all elements in the nth mode, and has complexity $\mathcal{O}(\tilde{D}_n D_n^2 + \hat{R}_n^3 D_n)$, which is quadratic in D_n.

In the next section, we introduce our tensor decomposition method as a sequentially truncated variation of the row-based subset model in (11), where we sequentially replace the tensor with a coreset of itself.

D. TENSOR DECOMPOSITION VIA SEQUENTIALLY TRUNCATED CORESETS

We now motivate our method for performing a coreset-based Tensor decomposition. We first revisit points mentioned in Section III.B. specifically discussing the advantages and disadvantages of the Chidori CUR decomposition. As we note previously, the Chidori CUR Decomposition is efficient because it only requires processing small subsets of the tensor – the "Chidori beams" – in order to calculate the mode's mapping matrices $\mathbf{M}_{\mathcal{I}_n}$. However, this may also lead to a significantly worse approximation quality for the decomposition, in the event that the randomly chosen subsets are not well representative of the entire tensor $\mathbf{X}$, or otherwise for decomposing tensors that are highly heterogeneous in nature. When approximation quality is a priority for both randomized and deterministic methods, it may be more prudent to use a method that does pass over all elements of the tensor, but preferably only once if computational efficiency is also a priority. These decompositions can provide significantly more representative subsets of the data while still maintaining excellent computational efficiency.

With this focus in mind, in order to provide a good balance between approximation error and efficiency, we instead consider a method inspired by ST-HOSVD that utilizes *sequentially truncated coresets* to perform the decomposition. Like ST-HOSVD, for each nth mode of the tensor, we would learn the mapping matrix $\mathbf{M}_{\mathcal{I}_n}$ and then replace the tensor with a truncated tensor, thus significantly decreasing the complexity of calculating $\mathbf{M}_{\mathcal{I}_n}$ for all remaining modes. However, whereas ST-HOSVD replaces the tensor with the PCs of the mode, we instead replace the tensor with the nth mode's coreset. These methods are closely connected by the fact that the coresets are trying to best preserve the PCs of the tensor, as evidence by the discrepancy cost in (12) and (13).

We now discuss details of our method's implementation to assist understanding the pseudocode provided in Algorithm 4. The factorization of the nth mode is initialized by selecting a subset $\mathcal{I}_n$, which as we mentioned in Section VI.C is in general $\mathcal{O}(D_n)$ for random subsets or $\mathcal{O}(D_n^2)$ for deterministic subsets. We then compute the pairwise inner products between the subset and the full set, given by the matrix $\mathbf{P}_{\mathcal{I}_n}$ $= (\mathbf{X}_{(n)})_{\mathcal{I}_n}\ \mathbf{X}_{(n)}^\top \in \mathbb{R}^{\hat{R}_n \times D_n}$, within which the submatrix $\mathbf{P}_{(\mathcal{I}_n,\mathcal{I}_n)} = (\mathbf{P}_{\mathcal{I}_n})_{(:,\mathcal{I}_n)} \in \mathbb{R}^{\hat{R}_n \times \hat{R}_n}$ provides the pairwise inner products within the subset. With both of these matrices, we can obtain the kernels of embeddings $\mathbf{P}_{(\mathcal{I}_n,\mathcal{I}_n)}^{\circ 2} \in \mathbb{R}^{\hat{R}_n \times \hat{R}_n}$ and $\mathbf{P}_{\mathcal{I}_n}^{\circ 2}\ \mathbf{1}_{D_n} \in \mathbb{R}^{\hat{R}_n}$ to perform NNLS and learn the coreset weights $\mathbf{w}_n \in \mathbb{R}^{\hat{R}_n}$. Arranging $\mathbf{w}_n$ in the diagonal matrix $\mathbf{W}_n$ $\in \mathbb{R}^{\hat{R}_n \times \hat{R}_n}$, the approximation in (9) given by $\mathbf{M}_{\mathcal{I}_n}\ (\mathbf{X}_{(n)})_{\mathcal{I}_n}$ can be weighted via $\mathbf{M}_{\mathcal{I}_n}\ \mathbf{W}_n^{-1}\ \mathbf{W}_n\ (\mathbf{X}_{(n)})_{\mathcal{I}_n}$, in which case the mapping matrix (accounting for weights) is given by $\mathbf{M}_{\mathcal{I}_n}$ $= \mathbf{P}_{\mathcal{I}_n}^\top \mathbf{P}_{(\mathcal{I}_n,\mathcal{I}_n)}^{-1}\ \mathbf{W}_n^{-1} \in \mathbb{R}^{D_n \times \hat{R}_n}$, and the weighted coreset is

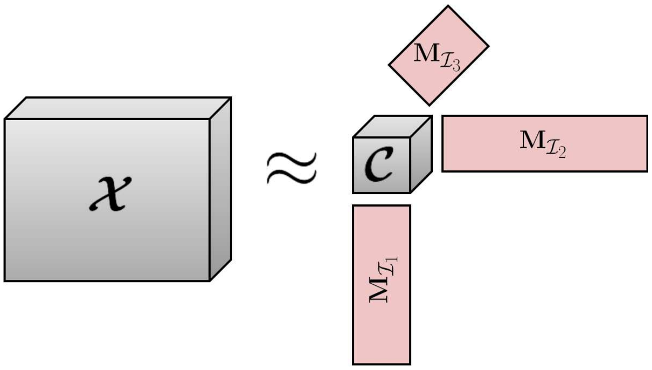

$\mathcal{X}$
$\approx$
$\mathcal{C}$
$\mathrm{M}_{\mathcal{I}_1}$
$\mathrm{M}_{\mathcal{I}_2}$
$\mathrm{M}_{\mathcal{I}_3}$

Fig. 1. Visualization of the tensor coreset decomposition (TCD) applied to a 3rd-order tensor $\mathcal{X}$. The core tensor $\mathcal{C}$ is a weighted subtensor (coreset) of $\mathcal{X}$.

Algorithm 4: **Tensor Coreset Decomposition (TCD)**

```
Input: $\mathcal{X} \in \mathbb{R}^{D_1 \times \ldots \times D_N}$ ($N$-mode tensor),
       $[\hat{R}_1, \ldots, \hat{R}_N]$ (subset sizes per mode)
Output: $\mathcal{X} \approx \mathcal{C} \times_1 \mathbf{M}_{\mathcal{I}_1} \times \ldots \times_N \mathbf{M}_{\mathcal{I}_N}$ , where
       $\mathcal{C} \in \mathbb{R}^{\hat{R}_1 \times \ldots \times \hat{R}_N}$ (coreset tensor),
       $\{\mathbf{M}_{\mathcal{I}_1}, \ldots, \mathbf{M}_{\mathcal{I}_N}\}$ (mapping matrices per mode
                                                     – like factor matrices),
       $\mathcal{I} = \{\mathcal{I}_1, \ldots, \mathcal{I}_N\}$ (subset index sets per mode)

  for each mode $n = 1 : N$
    unfold (matricize) tensor w.r.t . $n$th mode
      $\mathcal{X} \to \mathbf{X}_{(n)} \in \mathbb{R}^{D_n \times \tilde{D}_n}$, with $\tilde{D}_n = \prod_{\substack{m=1 \\ m \neq n}}^{N} D_m$
    select subset indices $\mathcal{I}_n$ for some $\hat{R}_n$ rows of $\mathbf{X}_{(n)}$,
      either randomly (using e.g. norm sampling),
      or deterministically (using e.g. greedy WKH).
    compute inner products of full mode with subset:
      $\mathbf{P}_{\mathcal{I}_n} = (\mathbf{X}_{(n)})_{\mathcal{I}_n} \, \mathbf{X}_{(n)}^{\top} \in \mathbb{R}^{\hat{R}_n \times D_n}$,
      also within $\mathbf{P}_{\mathcal{I}_n}$ the submatrix $\mathbf{P}_{(\mathcal{I}_n, \mathcal{I}_n)} \in \mathbb{R}^{\hat{R}_n \times \hat{R}_n}$
    using the kernels $\mathbf{P}_{(\mathcal{I}_n, \mathcal{I}_n)}^{\circ 2} \in \mathbb{R}^{\hat{R}_n \times \hat{R}_n}$
      and $\mathbf{P}_{\mathcal{I}_n}^{\circ 2} \, \mathbf{1}_{D_n} \in \mathbb{R}^{\hat{R}_n}$, perform kernel NNLS
      to learn coreset weights $\mathbf{w} \in \mathbb{R}^{\hat{R}_n}$.
    arrange weights into diagonal matrix $\mathbf{W}_n = \mathrm{diag}(\mathbf{w})$
    compute and store $n$th mode's mapping matrix:
      $\mathbf{M}_{\mathcal{I}_n} = \mathbf{P}_{\mathcal{I}_n}^{\top} \mathbf{P}_{(\mathcal{I}_n, \mathcal{I}_n)}^{-1} \, \mathbf{W}_n^{-1} \in \mathbb{R}^{D_n \times \hat{R}_n}$
    replace unfolded tensor with weighted coreset
      $\mathbf{X}_{(n)} \to \mathbf{W}_n \, (\mathbf{X}_{(n)})_{\mathcal{I}_n} \in \mathbb{R}^{\hat{R}_n \times \tilde{D}_n}$
    replace $n$th dimension $D_n \to \hat{R}_n$
    un-matricize the tensor $\mathbf{X}_{(n)} \to \mathcal{X}$
  end for
  $\mathcal{C} \to \mathcal{X}$
```

given by $\mathbf{W}_n \, (\mathbf{X}_{(n)})_{\mathcal{I}_n}$. Thus with the weights calculated, we compute the mapping matrix $\mathbf{M}_{\mathcal{I}_n}$, then truncate the nth mode by replacing it with the coreset $\mathbf{W}_n \, (\mathbf{X}_{(n)})_{\mathcal{I}_n}$, and finally un-matricize the tensor so that the entire process can be repeated for the remaining modes. Fig. 1 visualizes the tensor coreset decomposition (TCD).

After truncating over all modes, the resulting coreset core tensor $\mathcal{C} = \mathcal{X}_{\mathcal{I}} \times_1 \mathbf{W}_1 \times_2 \ldots \times_N \mathbf{W}_N$ is a subtensor $\mathcal{X}_{\mathcal{I}}$ weighted on each nth mode by weight matrix $\mathbf{W}_n$, and serves as a compressed form of $\mathcal{X}$ analogous to the principal component tensor $\mathcal{G}$ from HOSVD. The weights $\mathbf{W}_n$ are a key differentiator from other methods like the Chidori CUR decomposition, and the use of $\mathbf{W}_n$ within this sequentially method can allow for an excellent approximation to the HOSVD core tensor $\mathcal{G}$, and by extension, the tensor $\mathcal{X}$.

The method described in Algorithm 4 assumes that $\mathcal{X}$ is an asymmetric tensor, and does not preserve symmetry in the decomposition if $\mathcal{X}$ is symmetric across several modes. To retain symmetry in the decomposition in the event that $\mathcal{X}$ is symmetric, we simply compute only one factor matrix $\mathbf{M}_{\mathcal{I}_n}$ for one of the symmetric modes n, and reuse $\mathbf{M}_{\mathcal{I}_n}$ for all other modes symmetric to n, while truncating those other modes the same way $\mathbf{X}_{(n)} \to \mathbf{W}_n \, (\mathbf{X}_{(n)})_{\mathcal{I}_n}$.

In the next subsection, we compare our so called Tensor Coreset Decomposition (TCD) to the Chidori CUR decomposition from a computational complexity standpoint.

E. COMPUTATIONAL COMPLEXITY OF TCD

In this section, we discuss the complexities of TCD with random (norm sampled) or deterministic (WKH) subsets, compared to Chidori CUR with random (norm sampled) subsets. We retain notations such as $\tilde{D}_n^{(n)} = (\prod_{m=1}^{n-1} \hat{R}_m)(\prod_{m=n+1}^{N} D_m)$ for sequentially truncated methods like ST-HOSVD and TCD. For simplicity, we assume that the modes truncated with these methods are in the order $n = 1, \ldots, N$.

We first discuss complexity of Chidori CUR decomposition with random (norm sampled) subsets. The majority of complexity is in calculation of the norms of elements across the N modes of the original tensor $\mathcal{X}$, each mode of complexity $\mathcal{O}(D_n \tilde{D}_n)$. These are then followed by the significantly cheaper calculations of the mapping matrices per each Chidori Beam, of complexity $\mathcal{O}(\hat{R}_n D_n \hat{T}_n + \hat{R}_n^3 + \hat{R}_n^2 D_n)$, where we denote $\hat{T}_n = \prod_{\substack{m=1 \\ m \neq n}}^{N} \hat{R}_m$. The total complexity is thus: $\mathcal{O}\big(\sum_{n=1}^{N} (D_n \tilde{D}_n + \hat{R}_n D_n \hat{T}_n + \hat{R}_n^3 + \hat{R}_n^2 D_n)\big)$.

We then consider the complexity of TCD with random (norm sampled) subsets, which we refer to as TCD-R. The majority of complexity from each nth mode's truncation occurs from the norm sampling of complexity $\mathcal{O}(D_n \tilde{D}_n^{(n)})$,

Table 2. computational complexities of TCD and similar methods described in Sections II and III. For truncated methods, we assume that the truncation order is $n = 1, \ldots, N$.

ST-HOSVD	$\mathcal{O}\Big(\sum_{n-1}^{N} \min\big(D_n^2 \tilde{D}_n^{(n)}, (\tilde{D}_n^{(n)})^2 D_n\big)\Big)$
Chidori CUR	$\mathcal{O}\Big(\sum_{n=1}^{N} \big(D_n \tilde{D}_n + \hat{R}_n D_n \hat{T}_n + \hat{R}_n^3 + \hat{R}_n^2 D_n\big)\Big)$
TCD-R	$\mathcal{O}\Big(\sum_{n=1}^{N} \big((\hat{R}_n + 1) D_n \tilde{D}_n^{(n)} + \hat{R}_n^3 + \hat{R}_n^2 D_n\big)\Big)$
TCD-D	$\mathcal{O}\Big(\sum_{n=1}^{N} \big(D_n^2 \tilde{D}_n^{(n)} + \hat{R}_n^3 D_n + \hat{R}_n^2 D_n\big)\Big)$

and the calculation of inner products between the subset and full set $\mathbf{P}_{\mathcal{I}_n}$ of complexity $\mathcal{O}(\hat{R}_n D_n \tilde{D}_n^{(n)})$. These are then used by the significantly cheaper calculations of the coreset weights of complexity $\mathcal{O}(\hat{R}_n^3)$, and calculation of the mapping matrices $\mathbf{M}_{\mathcal{I}_n}$ of complexity $\mathcal{O}(\hat{R}_n^2 D_n)$ (re-using $\mathbf{P}^{-1}_{(\mathcal{I}_n,\mathcal{I}_n)}$ from the coreset weights). The total complexity is thus: $\mathcal{O}\Big(\sum_{n=1}^{N}\big((\hat{R}_n+1)D_n\tilde{D}_n^{(n)}+\hat{R}_n^3+\hat{R}_n^2 D_n\big)\Big)$.

Lastly, we consider the complexity of TCD with deterministic (WKH) subsets, which we refer to as TCD-D. The majority of complexity from each nth mode's truncation occurs from requiring calculation of $\mathbf{P}_n = \mathbf{X}_{(n)}\,\mathbf{X}_{(n)}^\top \in \mathbb{R}^{D_n\times D_n}$, the pairwise inner products over the entire nth mode, of complexity $\mathcal{O}(D_n^2\tilde{D}_n^{(n)})$. This is followed by the WKH subset selection of complexity $\mathcal{O}(\hat{R}_n^3 D_n)$, which yields indices $\mathcal{I}_n$ and weights $\mathbf{w}_n$, along with the significantly cheaper calculations of the mapping matrices $\mathbf{M}_{\mathcal{I}_n}$ of complexity $\mathcal{O}(\hat{R}_n^2 D_n)$ (where we can also re-use $\mathbf{P}^{-1}_{(\mathcal{I}_n,\mathcal{I}_n)}$ from the WKH). The total complexity is thus: $\mathcal{O}\Big(\sum_{n=1}^{N}\big(D_n^2\tilde{D}_n^{(n)}+\hat{R}_n^3 D_n+\hat{R}_n^2 D_n\big)\Big)$.

Table 2 provides complexities of these methods. We note that for symmetric tensors, all methods are capable of exploiting symmetry by re-using factor matrices across symmetric modes, as described in Section IV.D. In this case, the summation $\mathcal{O}(\sum_{n=1}^{N}(.))$ is truncated to only the number of unique modes (i.e., symmetric modes are only counted once).

In the next section, we experimentally test the TCD methods vs other efficient Tucker decomposition methods for approximating the HOSVD. We first demonstrate performance of methods on simulated data under various generative conditions. Later, we demonstrate these methods on real fMRI data in the form of functional connectivity maps (FNCs).

V. NUMERICAL EXPERIMENTS

We first introduce the performance measures used to compare the Tensor decomposition methods. Denoting a method's approximated tensor by $\hat{\boldsymbol{\mathcal{X}}}$, the relative approximation error of $\hat{\boldsymbol{\mathcal{X}}}$ is given by:

$$\operatorname{err}(\hat{\boldsymbol{\mathcal{X}}}) = \frac{\left\|\boldsymbol{\mathcal{X}}-\hat{\boldsymbol{\mathcal{X}}}\right\|_{\mathrm{F}}}{\|\boldsymbol{\mathcal{X}}\|_{\mathrm{F}}} \in [0\ \infty)$$

As the methods discussed in this paper are often used to approximate the HOSVD or ST-HOSVD, we also use a measure of distance between factors of the ST-HOSVD and factors of a method's estimated HOSVD. We introduce this new measure as "HOSVD distance", and note that its formulation utilizes the inter-symbol-interference (ISI) [111] used to evaluate the performance of blind source separation methods. Defining HOSVD distance, we denote $\boldsymbol{\mathcal{A}} = \{\mathbf{A}_1,\ldots,\mathbf{A}_N\}$ as the N factor matrices for the "true" ST-HOSVD, and denote $\hat{\boldsymbol{\mathcal{A}}} = \left\{\hat{\mathbf{A}}_1,\ldots,\hat{\mathbf{A}}_N\right\}$ as a method's corresponding estimated HOSVD factor matrices, obtained by converting a method's factorization into a HOSVD via Algorithm 3. Then the HOSVD distance between a method's estimated HOSVD factors $\hat{\boldsymbol{\mathcal{A}}}$ and the true factors $\boldsymbol{\mathcal{A}}$ is given by:

$$\text{HOSVD distance }(\boldsymbol{\mathcal{A}},\hat{\boldsymbol{\mathcal{A}}}) = \sum_{n=1}^{N}\operatorname{ISI}(\mathbf{A}_n^\top\hat{\mathbf{A}}_n) \tag{16}$$

Where the ISI of a matrix $\mathbf{U}\in\mathbb{R}^{N\times N}$ measures how close the matrix $\mathbf{G}$ is to a permuted diagonal matrix (a performance measure invariant to sign and permutation ambiguities of the factors), and is given by:

$$\begin{aligned}\operatorname{ISI}(\mathbf{U}) = \frac{1}{2N(N-1)}\Bigg[&\sum_{n=1}^{N}\Big(\sum_{m=1}^{N}\frac{|(\mathbf{U})_{(n,m)}|}{\max_p(|(\mathbf{U})_{(n,p)}|)}-1\Big)\\ &+\sum_{m=1}^{N}\Big(\sum_{n=1}^{N}\frac{|(\mathbf{U})_{(n,m)}|}{\max_p(|(\mathbf{U})_{(p,n)}|)}-1\Big)\Bigg]\end{aligned} \tag{17}$$

Finally, we also measure the CPU-time of the methods. For all performance evaluations done in Sections V and VI, we use the computational resources provided by the UMBC High Performance Computing Facility (HPCF), thus CPU-time is reflective of HPCF's capabilities.

A. EXPERIMENTS WITH SIMULATED DATA

Our generative model of a tensor $\boldsymbol{\mathcal{X}}$ is as follows. For a common dimensionality across the modes D, we model a tensor $\boldsymbol{\mathcal{X}}\in\mathbb{R}^{D\times\cdots\times D}$ as the sum of a low-rank signal tensor $\boldsymbol{\mathcal{X}}_S\in\mathbb{R}^{D\times\cdots\times D}$ and a full-rank noise tensor $\boldsymbol{\mathcal{X}}_N\in\mathbb{R}^{D\times\cdots\times D}$:

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{X}}_S + \eta\frac{\|\boldsymbol{\mathcal{X}}_S\|_{\mathrm{F}}}{\|\boldsymbol{\mathcal{X}}_N\|_{\mathrm{F}}}\,\boldsymbol{\mathcal{X}}_N\ , \tag{18}$$

where η is the signal to noise ratio (SNR) of $\boldsymbol{\mathcal{X}}$.

The signal tensor $\boldsymbol{\mathcal{X}}_S$ is given in the form $\boldsymbol{\mathcal{X}}_S = \boldsymbol{\mathcal{G}}\times_1\mathbf{A}_1\times_2\ldots\times_N\mathbf{A}_N$, where $\boldsymbol{\mathcal{G}}\in\mathbb{R}^{R\times\cdots\times R}$ is a core tensor for some "true core size" R, and $\mathbf{A}_n\in\mathbb{R}^{D\times R}$ for $n=1,\ldots,N$ are the factor matrices. The core tensor $\boldsymbol{\mathcal{G}}$, factor matrices $\mathbf{A}_n$, and noise tensor $\boldsymbol{\mathcal{X}}_N$ are all randomly generated with entries sampled from the standard Gaussian distribution.

We consider two sets of simulated experiments: one where the generative model follows a CPD model, and one where the generative model follows a Tucker model (respectively referred to in our experiments as "CPD model data" or "Tucker model data"). These experiments use the same conditions described above except for generation of core tensor $\boldsymbol{\mathcal{G}}$: the Tucker experiments generate all entries of $\boldsymbol{\mathcal{G}}$ from the standard Gaussian distribution, whereas the CPD experiments specify $\boldsymbol{\mathcal{G}}$ as a superdiagonal core tensor wherein all $(\boldsymbol{\mathcal{G}})_{(i,\ldots,i)}$ for $i=1,\ldots,R$ are drawn from the standard Gaussian distribution, and all other entries of $\boldsymbol{\mathcal{G}}$ equal 0.

As our paper focuses on subset-based methods for a Tucker decomposition, particularly those that approximate the HOSVD, we limit our results to variations of these methods. We thus include ST-HOSVD [89], Chidori CUR [79], [82], RST-CUR [80], a random-projection variant of HOSVD called RP-HOSVD [C] [83], [112], and another row-based tensor decomposition like those discussed in Section III.B., called RB-HOSVD [101]. While we also discuss the Fiber CUR [81] in Section III.B, we do not include Fiber CUR in our experiments as we observed poor performance compared to the other algorithms.

All of the methods and experiments are coded in MATLAB. According to the tested methods' respective papers, we use norm sampling to obtain random row subsets of the tensor unfoldings $\mathbf{X}_{(n)}$ for Chidori CUR and RB-HOSVD, and we use uniform sampling to to obtain random column subsets of $\mathbf{X}_{(n)}$ for RST-CUR. Our implementation of ST-HOSVD is from the tensor toolbox version 3.6 [113], and all other methods are coded via details given in their respective papers.

To simplify the experiments, we perform all of these methods with a common "estimated core size" $\hat{R}$ that is shared across the modes of the estimated tensor. The decomposition is then converted to an estimated HOSVD decomposition that also uses the same $\hat{R}$ for all modes of the tensor. Therefore, the true HOSVD factor matrices are given by $\mathbf{A}_n \in \mathbb{R}^{D\times\hat{R}}$ and the estimated factor matrices can be given by $\hat{\mathbf{A}}_n \in \mathbb{R}^{D\times\hat{R}}$, for $n = 1, \ldots, N$, in which case the $\mathbf{U}$ matrices in (17) are of size $\hat{R}\times\hat{R}$. Note here that the true HOSVD is performed for some choice of estimated core size $\hat{R}$ that may differ from the true core size R of $\boldsymbol{\mathcal{X}}_S$.

Under the generative model defined in (18), we vary these qualities of the model to test the methods' performances: estimated core size $\hat{R}$, the true core sizes R, the dimensionality of the modes (mode size) D, the SNR of the simulated data tensor η, and the number of modes N. All of our experiments use these default parameters: $R = 4$, $\hat{R} = 4$, $\eta = 10$, $N = 3$, and $D = 200$. Given the memory requirements of extremely large tensors, in the experiment that varies the number of modes, we restrict N to be either 3 or 4 modes and we use a smaller default mode size of $D = 80$.

For all plots where we display CPU time performance, we note that these plots were essentially identical for the CPD and Tucker modeled data, thus performance was effectively independent of the generative model's core tensor structure. Therefore, we only show the plot for the CPD model data. Additionally, we do not show figures for CPU time vs. the true core size R or the SNR η, as these experiments feature CPU times that are constant with respect to these variables.

Fig. 2 plots the methods' CPU time performance with respect to the mode size D. In this experiment ST-HOSVD is the slowest of the methods, followed by Chidori CUR, RP-HOSVD, RST-CUR, TCD-D, and TCD-R. With the default estimated core size $\hat{R} = 4$, we note that TCD-D can maintain fast times in the event that $\hat{R}$ is small, which works well for tensors that have a reasonably low ranks. We also note that Chidori CUR's slower performance is mainly due to the norm sampling over the entire tensor for each nth mode, in contrast to sampling over truncated tensors such as done in other methods. Chidori CUR is significantly faster when uniform sampling is done in place of norm sampling, with an accompanying degree of loss in approximation performance.

Fig. 3 plots the methods' CPU time performance with respect to the estimated core size $\hat{R}$. TCD-D faces larger complexity with higher R, whereas all other methods have complexity that only increases slightly with increasing R. This may motivate other methods besides TCD-D for when CPU time is a priority and larger $\hat{R}$ are desired. However, TCD-D is still unique among these methods for deterministically selecting elements from the modes. Thus, compared to these otherwise predominately randomized methods, TCD-D is perhaps unique in its utility for feature selection.

Fig. 4 plots the methods' CPU time performance with respect to the number of modes N, for $N = 3$ and $N = 4$. TCD-D and TCD-R are among the fastest methods in this experiment, and interestingly, TCD-D is the fastest despite being deterministic. We observe that this is due to how MATLAB's efficiency varies with respect to different mathematical operations: MATLAB is especially efficient in computing the Gram matrix $\mathbf{X}_{(n)}\mathbf{X}_{(n)}^\top \in \mathbb{R}^{D\times D}$, so much so that it can actually be more efficient to compute $\mathbf{X}_{(n)}\mathbf{X}_{(n)}^\top$ than even the fastest methods for calculating norms of rows of $\mathbf{X}_{(n)}$, which is required of the norm sampling approaches like TCD-R, Chidori CUR, and RB-HOSVD. Depending on the efficiency of the calculations, the programming environment used and the dimensions of the tensor, these methods may benefit by

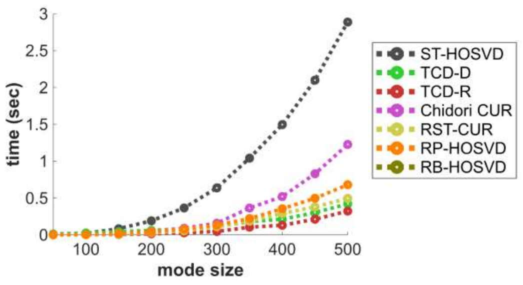


Fig. 2. CPU time w.r.t. mode size D.

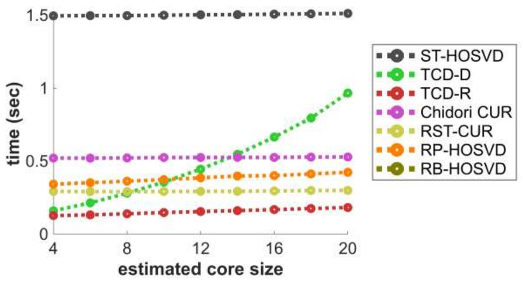


Fig. 3. CPU time w.r.t. estimated core size $\hat{R}$.

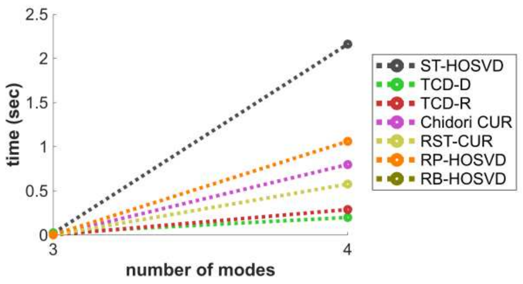


Fig. 4. CPU time w.r.t. number of modes N.

using $\mathbf{X}_{(n)}\mathbf{X}_{(n)}^\top$ to calculate the norms. At the same time, this also demonstrates the efficiency of the WKH procedure in TCD-D for smaller $\hat{R}$, since it does not lead to significant increases in complexity above the other methods.

Fig. 5 plots the methods' relative error performance with respect to the estimated core size $\hat{R}$. All methods' decompositions exponentially approach the true tensor in approximation quality with diminishing returns in $\hat{R}$. Performance of TCD-R in this experiment is comparable to Chidori CUR, with these methods only beaten by ST-HOSVD and TCD-D for lower $\hat{R}$.

Fig. 6 plots the methods' relative error performance with respect to the true core size R. Given a fixed estimated core size $\hat{R} = 4$, all decompositions perform worse as the true core size $\hat{R}$ increases, where with $R > 4$ the decompositions are effectively underparametrizing and/or undersampling their model of the tensor. Like in Fig. 5, in this experiment TCD-D has an estimation performance that is only slightly worse than ST-HOSVD. After these methods, TCD-R has the third best performance, exceeding that of Chidori CUR for larger $\hat{R}$.

Fig. 7 plots the methods' relative error performance with respect to the mode size D. These performances are mostly constant in D with the CPD data (left), but for the Tucker data, some methods like Chidori CUR and TCD-R feature slightly worse performances with larger D, up to diminishing returns. Most of the randomized methods have much more comparable relative errors for the CPD model data, with significantly higher spread with the Tucker model data.

Fig. 8 plots the methods' relative error performance with respect to the signal to noise ratio (SNR) η. Subject to diminishing returns, all methods perform significantly better in approximating the tensor with higher SNR, and TCD-D and TCD-R appear to provide some of the better approximations with lower SNR values. With higher SNR values, TCD-D's performance is comparable to ST-HOSVD and TCD-R's performance is comparable to Chidori CUR.

Fig. 9 plots the methods' relative error performance with respect to the number of modes N, for $N = 3$ and $N = 4$. An apparent disadvantage to Chidori CUR and TCD-R occurs when $N = 4$, in which case these methods' performances appear to suffer considerably, whereas all other methods are not as much affected by change of N.

We now discuss the methods' performances in terms of the HOSVD distance measure defined in (17). We note that we compare each algorithm's estimated HOSVD factors to the "true" factors estimated by ST-HOSVD for the same choice of $\hat{R}$, thus we don't include ST-HOSVD in these plots since it has a HOSVD distance of 0 with itself.

Fig. 10 plots the methods' HOSVD distances with respect to the estimated core size $\hat{R}$. All plots feature a clear U-shaped performance curve where the best performance generally occurs at $\hat{R} = 6$, slightly higher than the true core size $R = 4$. Interestingly, these U-shaped HOSVD distance vs. $\hat{R}$ plots are notably different in shape from the monotonically decreasing error vs. $\hat{R}$ plots in Fig. 5. While the relative error of the decompositions only decreases when the decompositions model allows for more complexity (via increasing $\hat{R}$), the HOSVD distance represents more of a measure of *parameter estimation*, where the desired parameters are the true ST-HOSVD factors, and are best estimated when the estimated number of factors $\hat{R}$ is close to the true number R.

Fig. 11 plots the methods' HOSVD distances with respect

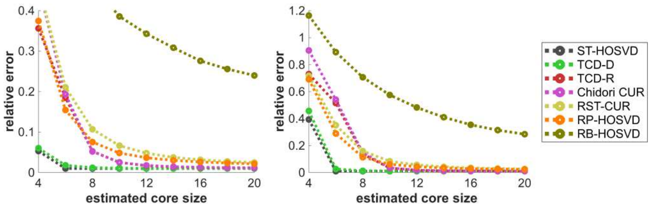


Fig. 5. Relative error w.r.t. the estimated core size $\hat{R}$.
Left: CPD model data. Right: Tucker model data.

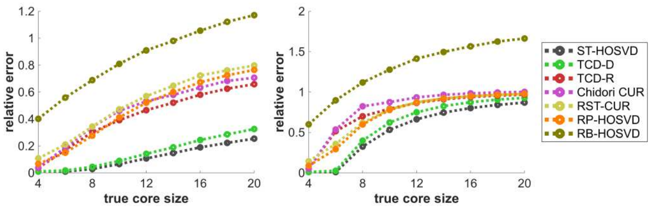


Fig. 6. Relative error w.r.t. the true core size R.
Left: CPD model data. Right: Tucker model data.

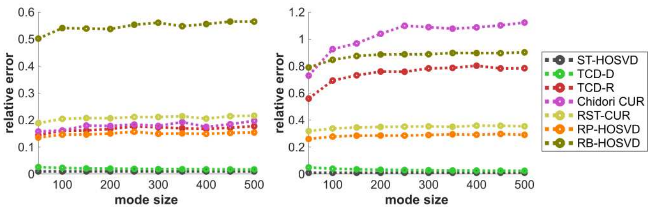


Fig. 7. Relative error w.r.t. the mode size D.
Left: CPD model data. Right: Tucker model data.

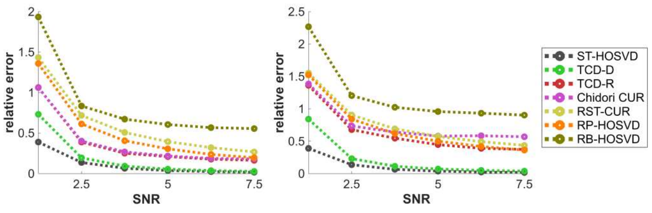


Fig. 8. Relative error w.r.t. the SNR η.
Left: CPD model data. Right: Tucker model data.

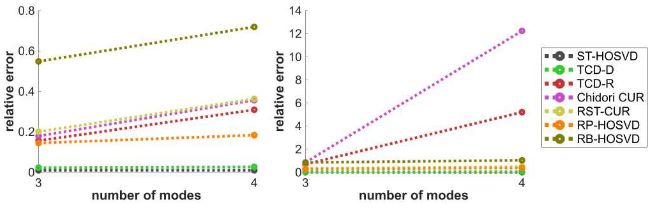


Fig. 9. Relative error w.r.t. the number of modes N.
Left: CPD model data. Right: Tucker model data.

to the true core size R. Whereas Fig. 10 shows a U-shaped curve with varying $\hat{R}$, Fig. 11 shows that increasing R strictly worsens the methods' performances as $R < \hat{R}$ for a fixed $\hat{R} = 4$. All methods perform poorly when R is too large for the Tucker model data. However with the CPD model data, TCD-D performs significantly better than all other methods, especially with large R.

Fig. 12 plots the methods' HOSVD distances with respect to the mode size D. Like in Fig. 7, performances are mostly constant in D with the CPD data (left), but for the Tucker data, all methods except TCD-D feature slightly worse performances with larger D, whereas TCD-D actually features slightly better performances for larger D, up to diminishing returns. We suspect the reason for TCD-D actually doing better for larger D is that as all other variables are fixed, the tensor is generated the same with different D but there are just more elements available to consider subsets over, in which case TCD-D's deterministic WKH has more options of a subset that better minimize the discrepancy measure, and thus better match the PCs of the tensor.

Fig. 13 plots the methods' HOSVD distances with respect to the SNR η. Like in Fig. 8, subject to diminishing returns, all methods perform significantly better with higher SNR, with TCD-R's performance slightly better than Chidori CUR but typically worse than RST-CUR and RP-HOSVD. Whereas in Fig. 8 all methods' relative errors nearly converge to 0 with increased SNR, TCD-D' HOSVD distance in Fig. 13 converges significantly faster to 0 with increased SNR than the other methods' HOSVD distances.

Fig. 14 plots the methods' HOSVD distances with respect to the number of modes N, for $N = 3$ and $N = 4$. Like in Fig. 9, TCD-R Chidori CUR perform worse with $N = 4$ with the Tucker model data, whereas all other methods' HOSVD distances are not as much affected by N.

To summarize these experiments, we observe that TCD-R is among the most efficient of these methods, and TCD-D is also efficient when $\hat{R}$ is small. In most experiments, TCD-R yields comparatively better approximation error and HOSVD distance performance vs. other methods with similar time complexities. Furthermore, TCD-D's performance is typically significantly better than all other tested methods, and even competes closely to that of ST-HOSVD despite using only a subset of the tensor's elements.

In the next section, we perform these methods on real data in the form of fMRI functional connectivity matrices (FNCs), where we visually demonstrate performance of these methods and also demonstrate the use of TCD-D for feature selection.

B. EXPERIMENT WITH FMRI DATA

Our experiments use resting-state fMRI data from the bipolar-schizophrenia network on intermediate phenotypes (B-SNIP) [114], [115], where our data tensor $\mathcal{X}$ was obtained from the acquisition and preprocessing steps described in [116], [117]. The main goals of these experiments are to:

- Demonstrate performance of the tensor decomposition methods on real fMRI data in terms of estimation quality and computational efficiency.
- Demonstrate TCD-D's ability (unique among these

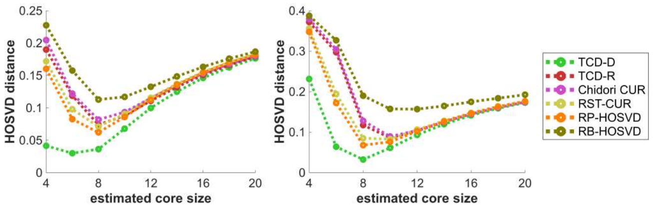


Fig. 10. HOSVD distance w.r.t. the estimated core size $\hat{R}$. Left: CPD model data. Right: Tucker model data.

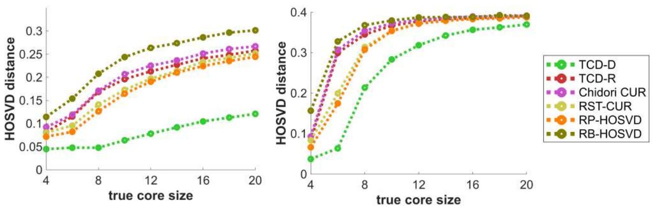


Fig. 11. HOSVD distance w.r.t. the true core size R. Left: CPD model data. Right: Tucker model data.

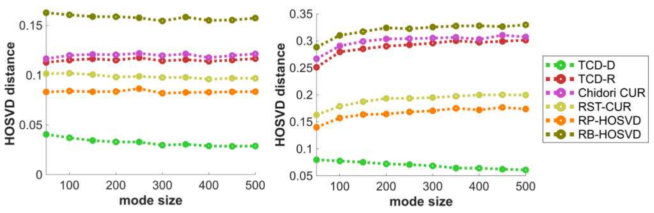


Fig. 12. HOSVD distance w.r.t. the mode size D. Left: CPD model data. Right: Tucker model data.

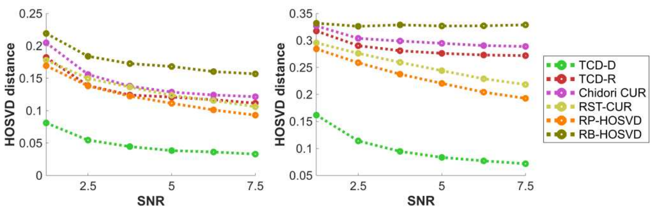


Fig. 13. HOSVD distance w.r.t. the SNR η. Left: CPD model data. Right: Tucker model data.

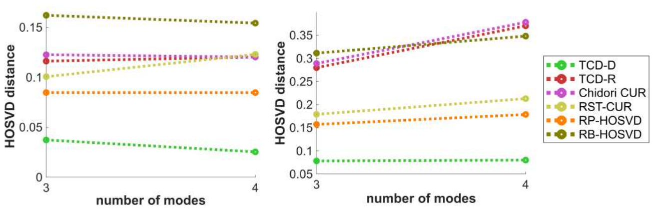


Fig. 14. HOSVD distance w.r.t. the number of modes N. Left: CPD model data. Right: Tucker model data.

methods) to perform feature selection within modes, selecting well-representative elements of the data. In our case, these elements are functional networks (FNs) which are typically used to characterize neurological phenomenon.

We now detail how the data tensor $\mathcal{X}$ was formed. The fMRI dataset includes 176 healthy control and 176 schizophrenia patients for a total of $K = 352$ subjects. The data was first preprocessed and then analyzed via constrained independent vector analysis (cIVA) to extract meaningful latent factors for describing the data. From each subject's data, 53 spatial factors were extracted which correspond to biologically important functional networks (FNs). These factors are representative of seven different functional domains: subcortical (SC, 5 FNs), auditory (AUD, 2 FNs), sensorimotor (MOT, 9 FNs), visual (VIS, 9 FNs), cognitive control (CC, 17 FNs), default mode (DMN, 7 FNs) and cerebellar (CB, 4 FNs). Corresponding to each of these 53 spatial factors are time course factors, representing amplitudes of the networks at each point of measurement, and the correlations between these time courses are particularly useful for representing relationships between the networks. All pairwise Pearson correlations between any two of the 53 networks' time courses is represented in a symmetric 53×53 matrix called a functional network connectivity (FNC) matrix. Our experiment constructs these FNC matrices across each of 352 subjects, and forms an FNC tensor $\mathcal{X} \in \mathbb{R}^{53\times53\times352}$.

A key factor in dealing with the data tensor is understanding its effective n-ranks given how the tensor was obtained. Our FNC data was extracted from functional networks that are expected to be maximally statistically independent from one another, being extracted from cIVA which maximizes statistical independence between networks. Therefore, we expect low correlation between the spatial components of different networks, and this can also result in time courses that demonstrate low correlation between disparate networks. This results in a tensor with effectively high n-ranks, thus decompositions of FNC tensors like $\mathcal{X}$ require higher numbers of factors $\hat{R}_n$ to adequately approximate the FNCs. This presents a challenge for the decomposition methods to approximate the tensor with relatively fewer factors, allowing us to better magnify and compare the methods' estimation capabilities.

Due to the higher n-ranks of the FNC tensor, we test the algorithms on two different forms of the FNC tensor: one being the original FNC tensor, and the other being the elementwise squaring of the FNC tensor. The elementwise squaring provides R-squared values representing the degree of association between the network time courses. Taking the elementwise square of these FNCs effectively increases the spread of the singular values of each mode unfolding $\mathbf{X}_{(n)}$, allowing for better approximation with lower-rank models while still maintaining an interpretable decomposition.

For our experiments, we did a prior exploratory analysis over several candidates of estimated numbers of factors $\hat{R}_n$, and ultimately implemented $[\hat{R}_1, \hat{R}_2, \hat{R}_3] = [20, 20, 352]$ for both forms of the tensor. The reasoning for these choice of $\hat{R}_n$ were as follows: to better exemplify the approximation quality differences between the methods, to reasonably approximate the FNCs without too many factors, and to provide a more parsimonious model which TCD-D can then use to select networks whose R-squared values are "well representative" of all R-squared values in $\mathcal{X}$. These 20 networks could then be interpreted as particularly informative for approximating the relationships between any of the 53 networks.

We use the same tensor decomposition methods in Section V.A to decompose our FNC tensor $\mathcal{X}$. In order to also exploit the symmetry of $\mathcal{X}$, we modify each of these methods to use the same symmetry exploiting process described at the end of Section IV.D. Therefore, since the first and second modes are symmetric (pertaining to the 53 networks), the same factor matrix is used for both of these modes, and the core tensor is thus also symmetric with respect to these modes.

As done in the previous section, our experiments measure performance via CPU time, relative error, and HOSVD distance. Additionally, we implement a measure of how consistent the methods' approximated HOSVD decompositions

Table 3. Performances of methods on the original FNC tensor $\mathcal{X} \in \mathbb{R}^{53\times53\times352}$, averaged over 1000 independent runs over the data. Best performances per measure are bolded.

	CPU-time (sec)	relative error	HOSVD distance	cross-distance
ST-HOSVD	0.026	**0.500**	**0**	**0**
TCD-D	0.025	0.650	0.128	**0**
TCD-R	0.005	0.686	0.130	0.145
Chidori CUR	0.007	0.687	0.133	0.150
RST-CUR	**0.002**	0.691	0.132	0.169
RP-HOSVD	0.006	0.689	0.207	0.169
RB-HOSVD	0.015	0.940	0.148	0.163

Table 4. Performances of methods on the elementwise squared FNC tensor $\mathcal{X} \in \mathbb{R}^{53\times53\times352}$, averaged over 1000 independent runs over the data. Best performances per measure are bolded.

	CPU-time (sec)	relative error	HOSVD distance	cross-distance
ST-HOSVD	0.026	**0.384**	**0**	**0**
TCD-D	0.025	0.464	0.150	**0**
TCD-R	0.005	0.514	0.155	0.155
Chidori CUR	0.007	0.520	0.160	0.157
RST-CUR	**0.002**	0.549	0.156	0.185
RP-HOSVD	0.006	0.528	0.218	0.175
RB-HOSVD	0.015	0.890	0.176	0.183

are with respect to different runs of the decompositions, which corresponds to different random subsets per run for the randomized methods. In defining this measure, we denote $\hat{\boldsymbol{\mathcal{A}}}^{[m]}$ as the approximated HOSVD factors from a mth run of a decomposition method over the data, and define the set of the $\hat{\boldsymbol{\mathcal{A}}}^{[m]}$ across M runs by the set $\boldsymbol{\mathcal{F}} = \left\{\hat{\boldsymbol{\mathcal{A}}}^{[1]}, \ldots, \hat{\boldsymbol{\mathcal{A}}}^{[M]}\right\}$. Then our measure of "cross-distance", the average distance between any two runs of a decomposition, is given by:

$$\text{cross-distance}(\boldsymbol{\mathcal{F}}) = \frac{\sum_{\substack{m_1=1 \\ m_2=1}}^{M} \text{HOSVD distance}\left(\hat{\boldsymbol{\mathcal{A}}}^{[m_1]}, \hat{\boldsymbol{\mathcal{A}}}^{[m_2]}\right)}{M^2}. \tag{19}$$

This "cross-distance" can be considered a generalization of the "cross-ISI" measure used to measure distances between runs for Blind Source Separation (BSS) methods [118].

Along with using cross-distance to measure the variability of the randomized methods, we also use cross-distance to obtain a single run that is the most well representative of all other runs, for which we may plot the FNCs approximated by this run to visually compare the average approximation quality of the methods. The plotted average FNCs were obtained by constructing the most representative run's approximate tensor $\hat{\boldsymbol{\mathcal{X}}}$ from its factorization, and then averaging the approximated subject FNCs across the 352 subjects.

Fig. 15 and Fig. 16 exhibit the average FNCs extracted from a typical run of each method, on the FNC tensor and squared FNC tensor respectively. In both forms of the data, the FNCs typically feature two well-defined blocks on the

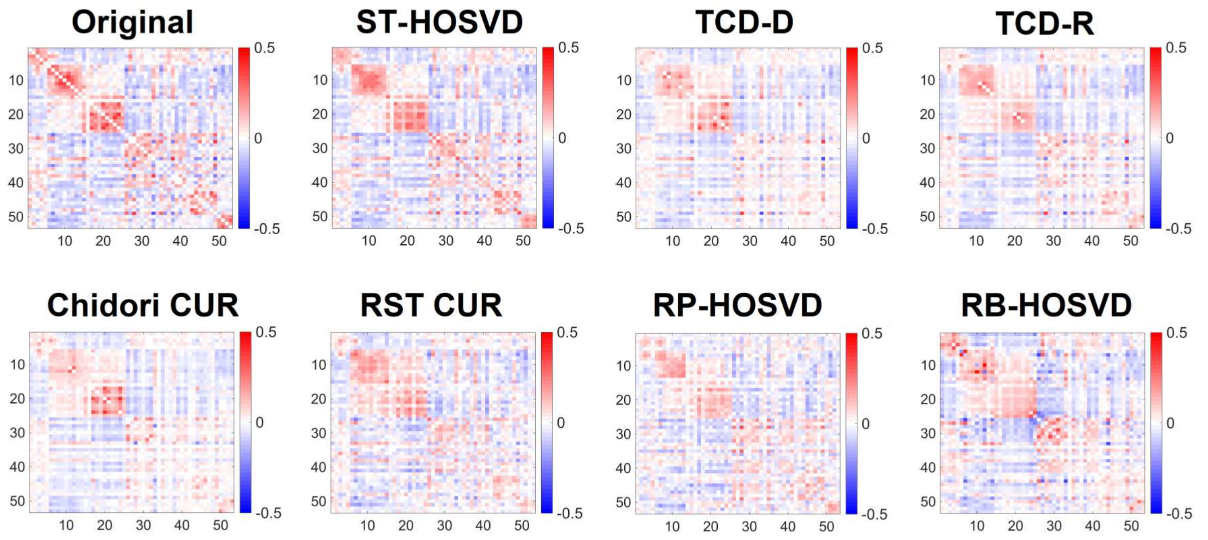


Fig. 15. Plots of the average FNCs obtained by the approximated original FNC tensor $\hat{\boldsymbol{\mathcal{X}}}$, for each method's most typical run (the run with the minimum cross-distance to all other runs). All methods used the ranks $[\hat{R}_1, \hat{R}_2, \hat{R}_3] = [20, 20, 352]$.

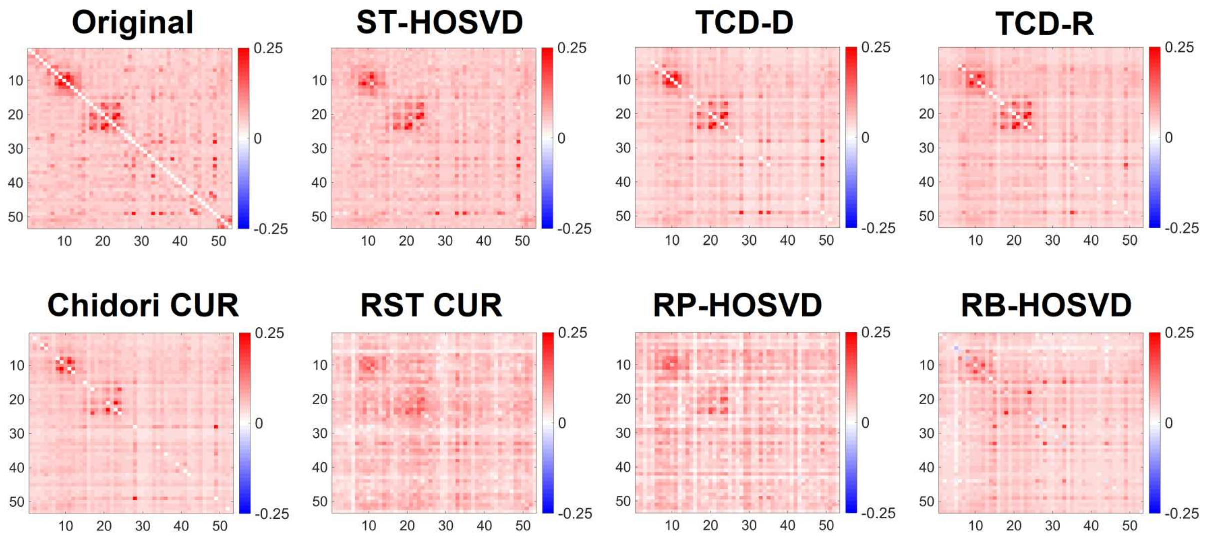


Fig. 16. Plots of the average FNCs obtained by the approximated elementwise squared FNC tensor $\hat{\boldsymbol{\mathcal{X}}}$, for each method's most typical run (the run with the minimum cross-distance to all other runs). All methods used the ranks $[\hat{R}_1, \hat{R}_2, \hat{R}_3] = [20, 20, 352]$.

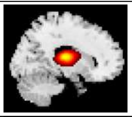
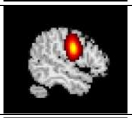
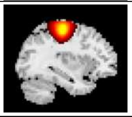
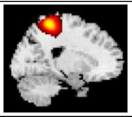
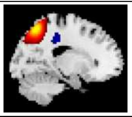
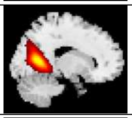
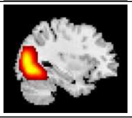
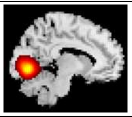
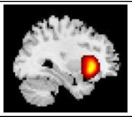
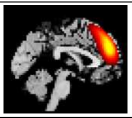
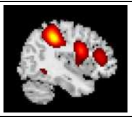
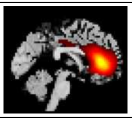
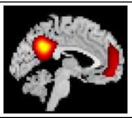
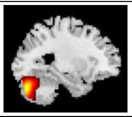
diagonal. These correspond to the motor (upper block) and visual (lower block) groups of networks, which feature high correlation and R-squared values within the groups. Because of the larger degree of association within these networks, their larger values in $\boldsymbol{\mathcal{X}}$ lead them to be especially important for approximating $\boldsymbol{\mathcal{X}}$. Viewing the averages of FNCs in Fig. 15, we observe all methods are able to reasonably approximate at least one of these blocks, with ST-HOSVD, TCD-D, TCD-R, Chidori CUR, and RP-HOSVD demonstrating the two well-defined blocks, and TCD-D and TCD-R having performance closest to ST-HOSVD. Viewing the average of squared FNCs in Fig. 16, we note that all methods except for RB-HOSVD demonstrate two clearly defined blocks, with TCD-D and TCD-R having performance closest to ST-HOSVD.

Tables 3 and 4 presents each method's performance measures on the FNC tensor. All methods provide relatively higher relative errors, as the decomposition ranks $[\hat{R}_1, \hat{R}_2, \hat{R}_3] = [20, 20, 352]$ are perhaps relatively conservative for the more heterogeneous nature of the FNC tensor. While in practice we select $\hat{R}_n$ to provide an approximation quality that is nearly identical to the original tensor, our choice of lower $\hat{R}_n$ is useful for better magnifying the approximation capabilities of the methods, which are clearly demonstrated in the much wider range of their values. ST-HOSVD provides the best relative error, and TCD-D features a comparatively similar error while simultaneously identifying representative networks. Among the more efficient methods, RST-CUR is the fastest method but has the second worst error and worst cross-distance, whereas TCD-R is the second fastest method with the third lowest error, HOSVD distance, and cross-distance. This demonstrates that TCD-D and TCD-R provide good performance measures given their time complexities, and can provide reasonably good approximations to the tensor with fewer factors $\hat{R}_n$.

Additionally, a key distinction between TCD-D and the other methods is that TCD-D deterministically selects elements that are well representative of the tensor. Thus, TCD-D is unique among these methods for the capability of performing feature selection with the tensor data. With this fMRI dataset, TCD-D deterministically selects a reasonably "best" subset of the factor networks. We now consider the interpretation of the TCD-D selected networks. We observed that several TCD-D selected networks were selected not only for the 20 selected networks of the original FNC data, but also the 20 selected networks of the elementwise squared data, highlighting the importance of these networks (a total of 14 networks shared between the two forms of the tensor, corresponding to the indices 5, 8, 9, 12, 15, 17, 23, 24, 27, 28, 33, 45, 49, 51). Table 5 overviews details of these 14 networks identified over both forms of the data tensor, including their associated factor index in the FNCs (their index $i_1 = i_2$ in $\boldsymbol{\mathcal{X}}$), the region of the brain the network corresponds to, and the group of networks it associates with.

These identified networks, including regions such as the thalamus, superior temporal gyrus, superior frontal gyrus, and posterior cingulate cortex, are significant as they represent

Table 5. Descriptions of 14 factors selected by TCD-D, shared between the 20 selected from the original FNC tensor and the 20 from the elementwise-squared FNC tensor.

Index	Region	Network	Component
5	Thalamus	subcortical (SC)	
8	Postcentral gyrus	sensorimotor (SM)	
9	Left postcentral gyrus	sensorimotor (SM)	
12	Superior parietal lobule	sensorimotor (SM)	
15	Superior parietal lobule	sensorimotor (SM)	
17	Calcarine gyrus	visual (VIS)	
23	Inferior occipital gyrus	visual (VIS)	
24	Lingual gyrus	visual (VIS)	
27	Inferior parietal lobule	sensorimotor (SM)	
28	Superior frontal gyrus	cognitive control (CC)	
33	Inferior parietal lobule	sensorimotor (SM)	
45	Anterior cingulate cortex	default-mode network (DMN))	
49	Posterior cingulate cortex	default-mode network (DMN)	
51	Cerebellar	Cerebellar (CB)	

crucial functional "blocks of networks" within the brain. Each of these networks is associated with specific functional domains, such as sensorimotor (e.g., left postcentral gyrus, superior parietal lobule), visual (e.g., inferior occipital gyrus), cognitive control (e.g., inferior parietal lobule), and the default mode network (e.g., posterior cingulate cortex). Clinically, these functional networks have been reported as significant brain regions highly associated with various psychiatric disorders. For instance, the superior frontal gyrus and posterior cingulate cortex have been identified in previous research as valuable biomarkers for different psychiatric conditions [116], [119]–[122]. Furthermore, the fact that 14 of the 20 networks were identified over both forms of the tensor (original FNCs, and elementwise squared FNCs) demonstrates robustness of the proposed TCD-D method, showing consistent identification of meaningful functional areas that are associated with several psychiatric disorders. For example, reduced connectivity between the posterior cingulate and frontal areas in patients with first-episode schizophrenia has been reported in [123]. The failure of appropriate posterior cingulate cortex deactivation has been reported as potential biomarker in traumatic brain injury and mental disorders like ADHD, autism and schizophrenia [124].

VI. CONCLUSION

This paper presents efficient Tucker decomposition methods via using a small subtensor as a multilinear basis over the full data tensor, which we refer to as tensor coreset decompositons (TCD). The methods operate by sequentially truncating the tensor by replacing it with a *coreset* of elements from one or more of the tensor's modes, with the coreset calculated such that it minimizes a discrepancy between itself and the HOSVD core tensor: principal components of the tensor's unfoldings. This process sub-sequentially estimates mapping matrices that serve as the decomposition's factor matrices, which can also be useful for efficiently approximating the tensor's HOSVD.

For quantifying the "representativeness" of a coreset over the data tensor, we introduced a discrepancy-based measure that has straightforward connections to the cost function of HOSVD. We use this measure to develop a new efficient nonnegative least squares (NNLS) procedure for selecting the coreset weights, such that we minimize the discrepancy with respect to a choice of subset.

For decompositions that put greater emphasis on efficiency, we proposed "TCD-R" which randomly selects the subsets using norm sampling. For decompositions that place greater emphasis on approximation quality, and utility of selecting well representative subsets and for feature selection, we proposed "TCD-D" which uses a deterministic subset selection scheme based on the method of weighted kernel herding (WKH). Compared to previous methods, TCD-D is notably unique for its ability to perform unsupervised feature selection within the modes of the tensor data.

Finally, we experimentally demonstrate that our methods generally provide good balances between efficiency, approximation error quality, and quality of factors when converted to a HOSVD. Furthermore, we demonstrate on real fMRI FNC data that TCD-D is able to identify meaningful subsets of functional networks which are able to well-approximate the relationships between all networks in the FNC tensor.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Evrim Acar for her expertise and helpful feedback on the paper.

References

[1] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[2] Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao, Danilo P Mandic, et al. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions. *Foundations and Trends® in Machine Learning*, 9(4-5):249–429, 2016.

[3] Andrzej Cichocki, Anh-Huy Phan, Qibin Zhao, Namgil Lee, Ivan Oseledets, Masashi Sugiyama, Danilo P Mandic, et al. Tensor networks for dimensionality reduction and large-scale optimization: Part 2 applications and future perspectives. *Foundations and Trends® in Machine Learning*, 9(6):431–673, 2017.

[4] Ali Zare, Alp Ozdemir, Mark A Iwen, and Selin Aviyente. Extension of PCA to higher order data structures: An introduction to tensors, tensor decompositions, and tensor PCA. *Proceedings of the IEEE*, 106(8):1341–1358, 2018.

[5] Mojtaba Taherisadr, Mohsen Joneidi, and Nazanin Rahnavard. EEG signal dimensionality reduction and classification using tensor decomposition and deep convolutional neural networks. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2019.

[6] Anh Huy Phan and Andrzej Cichocki. Tensor decompositions for feature extraction and classification of high dimensional datasets. *Nonlinear theory and its applications, IEICE*, 1(1):37–68, 2010.

[7] Fengyu Cong, Qiu-Hua Lin, Li-Dan Kuang, Xiao-Feng Gong, Piia Astikainen, and Tapani Ristaniemi. Tensor decomposition of EEG signals: a brief review. *Journal of neuroscience methods*, 248:59–69, 2015.

[8] Y-H Taguchi. Identification of candidate drugs using tensor-decomposition-based unsupervised feature extraction in integrated analysis of gene expression between diseases and DrugMatrix datasets. *Scientific reports*, 7(1):13733, 2017.

[9] Yuan Gao, Guangming Zhang, Chunchun Zhang, Jinke Wang, Laurence T Yang, and Yaliang Zhao. Federated tensor decomposition-based feature extraction approach for industrial IoT. *IEEE Transactions on Industrial Informatics*, 17(12):8541–8549, 2021.

[10] Haiyan Fan, Chang Li, Yulan Guo, Gangyao Kuang, and Jiayi Ma. Spatial–spectral total variation regularized low-rank tensor decomposition for hyperspectral image denoising. *IEEE Transactions on Geoscience and Remote Sensing*, 56(10):6196–6213, 2018.

[11] Tao Lin and Salah Bourennane. Survey of hyperspectral image denoising methods based on tensor decompositions. *EURASIP journal on Advances in Signal Processing*, 2013:1–11, 2013.

[12] Xiao Gong, Wei Chen, Jie Chen, and Bo Ai. Tensor denoising using low-rank tensor train decomposition. *IEEE Signal Processing Letters*, 27:1685–1689, 2020.

[13] Hongyan Zhang, Lu Liu, Wei He, and Liangpei Zhang. Hyperspectral image denoising with total variation regularization and nonlocal low-rank tensor decomposition. *IEEE Transactions on Geoscience and Remote Sensing*, 58(5):3071–3084, 2019.

[14] Jize Xue, Yongqiang Zhao, Wenzhi Liao, and Jonathan Cheung-Wai Chan. Nonlocal low-rank regularized tensor decomposition for hyperspectral image denoising. *IEEE Transactions on Geoscience and Remote Sensing*, 57(7):5174–5189, 2019.

[15] Zemin Zhang, Gregory Ely, Shuchin Aeron, Ning Hao, and Misha Kilmer. Novel methods for multilinear data completion and de-noising based on tensor-SVD. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3842–3849, 2014.

[16] Evrim Acar, Daniel M Dunlavy, Tamara G Kolda, and Morten Mørup. Scalable tensor factorizations for incomplete data. *Chemometrics and Intelligent Laboratory Systems*, 106(1):41–56, 2011.

[17] Gregory Ely, Shuchin Aeron, Ning Hao, and Misha E Kilmer. 5d seismic data completion and denoising using a novel class of tensor decompositions. *Geophysics*, 80(4):V83–V95, 2015.

[18] Huachun Tan, Guangdong Feng, Jianshuai Feng, Wuhong Wang, Yu-Jin Zhang, and Feng Li. A tensor-based method for missing traffic data completion. *Transportation Research Part C: Emerging Technologies*, 28:15–27, 2013.

[19] Qingquan Song, Hancheng Ge, James Caverlee, and Xia Hu. Tensor completion algorithms in big data analytics. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(1):1–48, 2019.

[20] Hong Chen, Mingwei Lin, Jiaqi Liu, Hengshuo Yang, Chao Zhang, and Zeshui Xu. NT-DPTC: a non-negative temporal dimension preserved tensor completion model for missing traffic data imputation. *Information Sciences*, 653:119797, 2024.

[21] Boaz Barak, Jonathan A Kelner, and David Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 143–151, 2015.

[22] Yanbo Zhang, Xuanqin Mou, Ge Wang, and Hengyong Yu. Tensor-based dictionary learning for spectral CT reconstruction. *IEEE transactions on medical imaging*, 36(1):142–154, 2016.

[23] Syed Zubair and Wenwu Wang. Tensor dictionary learning with sparse Tucker decomposition. In *2013 18th international conference on Digital Signal Processing (DSP)*, pages 1–6. IEEE, 2013.

[24] Diriba Gemechu. Sparse regularization based on orthogonal tensor dictionary learning for inverse problems. *Mathematical Problems in Engineering*, 2024(1):9655008, 2024.

[25] Yuhui Song, Zijun Gong, Yuanzhu Chen, and Cheng Li. Tensor-based sparse Bayesian learning with intra-dimension correlation. *IEEE Transactions on Signal Processing*, 71:31–46, 2023.

[26] Nicholas D Sidiropoulos, Lieven De Lathauwer, Xiao Fu, Kejun Huang, Evangelos E Papalexakis, and Christos Faloutsos. Tensor decomposition for signal processing and machine learning. *IEEE Transactions on signal processing*, 65(13):3551–3582, 2017.

[27] Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and Huy Anh Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE signal processing magazine*, 32(2):145–163, 2015.

[28] Lek-Heng Lim and Pierre Comon. Multiarray signal processing: Tensor decomposition meets compressed sensing. *Comptes Rendus Mecanique*, 338(6):311–320, 2010.

[29] Lieven De Lathauwer and Bart De Moor. From matrix to tensor: Multilinear algebra and signal processing. In *Institute of mathematics and its applications conference series*, volume 67, pages 1–16. Citeseer, 1998.

[30] Hongyang Chen, Fauzia Ahmad, Sergiy Vorobyov, and Fatih Porikli. Tensor decompositions in wireless communications and MIMO radar. *IEEE Journal of Selected Topics in Signal Processing*, 15(3):438–453, 2021.

[31] Mikael Sørensen and Lieven De Lathauwer. Coupled tensor decompositions for applications in array signal processing. In *2013 5th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 228–231. IEEE, 2013.

[32] Karelia Pena-Pena, Daniel L Lau, and Gonzalo R Arce. T-HGSP: Hypergraph signal processing using t-product tensor decompositions. *IEEE Transactions on Signal and Information Processing over Networks*, 9:329–345, 2023.

[33] Age K Smilde, Rasmus Bro, and Paul Geladi. *Multi-way analysis: applications in the chemical sciences*. John Wiley & Sons, 2005.

[34] Viktor Skantze, Mikael Wallman, Ann-Sofie Sandberg, Rikard Landberg, Mats Jirstrand, and Carl Brunius. Identification of metabotypes in complex biological data using tensor decomposition. *Chemometrics and Intelligent Laboratory Systems*, 233:104733, 2023.

[35] Yingyue Bi, Yingcong Lu, Zhen Long, Ce Zhu, and Yipeng Liu. Tensor decompositions: computations, applications, and challenges. *Tensors for Data Processing*, pages 1–30, 2022.

[36] Cristian Minoccheri, Reza Soroushmehr, Jonathan Gryak, and Kayvan Najarian. Tensor methods for clinical informatics. In *Artificial Intelligence in Healthcare and Medicine*, pages 261–281. CRC Press, 2022.

[37] Di Wang, Yao Zheng, and Guodong Li. High-dimensional low-rank tensor autoregressive time series modeling. *Journal of Econometrics*, 238(1):105544, 2024.

[38] Monica Billio, Roberto Casarin, Matteo Iacopini, and Sylvia Kaufmann. Bayesian dynamic tensor regression. *arXiv preprint arXiv:1709.09606*, 2017.

[39] Arash Golibagh Mahyari, David M Zoltowski, Edward M Bernat, and Selin Aviyente. A tensor decomposition-based approach for detecting dynamic network states from EEG. *IEEE Transactions on Biomedical Engineering*, 64(1):225–237, 2016.

[40] Su Wei, Yunbo Tang, Tengfei Gao, Yaodong Wang, Fan Wang, and Dan Chen. Scale-variant structural feature construction of EEG stream via component-increased Dynamic Tensor decomposition. *Knowledge-Based Systems*, 294:111747, 2024.

[41] Bhaskar Sen and Keshab K Parhi. Extraction of common task signals and spatial maps from group fMRI using a PARAFAC-based tensor decomposition technique. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1113–1117. IEEE, 2017.

[42] Christos Chatzichristos, Eleftherios Kofidis, Manuel Morante, and Sergios Theodoridis. Blind fMRI source unmixing via higher-order tensor decompositions. *Journal of neuroscience methods*, 315:17–47, 2019.

[43] Evrim Acar, Yuri Levin-Schwartz, Vince D Calhoun, and Tülay Adali. Tensor-based fusion of EEG and fMRI to understand neurological changes in schizophrenia. In *2017 IEEE international symposium on circuits and systems (ISCAS)*, pages 1–4. IEEE, 2017.

[44] Christos Chatzichristos, Eleftherios Kofidis, Wim Van Paesschen, Lieven De Lathauwer, Sergios Theodoridis, and Sabine Van Huffel. Early soft and flexible fusion of electroencephalography and functional magnetic resonance imaging via double coupled matrix tensor factorization for multisubject group analysis. *Human brain mapping*, 43(4):1231–1255, 2022.

[45] Evrim Acar, Canan Aykut-Bingol, Haluk Bingol, Rasmus Bro, and Bülent Yener. Multiway analysis of epilepsy tensors. *Bioinformatics*, 23(13):i10–i18, 2007.

[46] Evrim Acar, Marie Roald, Khondoker M Hossain, Vince D Calhoun, and Tülay Adali. Tracing evolving networks using tensor factorizations vs. ICA-based approaches. *Frontiers in neuroscience*, 16:861402, 2022.

[47] Eleftherios Kofidis and Phillip A Regalia. Tensor approximation and signal processing applications. *Contemporary Mathematics*, 280:103–134, 2001.

[48] Naimahmed Nesaragi, Shivnarayan Patidar, and Veerakumar Thangaraj. A correlation matrix-based tensor decomposition method for early prediction of sepsis from clinical data. *Biocybernetics and Biomedical Engineering*, 41(3):1013–1024, 2021.

[49] Eleftherios Kofidis. Adaptive joint channel estimation/data detection in flexible multicarrier MIMO systems — a tensor-based approach. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8721–8725. IEEE, 2024.

[50] Feng Xu, Matthew W Morency, and Sergiy A Vorobyov. DOA estimation for transmit beamspace MIMO radar via tensor decomposition with Vandermonde factor matrix. *IEEE Transactions on Signal Processing*, 70:2901–2917, 2022.

[51] Yuwang Ji, Qiang Wang, Xuan Li, and Jie Liu. A survey on tensor techniques and applications in machine learning. *IEEE Access*, 7:162950–162990, 2019.

[52] Stephan Rabanser, Oleksandr Shchur, and Stephan Günnemann. Introduction to tensor decompositions and their applications in machine learning. *arXiv preprint arXiv:1711.10781*, 2017.

[53] Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. Deepfake: improving fake news detection using tensor decomposition-based deep neural network. *The Journal of Supercomputing*, 77(2):1015–1037, 2021.

[54] Alexander Novikov, Pavel Izmailov, Valentin Khrulkov, Michael Figurnov, and Ivan Oseledets. Tensor train decomposition on tensorflow (t3f). *Journal of Machine Learning Research*, 21(30):1–7, 2020.

[55] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Tensor decomposition for compressing recurrent neural network. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[56] Koji Maruhashi, Masaru Todoriki, Takuya Ohwa, Keisuke Goto, Yu Hasegawa, Hiroya Inakoshi, and Hirokazu Anai. Learning multi-way relations via tensor decomposition with neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[57] Frank L Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1-4):164–189, 1927.

[58] Frank L Hitchcock. Multiple invariants and generalized rank of a p-way matrix or tensor. *Journal of Mathematics and Physics*, 7(1-4):39–79, 1928.

[59] Ignat Domanov and Lieven De Lathauwer. On the uniqueness of the canonical polyadic decomposition of third-order tensors—Part I: Basic

results and uniqueness of one factor matrix. *SIAM Journal on Matrix Analysis and Applications*, 34(3):855–875, 2013.

[60] Ignat Domanov and Lieven De Lathauwer. On the uniqueness of the canonical polyadic decomposition of third-order tensors—Part II: Uniqueness of the overall decomposition. *SIAM Journal on Matrix Analysis and Applications*, 34(3):876–903, 2013.

[61] Ignat Domanov and Lieven De Lathauwer. Canonical polyadic decomposition of third-order tensors: Relaxed uniqueness conditions and algebraic algorithm. *Linear Algebra and its Applications*, 513:342–375, 2017.

[62] Ledyard R Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, 15(122-137):3, 1963.

[63] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[64] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[65] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[66] Daniele Bigoni, Allan P Engsig-Karup, and Youssef M Marzouk. Spectral tensor-train decomposition. *SIAM Journal on Scientific Computing*, 38(4):A2405–A2439, 2016.

[67] Lars Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM journal on matrix analysis and applications*, 31(4):2029–2054, 2010.

[68] Nadav Cohen, Or Sharir, Yoav Levine, Ronen Tamari, David Yakira, and Amnon Shashua. Analysis and design of convolutional networks via hierarchical tensor decompositions. *arXiv preprint arXiv:1705.02302*, 2017.

[69] Lieven De Lathauwer. Decompositions of a higher-order tensor in block terms—part ii: Definitions and uniqueness. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1033–1066, 2008.

[70] Jinmian Ye, Linnan Wang, Guangxi Li, Di Chen, Shandian Zhe, Xinqi Chu, and Zenglin Xu. Learning compact recurrent neural networks with block-term tensor decomposition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9378–9387, 2018.

[71] Evrim Acar, Tamara G Kolda, and Daniel M Dunlavy. All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422*, 2011.

[72] Beyza Ermi , Evrim Acar, and A Taylan Cemgil. Link prediction in heterogeneous data via generalized coupled tensor factorization. *Data Mining and Knowledge Discovery*, 29:203–236, 2015.

[73] Shuo Zhou, Nguyen Xuan Vinh, James Bailey, Yunzhe Jia, and Ian Davidson. Accelerating online CP decompositions for higher order tensors. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1375–1384, 2016.

[74] Yishuai Du, Yimin Zheng, Kuang-chih Lee, and Shandian Zhe. Probabilistic streaming tensor decomposition. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 99–108. IEEE, 2018.

[75] Athanasios A Rontogiannis, Eleftherios Kofidis, and Paris V Giampouras. Online rank-revealing block-term tensor decomposition. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pages 1678–1682. IEEE, 2021.

[76] Furong Huang, UN Niranjan, Mohammad Umar Hakeem, and Animashree Anandkumar. Online tensor methods for learning latent variable models. *The Journal of Machine Learning Research*, 16(1):2797–2835, 2015.

[77] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006.

[78] Michael W Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

[79] Michael W Mahoney, Mauro Maggioni, and Petros Drineas. Tensor-CUR decompositions for tensor-based data. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 327–336, 2006.

[80] Petros Drineas and Michael W Mahoney. A randomized algorithm for a tensor-based generalization of the singular value decomposition. *Linear algebra and its applications*, 420(2-3):553–571, 2007.

[81] Cesar F Caiafa and Andrzej Cichocki. Generalizing the column–row matrix decomposition to multi-way arrays. *Linear Algebra and its Applications*, 433(3):557–573, 2010.

[82] HanQin Cai, Keaton Hamm, Longxiu Huang, and Deanna Needell. Mode-wise tensor decompositions: Multi-dimensional generalizations of CUR decompositions. *Journal of machine learning research*, 22(185):1–36, 2021.

[83] Salman Ahmadi-Asl, Stanislav Abukhovich, Maame G Asante-Mensah, Andrzej Cichocki, Anh Huy Phan, Tohishisa Tanaka, and Ivan Oseledets. Randomized algorithms for computation of Tucker decomposition and higher order SVD (HOSVD). *IEEE Access*, 9:28684–28706, 2021.

[84] Salman Ahmadi-Asl, Cesar F Caiafa, Andrzej Cichocki, Anh Huy Phan, Toshihisa Tanaka, Ivan Oseledets, and Jun Wang. Cross tensor approximation methods for compression and dimensionality reduction. *IEEE Access*, 9:150809–150838, 2021.

[85] Bokai Cao, Lifang He, Xiangnan Kong, S Yu Philip, Zhifeng Hao, and Ann B Ragin. Tensor-based multi-view feature selection with applications to brain diseases. In *2014 IEEE International Conference on Data Mining*, pages 40–49. IEEE, 2014.

[86] Aaron Smalter, Jun Huan, and Gerald Lushington. Feature selection in the tensor product feature space. In *2009 Ninth IEEE International Conference on Data Mining*, pages 1004–1009. IEEE, 2009.

[87] Jun Yu, Zhaoming Kong, Liang Zhan, Li Shen, and Lifang He. Tensor-based multi-modality feature selection and regression for alzheimer's disease diagnosis. *Computer science & information technology*, 12(18):123, 2022.

[88] Yongshan Zhang, Xinxin Wang, Zhihua Cai, Yicong Zhou, and S Yu Philip. Tensor-based unsupervised multi-view feature selection for image recognition. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2021.

[89] Nick Vannieuwenhoven, Raf Vandebril, and Karl Meerbergen. A new truncation strategy for the higher-order singular value decomposition. *SIAM Journal on Scientific Computing*, 34(2):A1027–A1052, 2012.

[90] Sinead A Williamson and Jette Henderson. Understanding collections of related datasets using dependent MMD coresets. *Information*, 12(10):392, 2021.

[91] Raaz Dwivedi and Lester Mackey. Generalized kernel thinning. *arXiv preprint arXiv:2110.01593*, 2021.

[92] Zohar Karnin and Edo Liberty. Discrepancy, coresets, and sketches in machine learning. In *Conference on Learning Theory*, pages 1975–1993. PMLR, 2019.

[93] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

[94] Ilya O Tolstikhin, Bharath K Sriperumbudur, and Bernhard Schölkopf. Minimax estimation of maximum mean discrepancy with radial kernels. *Advances in Neural Information Processing Systems*, 29, 2016.

[95] Ferenc Huszár and David Duvenaud. Optimally-weighted herding is Bayesian quadrature. *arXiv preprint arXiv:1204.1664*, 2012.

[96] Christos Boutsidis, Michael W Mahoney, and Petros Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 968–977. SIAM, 2009.

[97] Ahmed K Farahat, Ahmed Elgohary, Ali Ghodsi, and Mohamed S Kamel. Greedy column subset selection for large-scale data sets. *Knowledge and Information Systems*, 45:1–34, 2015.

[98] Jason Altschuler, Aditya Bhaskara, Gang Fu, Vahab Mirrokni, Afshin Rostamizadeh, and Morteza Zadimoghaddam. Greedy column subset selection: New bounds and distributed algorithms. In *International conference on machine learning*, pages 2539–2548. PMLR, 2016.

[99] Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)*, 51(6):1025–1041, 2004.

[100] Alessandro Rudi, Daniele Calandriello, Luigi Carratino, and Lorenzo Rosasco. On fast leverage score sampling and optimal learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[101] Xiaofeng Jiang, Xiaodong Wang, Jian Yang, Shangwu Chen, and Xi Qin. Faster TKD: Towards lightweight decomposition for large-scale tensors with randomized block sampling. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):7966–7979, 2022.

[102] Ali Civril. Column subset selection problem is UG-hard. *Journal of Computer and System Sciences*, 80(4):849–859, 2014.

[103] Ali Civril and Malik Magdon-Ismail. Column subset selection via sparse approximation of SVD. *Theoretical Computer Science*, 421:1–14, 2012.

[104] Ahmed K Farahat, Ali Ghodsi, and Mohamed S Kamel. An efficient greedy method for unsupervised feature selection. In *2011 IEEE 11th International Conference on Data Mining*, pages 161–170. IEEE, 2011.

[105] Ahmed K Farahat, Ali Ghodsi, and Mohamed S Kamel. A fast greedy algorithm for generalized column subset selection. *arXiv preprint arXiv:1312.6820*, 2013.

[106] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[107] Yifan Pi, Haoruo Peng, Shuchang Zhou, and Zhihua Zhang. A scalable approach to column-based low-rank matrix approximation. In *Twenty-Third International Joint Conference on Artificial Intelligence*. Citeseer, 2013.

[108] Bruno Ordozgoiti, Alberto Mozo, and Jesús García López de Lacalle. Regularized greedy column subset selection. *Information Sciences*, 486:393–418, 2019.

[109] Michal Derezinski, Rajiv Khanna, and Michael W Mahoney. Improved guarantees and a multiple-descent curve for column subset selection and the Nystrom method. *Advances in Neural Information Processing Systems*, 33:4953–4964, 2020.

[110] Rasmus Bro and Sijmen De Jong. A fast non-negativity-constrained least squares algorithm. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 11(5):393–401, 1997.

[111] Shun'ichi Amari Amari et al. Advances in neural information processing systems. In *Advances in neural information processing systems*, volume 8, page 757–763. Cambridge, MA: MIT Press, 1996.

[112] Guoxu Zhou, Andrzej Cichocki, and Shengli Xie. Decomposition of big tensors with low multilinear rank. *arXiv preprint arXiv:1412.1885*, 2014.

[113] Tamara G Kolda and Brett W Bader. Matlab tensor toolbox. Technical report, Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA …, 2006.

[114] Carol A Tamminga, Elena I Ivleva, Matcheri S Keshavan, Godfrey D Pearlson, Brett A Clementz, Bradley Witte, David W Morris, Jeffrey Bishop, Gunvant K Thaker, and John A Sweeney. Clinical phenotypes of psychosis in the Bipolar-Schizophrenia Network on Intermediate Phenotypes (B-SNIP). *American Journal of psychiatry*, 170(11):1263–1274, 2013.

[115] Carol A Tamminga, Godfrey Pearlson, Matcheri Keshavan, John Sweeney, Brett Clementz, and Gunvant Thaker. Bipolar and schizophrenia network for intermediate phenotypes: outcomes across the psychosis continuum. *Schizophrenia bulletin*, 40(Suppl_2):S131–S137, 2014.

[116] Hanlu Yang, Trung Vu, Qunfang Long, Vince Calhoun, and Tülay Adali. Identification of homogeneous subgroups from resting-state fMRI data. *Sensors*, 23(6):3264, 2023.

[117] Trung Vu, Francisco Laport, Hanlu Yang, Vince D Calhoun, and Tulay Adali. Constrained independent vector analysis with reference for multi-subject fMRI analysis. *arXiv preprint arXiv:2311.05049*, 2023.

[118] Qunfang Long, Chunying Jia, Zois Boukouvalas, Ben Gabrielson, Darren Emge, and Tulay Adali. Consistent run selection for independent component analysis: Application to fMRI analysis. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2581–2585. IEEE, 2018.

[119] Hideo Matsumoto, Andrew Simmons, Steven Williams, Michael Hadjulis, Roderic Pipe, Robin Murray, and Sophia Frangou. Superior temporal gyrus abnormalities in early-onset schizophrenia: similarities and differences with adult-onset schizophrenia. *American Journal of Psychiatry*, 158(8):1299–1304, 2001.

[120] Tao Li, Qiang Wang, Jie Zhang, Edmund T Rolls, Wei Yang, Lena Palaniyappan, Lu Zhang, Wei Cheng, Ye Yao, Zhaowen Liu, et al. Brain-wide analysis of functional connectivity in first-episode and chronic stages of schizophrenia. *Schizophrenia bulletin*, 43(2):436–448, 2017.

[121] Hanlu Yang, Mohammad ABS Akhonda, Fateme Ghayem, Qunfang Long, Vince D Calhoun, and Tülay Adali. Independent vector analysis based subgroup identification from multisubject fMRI data. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1471–1475. IEEE, 2022.

[122] H Yang, M Ortiz-Bouza, T Vu, F Laport, VD Calhoun, S Aviyente, and T Adali. Subgroup identification through multiplex community structure within functional connectivity networks. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2141–2145. IEEE, 2024.

[123] Sugai Liang, Wei Deng, Xiaojing Li, Qiang Wang, Andrew J Greenshaw, Wanjun Guo, Xiangzhen Kong, Mingli Li, Liansheng Zhao, Yajing Meng, et al. Aberrant posterior cingulate connectivity classify first-episode schizophrenia from controls: A machine learning study. *Schizophrenia research*, 220:187–193, 2020.

[124] Robert Leech and David J Sharp. The role of the posterior cingulate cortex in cognition and disease. *Brain*, 137(1):12–32, 2014.

BEN GABRIELSON received his B.A. degree in Physics from Franklin and Marshall College, Lancaster PA, USA, in 2015, and received his M.S. degree in Electrical Engineering from the University of Maryland, Baltimore County, Baltimore, MD, USA, in 2020. He is currently pursuing a PhD degree in Electrical Engineering at the University of Maryland, Baltimore County, under the supervision of Dr. Tülay Adali.

His research interests include matrix and tensor factorizations, with a particular emphasis on efficient implementations of blind source separation and joint blind source separation.

HANLU YANG (Graduate Student Member, IEEE) received her B.A. degree from Jilin University, China, and her M.S. degree from Temple University, Philadelphia, USA, both in Electrical Engineering. She is currently pursuing a PhD degree in Electrical Engineering at the University of Maryland, Baltimore County, USA, under the supervision of Dr. Tülay Adali.

Her research interests include matrix and tensor factorizations, machine/deep learning, statistical/graph signal processing, with applications in community detection, precision medicine, and large-scale neuroimaging data analysis.

TRUNG VU received the B.S. degree in Computer Science from Hanoi University of Science and Technology, Hanoi, Vietnam, in 2014. He received the Ph.D. degree in Computer Science at the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, Oregon, USA, in 2022. He is currently a postdoctoral research associate at the Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, USA.

His research interests include optimization methods, independent component analysis, and matrix factorization with applications in machine learning and signal processing. Dr. Vu received a Best Student Paper Award at IEEE MLSP 2019.

VINCE CALHOUN is founding director of the tri-institutional Center for Translational Research in Neuroimaging and Data Science (TReNDS) where he holds appointments at Georgia State, Georgia Tech and Emory. He is the author of more than 1,100 full journal articles. His work includes the development of flexible methods to analyze neuroimaging data including blind source separation, deep learning, multimodal fusion and genomics, and neuroinformatics tools.

Dr. Calhoun is a fellow of the Institute of Electrical and Electronic Engineers, The American Association for the Advancement of Science, The American Institute of Biomedical and Medical Engineers, The American College of Neuropsychopharmacology, The Organization for Human Brain Mapping (OHBM) and the International Society of Magnetic Resonance in Medicine. He currently serves on the IEEE BISP Technical Committee and is also a member of IEEE Data Science Initiative Steering Committee as well as the IEEE Brain Technical Committee.

TÜLAY ADALI is a Distinguished University Professor at the University of Maryland Baltimore County (UMBC), Baltimore, MD. She received the Ph.D. degree in Electrical Engineering from North Carolina State University, Raleigh, NC, USA, in 1992 and joined the faculty at UMBC the same year.

Prof. Adali has been active in conference organizations, and served or will serve as technical chair, 2017, 2025, special sessions chair, 2018, 2024, publicity chair, 2000, 2005, for the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), general/technical chair for the IEEE Machine Learning for Signal Processing (MLSP) and Neural Networks for Signal Processing Workshops 20012009, 2014, and 2023. She was the Chair of the NNSP/MLSP Technical Committee, 2003–2005 and 2011–2013, and served or is currently serving on numerous boards and technical committees of the SPS. She served as the Chair of the IEEE Brain Technical Community in 2023, and the Signal Processing Society (SPS) Vice President for Technical Directions 20192022.

She is currently the editor-in-chief of the Signal Processing Magazine. Prof. Adali is a Fellow of the IEEE, AIMBE, and AAIA, a Fulbright Scholar, and an IEEE SPS Distinguished Lecturer. She is the recipient of SPS Meritorious Service Award, a Humboldt Research Award, an IEEE SPS Best Paper Award, SPIE Unsupervised Learning and ICA Pioneer Award, Presidential Research Professorship at UMBC, the University System of Maryland Regents' Award for Research, and the NSF CAREER Award.