FAST AND MEMORY-EFFICIENT SUBSPACE EMBEDDINGS FOR TENSOR DATA WITH APPLICATIONS

By

Ali Zare

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computational Mathematics, Science and Engineering – Doctor of Philosophy

2022

ABSTRACT

FAST AND MEMORY-EFFICIENT SUBSPACE EMBEDDINGS FOR TENSOR DATA WITH APPLICATIONS

By

Ali Zare

The widespread use of multisensor technology and the emergence of big data sets have brought the necessity to develop more versatile tools to represent higher-order data with multiple aspects and high dimensionality. Data in the form of multidimensional arrays, also referred to as tensors, arise in a variety of applications including chemometrics, physics, hyperspectral imaging, high-resolution videos, neuroimaging, biometrics, and social network analysis. Early multiway data analysis approaches used to reformat such tensor data as large vectors or matrices and would then resort to dimensionality reduction methods developed for low-dimensional data. However, by vectorizing tensors, the inherent multiway structure of the data and the possible correlation between different dimensions will be lost, in some cases resulting in a degradation in the performance of vector-based methods. Moreover, in many cases, vectorizing tensors leads to vectors with extremely high dimensionality that might render most existing methods computationally impractical. In the case of dimension reduction, the enormous amount of memory needed to store the embedding matrix becomes the main obstacle. This highlights the need for approaches that are applied to tensor data in their multi-dimensional form. To reduce the dimension of an $n_1 \times n_2 \times \cdots \times n_d$ tensor to $m_1 \times m_2 \times \cdots \times m_d$ with $m_j \leq n_j$, MPCA[1] would change the memory requirement from $\prod_{j=1}^{d} m_j n_j$ for vector PCA to $\sum_{j=1}^{d} m_j n_j$, which can be a considerable improvement. On the other hand, tensor dimension reduction methods such as MPCA need training samples for the projection matrices to be learned. This makes such methods time consuming and computationally less efficient than oblivious approaches such as the Johnson-Lindenstrauss embedding. The term *oblivious* refers to

[1] Multilinear Principal Component Analysis

the fact that one does not need any data samples beforehand to learn the embedding that projects a new data sample onto a lower-dimensional space.

In this thesis, first a review of tensor concepts and algebra as well as common tensor decompositions is presented. Next, a modewise JL approach is proposed for compressing tensors without reshaping them into potentially very large vectors. Theoretical guarantees for the norm and inner product approximation errors as well as theoretical bounds on the embedding dimension are presented for data with low CP rank, and the corresponding effects of basis coherence assumptions are addressed. Experiments are performed using various choices of embedding matrices. Results verify the validity of one- and two-stage modewise JL embeddings in preserving the norm of MRI and synthesized data constructed from both coherent and incoherent bases. Two novel applications of the proposed modewise JL method are discussed. (i) Approximate solutions to least squares problems as a computationally efficient way of fitting tensor decompositions: The proposed approach is incorporated as a stage in the fitting procedure, and is tested on relatively low-rank MRI data. Results show improvement in computational complexity at a slight cost in the accuracy of the solution in the Euclidean norm. (ii) Many-Body Perturbation Theory problems involving energy calculations: In large model spaces, the dimension sizes of tensors can grow fast, rendering the direct calculation of perturbative correction terms challenging. The second-order energy correction term as well as the one-body radius correction are formulated and modeled as inner products in such a way that modewise JL can be used to reduce the computational complexity of the calculations. Experiments are performed on data from various nuclei in different model space sizes, and show that in the case of large model spaces, very good compression can be achieved at the price of small errors in the estimated energy values.

ACKNOWLEDGEMENTS

I would like to warmly thank my academic advisor, Dr. Mark Iwen, for his supervision and tutelage throughout the course of my Ph.D. studies. I truely appreciate his support, encouragement, and patience in the face of many challenges that I tackled on my academic journey at Michigan State University. My appreciation extends to the dissertation committee members, Dr. Selin Aviyente, Dr. Rongrong Wang, and Dr. Yuying Xie for their experience and knowledge that they supported me with during research and coursework. I also hereby thank Dr. Heiko Hergert who I collaborated with and learned from in the novel application of my thesis work to Many-Body Perturbation Theory problems in physics.

Finally, I would like to express my deepest gratitude to my dear parents and sister without whose tremendous and unconditional support throughout my life, and especially during the past few years, earning my Ph.D. degree would not have been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

The emergence of big data elicits the development of compression methods to efficiently represent such data without losing much information. One of the most well-known techniques to this end is PCA[1] which uses the linear structure of a high-dimensional vector and projects it onto the underlying lower-dimensional subspace [2]. However, when the number of dimensions in the data increases, as is the case with matrices and cubes, reshaping the data into a vector becomes troublesome in the sense that it will require huge amounts of memory to store the matrix that will project data elements onto their corresponding principal components. This problem, for instance, can be observed in the case of MRI[2] data. Take, for instance, a $240 \times 240 \times 155$ cube from an MRI data set, containing 8928000 data elements. To reduce the dimensionalty of this vector to 0.1% of its original size, a 8928×8928000 projection matrix needs to be generated. This means an approximate 594 Gigabytes of data only to store the matrix. It is obvious how intensive memory requirements could be if higher-dimensional data with larger mode sizes were to be dealt with. This simple example clearly demonstrates the importance of dimension reduction techniques that do not rely on the vectorization of higher-dimensional data, and that deal with such data in their original multidimensional form.

In addition to computational considerations, one can intuitively observe that if the multilinear structure of tensor data is changed, e.g. by vectorization, many conventional approaches that were initially developed for vector data might not yield the same satisfactory results. Generally speaking, although tensors are natural extensions to vectors and matrices, their multidimensional form adds extra complexity to their structure in a way that the notion low-dimensionality becomes challenging to address, especially simply as an extension of the same notion from matrices to tensors. This

[1] Principal Component Analysis
[2] Magnetic Resonance Imaging

issue will be addressed in Section 3.2 where tensor rank is introduced, and it is discussed that there are various notions of rank for tensor data. In the experiments of Section 6.2, it is shown that for higher-dimensional data, a conventional vector-based method such as PCA becomes both computationally more expensive and less accurate compared to its multilinear counterpart. In this case, an extension of PCA to tensor data, abbreviated to MPCA[3], will be presented in Chapter 6. This method performs PCA on fibers of the unfoldings of a tensor one mode at a time, and is specifically useful to compress tensors with low Tucker ranks [14].

An efficient method to compute MPCA would be to compress the data using the general randomized embedding proposed in [9], which constitutes the central body of work in this thesis, before starting the main algorithm. This approach can in general be applied as a preprocessing stage to large scale tensor data to alleviate the computational intensity of the subsequent processing scheme. To make tensor compression even more computationally efficient, randomized streaming mappings are of great value where one will not have to store a big tensor to compute its compressed form. Rather, a sketch of the unfoldings of the tensor along with a sketch of the core in the factorization of interest will be enough to construct an approximate version of the decomposition. Such a method has been proposed to compute of the Tucker approximation in [22], and can also be applied to MPCA.

On the other hand, dimension reduction methods such as PCA and its extensions, including MPCA, need a set of training samples (vectors, or tensors in the multilinear case) for the projection matrix or matrices to be found. This makes these methods time consuming and computationally less efficient than oblivious approaches such as the Johnson-Lindenstrauss embedding. The term *oblivious* refers to the fact that one does not need any data sample beforehand to project a new data sample onto a lower-dimensional subspace.

In recent years, many tensor dimension reduction techniques that can be applied to low-rank data have been proposed in the literature. Most of these methods are limited to some degree in

[3]Multilinear Principal Component Analysis

the sense that they either are limited in the number of data modes and/or lack general theoretical guarantees on the error in the geometry preserving property of the embedding. In [23], such theoretical guarantees have been presented for the vectorized form of a tensor projected using the Khatri-Rao product of individual random projections of smaller sizes, and have been extended to 2-mode tensors. In [18, 19], the CountSketch projection matrix has been extended to tensors, named TensorSketch, although the multilinear structure of the tensor is again not preserved. In a more recent version [20], however, TensorSketch is developed based on the Tucker format to extend CountSketch to tensor data. The TensorSketch method is mainly developed for polynomial kernels as a special case of rank-1 tensors [19, 3].

There are other more related methods that assume specific data structure for the input tensors and at the same time respect their multimodal structure. In a closely related method abbreviated to KFJLT[4] [10], a very large Fast Johnson-Lindenstrauss embedding matrix, addressed in Section 4.2.3, is used in the form of the Kronecker product of smaller fast embeddings, and is applied to a vector also having Kronecker structure corresponding to the vectorized form of a rank-1 tensor. The implementation of KFJLT, however, is done without actually forming the extremely large embedding matrix or input vector, and the elements of the compressed vector (or tensor, equivalently) can be calculated efficiently using the smaller embeddings implicitly applied to the corresponding tensor modes. In each mode, this is made possible by using random rows of the DFT matrix, a vector consisting of Rademacher random variables, and smaller vectors that form the Kronecker structure of the input data. The rank-1 property of the input tensor naturally draws one's attention to KFJLT being suitable for the efficient computation of CP decompositions. This method leads to lower computational cost at a small price in the embedding dimension size, and its computational efficiency originates from both the inherent speed of the fast embedding matrices used and also the Kronecker structure of the input data in the vectorized form. KFJLT is shown to work for tensors having general structure with a reduction in performance. A short summary of

[4]Kronecker Faster Johnson-Lindenstrauss Transform.

how computational efficiency is achieved in this method works is presented in Appendix C.

The work presented in this dissertation contains materials discussed in the following publications. In [27], extensions of PCA and its variants to tensor data are discussed for those who are well familiar with PCA methods for vector-type data. In [17], a multiscale HoSVD[5] approach has been developed to compress tensors in multiple scales. In the 0^{th} scale, truncated HoSVD is performed on data. The reconstruction error tensor is then partitioned into subtensors in the 1^{st} scale using a clustering algorithm, and truncated HoSVD is applied to each subtensor. This process can be repeated in higher scales depending on the needed tradeoff between reconstruction error and compression. In [9] which constitues the main body of this thesis presented in Chapter 4, a modewise Johnson-Lindenstrauss embedding has been proposed for compressing tensor data without reshaping the tensor into an extremely large vector. Theoretical guarantees for the approximation error have been presented for data with low CP rank.

[5]Higher-order Singular Value Dicomposition

CHAPTER 2

BACKGROUND: TENSOR BASICS AND ALGEBRA

In this chapter, basic concepts and algebraic relations that are used in the statement of problems and proofs are presented.

Notation. The type of letters used for tensors, matrices, vectors and scalars are as follows. Calligraphic boldface capital letters (e.g., $\mathcal{X}$) are used for tensors, boldface capital letters for matrices (e.g., $\mathbf{X}$), boldface lower-case letters for vectors (e.g., $\mathbf{x}$), and regular (lower-case or capital) letters for scalars (e.g., x or X).

For numbers in parentheses used as subscript or superscript, subscript refers to "unfoldings" while superscript denotes an object in a sequence of objects.

We assume $[d] := \{1, \ldots, d\}$ for all $d \in \mathbb{N}$.

Whenever used, the $\text{vec}(\cdot)$ operator generates the vectorized form of its argument.

In the following definitions, we assume $\mathcal{X} \in \mathbb{C}^{n_1 \times \ldots \times n_d}$.

Definition 2.0.1 *A "tensor" is a multi-dimensional array. A d-way, d-mode or d^{th}-order tensor is an element of the tensor product of d vector spaces. Figure 2.1 shows an example of a* 3*-mode tensor. A tensor is called "cubical" is all its modes are of the same size, i.e., $\mathcal{X} \in \mathbb{C}^{n \times \ldots \times n}$.*

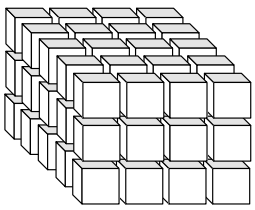


Figure 2.1: An example of a $3 \times 4 \times 5$ tensor.

One can stack 3-mode tensors along modes 1, 2, and 3 of a 3-mode tensor to visualize higher-order data in the 3-dimensional space, where the elements of the stacked versions can obviously be different. For instance, Figure 2.2 illustrates this idea where the 3-mode tensor of Figure 2.1 is

stacked along the 1st and 2nd dimensions to simulate the 4th and 5th modes, respectively, leading to a $3 \times 4 \times 5 \times 3 \times 2$ tensor.

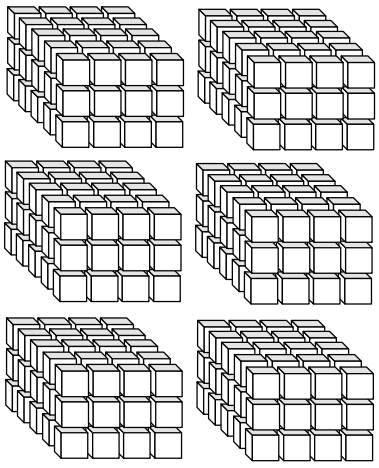


Figure 2.2: Visualization of a 5-mode tensor by stacking a 3-mode tensor along its 1st and 2nd modes. The result is a 5-mode tensor of size $3 \times 4 \times 5 \times 3 \times 2$. Note that the elements of the stacked versions are not necessarily the same as they are elements corresponding to different indices in the 5-mode tensor.

Definition 2.0.2 (Mode-j Fiber) *In a d-mode tensor $\mathcal{X}$, a mode-j fiber is obtained by fixing all but the j^{th} index, and is denoted by $\mathcal{X}_{i_1,\ldots,i_{j-1},:,j_{j+1},\ldots,i_d} \in \mathbb{C}^{n_j}$ for $i_j \in [n_j]$ and $j \in [d]$. There are $\prod_{i \neq j} n_j$ mode-j fibers. Figure 2.3 depicts how the fibers of a 3-mode tensor are formed.*

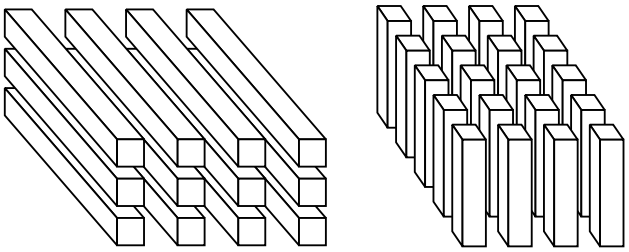


Figure 2.3: An example of the fibers of a 3-mode tensor. Left: mode-3 fibers. Right: mode-1 fibers.

Definition 2.0.3 (Mode-j Slice) *In a d-mode tensor $\mathcal{X}$, a mode-j "slice" is a $(d-1)$-mode subtensor obtained by fixing the j^{th} index. A mode-j slice of $\mathcal{X}$ is denoted by $\mathcal{X}_{:,\ldots,:,k,:,\ldots,:} \in \mathbb{C}^{n_1 \times \ldots \times n_{j-1} \times n_{j+1} \times \ldots \times n_d}$ for $i_j = k \in [n_j]$. There are n_j mode-j slices.*

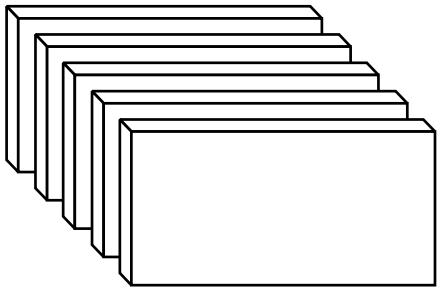


Figure 2.4: An example showing how the mode-3 slices of a 3-mode tensor are formed.

Definition 2.0.4 (Matricization of a Tensor) *The process of reshaping a tensor $\mathcal{X}$ into a matrix is called "matricization", "flattening" or "unfolding". The most common way of doing this is the mode-j unfolding, denoted by $\mathbf{X}_{(j)} \in \mathbb{C}^{n_j \times \prod_{i \neq j} n_i}$, which has all the mode-j fibers of $\mathcal{X}$ as its columns. The process of matricization of tensors is linear, in the sense that for $\mathcal{X}, \mathcal{Y} \in \mathbb{C}^{n_1 \times \ldots \times n_d}$, $(\alpha\mathcal{X} + \beta\mathcal{Y})_{(j)} = \alpha\mathbf{X}_{(j)} + \beta\mathbf{Y}_{(j)}$ for $j \in [d]$ and $\alpha, \beta \in \mathbb{C}$.*

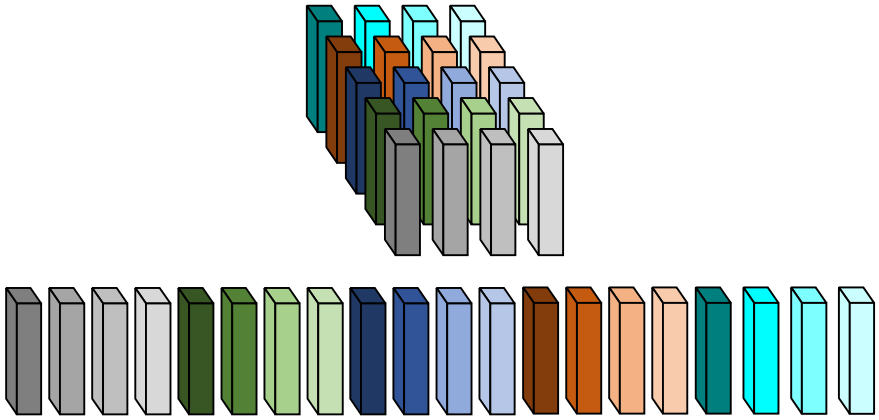


Figure 2.5: An example showing how the mode-1 unfolding of a 3-mode tensor is formed using its mode-1 fibers. Colors are used to show how this is done in a column-major format.

Lemma 2.0.1 *Assume we have a tensor $\mathcal{X} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d}$, and for $i_j \in [n_j]$, $j \in [d]$ and $\ell \in [\ \prod_{\substack{m=1 \\ m \neq j}}^{d} n_m\ ]$, we want to find the element (i_j, ℓ) of the mode-j unfolding $\mathbf{X}_{(j)}$ corresponding to the element $(i_1, i_2, \ldots, i_d)$ of $\mathcal{X}$ [1]. Then, for a given (i_j, ℓ), we have*

$$\mathbf{X}_{(j)}(i_j, \ell) = \mathcal{X}_{i_1, \ldots, i_{j-1}, i_j, i_{j+1}, \ldots, i_d},$$

[1] In this thesis, we always use column-major formatting when reshaping tensors to matrices/vectors and vice versa. This essentially means going from lower to higher modes when moving across dimensions.

where

$$i_l = \left\lfloor \frac{\ell - 1 - \sum\limits_{\substack{k=l+1 \\ k\neq j}}^{d} (i_k - 1) \prod\limits_{\substack{m=1 \\ m\neq j}}^{k-1} n_m}{\prod\limits_{\substack{m=1 \\ m\neq j}}^{l-1} n_m} \right\rfloor + 1, \quad \text{for } l \neq j, \tag{2.1}$$

starting from $l = d$ and going down to $l = 1$. Obviously, for $l = d$, the numerator is reduced to $\ell - 1$, and for $l = 1$, the denominator becomes 1. It is also clear that i_j will be the same in both the tensor and the unfolding.

On the other hand, assume we want to obtain a tensor $\mathcal{X}$ from its mode-j unfolding $\mathbf{X}_{(j)}$. Given indices $(i_1, \ldots, i_d)$, we want to find the corresponding coordinates (i_j, ℓ) in $\mathbf{X}_{(j)}$. We can do so using

$$\ell = 1 + \sum_{\substack{k=1 \\ k\neq j}}^{d} (i_k - 1) \prod_{\substack{m=1 \\ m\neq j}}^{k-1} n_m, \quad \text{for } \ell \in \Big[\prod_{\substack{m=1 \\ m\neq j}}^{d} n_m \Big], \tag{2.2}$$

meaning that

$$\mathcal{X}(i_1, \ldots, i_{j-1}, i_j, i_{j+1}, \ldots, i_d) = \mathbf{X}_{(j)}(i_j, \ell),$$

with ℓ defined above [12].

Definition 2.0.5 (The Standard Inner Product Space of *d*-mode Tensors) *The set of all d-mode tensors $\mathcal{X} \in \mathbb{C}^{n_1 \times \ldots \times n_d}$ forms a vector space over the field of complex numbers when equipped with component-wise addition and scalar multiplication. The inner product of $\mathcal{X}$ and $\mathcal{Y}$ is defined as*

$$\langle \mathcal{X}, \mathcal{Y} \rangle := \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \ldots \sum_{i_d=1}^{n_d} \mathcal{X}_{i_1,i_2,\ldots,i_d} \, \overline{\mathcal{Y}_{i_1,i_2,\ldots,i_d}}. \tag{2.3}$$

The standard Euclidean norm can be deduced from this inner product, as

$$\|\mathcal{X}\| := \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle} = \sqrt{\sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \ldots \sum_{i_d=1}^{n_d} \left| \mathcal{X}_{i_1,i_2,\ldots,i_d} \right|^2}. \tag{2.4}$$

Definition 2.0.6 (Tensor Outer Product) *The tensor outer product of two tensors* $\mathcal{X} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d}$ *and* $\mathcal{Y} \in \mathbb{C}^{n'_1 \times n'_2 \times \cdots \times n'_{d'}}$*, denoted by* $\mathcal{X} \bigcirc \mathcal{Y} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d \times n'_1 \times n'_2 \times \cdots \times n'_{d'}}$*, is a* $(d + d')$*-mode tensor whose entries are given by*

$$(\mathcal{X} \bigcirc \mathcal{Y})_{i_1,\ldots,i_d,i'_1,\ldots,i'_{d'}} = \mathcal{X}_{i_1,\ldots,i_d} \mathcal{Y}_{i'_1,\ldots,i'_{d'}}. \tag{2.5}$$

When $\mathcal{X}$ *and* $\mathcal{Y}$ *are both vectors, the tensor outer product will be reduced to the standard outer product.*

Definition 2.0.7 (Rank-1 Tensor) *A* d*-mode tensor* $\mathcal{X} \in \mathbb{C}^{n_1 \times \ldots \times n_d}$ *is rank-*1 *if it can be written as the outer product of* d *vectors, i.e.,*

$$\mathcal{X} = \mathbf{x}^{(1)} \bigcirc \mathbf{x}^{(2)} \bigcirc \ldots \bigcirc \mathbf{x}^{(d)} =: \bigcirc_{j=1}^{d} \mathbf{x}^{(j)}, \tag{2.6}$$

where $\mathbf{x}^{(j)} \in \mathbb{C}^{n_j}$ *for* $j \in [d]$*.*

Definition 2.0.8 (j-mode Product) *The* j*-mode product of a* d*-mode tensor* $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_{j-1} \times n_j \times n_{j+1} \times \cdots \times n_d}$ *with a matrix* $\mathbf{U} \in \mathbb{C}^{m_j \times n_j}$ *is another* d*-mode tensor* $\mathcal{X} \times_j \mathbf{U} \in \mathbb{C}^{n_1 \times \cdots \times n_{j-1} \times m_j \times n_{j+1} \times \cdots \times n_d}$ *whose entries are given by*

$$(\mathcal{X} \times_j \mathbf{U})_{i_1,\ldots,i_{j-1},\ell,i_{j+1},\ldots,i_d} = \sum_{i_j=1}^{n_j} \mathcal{X}_{i_1,\ldots,i_j,\ldots,i_d} \mathbf{U}_{\ell,i_j}. \tag{2.7}$$

for all $(i_1, \ldots, i_{j-1}, \ell, i_{j+1}, \ldots, i_d) \in [n_1] \times \cdots \times [n_{j-1}] \times [m_j] \times [n_{j+1}] \times \cdots \times [n_d]$*. In terms of the mode-*j *unfoldings of* $\mathcal{X} \times_j \mathbf{U}$ *and* $\mathcal{X}$*, it can be observed that* $(\mathcal{X} \times_j \mathbf{U})_{(j)} = \mathbf{U}\mathbf{X}_{(j)}$ *holds for all* $j \in [d]$*.*

Lemma 2.0.2 *Let* $\mathcal{X}, \mathcal{Y} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d}$*,* $\mathcal{A}, \mathcal{B} \in \mathbb{C}^{n'_1 \times n'_2 \times \cdots \times n'_{d'}}$*,* $\alpha, \beta \in \mathbb{C}$*, and* $\mathbf{U}_\ell, \mathbf{V}_\ell \in \mathbb{C}^{m_\ell \times n_\ell}$ *for all* $\ell \in [d]$*. The following four properties hold:*

(a) $(\alpha\mathcal{X} + \beta\mathcal{Y}) \bigcirc \mathcal{A} = \alpha\mathcal{X} \bigcirc \mathcal{A} + \beta\mathcal{Y} \bigcirc \mathcal{A} = \mathcal{X} \bigcirc \alpha\mathcal{A} + \mathcal{Y} \bigcirc \beta\mathcal{A}$

(b) $\langle \mathcal{X} \bigcirc \mathcal{A}, \mathcal{Y} \bigcirc \mathcal{B} \rangle = \langle \mathcal{X}, \mathcal{Y} \rangle \langle \mathcal{A}, \mathcal{B} \rangle$

(c) $(\alpha\mathcal{X} + \beta\mathcal{Y}) \times_j \mathbf{U}_j = \alpha\left(\mathcal{X} \times_j \mathbf{U}_j\right) + \beta\left(\mathcal{Y} \times_j \mathbf{U}_j\right)$.

(d) $\mathcal{X} \times_j \left(\alpha\mathbf{U}_j + \beta\mathbf{V}_j\right) = \alpha\left(\mathcal{X} \times_j \mathbf{U}_j\right) + \beta\left(\mathcal{X} \times_j \mathbf{V}_j\right)$.

(e) If $j \neq \ell$ *then* $\mathcal{X} \times_j \mathbf{U}_j \times_\ell \mathbf{V}_\ell = \left(\mathcal{X} \times_j \mathbf{U}_j\right) \times_\ell \mathbf{V}_\ell = (\mathcal{X} \times_\ell \mathbf{V}_\ell) \times_j \mathbf{U}_j = \mathcal{X} \times_\ell \mathbf{V}_\ell \times_j \mathbf{U}_j$.

(f) If $W \in \mathbb{C}^{p \times m_j}$ *then* $\mathcal{X} \times_j \mathbf{U}_j \times_j \mathbf{W} = \left(\mathcal{X} \times_j \mathbf{U}_j\right) \times_j \mathbf{W} = \mathcal{X} \times_j \left(\mathbf{W}\mathbf{U}_j\right) = \mathcal{X} \times_j \mathbf{W}\mathbf{U}_j$.

Proof *The proof of (a) can be done element-wise.*

$$\begin{aligned}((\alpha\mathcal{X} + \beta\mathcal{Y}) \bigcirc \mathcal{A})_{i_1,\ldots,i_d,i'_1,\ldots,i'_{d'}} &= (\alpha\mathcal{X} + \beta\mathcal{Y})_{i_1,\ldots,i_d}\, \mathcal{A}_{i'_1,\ldots,i'_{d'}} \\ &= \left(\alpha\mathcal{X}_{i_1,\ldots,i_d} + \beta\mathcal{Y}_{i_1,\ldots,i_d}\right) \mathcal{A}_{i'_1,\ldots,i'_{d'}}.\end{aligned}$$

To prove (b), we note that

$$\begin{aligned}\langle \mathcal{X} \bigcirc \mathcal{A}, \mathcal{Y} \bigcirc \mathcal{B} \rangle &= \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \sum_{i'_1=1}^{n'_1} \cdots \sum_{i'_d=1}^{n'_{d'}} \mathcal{X}_{i_1,i_2,\ldots,i_d} \mathcal{A}_{i'_1,\ldots,i'_{d'}} \overline{\mathcal{Y}_{i_1,i_2,\ldots,i_d}}\, \overline{\mathcal{B}_{i'_1,\ldots,i'_{d'}}} \\ &= \left(\sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \mathcal{X}_{i_1,i_2,\ldots,i_d} \overline{\mathcal{Y}_{i_1,i_2,\ldots,i_d}}\right) \left(\sum_{i'_1=1}^{n'_1} \cdots \sum_{i'_d=1}^{n'_{d'}} \mathcal{A}_{i'_1,\ldots,i'_{d'}} \overline{\mathcal{B}_{i'_1,\ldots,i'_{d'}}}\right) \\ &= \langle \mathcal{X}, \mathcal{Y} \rangle \langle \mathcal{A}, \mathcal{B} \rangle.\end{aligned}$$

The proof of (c), (d), and (f) can be done using the definition of the mode-j *unfolding. For (e), suppose that* $\ell > j$ *(the case where* $\ell < j$ *is similar). Set* $\mathbf{U} := \mathbf{U}_j$ *and* $\mathbf{V} := \mathbf{V}_\ell$ *to simplify subscript*

notation. We have for all $k \in [m_j]$, $l \in [m_\ell]$, *and* $i_q \in [n_q]$ *with* $q \notin \{j, \ell\}$ *that*

$$\begin{aligned}
\left(\left(\mathcal{X} \times_j \mathbf{U}\right) \times_\ell \mathbf{V}\right)_{i_1,\ldots,i_{j-1},k,i_{j+1},\ldots,i_{\ell-1},l,i_{\ell+1},\ldots,i_d} &= \sum_{i_\ell=1}^{n_\ell} \left(\mathcal{X} \times_j \mathbf{U}\right)_{i_1,\ldots,i_{j-1},k,i_{j+1},\ldots,i_\ell,\ldots,i_d} \mathbf{V}_{l,i_\ell} \\
&= \sum_{i_\ell=1}^{n_\ell} \left(\sum_{i_j=1}^{n_j} \mathcal{X}_{i_1,\ldots,i_j,\ldots,i_\ell,\ldots,i_d} \mathbf{U}_{k,i_j} \right) \mathbf{V}_{l,i_\ell} \\
&= \sum_{i_j=1}^{n_j} \left(\sum_{i_\ell=1}^{n_\ell} \mathcal{X}_{i_1,\ldots,i_j,\ldots,i_\ell,\ldots,i_d} \mathbf{V}_{l,i_\ell} \right) \mathbf{U}_{k,i_j} \\
&= \sum_{i_j=1}^{n_j} \left(\mathcal{X} \times_\ell \mathbf{V}\right)_{i_1,\ldots,i_j,\ldots,i_{\ell-1},l,i_{\ell+1},\ldots,i_d} \mathbf{U}_{k,i_j} \\
&= \left(\left(\mathcal{X} \times_\ell \mathbf{V}\right) \times_j \mathbf{U}\right)_{i_1,\ldots,i_{j-1},k,i_{j+1},\ldots,i_{\ell-1},l,i_{\ell+1},\ldots,i_d}.
\end{aligned}$$

Note 2.0.1 *Unfolding the tensor* $\mathcal{Y} = \mathcal{X} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \ldots \times_d \mathbf{U}^{(d)} =: \mathcal{X} \bigtimes_{j=1}^{d} \mathbf{U}^{(j)}$ *along the* j^{th} *mode is equivalent to*

$$\mathbf{Y}_{(j)} = \mathbf{U}^{(j)} \mathbf{X}_{(j)} (\mathbf{U}^{(d)} \otimes \cdots \otimes \mathbf{U}^{(j+1)} \otimes \mathbf{U}^{(j-1)} \otimes \cdots \otimes \mathbf{U}^{(1)})^\top, \tag{2.8}$$

where $\otimes$ *denotes the matrix Kronecker product. If* $\mathcal{X}$ *is a superdiagonal tensor*[2]*, then all matrices* $\mathbf{U}^{(j)}$ *must have the same number of columns, and (2.8) will be simplified to*

$$\mathbf{Y}_{(j)} = \mathbf{U}^{(j)} \mathbf{X} \, (\mathbf{U}^{(d)} \odot \cdots \odot \mathbf{U}^{(j+1)} \odot \mathbf{U}^{(j-1)} \odot \cdots \odot \mathbf{U}^{(1)})^\top, \tag{2.9}$$

where $\mathbf{X}$ *is a diagonal matrix with the superdiagonal of* $\mathcal{X}$ *as its diagonal. The symbol* $\odot$ *denotes the Khatri-Rao product, which is defined as the column-wise matching Kronecker product, i.e., for matrices* $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_J] \in \mathbb{C}^{I \times J}$ *and* $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_J] \in \mathbb{C}^{K \times J}$,

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \ldots, \mathbf{a}_J \otimes \mathbf{b}_J] \in \mathbb{C}^{IK \times J}.$$

The reason for (2.9) being a simplified form of (2.8) lies in the fact that when a d*-mode tensor* $\mathcal{X} \in \mathbb{C}^{n \times \ldots \times n}$ *is superdiagonal, all columns of* $\mathbf{X}_{(j)}$ *are zeros except for* n *of them spread evenly,*

[2] A d-mode superdiagonal tensor $\mathcal{X} \in \mathbb{C}^{n \times \ldots \times n}$ is cubical and has nonzero elements only at indices $(i, \ldots, i)$ for $i \in [n]$.

where in the ℓ^{th} column, only the entry in position $((\ell-1) \bmod n)+1$ is nonzero[3]*. This means that all but matching columns in the Kronecker product will be crossed out in the final result, simplifying the kronecker product to the Khatri-Rao product, and reducing $\mathbf{X}_{(j)}$ in (2.8) to a diagonal matrix $\mathbf{X}$ in (2.9).*

Note 2.0.2 *Vectorizing the tensor $\mathcal{Y} = \mathcal{X} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \ldots \times_d \mathbf{U}^{(d)}$, it is straightforward to show that*

$$\mathbf{y} = \left(\mathbf{U}^{(d)} \otimes \cdots \otimes \mathbf{U}^{(2)} \otimes \mathbf{U}^{(1)}\right) \mathbf{x}, \tag{2.10}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the vectorized forms of $\mathcal{X}$ and $\mathcal{Y}$, respectively.

Definition 2.0.9 (j-mode Vector Product) *The j-mode product of a d-mode tensor $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$ with a vector $\mathbf{v} \in \mathbb{C}^{n_j}$ is a $(d-1)$-mode tensor, and is denoted by $\mathcal{X} \bullet_j \mathbf{v}$, whose elements are obtained using*

$$\left(\mathcal{X} \bullet_j \mathbf{v}\right)_{i_1,\ldots,i_{j-1},i_{j+1},\ldots,i_d} = \sum_{i_j=1}^{n_j} \mathcal{X}_{i_1,\ldots,i_j,\ldots,i_d} \mathbf{v}_{i_j}, \tag{2.11}$$

which means that the j^{th} mode of $\mathcal{X}$ is contracted with $\mathbf{v}$. If we want to keep the j^{th} mode with dimension size 1, meaning $\mathcal{X} \bullet_j \mathbf{v} \in \mathbb{C}^{n_1 \times n_{j-1} \times 1 \times n_{j+1} \ldots \times n_d}$, a useful interpretation of this will be

$$\mathcal{X} \bullet_j \mathbf{v} = \mathcal{X} \times_j \mathbf{v}^\top, \tag{2.12}$$

which is equivalent to

$$\mathbf{v}^\top \mathbf{X}_{(j)} = \left(\mathcal{X} \bullet_j \mathbf{v}\right)_{(j)} = \left(vec\left(\mathcal{X} \bullet_j \mathbf{v}\right)\right)^\top. \tag{2.13}$$

The mode-j vector product can be used to define eigenvalue problems for tensors. For a super-symmetric tensor[4] *$\mathcal{X} \in \mathbb{C}^{n \times \ldots \times n}$, the scalar λ is an eigenvalue with the corresponding eigenvector $\mathbf{v} \in \mathbb{C}^n$ if*

$$\mathcal{X} \bullet_2 \mathbf{v} \bullet_3 \mathbf{v} \cdots \bullet_d \mathbf{v} = \lambda \mathbf{v}.$$

[3] For integers a and b, we assume that $a \bmod b \in \{0, \ldots, b-1\}$.

[4] A cubical tensor is called supersymmetric if its elements remain the same under any permutaion of indices. Obviously, superdiagonal tensors are also supersymmetric.

The j-mode vector product is also used in developing Support Vector Machines for tensors, where d optimization problems for d modes of a tensor are mixed to form one problem for the whole tensor [6].

Definition 2.0.10 (Tensor (k, j)-Contraction) *Consider tensors $\mathcal{X} \in \mathbb{C}^{n_1 \times \ldots \times n_j \times \ldots \times n_d}$ and $\mathcal{Y} \in \mathbb{C}^{m_1 \times \ldots \times m_{k-1} \times n_j \times m_{k+1} \times \ldots \times m_{d'}}$. Then, for each $j \in [d]$ and $k \in [d']$, the (k, j)-contraction of $\mathcal{X}$ and $\mathcal{Y}$, which is the contraction of modes j of $\mathcal{X}$ and k of $\mathcal{Y}$, is a $(d + d' - 2)$-dimensional array denoted by*

$$\mathcal{Z} := \mathcal{X} \times_j^k \mathcal{Y} \in \mathbb{C}^{n_1 \times \ldots \times n_{j-1} \times m_\ell \times n_{j+1} \times \ldots \times n_d \times m_{L'_1} \times \ldots \times m_{L'_{d'-2}}},$$

where $\ell = \min\{[d'] \setminus k\}$, $L' := [d'] \setminus \{k, \ell\}$, and L'_h denotes the h^{th} element of the set L'. It is observed that $\ell = 1$ for all choices of k except for $k = 1$ in which case $\ell = 2$. Element-wise,

$$\begin{gathered}\mathcal{Z}_{i_1,\ldots,i_{j-1},q_\ell,i_{j+1},\ldots,i_d,q_{L'_1},\ldots,q_{L'_{d'-2}}} = \sum_{i_j=1}^{n_j} \mathcal{X}_{i_1,\ldots,i_j,\ldots,i_d} \mathcal{Y}_{q_1,\ldots,q_{k-1},i_j,q_{k+1},\ldots,q_{d'}}, \\ \text{for} \qquad i_j \in [n_j], j \in [d], q_\ell \in [m_\ell], q_{L'_i} \in [m_{L'_h}], h \in [d' - 2].\end{gathered} \tag{2.14}$$

Note 2.0.3 *If $k = d' = 2$, then $\ell = \min\{[2] \setminus 2\} = 1$ and $L' = [2] \setminus \{1, 2\} = \emptyset$, and the $(2, j)$-contraction of $\mathcal{X}$ and $\mathcal{Y}$ (which is a matrix now, denoted by $\mathbf{Y}$) is reduced to the familiar j-mode product $\mathcal{X} \times_j \mathbf{Y}$.*

Note 2.0.4 *One can also define the (k, j)-contraction of $\mathcal{X}$ and $\mathcal{Y}$ in a way that modes of $\mathcal{Y}$ are interleaved right after contracting the j^{th} mode of $\mathcal{X}$, i.e.,*

$$\mathcal{Z} \in \mathbb{C}^{n_1 \times \ldots \times n_{j-1} \times m_\ell \times m_{L'_1} \times \ldots \times m_{L'_{d'-2}} \times n_{j+1} \times \ldots \times n_d},$$

and

$$\begin{gathered}\mathcal{Z}_{i_1,\ldots,i_{j-1},q_\ell,q_{L'_1},\ldots,q_{L'_{d'-2}},i_{j+1},\ldots,i_d} = \sum_{i_j=1}^{n_j} \mathcal{X}_{i_1,\ldots,i_j,\ldots,i_d} \mathcal{Y}_{q_1,\ldots,q_{k-1},i_j,q_{k+1},\ldots,q_{d'}}, \\ \text{for} \qquad i_j \in [n_j], j \in [d], q_\ell \in [m_\ell], q_{L'_i} \in [m_{L'_i}], i \in [d' - 2].\end{gathered} \tag{2.15}$$

CHAPTER 3

TENSOR DECOMPOSITIONS AND RANK

In this section, a short review on the more commonly used tensor decompositions is presented. For simplicity, the elements of the tensors as well as scalars are defined over the field of real numbers. Results can be extended to the field of complex numbers with slight modifications. This will specifically be the case in Chapter 4.

3.1 The CANDECOMP/PARAFAC Decomposition

This factorization, abbreviated to CPD, decomposes a tensor $\mathcal{X}$ into the (weighted) sum of rank-1 tensors [12]. For $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$,

$$\mathcal{X} \approx \hat{\mathcal{X}} = \sum_{k=1}^{r} g_k \, \mathbf{a}_k^{(1)} \bigcirc \mathbf{a}_k^{(2)} \bigcirc \cdots \bigcirc \mathbf{a}_k^{(d)}, \tag{3.1}$$

where $\bigcirc$ denotes the tensor outer product. The vector $\mathbf{a}_k^{(j)} \in \mathbb{R}^{n_j}$ can be considered as the k^{th} column in a matrix $\mathbf{A}^{(j)} \in \mathbb{R}^{n_j \times r}$ for $j \in [d]$. The scalar g_k can be considered as the k^{th} element of a vector $\mathbf{g}$. Therefore, if $\mathbf{g}$ is set as the superdiagonal of a diagonal tensor $\mathcal{G}$, called the core tensor, then

$$\hat{\mathcal{X}} = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times \cdots \times_d \mathbf{A}^{(d)}. \tag{3.2}$$

Component-wise, we have

$$\hat{\mathcal{X}}_{i_1,\ldots,i_d} = \sum_{k=1}^{r} g_k \, \mathbf{A}^{(1)}_{i_1,k} \mathbf{A}^{(2)}_{i_2,k} \ldots \mathbf{A}^{(d)}_{i_d,k}. \tag{3.3}$$

Considering the superdiagonality of $\mathcal{G}$, and according to (2.9), the relation between the unfoldings of $\mathcal{X}$ and $\mathcal{G}$ may be written as

$$\hat{\mathbf{X}}_{(j)} = \mathbf{A}^{(j)} \mathbf{G} (\mathbf{A}^{(d)} \odot \cdots \odot \mathbf{A}^{(j+1)} \odot \mathbf{A}^{(j-1)} \odot \cdots \odot \mathbf{A}^{(1)})^{\top}, \tag{3.4}$$

where $\mathbf{G}$ is a diagonal matrix with $\mathbf{g}$ as its diagonal.

3.1.1 Uniquensess of CPD

CPD is unique under weak conditions; there is a permutation and scaling indeterminacy. For a permutation matrix $\Pi \in \mathbb{R}^{r\times r}$,

$$\hat{\mathcal{X}} = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times \cdots \times_d \mathbf{A}^{(d)} = \mathcal{G} \times_1 \left(\mathbf{A}^{(1)}\Pi\right) \times \cdots \times_d \left(\mathbf{A}^{(d)}\Pi\right), \tag{3.5}$$

implying that as long as the columns of the factor matrices are permuted in the same way, the factorization will not change. This is also evident from (3.1) where the order in which the terms in the summation are added together does not matter. As for the scaling indeterminacy, we observe that

$$\hat{\mathcal{X}} = \sum_{k=1}^{r} g_k \ \left(\alpha_k^{(1)}\mathbf{a}_k^{(1)}\right) \bigcirc \left(\alpha_k^{(2)}\mathbf{a}_k^{(2)}\right) \bigcirc \cdots \bigcirc \left(\alpha_k^{(d)}\mathbf{a}_k^{(d)}\right), \tag{3.6}$$

as long as $\alpha_k^{(1)}\alpha_k^{(2)}\ldots\alpha_k^{(d)} = 1$.

A sufficient condition for the uniqueness of CPD is [21]

$$\sum_{j=1}^{d} k_{\mathbf{A}^{(j)}} \geq 2r + d - 1, \tag{3.7}$$

where $k_{\mathbf{A}}$ is the k-rank of a matrix $\mathbf{A}$, and is defined as the largest number k such that any k columns of $\mathbf{A}$ are linearly independent. Equation (3.7) is also the necessary condition for uniqueness of CPD for $r = 2, 3$ but not for $r \geq 4$. In its general form, the necessary condition for the uniqueness of CPD is [26]

$$\min_{j\in[d]} \operatorname{rank}\left(\mathbf{A}^{(1)} \odot \cdots \odot \mathbf{A}^{(j-1)} \odot \mathbf{A}^{(j+1)} \odot \cdots \odot \mathbf{A}^{(d)}\right) = r. \tag{3.8}$$

However, noting that

$$\operatorname{rank}(\mathbf{A} \odot \mathbf{B}) \leq \operatorname{rank}(\mathbf{A} \otimes \mathbf{B}) \leq \operatorname{rank}(\mathbf{A})\ \operatorname{rank}(\mathbf{B}),$$

then (3.8) can be simplified to

$$\min_{j\in[d]} \left(\prod_{\substack{m=1 \\ m\neq j}}^{d} \operatorname{rank}\left(\mathbf{A}^{(m)}\right) \right) \geq r. \tag{3.9}$$

Algorithm 3.1: CPD-ALS [12]

initialize $\mathbf{A}^{(j)} \in \mathbb{R}^{n_j \times r}$ for $j \in [d]$
repeat
  for $j = 1, \ldots, d$ **do**
    $\mathbf{V} \leftarrow \mathbf{A}^{(1)\top}\mathbf{A}^{(1)} * \cdots * \mathbf{A}^{(j-1)\top}\mathbf{A}^{(j-1)} * \mathbf{A}^{(j+1)\top}\mathbf{A}^{(j+1)} * \cdots * \mathbf{A}^{(d)\top}\mathbf{A}^{(d)}$
    $\mathbf{A}^{(j)} \leftarrow \mathbf{X}_{(j)} \left(\mathbf{A}^{(d)} \odot \cdots \odot \mathbf{A}^{(j+1)} \odot \mathbf{A}^{(j-1)} \odot \cdots \odot \mathbf{A}^{(1)}\right) \mathbf{V}^{\dagger}$
    normalize columns of $\mathbf{A}^{(j)}$ storing norms as $\mathbf{g}$
  end for
until fit ceases to improve or maximum iterations exhausted
return $\mathbf{g}, \mathbf{A}^{(j)}$ for $j \in [d]$

3.1.2 Computing CPD

At first, assume the number of rank-1 tensors is known beforehand. The problem is now the calculation of factor matrices $\mathbf{A}^{(j)}$ for $j \in [d]$ and $\mathbf{g}$ in (3.1), i.e. the solution to

$$\min_{\hat{\mathcal{X}}} \|\mathcal{X} - \hat{\mathcal{X}}\| \text{ with } \hat{\mathcal{X}} = \sum_{k=1}^{r} g_k \, \mathbf{a}_k^{(1)} \bigcirc \mathbf{a}_k^{(2)} \bigcirc \cdots \bigcirc \mathbf{a}_k^{(d)}. \tag{3.10}$$

Alternating Least Squares (ALS) is a common way of finding the fit. For instance, assume that $\mathcal{X}$ is a 3-mode tensor. Then, in light of (3.10) and given that the Euclidean norm of a tensor is equal to the Frobenius norm of any of its unfoldings, finding $\mathbf{A}^{(1)}$ would be done by solving

$$\mathbf{A}^{(1)} = \min_{\hat{\mathbf{A}}} \left\| \mathbf{X}_{(1)} - \hat{\mathbf{A}} \left(\mathbf{A}^{(3)} \odot \mathbf{A}^{(2)}\right)^{\top} \right\|_F, \tag{3.11}$$

where $\hat{\mathbf{A}} = \mathbf{A}\mathbf{G}$ with $\mathbf{G}$ being an $r \times r$ diagonal matrix with g_k forming its diagonal. The optimal solution to (3.11) would be $\hat{\mathbf{A}} = \mathbf{X}_{(1)} \left(\left(\mathbf{A}^{(3)} \odot \mathbf{A}^{(2)}\right)^{\top}\right)^{\dagger}$ which can be rearranged as

$$\hat{\mathbf{A}} = \mathbf{X}_{(1)} \left(\mathbf{A}^{(3)} \odot \mathbf{A}^{(2)}\right) \left(\mathbf{A}^{(3)\top}\mathbf{A}^{(3)} * \mathbf{A}^{(2)\top}\mathbf{A}^{(2)}\right)^{\dagger}.$$

Here, the symbol $*$ denotes the Hadamard product, and $\dagger$ represents the pseudo-inverse of a matrix. Extending the same idea to a d-mode tensor is outlined in Algorithm 3.1, and is called CPD-ALS. The initialization for $\mathbf{A}^{(j)}$ could be either random or using r leading left singular vectors of $\mathbf{X}_{(j)}$.

The remaining question is how to choose r. Most methods fit multiple CP decompositions with different number of components until one is *good*, i.e., the one that yields an exact representation in the Euclidean norm.

For noiseless data, CPD is computed for $r = 1, 2, \ldots,$ and the first value of r that gives a 100% fit is chosen as rank. This may indeed not work in the case of degenerate tensors (see 3.2.1). For noisy data, which is almost always the case, the fit alone fails to determine rank. A commonly used consistency diagnostic called CORCONDIA[1] is employed to determine the proper number of components [4]. Assume the factor matrices $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d)}$ are fixed, i.e., they have been obtained using a CP procedure. A Tucker model (see 3.3) is next assumed to represent the data as in

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{A}^{(1)} \times \cdots \times_d \mathbf{A}^{(d)}.$$

Noting that the Euclidean norm of a tensor is equal to the Frobenius norm of any of its unfoldings as well as the 2-norm of its vectorized form, the core $\mathcal{G}$ can be found by solving

$$\min_{\mathbf{G}_{(j)}} \left\| \mathbf{X}_{(j)} - \mathbf{A}^{(j)} \mathbf{G}_{(j)} \left(\bigotimes_{\substack{\ell=d \\ \ell \neq j}}^{1} \mathbf{A}^{(\ell)} \right)^{\top} \right\|_F^2 = \min_{\mathrm{vec}(\mathcal{G})} \left\| \mathrm{vec}\,(\mathcal{X}) - \left(\bigotimes_{\ell=d}^{1} \mathbf{A}^{(\ell)} \right) \mathrm{vec}\,(\mathcal{G}) \right\|_2^2,$$

for $j \in [d]$, which can be treated as a least squares problem. Now, the question is how close the core is to a diagonal tensor with a superdiagonal of ones. If there is a 100% match, then the perfect fit has been found. The reason why a diagonal core is sought is that in a perfect CP model, interaction exists only between parallel factors of different modes.

3.2 Tensor Rank

The rank of a tensor $\mathcal{X}$ is defined as the smallest number of rank-1 tensors that generate $\mathcal{X}$ as their sum. In other words, it is the smallest number of components in an *exact* CP decomposition.

Although this definition is similar to the definition of rank in matrices, the properties of tensor rank are very different from matrix rank. The major difference is that there is no straightforward algorithm to compute the rank of a tensor, and in practice, it is determined numerically by fitting various rank-r CP models.

[1] CORe CONsistency DIAgnostic

Other types of rank that are used for tensors are *maximum rank* and *typical rank*. Maximum rank is defined as the largest attainable rank of a tensor. Typical rank is defined as any rank that occurs with probability greater than zero when the elements of the tensor are drawn randomly from a uniform continuous distribution. Typical rank and maximum rank are the same for matrices. However, they may be different for tensors, and there might be more than one typical rank.

3.2.1 Low-rank approximation and border rank

For a matrix $\mathbf{A}$ with rank r and a decomposition of the form $\mathbf{A} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ where $\sigma_1 \geq \cdots \geq \sigma_r$, the *best* rank-$k$ approximation ($k \leq r$) will be obtained by keeping the k leading factors, i.e., $\hat{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$. For tensors, this might not be the case; the best rank-k approximation may not even exist, which is a problem of degeneracy. A tensor is *degenerate* if it can be approximated arbitrarily well by a factorization of lower rank.

When a low-rank approximation does not exist for a tensor, it is useful to introduce the concept of *border rank*. It is defined as the minimum number of rank-one tensors that approximate it with arbitrarily small non-zero error, i.e.,

$$\widetilde{\text{rank}}(\mathcal{X}) = \min\{\, r \mid \forall \varepsilon > 0,\ \exists \mathcal{E};\ \|\mathcal{X} - \mathcal{E}\| < \varepsilon,\ \text{rank}(\mathcal{E}) = r\}. \tag{3.12}$$

Obviously, $\widetilde{\text{rank}}(\mathcal{X}) \leq \text{rank}(\mathcal{X})$.

3.3 Compression and the Tucker Decomposition

The Tucker decomposition can be considered as an extension to CPD, as well as a higher-order principal component analysis. A tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ is decomposed in the Tucker format in the following way.

$$\begin{aligned} \mathcal{X} \approx \hat{\mathcal{X}} &= \sum_{k_1=1}^{r_1} \cdots \sum_{k_d=1}^{r_d} \mathcal{G}_{k_1,\ldots,k_d}\, \mathbf{a}_{k_1}^{(1)} \bigcirc \cdots \bigcirc \mathbf{a}_{k_d}^{(d)} \\ &= \mathcal{G} \times_1 \mathbf{A}^{(1)} \times \cdots \times_d \mathbf{A}^{(d)}, \end{aligned} \tag{3.13}$$

where $\mathcal{G} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ is the core tensor with $r_j \leq n_j$ for $j \in [d]$, and $\mathbf{A}^{(j)} \in \mathbb{R}^{n_j \times r_j}$ is the j^{th} factor matrix whose k_j^{th} column is $\mathbf{a}^{(j)}_{k_j}$ for $k_j \in [r_j]$. Component-wise, we have

$$\hat{\mathcal{X}}_{i_1,\ldots,i_d} = \sum_{k_1=1}^{r_1} \cdots \sum_{k_d=1}^{r_d} \mathcal{G}_{k_1,\ldots,k_d} \, \mathbf{A}^{(1)}_{i_1,k_1} \mathbf{A}^{(2)}_{i_2,k_2} \ldots \mathbf{A}^{(d)}_{i_d,k_d}. \tag{3.14}$$

As can be seen, Tucker approximates a tensor as the linear combination of $\prod_{j=1}^{d} r_j$ rank-1 tensors. Unlike CPD where the interaction between modes is restricted to matching columns of the factor matrices, in Tucker, this interaction occurs between all possible combinations of columns, and the level of interaction is governed by the elements of $\mathcal{G}$.

It is observed that if $r_j < n_j$ for at least one j, the size of the core tensor $\mathcal{G}$ will be smaller than the size of $\mathcal{X}$. This means that $\mathcal{G}$ can be thought of as a compressed version of $\mathcal{X}$.

3.3.1 j-rank

The column rank of $\mathbf{X}_{(j)}$ is defined as the j-rank of $\mathcal{X}$ and is denoted by $\text{rank}_j(\mathcal{X})$. If $r_j = \text{rank}_j(\mathcal{X})$ for $j \in [d]$, then $\mathcal{X}$ is said to have an exact rank-$(r_1, r_2, \ldots, r_d)$ Tucker decomposition. Obviously, $r_j \leq n_j$ and $\text{rank}_j(\mathcal{X}) \leq \min\{n_j, \prod_{\ell \neq j} n_\ell\}$. If $r_j \leq \text{rank}_j(\mathcal{X})$ for at least one j, then $\mathcal{X}$ cannot be reconstructed exactly from its Tucker representation.

3.3.2 Computing the Tucker Decomposition

In one of the first methods developed to compute the Tucker decomposition, the basic idea is to find those components (rank-1 tensors) that capture the most variations in each mode. This method is known as the Higher-order SVD (HoSVD) as a generalization of the matrix SVD, and computes the singular vectors of the mode-j unfoldings of a tensor $\mathcal{X}$. For $r_j \leq \text{rank}_j(\mathcal{X})$, the method is called truncated HoSVD. This method is not optimal, but it can be used as a good starting point in an ALS algorithm whose goal is to compute the Tucker decomposition.

Algorithm 3.2: HOOI-ALS [12]

initialize $\mathbf{A}^{(j)} \in \mathbb{R}^{n_j \times r_j}$ for $j \in [d]$ using HoSVD
repeat
  for $j = 1, \ldots, d$ **do**
    $\mathcal{Y} \leftarrow \mathcal{X} \times_1 \mathbf{A}^{(1)\top} \times \cdots \times_{j-1} \mathbf{A}^{(j-1)\top} \times_{j+1} \mathbf{A}^{(j+1)\top} \times_d \mathbf{A}^{(d)\top}$
    $\mathbf{A}^{(j)} \leftarrow r_j$ leading left singular vectors of $\mathbf{Y}_{(j)}$
  end for
until fit ceases to improve or maximum iterations exhausted
$\mathcal{G} \leftarrow \mathcal{X} \times_1 \mathbf{A}^{(1)\top} \cdots \times_d \mathbf{A}^{(d)\top}$
return $\mathcal{G}, \mathbf{A}^{(j)}$ for $j \in [d]$

To compute HoSVD, r_j leading left singular vectors of $\mathbf{X}_{(j)}$ is set as $\mathbf{A}^{(j)}$ for all $j \in [d]$. Then the core tensor is computed using

$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{A}^{(1)\top} \times \cdots \times_d \mathbf{A}^{(d)\top}.$$

The Higher-Order Orthogonal Iteration, abbreviated to HOOI, is used as the ALS method that takes the result of HoSVD as input. The optimization problem that is solved by HOOI is expressed as

$$\min_{\substack{\text{subject to } \mathcal{G} \in \mathbb{R}^{r_1 \times \cdots \times r_d} \\ \mathbf{A}^{(j)} \in \mathbb{R}^{n_j \times r_j} \text{ and column-wise orthogonal}}} \|\mathcal{X} - \mathcal{G} \times_1 \mathbf{A}^{(1)} \cdots \times_d \mathbf{A}^{(d)}\|. \tag{3.15}$$

The pseudo-code for HOOI is shown in Algorithm 3.2.

3.3.3 Uniqueness of Tucker

The Tucker decomposition is not unique. The core tensor can be modified as long as the inverse modification is applied to the factor matrices. For instance, consider the 3-mode case. For nonsingular matrices $\mathbf{U} \in \mathbb{R}^{n_1 \times n_1}$, $\mathbf{V} \in \mathbb{R}^{n_2 \times n_2}$ and $\mathbf{W} \in \mathbb{R}^{n_3 \times n_3}$, we have

$$\mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)} = (\mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}) \times_1 \left(\mathbf{A}^{(1)}\mathbf{U}^{-1}\right) \times_2 \left(\mathbf{A}^{(2)}\mathbf{V}^{-1}\right) \times_3 \left(\mathbf{A}^{(3)}\mathbf{W}^{-1}\right).$$

Proof *Let* $\mathcal{H} := (\mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}) \times_1 \left(\mathbf{A}^{(1)}\mathbf{U}^{-1}\right) \times_2 \left(\mathbf{A}^{(2)}\mathbf{V}^{-1}\right) \times_3 \left(\mathbf{A}^{(3)}\mathbf{W}^{-1}\right)$, *then we have*

$$\begin{aligned}
\mathbf{H}_{(1)} &= \mathbf{A}^{(1)}\mathbf{U}^{-1} \left(\mathcal{G} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\right)_{(1)} \left(\mathbf{A}^{(3)}\mathbf{W}^{-1} \otimes \mathbf{A}^{(2)}\mathbf{V}^{-1}\right)^\top \\
&= \mathbf{A}^{(1)}\mathbf{U}^{-1}\mathbf{U}\mathbf{G}_{(1)} \left(\mathbf{W} \otimes \mathbf{V}\right)^\top \left(\mathbf{A}^{(3)}\mathbf{W}^{-1} \otimes \mathbf{A}^{(2)}\mathbf{V}^{-1}\right)^\top \\
&= \mathbf{A}^{(1)}\mathbf{G}_{(1)} \left[\left(\mathbf{A}^{(3)}\mathbf{W}^{-1} \otimes \mathbf{A}^{(2)}\mathbf{V}^{-1}\right) \left(\mathbf{W} \otimes \mathbf{V}\right)\right]^\top \\
&= \mathbf{A}^{(1)}\mathbf{G}_{(1)} \left(\mathbf{A}^{(3)}\mathbf{W}^{-1}\mathbf{W} \otimes \mathbf{A}^{(2)}\mathbf{V}^{-1}\mathbf{V}\right)^\top = \mathbf{A}^{(1)}\mathbf{G}_{(1)} \left(\mathbf{A}^{(3)} \otimes \mathbf{A}^{(2)}\right)^\top \\
&= \left(\mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}\right)_{(1)},
\end{aligned} \tag{3.16}$$

implying the two tensors are equal. A similar approach can be applied to modes 2 *and* 3.

3.4 Tensor-Train Decomposition

The Tensor-Train decomposition, which is also known as MPS[2] in the physics community, decomposes a d-mode tensor into a chain of d lower-dimensional tensors of at most 3 modes [16]. A tensor $\mathcal{X}$ is approximated by another tensor $\hat{\mathcal{X}}$ whose elements are expressed as contractions of lower-dimensional tensors $\mathcal{G}^{(j)}$ for $j \in [d]$,

$$\hat{\mathcal{X}}_{i_1,\ldots,i_d} = \sum_{\alpha_0=1}^{r_0} \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_d=1}^{r_d} \mathcal{G}^{(1)}_{\alpha_0,i_1,\alpha_1} \mathcal{G}^{(2)}_{\alpha_1,i_2,\alpha_2} \cdots \mathcal{G}^{(d)}_{\alpha_{d-1},i_d,\alpha_d}, \tag{3.17}$$

where $\mathcal{G}^{(j)} \in \mathbb{R}^{r_{j-1} \times n_j \times r_j}$ for $j \in [d]$, and $r_0 = r_d = 1$, i.e., $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(d)}$ are in fact matrices. In other words, elements of $\hat{\mathcal{X}}$ can be obtained by employing

$$\hat{\mathcal{X}}_{i_1,\ldots,i_d} = \mathbf{G}^{(1)}\left(i_1\right) \mathbf{G}^{(2)}\left(i_2\right) \cdots \mathbf{G}^{(d)}\left(i_d\right), \tag{3.18}$$

where $\mathbf{G}^{(j)}\left(i_j\right) \in \mathbb{R}^{r_{j-1} \times r_j}$ for $j \in [d]$ is the i_j^{th} lateral slice of $\mathcal{G}^{(j)}$. The tensor-Train decomposition is calculated by a sequence of SVD's starting with $\mathbf{X}_{(1)}$. Assuming the Tensor-Train ranks r_j are known, a simplified version of the Tensor-Train algorithm is presented in Algorithm 3.3. For a more detailed version also including how to choose the ranks r_j, see [16].

[2]Matrix Product State

Algorithm 3.3: Tensor-Train [16]

Compute the SVD of $\mathbf{X}_{(1)}$: $\mathbf{X}_{(1)} = \mathbf{U}^{(1)}\mathbf{S}^{(1)}\mathbf{V}^{(1)\top} \in \mathbb{R}^{n_1 \times \prod_{\ell \neq 1} n_\ell}$.
Compute $\mathcal{G}^{(1)} = \mathbf{U}^{(1)}(:, 1 : r_1) \in \mathbb{R}^{n_1 \times r_1}$.
for $j = 2, \ldots, d-1$ **do**
  Compute $\mathbf{W}^{(j-1)} = \mathbf{S}^{(j-1)}\mathbf{V}^{(j-1)\top} \in \mathbb{R}^{r_{j-1} \times \prod_{\ell \notin [j-1]} n_\ell}$.
  Reshape $\mathbf{W}^{(j-1)}$ into $\mathbf{W}^{(j-1)} \in \mathbb{R}^{r_{j-1} n_j \times \prod_{\ell \notin [j]} n_\ell}$.
  Compute $\mathbf{W}^{(j-1)} = \mathbf{U}^{(j)}\mathbf{S}^{(j)}\mathbf{V}^{(j)\top}$.
  Truncate $\mathbf{U}^{(j)} \in \mathbb{R}^{r_{j-1} n_j \times r_{j-1} n_j}$ to get $\mathcal{G}^{(j)} = \mathbf{U}^{(j)}(:, 1 : r_j)$.
  Reshape $\mathcal{G}^{(j)}$ into $\mathcal{G}^{(j)} \in \mathbb{R}^{r_{j-1} \times n_j \times r_j}$.
end for
Compute $\mathcal{G}^{(d)} = \mathbf{W}^{(d-2)} = \mathbf{S}^{(d-1)}\mathbf{V}^{(d-1)\top} \in \mathbb{R}^{r_{d-1} \times n_d}$.
return $\mathcal{G}^{(j)}$ for $j \in [d]$.

Consider another definition of unfoldings of a tensor $\mathcal{X}$, defined by

$$\mathbf{X}_j = \mathcal{X}_{i_1,\ldots,i_j;i_{j+1},\ldots,i_d} \in \mathbb{R}^{\prod_{\ell=1}^{j} n_\ell \times \prod_{\ell=j+1}^{d} n_\ell}, \tag{3.19}$$

where the indices are divided into two row and column groups for $j \in [d]$. Obviously, $\mathbf{X}_1 = \mathbf{X}_{(1)}$, and $\mathbf{X}_d$ is the vectorized version of $\mathcal{X}$. The following theorems hold [16].

Theorem 3.4.1 *If rank*$(\mathbf{X}_j) = r_j$ *for each unfolding* $\mathbf{X}_j$ *of a tensor* $\mathcal{X}$ *for all* $j \in [d]$*, then there exists a decomposition* (3.17) *with Tensor-Train ranks no greater than* r_j*.*

Theorem 3.4.2 *Consider the* δ*-truncated SVD of* $\mathbf{X}_j$ *in the sense that*

$$\mathbf{X}_j = \mathbf{U}\mathbf{S}\mathbf{V}^\top + \mathbf{E}$$

for $\|\mathbf{E}\|_F \leq \delta$ *and* $r_k = rank_\delta(\mathbf{X}_j)$*, where* $rank_\delta(\mathbf{X}_j)$ *is the* δ*-rank of* $\mathbf{X}_j$ *defined as the minimum rank*$(\mathbf{B})$ *over all matrices* $\mathbf{B}$ *satisfying* $\|\mathbf{A} - \mathbf{B}\|_F \leq \delta$*. If*

$$\delta = \frac{\varepsilon}{\sqrt{d-1}}\|\mathcal{X}\|,$$

then

$$\|\mathcal{X} - \hat{\mathcal{X}}\| \leq \varepsilon\|\mathcal{X}\|.$$

Theorem 3.4.3 *Suppose that the unfolding matrices* $\mathbf{X}_j$ *have low ranks* r_j *only approximately, i.e.,* $\mathbf{X}_j = \mathbf{R}_j + \mathbf{E}_j$ *for all* $j \in [d]$*, such that* $rank(R_j) = r_j$ *and* $\|\mathbf{E}_j\|_F = \varepsilon_j$*. Then Algorithm 3.3 computes a tensor* $\hat{\mathcal{X}}$ *with Tensor-Train ranks* r_j *and* $\|\mathcal{X} - \hat{\mathcal{X}}\| \leq \sqrt{\sum_{j=1}^{d} \varepsilon_j^2}$.

Corollary 1 *If a tensor* $\mathcal{X}$ *admits a rank-r CP decomposition with accuracy* ε*, there exists a Tensor-Train decomposition with Tensor-Train ranks* $r_j \leq r$ *and accuracy* $\sqrt{d-1}\varepsilon$.

Corollary 2 *Given a tensor* $\mathcal{X}$ *and rank bounds* r_j*, the best approximation to* $\mathcal{X}$ *in the Euclidean norm with Tensor-Train ranks bounded by* r_j *always exists, and the Tensor-Train approximation* $\hat{\mathcal{X}}$ *is quasi-optimal. i.e.,* $\|\mathcal{X} - \hat{\mathcal{X}}\| \leq \sqrt{d-1}\|\mathcal{X} - \mathcal{X}^{\text{best}}\|$.

CHAPTER 4

DIMENSIONALITY REDUCTION OF TENSOR DATA: MODEWISE RANDOM PROJECTIONS

Johnson-Lindenstrauss embeddings, called JL from here on for the sake of brevity, provide a simple yet powerful tool for dimension reduction of high-dimensional data using linear random projections. By performing JL on mode-j fibers of a tensor $\mathcal{X}$, the dimentionality of all modes can be reduced to yield a projected tensor of much smaller size, without first vectorizing the tensor. It is then expected that the Euclidean norm of the projected tensor remains preserved to within a predictable error. In this chapter, theoretical guarantees for the geometry preserving properties of modewise random projections as JL embeddings are presented. More theorems, detailed discussions and proofs are provided in [9].

4.1 Johnson-Lindenstrauss Embeddings for Tensor Data

In this section, a brief overview of the necessary tools that will used to extend JL embeddings to higher-order data will be presented, as well as the theorems providing the underlying theory of modewie JL embeddings.

Definition 4.1.1 (ε-JL embedding) *A matrix $\mathbf{A} \in \mathbb{C}^{m\times N}$ is an ε-JL embedding of a set $S \subset \mathbb{C}^N$ into $\mathbb{C}^m$ if*

$$\|\mathbf{Ax}\|_2^2 = (1+\varepsilon_{\mathbf{x}})\,\|\mathbf{x}\|_2^2, \tag{4.1}$$

for $|\varepsilon_{\mathbf{x}}| \leq \varepsilon$ and all $\mathbf{x} \in S$.

Assuming the elements of $\mathbf{A}$ are subgaussian random variables and that $|S| = M$, then (4.1) holds for all $\mathbf{x} \in S$ with probability $p \geq 1 - 2\exp\left(-Cm\varepsilon^2\right)$ if $m \geq C\frac{\log M}{\varepsilon^2}$, where C is an absolute constant [25]. A brief statement of the JL lemma along with its relation with random projections is presented in Appendix A.3.

Lemma 4.1.1 *Let* $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$ *and suppose that* $\mathbf{A} \in \mathbb{C}^{m \times n}$ *is an* ε*-JL embedding of the vectors*

$$\{\mathbf{x} - \mathbf{y}, \mathbf{x} + \mathbf{y}, \mathbf{x} - \mathbb{i}\mathbf{y}, \mathbf{x} + \mathbb{i}\mathbf{y}\} \subset \mathbb{C}^n$$

into $\mathbb{C}^m$*. Then,*

$$|\langle \mathbf{Ax}, \mathbf{Ay} \rangle - \langle \mathbf{x}, \mathbf{y} \rangle| \;\leq\; 2\varepsilon \left(\|\mathbf{x}\|_2^2 + \|\mathbf{y}\|_2^2 \right) \;\leq\; 4\varepsilon \max \left\{ \|\mathbf{x}\|_2^2, \|\mathbf{y}\|_2^2 \right\}.$$

Proof *Using the polarization identity for inner products, we have that*

$$\begin{aligned}
|\langle \mathbf{Ax}, \mathbf{Ay} \rangle - \langle \mathbf{x}, \mathbf{y} \rangle| = \left| \frac{1}{4} \sum_{\ell=0}^{3} \mathbb{i}^\ell \left(\left\| \mathbf{Ax} + \mathbb{i}^\ell \mathbf{Ay} \right\|_2^2 - \left\| \mathbf{x} + \mathbb{i}^\ell \mathbf{y} \right\|_2^2 \right) \right| \;&=\; \left| \frac{1}{4} \sum_{\ell=0}^{3} \mathbb{i}^\ell \varepsilon_\ell \left\| \mathbf{x} + \mathbb{i}^\ell \mathbf{y} \right\|_2^2 \right| \\
&\leq \frac{1}{4} \sum_{\ell=0}^{3} \varepsilon \left(\|\mathbf{x}\|_2 + \|\mathbf{y}\|_2 \right)^2 \;=\; \varepsilon \left(\|\mathbf{x}\|_2 + \|\mathbf{y}\|_2 \right)^2 \\
&= \varepsilon \left(\|\mathbf{x}\|_2^2 + \|\mathbf{y}\|_2^2 + 2\|\mathbf{x}\|_2 \|\mathbf{y}\|_2 \right) \\
&\leq 2\varepsilon \left(\|\mathbf{x}\|_2^2 + \|\mathbf{y}\|_2^2 \right) \;\leq\; 4\varepsilon \max \left\{ \|\mathbf{x}\|_2^2, \|\mathbf{y}\|_2^2 \right\},
\end{aligned}$$

where the second to last inequality follows from Young's inequality for products. In the second equality, ε_ℓ *denotes the amount of distortion applied to* $\left\| \mathbf{x} + \mathbb{i}^\ell \mathbf{y} \right\|_2^2$ *by the JL matrix* $\mathbf{A}$*, where* $|\varepsilon_\ell| \leq \varepsilon$*.*

Extending vectors to tensors, one can define a tensor ε-JL embedding in a similar way as follows.

Definition 4.1.2 (Tensor ε-JL embedding) *A linear operator* $L : \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_d} \to \mathbb{C}^{m_1 \times \cdots \times m_{d'}}$ *is an* ε*-JL embedding of a set* $S \subset \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_d}$ *into* $\mathbb{C}^{m_1 \times \cdots \times m_{d'}}$ *if*

$$\|L(\mathcal{X})\|^2 = (1 + \varepsilon_{\mathcal{X}}) \|\mathcal{X}\|^2$$

holds for some $\varepsilon_{\mathcal{X}} \in (-\varepsilon, \varepsilon)$ *for all* $\mathcal{X} \in S$*.*

The following lemma shows that the Tensor ε-JL embedding will preserve the pairwise inner product of tensors.

Lemma 4.1.2 *If $\mathcal{X}, \mathcal{Y} \in \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_d}$ and suppose that L is an ε-JL embedding of the tensors*

$$\{\mathcal{X} - \mathcal{Y}, \mathcal{X} + \mathcal{Y}, \mathcal{X} - \mathbb{i}\mathcal{Y}, \mathcal{X} + \mathbb{i}\mathcal{Y}\} \subset \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_d}$$

into $\mathbb{C}^{m_1 \times \cdots \times m_{d'}}$. Then,

$$|\langle L(\mathcal{X}),\ L(\mathcal{Y})\rangle - \langle \mathcal{X},\ \mathcal{Y}\rangle| \ \leq\ 2\varepsilon\left(\|\mathcal{X}\|^2 + \|\mathcal{Y}\|^2\right) \ \leq\ 4\varepsilon \cdot \max\left\{\|\mathcal{X}\|^2, \|\mathcal{Y}\|^2\right\}.$$

Proof *The proof will be similar to what was presented in the proof of Lemma 4.1.1, and by replacing $\mathbf{A}x$ with $L(\mathcal{X})$ and using the linearity of L.*

When a more general set is being projected using JL embeddings, a discretization scheme can be used in order to embed a finite set (see Appendix A.2, and [25] for more details). This applies to, for instance, a low-rank subspace of tensors. In such cases, due to the linearity of the embedding, discretization can rather be done on the unit ball in that subspace[1]. In the following lemma, a JL embedding result for a subspace is presented based on a covering argument.

Lemma 4.1.3 *Fix $\varepsilon \in (0, 1)$. Let $\mathcal{L}$ be an r-dimensional subspace of $\mathbb{C}^n$, and let $C \subset \mathcal{L}$ be an $(\varepsilon/16)$-net of the $(r-1)$-dimensional Euclidean unit sphere $\mathcal{S}_{\ell^2} \subset \mathcal{L}$. Then, if $\mathbf{A} \in \mathbb{C}^{m \times n}$ is an $(\varepsilon/2)$-JL embedding of C, it will also satisfy*

$$(1-\varepsilon)\|\mathbf{x}\|_2^2 \leq \|\mathbf{A}\mathbf{x}\|_2^2 \leq (1+\varepsilon)\|\mathbf{x}\|_2^2, \tag{4.2}$$

for all $\mathbf{x} \in \mathcal{L}$. Furthermore, one can observe that $|C| \leq \left(\frac{47}{\varepsilon}\right)^r$.

[1] Any point in an r-dimensional subspace can be represented by a linear combination of r basis elements in the discretized unit ball of that subspace.

Proof *Noting that* $\mathbf{A}$ *is a linear embedding, it is enough to prove* (4.2) *for an arbitrary* $\mathbf{x} \in \mathcal{S}_{\ell^2}$ *as any point in the subspace* $\mathcal{L}$ *can be scaled up/down to a point on the unit sphere in* $\mathcal{L}$*. To prove the upper bound, let* $\Delta := \|\mathbf{A}\|_{2\to 2}$ *and choose an element* $\mathbf{y} \in C$ *such that* $\|\mathbf{x} - \mathbf{y}\|_2 \leq \varepsilon/16$*. Noting* $\|\mathbf{x}\|_2 = 1$*, we can write*

$$\begin{aligned}\|\mathbf{Ax}\|_2 - \|\mathbf{x}\|_2 &\leq \|\mathbf{Ay}\|_2 + \|\mathbf{A}(\mathbf{x}-\mathbf{y})\|_2 - 1 \leq \sqrt{1+\varepsilon/2} - 1 + \|\mathbf{A}(\mathbf{x}-\mathbf{y})\|_2 \\ &\leq \sqrt{1+\varepsilon/2} - 1 + \|\mathbf{A}\|_{2\to 2}\|\mathbf{x}-\mathbf{y}\|_2 \leq (1+\varepsilon/4) - 1 + \Delta\varepsilon/16 \\ &= (\varepsilon/4)(1+\Delta/4)\end{aligned}$$

for all $\mathbf{x} \in \mathcal{S}_{\ell^2}$*. Since this upper bound holds for all* $\mathbf{x}$ *with* $\|\mathbf{x}\|_2 = 1$*, it will also hold for the maximizer of* $\|\mathbf{Ax}\|_2$ *with* $\|\mathbf{x}\|_2 = 1$*, meaning for that* $\mathbf{x}$*,* $\|\mathbf{Ax}\|_2 = \|\mathbf{A}\|_{2\to 2}$ *so that* $\Delta - 1 \leq (\varepsilon/4)(1+\Delta/4)$ *must also hold. Therefore,* $\Delta \leq 1 + \varepsilon/4 + \Delta\varepsilon/16 \implies \Delta \leq \frac{1+\varepsilon/4}{1-\varepsilon/16} \leq 1 + \varepsilon/3$*. The upper bound now follows as* $\|\mathbf{Ax}\|_2 \leq \Delta = \sup_{\mathbf{z}\in\mathcal{S}_{\ell^2}} \|\mathbf{Az}\|_2$ *for all* $\mathbf{x} \in \mathcal{S}_{\ell^2}$*.*

For the lower bound, let $\delta := \inf_{\mathbf{z}\in\mathcal{S}_{\ell^2}} \|\mathbf{Az}\|_2 \geq 0$*, and we also note that there exists an element of the compact set* $\mathcal{S}_{\ell^2}$ *that realizes* δ*. Similar to the proof of the upper bound, we consider this minimizing vector* $\mathbf{x} \in \mathcal{S}_{\ell^2}$ *and choose an element* $\mathbf{y} \in C$ *with* $\|\mathbf{x} - \mathbf{y}\|_2 \leq \varepsilon/16$ *in order to observe that*

$$\begin{aligned}\delta - 1 = \|\mathbf{Ax}\|_2 - \|\mathbf{x}\|_2 &\geq \|\mathbf{Ay}\|_2 - \|\mathbf{A}(\mathbf{x}-\mathbf{y})\|_2 - 1 \geq \sqrt{1-\varepsilon/2} - 1 - \|\mathbf{A}(\mathbf{x}-\mathbf{y})\|_2 \\ &\geq \sqrt{1-\varepsilon/2} - 1 - \|\mathbf{A}\|_{2\to 2}\|\mathbf{x}-\mathbf{y}\|_2 \geq (1-\varepsilon/3) - 1 - \Delta\varepsilon/16 \\ &\geq -(\varepsilon/3 + \varepsilon/16\,(1+\varepsilon/3)) \geq -(\varepsilon/3 + \varepsilon/16 + \varepsilon/48) = -5\varepsilon/12.\end{aligned}$$

Thus, $\delta \geq 1 - 5\varepsilon/12 \geq 1 - \varepsilon$*. This proves the lower bound as* $\|\mathbf{Ax}\|_2 \geq \delta$ *for all* $\mathbf{x} \in \mathcal{S}_{\ell^2}$*. The proof of the upper bound on* $|C|$ *can be found in Appendix C of [5].*

Note 4.1.1 *According to the lower and upper bounds proved above,* (4.2) *can use a tighter bound, i.e.,*

$$(1-\varepsilon/2)\|\mathbf{x}\|_2^2 \leq \|\mathbf{Ax}\|_2^2 \leq (1+\varepsilon/2)\|\mathbf{x}\|_2^2,$$

as $\delta \geq 1 - 5\varepsilon/12 \geq 1 - \varepsilon/2$ *and* $\Delta \leq 1 + \varepsilon/3 \leq 1 + \varepsilon/2$. *This means the* $(\varepsilon/2)$*-JL property of* $\mathbf{A}$ *assumed to hold for the elements of* C *is carried over to all elements of the subspace.*

In Lemma 4.1.3, the norms of vectors in a subspace are preserved by preserving the norms of all points in the discretized unit ball in that subspace. This makes the dependence on the subspace dimension r exponential according to $|C| \leq (47/\varepsilon)^r$. The following lemma uses a coarser discretization to improve the dependence on r so that a better target dimension can be achieved for the JL embedding. This is done by preserving the norms of an orthonormal basis to, by linearity, control the norms of all points in the subspace. If the angles between the elements of the orthonormal basis are preserved very accurately, then the projected basis will also be *approximately* orthonormal, and the norms of the points that are in the span of the orthonormal basis will also be preserved. Requiring the preservation of the aformentioned angles to be accurate imposes, in turn, a more strict bound on the norm-preserving property of the embedding. This concept is presented in the following lemma.

Lemma 4.1.4 *Fix* $\varepsilon \in (0, 1)$ *and let* $\mathcal{L}$ *be an* r*-dimensional subspace of* $\mathbb{C}^{n_1 \times \cdots \times n_d}$ *spanned by a set of* r *orthonormal basis tensors* $\{\mathcal{T}_k\}_{k \in [r]}$. *If* L *is an* $(\varepsilon/4r)$*-JL embedding of the* $4\binom{r}{2} + r = 2r^2 - r$ *tensors*

$$\left(\bigcup_{1 \leq h < k \leq r} \{\mathcal{T}_k - \mathcal{T}_h, \mathcal{T}_k + \mathcal{T}_h, \mathcal{T}_k - \mathbb{i}\mathcal{T}_h, \mathcal{T}_k + \mathbb{i}\mathcal{T}_h\} \right) \bigcup \{\mathcal{T}_k\}_{k \in [r]} \subset \mathcal{L}$$

into $\mathbb{C}^{m_1 \times \cdots \times m_{d'}}$, *then*

$$\left| \|L(\mathcal{X})\|^2 - \|\mathcal{X}\|^2 \right| \leq \varepsilon \|\mathcal{X}\|^2$$

holds for all $\mathcal{X} \in \mathcal{L}$.

Proof *According to Lemma 4.1.2, one can see that* $|\varepsilon_{k,h}| := |\langle L(\mathcal{T}_k), L(\mathcal{T}_h) \rangle - \langle \mathcal{T}_k, \mathcal{T}_h \rangle| \leq \varepsilon/r$

for all $h, k \in [r]$. *Thus, for any* $\mathcal{X} = \sum_{k=1}^{r} \alpha_k \mathcal{T}_k \in \mathcal{L}$,

$$\begin{aligned}\left|\|L(\mathcal{X})\|^2 - \|\mathcal{X}\|^2\right| &= \left|\sum_{k=1}^{r}\sum_{h=1}^{r} \alpha_k \overline{\alpha_h}\left(\langle L(\mathcal{T}_k),\, L(\mathcal{T}_h)\rangle - \langle \mathcal{T}_k,\, \mathcal{T}_h\rangle\right)\right| = \left|\sum_{k=1}^{r}\sum_{h=1}^{r} \alpha_k \overline{\alpha_h}\varepsilon_{k,h}\right| \\ &\leq \sum_{k=1}^{r}|\alpha_k| \sum_{h=1}^{r}|\alpha_h|\,|\varepsilon_{k,h}| \leq \frac{\varepsilon}{r}\|\boldsymbol{\alpha}\|_1^2 \leq \varepsilon\|\boldsymbol{\alpha}\|_2^2,\end{aligned}$$

where we have used the relation $\|\boldsymbol{\alpha}\|_1 \leq \sqrt{r}\,\|\boldsymbol{\alpha}\|_2$ *to obtain the last inequality*[2]. *To finish the proof, we must show that* $\|\mathcal{X}\|^2 = \|\boldsymbol{\alpha}\|_2^2$. *Due to the orthonormality of the basis tensors* $\{\mathcal{T}_k\}_{k\in[r]}$, *one may write*

$$\|\mathcal{X}\|^2 = \langle \mathcal{X}, \mathcal{X}\rangle = \sum_{k=1}^{r}\sum_{h=1}^{r} \alpha_k \overline{\alpha_h}\,\langle \mathcal{T}_k, \mathcal{T}_h\rangle = \sum_{k=1}^{r}|\alpha|^2 = \|\boldsymbol{\alpha}\|^2.$$

4.2 Johnson-Lindenstrauss Embedings for Low-Rank Tensors

In this section, JL embeddings are discussed along the lines of Lemmas 4.1.3 and 4.1.4 in the case of low-CP-rank tensors, i.e., tensors that can be expressed as the weighted sum of a number of rank-1 tensors.

4.2.1 Geometry-Preserving Property of JL Embeddings for Low-Rank Tensors

The main purpose of this section is to show how employing modewise JL embeddings affect the norm of a low-rank tensor and its inner product with another low-rank tensor. Considering rank-r tensors as members of an r-dimensional tensor subspace spanned by r rank-1 basis tensors, we are essentially assuming that the these basis tensors always exist. This, however, is not guaranteed, and we are not even able to guarantee that there always exist a sufficiently incoherent basis of r rank-1 tensors that span any rank-r subspace. To incorporate coherence into our analysis, the concepts of modewise and basis coherence are introduced in the following. Next, the norm and

[2]Using the Cauchy-Schwarz ineqquality for vectors $\boldsymbol{\alpha}$ and $\mathbf{1}$, we have $\|\boldsymbol{\alpha}\|_1 = \langle|\boldsymbol{\alpha}|, \mathbf{1}\rangle \leq \|\boldsymbol{\alpha}\|_2\|\mathbf{1}\|_2 = \sqrt{r}\|\boldsymbol{\alpha}\|_2$, where $|\boldsymbol{\alpha}|$ is a vector whose elements are the absolute values of the elements of $\boldsymbol{\alpha}$, and $\mathbf{1}$ is a vector of all ones.

inner product preservation property in the Johnson-Lindenstrauss embeddings of low-rank tensors will be discussed.

Definition 4.2.1 (Modewise Coherence) *Assume that $\mathcal{X}$ admits a decomposition of rank r in the "standard" form, i.e.,*

$$\mathcal{X} = \sum_{k=1}^{r} \alpha_k \, \mathbf{x}_k^{(1)} \bigcirc \mathbf{x}_k^{(2)} \bigcirc \cdots \bigcirc \mathbf{x}_k^{(d)} = \sum_{k=1}^{r} \alpha_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)}, \tag{4.3}$$

where $\|\mathbf{x}_k^{(\ell)}\|_2^2 = 1$ for $k \in [r]$ and $\ell \in [d]$. The maximum modewise coherence of $\mathcal{X}$ is defined as

$$\mu_{\mathcal{X}} := \max_{\ell \in [d]} \mu_{\mathcal{X},\ell} := \max_{\ell \in [d]} \max_{\substack{k,h \in [r] \\ k \neq h}} \left| \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \right|. \tag{4.4}$$

where $\mu_{\mathcal{X},\ell} \in [0,1]$ for $\ell \in [d]$ is the modewise coherence for mode ℓ. Also obviously, $\mu_{\mathcal{X}} \in [0,1]$.

Definition 4.2.2 (Basis Coherence) *Let $\mathcal{B}$ be a set of r rank-1 tensors, defined as*

$$\mathcal{B} := \left\{ \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \;\middle|\; k \in [r] \right\} \subset \mathbb{C}^{n_1 \times \cdots \times n_d}$$

with $\|\mathbf{x}_k^{(\ell)}\|_2 = 1$ for $k \in [r]$ and $\ell \in [d]$. Let $\mathcal{L} := \text{span}\left(\left\{ \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \;\middle|\; k \in [r] \right\} \right)$ be the span of $\mathcal{B}$. For the set $\mathcal{B}$, the basis coherence is defined as

$$\mu'_{\mathcal{B}} := \max_{\substack{k,h \in [r] \\ k \neq h}} \left\langle \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)}, \bigcirc_{\ell=1}^{d} \mathbf{x}_h^{(\ell)} \right\rangle = \max_{\substack{k,h \in [r] \\ k \neq h}} \prod_{\ell=1}^{d} \left| \left\langle \mathbf{x}_k^{(\ell)}, \, \mathbf{x}_h^{(\ell)} \right\rangle \right|. \tag{4.5}$$

It is easy to verify that $\mu'_{\mathcal{B}} \in [0,1]$.

Note 4.2.1 *Looking at* (4.5)*, one can observe that $\mu'_{\mathcal{B}}$ is in fact the maximum absolute inner product[3] of the pairs of rank-1 tensors that form the basis $\mathcal{B}$, hence the name basis coherence.*

We will also use, with some abuse of notation, the (maximum) modewise coherence and basis coherence for any $\mathcal{X} \in \mathcal{L} = \text{span}(\mathcal{B})$ and the basis $\mathcal{B}$ interchangeably, i.e., $\mu_{\mathcal{X},\ell} = \mu_{\mathcal{B},\ell}$, $\mu_{\mathcal{X}} = \mu_{\mathcal{B}}$, and $\mu'_{\mathcal{X}} = \mu'_{\mathcal{B}}$. Finally, it can also be inferred that

$$\mu'_{\mathcal{X}} \leq \prod_{\ell=1}^{d} \mu_{\mathcal{X},\ell} \leq \mu_{\mathcal{X}}^{d}.$$

[3]Defined as per (2.3), or equivalently, the inner product of the vectorized form of the tensors.

It should also be noted that $\mu_{\mathcal{X},\ell}$, $\mu_{\mathcal{X}}$, and $\mu_{\mathcal{X}}^{'}$ always depend on the choice of the particular basis tensors forming $\mathcal{B}$.

To further prepare grounds for the main JL embedding result, we discuss how modewise JL embeddings affect the structure, modewise coherence, and the norm of a low-rank tensor of the form (4.3), as well as the inner product of two such low-rank tensors. This will be done through the next few lemmas and theorems below.

Lemma 4.2.1 *Let $j \in [d]$, $\mathbf{A} \in \mathbb{C}^{m \times n_j}$, and $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$ be a rank-r tensor as per* (4.3) *such that* $\min_{k \in [r]} \left\| \mathbf{A}\mathbf{x}_k^{(j)} \right\|_2 > 0$. *Then $\mathcal{X}' := \mathcal{X} \times_j \mathbf{A}$ can be written in standard form as*

$$\mathcal{X}' = \sum_{k=1}^{r} \alpha_k \left\| \mathbf{A}\mathbf{x}_k^{(j)} \right\|_2 \left(\left(\bigcirc_{\ell=1}^{j-1} \mathbf{x}_k^{(\ell)} \right) \bigcirc \frac{\mathbf{A}\mathbf{x}_k^{(j)}}{\left\| \mathbf{A}\mathbf{x}_k^{(j)} \right\|_2} \bigcirc \left(\bigcirc_{\ell=j+1}^{d} \mathbf{x}_k^{(\ell)} \right) \right).$$

Furthermore, the modewise coherence of $\mathcal{X}'$ as above will satisfy

$$\mu_{\mathcal{X}',j} \;=\; \max_{\substack{k,h \in [r] \\ k \neq h}} \frac{\left| \left\langle \mathbf{A}\mathbf{x}_k^{(j)}, \mathbf{A}\mathbf{x}_h^{(j)} \right\rangle \right|}{\left\| \mathbf{A}\mathbf{x}_k^{(j)} \right\|_2 \left\| \mathbf{A}\mathbf{x}_h^{(j)} \right\|_2}$$

so that

$$\mu_{\mathcal{X}'} = \;\max \left(\mu_{\mathcal{X}',j}, \max_{\ell \in [d] \setminus \{j\}} \max_{\substack{k,h \in [r] \\ k \neq h}} \left| \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \right| \right).$$

Proof *Using Lemma 2.0.2, the linearity of tensor matricization, and* (2.8) *we can see that the mode-j unfolding of $\mathcal{X}'$ satisfies*

$$\begin{aligned} \mathbf{X}'_{(j)} \;=\; \mathbf{A}\mathbf{X}_{(j)} \;=\; \mathbf{A} \sum_{k=1}^{r} \alpha_k \left(\bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \right)_{(j)} \;=\; \sum_{k=1}^{r} \alpha_k \mathbf{A}\mathbf{x}_k^{(j)} \left(\otimes_{\ell \neq j} \mathbf{x}_k^{(\ell)} \right)^{\top} \\ = \;\sum_{k=1}^{r} \left(\alpha_k \left\| \mathbf{A}\mathbf{x}_k^{(j)} \right\|_2 \right) \frac{\mathbf{A}\mathbf{x}_k^{(j)}}{\left\| \mathbf{A}\mathbf{x}_k^{(j)} \right\|_2} \left(\otimes_{\ell \neq j} \mathbf{x}_k^{(\ell)} \right)^{\top}. \end{aligned}$$

where $\otimes_{\ell \neq j} \mathbf{x}_k^{(\ell)} = \mathbf{x}_k^{(d)} \otimes \cdots \otimes \mathbf{x}_k^{(j+1)} \otimes \mathbf{x}_k^{(j-1)} \otimes \cdots \otimes \mathbf{x}_k^{(1)}$. *Folding* $\mathbf{X}'_{(j)}$ *back into a* d*-mode tensor then gives us our first equality. The second two equalities follow directly from the definition of modewise coherence.*

The following lemma will be helpful in proving the norm-preserving property of modewise JL embeddings; it provides a useful relation expressing the norm of the j-mode product of a tensor that is in the standard form (4.3) in terms of inner products of its individual factor vectors projected by the embedding.

Lemma 4.2.2 *Let* $j \in [d]$, $\mathbf{A} \in \mathbb{C}^{m \times n_j}$, *and* $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$ *be a rank-*r *tensor in standard form as per* (4.3)*. Then,*

$$\|\mathcal{X} \times_j \mathbf{A}\|^2 \;=\; \sum_{k,h=1}^{r} \sum_{a=1}^{\prod_{\ell \neq j} n_\ell} \alpha_k \left(\otimes_{\ell \neq j} \mathbf{x}_k^{(\ell)}\right)_a \overline{\alpha_h \left(\otimes_{\ell \neq j} \mathbf{x}_h^{(\ell)}\right)_a} \left\langle \mathbf{A}\mathbf{x}_k^{(j)}, \mathbf{A}\mathbf{x}_h^{(j)} \right\rangle .$$

where $(\cdot)_a$ *denotes the* a^{th} *element of a vector.*

Proof *Using the linearity of* j*-mode products, tensor matricization, and observing that the Euclidean norm of a tensor is equal to the Frobenius norm of any of its unfoldings, one can write*

$$\begin{aligned}
\|\mathcal{X} \times_j \mathbf{A}\|^2 &= \left\| \sum_{k=1}^{r} \alpha_k \left(\left(\bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \right) \times_j \mathbf{A} \right) \right\|^2 = \left\| \sum_{k=1}^{r} \alpha_k \left(\left(\bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \right) \times_j \mathbf{A} \right)_{(j)} \right\|_{\mathrm{F}}^2 \\
&= \left\| \sum_{k=1}^{r} \alpha_k \mathbf{A}\mathbf{x}_k^{(j)} \left(\otimes_{\ell \neq j} \mathbf{x}_k^{(\ell)} \right)^\top \right\|_{\mathrm{F}}^2 \\
&= \sum_{k,h=1}^{r} \left\langle \alpha_k \mathbf{A}\mathbf{x}_k^{(j)} \left(\otimes_{\ell \neq j} \mathbf{x}_k^{(\ell)} \right)^\top, \alpha_h \mathbf{A}\mathbf{x}_h^{(j)} \left(\otimes_{\ell \neq j} \mathbf{x}_h^{(\ell)} \right)^\top \right\rangle_{\mathrm{F}}
\end{aligned}$$

where $\|\cdot\|_{\mathrm{F}}$ *and* $\langle \cdot, \cdot \rangle_{\mathrm{F}}$ *denote the Frobenius matrix norm and inner product, respectively. Computing the Frobenius inner products above columnwise, and noting that there are* $\prod_{\ell \neq j} n_\ell$ *columns in the mode-*j *unfolding, one can further see that*

$$\|\mathcal{X} \times_j \mathbf{A}\|^2 \;=\; \sum_{k,h=1}^{r} \sum_{a=1}^{\prod_{\ell \neq j} n_\ell} \alpha_k \left(\otimes_{\ell \neq j} \mathbf{x}_k^{(\ell)}\right)_a \overline{\alpha_h \left(\otimes_{\ell \neq j} \mathbf{x}_h^{(\ell)}\right)_a} \left\langle \mathbf{A}\mathbf{x}_k^{(j)}, \mathbf{A}\mathbf{x}_h^{(j)} \right\rangle ,$$

completing the proof.

The following theorem demonstrates that a single modewise JL embedding of any low-rank tensor $\mathcal{X}$ of the standard form (4.3) will preserve its norm up to an error depending on the overall ℓ^2-norm of its coefficients $\boldsymbol{\alpha} \in \mathbb{C}^r$. It should also be noted that in establishing proofs for the following lemmas and theorems, we will encounter situations where applying the polarization identity for inner products (recall how it was used in the proofs of Lemmas 4.1.1 and 4.1.2) will lead to the requirement that the JL property of matrices involved must hold for a set of vectors and combinations of them. For low-rank tensors under consideration, this set will include the vectors forming (4.3), and the corresponding set of interest will be

$$\mathcal{S}'_j = \left(\bigcup_{1 \le h < k \le r} \left\{ \mathbf{x}_k^{(j)} - \mathbf{x}_h^{(j)}, \mathbf{x}_k^{(j)} + \mathbf{x}_h^{(j)}, \mathbf{x}_k^{(j)} - \mathbb{i}\mathbf{x}_h^{(j)}, \mathbf{x}_k^{(j)} + \mathbb{i}\mathbf{x}_h^{(j)} \right\} \right) \bigcup \left\{ \mathbf{x}_k^{(j)} \right\}_{k \in [r]} \subset \mathbb{C}^{n_j}, \quad (4.6)$$

containing $4\binom{r}{2} + r = 2r^2 - r$ vectors for each mode $j \in [d]$.

Theorem 4.2.1 *Let $j \in [d]$ and $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$ be a rank-r tensor as per* (4.3)*. Suppose that $\mathbf{A} \in \mathbb{C}^{m \times n_j}$ is an $(\varepsilon/4)$-JL embedding of the vectors in the set defined in* (4.6) *into $\mathbb{C}^m$. Let $\mathcal{X}' := \mathcal{X} \times_j \mathbf{A}$ and rewrite it in standard form so that*

$$\mathcal{X}' = \sum_{k=1}^{r} \alpha'_k \left(\left(\bigcirc_{\ell<j} \mathbf{x}_k^{(\ell)} \right) \bigcirc \frac{\mathbf{A}\mathbf{x}_k^{(j)}}{\left\| \mathbf{A}\mathbf{x}_k^{(j)} \right\|_2} \bigcirc \left(\bigcirc_{\ell>j} \mathbf{x}_k^{(\ell)} \right) \right).$$

Then all of the following hold:

(a) $\left| \alpha'_k - \alpha_k \right| \le \varepsilon |\alpha_k|/4$ *for all* $k \in [r]$ *so that* $\|\boldsymbol{\alpha}'\|_\infty \le (1 + \varepsilon/4)\|\boldsymbol{\alpha}\|_\infty$

(b) $\mu_{\mathcal{X}',j} \le \frac{\mu_{\mathcal{X},j} + \varepsilon}{1 - \varepsilon/4}$, *and* $\mu_{\mathcal{X}',\ell} = \mu_{\mathcal{X},\ell}$ *for all* $\ell \in [d] \setminus \{j\}$

(c) $\left| \|\mathcal{X}'\|^2 - \|\mathcal{X}\|^2 \right| \le \varepsilon \left(1 + \sqrt{r(r-1)} \prod_{\ell \ne j} \mu_{\mathcal{X},\ell} \right) \|\boldsymbol{\alpha}\|_2^2 \le \varepsilon \left(1 + r\mu_{\mathcal{X}}^{d-1} \right) \|\boldsymbol{\alpha}\|_2^2 \le \varepsilon(r+1)\|\boldsymbol{\alpha}\|_2^2$

Proof *Properties are proved in order below.*

(a) By Lemma 4.2.1, one can write for all $k \in [r]$ *that*

$$\left|\alpha'_k - \alpha_k\right| \;=\; \left|\alpha_k \left\|\mathbf{A}\mathbf{x}_k^{(j)}\right\|_2 - \alpha_k\right| \;=\; |\alpha_k| \left|\left\|\mathbf{A}\mathbf{x}_k^{(j)}\right\|_2 - 1\right| \;\leq\; \varepsilon|\alpha_k|/4.$$

(b) Lemma 4.2.1 and the definition of j*-mode coherence lead to*

$$\mu_{\mathcal{X}',j} = \max_{\substack{k,h\in[r]\\ k\neq h}} \frac{\left|\left\langle \mathbf{A}\mathbf{x}_k^{(j)}, \mathbf{A}\mathbf{x}_h^{(j)}\right\rangle\right|}{\left\|\mathbf{A}\mathbf{x}_k^{(j)}\right\|_2 \left\|\mathbf{A}\mathbf{x}_h^{(j)}\right\|_2} \;\leq\; \max_{\substack{k,h\in[r]\\ k\neq h}} \frac{\left|\left\langle \mathbf{x}_k^{(j)}, \mathbf{x}_h^{(j)}\right\rangle\right| + \varepsilon}{1-\frac{\varepsilon}{4}} = \frac{\mu_{\mathcal{X},j}+\varepsilon}{1-\frac{\varepsilon}{4}},$$

where the inequality follows from $\mathbf{A}$ *being an* $(\varepsilon/4)$*-JL embedding and Lemma 4.1.1.*

(c) Applying Lemma 4.2.2 one can observe that

$$\|\mathcal{X}'\|^2 - \|\mathcal{X}\|^2 = \sum_{k,h=1}^{r} \sum_{a=1}^{\prod_{\ell\neq j} n_\ell} \alpha_k \left(\otimes_{\ell\neq j}\mathbf{x}_k^{(\ell)}\right)_a \overline{\alpha_h \left(\otimes_{\ell\neq j}\mathbf{x}_h^{(\ell)}\right)_a} \left(\left\langle \mathbf{A}\mathbf{x}_k^{(j)}, \mathbf{A}\mathbf{x}_h^{(j)}\right\rangle - \left\langle \mathbf{x}_k^{(j)}, \mathbf{x}_h^{(j)}\right\rangle\right). \quad (4.7)$$

Lemma 4.1.1 can be applied to each inner product in (4.7) *to get*

$$\left\langle \mathbf{A}\mathbf{x}_k^{(j)}, \mathbf{A}\mathbf{x}_h^{(j)}\right\rangle \;=\; \left\langle \mathbf{x}_k^{(j)}, \mathbf{x}_h^{(j)}\right\rangle + \varepsilon_{k,h}$$

for some $\varepsilon_{k,h} \in \mathbb{C}$ *with* $\left|\varepsilon_{k,h}\right| \leq \varepsilon$*. As a result we have that*

$$\begin{aligned}
\left|\|\mathcal{X}'\|^2 - \|\mathcal{X}\|^2\right| \;&=\; \left|\sum_{k,h=1}^{r} \sum_{a=1}^{\prod_{\ell\neq j} n_\ell} \alpha_k \left(\otimes_{\ell\neq j}\mathbf{x}_k^{(\ell)}\right)_a \overline{\alpha_h \left(\otimes_{\ell\neq j}\mathbf{x}_h^{(\ell)}\right)_a} \varepsilon_{k,h}\right|. \\
&=\; \left|\sum_{k,h=1}^{r} \alpha_k\overline{\alpha_h}\varepsilon_{k,h} \sum_{a=1}^{\prod_{\ell\neq j} n_\ell} \left(\otimes_{\ell\neq j}\mathbf{x}_k^{(\ell)}\right)_a \overline{\left(\otimes_{\ell\neq j}\mathbf{x}_h^{(\ell)}\right)_a}\right| \\
&=\; \left|\sum_{k,h=1}^{r} \alpha_k\overline{\alpha_h}\varepsilon_{k,h} \left\langle \bigcirc_{\ell\neq j}\mathbf{x}_k^{(\ell)}, \bigcirc_{\ell\neq j}\mathbf{x}_h^{(\ell)}\right\rangle\right| = \left|\sum_{k,h=1}^{r} \alpha_k\overline{\alpha_h}\varepsilon_{k,h} \prod_{\ell\neq j} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)}\right\rangle\right| \\
&\leq \left|\sum_{k=1}^{r} |\alpha_k|^2\varepsilon_{k,k} \prod_{\ell\neq j} \left\|\mathbf{x}_k^{(\ell)}\right\|^2\right| + \left|\sum_{k\neq h} \alpha_k\overline{\alpha_h}\varepsilon_{k,h} \prod_{\ell\neq j} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)}\right\rangle\right|.
\end{aligned}$$

Noting that $\prod_{\ell \neq j} \left\| \mathbf{x}_k^{(\ell)} \right\|^2 = 1$ *as* $\left\| \mathbf{x}_k^{(\ell)} \right\|_2 = 1$ *for all* $\ell \in [d]$ *and* $k \in [r]$*, it follows that*

$$\begin{aligned} \left| \|\mathcal{X}'\|^2 - \|\mathcal{X}\|^2 \right| &\leq \varepsilon \left| \sum_{k=1}^{r} |\alpha_k|^2 \right| + \left| \sum_{k \neq h} \alpha_k \overline{\alpha_h} \varepsilon_{k,h} \prod_{\ell \neq j} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \right| \\ &= \varepsilon \|\boldsymbol{\alpha}\|_2^2 + \left| \left\langle \mathbf{E}^\top \boldsymbol{\alpha}, \boldsymbol{\alpha} \right\rangle \right| \leq \left(\varepsilon + \left\| \mathbf{E}^\top \right\|_{2 \to 2} \right) \|\boldsymbol{\alpha}\|_2^2, \end{aligned}$$

where $\mathbf{E} \in \mathbb{C}^{r \times r}$ *is zero on its diagonal,* $E_{k,h} = \varepsilon_{k,h} \prod\limits_{\ell \neq j} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle$ *for* $k \neq h$*, and the operator norm* $\|\mathbf{E}^\top\|_{2 \to 2}$ *satisfies*

$$\left\| \mathbf{E}^\top \right\|_{2 \to 2} \leq \|\mathbf{E}\|_F \leq \sqrt{\sum_{k \neq h} \left| \prod_{\ell \neq j} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \right|^2 \varepsilon^2} = \varepsilon \cdot \sqrt{\sum_{k \neq h} \left| \prod_{\ell \neq j} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \right|^2}.$$

Finally, the definition of $\mu_{\mathcal{X}}$ *implies that*

$$\|\mathbf{E}\|_{2 \to 2} \leq \varepsilon \sqrt{r(r-1)} \prod_{\ell \neq j} \mu_{\mathcal{X},\ell} \leq \varepsilon r \mu_{\mathcal{X}}^{d-1}.$$

Thus, the desired bound can be obtained, i.e.,

$$\left| \|\mathcal{X}'\|^2 - \|\mathcal{X}\|^2 \right| \leq \varepsilon \left(1 + \sqrt{r(r-1)} \prod_{\ell \neq j} \mu_{\mathcal{X},\ell} \right) \|\boldsymbol{\alpha}\|_2^2 \leq \varepsilon \left(1 + r \mu_{\mathcal{X}}^{d-1} \right) \|\boldsymbol{\alpha}\|_2^2.$$

Theorem 4.2.2 provides an upper bound for the distortion in the norm of a tensor when modewise JL embeddings are applied to all its modes. The following remark provides a useful tool in the proof of the theorem.

Remark 4.2.1 *Let* $c, d \in \mathbb{R}^+$*. Then,* $\left(1 + \frac{c}{d}\right)^d \leq \mathrm{e}^c$.

Theorem 4.2.2 *Let* $\varepsilon \in (0, 3/4]$*. Assume that* $\mathcal{X}$ *admits a decomposition of rank* r *in the standard form as per* (4.3)*. Also, assume that the matrices* $\mathbf{A}^{(j)} \in \mathbb{C}^{m_j \times n_j}$ *are* $\left(\frac{\varepsilon}{4d}\right)$*-JL embeddings of the vectors in* $\mathcal{S}_j'$ *as per* (4.6) *into* $\mathbb{C}^{m_j}$ *for each* $j \in [d]$*. If*

$$\mathcal{Y} = \mathcal{X} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times \cdots \times_d \mathbf{A}^{(d)}, \tag{4.8}$$

then,

$$\begin{aligned}\left|\|\mathcal{Y}\|^2 - \|\mathcal{X}\|^2\right| &\leq \varepsilon\left(\mathbb{e} + \mathbb{e}^2\sqrt{r\,(r-1)}\max\left(\varepsilon^{d-1}, \mu_{\mathcal{X}}^{d-1}\right)\right)\|\boldsymbol{\alpha}\|_2^2 \\ &\leq \varepsilon\mathbb{e}^2\,(r+1)\,\|\boldsymbol{\alpha}\|_2^2,\end{aligned} \tag{4.9}$$

and if the maximum modewise coherence of $\mathcal{X}$ is zero, i.e., if $\mu_{\mathcal{X}} = 0$, then

$$\left|\|\mathcal{Y}\|^2 - \|\mathcal{X}\|^2\right| \leq \left(\varepsilon + \mathbb{e}\sqrt{r\,(r-1)}\varepsilon^d\right)\mathbb{e}\|\boldsymbol{\alpha}\|_2^2. \tag{4.10}$$

Proof *Let $\mathcal{X}^{(0)} := \mathcal{X}$ and $\mathcal{X}^{(d)} := \mathcal{Y}$, and for each $j \in [d]$ define the partially compressed tensor*

$$\begin{aligned}\mathcal{X}^{(j)} &:= \mathcal{X} \times_1 \mathbf{A}^{(1)} \cdots \times_j \mathbf{A}^{(j)} = \sum_{k=1}^{r} \alpha_k \left(\bigcirc_{\ell=1}^{j} \mathbf{A}^{(\ell)}\mathbf{x}_k^{(\ell)}\right) \bigcirc \left(\bigcirc_{\ell=j+1}^{d} \mathbf{x}_k^{(\ell)}\right) \\ &=: \sum_{k=1}^{r} \alpha_{j,k} \bigcirc_{\ell=1}^{d} \mathbf{x}_{j,k}^{(\ell)},\end{aligned} \tag{4.11}$$

expressed in standard form via j applications of Lemma 4.2.1. By looking closely at the second and third equalities above, one can observe that for all $j \in [d]$, $\alpha_{j,k} = \alpha_k \prod_{\ell=1}^{j} \left\|\mathbf{A}^{(\ell)}\mathbf{x}_k^{(\ell)}\right\|$, as well as $\mathbf{x}_{j,k}^{(\ell)} = \mathbf{A}^{(\ell)}\mathbf{x}_k^{(\ell)} / \left\|\mathbf{A}^{(\ell)}\mathbf{x}_k^{(\ell)}\right\|$ for $\ell \in [j]$ and $\mathbf{x}_{j,k}^{(\ell)} = \mathbf{x}_k^{(\ell)}$ for $\ell > j$.

The first two parts of Theorem 4.2.1 can be used to write

(i) $\left|\alpha_{j,k} - \alpha_{j-1,k}\right| \leq \varepsilon|\alpha_{j-1,k}|/4d$ *so that* $|\alpha_{j,k}| \leq (1 + \varepsilon/4d)|\alpha_{j-1,k}|$ *holds for all* $k \in [r]$*, and*

(ii) $\mu_{\mathcal{X}^{(j)},j} \leq (\mu_{\mathcal{X}^{(j-1)},j} + \varepsilon/d)/(1 - \varepsilon/4d)$*, and* $\mu_{\mathcal{X}^{(j)},\ell} = \mu_{\mathcal{X}^{(j-1)},\ell}$ *for all* $\ell \in [d] \setminus \{j\}$*,*

for $j \in [d]$. Using these facts inductively, it can also be established that both

$$|\alpha_{j,k}| \leq (1 + \varepsilon/4d)^j |\alpha_k|, \tag{4.12}$$

and

$$\prod_{\ell \neq j} \mu_{\mathcal{X}^{(j-1)},\ell} \leq \left(\prod_{\ell<j} \frac{\mu_{\mathcal{X},\ell} + \varepsilon/d}{1 - \varepsilon/4d}\right)\prod_{\ell>j} \mu_{\mathcal{X},\ell} \leq \left(\frac{\mu_{\mathcal{X}} + \varepsilon/d}{1 - \varepsilon/4d}\right)^{j-1} \mu_{\mathcal{X}}^{d-j}, \tag{4.13}$$

hold for all $k \in [r]$ *and* $j \in [d]$. *To prove* (4.13), *one can write* $\prod_{\ell \neq j} \mu_{\mathcal{X}^{(j-1)},\ell} = \prod_{\ell<j} \mu_{\mathcal{X}^{(j-1)},\ell} \prod_{\ell>j} \mu_{\mathcal{X}^{(j-1)},\ell}$. *The second term on the right-hand side is equal to* $\prod_{\ell>j} \mu_{\mathcal{X},\ell}$ *since* $\ell > j$. *For the first term, we have*

$$\begin{aligned}
\prod_{\ell<j} \mu_{\mathcal{X}^{(j-1)},\ell} &= \left(\prod_{\ell=1}^{j-2} \mu_{\mathcal{X}^{(j-1)},\ell}\right) \mu_{\mathcal{X}^{(j-1)},j-1} = \left(\prod_{\ell=1}^{j-3} \mu_{\mathcal{X}^{(j-1)},\ell}\right) \mu_{\mathcal{X}^{(j-1)},j-2} \mu_{\mathcal{X}^{(j-1)},j-1} \\
&= \left(\prod_{\ell=1}^{j-3} \mu_{\mathcal{X}^{(j-1)},\ell}\right) \mu_{\mathcal{X}^{(j-2)},j-2} \mu_{\mathcal{X}^{(j-1)},j-1} = \cdots = \prod_{\ell=1}^{j-1} \mu_{\mathcal{X}^{(\ell)},\ell} \\
&\leq \prod_{\ell=1}^{j-1} \frac{\mu_{\mathcal{X}^{(\ell-1)},\ell} + \varepsilon/d}{1-\varepsilon/4d} = \prod_{\ell=1}^{j-1} \frac{\mu_{\mathcal{X},\ell} + \varepsilon/d}{1-\varepsilon/4d} \leq \left(\frac{\mu_{\mathcal{X}} + \varepsilon/d}{1-\varepsilon/4d}\right)^{j-1}.
\end{aligned} \tag{4.14}$$

In (4.13), $\mu_{\mathcal{X}}^0 = 1$ *is assumed even if* $\mu_{\mathcal{X}} = 0$ *since this still yields the correct bound in the case where* $j = d$ *and* $\mu_{\mathcal{X}} = 0$.

To get the desired error bound, we can now see that

$$\begin{aligned}
\left|\|\mathcal{X}\|^2 - \|\mathcal{Y}\|^2\right| &= \left|\sum_{j=0}^{d-1} \left\|\mathcal{X}^{(j)}\right\|^2 - \left\|\mathcal{X}^{(j+1)}\right\|^2\right| \\
&\leq \frac{\varepsilon}{d} \sum_{j=0}^{d-1} \left(1 + \sqrt{r(r-1)} \prod_{\ell \neq j+1} \mu_{\mathcal{X}^{(j)},\ell}\right) \|\boldsymbol{\alpha}_j\|_2^2 \\
&\leq \frac{\varepsilon}{d} \sum_{j=0}^{d-1} \left(1 + \sqrt{r(r-1)} \left(\frac{\mu_{\mathcal{X}} + \varepsilon/d}{1-\varepsilon/d4}\right)^j \mu_{\mathcal{X}}^{d-1-j}\right) (1+\varepsilon/4d)^{2j} \|\boldsymbol{\alpha}\|_2^2 \\
&\leq \frac{\varepsilon}{d} \sum_{j=0}^{d-1} \left(1 + \sqrt{r(r-1)} \left(\frac{\mu_{\mathcal{X}} + \varepsilon/d}{1-\varepsilon/d4}\right)^j \mu_{\mathcal{X}}^{d-1-j}\right) (1+9\varepsilon/16d)^j \|\boldsymbol{\alpha}\|_2^2,
\end{aligned}$$

where the third part of Theorem 4.2.1, as well as (4.12) *and* (4.13) *have been used. Considering each term in the upper bound above separately, we have that*

$$\left|\|\mathcal{X}\|^2 - \|\mathcal{Y}\|^2\right| \leq \frac{\varepsilon}{d} \|\boldsymbol{\alpha}\|_2^2 \left(T_1 + \sqrt{r(r-1)} T_2\right)$$

where

$$T_1 := \sum_{j=0}^{d-1} (1+9\varepsilon/16d)^j = \frac{(1+9\varepsilon/16d)^d - 1}{9\varepsilon/16d} \leq \text{e} d$$

using $9\varepsilon/16 < 1$*, and where*

$$T_2 := \sum_{j=0}^{d-1} \left(\frac{\mu_{\mathcal{X}} + \varepsilon/d}{1 - \varepsilon/d4}\right)^j \mu_{\mathcal{X}}^{d-1-j} (1 + 9\varepsilon/16d)^j \;\leq\; \sum_{j=0}^{d-1} (\mu_{\mathcal{X}} + \varepsilon/d)^j \, \mu_{\mathcal{X}}^{d-1-j} (1 + \varepsilon/d)^j$$

for $\varepsilon \leq 3/4$*.*

Continuing to bound the second term we will consider three cases. First, if $\mu_{\mathcal{X}} = 0$ *then*

$$T_2 \;\leq\; (\varepsilon/d)^{d-1} \, (1 + \varepsilon/d)^{d-1} \;\leq\; \mathrm{e} \, (\varepsilon/d)^{d-1} \,,$$

using Remark 4.2.1 and that $\varepsilon < 1$*. Second, if* $0 < \mu_{\mathcal{X}} \leq \varepsilon$ *then*

$$\begin{aligned} T_2 \;&\leq\; \sum_{j=0}^{d-1} (\varepsilon + \varepsilon/d)^j \, \varepsilon^{d-1-j} (1 + \varepsilon/d)^j \;=\; \varepsilon^{d-1} \sum_{j=0}^{d-1} (1 + 1/d)^j \, (1 + \varepsilon/d)^j \\ &\leq\; \varepsilon^{d-1} d \, (1 + 1/d)^d \, (1 + \varepsilon/d)^d \;\leq\; d\mathrm{e}^2 \varepsilon^{d-1}, \end{aligned}$$

using Remark 4.2.1 and that $\varepsilon < 1$ *once more. If, however,* $\mu_{\mathcal{X}} > \varepsilon$ *then we can see that*

$$\begin{aligned} T_2 \;&\leq\; \mu_{\mathcal{X}}^{d-1} \sum_{j=0}^{d-1} (1 + \varepsilon/\mu_{\mathcal{X}} d)^j \, (1 + \varepsilon/d)^j \;\leq\; \mu_{\mathcal{X}}^{d-1} \sum_{j=0}^{d-1} (1 + 1/d)^j \, (1 + \varepsilon/d)^j \\ &\leq\; \mu_{\mathcal{X}}^{d-1} \cdot d \, (1 + 1/d)^d \, (1 + \varepsilon/d)^d \;\leq\; \mu_{\mathcal{X}}^{d-1} \, d \, \mathrm{e}^{1+\varepsilon} \;\leq\; d\mathrm{e}^2 \mu_{\mathcal{X}}^{d-1}, \end{aligned}$$

where we have again utilized Remark 4.2.1. The desired result now follows.

Theorem 4.2.2 expresses the distortion in the Euclidean norm of a low-rank tensor $\mathcal{X}$ after applying modewise JL embeddings in terms of its low-rank expansion coefficients norm $\|\boldsymbol{\alpha}\|_2$. The following lemma helps express the distortion in terms of the norm of a tensor $\mathcal{X}$ in with sufficiently small modewise coherence by establishing its relation to the norm of $\boldsymbol{\alpha}$, as this is usually the convention in expressing error guarantees for JL embeddings.

Lemma 4.2.3 *Let* $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$ *be a rank-*r *tensor in the standard form as per* (4.3) *with basis coherence* $\mu'_{\mathcal{X}} < (r-1)^{-1}$*. Then,*

$$\begin{aligned} \|\boldsymbol{\alpha}\|_2^2 \;&\leq\; \left(\frac{1}{1 - (r-1)\mu'_{\mathcal{X}}}\right) \|\mathcal{X}\|^2 \;\leq\; \left(\frac{1}{1 - (r-1)\prod_{\ell=1}^{d} \mu_{\mathcal{X},\ell}}\right) \|\mathcal{X}\|^2 \\ &\leq\; \left(\frac{1}{1 - (r-1)\mu_{\mathcal{X}}^d}\right) \|\mathcal{X}\|^2. \end{aligned} \tag{4.15}$$

Proof *To establish the result, one can write*

$$\begin{aligned}\|\mathcal{X}\|^2 = \langle \mathcal{X}, \mathcal{X}\rangle &= \left\langle \sum_{k=1}^{r} \alpha_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)}, \sum_{k=1}^{r} \alpha_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \right\rangle = \sum_{k,h=1}^{r} \alpha_k \overline{\alpha_h} \left\langle \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)}, \bigcirc_{\ell=1}^{d} \mathbf{x}_h^{(\ell)} \right\rangle \\ &= \sum_{k,h=1}^{r} \alpha_k \overline{\alpha_h} \prod_{\ell=1}^{d} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle = \sum_{k=1}^{r} |\alpha_k|^2 + \sum_{k\neq h}^{r} \alpha_k \overline{\alpha_h} \prod_{\ell=1}^{d} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \\ &= \|\boldsymbol{\alpha}\|^2 + \sum_{k\neq h}^{r} \alpha_k \overline{\alpha_h} \prod_{\ell=1}^{d} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle.\end{aligned}$$

where (A.8) *has been used to get the fourth equality. Therefore,*

$$\begin{aligned}\left| \|\mathcal{X}\|^2 - \|\boldsymbol{\alpha}\|_2^2 \right| &= \left| \sum_{k\neq h}^{r} \alpha_k \overline{\alpha_h} \prod_{\ell=1}^{d} \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \right| \leq \sum_{k\neq h}^{r} |\alpha_k \overline{\alpha_h}| \prod_{\ell=1}^{d} \left| \left\langle \mathbf{x}_k^{(\ell)}, \mathbf{x}_h^{(\ell)} \right\rangle \right| \leq \mu_{\mathcal{X}}^{'} \sum_{k\neq h}^{r} |\alpha_k \overline{\alpha_h}| \\ &= \mu_{\mathcal{X}}^{'} \left[\left(\sum_{k=1}^{r} |\alpha_k| \right)^2 - \sum_{k=1}^{r} |\alpha_k|^2 \right] = \mu_{\mathcal{X}}^{'} \left(\|\boldsymbol{\alpha}\|_1^2 - \|\boldsymbol{\alpha}\|_2^2 \right) \leq \mu_{\mathcal{X}}^{'} (r-1) \|\boldsymbol{\alpha}\|_2^2,\end{aligned}$$

yielding the result and implying that $\mu_{\mathcal{X}}^{'}(r-1) < 1$ *should also hold given that both* $\|\mathcal{X}\|$ *and* $\|\boldsymbol{\alpha}\|_2$ *are non-negative numbers. The relation* $\|\boldsymbol{\alpha}\|_1 \leq \sqrt{r}\|\boldsymbol{\alpha}\|_2$ *has been used to get the final inequality.*

Theorem 4.2.2 guarantees that modewise JL embeddings approximately preserve the norms of all tensors in the form of (4.3). Theorem 4.2.3 below states the inner product preservation property of JL embeddimgs for low-rank tensors, and guarantees that the inner products of all tensors in the span of the set $\mathcal{B} := \left\{ \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{\ell} \middle| k \in [r] \right\} \in \mathbb{C}^{n_1 \cdots \times n_d}$ are preserved.

Theorem 4.2.3 *Suppose that* $\mathcal{X}_1, \mathcal{X}_2 \in \mathcal{L} \subset \mathbb{C}^{n_1 \times \cdots \times n_d}$ *have standard forms as per* (4.3)*, in terms of the elements of the basis* $\mathcal{B} := \left\{ \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{\ell} \middle| k \in [r] \right\} \in \mathbb{C}^{n_1 \cdots \times n_d}$*, given by*

$$\mathcal{X}_1 = \sum_{k=1}^{r} \beta_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)}, \text{ and } \mathcal{X}_2 = \sum_{k=1}^{r} \alpha_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)}.$$

Let $\varepsilon \in (0, 3/4]$*, and* $\mathbf{A}^{(j)} \in \mathbb{C}^{m_j \times n_j}$ *be defined as per Theorem 4.2.2 for each* $j \in [d]$*. Then,*

$$\begin{aligned}\left| \left\langle \mathcal{X}_1 \bigtimes_{j=1}^{d} \mathbf{A}^{(j)}, \mathcal{X}_2 \bigtimes_{j=1}^{d} \mathbf{A}^{(j)} \right\rangle - \langle \mathcal{X}_1, \mathcal{X}_2 \rangle \right| &\leq 2\varepsilon' \left(\|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{\alpha}\|_2^2 \right) \leq 4\varepsilon' \cdot \max\left\{ \|\boldsymbol{\beta}\|_2^2, \|\boldsymbol{\alpha}\|_2^2 \right\} \\ &\leq 4\varepsilon' \cdot \frac{\max\left\{ \|\mathcal{X}_1\|^2, \|\mathcal{X}_2\|^2 \right\}}{1 - (r-1)\mu_{\mathcal{B}}'},\end{aligned}$$

where

$$\varepsilon' := \begin{cases} \left(\varepsilon + \mathbb{e}\sqrt{r(r-1)}\varepsilon^d\right)\mathbb{e} & \text{if } \mu_{\mathcal{B}} = 0, \\ \varepsilon\left(\mathbb{e} + \mathbb{e}^2\sqrt{r(r-1)}\max\left(\varepsilon^{d-1}, \mu_{\mathcal{B}}^{d-1}\right)\right) & \text{otherwise.} \end{cases} \tag{4.16}$$

Proof *Using the polarization identity in combination with Lemma 2.0.2 and Theorem 4.2.2 we can see that*

$$\begin{aligned}
&\left|\left\langle \mathcal{X}_1 \bigtimes_{j=1}^{d} \mathbf{A}^{(j)},\ \mathcal{X}_2 \bigtimes_{j=1}^{d} \mathbf{A}^{(j)} \right\rangle - \langle \mathcal{X}_1,\ \mathcal{X}_2 \rangle\right| \\
&= \left| \frac{1}{4} \sum_{\ell=0}^{3} \mathbb{i}^\ell \left(\left\| \mathcal{X}_1 \bigtimes_{j=1}^{d} \mathbf{A}^{(j)} + \mathbb{i}^\ell \mathcal{X}_2 \bigtimes_{j=1}^{d} \mathbf{A}^{(j)} \right\|_2^2 - \left\| \mathcal{X}_1 + \mathbb{i}^\ell \mathcal{X}_2 \right\|_2^2 \right) \right| \\
&= \left| \frac{1}{4} \sum_{\ell=0}^{3} \mathbb{i}^\ell \left(\left\| \left(\mathcal{X}_1 + \mathbb{i}^\ell \mathcal{X}_2 \right) \bigtimes_{j=1}^{d} \mathbf{A}^{(j)} \right\|_2^2 - \left\| \mathcal{X}_1 + \mathbb{i}^\ell \mathcal{X}_2 \right\|_2^2 \right) \right| \\
&\leq \frac{1}{4} \sum_{\ell=0}^{3} \left| \left\| \left(\mathcal{X}_1 + \mathbb{i}^\ell \mathcal{X}_2 \right) \bigtimes_{j=1}^{d} \mathbf{A}^{(j)} \right\|_2^2 - \left\| \mathcal{X}_1 + \mathbb{i}^\ell \mathcal{X}_2 \right\|_2^2 \right| \\
&\leq \frac{1}{4} \sum_{\ell=0}^{3} \varepsilon' \left\| \boldsymbol{\beta} + \mathbb{i}^\ell \boldsymbol{\alpha} \right\|_2^2 \leq \frac{1}{4} \sum_{\ell=0}^{3} \varepsilon' \left(\|\boldsymbol{\beta}\|_2 + \|\boldsymbol{\alpha}\|_2 \right)^2 = \varepsilon' \left(\|\boldsymbol{\beta}\|_2 + \|\boldsymbol{\alpha}\|_2 \right)^2 \\
&\leq 2\varepsilon' \left(\|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{\alpha}\|_2^2 \right) \ \leq\ 4\varepsilon' \max\left\{ \|\boldsymbol{\beta}\|_2^2, \|\boldsymbol{\alpha}\|_2^2 \right\},
\end{aligned}$$

where the third to last and second to last inequalities follow from the triangle inequality and Young's inequality for products, respectively. Applying Lemma 4.2.3 to relate the Euclidean norms of $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ to $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively, leads to the final result.

To sum up, theorems 4.2.2 and 4.2.3 guarantee that modewise JL embeddings approximately preserve the norms of and inner products between all tensors in the span of the set

$$\mathcal{B} := \left\{ \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \mid k \in [r] \right\} \subset \mathbb{C}^{n_1 \times \cdots \times n_d}.$$

4.2.1.1 Computational Complexity of Modewise Johnson-Lindenstrauss Embeddings

Assuming $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ and $\mathbf{A}^{m_j \times n_j}$ with $m_j \leq n_j$, we have $\mathcal{X} \bigtimes_{j=1}^{d} \mathbf{A}^{(j)} \in \mathbb{R}^{m_1 \times \ldots \times m_d}$. The total operation count[4] of the embedding is the sum of the operation counts for each j-mode product in each mode. Therefore, one can show that the operations count will be

$$O\left(m_1 n_1 \ldots n_d\right) + O\left(m_1 m_2 n_2 \ldots n_d\right) + \cdots + O\left(m_1 m_2 \ldots m_d n_d\right) = O\left(m_1 n_1 \ldots n_d\right).$$

On the other hand, if $\mathcal{X}$ is vectorized, to achieve the same compression, it should be left-multiplied by a JL matrix $\mathbf{A} \in \mathbb{R}^{m_1 \ldots m_d \times n_1 \ldots n_d}$. The computational complexity would then be $O\left(m_1 n_1 \ldots m_d n_d\right)$ which is significantly higher than the complexity of the modewise approach.

4.2.2 Main Theorems: Oblivious Tensor Subspace Embeddings

So far, no assumption has been made about the type of JL embeddings that have been considered for dimension reduction of tensor data. In this section, the main theorems establishing bounds on the embedding dimension are presented in the case where randomness is incorporated into the JL embeddings being used. In the case of finite-dimensional bases, the JL embeddings of interest will be matrices drawn from random distributions, or are contructed as the product of matrices some of which have random properties. This section starts with the definition of the family of η-optimal JL embedding distributions, followed by the main theorems.

Definition 4.2.3 *Fix $\eta \in (0, 1/2)$ and let $\left\{\mathcal{D}_{(m,n)}\right\}_{(m,n)\in\mathbb{N}\times\mathbb{N}}$ be a family of probability distributions where each $\mathcal{D}_{(m,n)}$ is a distribution over $m \times n$ matrices. We will refer to any such family of distributions as being an η-optimal family of JL embedding distributions if there exists an absolute constant $C \in \mathbb{R}^+$ such that, for any given $\varepsilon \in (0, 1)$, $m, n \in \mathbb{N}$ with $m < n$, and nonempty set $\mathcal{S} \subset \mathbb{C}^n$ of cardinality*

$$|S| \leq \eta \exp\left(\frac{\varepsilon^2 m}{C}\right),$$

[4]Here, the elements of tensors and matrices are assumed to belong to the field of real numbers, for simplicity. The operation count can be updated accordingly when the field of complex numbers is considered.

a matrix $\mathbf{A} \sim \mathcal{D}_{(m,n)}$ *will be an* ε*-JL embedding of* $\mathcal{S}$ *into* $\mathbb{C}^m$ *with probability at least* $1 - \eta$*.*

In fact, many η-optimal families of JL embedding distributions exist for any given $\eta \in (0, 1/2)$, including, e.g., those associated with random matrices having independent and identically distributed (i.i.d.) sub-Gaussian entries (see Lemma 9.35 in [5]).

We are now ready to state the main oblivious subspace property of JL embeddings for low-rank tensors. It should be noted, however, that as Lemma 4.2.3 suggests, an incoherence assumption is necessary to establish a relation between the Euclidean norm of a low-rank tensor and the norm of its expansion coefficients in the standard form. This assumption will be necessary for the proof of Theorem 4.2.4 to work.

Theorem 4.2.4 *Fix* $\varepsilon, \eta \in (0, 1/2)$ *and* $d \geq 2$*. Let* $\mathcal{L}$ *be an* r*-dimensional subspace of* $\mathbb{C}^{n_1 \times \cdots \times n_d}$ *spanned by a basis of rank-*1 *tensors* $\mathcal{B} := \left\{ \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \;\middle|\; k \in [r] \right\}$ *with modewise coherence as per* (4.4) *satisfying* $\mu_{\mathcal{B}}^{d-1} < 1/2r$*. For each* $j \in [d]$ *draw* $\mathbf{A}^{(j)} \in \mathbb{C}^{m_j \times n_j}$ *with*

$$m_j \geq \tilde{C} \cdot r^{2/d} d^2 / \varepsilon^2 \cdot \ln\left(2r^2 d/\eta\right) \tag{4.17}$$

from an (η/d)*-optimal family of JL embedding distributions, where* $\tilde{C} \in \mathbb{R}^+$ *is an absolute constant. Then, with probability at least* $1 - \eta$ *we have*

$$\left| \left\| \mathcal{X} \times_1 \mathbf{A}^{(1)} \cdots \times_d \mathbf{A}^{(d)} \right\|^2 - \|\mathcal{X}\|^2 \right| \leq \varepsilon \, \|\mathcal{X}\|^2 , \tag{4.18}$$

for all $\mathcal{X} \in \mathcal{L}$*.*

Proof *Let* $\mathcal{B}$ *be a set of* r *rank-*1 *tensors, defined as*

$$\mathcal{B} := \left\{ \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \;\middle|\; k \in [r] \right\} \subset \mathbb{C}^{n_1 \times \cdots \times n_d}$$

with $\|\mathbf{x}_k^{(\ell)}\|_2 = 1$ *for* $k \in [r]$ *and* $\ell \in [d]$*, and let* $\mathcal{L} := \mathrm{span}\,(\mathcal{B})$*. We first note the coherence assumption* $\mu_{\mathcal{B}}^{d-1} < 1/2r$ *guarantees that*

$$\mu'_{\mathcal{B}} \leq \mu_{\mathcal{B}}^{d} \leq \mu_{\mathcal{B}}^{d-1} < 1/2r < 1/2(r-1),$$

which can be rearranged as

$$4/(1-(r-1)\mu'_{\mathcal{B}}) \le 8.$$

Also, letting $\delta := (1/r)^{1/d}\varepsilon/16\mathrm{e}$ *and according to* (4.16)*, it is enough to have* $\varepsilon \ge 8\delta\mathrm{e} + 8\delta\mathrm{e}^2 r\max(\delta^{d-1}, \mu_{\mathcal{B}}^{d-1})$ *so that each embedding* $\mathbf{A}^{(j)}$ *will be a* $(\delta/4d)$*-JL embedding of the set* S'_j *in* (4.6) *into* $\mathbb{C}^{m_j}$ *where* $\varepsilon'(\delta)$ *is defined by* (4.16) *and* $\varepsilon \ge 8\varepsilon'$*. Furthermore, if* $\mathbf{A}^{(j)}$ *is taken from an* η/d*-optimal family of JL distributions, it will also be a* $(\delta/4d)$*-JL embedding of* S'_j *in* (4.6) *into* $\mathbb{C}^{m_j}$ *with probability* $1-\eta/d$ *if*

$$|S'_j| = 2r^2 - r \le \frac{\eta}{d}\exp\left(\frac{\delta^2 m_j}{16d^2 C}\right),$$

which is satisfied for each m_j *defined in* (4.17)*. Finally, taking union bound over all* d *modes concludes the proof.*

Note 4.2.2 *Theorem 4.2.4 can be used in the special case where* $\mathcal{X}$ *is a matrix* $\mathbf{X} \in \mathbb{C}^{n_1\times n_2}$*. In this case, the CP-rank is the usual matrix rank, and the CP decomposition becomes the regular SVD decomposition of the matrix, which can be computed efficiently in parallel (see, e.g., [8]). In particular, the basis vectors are orthogonal to each other in this case. The result of Theorem 4.2.4 implies that taking* $\mathbf{A}$ *and* $\mathbf{B}$ *as matrices belonging to the* $(\eta/2)$*-JL embedding family and of sizes* $n_1\times m_1$ *and* $n_2\times m_2$*, respectively, such that* $m_j \gtrsim r\ln(r/\sqrt{\eta})/\varepsilon^2$ *(for* $j=1,2$*), we get the following JL-type result for the Frobenius matrix norm: with probability* $1-\eta$*,*

$$\|\mathbf{A}^T\mathbf{X}\mathbf{B}\|_F^2 = (1+\tilde{\varepsilon})\|\mathbf{X}\|_F^2 \quad \text{for some } |\tilde{\varepsilon}| \le \varepsilon.$$

Theorem 4.2.5 *Fix* $\varepsilon, \eta \in (0, 1/2)$ *and* $d \ge 3$*. Let* $\mathcal{X} \in \mathbb{C}^{n_1\times\cdots\times n_d}$*,* $n := \max_j n_j \ge 4r+1$*, and let* $\mathcal{L}$ *be an* r*-dimensional subspace of* $\mathbb{C}^{n_1\times\cdots\times n_d}$ *spanned by a basis* $\mathcal{B} := \left\{\bigcirc_{\ell=1}^{d}\mathbf{x}_k^{(\ell)} \;\middle|\; k\in[r]\right\}$ *of rank-*1 *tensors, with modewise coherence satisfying* $\mu_{\mathcal{B}}^{d-1} < 1/2r$*. For each* $j\in[d]$*, draw* $\mathbf{A}^{(j)} \in \mathbb{C}^{m_j\times n_j}$ *with*

$$m_j \ge C_j \cdot r d^3/\varepsilon^2 \cdot \ln\left(n/\sqrt[d]{\eta}\right) \tag{4.19}$$

from an $(\eta/4d)$-optimal family of JL embedding distributions, where $C_j \in \mathbb{R}^+$ is an absolute constant. Furthermore, let $\mathbf{A} \in \mathbb{C}^{m' \times \prod_{\ell=1}^{d} m_\ell}$ with

$$m' \geq C' r \cdot \varepsilon^{-2} \cdot \ln\left(\frac{47}{\varepsilon \sqrt[4]{\eta}}\right)$$

be drawn from an $(\eta/2)$-optimal family of JL embedding distributions, where $C' \in \mathbb{R}^+$ is an absolute constant. Define $\tilde{L} : \mathbb{C}^{n_1 \times \cdots \times n_d} \to \mathbb{C}^{m_1 \times \cdots \times m_d}$ by $\tilde{L}(\mathcal{Z}) = \mathcal{Z} \times_1 \mathbf{A}^{(1)} \cdots \times_d \mathbf{A}^{(d)}$. Then, with probability at least $1 - \eta$, the linear operator $\mathbf{A} \circ \text{vect} \circ \tilde{L} : \mathbb{C}^{n_1 \times \cdots \times n_d} \to \mathbb{C}^{m'}$ satisfies

$$\left| \left\| \mathbf{A} \left(\text{vect} \circ \tilde{L} \left(\mathcal{X} - \mathcal{Y} \right) \right) \right\|_2^2 - \|\mathcal{X} - \mathcal{Y}\|^2 \right| \leq \varepsilon \|\mathcal{X} - \mathcal{Y}\|^2$$

for all $\mathcal{Y} \in \mathcal{L}$.

Proof *To begin, we note that $\mathbf{A}$ will satisfy the conditions required by Theorem 5.1.1 with probability at least $1 - \eta/2$ as a consequence of Lemma 4.1.3. Thus, if we can also establish that $\tilde{L}$ will satisfy the conditions required by Theorem 5.1.1 with probability at least $1 - \eta/2$, we will be finished with our proof by Theorem 5.1.1 and the union bound.*

To establish that $\tilde{L}$ satisfies the conditions required by Theorem 5.1.1 with probability at least $1 - \eta/2$, it suffices to prove that

(a) $\tilde{L}$ will be an $(\varepsilon/6)$-JL embedding of all $\mathcal{Y} \in \mathcal{L}$ into $\mathbb{C}^{m_1 \times \cdots \times m_d}$ with probability at least $1 - \eta/4$, and that

(b) $\tilde{L}$ will be an $(\varepsilon/24\sqrt{r})$-JL embedding of the $4r+1$ tensors $\mathcal{S}' \cup \{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\} \subset \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_d}$ into $\mathbb{C}^{m_1 \times \cdots \times m_d}$ with probability at least $1 - \eta/4$, where the set $\mathcal{S}'$ is defined as in Theorem 5.1.1,

and apply yet another union bound.

To show that (a) holds, we will utilize Theorem 4.2.2 and Lemma 4.2.3. Since each $\mathbf{A}^{(j)}$ matrix is an $(\eta/4d)$-optimal JL embedding and the sets $\mathcal{S}'_j$ defined as in (4.6) are such that $|\mathcal{S}'_j| < n^d$, we know that each $\mathbf{A}^{(j)}$ is an $\left(\varepsilon/480d\sqrt{r}\right)$-JL embedding of $\mathcal{S}'_j$ into $\mathbb{C}^{m_j}$ with probability[5] at

[5]Here we also implicitly use the fact that $\sqrt[d]{d} \leq \sqrt[e]{e}$ holds for all $d > 0$ in order to avoid a $\sqrt[d]{d}$ term appearing inside the logarithm in (4.19).

least $1 - \eta/4d$. *Thus, Theorem 4.2.2 holds with* $\varepsilon \to \varepsilon/120\sqrt{r}$ *with probability at least* $1 - \eta/4$. *Note that the modewise coherence assumption that* $\mu_{\mathcal{B}}^{d-1} < 1/2r$ *both allows* ε^{d-1} *to reduce the* $\sqrt{r(r-1)}$ *factor in* (4.9) *to a size less than one for any* $\varepsilon \leq 1/\sqrt{r} \leq (1/r)^{1/(d-1)}$, *and also allows Lemma 4.2.3 to guarantee that* $\|\boldsymbol{\alpha}\|_2^2 < 2\|\mathcal{Y}\|^2$ *holds for all* $\mathcal{Y} \in \mathcal{L}$. *Hence, applying Theorem 4.2.2 with* $\varepsilon \to \varepsilon/120\sqrt{r}$ *will ensure that* $\tilde{L}$ *is an* $(\varepsilon/6)$*-JL embedding of all* $\mathcal{Y} \in \mathcal{L}$ *into* $\mathbb{C}^{m_1 \times \cdots \times m_d}$.

To show that (b) holds we will utilize Lemma 5.1.1. Note that the $\mathcal{S}_j$ *sets defined in Lemma 5.1.1 all have cardinalities* $|\mathcal{S}_j| \leq pn^{d-1}$, *where* $p = 4r + 1 \leq n$ *in our current setting. As a consequence, we can see that the conditions of Lemma 5.1.1 will be satisfied with* $\varepsilon \to \varepsilon/24\sqrt{r}$ *for all* $j \in [d]$ *with probability at least* $1 - \eta/4$ *by the union bound. Hence, both (a) and (b) hold and our proof is concluded.*

Figure 4.1 provides a schematic view of the 2-stage JL embedding introduced in Theorem 4.2.5 on a $3 \times 4 \times 5$ sample tensor.

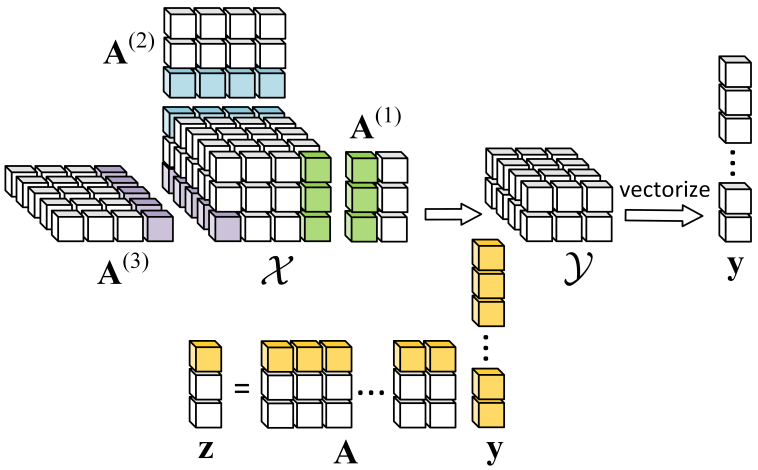


Figure 4.1: An example of 2-stage JL embedding applied to a 3-dimensional tensor $\mathcal{X} \in \mathbb{R}^{3\times4\times5}$. The output of the 1st stage is the projected tensor $\mathcal{Y} = \mathcal{X} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}$, where $\mathbf{A}^{(j)}$ are JL matrices for $j \in \{1, 2, 3\}$, $\mathbf{A}^{(1)} \in \mathbb{R}^{2\times3}$, $\mathbf{A}^{(2)} \in \mathbb{R}^{3\times4}$, and $\mathbf{A}^{(3)} \in \mathbb{R}^{4\times5}$, resulting in $\mathcal{Y} \in \mathbb{R}^{2\times3\times4}$. Matching colors have been used to show how the rows of $\mathbf{A}^{(j)}$ interact with the mode-j fibers of $\mathcal{X}$ (and the intermediate partially compressed tensors) to generate the elements of the mode-j unfolding of the result after each j-mode product. Next, the resulting tensor is vectorized (leading to $\mathbf{y} \in \mathbb{R}^{24}$), and a 2nd-stage JL is then performed to obtain $\mathbf{z} = \mathbf{A}\mathbf{y}$ where $\mathbf{A} \in \mathbb{R}^{3\times24}$, and $\mathbf{z} \in \mathbb{R}^3$.

Note 4.2.3 (About r and ε Dependence) *Fix d, n, and η. Looking at Theorem 4.2.5 we can see that it's intermediate embedding dimension is*

$$\prod_{\ell=1}^{d} m_\ell \leq C_{d,\eta,n}^{d} r^d \varepsilon^{-2d}$$

which effectively determines its overall storage complexity. Hence, Theorem 4.2.5 will only result in an improved memory complexity over the straightforward single-stage vectorization approach if, e.g., the rank r of $\mathcal{L}$ is relatively small. The purpose of facultative vectorization and subsequent multiplication by an additional JL transform $\mathbf{A}$ in Theorem 4.2.5 is to reduce the resulting final embedding dimension to the near-optimal order $O(r\varepsilon^{-2})$ from total dimension $O_{\eta,n}(d^{3d} r^d \varepsilon^{-2d})$ that we have after the modewise compression.

4.2.3 Fast and Memory-Efficient Modewise Johnson-Lindenstrauss Embeddings

In this section we consider a fast Johnson-Lindenstrauss transform for tensors recently introduced in [10], which is effectively based on applying fast JL transforms [13] in a modewise fashion.[6] Given a tensor $\mathcal{Z} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$ the transform takes the form

$$L_{\mathrm{FJL}}(\mathcal{Z}) \;:=\; \sqrt{\frac{N}{m}}\, \mathbf{R} \left(\mathrm{vect}\left(\mathcal{Z} \times_1 \mathbf{F}^{(1)} \mathbf{D}^{(1)} \cdots \times_d \mathbf{F}^{(d)} \mathbf{D}^{(d)} \right) \right) \tag{4.20}$$

where $\mathrm{vect} : \mathbb{C}^{n_1 \times \cdots \times n_d} \to \mathbb{C}^N$ for $N := \prod_{\ell=1}^{d} n_\ell$ is the vectorization operator, $\mathbf{R} \in \{0,1\}^{m \times N}$ is a matrix containing m rows selected randomly from the $N \times N$ identity matrix, $\mathbf{F}^{(\ell)} \in \mathbb{C}^{n_\ell \times n_\ell}$ is a unitary discrete Fourier transform matrix for all $\ell \in [d]$, and $\mathbf{D}^{(\ell)} \in \mathbb{C}^{n_\ell \times n_\ell}$ is a diagonal matrix with n_ℓ random ± 1 entries for all $\ell \in [d]$. The following theorem is proven about this transform in [10, 13].

Theorem 4.2.6 (See Theorem 2.1 and Remark 4 in [10]) *Fix $d \geq 1$, $\varepsilon, \eta \in (0,1)$, and $N \geq C'/\eta$ for a sufficiently large absolute constant $C' \in \mathbb{R}^+$. Consider a finite set $\mathcal{S} \subset \mathbb{C}^{n_1 \times \cdots \times n_d}$ of cardinality*

[6] In fact, the fast transform described here differs cosmetically from the form in which it is presented in [10]. However, one can easily see they are equivalent using (2.10).

$p = |\mathcal{S}|$, and let $L_{\text{FJL}} : \mathbb{C}^{n_1 \times \cdots \times n_d} \to \mathbb{C}^m$ be defined as above in (4.20) *with*

$$m \;\geq\; C\left[\varepsilon^{-2} \cdot \log^{2d-1}\left(\frac{\max(p,N)}{\eta}\right) \cdot \log^4\left(\frac{\log\left(\frac{\max(p,N)}{\eta}\right)}{\varepsilon}\right) \cdot \log N\right],$$

where $C > 0$ is an absolute constant. Then with probability at least $1 - \eta$ the linear operator L_{FJL} is an ε-JL embedding of $\mathcal{S}$ into $\mathbb{C}^m$. If $d = 1$ then we may replace $\max(p, N)$ with p inside all of the logarithmic factors above (see [13]).

Note that the fast transform L_{FJL} requires only $O\left(m \log N + \sum_\ell n_\ell\right)$ i.i.d. random bits and memory for storage. Thus, it can be used to produce fast and low memory complexity oblivious subspace embeddings. The next Theorem does so.

Theorem 4.2.7 *Fix $\varepsilon, \eta \in (0, 1/2)$ and $d \geq 2$. Let $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$, $N = \prod_{\ell=1}^d n_\ell \geq 4C'/\eta$ for an absolute constant $C' > 0$, $\mathcal{L}$ be an r-dimensional subspace of $\mathbb{C}^{n_1 \times \cdots \times n_d}$ for $\max\left(2r^2 - r, 4r\right) \leq N$, and $L_{\text{FJL}} : \mathbb{C}^{n_1 \times \cdots \times n_d} \to \mathbb{C}^{m_1}$ be defined as above in* (4.20) *with*

$$m_1 \;\geq\; C_1\left[C_2^d\left(\frac{r}{\varepsilon}\right)^2 \cdot \log^{2d-1}\left(\frac{N}{\eta}\right) \cdot \log^4\left(\frac{\log\left(\frac{N}{\eta}\right)}{\varepsilon}\right) \cdot \log N\right],$$

where $C_1, C_2 > 0$ are absolute constants. Furthermore, let $\mathbf{L}'_{\text{FJL}} \in \mathbb{C}^{m_2 \times m_1}$ be defined as above in (4.20) *for $d = 1$ with*

$$m_2 \;\geq\; C_3\left[r \cdot \varepsilon^{-2} \cdot \log\left(\frac{47}{\varepsilon\sqrt[4]{\eta}}\right) \cdot \log^4\left(\frac{r\log\left(\frac{47}{\varepsilon\sqrt[4]{\eta}}\right)}{\varepsilon}\right) \cdot \log m_1\right],$$

where $C_3 > 0$ is an absolute constant. Then, with probability at least $1 - \eta$ it will be the case that

$$\left|\left\|\mathbf{L}'_{\text{FJL}}\left(L_{\text{FJL}}\left(\mathcal{X} - \mathcal{Y}\right)\right)\right\|_2^2 - \left\|\mathcal{X} - \mathcal{Y}\right\|^2\right| \leq \varepsilon \left\|\mathcal{X} - \mathcal{Y}\right\|^2$$

holds for all $\mathcal{Y} \in \mathcal{L}$.

In addition, the $\left(\mathbf{L}'_{\text{FJL}}, L_{\text{FJL}}\right)$ transform pair requires only $O\left(m_1 \log N + \sum_\ell n_\ell\right)$ random bits and memory for storage (assuming w.l.o.g. that $m_2 \leq m_1$), and $\mathbf{L}'_{\text{FJL}} \circ L_{\text{FJL}} : \mathbb{C}^{n_1 \times \cdots \times n_d} \to \mathbb{C}^{m_2}$ can be applied to any tensor in just $O\left(N \log N\right)$-time.

Proof *Let* $\{\mathcal{T}_k\}_{k\in[r]}$ *be an orthonormal basis for* $\mathcal{L}$ *(note that these basis tensors need not be low-rank), and* $\mathbb{P}_{\mathcal{L}^\perp} : \mathbb{C}^{n_1\times\cdots\times n_d} \to \mathbb{C}^{n_1\times\cdots\times n_d}$ *be the orthogonal projection operator onto the orthogonal complement of* $\mathcal{L}$*. Theorem 5.1.1 combined with Lemmas 4.1.4 and 4.1.3 imply that the result will be proven if all of the following hold:*

(i) L_{FJL} *is an* $(\varepsilon/24r)$*-JL embedding of the* $2r^2 - r$ *tensors*

$$\left(\bigcup_{1\leq h<k\leq r} \{\mathcal{T}_k - \mathcal{T}_h, \mathcal{T}_k + \mathcal{T}_h, \mathcal{T}_k - \mathbb{i}\mathcal{T}_h, \mathcal{T}_k + \mathbb{i}\mathcal{T}_h\}\right) \bigcup \{\mathcal{T}_k\}_{k\in[r]} \subset \mathcal{L}$$

into $\mathbb{C}^{m_1}$*,*

(ii) L_{FJL} *is an* $(\varepsilon/6)$*-JL embedding of* $\{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\}$ *into* $\mathbb{C}^{m_1}$*,*

(iii) L_{FJL} *is an* $(\varepsilon/24\sqrt{r})$*-JL embedding of the* $4r$ *tensors*

$$\bigcup_{k\in[r]} \left\{ \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} - \mathcal{T}_k, \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} + \mathcal{T}_k, \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} - \mathbb{i}\mathcal{T}_k, \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} + \mathbb{i}\mathcal{T}_k \right\} \subset \mathbb{C}^{n_1\times\ldots\times n_d}$$

into $\mathbb{C}^{m_1}$*, and*

(iv) $\mathbf{L}'_{\text{FJL}}$ *is an* $(\varepsilon/6)$*-JL embedding of a minimal* $(\varepsilon/16)$*-cover,* C*, of the* r*-dimensional Euclidean unit sphere in the subspace* $\mathcal{L}' \subset \mathbb{C}^{m_1}$ *from Theorem 5.1.1 with* $L = L_{\text{FJL}}$ *into* $\mathbb{C}^{m_2}$*. Here we note that* $|C| \leq \left(\frac{47}{\varepsilon}\right)^r$*.*

Furthermore, if m_1 *and* m_2 *are chosen as above for sufficiently large absolute constants* C_1*,* C_2*, and* C_3*, then Theorem 4.2.6 implies that each of* $(i) - (iv)$ *above will fail to hold with probability at most* $\eta/4$*. The desired result now follows from the union bound.*

The number of random bits and storage complexity follows directly form Theorem 4.2.6 after noting that each row of $\mathbf{R}$ *in* (4.20) *is determined by* $\mathcal{O}(\log N)$ *bits. The fact that* $\mathbf{L}'_{\text{FJL}} \circ L_{\text{FJL}}$ *can be applied to any tensor* $\mathcal{Z}$ *in* $\mathcal{O}(N \log N)$*-time again follows from the form of* (4.20)*. Note that each* j*-mode product with* $\mathbf{F}^{(j)}\mathbf{D}^{(j)}$ *involves* $\prod_{\ell\neq j} n_\ell$ *multiplications of* $\mathbf{F}^{(j)}\mathbf{D}^{(j)}$ *against all the mode-*j *fibers of the given tensor* $\mathcal{Z}$*, each of which can be performed in* $\mathcal{O}(n_j \log(n_j))$*-time using fast*

Fourier transform techniques (or approximated even more quickly using sparse Fourier transform techniques if n_j is itself very large). The required vectorization and applications of $\mathbf{R}$ *can then be performed in just $O(N)$-time thereafter. Finally, Fourier transform techniques can again be used to also apply $\mathbf{L}'_{\text{FJL}}$ in $O(m_1 \log m_1)$-time.*

4.3 Experiments

In this section, it is shown that the norms of several different types of (approximately) low-rank data can be preserved using JL embeddings. The data sets used in the experiments consist of

1. *MRI data*: This data set contains three 3-mode MRI images of size $240 \times 240 \times 155$ [1].

2. *Randomly generated data*: This data set contains 10 rank-10 4-mode tensors. Each test tensor is a $100 \times 100 \times 100 \times 100$ array that is created by adding 10 randomly generated rank-1 tensors. More specifically, each rank-10 tensor is generated according to

$$\mathcal{X}^{(m)} = \sum_{k=1}^{r} \bigcirc_{j=1}^{d} \mathbf{x}_k^{(j)},$$

   where $m \in [10]$, $r = 10$, $d = 4$ and $\mathbf{x}_k^{(j)} \in \mathbb{R}^{100}$. In the Gaussian case, each entry of $\mathbf{x}_k^{(j)}$ is drawn independently from the standard Gaussian distribution $\mathcal{N}(0, 1)$. In the case of coherent data, low-variance Gaussian noise is added to a constant, i.e., each entry $\mathbf{x}_{k,\ell}^{(j)}$ of $\mathbf{x}_k^{(j)}$ is set as $1 + \sigma g_{k,\ell}^{(j)}$ with $g_{k,\ell}^{(j)}$ being an i.i.d. standard Gaussian random variable defined above, and σ^2 denoting the desired variance. In the experiments of this section, $\sigma = \sqrt{0.1}$ is used. In both cases, the 2-norm of $\mathbf{x}_k^{(j)}$ is also normalized to 1.

   The reason for running experiments on both Gaussian and coherent data is to show that although coherence requirements presented in section 4.2 are used to help get general theoretical results for a large class of modewise JL embeddings, they do not seem to be necessary in practice.

When JL embeddings are applied, experiments are performed using Gaussian JL matrices as well as Fast JL matrices. For Gaussian JL, $\mathbf{A}_j = \frac{1}{\sqrt{m}}\mathbf{G}$ is used for all $j \in [d]$, where m is the target dimension and each entry in $\mathbf{G}$ is an i.i.d. standard Gaussian random variable $\mathbf{G}_{i,j} \sim \mathcal{N}(0,1)$. For Fast JL, $\mathbf{A}_j = \frac{1}{\sqrt{m}}\mathbf{RFD}$ is used for all $j \in [d]$, where $\mathbf{R}$ denotes the random restriction matrix, $\mathbf{F}$ is the unitary DFT matrix scaled by $\sqrt{n_j}$,[7] and $\mathbf{D}$ is a diagonal matrix with Rademacher random variables forming its diagonal [13]. The embedded version of a test tensor $\mathcal{X}$ is always denoted by $L(\mathcal{X})$, and is calculated by

$$L(\mathcal{X}) = \begin{cases} \mathcal{X} \times_1 \mathbf{A}^{(1)} \times \cdots \times_d \mathbf{A}^{(d)}, & \text{1-stage JL} \\ \\ \mathbf{A}\left(\text{vec}\left(\mathcal{X} \times_1 \mathbf{A}^{(1)} \times \cdots \times_d \mathbf{A}^{(d)}\right)\right), & \text{2-stage JL} \end{cases} \tag{4.21}$$

where $\mathbf{A}$ is a JL matrix used in the 2$^{\text{nd}}$ stage. Obviously, $L(\mathcal{X})$ is a vector in the 2-stage case.

4.3.1 Effect of JL Embeddings on Norm

In this section, numerical results have been presented, showing the effect of mode-wise JL embedding on the norm of 3 MRI 3-mode images treated as generic tensors, as well as randomly generated data.

The compression ratio for the j^{th} mode, denoted by $c_1^{(j)}$, is defined as the compression in the size of each of the mode-j fibers, i.e.,

$$c_1^{(j)} = \frac{m_j}{n_j}.$$

The target dimension m_j in JL matrices is chosen as $m_j = \lceil c_1 n_j \rceil$ for all $j \in [d]$, to ensure that *at least* a fraction c_1 of the ambient dimension in each mode is preserved. In the experiments, the compression ratio is set to be the same for all modes, i.e., $c_1^{(j)} = c_1$ for all $j \in [d]$. In the case of a 2-stage JL embedding, the target dimension m of the secondary JL embedding is chosen as

$$m = \lceil c_2 N \rceil,$$

[7] Recall that n_j is the size of the mode-j fibers of the input tensor.

where c_2 is the compression ratio in the 2nd stage, and N is the length of the vectorized projected tensor after the modewise JL embedding. The total achieved compression is calculated by $c_{tot} = c_2 \left(\prod_{j=1}^{d} c_1^{(j)} \right)$. When the 2nd stage embedding is skipped, $c_{tot} = \prod_{j=1}^{d} c_1^{(j)}$. In all experiments of this section, when a 2-stage embedding is performed, $c_2 = 0.05$. Also, in figure legends, when two JL types are listed together, the first and second terms refer to the first and second stages, respectively. For example, in 'Gaussian+RFD', Gaussian and RFD JL embeddings were used in the first and second stages, respectively. The term 'vec' in the legends refers to vectorizing the data.

Assuming $\mathcal{X}$ denotes the original tensor and $L(\mathcal{X})$ is the projected result, the relative norm of $\mathcal{X}$ is defined by

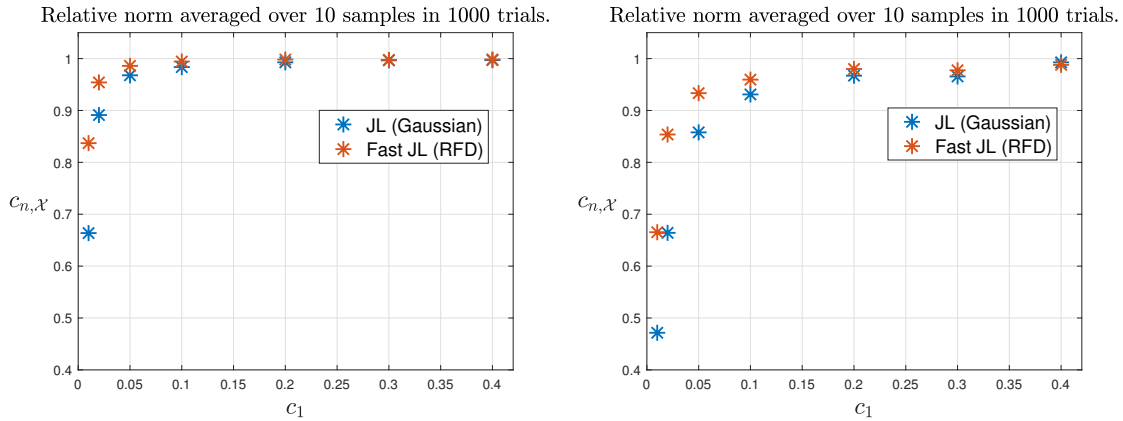
$$c_{n,\mathcal{X}} = \frac{\|L(\mathcal{X})\|}{\|\mathcal{X}\|}.$$

The results of this section depict the interplay between $c_{n,\mathcal{X}}$ and c_1 for randomly generated data, and $c_{n,\mathcal{X}}$ versus c_{tot} for MRI data, where the numbers have been averaged over 1000 trials, as well as over all samples for each value of c_1 or c_{tot}. In the case of Figure 4.2, 1000 randomly generated JL matrices were applied to each mode of all 10 randomly generated tensors. The results there indicate that the modewise embedding methods proposed herein still work on relatively coherent data despite the incoherence assumptions utilized in their theoretical analysis (recall Section 4.2). In Figure 4.3, 1000 JL embedding choices have been averaged over each of the 3 MRI images as well as the 3 images themselves. As expected, it can be observed in both figures that increasing the compression ratio leads to better norm (and distance) preservation.

The MRI data experiments were done using various combinations of JL matrices in the first and second stages, and were compared with the 1-stage (modewise) case and also JL applied to vectorized data. In Figure 4.3b, the runtime plots show that vectorizing the data before applying JL embeddings is the most computationally intensive way of compressing the data, although it preserves norms the best, as Figure 4.3a demonstrates. Due to the small mode sizes of the MRI data used in the experiments, modewise fast JL does not outperform modewise Gaussian JL in terms of computational efficiency in the modewise embeddings as one might initially expect (see the red

and blue curves). This is likely due to the fact that the individual mode sizes are too small to benefit from the FFT (recall all modes are ≤ 240 in size), together with the need of Fourier methods to use less efficient complex number arithmetic. However, when the 2-stage JL is employed for larger compression ratios, the vectorized data after the first stage compression is large enough to make the efficiency of fast JL over Gaussian JL embeddings clear (compare, e.g., the yellow and purple curves). Also the small sizes of modes make the use of explicitly constructed $\frac{1}{\sqrt{m}}\mathbf{RFD}$ matrices more efficient than taking the FFT of mode fibers.

It should be noted that in the second stage of the 2-stage JL throughout the experiments of this thesis, the matrix $\frac{1}{\sqrt{m}}\mathbf{RFD}$ is not constructed explicitly as this would be inefficient due to the large size of the vector that $\frac{1}{\sqrt{m}}\mathbf{RFD}$ is applied to. Instead, the signs of the vector are randomly changed (the effect of $\mathbf{D}$) followed by a Fourier transform (the effect of $\mathbf{F}$). Finally m samples of the resulting vector are picked at random with replacement (the effect of $\mathbf{R}$) after which the scale $1/\sqrt{m}$ is applied. This allows one to notice the computational efficiency of FFT in the fast JL embedding.

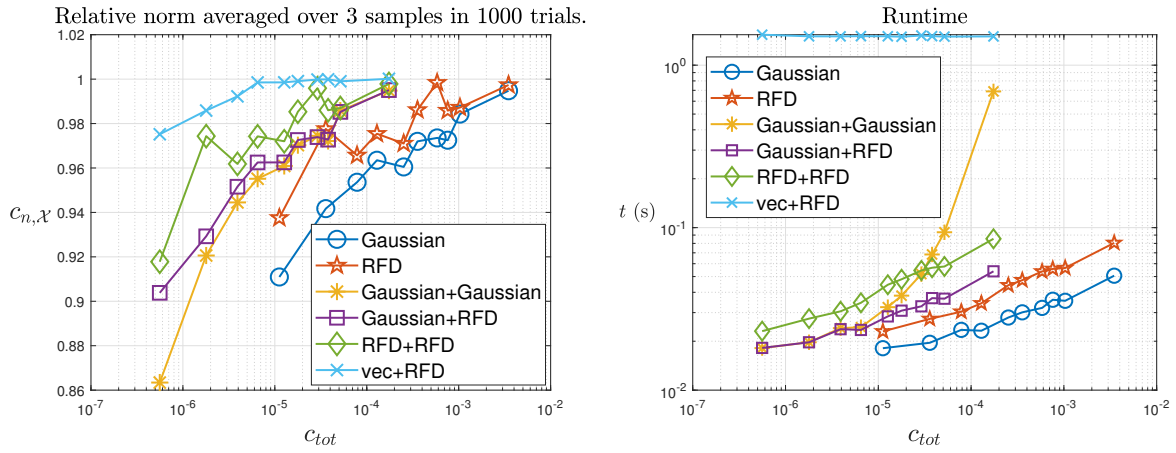


(a) (b)

Figure 4.2: Relative norm of randomly generated 4-dimensional data. Here, the total compression will be $c_{tot} = c_1^4$. (a) Gaussian data. (b) Coherent data. Note that the modewise approach still preserves norms well for the coherent data indicating that the incoherence assumptions utilized in Section 4.2 can likely be relaxed.

By looking at Figure 4.2 We observe that the proposed modewise JL approach leads to very good norm preservation for data generated from both coherent and incoherent factors. Specifically,

the compression listed on the horizontal axis is for one mode, and given that the synthesized data samples are 4-mode tensors, the total compression is very good.



(a) (b)

Figure 4.3: Simulation results averaged over 1000 trials for 3 MRI data samples, where each sample is 3-dimensional. In the 2-stage cases, $c_2 = 0.05$ has been used. (a) Relative norm. (b) Runtime.

Figure 4.3b depicts the results of the proposed modewise JL method on three MRI data samples. Although part (a) shows relative superiority of the 'vec+RFD' in terms of accuracy, part (b) suggests that for good compression values, the 1-stage and 2-stage JL approaches yields much smaller runtimes. The reason 'vec+RFD' is leading to an almost horizontal line in part (b) lies in the way $\frac{1}{\sqrt{m}}\mathbf{RFD}$ is applied. As the matrix is not formed explicitly and the only part that determines the compression is the restriction (which does not inflict any computational load if it is simply picking random samples from a vector), choosing various compression values does not alter the runtime. However, if one explicitly forms and applies $\frac{1}{\sqrt{m}}\mathbf{RFD}$, the runtime will change with the chosen compression.

CHAPTER 5

APPLICATIONS OF MODEWISE JOHNSON-LINDENSTRAUSS EMBEDDINGS

In this chapter, two cases are presented where modewise JL embeddings can be used to reduce the computational cost of computationally intensive problems.

5.1 Application to Least Squares Problems and CPD Fitting

Tensor decomposition problems usually involve fitting a low-rank approximation to a given tensor that is assumed to have a low rank of some type. In this section, it is shown that modewise JL embeddings offer an efficient way to reduce the computational cost of such fitting problems through dimension reduction at the cost of an approximation error.

Consider a tensor $\mathcal{X}$ which is assumed to have low CP rank r. We would like to approximate $\mathcal{X}$ in the Euclidean norm with a tensor $\mathcal{Y}$ expressed in the standard form as per (4.3). As mentioned in Chapter 3, a common fitting method is the Alternating Least Squares, where the factors representing the rank-r subspace are solved for one mode at a time. One can start from a random subspace and improve the least squares error mode by mode through multiple iterations. Since the subspace of interest is changing throughout the fitting process, oblivious subspace embeddings would be a natural choice to reduce the fitting problem size. For an arbitrary tensor $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$, the fitting process involves solving

$$\underset{\tilde{\mathbf{x}}_1^{(j)},\ldots,\tilde{\mathbf{x}}_r^{(j)} \in \mathbb{C}^{n_j}}{\arg\min} \left\| \mathcal{X} - \sum_{k=1}^{r} \alpha_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \right\| \tag{5.1}$$

for each mode $j \in [d]$ after fixing $\left\{ \mathbf{x}_k^{(\ell)} \right\}_{k \in [r], \ell \in [d] \setminus \{j\}}$. Here, $\mathbf{x}_k^{(j)} = \tilde{\mathbf{x}}_k^{(j)} / \|\tilde{\mathbf{x}}_k^{(j)}\|_2 \ \forall j, k$ and $\alpha_k = \prod_{\ell=1}^{d} \|\tilde{\mathbf{x}}_k^{(\ell)}\|_2$. One then varies j through all values in $[d]$ solving (5.1) for each j in order to update $\mathbf{x}_k^{(j)} \ \forall j, k$. Sweeping through all modes usually takes place in numerous iterations until convergence is achieved, meaning the fit stops to improve, or the maximum number of iterations is exhausted. This in turn means a high computational load, and makes it particularly important to

solve each least squares problem (5.1) efficiently by reducing the problem size. To see how this can be done to solve (5.1), one may write

$$\left\| \mathcal{X} - \sum_{k=1}^{r} \alpha_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \right\|^2 = \left\| \mathbf{X}_{(j)} - \sum_{k=1}^{r} \alpha_k \mathbf{x}_k^{(j)} \left(\bigotimes_{\substack{\ell=d \\ \ell \neq j}}^{1} \mathbf{x}_k^{(\ell)} \right)^\top \right\|_{\mathrm{F}}^2,$$

as the Euclidean norm of a tensor is equal to the Frobenius norm of any of its unfoldings. By looking closely at the right-hand side of the above equation, one can see that the Frobenius norm squared can be calculated row-wise (also note that the Frobenius norm is equivalent with the 2-norm for vectors, i.e., rows or columns of a matrix). Denoting row h of $\mathbf{X}_{(j)}$ by $\mathbf{x}_{j,h}$ and element h of $\mathbf{x}_k^{(\ell)}$ by $x_{k,h}^{(j)}$, we can write

$$\begin{aligned} \left\| \mathcal{X} - \sum_{k=1}^{r} \alpha_k \bigcirc_{\ell=1}^{d} \mathbf{x}_k^{(\ell)} \right\|^2 &= \sum_{h=1}^{n_j} \left\| \mathbf{x}_{j,h} - \sum_{k=1}^{r} \alpha_k x_{k,h}^{(j)} \left(\bigotimes_{\substack{\ell=d \\ \ell \neq j}}^{1} \mathbf{x}_k^{(\ell)} \right)^\top \right\|_2^2 \\ &= \sum_{h=1}^{n_j} \left\| \mathcal{X}^{(j,h)} - \sum_{k=1}^{r} \alpha'_{j,h,k} \bigcirc_{\substack{\ell=1 \\ \ell \neq j}}^{d} \mathbf{x}_k^{(\ell)} \right\|^2, \end{aligned} \tag{5.2}$$

where $\alpha'_{j,h,k} = \alpha_k x_{k,h}^{(j)}$ with α_k is known for $k \in [r]$ from (5.1) and $\mathcal{X}^{(j,h)}$ is the tensorized form of $\mathbf{x}_{j,h}$ which is in fact the h^{th} mode-j slice of $\mathcal{X}$. It is also clear that the original problem (5.1) can be modeled as n_j independent least squares problems that can be solved in parallel if needed, with each least squares problem involving a $(d-1)$-mode tensor $\mathcal{X}^{(j,h)}$. Essentially, this would mean that for mode j, one has to solve n_j minimization problems, each of the following form.

$$\underset{\boldsymbol{\alpha}'_{j,h} \in \mathbb{C}^r}{\arg\min} \left\| \mathcal{X}^{(j,h)} - \sum_{k=1}^{r} \alpha'_{j,h,k} \bigcirc_{\substack{\ell=1 \\ \ell \neq j}}^{d} \mathbf{x}_k^{(\ell)} \right\|. \tag{5.3}$$

Now, assuming that for each mode j, the factors $\{\mathbf{x}_k^{(\ell)}\}$ are sufficiently incoherent for $k \in [r]$ and $\ell \in [d] \setminus j$, we can use our modewise JL embedding method to solve a compressed version of each

least squares problem in (5.3) in the following way.

$$\underset{\boldsymbol{\alpha}'_{j,h}\in\mathbb{C}^r}{\arg\min}\left\|\mathcal{X}^{(j,h)}\bigtimes_{\substack{\ell'=1\\ \ell'\neq j}}^{d}\mathbf{A}^{(\ell')}-\sum_{k=1}^{r}\alpha'_{j,h,k}\bigcirc_{\substack{\ell=1\\ \ell\neq j}}^{d}\mathbf{x}_k^{(\ell)}\bigtimes_{\substack{\ell'=1\\ \ell'\neq j}}^{d}\mathbf{A}^{(\ell')}\right\| \tag{5.4}$$

We can then update each entry of $\tilde{\mathbf{x}}_k^{(j)}$ by setting $\tilde{x}_{k,h}^{(j)}=\alpha'_{j,h,k}/\alpha_k$ for all $h\in[n_j]$ and $k\in[r]$. To show that the solutions to (5.4) and (5.3) are close, we first establish that

$$\left\|\mathcal{X}^{(j,h)}\bigtimes_{\substack{\ell'=1\\ \ell'\neq j}}^{d}\mathbf{A}^{(\ell)}\right\|\approx\left\|\mathcal{X}^{(j,h)}\right\|$$

can also hold for all $j\in[d]$ and $h\in[n_j]$. This is done in the following lemma.

Lemma 5.1.1 *Let $\varepsilon\in(0,1)$, $\mathcal{Z}^{(1)},\ldots,\mathcal{Z}^{(p)}\in\mathbb{C}^{n_1\times\cdots\times n_d}$, and $\mathbf{A}^{(1)}\in\mathbb{C}^{m_1\times n_1}$ be an $(\varepsilon/\circledast d)$-JL embedding of the all $p\left(\prod_{\ell=2}^{d}n_\ell\right)$ mode-1 fibers of all p of these tensors,*

$$\mathcal{S}_1:=\bigcup_{t\in[p]}\left\{\mathcal{Z}^{(t)}_{:,i_2,\ldots,i_d}\mid\forall i_\ell\in[n_\ell],\ \ell\in[d]\setminus\{1\}\right\}\subset\mathbb{C}^{n_1},$$

into $\mathbb{C}^{m_1}$. Next, set $\mathcal{Z}^{(1,t)}:=\mathcal{Z}^{(t)}\times_1\mathbf{A}^{(1)}\in\mathbb{C}^{m_1\times n_2\times\cdots\times n_d}\ \forall t\in[p]$, and then let $\mathbf{A}^{(2)}\in\mathbb{C}^{m_2\times n_2}$ be an $(\varepsilon/\circledast d)$-JL embedding of all $p\left(m_1\prod_{\ell=3}^{d}n_\ell\right)$ mode-2 fibers

$$\mathcal{S}_2:=\bigcup_{t\in[p]}\left\{\mathcal{Z}^{(1,t)}_{i_1,:,i_3,\ldots,i_d}\mid\forall i_1\in[m_1]\ \&\ i_\ell\in[n_\ell],\ \ell\in[d]\setminus[2]\right\}\subset\mathbb{C}^{n_2}$$

into $\mathbb{C}^{m_2}$. Continuing inductively, for each $j\in[d]\setminus[2]$ and $t\in[p]$ set $\mathcal{Z}^{(j-1,t)}:=\mathcal{Z}^{(j-2,t)}\times_{j-1}\mathbf{A}^{(j-1)}\in\mathbb{C}^{m_1\times\cdots\times m_{j-1}\times n_j\times\cdots\times n_d}$, and then let $\mathbf{A}^{(j)}\in\mathbb{C}^{m_j\times n_j}$ be an $(\varepsilon/\circledast d)$-JL embedding of all $p\left(\prod_{\ell=1}^{j-1}m_\ell\right)\left(\prod_{\ell=j+1}^{d}n_\ell\right)$ mode-j fibers

$$\mathcal{S}_j:=\bigcup_{t\in[p]}\left\{\mathcal{Z}^{(j-1,t)}_{i_1,\ldots,i_{j-1},:,i_{j+1},\ldots,i_d}\mid\forall i_\ell\in[m_\ell],\ell\in[j-1]\ \&\ i_\ell\in[n_\ell],\ell\in[d]\setminus[j],\right\}\subset\mathbb{C}^{n_j}$$

into $\mathbb{C}^{m_j}$. Then,

$$\left|\left\|\mathcal{Z}^{(t)}\right\|^2-\left\|\mathcal{Z}^{(t)}\times_1\mathbf{A}^{(1)}\cdots\times_d\mathbf{A}^{(d)}\right\|^2\right|\ \leq\ \varepsilon\left\|\mathcal{Z}^{(t)}\right\|^2$$

will hold for all $t\in[p]$.

Proof *Fix* $t \in [p]$ *and let* $\mathcal{X}^{(0)} := \mathcal{Z}^{(t)}$, $\mathcal{X}^{(j)} := \mathcal{Z}^{(j,t)}$ *for all* $j \in [d-1]$, *and* $\mathcal{X}^{(d)} := \mathcal{Z}^{(d-1,t)} \times_d \mathbf{A}^{(d)} = \mathcal{Z}^{(t)} \times_1 \mathbf{A}^{(1)} \cdots \times_d \mathbf{A}^{(d)}$. *Choose any* $j \in [d]$, *and let* $\mathbf{x}_{j,h} \in \mathbb{C}^{n_j}$ *denote the* h^{th} *column of the mode-*j *unfolding of* $\mathcal{X}^{(j-1)}$, *denoted by* $\mathbf{X}_{(j)}^{(j-1)}$. *It is easy to see that each* $\mathbf{x}_{j,h}$ *is a mode-*j *fiber of* $\mathcal{X}^{(j-1)} = \mathcal{Z}^{(j-1,t)}$ *for each* $1 \leq h \leq N'_j := \left(\prod_{\ell=1}^{j-1} m_\ell\right)\left(\prod_{\ell=j+1} n_\ell\right)$. *Thus, we can see that*

$$\begin{aligned}
\left|\left\|\mathcal{X}^{(j-1)}\right\|^2 - \left\|\mathcal{X}^{(j)}\right\|^2\right| &= \left|\left\|\mathcal{X}^{(j-1)}\right\|^2 - \left\|\mathcal{X}^{(j-1)} \times_j \mathbf{A}^{(j)}\right\|^2\right| = \left|\left\|\mathbf{X}_{(j)}^{(j-1)}\right\|_{\mathrm{F}}^2 - \left\|\mathbf{A}^{(j)}\mathbf{X}_{(j)}^{(j-1)}\right\|_{\mathrm{F}}^2\right| \\
&= \left|\sum_{h=1}^{N'_j} \|\mathbf{x}_{j,h}\|_2^2 - \left\|\mathbf{A}^{(j)}\mathbf{x}_{j,h}\right\|_2^2\right| \leq \sum_{h=1}^{N'_j} \left|\|\mathbf{x}_{j,h}\|_2^2 - \|\mathbf{A}^{(j)}\mathbf{x}_{j,h}\|_2^2\right| \\
&\leq \frac{\varepsilon}{\mathbb{e}d} \sum_{h=1}^{N'_j} \|\mathbf{x}_{j,h}\|_2^2 = \frac{\varepsilon}{\mathbb{e}d} \left\|\mathbf{X}_{(j)}^{(j-1)}\right\|_{\mathrm{F}}^2 = \frac{\varepsilon}{\mathbb{e}d} \left\|\mathcal{X}^{(j-1)}\right\|^2.
\end{aligned}$$

A short induction argument now reveals that $\left\|\mathcal{X}^{(j)}\right\|^2 \leq \left(1 + \frac{\varepsilon}{\mathbb{e}d}\right)^j \left\|\mathcal{X}^{(0)}\right\|^2$ *holds for all* $j \in [d]$. *As a result we can now see that*

$$\begin{aligned}
\left|\left\|\mathcal{X}^{(0)}\right\|^2 - \left\|\mathcal{X}^{(d)}\right\|^2\right| &= \left|\sum_{j=1}^{d} \left\|\mathcal{X}^{(j-1)}\right\|^2 - \left\|\mathcal{X}^{(j)}\right\|^2\right| \leq \sum_{j=1}^{d} \left|\left\|\mathcal{X}^{(j-1)}\right\|^2 - \left\|\mathcal{X}^{(j)}\right\|^2\right| \leq \frac{\varepsilon}{\mathbb{e}d} \sum_{j=1}^{d} \left\|\mathcal{X}^{(j-1)}\right\|^2 \\
&\leq \frac{\varepsilon}{\mathbb{e}d} \sum_{j=1}^{d} \left(1 + \frac{\varepsilon}{\mathbb{e}d}\right)^{j-1} \left\|\mathcal{X}^{(0)}\right\|^2 \leq \frac{\varepsilon}{\mathbb{e}} \left(1 + \frac{\varepsilon}{\mathbb{e}d}\right)^d \left\|\mathcal{X}^{(0)}\right\|^2.
\end{aligned}$$

holds. The desired result now follows from Remark 4.2.1.

Now, we can use the result of Lemma 5.1.1 to prove that the solution to (5.4) will be a close approximation to that of (5.3) if the matrices $\mathbf{A}^{(j)}$ are chosen appropriately. We have the following general result which directly applies to least squares problems as per (5.4) when $\tilde{L}(\mathcal{Z}) := \mathcal{Z} \bigtimes_{\substack{\ell=1 \\ \ell \neq j}}^{d} \mathbf{A}^{(\ell)}$ and $\mathbf{A} = \mathbf{I}$.

Theorem 5.1.1 (Embeddings for Compressed Least Squares) *Let* $\mathcal{X} \in \mathbb{C}^{n_1 \times \cdots \times n_d}$, $\mathcal{L}$ *be an* r*-dimensional subspace of* $\mathbb{C}^{n_1 \times \cdots \times n_d}$ *spanned by a set of orthonormal basis tensors* $\{\mathcal{T}_k\}_{k \in [r]}$, *and* $\mathbb{P}_{\mathcal{L}^\perp} : \mathbb{C}^{n_1 \times \cdots \times n_d} \to \mathbb{C}^{n_1 \times \cdots \times n_d}$ *be the orthogonal projection operator on the orthogonal complement*

of $\mathcal{L}$. Fix $\varepsilon \in (0,1)$ and suppose that the linear operator $\tilde{L} : \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d} \to \mathbb{C}^{m_1 \times \cdots \times m_{d'}}$ has both of the following properties:

(i) $\tilde{L}$ is an $(\varepsilon/6)$-JL embedding of all $\mathcal{Y} \in \mathcal{L} \cup \{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\}$ into $\mathbb{C}^{m_1 \times \cdots \times m_{d'}}$, and

(ii) $\tilde{L}$ is an $(\varepsilon/24\sqrt{r})$-JL embedding of the $4r$ tensors

$$\mathcal{S}' := \bigcup_{k \in [r]} \left\{ \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} - \mathcal{T}_k, \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} + \mathcal{T}_k, \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} - \mathbb{i}\mathcal{T}_k, \frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|} + \mathbb{i}\mathcal{T}_k \right\} \subset \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d}$$

into $\mathbb{C}^{m_1 \times \cdots \times m_{d'}}$.

Furthermore, let vect $: \mathbb{C}^{m_1 \times \cdots \times m_{d'}} \to \mathbb{C}^{\prod_{\ell=1}^{d'} m_\ell}$ *be a reshaping vectorization operator, and $\mathbf{A} \in \mathbb{C}^{m \times \prod_{\ell=1}^{d'} m_\ell}$ be an $(\varepsilon/3)$-JL embedding of the $(r+1)$-dimensional subspace*

$$\mathcal{L}' := \operatorname{span}\left\{ \operatorname{vect} \circ \tilde{L}\left(\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\right),\ \operatorname{vect} \circ \tilde{L}\left(\mathcal{T}_1\right),\ \ldots,\ \operatorname{vect} \circ \tilde{L}\left(\mathcal{T}_r\right) \right\} \subset \mathbb{C}^{\prod_{\ell=1}^{d'} m_\ell}$$

into $\mathbb{C}^m$. Then,

$$\left| \left\| \mathbf{A}\left(\operatorname{vect} \circ \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right)\right) \right\|_2^2 - \|\mathcal{X} - \mathcal{Y}\|^2 \right| \leq \varepsilon \|\mathcal{X} - \mathcal{Y}\|^2$$

holds for all $\mathcal{Y} \in \mathcal{L}$.

Proof *Note that the theorem will be proven if $\tilde{L}$ is an $(\varepsilon/3)$–JL embedding of all tensors of the form $\{\mathcal{X} - \mathcal{Y} \mid \mathcal{Y} \in \mathcal{L}\}$ into $\mathbb{C}^{m_1 \times \cdots \times m_{d'}}$ since any such tensor $\mathcal{X} - \mathcal{Y}$ will also have* vect$\circ \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right) \in \mathcal{L}'$ *so that*

$$\begin{aligned}
&\left| \left\| \mathbf{A}\left(\operatorname{vect} \circ \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right)\right) \right\|_2^2 - \|\mathcal{X} - \mathcal{Y}\|^2 \right| \\
&\qquad \leq \left| \left\| \mathbf{A}\left(\operatorname{vect} \circ \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right)\right) \right\|_2^2 - \left\| \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right) \right\|^2 \right| + \left| \left\| \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right) \right\|^2 - \|\mathcal{X} - \mathcal{Y}\|^2 \right| \\
&\qquad \leq \left| \left\| \mathbf{A}\left(\operatorname{vect} \circ \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right)\right) \right\|_2^2 - \left\| \operatorname{vect} \circ \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right) \right\|_2^2 \right| + \frac{\varepsilon}{3} \|\mathcal{X} - \mathcal{Y}\|^2 \\
&\qquad \leq \frac{\varepsilon}{3} \left\| \operatorname{vect} \circ \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right) \right\|_2^2 + \frac{\varepsilon}{3} \|\mathcal{X} - \mathcal{Y}\|^2 \\
&\qquad = \frac{\varepsilon}{3} \left\| \tilde{L}\left(\mathcal{X} - \mathcal{Y}\right) \right\|^2 + \frac{\varepsilon}{3} \|\mathcal{X} - \mathcal{Y}\|^2 \\
&\qquad \leq \frac{\varepsilon}{3}\left(1 + \frac{\varepsilon}{3}\right) \|\mathcal{X} - \mathcal{Y}\|^2 + \frac{\varepsilon}{3} \|\mathcal{X} - \mathcal{Y}\|^2 \ \leq\ \varepsilon \|\mathcal{X} - \mathcal{Y}\|^2 .
\end{aligned}$$

Let $\mathbb{P}_{\mathcal{L}}$ be the orthogonal projection operator onto $\mathcal{L}$. Our first step in establishing that $\tilde{L}$ is an $(\varepsilon/3)$–JL embedding of all tensors of the form $\{\mathcal{X} - \mathcal{Y} \mid \mathcal{Y} \in \mathcal{L}\}$ into $\mathbb{C}^{m_1 \times \cdots \times m_{d'}}$ will be to show that $\tilde{L}$ preserves all the angles between $\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})$ and $\mathcal{L}$ well enough that the Pythagorean theorem

$$\|\mathcal{X} - \mathcal{Y}\|^2 \;=\; \|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X}) + \mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\|^2 \;=\; \|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|^2 + \|\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\|^2$$

still approximately holds for all $\mathcal{Y} \in \mathcal{L}$ after $\tilde{L}$ is applied. Toward that end, let $\boldsymbol{\gamma} \in \mathbb{C}^r$ be such that $\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y} = \sum_{k\in[r]} \gamma_k \mathcal{T}_k$ and note that $\|\boldsymbol{\gamma}\|_2 = \|\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\|$ due to the orthonormality of $\{\mathcal{T}_k\}_{k\in[r]}$. Appealing to Lemma 4.1.2 we now have that

$$\begin{aligned}
\left|\left\langle \tilde{L}\left(\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\right),\, \tilde{L}\left(\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\right)\right\rangle\right| \;&=\; \|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\| \left|\sum_{k\in[r]} \gamma_k \left\langle \tilde{L}(\mathcal{T}_k),\, \tilde{L}\left(\frac{\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})}{\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|}\right)\right\rangle\right| \\
&\leq\; \|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\| \left(\frac{\varepsilon}{6\sqrt{r}}\right) \sum_{k\in[r]} |\gamma_k| \;\leq\; \frac{\varepsilon}{6} \|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\| \, \|\boldsymbol{\gamma}\|_2 \\
&\leq \frac{\varepsilon}{12}\left(\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|^2 + \|\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\|^2\right) \;=\; \frac{\varepsilon}{12}\|\mathcal{X} - \mathcal{Y}\|^2.
\end{aligned} \tag{5.5}$$

Using (5.5) *we can now see that*

$$\begin{aligned}
&\left| \left\|\tilde{L}(\mathcal{X} - \mathcal{Y})\right\|_2^2 - \|\mathcal{X} - \mathcal{Y}\|^2 \right| \\
&\qquad = \left| \left\|\tilde{L}(\mathcal{X} - \mathcal{Y})\right\|_2^2 - \|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|^2 - \|\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\|^2 \right| \\
&\qquad \leq \left| \left\|\tilde{L}\left(\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\right)\right\|^2 - \|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|^2 \right| + \left| \left\|\tilde{L}\left(\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\right)\right\|^2 - \|\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\|^2 \right| \\
&\qquad\qquad + 2\left|\left\langle \tilde{L}\left(\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\right),\, \tilde{L}\left(\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\right)\right\rangle\right| \\
&\qquad \leq \frac{\varepsilon}{6}\left(\|\mathbb{P}_{\mathcal{L}^\perp}(\mathcal{X})\|^2 + \|\mathbb{P}_{\mathcal{L}}(\mathcal{X}) - \mathcal{Y}\|^2 + \|\mathcal{X} - \mathcal{Y}\|^2\right) \;=\; \frac{\varepsilon}{3}\|\mathcal{X} - \mathcal{Y}\|^2.
\end{aligned}$$

Thus, $\tilde{L}$ has the desired JL-embedding property required to conclude the proof.

5.1.1 Experiments: Effect of JL Embeddings on Least Squares Solutions

In this section, trial least squares experiments with compressed tensor data are performed to show the effect of modewise JL embeddings on solutions to least squares problems. In the experiments

of this section, the first sample of the three MRI data samples used in Section 4.3 is employed. Again, all experiments were carried out in MATLAB.

First, it is shown that this MRI sample has a relatively low-rank CP representations by plotting its CP reconstruction error for various choices of rank. Next, the effect of modewise JL on least squares solutions is investigated by solving for the coefficients of the CP decomposition of the MRI sample in a least squares problem. This will be done by performing 1-stage (modewise) and 2-stage JL on the data, which we call compressed least squares, and will be compared with the case where a regular uncompressed least squares problem is solved instead.

5.1.1.1 CPD Reconstruction

Before the experimental results, a short description of the basic form of CPD calculation is reviewd. Given a tensor $\mathcal{X}$, assume r is known beforehand. The problem is now the calculation of $\mathbf{x}_k^{(j)}$ for $j \in [d]$ and $k \in [r]$ and $\boldsymbol{\alpha}$ in (4.3), i.e. the solution to

$$\min_{\hat{\mathcal{X}}} \|\mathcal{X} - \hat{\mathcal{X}}\| \text{ with } \hat{\mathcal{X}} = \sum_{k=1}^{r} \alpha_k \, \mathbf{x}_k^{(1)} \bigcirc \mathbf{x}_k^{(2)} \bigcirc \cdots \bigcirc \mathbf{x}_k^{(d)}. \tag{5.6}$$

As the Euclidean norm a d-mode tensor is equal to the Frobenius norm of its mode-j unfoldings for $j \in [d]$, by letting $\mathbf{x}_k^{(j)}$ be the k^{th} column of a matrix $\mathbf{X}^{(j)} \in \mathbb{C}^{n_j \times r}$, the above minimization problem can be written as

$$\min_{\hat{\mathbf{X}}^{(j)}} \left\| \mathbf{X}_{(j)} - \hat{\mathbf{X}}^{(j)} \left(\mathbf{X}^{(d)} \odot \cdots \odot \mathbf{X}^{(j+1)} \odot \mathbf{X}^{(j-1)} \odot \cdots \odot \mathbf{X}^{(1)} \right)^{\top} \right\|_{\text{F}}$$

where $\hat{\mathbf{X}}^{(j)} = \mathbf{X}^{(j)} \text{diag}\,(\boldsymbol{\alpha})$, and the operator $\text{diag}(\cdot)$ creates a diagonal matrix with $\boldsymbol{\alpha}$ as its diagonal. Once solved for, the columns of $\hat{\mathbf{X}}^{(j)}$ can then be normalized and used to form the coefficients $\alpha_k = \prod_{j=1}^{d} \|\hat{\mathbf{x}}_k^{(j)}\|_2$ for $k \in [r]$, although this is optional, i.e., if the columns are not normalized, the coefficients α_k in the factorization will all be ones. This procedure is repeated iteratively until the fit ceases to improve (the objective function stops improving with respect to a tolerance) or the maximum number of iterations are exhausted. To choose the rank of the decomposition as well as

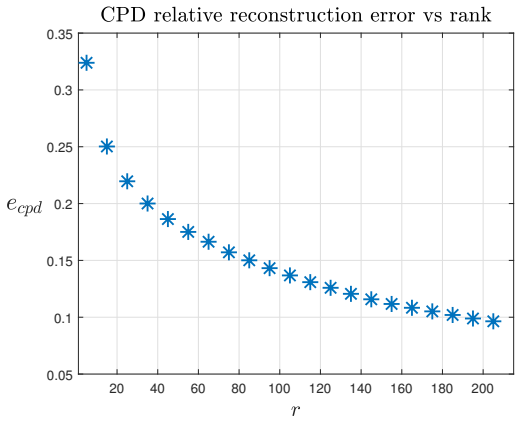


Figure 5.1: Relative reconstruction error of CPD calculated for different values of rank r for MRI data. As the rank increases, the error becomes smaller.

obtaining the best estimates for $\mathbf{X}^{(j)}$, a commonly used consistency diagnostic called CORCONDIA can be employed as explained in Section 3.1.2.

Now, the relative reconstruction error of CPD is calculated and plotted for various values of rank r. Assuming $\mathcal{X}$ represents the data, this error is defined as

$$e_{cpd} = \frac{\|\mathcal{X} - \hat{\mathcal{X}}\|}{\|\mathcal{X}\|},$$

where $\hat{\mathcal{X}}$ denotes the reconstruction of $\mathcal{X}$. Figure 5.1 displays the results.

5.1.1.2 Compressed Least Squares Performance

Let $\mathbf{x}_k^{(j)}$ be known in

$$\mathcal{X} \approx \sum_{k=1}^{r} \alpha_k \bigcirc_{j=1}^{d} \mathbf{x}_k^{(j)},$$

for $k \in [r]$ and $j \in [d]$. They can be obtained from a previous iteration in the CPD fitting procedure. Here, they come from the CPD of the data calculated in section 5.1.1.1. Also, assume these vectors have unit norms. In general, as stated in section 5.1.1.1, when $\mathbf{x}_k^{(j)}$ are obtained using a CPD algorithm, they do not necessarily have unit norms. Therefore, they are normalized and the norms are absorbed into the coefficients of CPD. In other words, $\alpha_k = \prod_{j=1}^{d} \|\mathbf{x}_k^{(j)}\|_2$ for $k \in [r]$. If the normalization of the vectors is not performed, $\alpha_k = 1$ for $k \in [r]$. The coefficients of the CPD

fit are the solutions to the following least squares problem,

$$\boldsymbol{\alpha} = \arg\min_{\boldsymbol{\beta}} \left\| \mathcal{X} - \sum_{k=1}^{r} \beta_k \bigcirc_{j=1}^{d} \mathbf{x}_k^{(j)} \right\|.$$

As normalization of $\mathbf{x}_k^{(j)}$ was not performed when computing the CPD of the data in these experiments, the true solution will be $\boldsymbol{\alpha} = \mathbf{1}$. An approximate solution for the coefficients can be obtained by solving for

$$\boldsymbol{\alpha}_P = \arg\min_{\boldsymbol{\beta}} \left\| L(\mathcal{X}) - L\left(\sum_{k=1}^{r} \beta_k \bigcirc_{j=1}^{d} \mathbf{x}_k^{(j)} \right) \right\|,$$

where $\boldsymbol{\alpha}_P$ is the vector $\boldsymbol{\alpha}$ estimated for randomly projected data, and $L(\mathcal{X})$ is defined as per (4.21). This is in fact simply another way of demonstrating that solving (5.4) yields an approximate solution to (5.3) for a $(d-1)$-mode tensor. Of course, both of these problems can be solved using the vectorized versions of the tensors instead. Indeed, for $\boldsymbol{\alpha}_P$, vectorization should be done after random projection of $\mathcal{X}$ and the rank-1 tensors, i.e.,

$$\boldsymbol{\alpha}_P = \arg\min_{\boldsymbol{\beta}} \|\mathbf{x}_P - \mathbf{B}\boldsymbol{\beta}\|_2 = (\mathbf{B}^*\mathbf{B})^{-1}\mathbf{B}^*\mathbf{x}_P = \mathbf{B}^\dagger \mathbf{x}_P,$$

where $\mathbf{B}^\dagger$ denotes the pseudo-inverse of $\mathbf{B}$, $\mathbf{x}_P = \text{vec}(L(\mathcal{X}))$, and $\mathbf{B}$ is a matrix whose k^{th} column is $\text{vec}\left(L\left(\bigcirc_{j=1}^{d}\mathbf{x}_k^{(j)}\right)\right)$[1] for $k \in [r]$.[2] The error measure used to evaluate the approximate solution is defined as

$$e_r = \left| \frac{e_P - e_T}{e_T} \right|,$$

where $e_T = \left\| \mathcal{X} - \sum_{k=1}^{r} \alpha_k \bigcirc_{j=1}^{d} \mathbf{x}_k^{(j)} \right\|$ and $e_P = \left\| \mathcal{X} - \sum_{k=1}^{r} \alpha_{P,k} \bigcirc_{j=1}^{d} \mathbf{x}_k^{(j)} \right\|$. This in fact compares the true CPD reconstruction error and the reconstruction error calculated using the approximate solution for the CPD coefficients $\boldsymbol{\alpha}_P$. The results are shown in Figure 5.2.

[1] Again, it is clear that in the 2-stage case, $L(\mathcal{X})$ and $L\left(\bigcirc_{j=1}^{d}\mathbf{x}_k^{(j)}\right)$ are vectors, and therefore, the operator vec $(\cdot)$ does not change the result.

[2] The backslash operator was used to actually solve the resulting least squares problems in MATLAB.

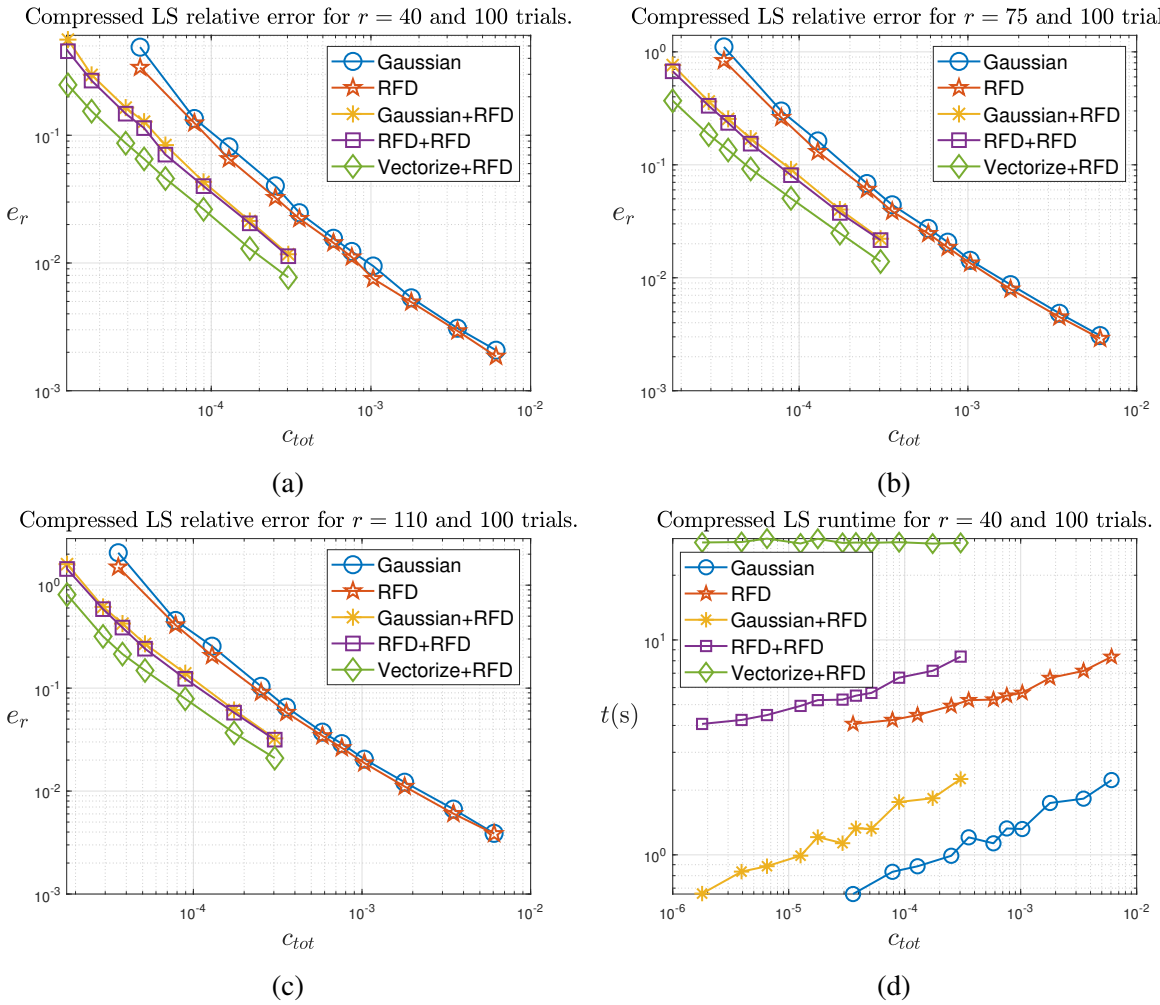


Figure 5.2: Effect of JL embeddings on the relative reconstruction error of least squares estimation of CPD coefficients. In the 2-stage cases, $c_2 = 0.05$ has been used. (a) $r = 40$. (b) $r = 75$. (c) $r = 110$. (d) Average runtime for $r = 40$. The other runtime plots for $r = 75$ and $r = 110$ are qualitatively identical.

In Figure 5.2, the compressed least squares results for the aforementioned MRI data sample have been plotted. We can observe that as we choose a higher rank for the CPD model, we obtain a smaller error in the estimated coefficients of $\boldsymbol{\alpha}$. As expected, the 'Vectorize+RFD' case yields the most accurate results by a small margin. However, its runtime is considerably larger due to the huge size of the vectorized tensor, although the it is benefiting from the computational efficiency of the FFT.[3] To see why the runtime plot is almost flat regardless of the chosen compression, see

[3]For information about how RFD is applied after vectorization, refer to Section 4.3.1.

the discussion at the end of Section 4.3.1.

5.2 Application to Many-Body Purturbation Theory Problems

This section provides a framework for modeling the energy correction terms as the sum of multiple inner products between tensors, so that each inner product can be approximated according to the geometry preserving property of JL embeddings as outlined in Lemma 4.1.1. The idea is to calculate the inner product of tensors with reduced dimensions to obtain an approximate value of the true energy terms. In doing so, it is assumed that the data lie on a low-rank inner product space of tensors.

5.2.1 Second-order energy correction

The 2$^{\text{nd}}$-order energy correction term is defined as:

$$E^{(2)} = \sum_{J=0}^{N_J-1} E^{(2)}(J), \tag{5.7}$$

where N_J is the number of blocks, and

$$E^{(2)}(J) = -\frac{1}{4}(2J+1) \sum_{ijkl} \mathcal{H}_{ijkl}\, \mathcal{H}_{klij} \mathcal{D}_{ijkl} = -\frac{1}{4}(2J+1)\, \langle \mathcal{H}, \tilde{\mathcal{H}} \rangle, \tag{5.8}$$

in which the Hamiltonian tensor $\mathcal{H} \in \mathbb{R}^{n\times n\times n\times n}$ should be updated for each value of J, and $\mathcal{D}$ has the same dimensions as $\mathcal{H}$ and is calculated from single-particle energy values. The tensor $\tilde{\mathcal{H}}$ is a permuted version of the $\mathcal{H}$ multiplied component-wise by $\mathcal{D}$, i.e.,

$$\tilde{\mathcal{H}}_{ijkl} = \mathcal{H}_{klij} \mathcal{D}_{ijkl}. \tag{5.9}$$

Now, an approximation of (5.8) can be computed by randomly projecting $\mathcal{H}$ and $\tilde{\mathcal{H}}$ onto a lower-dimensional space using mode-wise Johnson-Lindenstrauss embeddings:

$$\mathcal{S} = \mathcal{H} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(2)} \times_4 \mathbf{A}^{(4)}, \tag{5.10}$$

$$\tilde{\mathcal{S}} = \tilde{\mathcal{H}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(2)} \times_4 \mathbf{A}^{(4)}, \tag{5.11}$$

where $\mathbf{A}^{(j)} \in \mathbb{R}^{m_j \times n}$ are JL matrices and $m_j \leq n$ for $j \in [4]$. Now, with high probability,

$$\langle \mathcal{H}, \tilde{\mathcal{H}} \rangle \approx \langle \mathcal{S}, \tilde{\mathcal{S}} \rangle, \tag{5.12}$$

to within an adjustable error that is related to the target dimension sizes m_j.

A 2nd stage JL embedding can be applied to the vectorized versions of $\mathcal{S}$ and $\tilde{\mathcal{S}}$ to further compress the projected tensors before computing the approximate inner product. This is done according to

$$\mathbf{s}_p = \mathbf{A}\text{vect}\left(\mathcal{S}\right), \tag{5.13}$$

where $\mathbf{s}_p \in \mathbb{R}^m$ and $\mathbf{A} \in \mathbb{R}^{m \times \prod_{j=1}^{d} m_j}$.

Note 5.2.1 *It is observed that assuming real arithmetic, the operations count to directly caclulate the inner product is $O(n_1 \ldots n_d)$ while the computational complexity of a one-stage JL embedding is $O(m_1 n_1 \ldots n_d)$ as discussed in Section 4.2.1.1, which is obviously higher. However, the same compressed tensors can be used in the process of calculating many observables including the higher-order purturbative terms such as the third-order energy correction and radius corrections. As the number of such terms increases, the overall computational complexity will become much lower when compressed tensors are used to approximate observables.*

5.2.2 Radius Corrections

In what follows, a one-stage (modewise) JL compression scheme is discussed. Obviously, in both cases shown below, a second stage JL compression can also be performed after vectorizing the result of the first stage.

Particle Term:

The one-body particle term is expressed in the following way

$$\begin{aligned} R_1 &= \frac{1}{2} \sum_{ijklm} \mathcal{H}_{ijkl} \mathcal{D}_{ijkl} \mathcal{D}_{mjkl} \mathcal{H}_{klmj} R_{mi} \\ &= \frac{1}{2} \sum_{ijkl} \mathcal{H}_{ijkl} \mathcal{D}_{ijkl} \sum_{m} \mathcal{D}_{mjkl} \mathcal{H}_{klmj} R_{mi} \\ &= \frac{1}{2} \left\langle \check{\mathcal{H}}, \hat{\mathcal{H}} \right\rangle, \end{aligned} \tag{5.14}$$

where $\check{\mathcal{H}}$ is obtained by the component-wise product of $\mathcal{H}$ and $\mathcal{D}$, i.e., $\check{\mathcal{H}}_{ijkl} = \mathcal{H}_{ijkl} \mathcal{D}_{ijkl}$,

$$\hat{\mathcal{H}}_{ijkl} = \sum_{m} \mathcal{D}_{mjkl} \mathcal{H}_{klmj} R_{mi} = \sum_{m} \tilde{\mathcal{H}}_{mjkl} R_{mi}, \tag{5.15}$$

and $\mathbf{R}$ is the radius operator and a square matrix. Here, $\tilde{\mathcal{H}}$ is defined in (5.9). We can observe that $\hat{\mathcal{H}} = \tilde{\mathcal{H}} \times_1 \mathbf{R}^\top$. Therefore, the approximate correction term would be calculated as

$$R_1 \approx \frac{1}{2} \left\langle \mathcal{H}_{p_1}, \mathcal{H}_{p_2} \right\rangle,$$

where

$$\mathcal{H}_{p_1} = \check{\mathcal{H}} \bigtimes_{\ell=1}^{4} \mathbf{A}^{(\ell)}, \tag{5.16}$$

and

$$\mathcal{H}_{p_2} = \hat{\mathcal{H}} \bigtimes_{\ell=1}^{4} \mathbf{A}^{(\ell)} = \tilde{\mathcal{H}} \times_1 (\mathbf{A}^{(1)} \mathbf{R}^\top) \bigtimes_{\ell=2}^{4} \mathbf{A}^{(\ell)}. \tag{5.17}$$

Hole Term:

Calculations for the one-body hole term are very similar to the first term, as shown below.

$$R_2 = \frac{1}{2} \sum_{ijklm} \mathcal{H}_{ijkl} \mathcal{D}_{ijkl} \mathcal{H}_{mlij} \mathcal{D}_{ijml} R_{km} \tag{5.18}$$

$$= \frac{1}{2} \sum_{ijkl} \mathcal{H}_{ijkl} \mathcal{D}_{ijkl} \sum_{m} \mathcal{D}_{ijml} \mathcal{H}_{mlij} R_{km} \tag{5.19}$$

$$= \frac{1}{2} \left\langle \check{\mathcal{H}}, \bar{\mathcal{H}} \right\rangle, \tag{5.20}$$

where

$$\bar{\mathcal{H}}_{ijkl} = \sum_{m} \mathcal{D}_{ijml} \mathcal{H}_{mlij} R_{km} = \sum_{m} \tilde{\mathcal{H}}_{ijml} R_{km}. \tag{5.21}$$

We have that

$$\bar{\mathcal{H}} = \tilde{\mathcal{H}} \times_3 \mathbf{R}. \tag{5.22}$$

Therefore,

$$R_2 \approx \frac{1}{2} \left\langle \mathcal{H}_{p_1}, \mathcal{H}_{p_2} \right\rangle, \tag{5.23}$$

where

$$\mathcal{H}_{p_1} = \check{\mathcal{H}} \bigtimes_{\ell=1}^{4} \mathbf{A}^{(\ell)} \tag{5.24}$$

as in the case of the first correction term, and

$$\begin{aligned} \mathcal{H}_{p_2} &= \bar{\mathcal{H}} \bigtimes_{\ell=1}^{4} \mathbf{A}^{(\ell)} \\ &= \tilde{\mathcal{H}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 (\mathbf{A}^{(3)}\mathbf{R}) \times_4 \mathbf{A}^{(4)}. \end{aligned} \tag{5.25}$$

It can be observed that all one needs to calculate the approximations to R_1 and R_2 is the two tensors $\check{\mathcal{H}}$ and $\tilde{\mathcal{H}}$. This has been depicted in the block diagram of Figure 5.3. In many cases, due to the symmetry in $\mathcal{H}$, we have that $\check{\mathcal{H}} = \tilde{\mathcal{H}}$ which further reduces the storage requirements.

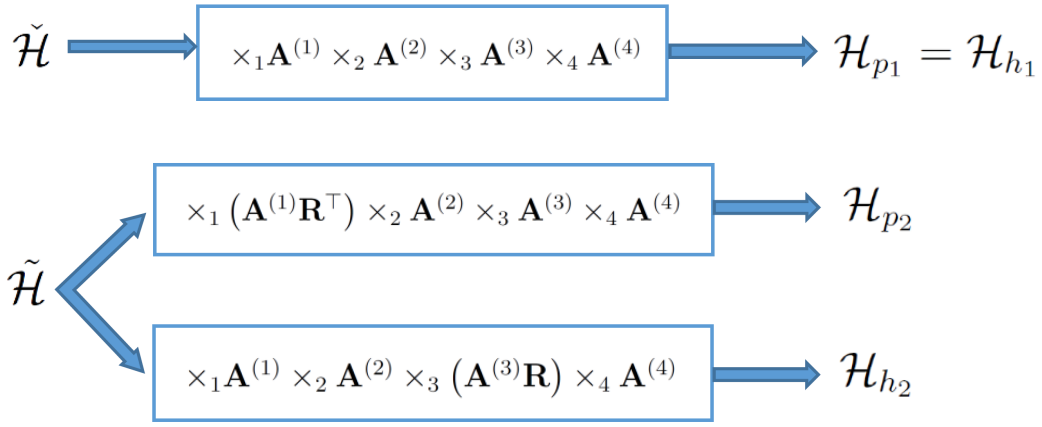


Figure 5.3: A block diagram showing how the approximations to R_1 and R_2 are calculated.

5.2.3 Third-order energy correction

The 3rd-order energy correction term is defined as

$$E^{(3)} = \sum_{J=0}^{N_J-1} E^{(3)}(J), \tag{5.26}$$

where the term $E^{(3)}(J)$ is calculated in each of the following settings.

5.2.3.1 Particle-Particle

In the particle-particle case,

$$E^{(3)}(J) = \frac{1}{8}(2J+1) \sum_{i,j,k,l,m,n} \mathcal{H}(i,j,k,l)\,\mathcal{H}(k,l,m,n)\,\mathcal{H}(m,n,i,j)\,\mathcal{D}(k,l,i,j)\,\mathcal{D}(m,n,i,j). \tag{5.27}$$

Again, the Hamiltonian tensor $\mathcal{H}$ should be updated for each value of J. To calculate the sum, one can use a scheme similar to the one used in section 5.2.1, but this time with 6 dimensions: generate 6-dimensional tensors by regrouping the terms in (5.27) as $\mathcal{H}_1$ and $\mathcal{H}_2$, and then calculate the inner product between $\mathcal{H}_1$ and $\mathcal{H}_2$. There are multiple ways to group the terms in (5.27). In the following, two grouping options are listed.

$$\text{Option 1}: \begin{cases} \mathcal{H}_1(i,j,k,l,m,n) = \frac{1}{8}\mathcal{H}(i,j,k,l)\mathcal{H}(k,l,m,n) \\ \mathcal{H}_2(i,j,k,l,m,n) = \mathcal{H}(m,n,i,j)\mathcal{D}(k,l,i,j)\mathcal{D}(m,n,i,j), \end{cases} \tag{5.28}$$

$$\text{Option 2}: \begin{cases} \mathcal{H}_1(i,j,k,l,m,n) = \frac{1}{8}\mathcal{H}(i,j,k,l)\mathcal{H}(k,l,m,n)\mathcal{D}(k,l,i,j) \\ \mathcal{H}_2(i,j,k,l,m,n) = \mathcal{H}(m,n,i,j)\mathcal{D}(m,n,i,j). \end{cases} \tag{5.29}$$

Now, if $\mathcal{S}_1$ and $\mathcal{S}_2$ are the projected versions of $\mathcal{H}_1$ and $\mathcal{H}_2$, we expect that

$$E^{(3)}(J) = (2J+1)\,\langle \mathcal{H}_1, \mathcal{H}_2 \rangle \approx (2J+1)\,\langle \mathcal{S}_1, \mathcal{S}_2 \rangle.$$

The problem with this approach lies in the fact that when the dimension sizes increase, the 6-mode tensors become problematic in terms of storage. For instance, for $\mathcal{H} \in \mathbb{R}^{100\times100\times100\times100}$,

7.45 TB of space is needed to store each of $\mathcal{H}_1$ and $\mathcal{H}_2$. To overcome this problem, we can reshape the tensors and perform the projections as explained below.

It is observed that the indices of the hypothetical 6-mode tensors always appear in groups of two in the inner product summation. Therefore, they can be reshaped into 3-mode tensors by regrouping the indices, and one may perform mode-wise JL on the reshaped tensors.

$$\text{Option 1}: \begin{cases} \mathcal{H}_1(p,q,r) = \frac{1}{8}\mathcal{H}(p,q)\mathcal{H}(q,r) \\ \mathcal{H}_2(p,q,r) = \mathcal{H}(r,p)\mathcal{D}(q,p)\mathcal{D}(r,p) = \tilde{\mathcal{H}}(r,p)\mathcal{D}(q,p), \end{cases} \tag{5.30}$$

where $\tilde{\mathcal{H}}$ is defined similarly as in (5.9). Here, p represents all relevant pairs of i and j, q encodes all pairs of k and l, and r represents all pairs of m and n in the grouping operation[4]. The repetitive patterns existing in 3-mode tensors that now can be formed using combinations of matrices, as well as reducing the size of two modes at once, as a result of combining two indices into one index, will provide the tools to avoid dealing with extremely large tensors when performing the projections. For instance, to project $\mathcal{H}_1$ in (5.30), one must calculate

$$\mathcal{P}_1\left(i_1,i_2,i_3\right) = \sum_{p,q,r} \mathcal{H}_1(p,q,r)\mathbf{A}^{(1)}\left(i_1,p\right)\mathbf{A}^{(2)}\left(i_2,q\right)\mathbf{A}^{(3)}\left(i_3,r\right),$$

which is the element-wise version of

$$\mathcal{P}_1 = \mathcal{H}_1 \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}.$$

According to the way $\mathcal{H}_1$ is defined it will be possible to decompose the triple summation into separate sums for two of the mode-wise projections. This way, one can obtain the fully-projected tensor $\mathcal{P}_1$ by only dealing with 2-mode (partially) compressed arrays. Algebraic details on how the mode-wise projections can be done in a memory efficient way are presented in Appendix B.1.

Due to the resemblance of options 1 and 2 in terms of the method used, only option 1 will be considered for future experiments, and also in the Hole-Hole and Particle-Hole settings, only one option will be discussed.

[4]For instance, if column-major formatting is used, and assuming the indices start from 1, the relation between p, i and j is $p = i + (j-1)N$, where $i, j \in [N]$.

5.2.3.2 Hole-Hole

In this case, the energy term for each value of J is expressed by

$$E^{(3)}(J) = \frac{1}{8}(2J+1) \sum_{i,j,k,l,m,n} \mathcal{H}(i,j,k,l)\, \mathcal{H}(k,l,m,n)\, \mathcal{H}(m,n,i,j)\, \mathcal{D}(m,n,i,j)\, \mathcal{D}(m,n,k,l). \tag{5.31}$$

For simplicity, only one option will be used to form $\mathcal{H}_1$ and $\mathcal{H}_2$, where

$$\begin{cases} \mathcal{H}_1(p,q,r) = \frac{1}{8}\mathcal{H}(p,q)\mathcal{H}(q,r) \\ \mathcal{H}_2(p,q,r) = \mathcal{H}(r,p)\mathcal{D}(r,p)\mathcal{D}(r,q) = \tilde{\mathcal{H}}(r,p)\mathcal{D}(r,q). \end{cases} \tag{5.32}$$

Again, i and j are combined to form p, k and l are grouped to form q, and m and n are combined to form r, as explained above. Details on the calculations of mode-wise projections are presented in Appendix B.2.

5.2.3.3 Particle-Hole

In this case, the energy term for each value of J is calculated by

$$E^{(3)}(J) = (2J+1) \sum_{i,j,k,l,m,n} \mathcal{H}_p(i,j,k,l)\, \mathcal{H}_p(k,l,m,n)\, \mathcal{H}_p(m,n,i,j)\, \mathcal{D}(k,j,l,i)\, \mathcal{D}(j,m,i,n), \tag{5.33}$$

where the Hamiltonians are obtained after a Pandya transform shown by the subscript p in the summation. To make the process of reshaping the data into 3-mode tensors possible, the dimensions of $\mathcal{D}$ should be permuted to get

$$\mathcal{D}_1(i,j,k,l) = \mathcal{D}(k,j,l,i)$$

$$\mathcal{D}_2(m,n,i,j) = \mathcal{D}(j,m,i,n).$$

Then, we can choose

$$\begin{cases} \tilde{\mathcal{H}}_1(i,j,k,l) = \mathcal{H}_p(i,j,k,l)\mathcal{D}_1(i,j,k,l) \\ \tilde{\mathcal{H}}_2(m,n,i,j) = \mathcal{H}_p(m,n,i,j)\mathcal{D}_2(m,n,i,j), \end{cases} \tag{5.34}$$

eMax	4	6	8	10	12	14
n	30	56	90	132	174	216

Table 5.1: Basis truncation parameters and mode dimensions for single-particle bases labeled by eMax.

leading to the reshaped version

$$\begin{cases} \mathcal{H}_1(p,q,r) = \tilde{\mathcal{H}}_1(p,q)\mathcal{H}_p(q,r) \\ \mathcal{H}_2(p,q,r) = \tilde{\mathcal{H}}_2(r,p). \end{cases} \tag{5.35}$$

Memory efficient calculations for the mode-wise projections can be found in Appendix B.3.

5.2.4 Experiments

In this section, numerical results are provided to demonstrate how mode-wise JL embeddings affect the accuracy of energy calculations. Experiments are done for different data sizes, i.e., for $\mathcal{H}, \mathcal{D} \in \mathbb{R}^{n\times n\times n\times n}$ where the dimension size n is chosen from the set of number listed in Table 5.1. In each case, the relative error in $E^{(2)}$, $E^{(3)}$, and the radius correction terms R_1 and R_2 defined by (5.36), (5.37), and (5.38), and are plotted for various values of compression.

$$\overline{\Delta E^{(2)}} = \text{mean}\left(\left| \frac{E_p^{(2)} - E^{(2)}}{E^{(2)}} \right| \right). \tag{5.36}$$

$$\overline{\Delta E^{(3)}} = \text{mean}\left(\left| \frac{E_p^{(3)} - E^{(3)}}{E^{(3)}} \right| \right). \tag{5.37}$$

$$\overline{\Delta R} = \text{mean}\left(\left| \frac{R_p - R}{R} \right| \right). \tag{5.38}$$

where the subscript p is used to denote the corresponding value calculated after the projection of tensors, and mean(X) denotes the mean of X. Compression in mode j is defined by

$$c_j = \frac{m_j}{N}, \tag{5.39}$$

where N and m_j denote the size of mode j before and after projection, respectively. The target dimension m_j in JL matrices is chosen as $m_j = \lceil c_j n \rceil$ for all j, to ensure that at least a fraction c_j of the ambient dimension in mode j is preserved. In the experiments, compression is chosen the same for all modes, i.e., $c_j = c$ for all j. It should be noted that for the $E^{(3)}$ calculations, the size of each dimension in the reshaped data is $N = n^2$, while for the $E^{(2)}$ and R calculations, $N = n$.

5.2.4.1 $E^{(2)}$ Experiments

Experiment results for O16 have been plotted in Figures 5.4, 5.5, and 5.6. In Figure 5.7, the relative error in $E^{(2)}$ has been plotted for two compression levels for O16 and Sn132. These results clearly show that JL embeddings result in smaller error values when data size increases. This dependence is almost log-linear.

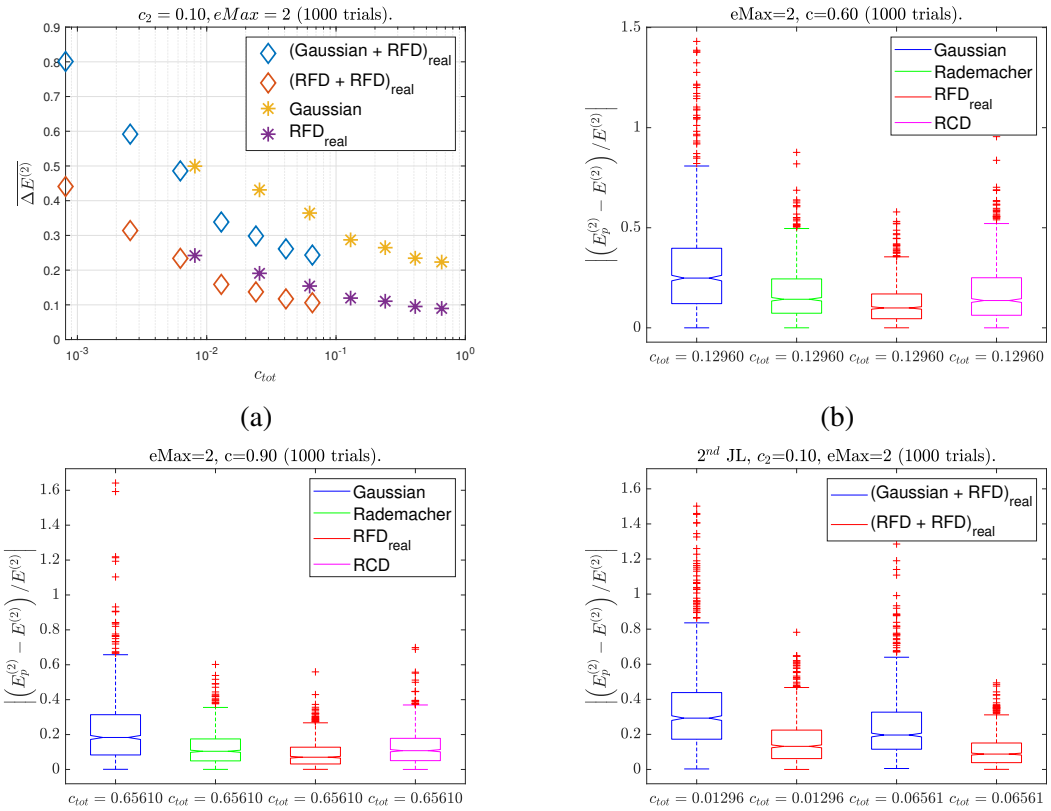


Figure 5.4: $E^{(2)}$ experiment results for O16, $eMax = 2$.

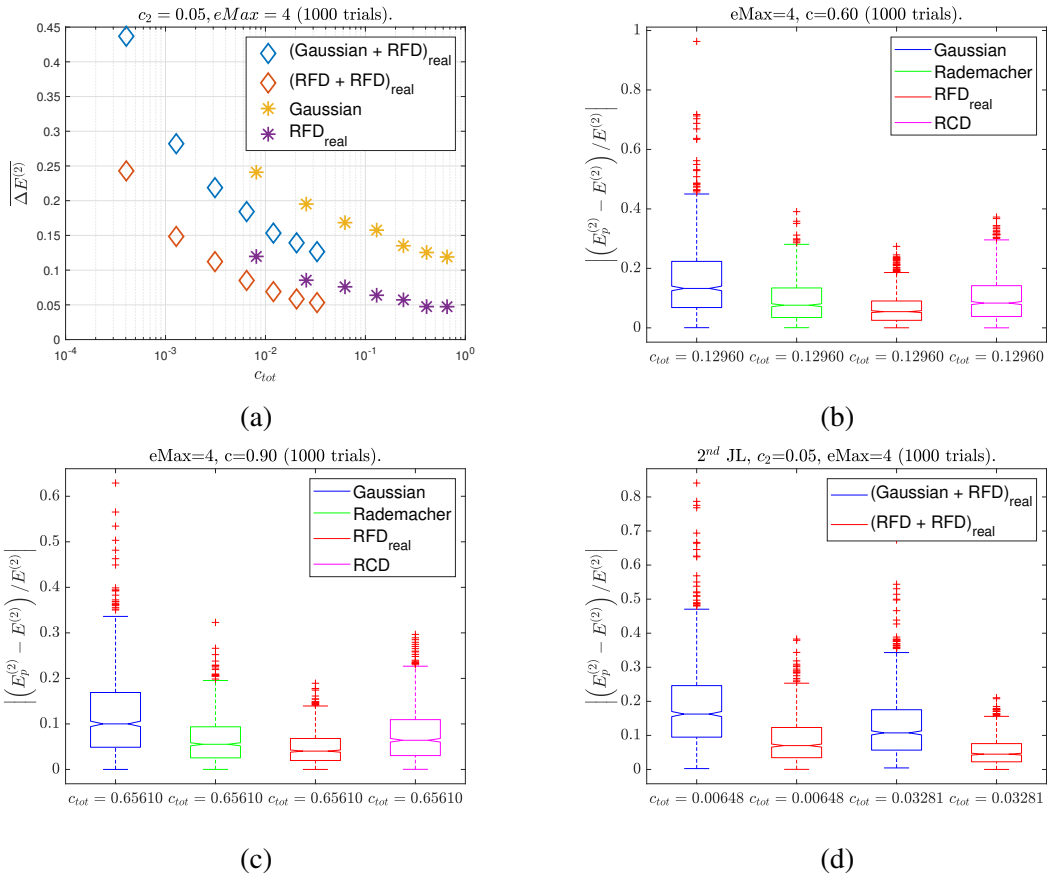


Figure 5.5: $E^{(2)}$ experiment results for O16, $eMax = 4$.

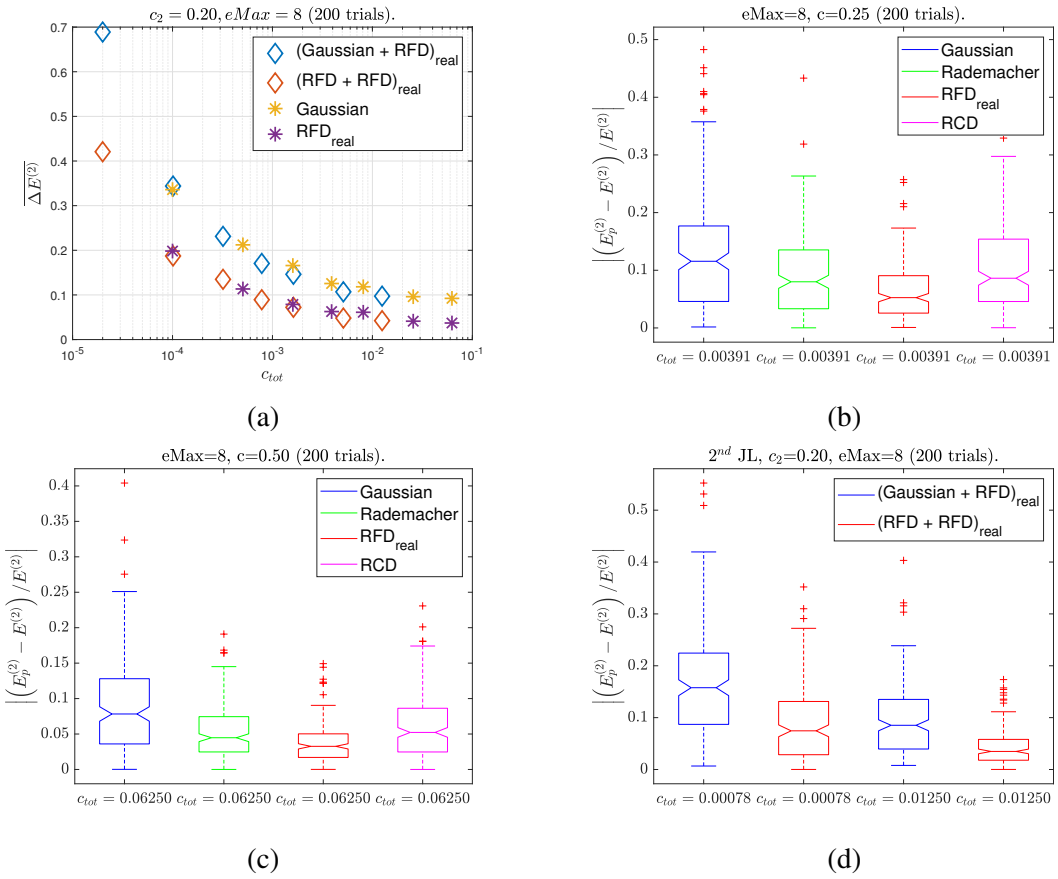


Figure 5.6: $E^{(2)}$ experiment results for O16, $eMax = 8$.

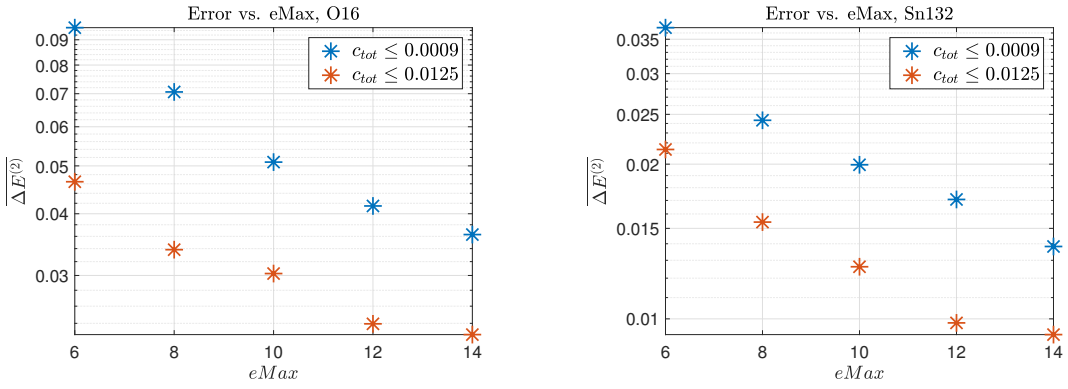


Figure 5.7: Relative error in $E^{(2)}$ for total compression values of 0.0009 and 0.0125. (a) O16. (b) Sn132.

5.2.4.2 Radius Correction Experiments

The experiments of this section were done on the data of Tin (Sn132) and Calcium (Ca48) for n = 216 or eMax= 14. The results can be viewed in Figure 5.8.

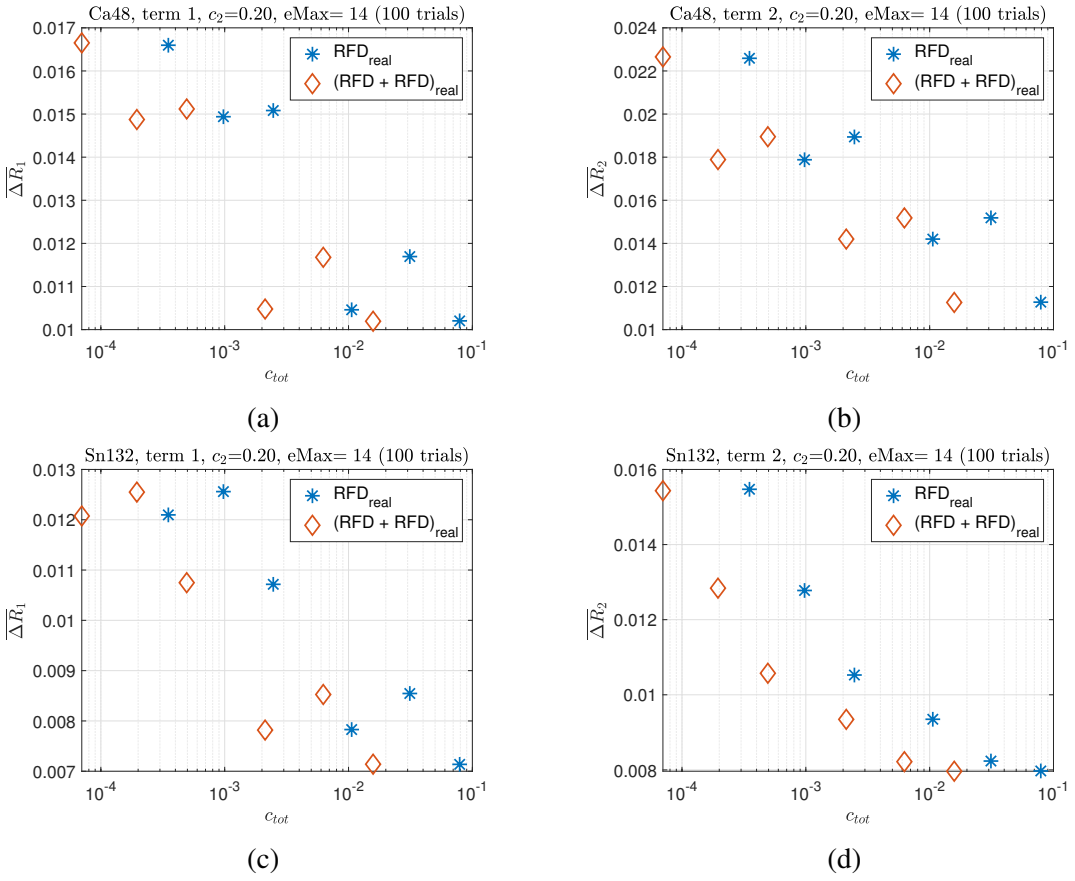
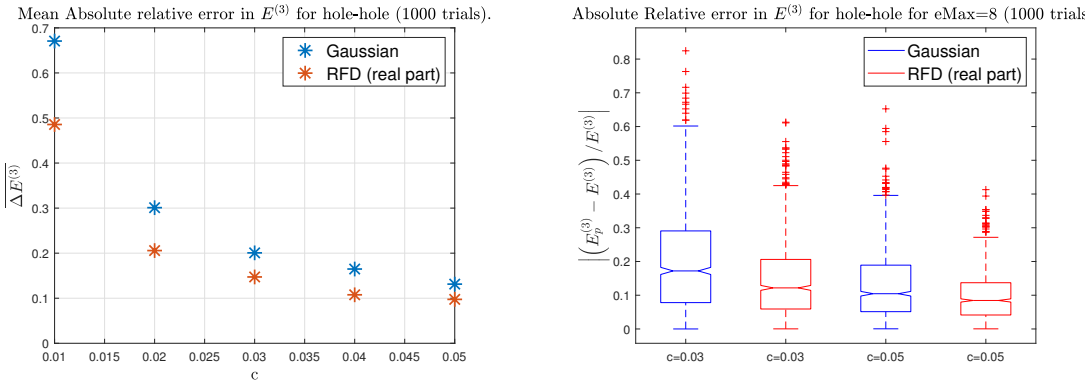


Figure 5.8: Radius correction results, for interaction em1.8 – 2.0 and eMax= 14. (a) Ca48, particle term. (b) Ca48, hole term. (c) Sn132, particle term. (d) Sn132, hole term.

5.2.4.3 $E^{(3)}$ Experiments

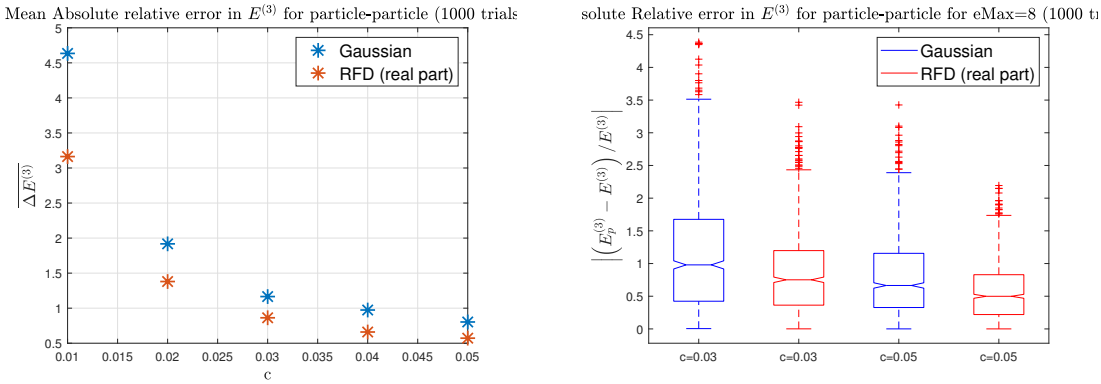
The reason behind $E^{(3)}$ experiments not working as well as $E^{(2)}$ is that the two tensors forming the inner product become nearly orthogonal in the $E^{(3)}$ case, in such a way that after projection, their inner product becomes much smaller than their individual norms. In other words, two tensors that are not originally orthogonal are made close to orthogonal after projection onto the lower-

dimensional subspace. One can think of a criterion that should be met for the inner product preservation to work, i.e., $|\langle \mathbf{Ax}, \mathbf{Ay} \rangle| \geq \bar{\varepsilon} \, \|\mathbf{Ax}\| \, \|\mathbf{Ay}\|$ for some small $\bar{\varepsilon}$, where $\mathbf{x}$ and $\mathbf{y}$ are fibers in unfoldings of a tensor.



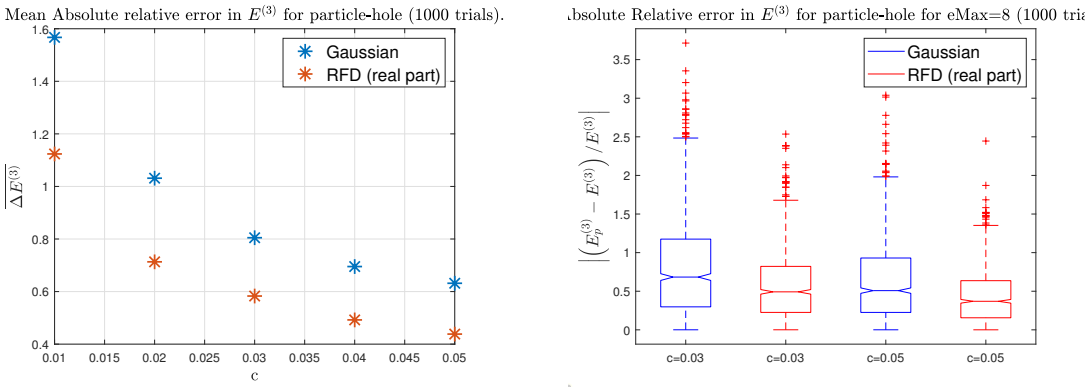
(a) (b)

Figure 5.9: Mean absolute relative error in $E^{(3)}$ for hole-hole and $eMax = 8$.



(a) (b)

Figure 5.10: Mean absolute relative error in $E^{(3)}$ for particle-particle and $eMax = 8$.



(a) (b)

Figure 5.11: Mean absolute relative error in $E^{(3)}$ for particle-hole and $eMax = 8$.

CHAPTER 6

EXTENSION OF VECTOR-BASED METHODS TO TENSORS AND FUTURE WORK

In this chapter, two of the conventional methods developed for vector data are extended to tensors while keeping the miltilinear structure of the data. The computational load and memory requirements of these methods can be mitigated by applying modewise JL embeddings to the inout data. For instance, in Algorithm 6.1 of Section 6.1, the training tensors $\mathcal{X}^{(m)}$ can be compressed to a much smaller size by applying modewise JL prior to solving for the projection matrices $\tilde{\mathbf{U}}^{(j)}$. The same set of JL embeddings used to compress all the training samples could also be used for test data, meaning they need to be generated only once. The result will be an approximate yet easier to obtain set of projection matrices that will be used to compute the approximate MPCA output. In another possible way, randomized methods can be used to obtain an approximate low-rank representation of data during the SVD stage of MPCA (in step 2 of Algorithm 6.1 in the following, SVD could be used on $\sum_{m=1}^{M} \tilde{\mathbf{X}}_{(j)}^{(m)}$ instead of the eigen-decomposition of $\sum_{m=1}^{M} \tilde{\mathbf{X}}_{(j)}^{(m)} \tilde{\mathbf{X}}_{(j)}^{(m)\top}$).

In a similar manner, the support vector machine methods mentioned in Section 6.3 involve the computation of the CP decomposition of a tensor. Obtaining CPD requires solving a least squares problem for each mode of the input data, which in turn can be made faster and memory-efficient by solving the least squares problem using a sketched (compressed) version of the known variables in, e.g., (3.11). This leads to an approximate and a computationally more efficient implementation of CPD to be used in the main support vector machine approach.

6.1 Multilinear Principal Component Analysis

Multilineal Principal Component Analysis, abbreviated to MPCA, is a dimensionality reduction scheme for tensor objects. As an extension to regular PCA, it is similar in structure to the Tucker decomposition and defines its goal to capture as much variations as possible across the modes of a tensor object.

Definition 6.1.1 *Let $\{\mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(M)}\}$ be a set of tensor objects in $\mathbb{R}^{n_1 \times \cdots \times n_d}$. The total scatter of these tensors is defined as*

$$\Psi_{\mathcal{X}} = \sum_{m=1}^{M} \|\mathcal{X}^{(m)} - \bar{\mathcal{X}}\|^2,$$

where $\bar{\mathcal{X}}$ is the mean tensor calculated by

$$\bar{\mathcal{X}} = \frac{1}{M} \sum_{m=1}^{M} \mathcal{X}^{(m)}.$$

6.1.1 Problem Statement

Consider $\{\mathcal{X}^{(m)}\}_{m=1}^{M}$ for training. The objective is to define a multilinear transformation $\{\tilde{\mathbf{U}}^{(j)} \in \mathbb{R}^{n_j \times P_j}\}_{j=1}^{d}$ that maps the tensor space $\mathbb{R}^{n_1 \times \cdots \times n_d}$ into a tensor subspace $\mathbb{R}^{P_1 \times \cdots \times P_d}$ with $P_j \leq n_j$ for $j \in [d]$, such that

$$\mathcal{Y}^{(m)} = \mathcal{X}^{(m)} \times_1 \tilde{\mathbf{U}}^{(1)\top} \times \cdots \times_d \tilde{\mathbf{U}}^{(T)\top} \in \mathbb{R}^{P_1 \times \cdots \times P_d}; \quad m \in [M],$$

capture most of the variations in $\{\mathcal{X}^{(m)}\}_{m=1}^{M}$ measured by the total scatter $\Psi_{\mathcal{Y}}$, i.e.,

$$\{\tilde{\mathbf{U}}^{(j)}\}_{j=1}^{d} = \arg \max_{\tilde{\mathbf{U}}^{(1)}, \ldots, \tilde{\mathbf{U}}^{(d)}} \Psi_{\mathcal{Y}}. \tag{6.1}$$

A pseudo-code outlining the steps of MPCA is shown in Algorithm 6.1 [14], with K denoting the maximal number of allowed iterations.

Theorem 6.1.1 *Let $\{\tilde{\mathbf{U}}^{(j)}\}_{j=1}^{d}$ be the solution to* (6.1)*. Then, given $\tilde{\mathbf{U}}^{(1)}, \ldots, \tilde{\mathbf{U}}^{(j-1)}, \tilde{\mathbf{U}}^{(j+1)}, \ldots, \tilde{\mathbf{U}}^{(d)}$, the matrix $\tilde{\mathbf{U}}^{(j)}$ consists of P_j eigenvectors corresponding to the P_j largest eigenvalues of the matrix*

$$\mathbf{\Phi}^{(j)} = \sum_{m=1}^{M} \left(\mathbf{X}_{(j)}^{(m)} - \bar{\mathbf{X}}_{(j)}\right) \tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}} \tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}}^{\top} \left(\mathbf{X}_{(j)}^{(m)} - \bar{\mathbf{X}}_{(j)}\right)^{\top}, \tag{6.2}$$

where

$$\tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}} = \left(\tilde{\mathbf{U}}^{(d)} \otimes \cdots \otimes \tilde{\mathbf{U}}^{(j+1)} \otimes \tilde{\mathbf{U}}^{(j-1)} \otimes \cdots \otimes \tilde{\mathbf{U}}^{(1)}\right)^{\top}$$

for $j \in [d]$.

Algorithm 6.1: MPCA [14]

Require: Training samples $\mathcal{X}^{(m)} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ for $m \in [M]$.
Ensure: Projection matrices $\tilde{\mathbf{U}}^{(j)} \in \mathbb{R}^{P_j \times n_j}$ for $n \in [d]$.
(1) Center input samples: $\tilde{\mathcal{X}}_m = \mathcal{X}_m - \bar{\mathcal{X}}$.
(2) Initialization: Calculate the eigen-decomposition of $\mathbf{\Phi}^{(j)*} = \sum_{m=1}^{M} \tilde{\mathbf{X}}_{(j)}^{(m)} \tilde{\mathbf{X}}_{(j)}^{(m)\top}$. Set $\tilde{\mathbf{U}}^{(j)}$ to consist of the P_j leading eigenvectors for $j \in [d]$.
(3) Local optimization:
Calculate $\tilde{\mathcal{Y}}^{(m)} = \tilde{\mathcal{X}}^{(m)} \times_1 \tilde{\mathbf{U}}^{(1)\top} \times \cdots \times_d \tilde{\mathbf{U}}^{(T)\top}$ for $m \in [M]$.
Calculate $\Psi_{\mathcal{Y}_0} = \sum_{m=1}^{M} \|\tilde{\mathcal{Y}}^{(m)}\|_F^2$.
for $k = 1, \ldots, K$ **do**
  for $j = 1, \ldots, d$ **do**
    Set the matrix $\tilde{\mathbf{U}}^{(j)}$ to consist of the P_j leading eigenvectors of $\mathbf{\Phi}^{(j)}$ defined in (6.2).
  end for
  Calculate $\tilde{\mathcal{Y}}_m$ for $m \in [M]$, and $\Psi_{\mathcal{Y}_k}$.
  If $\Psi_{\mathcal{Y}_k} - \Psi_{\mathcal{Y}_{k-1}} < \eta$, break, and go to step 4.
end for
(4) Projection: Obtain the feature tensors as $\mathcal{Y}^{(m)} = \mathcal{X}^{(m)} \times_1 \tilde{\mathbf{U}}^{(1)\top} \times \cdots \times_d \tilde{\mathbf{U}}^{(T)\top}$ for $m \in [M]$.

Proof *The Euclidean norm of a tensor is equal to the Frobenius norm of any of its unfoldings. Therefore, the total scatter of the projected samples can be written as*

$$
\begin{aligned}
\Psi_{\mathcal{Y}} &= \sum_{m=1}^{M} \|\mathcal{Y}^{(m)} - \bar{\mathcal{Y}}\|^2 = \sum_{m=1}^{M} \|\mathbf{Y}_{(j)}^{(m)} - \bar{\mathbf{Y}}_{(j)}\|_F^2 \\
&= \sum_{m=1}^{M} \|\tilde{\mathbf{U}}^{(j)} \left(\mathbf{X}_{(j)}^{(m)} - \bar{\mathbf{X}}_{(j)} \right) \tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}} \|_F^2 \\
&= \sum_{m=1}^{M} trace \left(\tilde{\mathbf{U}}^{(j)} \left(\mathbf{X}_{(j)}^{(m)} - \bar{\mathbf{X}}_{(j)} \right) \tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}} \tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}}^{\top} \left(\mathbf{X}_{(j)}^{(m)} - \bar{\mathbf{X}}_{(j)} \right)^{\top} \tilde{\mathbf{U}}^{(j)\top} \right) \\
&= trace \left(\tilde{\mathbf{U}}^{(j)} \sum_{m=1}^{M} \left(\mathbf{X}_{(j)}^{(m)} - \bar{\mathbf{X}}_{(j)} \right) \tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}} \tilde{\mathbf{U}}_{\mathbf{\Phi}^{(j)}}^{\top} \left(\mathbf{X}_{(j)}^{(m)} - \bar{\mathbf{X}}_{(j)} \right)^{\top} \tilde{\mathbf{U}}^{(j)\top} \right) \\
&= trace \left(\tilde{\mathbf{U}}^{(j)} \mathbf{\Phi}^{(j)} \tilde{\mathbf{U}}^{(j)\top} \right),
\end{aligned}
$$

which turns into an eigenvalue problem when $\Psi_{\mathcal{Y}}$ is to be maximized.

6.1.2 Full Projection

If $P_j = n_j$ for $j \in [d]$, it is easy to show that $\tilde{\mathbf{U}}^{(j)}_{\mathbf{\Phi}^{(j)}} \tilde{\mathbf{U}}^{(j)\top}_{\mathbf{\Phi}^{(j)}} = \mathbf{I}$, then in the optimal case, $\mathbf{\Phi}^{(j)*} = \sum_{m=1}^{M} \left(\mathbf{X}^{(m)}_{(j)} - \bar{\mathbf{X}}^{(m)}_{(j)}\right)\left(\mathbf{X}^{(m)}_{(j)} - \bar{\mathbf{X}}^{(m)}_{(j)}\right)^{\top}$. In this case, $\mathbf{U}^{(j)*}$ is the optimal solution for $\tilde{\mathbf{U}}^{(j)}$, and consists of the eigenvectors of $\mathbf{\Phi}^{(j)*}$.

The total scatter tensor $\mathcal{Y}^{*}_{\text{var}}$ of the full projection is defined as

$$\mathcal{Y}^{*}_{\text{var}} = \sum_{m=1}^{M} \left(\mathcal{Y}^{(m)*} - \bar{\mathcal{Y}}^{*}\right)^{2}, \tag{6.3}$$

where the exponentiation is done component-wise, $\mathcal{Y}^{(m)*}$ is the full projection of the m^{th} sample $\mathcal{X}^{(m)}$ and $\bar{\mathcal{Y}}^{*}$ is the mean of the fully projected samples. The following two observations can be made.

(a) The i_j^{th} mode-j eigenvalue $\lambda^{(j)*}_{i_j}$ is the sum of all entries of the i_j^{th} mode-j slice of $\mathcal{Y}^{*}_{\text{var}}$, where $i_j \in [n_j]$ for $j \in [d]$.

(b) Every sample tensor $\mathcal{X}^{(m)}$ can be represented as an expansion in the subspace spanned by rank-1 tensors, called eigentensors. This is shown by

$$\mathcal{X}^{(m)} \approx \sum_{i_1=1}^{P_1} \sum_{i_2=1}^{P_2} \cdots \sum_{i_d=1}^{P_d} \mathcal{Y}^{(m)*}_{i_1,i_2,\ldots,i_d} \tilde{\mathbf{u}}^{(1)}_{i_1} \bigcirc \tilde{\mathbf{u}}^{(2)}_{i_2} \bigcirc \cdots \bigcirc \tilde{\mathbf{u}}^{(d)}_{i_d}, \tag{6.4}$$

where $\tilde{\mathbf{u}}^{(j)}_{i_j}$ is the i_j^{th} column of $\tilde{\mathbf{U}}^{(j)}_{i_j}$ for $i_j \in [P_j]$ and $j \in [d]$.

6.1.3 Initialization by Full Projection Truncation (FPT)

To initialize the MPCA algorithm, assume the first $P_j < n_j$ leading eigenvectors of $\mathbf{\Phi}^{(n)*}$ form $\tilde{\mathbf{U}}^{(j)}$. It is shown in [14] that if a nonzero eigenvalue is truncated in one mode, the eigenvalues in all other modes tend to decrease in magnitude, and therefore, the optimality of (6.1) is affected negatively. Thus, the eigen-decomposition needs to be updated in all other modes. If the total scatter of the projected samples in FPT is denoted by $\Psi_{\mathcal{Y}_0}$, then the loss of variations due to FPT is

bounded by

$$\max_j \sum_{i_j=P_j+1}^{n_j} \lambda_{i_j}^{(j)*} \leq \Psi_{\mathcal{X}} - \Psi_{\mathcal{Y}_0} \leq \sum_{j=1}^{d} \sum_{i_j=P_j+1}^{n_j} \lambda_{i_j}^{(j)*}. \tag{6.5}$$

6.1.4 Determination of subspace Dimensions P_j

A simple yet commonly used method to choose P_j is to pick the minimum value P_j such that

$$\frac{\sum_{i_j=1}^{P_j} \lambda_{i_j}^{(j)*}}{\sum_{i_j=1}^{n_j} \lambda_{i_j}^{(j)*}} \geq \tau, \tag{6.6}$$

where τ is a predetermined threshold set by the user.

6.1.5 Feature Extraction and Classification

The set of projection matrices $\{\tilde{\mathbf{U}}^{(j)}\}_{j=1}^{d}$ obtained using the training samples can be employed to project any new sample onto the same subspace. The projected samples (training or test) can be either directly used in classification, or they can undergo further processing. In LDA[1], the elements of the projected samples $\mathcal{Y}^{(m)}$ are vectorized to yield $\mathbf{y}^{(m)}$ and are ordered according to the Fisher score to maximize the between-class to in-class discriminability. Next, a predetermined number of features with the highest Fisher scores are selected for classification.

Or, to maximize between-class to in-class discriminability, $\mathbf{y}^{(m)}$ can be projected onto the LDA space by

$$\mathbf{z}^{(m)} = \mathbf{V}_{\text{lda}}^{\top} \mathbf{y}^{(m)},$$

where for C classes, $\mathbf{V}_{\text{lda}}$ consists of $n_z \leq C-1$ of the leading generalized eigenvectors of $\mathbf{S}_W = \sum_{m=1}^{M} \left(\mathbf{y}^{(m)} - \bar{\mathbf{y}}_c^{(m)}\right) \left(\mathbf{y}^{(m)} - \bar{\mathbf{y}}_c^{(m)}\right)^{\top}$ and $\mathbf{S}_B = \sum_{c=1}^{C} N_c \left(\bar{\mathbf{y}}_c - \bar{\mathbf{y}}\right) \left(\bar{\mathbf{y}}_c - \bar{\mathbf{y}}\right)^{\top}$. Here, N_c is the number of samples in class c and $\bar{\mathbf{y}}_c^{(m)}$ denotes the in-class mean for the m^{th} training sample, i.e.,

[1] Linear Discriminant Analysis

$\bar{\mathbf{y}}_c^{(m)} \in \{\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_C\}$ where $\bar{\mathbf{y}}_c$ is the mean of class c. In fact, the LDA projection matrix is obtained by solving

$$\mathbf{V}_{\text{lda}} = \arg\max_{\mathbf{V}} \frac{|\mathbf{V}^\top \mathbf{S}_B \mathbf{V}|}{|\mathbf{V}^\top \mathbf{S}_W \mathbf{V}|} = \left[\mathbf{v}_1 \ldots \mathbf{v}_{n_z}\right].$$

Now, $\mathbf{z}^{(m)}$ is used for classification as the projected training sample, and $\mathbf{V}_{\text{lda}}$ is applied to any vectorized projected test data $\mathbf{y}$ to further project it onto the LDA space according to $\mathbf{z} = \mathbf{V}_{\text{lda}}^\top \mathbf{y}$.

6.2 Comparison between PCA, MPCA and MPS

Here, the methods PCA, MPCA and MPS have been compared in terms of computational complexity measured by training time, and classification Success Rate (CSR). The data were first projected onto the subspace, and then the features were either directly used in nearest neighbor (1NN) classification, or they were further sorted in descending order according to the Fisher score and 100 features were selected to be used in 1NN. The following data sets were used in the experiments.

COIL-100 *data set*: 7200 images collected from 100 objects taken at 5° pose intervals, creating a 3-mode tensor $\mathcal{X} \in \mathbb{R}^{128\times128\times7200}$ [15]. Sample images from this library can be viewed in Figure 6.1.

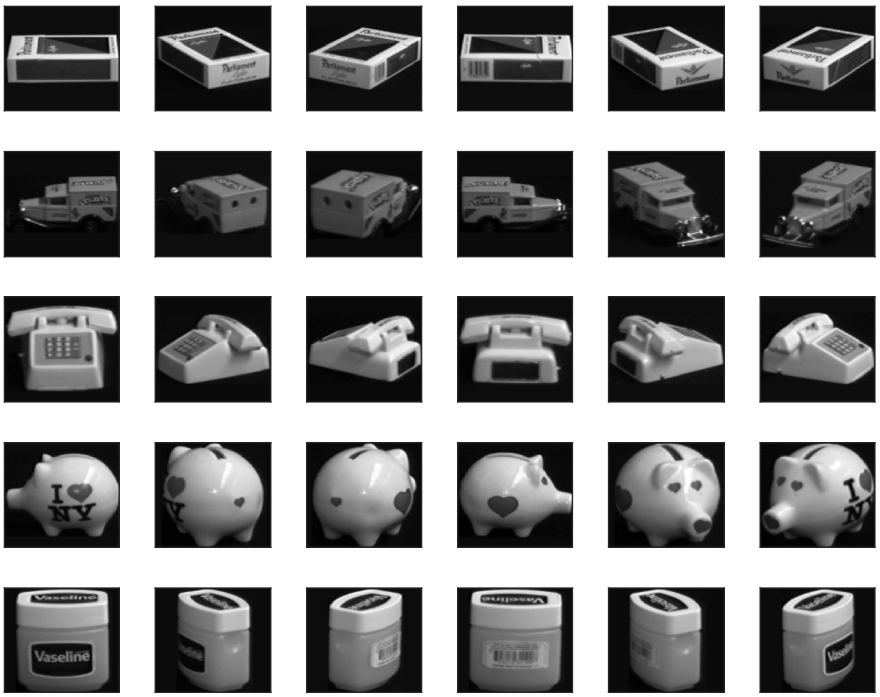


Figure 6.1: Gray-scale sample images of five objects in the COIL-100 database.

MRI data: A 4-mode tensor $\mathcal{X} \in \mathbb{R}^{240\times240\times155\times51}$ comprised of 51 MRI samples [1]. A lateral slice of a sample MRI image is shown in Figure 6.2.

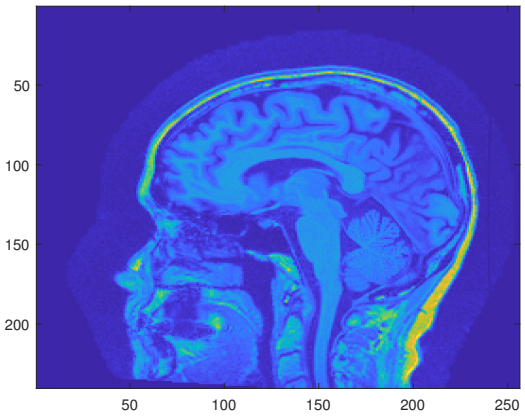
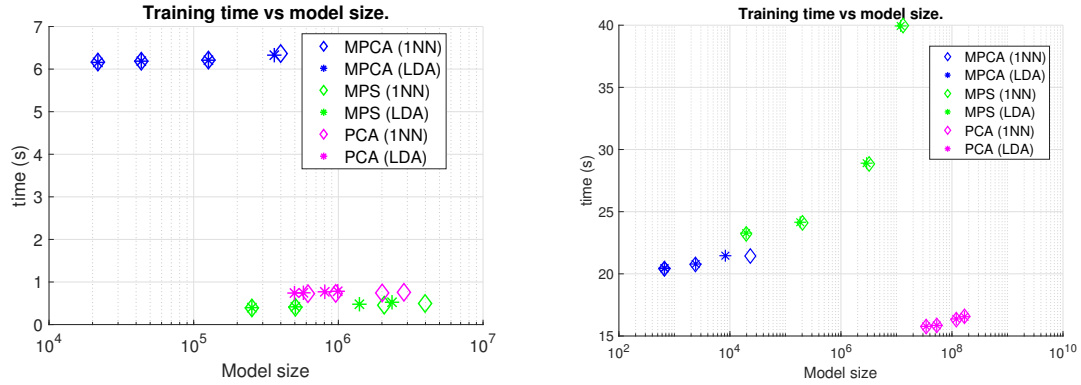
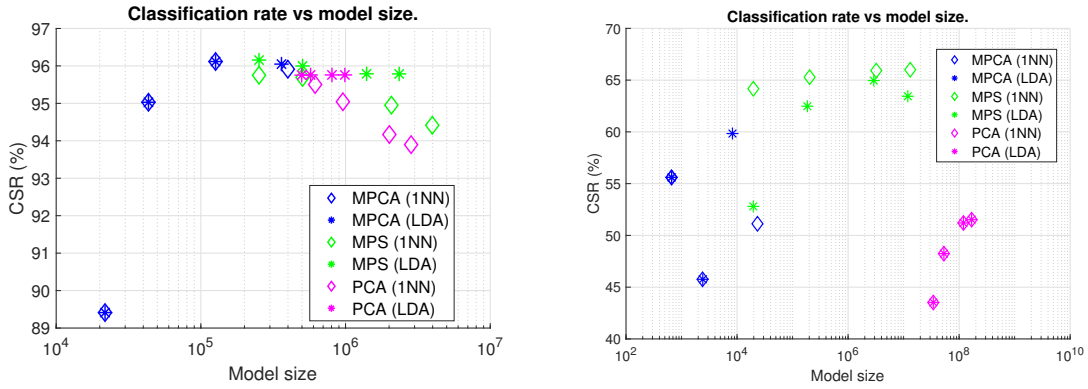


Figure 6.2: A lateral slice of a sample MRI image.



(a) COIL-100 (b) MRI

Figure 6.3: Training time. (a) COIL-100. (b) MRI.



(a) (b)

Figure 6.4: Classification Success Rate. (a) COIL-100. (b) MRI.

6.3 Extension of Support Vector Machine to Tensors

In this section, 3 tensor-based methods used to extend the regular support vector machine to tensor data are summarized. Consider M d-mode training sample tensors $\{\mathcal{X}^{(m)}\}_{m=1}^{M} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ with corresponding labels $\{y_m\}_{m=1}^{M} \in \{-1, +1\}$.

6.3.1 Support Tensor Machine

The soft-margin Support Tensor Machine for binary classification of data is composed of d quadratic programming problems with inequality constraints, where the j^{th} problem is described as [24]:

$$\begin{aligned}\min_{\mathbf{w}^{(j)},b^{(j)},\boldsymbol{\xi}^{(j)}} J\left(\mathbf{w}^{(j)},b^{(j)},\boldsymbol{\xi}^{(j)}\right) &= \frac{1}{2}\|\mathbf{w}^{(j)}\|_2^2 \prod_{\substack{i=1\\ i\neq j}}^{d} \|\mathbf{w}^{(i)}\|_2^2 + C\sum_{m=1}^{M}\xi_m^{(j)} \\ \text{subject to } \; y_m &\left(\mathbf{w}^{(j)\top}\left(\mathcal{X}^{(m)}\bigtimes_{\substack{i=1\\ i\neq j}}^{d}\mathbf{w}^{(j)\top}\right)+b^{(j)}\right)\geq 1-\xi_m^{(j)} \\ \xi_m^{(j)} &\geq 0, m\in[M],\end{aligned} \tag{6.7}$$

for $j \in [d]$, where $\mathbf{w}^{(j)} \in \mathbb{R}^{n_j}$ is the normal to the j^{th} hyperplane corresponding to the j^{th} mode, $b^{(j)}$ is the bias, $\xi_m^{(j)}$ is error of the m^{th} training sample, and C is the trade-off between the classification error and the amount of margin violation. These d optimization problems have no closed-form solution, and need to be solved iteratively using the alternating projection algorithm. All d normal vectors are randomly initialized. In each iteration, for each mode j, $\{\mathbf{w}^{(k)}\}_{k\neq j}$ are fixed and (6.7) is solved for $\mathbf{w}^{(j)}$. Iterations continue until convergence is reached. Convergence criterion considering iterations t and $t-1$ is set as

$$\sum_{j=1}^{d}\left(\left|\frac{\mathbf{w}_t^{(j)\top}\mathbf{w}_{t-1}^{(j)}}{\|\mathbf{w}_t^{(j)}\|^2}\right|-1\right)\leq\varepsilon,$$

for some ε. Once the STM model has been solved, the binary classifier will determine the class of a test sample $\mathcal{X}$ based on the decision rule

$$y(\mathcal{X}) = \text{sign}\left(\mathcal{X}\bigtimes_{i=1}^{d}\mathbf{w}^{(j)\top}+b\right). \tag{6.8}$$

Further, assume that $\xi_m = \max_{j\in[d]}\{\xi_m^{(j)}\}$, and that the d normal vectors $\mathbf{w}^{(j)}$ form a rank-1 tensor $\mathcal{W} = \mathbf{w}^{(1)} \bigcirc \mathbf{w}^{(2)} \bigcirc \cdots \bigcirc \mathbf{w}^{(d)}$ [6]. In this case, the following can be observed.

$$\|\mathcal{W}\|^2 = \langle\mathcal{W},\mathcal{W}\rangle = \sum_{i_1,\ldots,i_d}\mathcal{W}_{i_1,\ldots,i_d}^2 = \sum_{i_1,\ldots,i_d}\mathbf{w}_{i_1}^{(1)2}\ldots\mathbf{w}_{i_d}^{(d)2} = \prod_{j=1}^{d}\|\mathbf{w}^{(j)}\|_2^2, \tag{6.9}$$

and

$$\mathbf{w}^{(j)\top}\left(\mathcal{X}^{(m)}\bigtimes_{\substack{i=1\\ i\neq j}}^{d}\mathbf{w}^{(j)\top}\right)=\mathcal{X}^{(m)}\bigtimes_{i=1}^{d}\mathbf{w}^{(j)\top}=\sum_{i_1,\ldots,i_d}\mathcal{X}^{(m)}_{i_1,\ldots,i_d}\mathbf{w}^{(1)}_{i_1}\ldots\mathbf{w}^{(d)}_{i_d}=\left\langle\mathcal{X}^{(m)},\mathcal{W}\right\rangle, \quad (6.10)$$

where (2.11) and (2.12) have been used in the penultimate equality. This result can be used to write the problem as

$$\begin{aligned}\min_{\mathcal{W},b,\boldsymbol{\xi}} J\left(\mathcal{W},b,\boldsymbol{\xi}\right)&=\frac{1}{2}\|\mathcal{W}\|^2+C\sum_{m=1}^{M}\xi_m\\ \text{subject to }\ & y_m\left(\left\langle\mathcal{X}^{(m)},\mathcal{W}\right\rangle+b\right)\geq 1-\xi_m\\ & \xi_m\geq 0,\end{aligned} \quad (6.11)$$

By forming the Lagrangian function with Lagrange multipliers α and λ, and taking partial derivatives with respect to $\mathcal{W}$, b and ξ_m, we obtain $\mathcal{W}=\sum_{m=1}^{M}\alpha_m y_m\mathcal{X}^{(m)}$, $\sum_{m=1}^{M}\alpha_m y_m=0$ and $\alpha_m+\lambda_m=C$. Then, the dual problem can be written in the following form.

$$\begin{aligned}\max_{\boldsymbol{\alpha}} &= \mathbf{1}^\top\boldsymbol{\alpha}-\frac{1}{2}\boldsymbol{\alpha}^\top\mathbf{S}\boldsymbol{\alpha}\\ \text{subject to }\ & \sum_{m=1}^{M}\alpha_m y_m=0\\ & 0\leq\alpha_m\leq C,\ \ m\in[M],\end{aligned} \quad (6.12)$$

where $\mathbf{S}_{pq}=y_p y_q\left\langle\mathcal{X}^{(p)},\mathcal{X}^{(q)}\right\rangle$. Therefore, after solving the model, the binary classifier will be

$$y(\mathcal{X})=\operatorname{sign}\left(\langle\mathcal{X},\mathcal{W}\rangle+b\right), \quad (6.13)$$

for a test tensor $\mathcal{X}$. This procedure is equivalent to solving the regular soft-margin SVM if the tensors are first vectorized. However, for large tensors vectorized, SVM suffers significantly from the curse of dimensionality and also small training sample count compared to the dimensionality of each sample making it susceptible to new data.

6.3.2 Support Higher-order Tensor Machine

Support Higher-order Tensor Machine (abbreviated to SHTM) assumes that the data samples admit low-rank CP decompositions, which allows for another form of the objective function in the dual problem. As can be expected, in obtaining the CPD of data, a sketched least squares problem can be solved in each mode of the training and test samples to obtain approximate but more efficient versions of the corresponding CP decompositions. Let the CPD of $\mathcal{X}^{(p)}$ and $\mathcal{X}^{(q)}$ be $\mathcal{X}^{(p)} \approx \sum_{k=1}^{r} \mathbf{x}_{pk}^{(1)} \bigcirc \cdots \bigcirc \mathbf{x}_{pk}^{(d)}$ and $\mathcal{X}^{(q)} \approx \sum_{k=1}^{r} \mathbf{x}_{qk}^{(1)} \bigcirc \cdots \bigcirc \mathbf{x}_{qk}^{(d)}$, where $\mathbf{x}_{qk}^{(j)}$ and $\mathbf{x}_{pk}^{(j)}$ represent the k^{th} column in the j^{th} factor matrix of $\mathcal{X}^{(p)}$ and $\mathcal{X}^{(q)}$, respectively. Therefore, the elements of $\mathbf{S}$ in (6.12) will be

$$\begin{aligned}
\mathbf{S}_{pq} &= y_p y_q \left\langle \mathcal{X}^{(p)}, \mathcal{X}^{(q)} \right\rangle = y_p y_q \sum_{k,h=1}^{r} \left\langle \mathbf{x}_{pk}^{(1)} \bigcirc \cdots \bigcirc \mathbf{x}_{pk}^{(d)}, \mathbf{x}_{qh}^{(1)} \bigcirc \cdots \bigcirc \mathbf{x}_{qh}^{(d)} \right\rangle \\
&= y_p y_q \sum_{k,h=1}^{r} \sum_{i_1,\ldots,i_d} \left(\mathbf{x}_{pk}^{(1)} \bigcirc \cdots \bigcirc \mathbf{x}_{pk}^{(d)} \right)_{i_1,\ldots,i_d} \overline{\left(\mathbf{x}_{qh}^{(1)} \bigcirc \cdots \bigcirc \mathbf{x}_{qh}^{(d)} \right)_{i_1,\ldots,i_d}} \\
&= y_p y_q \sum_{k,h=1}^{r} \prod_{j=1}^{d} \left\langle \mathbf{x}_{pk}^{(j)}, \mathbf{x}_{qh}^{(j)} \right\rangle .
\end{aligned}$$

Now, (6.12) can be solved using Sequential Minimal Optimization [11] in one iteration. The binary classifier for a test sample $\mathcal{X} \approx \sum_{h=1}^{r} \mathbf{x}_{h}^{(1)} \bigcirc \cdots \bigcirc \mathbf{x}_{h}^{(d)}$ will decide the class based on

$$y(\mathcal{X}) = \operatorname{sign}\left(\langle \mathcal{X}, \mathcal{W} \rangle + b \right) = \operatorname{sign}\left(\sum_{m=1}^{M} \sum_{k=1}^{r} \sum_{h=1}^{r} \alpha_m y_m \prod_{j=1}^{d} \left\langle \mathbf{x}_{mk}^{(j)}, \mathbf{x}_{h}^{(j)} \right\rangle + b \right), \tag{6.14}$$

which clearly shows that the curse of dimensionality will not be an issue as only the much smaller factors of the CPD of data are incorporated in the classification operation.

6.3.3 Kernelized Support Tensor Machine

Aside from the low-rank structure that is assumed for the weight tensor $\mathcal{W}$ and the data samples, STM and SHTM are only taking into account the multilinear structure in tensors, and are missing any nonlinearities that might exist in the data. For this reason, STM and SHTM will yield

suboptimal results. A kernelized approach can lead to improved performance by capturing the existing nonlinearities in the data. In [7], the primal problem for Kernelized Support Tensor Machine (KSTM) is stated as

$$\arg\min_{\mathcal{W},b} L\left(y, \langle \mathcal{W}, \mathcal{X} \rangle + b\right) + P\left(\mathcal{W}\right) + \Omega\left(\mathcal{X}\right), \tag{6.15}$$

where L is a loss function, $P\left(\mathcal{W}\right)$ is a penalty function, and $\Omega\left(\mathcal{X}\right)$ is a specific constraint imposed on the training samples. Before going into details about the terms in (6.15), the tensor Reproducing Kernel Hilbert Space needs to be defined.

Note: For a domain $\mathcal{D}$, a reproducing kernel $\kappa : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ is a kernel function associated with a feature map $\phi : \mathcal{D} \to \mathbb{R}^{\mathcal{D}}$, where $\mathbb{R}^{\mathcal{D}}$ denotes the space of functions from $\mathcal{D}$ to $\mathbb{R}$, with the property that any function $f \in \mathbb{R}^{\mathcal{D}}$ can be reproduced pointwise by calculating the inner product of the kernel and f, i.e., $f(x) = \langle f(\cdot), \kappa(\cdot, x) \rangle$ for all $x \in \mathcal{D}$, where $\kappa(\cdot, x) = \phi(x)$.

A direct consequence of this property is that any inner product defined on $\mathbb{R}^{\mathcal{D}}$ can be calculated by the kernel as $\langle \phi(x_1), \phi(x_2) \rangle = \langle \kappa(\cdot, x_1), \kappa(\cdot, x_2) \rangle = \kappa(x_2, x_1) = \kappa(x_1, x_2)$ for all $x_1, x_2 \in \mathcal{D}$.

Definition 6.3.1 *Tensor Product Reproducing Kernel Hilbert Space*
For $j \in [d]$, let $\left(\mathcal{H}_j, \langle ., . \rangle_j, \kappa^{(j)}\right)$ be a reproducing kernel Hilbert space (RKHS) of functions on a set $\mathcal{S}_j$ with a reproducing kernel $\kappa^{(j)} : \mathcal{S}_j \times \mathcal{S}_j \to \mathbb{R}$ and the inner product operator $\langle \cdot, \cdot \rangle_j$. The space $\mathcal{H} = \mathcal{H}_1 \bigcirc \cdots \bigcirc \mathcal{H}_d$ is called a tensor product RKHS of functions on the domain $\mathcal{S} := \mathcal{S}_1 \times \cdots \times \mathcal{S}_d$. In particular, assume that $x = \left(x^{(1)}, \ldots, x^{(d)}\right) \in \mathcal{S}$ is a tuple. Let the tensor product space formed by the linear combinations of the functions $f^{(j)}$ for $j \in [d]$ be defined as

$$f^{(1)} \bigcirc \cdots \bigcirc f^{(d)} : x \mapsto \prod_{j=1}^{d} f^{(j)}\left(x^{(j)}\right), \quad f^{(j)} \in \mathcal{H}_j$$

Then for a multi-index $\boldsymbol{k} = (k_1, \ldots, k_d)$, it holds that

$$\sum_{\boldsymbol{k}} \mathcal{G}_{\boldsymbol{k}} \left(f_{k_1}^{(1)} \bigcirc \cdots \bigcirc f_{k_d}^{(d)}\right)(x) = \sum_{\boldsymbol{k}} \mathcal{G}_{\boldsymbol{k}} \prod_{j=1}^{d} f_{k_j}^{(j)}\left(x^{(j)}\right) = \sum_{\boldsymbol{k}} \mathcal{G}_{\boldsymbol{k}} \prod_{j=1}^{d} \left\langle f_{k_j}^{(j)}, k_x^{(j)} \right\rangle_j, \tag{6.16}$$

where $\mathcal{G}_{\boldsymbol{k}}$ is the combination coefficient, and $k_x^{(j)}$ is the function $k^{(j)}\left(\cdot, x^{(j)}\right) : t \mapsto k^{(j)}\left(t, x^{(j)}\right)$ in the sense that $f_{k_j}^{(j)}\left(x^{(j)}\right) = \left\langle f_{k_j}^{(j)}(\cdot), k_x^{(j)}\left(\cdot, x^{(j)}\right)\right\rangle_j$.

By looking closely at (6.16), it is observed that if in a special case, we let $f_{k_j}^{(j)}\left(x^{(j)}\right) = \mathbf{u}_{k_j}^{(j)}\left(i_j\right)$ for $k_j \in [r_j]$ and $i_j \in [n_j]$ for $j \in [d]$, then

$$\left(f_{k_1}^{(1)} \bigcirc \cdots \bigcirc f_{k_d}^{(d)}\right)\left(x^{(1)}, \ldots, x^{(d)}\right) = \left(\mathbf{u}_{k_1}^{(1)} \bigcirc \cdots \bigcirc \mathbf{u}_{k_d}^{(d)}\right)(i_1, \ldots, i_d) = \mathbf{u}_{k_1}^{(1)}(i_1) \ldots \mathbf{u}_{k_d}^{(d)}(i_d)$$

and (6.16) presents a general form of the Tucker decomposition, in a kernelized form, where $\mathbf{u}_{k_j}^{(j)}\left(i_j\right)$ can be represented as the inner product of a kernel and $\mathbf{u}_{k_j}^{(j)}$, and $\mathcal{G}$ plays the role of the core tensor. Therefore (6.16) represents a kernelized tensor factorization. Treating $\mathcal{X} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ as an element of the tensor product RKHS $\mathcal{H}$, assume that it has a low-rank structure in $\mathcal{H}$, such that

$$\arg \min_{\mathcal{G}, \{\mathbf{U}^{(j)}\}_{j=1}^d} \|\mathcal{X} - \sum_{k_1, \ldots, k_d} \mathcal{G}_{k_1, \ldots, k_d}\left(\bigcirc_{j=1}^d \mathbf{u}_{k_j}^{(j)}\right)\|^2 = \arg \min_{\mathcal{G}, \{\mathbf{U}^{(j)}\}_{j=1}^d} \|\mathcal{X} - \mathcal{G} \bigtimes_{j=1}^d \left(\mathbf{K}^{(j)}\mathbf{U}^{(j)}\right)\|^2, \tag{6.17}$$

where $\mathbf{K}^{(j)} \in \mathbb{R}^{n_j \times n_j}$ is a symmetric kernel matrix whose i_j^{th} row/column is $k_x^{(j)}\left(\cdot, i_j\right)$ for $i_j \in [n_j]$, and $\mathbf{U}^{(j)} \in \mathbb{R}^{n_j \times r_j}$ has $\mathbf{u}_{k_j}^{(j)}$ as its k_j^{th} column for $k_j \in [r_j]$. To get from the left-hand side to the right-hand side, one can use the reproducing property of the kernel, i.e.,

$$\left(\bigcirc_{j=1}^d \mathbf{u}_{k_j}^{(j)}\right)_{i_1, \ldots, i_d} = \prod_{j=1}^d \mathbf{u}_{k_j}^{(j)}\left(i_j\right) = \prod_{j=1}^d \left\langle \mathbf{u}_{k_j}^{(j)}, k_x^{(j)}\left(\cdot, i_j\right)\right\rangle,$$

In a kernelized CP factorization (KCP), the objective function of (6.17) will be the same except that $\mathcal{G}$ will be a diagonal tensor, and if the elements of its superdiagonal are absorbed into the factor matrices, then the primal model of KSTM for M training samples $\{\mathcal{X}^{(m)}, y_m\}_{m=1}^M$ is stated as

$$\begin{aligned} \min_{\{\mathbf{U}_m^{(j)}\}_{j=1}^d, \{\mathbf{K}^{(j)}\}_{j=1}^d, \mathcal{W}, b} \quad & \gamma \sum_{m=1}^M \|\mathcal{X}^{(m)} - \mathcal{I} \bigtimes_{j=1}^d \left(\mathbf{K}^{(j)}\mathbf{U}_m^{(j)}\right)\|^2 \\ & + \langle \mathcal{W}, \mathcal{W} \rangle \\ & + C \sum_{m=1}^M \left[1 - y_m\left(\left\langle \mathcal{W}, \hat{\mathcal{X}}^{(m)}\right\rangle + b\right)\right]_+, \end{aligned} \tag{6.18}$$

where $\mathcal{I}$ denotes the identity tensor and $[1-x]_+ = \max(0, 1-x)^p$ for $p = 1$ or $p = 2$. Also, $\hat{\mathcal{X}}^{(m)}$ is the CP reconstruction of $\mathcal{X}^{(m)}$ using $\{\mathbf{U}_m^{(j)}\}_{j=1}^d$. Comparing (6.18) with (6.15), the first term is $\Omega(\mathcal{X})$ representing the total KCP reconstruction error of training samples penalized by γ, the second term is $P(\mathcal{W})$, and the third term corresponds to $L(y, \langle \mathcal{W}, \mathcal{X} \rangle + b)$. All training samples are sharing the same set of kernel matrices for a specific mode $j \in [d]$ which makes characterization of tensor data possible by taking into account both common and discriminative features. Solving (6.18) in the dual domain is very complicated due to the inherent coupling between the weight tensor $\mathcal{W}$ and factor matrices $\{\mathbf{U}_m^{(j)}\}_{j=1}^d$. The kernel trick is used to implicitly capture the nonlinear structures in the data. If $\mathcal{W}$ is replaced with a function

$$f(\cdot) = \sum_{m=1}^{M} \beta_m \, \kappa(\cdot, \hat{\mathcal{X}}^{(m)}),$$

represented as the linear combination of a kernel function $\kappa(\cdot, \hat{\mathcal{X}}^{(m)})$ for M reconstructed training data samples, then (6.18) can be transformed to

$$\begin{aligned} \min_{\{\mathbf{U}_m^{(j)}\}_{j=1}^d, \{\mathbf{K}^{(j)}\}_{j=1}^d, \boldsymbol{\beta}, b} \; & \gamma \sum_{m=1}^{M} \| \mathcal{X}^{(m)} - \mathcal{I} \bigtimes_{j=1}^{d} \left(\mathbf{K}^{(j)} \mathbf{U}_m^{(j)} \right) \|^2 \\ & + \lambda \sum_{i,j=1}^{M} \beta_i \beta_j \, \kappa(\hat{\mathcal{X}}^{(i)}, \hat{\mathcal{X}}^{(j)}) \\ & + \sum_{m=1}^{M} \left[1 - y_m \left(\sum_{j=1}^{M} \beta_j \kappa(\hat{\mathcal{X}}^{(m)}, \hat{\mathcal{X}}^{(j)}) + b \right) \right]_+, \end{aligned} \tag{6.19}$$

where $\lambda = 1/C$ is the weight between the loss function and the margin, and γ controls the tradeoff between discriminative components and reconstruction error. Letting $\hat{\mathbf{K}}_{i,j} = \kappa(\hat{\mathcal{X}}^{(i)}, \hat{\mathcal{X}}^{(j)})$ denote the elements of the symmetric kernel matrix $\hat{\mathbf{K}}$, the so called *dual form* of (6.18) is obtained as

$$\begin{aligned} \min_{\{\mathbf{U}_m^{(j)}\}_{j=1}^d, \{\mathbf{K}^{(j)}\}_{j=1}^d, \boldsymbol{\beta}, b} \; & \gamma \sum_{m=1}^{M} \| \mathcal{X}^{(m)} - \mathcal{I} \bigtimes_{j=1}^{d} \left(\mathbf{K}^{(j)} \mathbf{U}_m^{(j)} \right) \|^2 \\ & + \lambda \boldsymbol{\beta}^\top \hat{\mathbf{K}} \boldsymbol{\beta} \\ & + \sum_{m=1}^{M} \left[1 - y_m \left(\hat{\mathbf{k}}_m^\top \boldsymbol{\beta} + b \right) \right]_+, \end{aligned} \tag{6.20}$$

where $\hat{\mathbf{k}}_m$ denotes the m^{th} row/column of $\hat{\mathbf{K}}$. This objective function is non-convex and finding a global minimum might not be possible. Therefore, an iterative scheme has been derived in [7] by alternatively finding the local minimum of the objective function for each variable by fixing the others. This is done by updating $\mathbf{K}^{(j)}$, $\boldsymbol{\beta}$, $\mathbf{U}_m^{(j)}$ and b consecutively in each iteration. Let $z_m = \hat{\mathbf{k}}_m^\top \boldsymbol{\beta} + b$ in the following steps whenever it is used. Also, let $[1-x]_+ = \max(0, 1-x)^2$.

At each iteration, solving (6.20) includes the following steps.

(a) Update $\mathbf{K}^{(j)}$: Since there is no supervised information involving $\mathbf{K}^{(j)}$, the optimization technique in CPD is used by finding the solution to the following system of equations for $j \in [d]$.

$$\mathbf{K}^{(j)} \sum_{m=1}^{M} \left(\mathbf{U}_m^{(j)} \mathbf{W}_m^{(-j)} \right) = \sum_{m=1}^{M} \left(\mathbf{X}_{(j)}^{(m)} \mathbf{V}_m^{(-j)} \right)$$

where $\mathbf{X}_{(j)}^{(m)}$ is the mode-j unfolding of $\mathcal{X}^{(m)}$, $\mathbf{V}_m^{(-j)} = \bigodot_{\substack{\ell=d \\ \ell \neq j}}^{1} \left(\mathbf{K}^{(j)} \mathbf{U}_m^{(j)} \right)$ and $\mathbf{W}_m^{(-j)} = \left(\mathbf{V}_m^{(-j)} \right)^\top \mathbf{V}_m^{(-j)}$.

(b) Update $\boldsymbol{\beta}$: A data point is a support vector (support tensor, in fact) if $y_m z_m < 1$ for that point, i.e., if the loss is nonzero for it. After reordering the training samples such that the first M_s samples are support tensors, $\boldsymbol{\beta}$ can be found by setting the first-order gradient of the objective function to zero, i.e.,

$$\nabla_{\boldsymbol{\beta}} = 2 \left(\lambda \hat{\mathbf{K}} \boldsymbol{\beta} + \hat{\mathbf{K}} \mathbf{I}^0 \left(\hat{\mathbf{K}} \boldsymbol{\beta} - \mathbf{y} + b\mathbf{1} \right) \right) = 0,$$

where $\mathbf{I}^0 = \begin{bmatrix} \mathbf{I}_{M_s} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$ and $\mathbf{y}$ is the vector of labels.

(c) Update $\mathbf{U}_m^{(j)}$: The kernel function κ is inherently coupled with $\mathbf{U}_m^{(j)}$, which underlines the importance of choosing an appropriate kernel when calculating the gradient of the objective function with respect to $\mathbf{U}_m^{(j)}$. Before that, it should be made clear how the kernel κ is related to $\mathbf{U}_m^{(j)}$. Mapping the tensors into a tensor Hilbert space of higher dimension, one can retain

the multilinear structure of data as well as capture existing nonlinearities. The mapping function is defined as

$$\phi : \sum_{k=1}^{r} \bigcirc_{j=1}^{d} \mathbf{u}_k^{(j)} \rightarrow \sum_{k=1}^{r} \bigcirc_{j=1}^{d} \phi\left(\mathbf{u}_k^{(j)}\right),$$

which assumes mapping the tensor data into a tensor Hilbert space and then performing CPD. In other words, the feature map ϕ acts on the individual factors of CPD while retaining the low-rank structure of the data. The kernel will now be the standard inner product of tensors on the higher-dimensional space, i.e., one can find a dual structure-preserving kernel function by writing

$$\begin{aligned}
\kappa\left(\hat{\mathcal{X}}^{(i)}, \hat{\mathcal{X}}^{(j)}\right) &= \kappa\left(\sum_{k=1}^{r} \bigcirc_{\ell=1}^{d} \mathbf{u}_{ik}^{(\ell)}, \sum_{h=1}^{r} \bigcirc_{\ell=1}^{d} \mathbf{u}_{jh}^{(\ell)}\right) = \sum_{k,h=1}^{r} \kappa\left(\bigcirc_{\ell=1}^{d} \mathbf{u}_{ik}^{(\ell)}, \bigcirc_{\ell=1}^{d} \mathbf{u}_{jh}^{(\ell)}\right) \\
&= \sum_{k,h=1}^{r} \left\langle \phi\left(\bigcirc_{\ell=1}^{d} \mathbf{u}_{ik}^{(\ell)}\right), \phi\left(\bigcirc_{\ell=1}^{d} \mathbf{u}_{jh}^{(\ell)}\right)\right\rangle = \sum_{k,h=1}^{r} \left\langle \bigcirc_{\ell=1}^{d} \phi\left(\mathbf{u}_{ik}^{(\ell)}\right), \bigcirc_{\ell=1}^{d} \phi\left(\mathbf{u}_{jh}^{(\ell)}\right)\right\rangle \\
&= \sum_{k,h=1}^{r} \prod_{\ell=1}^{d} \left\langle \phi\left(\mathbf{u}_{ik}^{(\ell)}\right), \phi\left(\mathbf{u}_{jh}^{(\ell)}\right)\right\rangle = \sum_{k,h=1}^{r} \prod_{\ell=1}^{d} \kappa\left(\mathbf{u}_{ik}^{(\ell)}, \mathbf{u}_{jh}^{(\ell)}\right),
\end{aligned} \tag{6.21}$$

implying that to apply the kernel to two tensors, it suffices to apply it to the their factors in their respective CPD representations. The second equality results from the bilinearity of the kernel κ. Examples of commonly used kernels include $\kappa(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$ for the inner product kernel, and $\kappa(\mathbf{x}, \mathbf{y}) = \exp\left(-\sigma \|\mathbf{x} - \mathbf{y}\|_2^2\right)$ for the Gaussian kernel where σ controls the width of the kernel.

After choosing a kernel and thus knowing how it is related to the factor matrices $\mathbf{U}_m^{(j)}$, the gradient of the objective function with respect to $\mathbf{U}_m^{(j)}$ can be calculated and set to zero to solve for the updated $\mathbf{U}_m^{(j)}$. An explicit form of the gradient can be found in [7].

(d) Update b: Setting the first-order gradient of the objective function with respect to b to zero

yields the solution for b, i.e.,

$$\nabla_b = 2 \sum_{m=1}^{M_s} (z_m - y_m) = 0,$$

which can be solved as $\boldsymbol{\beta}$ has been estimated and $\hat{\mathbf{K}}$ is known.

Classification. After solving (6.20) using the training data, the shared kernel matrices $\{\mathbf{K}^{(j)}\}_{j=1}^{d}$ can be utilized to compute the KCP factorization of a test sample $\mathcal{X}$. Computing the CPD of $\mathcal{X}$ yields its CP factor matrices $\{\mathbf{U}^{(j)}\}_{j=1}^{d}$. Letting $\mathbf{K}^{(j)}\mathbf{V}^{(j)} = \mathbf{U}^{(j)}$ and thus $\mathbf{V}^{(j)} = \mathbf{K}^{(j)^{-1}}\mathbf{U}^{(j)}$, the KCP reconstruction of $\mathcal{X}$ is calculated using $\{\mathbf{V}^{(j)}\}_{j=1}^{d}$, i.e., $\hat{\mathcal{X}} = \mathcal{I} \times_{j=1}^{d} \mathbf{V}^{(j)}$. Finally, the test sample $\mathcal{X}$ can be classified according to

$$y(\mathcal{X}) = \operatorname{sign}\left(\sum_{m=1}^{M} \beta_m \, \kappa\left(\hat{\mathcal{X}}, \hat{\mathcal{X}}^{(m)}\right) + b \right).$$

using the reconstructed training samples. In this case, (6.21) can be used to calculate the kernel as

$$\kappa\left(\hat{\mathcal{X}}, \hat{\mathcal{X}}^{(m)}\right) = \sum_{k,h=1}^{r} \prod_{\ell=1}^{d} \kappa\left(\mathbf{v}_k^{(\ell)}, \mathbf{u}_{mh}^{(\ell)}\right),$$

instead of dealing with the actual reconstructed data. For the training samples, it is important to note that for $j \in [d]$ and $m \in [M]$, $\mathbf{U}_m^{(j)} = \mathbf{K}^{(j)}\mathbf{U}_m^{(j)}$ according to the reproducing property of $k_x^{(j)}$, which explains why the CP reconstruction of training data can be used in the kernel instead of their KCP reconstruction, as the KCP and CP reconstructions of the training samples are equivalent. However, the reproducing property of $\mathbf{K}^{(j)}$ will not necessarily hold for test data, and therefore, the assumption $\mathbf{K}^{(j)}\mathbf{V}^{(j)} = \mathbf{U}^{(j)}$ is made in this case, where $\mathbf{U}^{(j)}$ are the corresponding CP factor matrices of the test tensor.

6.4 Future Work

Directions for future research based on oblivious subspace embeddings include

- Extend the results of JL on tensors with low CP rank to tensors with low Tucker rank without orthogonality constraints. The goal is to obtain theoretical bounds on the deviation in the

norm and inner product of randomly projected tensors as well as lower bounds on embedding dimensions.

- In settings where data points lie on a low-rank subspace, the following can be investigated.
  - Compressed MPCA. One can use JL embeddings to project tensors before performing MPCA to solve for the principal components in the low-rank subspace.
  - Compressed Tensor SVM. One can use JL embeddings to project tensors with low-rank expansions before SVM classification. In other words, the separating hyperplane can be found in a lower-dimensional subspace.
- Modewise JL in third-order MBPT calculations where higher-order tensors are formed from lower-order data replicated in certain dimensions. As was seen, in this setting, non-orthogonal tensors sometimes become nearly orthogonal after projection. The role of repetitive patters forming the higher-dimensional tensors as well as sparsity can be investigated.

APPENDICES

APPENDIX A

USEFUL NOTIONS, DEFINITIONS, AND RELATIONS

In this appendix chapter, some of the terms and mathematical definitions used in derivations throughout the thesis are presented.

A.1 Operations on Rank-1 tensors

In this section, useful relations are presented that explain how rank-1 tensors are reshaped into vectors or unfoldings, and how j-mode products are done.

A.1.1 Reshaping rank-1 tensors

A.1.1.1 Vectorization

For a rank-1 tensor $\mathcal{X} = \bigcirc_{\ell=1}^{d} \mathbf{x}^{(\ell)}$, the vectorized form will be obtained according to

$$\text{vec}\,(\mathcal{X}) = \begin{cases} \bigotimes_{\ell=d}^{1} \mathbf{x}^{(\ell)} & \text{column} - \text{major}, \\ \bigotimes_{\ell=1}^{d} \mathbf{x}^{(\ell)} & \text{row} - \text{major}. \end{cases} \tag{A.1}$$

A.1.1.2 Mode-j Unfolding

The mode-j unfolding a rank-1 tensor $\mathcal{X} = \bigcirc_{\ell=1}^{d} \mathbf{x}^{(\ell)}$ can be calculated using

$$\mathbf{X}_{(j)} = \begin{cases} \mathbf{x}^{(j)} \bigcirc \text{vec}\left(\bigcirc_{\substack{\ell=1 \\ \ell \neq j}}^{d} \mathbf{x}^{(\ell)} \right) = \mathbf{x}^{(j)} \left(\mathbf{x}^{(d)} \otimes \cdots \otimes \mathbf{x}^{(j+1)} \otimes \mathbf{x}^{(j-1)} \otimes \cdots \otimes \mathbf{x}^{(1)} \right)^{\top}, & \text{column} - \text{major}, \\ \mathbf{x}^{(j)} \bigcirc \text{vec}\left(\bigcirc_{\substack{\ell=1 \\ \ell \neq j}}^{d} \mathbf{x}^{(\ell)} \right) = \mathbf{x}^{(j)} \left(\mathbf{x}^{(1)} \otimes \cdots \otimes \mathbf{x}^{(j-1)} \otimes \mathbf{x}^{(j+1)} \otimes \cdots \otimes \mathbf{x}^{(d)} \right)^{\top}, & \text{row} - \text{major}. \end{cases} \tag{A.2}$$

In the rest of this section, only the column-major relations will be presented and used as this has been the case throughout this thesis.

A.1.2 j-mode Product

For a rank-1 tensor $\mathcal{X} = \bigcirc_{\ell=1}^{d} \mathbf{x}^{(\ell)}$, the j-mode product can be done by applying the matrix of interest to the mode-j factor $\mathbf{x}^{(j)}$, i.e.,

$$\mathcal{X} \times_j \mathbf{A}^{(j)} = \bigcirc_{\ell=1}^{j-1} \mathbf{x}^{(\ell)} \bigcirc \left(\mathbf{A}^{(j)} \mathbf{x}^{(j)} \right) \bigcirc_{\ell=j+1}^{d} \mathbf{x}^{(\ell)} \tag{A.3}$$

The proof is simply done by considering the mode-j unfolding.

$$\left(\mathcal{X} \times_j \mathbf{A}^{(j)} \right)_{(j)} = \mathbf{A}^{(j)} \left(\bigcirc_{\ell=1}^{d} \mathbf{x}^{(\ell)} \right)_{(j)} = \mathbf{A}^{(j)} \mathbf{x}^{(j)} \bigcirc \operatorname{vec} \left(\bigcirc_{\substack{\ell=1 \\ \ell \neq j}}^{d} \mathbf{x}^{(\ell)} \right) \tag{A.4}$$

Reshaping to tensor form, (A.3) is obtained.

A more general relation involving all modes can be established through a similar approach. Consider a rank-1 tensor $\mathcal{X} = \bigcirc_{\ell=1}^{d} \mathbf{x}^{(\ell)}$, and let $\mathcal{Y} := \mathcal{X} \bigtimes_{\ell=1}^{d} \mathbf{A}^{(\ell)}$. Then, $\mathcal{Y}$ is also a rank-1 tensor, and

$$\mathcal{Y} = \bigcirc_{\ell=1}^{d} \mathbf{A}^{(\ell)} \mathbf{x}^{(\ell)}. \tag{A.5}$$

To show this, the mode-j unfolding of $\mathcal{Y}$ is calculated as follows.

$$\begin{aligned}
\mathbf{Y}_{(j)} &= \mathbf{A}^{(j)} \mathbf{X}_{(j)} \left(\mathbf{A}^{(d)} \otimes \cdots \otimes \mathbf{A}^{(j+1)} \otimes \mathbf{A}^{(j-1)} \otimes \cdots \otimes \mathbf{A}^{(1)} \right)^{\top} \\
&= \mathbf{A}^{(j)} \mathbf{x}^{(j)} \left(\mathbf{x}^{(d)} \otimes \cdots \otimes \mathbf{x}^{(j+1)} \otimes \mathbf{x}^{(j-1)} \otimes \cdots \otimes \mathbf{x}^{(1)} \right)^{\top} \\
&\quad \left(\mathbf{A}^{(d)} \otimes \cdots \otimes \mathbf{A}^{(j+1)} \otimes \mathbf{A}^{(j-1)} \otimes \cdots \otimes \mathbf{A}^{(1)} \right)^{\top} \\
&= \mathbf{A}^{(j)} \mathbf{x}^{(j)} \left[\left(\mathbf{A}^{(d)} \otimes \ldots \otimes \mathbf{A}^{(j+1)} \otimes \mathbf{A}^{(j-1)} \otimes \ldots \otimes \mathbf{A}^{(1)} \right) \left(\mathbf{x}^{(d)} \otimes \ldots \otimes \mathbf{x}^{(j+1)} \otimes \mathbf{x}^{(j-1)} \otimes \ldots \otimes \mathbf{x}^{(1)} \right) \right]^{\top} \\
&= \mathbf{A}^{(j)} \mathbf{x}^{(j)} \left(\mathbf{A}^{(d)} \mathbf{x}^{(d)} \otimes \cdots \otimes \mathbf{A}^{(j+1)} \mathbf{x}^{(j+1)} \otimes \mathbf{A}^{(j-1)} \mathbf{x}^{(j-1)} \otimes \cdots \otimes \mathbf{A}^{(1)} \mathbf{x}^{(1)} \right)^{\top}.
\end{aligned} \tag{A.6}$$

This is clearly the mode-j unfolding of a rank-1 tensor which can be formed by reshaping (A.6) to tensor structure to obtain (A.5). To get from the penultimate equality to the last one, the following matrix Kronecker product property was used.

$$\left(\mathbf{A}^{(1)} \otimes \cdots \otimes \mathbf{A}^{(d)} \right) \left(\mathbf{B}^{(1)} \otimes \cdots \otimes \mathbf{B}^{(d)} \right) = \mathbf{A}^{(1)} \mathbf{B}^{(1)} \otimes \cdots \otimes \mathbf{A}^{(d)} \mathbf{B}^{(d)}. \tag{A.7}$$

A.1.3 Inner Product

Consider rank-1 tensors $\mathcal{X} = \bigcirc_{\ell=1}^{d} \mathbf{x}^{(\ell)}$ and $\mathcal{Y} = \bigcirc_{\ell=1}^{d} \mathbf{y}^{(\ell)}$. To simplify the inner product defined as per (2.3), one can use the fact that an element of a rank-1 tensor at index set $(i_1, i_2, \ldots, i_d)$ is the product of the individual elements of its factor vectors at indices i_1, i_2, ... , and i_d. The inner product is then written as

$$\begin{aligned} \langle \mathcal{X}, \mathcal{Y} \rangle &= \sum_{i_1,\ldots,i_d} \mathcal{X}_{i_1,\ldots,i_d} \overline{\mathcal{Y}}_{i_1,\ldots,i_d} = \sum_{i_1,\ldots,i_d} \mathbf{x}_{i_1}^{(1)} \ldots \mathbf{x}_{i_d}^{(d)} \overline{\mathbf{y}_{i_1}^{(1)} \ldots \mathbf{y}_{i_d}^{(d)}} \\ &= \sum_{i_1} \mathbf{x}_{i_1}^{(1)} \overline{\mathbf{y}_{i_1}^{(1)}} \cdots \sum_{i_d} \mathbf{x}_{i_d}^{(1)} \overline{\mathbf{y}_{i_d}^{(1)}} = \prod_{\ell=1}^{d} \langle \mathbf{x}^{(\ell)}, \mathbf{y}^{(\ell)} \rangle. \end{aligned} \tag{A.8}$$

This means the inner product of rank-1 tensors boils down to the inner product of their factor vectors.

A.2 ε-nets as a Means of Set Discretization

This is a useful discretization concept that allowes for obtaining large deviation inequalities when assessing the stattistical properties of random variables/vectors/matrices/tensors [25].

Definition A.2.1 (ε-net) *Let $(\mathcal{S}, d)$ be a metric space, and let $\mathcal{S}_0$ be a subset of S. A subset $\mathcal{N} \subseteq \mathcal{S}_0$ is called an ε-net of $\mathcal{S}_0$ if for $\varepsilon > 0$, we have*

$$\forall x \in \mathcal{S}_0 \;\; \exists x_0 \in \mathcal{N} : d(x, x_0) \leq \varepsilon, \tag{A.9}$$

meaning any given point in $\mathcal{S}_0$ is within a distance ε of a point in $\mathcal{N}$. This also means that $\mathcal{N}$ is an ε-net of $\mathcal{S}_0$ if and only if $\mathcal{S}_0$ can be fully covered by balls centered in the elements of $\mathcal{N}$ and with radii ε.

It can be observed that the balls of radii ε in the above definition may have overlaps. This puts a lower bound on the number for such balls, paving the way for the following definition.

Definition A.2.2 (Covering Number) *The smallest possible cardinality of an ε-net $\mathcal{N}$ of $\mathcal{S}_0$ in Definition A.2.1 is called the convering number of $\mathcal{S}_0$ and is denoted by $C(\mathcal{S}_0, \varepsilon)$. Equivalently, it*

is the smallest number of closed balls with centers in the elements of $\mathcal{N}$ and radii ε whose union covers $\mathcal{S}_0$.

Note A.2.1 *The covering number of the Euclidean ball B_2^n satisfies the following inequality.*

$$\left(\frac{1}{\varepsilon}\right)^n \leq C(B_2^n, \varepsilon) \leq \left(1 + \frac{2}{\varepsilon}\right)^n \tag{A.10}$$

A.3 Random Projections and Johnson-Lindenstrauss Embeddings

In this section, a brief introduction to random projections in the context of JL embeddings is presented. More details can be found in [25]. Suppose that we have N data points $\{\mathbf{x}_i \in \mathbb{R}^n\}_{i=1}^N$ and we want to project them onto $\{\mathbf{y}_i \in \mathbb{R}^m\}_{i=1}^N$ where $m \ll n$, in a way that the geometry of the data is preserved, i.e.,

$$\left\|\mathbf{x}_i - \mathbf{x}_j\right\| \approx \left\|\mathbf{y}_i - \mathbf{y}_j\right\| \quad \text{for } i \neq j.$$

We would also like to know what the smallest m that make this possible is. First, we review notions that will be later useful. A general group of random variables called sub-Gaussian random variables are of special interest in this discussion.

Definition A.3.1 (Sub-Gaussian Random Variable) *For a sub-Gaussian random variable X, the following properties hold and are equivalent. The constants K_i differ from each other by an absolute constant.*

1. $\mathbb{P}\{|X| > t\} \leq 2\exp\left(-t^2/K_1^2\right)$, *for all* $t > 0$.
2. $\|X\|_p := \left(\mathbb{E}\,|X|^p\right)^{1/p} \leq K_2\sqrt{p}$, *for all* $p > 1$.
3. *The MGF of* X^2 *satisfies* $\mathbb{E}\exp\left(\lambda^2 X^2\right) \leq \exp\left(K^3\lambda^2\right)$ *for all* λ *such that* $|\lambda| < 1/K_3$.
4. *The MGF of* X^2 *is bounded at some point, namely,* $\mathbb{E}\exp\left(X^2/K_4^2\right) \leq 2$.
5. *If* $\mathbb{E}\,X = 0$, *the above four properties are equal to the MGF of* X *satisfying* $\mathbb{E}\exp\left(\lambda X\right) \leq \mathbb{E}\exp\left(K_5^2\lambda^2\right)$, *for all* $\lambda \in \mathbb{R}$.

All Guassian random variables and bounded random variables are sub-gaussian.

Definition A.3.2 (Sub-Gaussian Norm) *The sub-Gaussian norm of a random variable X, denoted by $\|X\|_{\psi_2}$, is defined as*

$$\|X\|_{\psi_2} = \inf \left\{ t > 0 : \mathbb{E} \exp\left(X^2/t^2\right) \leq 2 \right\}.$$

An alternative definition of the sub-Gaussian norm of a random variable X is

$$\|X\|_{\psi_2} = \sup_{p \geq 1} p^{-\frac{1}{2}} \|X\|_p .$$

Definition A.3.3 (Random Subspace) *Let $G_{n,m}$ denote the Grassmannian[1], i.e., the set of all m-dimensional subspaces in $\mathbb{R}^n$. We say that E is a random m-dimensional subspace of $\mathbb{R}^n$ uniformly distributed in $G_{n,m}$ if E is a random m-dimensional subspace of $\mathbb{R}^n$ whose distribution is rotation-invariant, i.e.,*

$$\mathbb{P}\{E \in \mathcal{E}\} = \mathbb{P}\{\mathbf{U}(E) \in \mathcal{E}\},$$

for any fixed subset $\mathcal{E} \subset G_{n,m}$, where $\mathbf{U}$ is an $n \times n$ orthogonal matrix.

Proposition A.3.1 (Random Projection) *Let $\mathbf{P}$ be a projection from $\mathbb{R}^n$ onto a random m-dimensional subspace of $\mathbb{R}^n$ uniformly distributed in $G_{n,m}$. Let $\mathbf{z} \in \mathbb{R}^n$ be a fixed point and $\varepsilon > 0$. Then,*

1. $\| \|\mathbf{Pz}\|_2 \|_2 := \left(\mathbb{E} \|\mathbf{Pz}\|_2^2 \right)^{1/2} = \sqrt{\frac{m}{n}} \|\mathbf{z}\|_2$.
2. *With probability at least $1 - 2\exp\left(-c\varepsilon^2 m\right)$, we have that*

$$(1-\varepsilon)\sqrt{\frac{m}{n}} \|\mathbf{z}\|_2 \leq \|\mathbf{Pz}\|_2 \leq (1+\varepsilon)\sqrt{\frac{m}{n}} \|\mathbf{z}\|_2 ,$$

where c is an absolute contant.

[1] Also called the Grassmann manifold.

Proof *Since we can normalize the terms by dividing by* $\|\mathbf{z}\|_2$*, without loss of generality, we may assume that* $\|\mathbf{z}\|_2 = 1$*. Using rotation-invariance, one can show that the random projection of a fixed point is equivalent in distribution to the fixed projection of a random point uniformly distributed on the unit sphere* $S^{n-1} \in \mathbb{R}^n$*. This is denoted by* $\mathbf{z} \sim \text{uniform}\left(S^{n-1}\right)$*. We choose this fixed projection to be in such a way that it picks the first m entries of* $\mathbf{z} \in \mathbb{R}^n$*, i.e.,*

$$\mathbf{Pz} = [z_1, z_2, \ldots, z_m]^\top.$$

In other words, $\mathbf{P} = [\mathbf{I}_m | \, \mathbf{0}]_{m \times n}$*, where* $\mathbf{0} \in \mathbb{R}^{m \times (n-m)}$ *is a matrix of all zeros. Therefore, we have that*

$$\mathbb{E} \, \|\mathbf{Pz}\|_2^2 = \mathbb{E} \sum_{i=1}^{m} z_i^2 = \sum_{i=1}^{m} \mathbb{E} z_i^2 = m \mathbb{E} z_i^2,$$

since all z_i *are drawn from the same distribution. We also know that* $\|\mathbf{z}\|_2^2 = 1$*. Taking expectation of both sides we can write*

$$\mathbb{E} \sum_{i=1}^{n} z_i^2 = 1,$$

resulting in $\mathbb{E} z_i^2 = 1/n$*. Therefore,* $\mathbb{E} \, \|\mathbf{Pz}\|_2^2 = m/n$*. This proves* 1*.*

To prove 2*, we may use the large deviation inequality for the concentration of Lipschitz-continuous functions on the unit sphere.*[2] *In particular, if* $f(\mathbf{z})$ *is Lipschitz-continuous, and* $\mathbf{z} \sim \text{uniform}\left(S^{n-1}\right)$*, then*

$$\mathbb{P}\left\{|f(\mathbf{z}) - \|f(\mathbf{z})\|_2| > t\right\} \leq 2\text{exp}\left(-\frac{cnt^2}{L^2}\right),$$

where c *is an absolute constant and* L *is the Lipschitz constant of* $f(\mathbf{z})$*.*[3] *Here,* $\|f(\mathbf{z})\|_2 = \left(\mathbb{E}\,|f(\mathbf{z})|^2\right)^{1/2}$ *as* $f(\mathbf{z})$ *is a random variable. In our case,* $f(\mathbf{z}) = \|\mathbf{Pz}\|_2$*. Since* $f(\mathbf{z})$ *is Lipschitz, and therefore continuous, we can use the mean value theorem to write*

$$|f(\mathbf{z}) - f(\mathbf{y})| = \left|\nabla^\top f\left(\mathbf{z}_0\right)(\mathbf{z} - \mathbf{y})\right| \leq \|\nabla f\left(\mathbf{z}_0\right)\|_2 \, \|\mathbf{z} - \mathbf{y}\|_2$$

[2] If $f : \mathbb{R}^n \to \mathbb{R}$ is Lipschitz with Lipschitz constant L, then $f(\mathbf{x})$ is sub-Gaussian for $\mathbf{x} \sim \text{uniform}\left(S^{n-1}\right)$, and we have that $\mathbb{P}\left(|f(\mathbf{x}) - M| > t\right) \leq 2\text{exp}\left(-\frac{nt^2}{2L^2}\right)$ for $t > 0$, where M is the median of f. This means with high probability, $f(\mathbf{x}) \approx M$ on the unit sphere. A similar inequality can be obtained when M is replaced with $\mathbb{E}f(\mathbf{z})$ which, in turn, can be substituted by $\|f(\mathbf{z})\|_2$. For details, see Chapter 5 in [25].

[3] Here, we have also replaced $\mathbb{E}f(\mathbf{z})$ by $\|f(\mathbf{z})\|_2$ in the large deviation inequality [25].

where $\mathbf{y}, \mathbf{z} \in \mathbb{R}^n$ *and* $\mathbf{z}_0$ *is a point on the line segment between* $\mathbf{z}$ *and* $\mathbf{y}$*. We can bound the gradient of* $f(\mathbf{z})$ *to show that* $\|f(\mathbf{z}) - f(\mathbf{y})\|_2 \leq \|\mathbf{z} - \mathbf{y}\|_2$ *meaning that we can let* $L = 1$*. On the other hand, we may rearrange the inequalily in* 2 *as*

$$\left| \|\mathbf{P}\mathbf{z}\|_2 - \sqrt{\frac{m}{n}} \|\mathbf{z}\|_2 \right| \leq \varepsilon \sqrt{\frac{m}{n}} \|\mathbf{z}\|_2$$

Noting that $t = \varepsilon\sqrt{\frac{m}{n}} \|\mathbf{z}\|_2$ *in the large deviation inequality, and using the result from* 1*, we can obtain the desired result.*

Lemma A.3.1 (Johnson-Lindenstrauss Lemma for Point Sets) *Let* $X \subseteq \mathbb{R}^n$ *be a set of* N *points in* $\mathbb{R}^n$*. Then, for an absolute constant* C*, there exists a linear map* $\mathbf{A} = \sqrt{\frac{n}{m}}\, \mathbf{P} \in \mathbb{R}^{m \times n}$ *where* $m \geq C\frac{\log N}{\varepsilon^2}$ *such that for all* $\mathbf{x}, \mathbf{y} \in X$*,*

$$(1 - \varepsilon) \|\mathbf{x} - \mathbf{y}\|_2 \leq \|\mathbf{A}(\mathbf{x} - \mathbf{y})\|_2 \leq (1 + \varepsilon) \|\mathbf{x} - \mathbf{y}\|_2$$

with probability at least $1 - 2\exp\left(-Cm\varepsilon^2\right)$*. This means that* $\mathbf{A}$ *is an approximate isometry on* X*.*

Proof *We showed how the random projection* $\mathbf{P}$ *acts on a vector in* X*. Now, consider the difference set* $X - X := \{\mathbf{x} - \mathbf{y} \mid \mathbf{x}, \mathbf{y} \in X\}$*. We want to show for all* $\mathbf{z} \in X - X$*,*

$$(1 - \varepsilon) \|\mathbf{z}\|_2 \leq \|\mathbf{A}\mathbf{z}\|_2 \leq (1 + \varepsilon) \|\mathbf{z}\|_2 .$$

Replacing $\mathbf{A}$ *by* $\sqrt{\frac{n}{m}}\mathbf{P}$*, we have*

$$\sqrt{\frac{m}{n}} (1 - \varepsilon) \|\mathbf{z}\|_2 \leq \|\mathbf{P}\mathbf{z}\|_2 \leq \sqrt{\frac{m}{n}} (1 + \varepsilon) \|\mathbf{z}\|_2 ,$$

which we already know holds with probability at least $1 - 2\exp\left(cm\varepsilon^2\right)$ from Proposition A.3.1. Now, taking union bound over all points in X, we may conclude that the desired result holds with probability at least

$$1 - 2|X - X| \exp\left(-cm\varepsilon^2\right) = 1 - 2N^2 \exp\left(-cm\varepsilon^2\right) = 1 - \exp\left(-cm\varepsilon^2 + \log 2N^2\right).$$

Since this probability must be non-negative, we have that

$$cm\varepsilon^2 \geq \log 2N^2,$$

and therefore,

$$m \geq C\frac{\log N}{\varepsilon^2},$$

completing the proof.

APPENDIX B

MEMORY-EFFICIENT MODE-WISE PROJECTION CALCULATIONS OF THE ENERGY TERMS

In the following, it is assumed that n_j, $j \in \{1,2,3\}$ is the dimension size of the 3-mode tensors as the reshaped versions of the hypothetical 6-mode arrays, and m_j is the corresponding size after projection.

B.1 Particle-Particle

Denoting the projected version of $\mathcal{H}_1$ by $\mathcal{P}_1$, one can calculate its elements by

$$\begin{aligned}\mathcal{P}_1\left(i_1,i_2,i_3\right) &= \sum_{p,q,r} \mathcal{H}\left(p,q\right)\mathcal{H}\left(q,r\right)\mathbf{A}^{(1)}\left(i_1,p\right)\mathbf{A}^{(2)}\left(i_2,q\right)\mathbf{A}^{(3)}\left(i_3,r\right) \\ &= \sum_{q} \mathbf{A}^{(2)}\left(i_2,q\right)\left(\sum_{p}\mathbf{A}^{(1)}\left(i_1,p\right)\mathcal{H}\left(p,q\right)\right)\left(\sum_{r}\mathbf{A}^{(3)}\left(i_3,r\right)\mathcal{H}^{T}\left(r,q\right)\right),\end{aligned} \tag{B.1}$$

for $i_j \in [m_j]$, $j \in \{1,2,3\}$. Now, letting $\mathcal{H}' = \mathbf{A}^{(1)}\mathcal{H}$ and $\mathcal{H}'' = \mathbf{A}^{(3)}\mathcal{H}^T$, it can be observed that $\mathcal{P}_1 = \mathcal{H}''' \times_2 \mathbf{A}^{(2)}$ where $\mathcal{H}'''$ is a tensor that is formed element-wise according to

$$\mathcal{H}'''\left(i_1,q,i_3\right) = \mathcal{H}'\left(i_1,q\right)\mathcal{H}''\left(i_3,q\right).$$

Indeed, the mode-2 unfolding of $\mathcal{H}'''$ is the result of the Hadamard product of the $n_2 \times m_1 m_3$ matrices $\mathbf{G}_1$ and $\mathbf{G}_2$, where $\mathbf{G}_1$ is formed by replicating $\mathcal{H}'^T$ across its second dimension m_3 times, and $\mathbf{G}_2$ is formed by replicating each column of $\mathcal{H}''^T$ m_1 times[1]. Left-multiplying the mode-2 unfolding of $\mathcal{H}'''$ by $\mathbf{A}^{(2)}$ and folding back to tensor shape yields $\mathcal{P}_1$.

Letting $\mathcal{P}_2$ denote the projected version of $\mathcal{H}_2$, it is observed that

[1]For implementation, one does not have to store replicated versions of data as this would be inefficient use of memory.

$$\begin{aligned}\mathcal{P}_2\left(i_1,i_2,i_3\right) &= \sum_{p,q,r} \tilde{\mathcal{H}}\left(r,p\right)\mathcal{D}\left(q,p\right)\mathbf{A}^{(1)}\left(i_1,p\right)\mathbf{A}^{(2)}\left(i_2,q\right)\mathbf{A}^{(3)}\left(i_3,r\right) \\ &= \sum_{p}\mathbf{A}^{(1)}\left(i_1,p\right)\left(\sum_{q}\mathbf{A}^{(2)}\left(i_2,q\right)\mathcal{D}\left(q,p\right)\right)\left(\sum_{r}\mathbf{A}^{(3)}\left(i_3,r\right)\tilde{\mathcal{H}}^{T}\left(r,p\right)\right),\end{aligned} \tag{B.2}$$

Again, letting $\mathcal{H}' = \mathbf{A}^{(2)}\mathcal{D}$ and $\mathcal{H}'' = \mathbf{A}^{(3)}\tilde{\mathcal{H}}$, it can be observed that $\mathcal{P}_2 = \mathcal{H}''' \times_1 \mathbf{A}^{(1)}$ where $\mathcal{H}'''$ is defined element-wise by

$$\mathcal{H}'''\left(p,i_2,i_3\right) = \mathcal{H}'\left(i_2,p\right)\mathcal{H}''\left(i_3,p\right).$$

The mode-1 unfolding of $\mathcal{H}'''$ is obtained by the Hadamard product of the $n_1 \times m_2 m_3$ matrices $\mathbf{G}_1$ and $\mathbf{G}_2$, where $\mathbf{G}_1$ is formed by replicating $\mathcal{H}'^{T}$ across its second dimension m_3 times, and $\mathbf{G}_2$ is formed by replicating each column of $\mathcal{H}''^{T}$ m_2 times. It suffices now to left-multiply the mode-1 unfolding of $\mathcal{H}'''$ by $\mathbf{A}^{(1)}$ followed by folding back to tensor form to obtain $\mathcal{P}_2$.

B.2 Hole-Hole

Calculations are done in a similar way to that of Appendix B.1. The projection of $\mathcal{H}_1$ will be exactly the same. For $\mathcal{H}_2$, one can write

$$\begin{aligned}\mathcal{P}_2\left(i_1,i_2,i_3\right) &= \sum_{p,q,r} \tilde{\mathcal{H}}\left(r,p\right)\mathcal{D}\left(r,q\right)\mathbf{A}^{(1)}\left(i_1,p\right)\mathbf{A}^{(2)}\left(i_2,q\right)\mathbf{A}^{(3)}\left(i_3,r\right) \\ &= \sum_{r}\mathbf{A}^{(3)}\left(i_3,r\right)\left(\sum_{p}\mathbf{A}^{(1)}\left(i_1,p\right)\tilde{\mathcal{H}}^{T}\left(p,r\right)\right)\left(\sum_{q}\mathbf{A}^{(2)}\left(i_2,q\right)\mathcal{D}^{T}\left(q,r\right)\right),\end{aligned} \tag{B.3}$$

Letting $\mathcal{H}' = \mathbf{A}^{(1)}\tilde{\mathcal{H}}^{T}$ and $\mathcal{H}'' = \mathbf{A}^{(2)}\mathcal{D}^{T}$, it can be observed that $\mathcal{P}_2 = \mathcal{H}''' \times_3 \mathbf{A}^{(3)}$ where $\mathcal{H}'''$ is formed element-wise according to

$$\mathcal{H}'''\left(i_1,i_2,r\right) = \mathcal{H}'\left(i_1,r\right)\mathcal{H}''\left(i_2,r\right).$$

The mode-3 unfolding of $\mathcal{H}'''$ is obtained by the Hadamard product of the $n_3 \times m_1 m_2$ matrices $\mathbf{G}_1$ and $\mathbf{G}_2$, where $\mathbf{G}_1$ is formed by replicating $\mathcal{H}'^{T}$ across its second dimension m_2 times, and $\mathbf{G}_2$

is formed by replicating each column of $\mathcal{H}''^T$ m_1 times, where the replication is not performed explicitly to save time and memory as mentioned above. The remaining step is to left-multiply the mode-3 unfolding of $\mathcal{H}'''$ by $\mathbf{A}^{(3)}$ and fold back to tensor shape to get $\mathcal{P}_2$.

B.3 Particle-Hole

The projection of $\mathcal{H}_1$ can be calculated using

$$\mathcal{P}_1\left(i_1,i_2,i_3\right)=\sum_{p,q,r}\tilde{\mathcal{H}}_1\left(p,q\right)\mathcal{H}_p\left(q,r\right)\mathbf{A}^{(1)}\left(i_1,p\right)\mathbf{A}^{(2)}\left(i_2,q\right)\mathbf{A}^{(3)}\left(i_3,r\right), \tag{B.4}$$

which is the same as (B.1) after replacing $\mathcal{H}\left(q,r\right)$ by $\mathcal{H}_p\left(q,r\right)$ and $\mathcal{H}\left(p,q\right)$ by $\tilde{\mathcal{H}}_1\left(q,r\right)$ in (B.1).

For $\mathcal{H}_2$, the projected tensor $\mathcal{P}_2$ is calculated using

$$\begin{aligned}\mathcal{P}_2\left(i_1,i_2,i_3\right)&=\sum_{p,q,r}\tilde{\mathcal{H}}_2\left(r,p\right)\mathcal{H}_p\mathbf{A}^{(1)}\left(i_1,p\right)\mathbf{A}^{(2)}\left(i_2,q\right)\mathbf{A}^{(3)}\left(i_3,r\right)\\&=\sum_{q}\mathbf{A}^{(2)}\left(i_2,q\right)\left(\sum_{p}\mathbf{A}^{(1)}\left(i_1,p\right)\left(\sum_{r}\mathbf{A}^{(3)}\left(i_3,r\right)\tilde{\mathcal{H}}_2\left(r,p\right)\right)\right)\\&=\mathcal{H}''\left(i_1,i_3\right)\sum_{q}\mathbf{A}^{(2)}\left(i_2,q\right).\end{aligned} \tag{B.5}$$

where $\mathcal{H}''=\mathbf{A}^{(1)}\mathcal{H}'^T$ and $\mathcal{H}'=\mathbf{A}^{(3)}\tilde{\mathcal{H}}_2$.

APPENDIX C

FASTER KRONECKER JOHNSON-LINDENTRAUSS TRANSFORM

In this appendix, the idea behind the method proposed in [10] is outlined in short. Here, the notation is kept similar to that of [10], and is slightly different from the notation used in Section 4.2.3. Assume that $N = \prod_{k=1}^{d} n_k$, and consider the JL matrix

$$\mathbf{\Phi} = \sqrt{\frac{N}{m_{\text{kron}}}}\, \mathbf{S} \bigotimes_{k=d}^{1} \left(\mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \right) \in \mathbb{C}^{m_{\text{kron}} \times N} \tag{C.1}$$

where $\mathbf{S} \in \mathbb{R}^{m_{\text{kron}}}$ is a random sampling matrix (similar to $\mathbf{R}$ in **RFD**), $\mathbf{D}_{\xi_{n_k}} \in \mathbb{R}^{n_k \times n_k}$ is a diagonal matrix with Rademacher random variables on its diagonal, and $\mathbf{F}_{n_k} \in \mathbb{C}^{n_k \times n_k}$ is the unitary DFT matrix. If a vector $\mathbf{x} \in \mathbb{C}^{N \times 1}$ has Kronecker structure, meaning it can be written as $\mathbf{x} = \bigotimes_{d}^{1} \mathbf{x}_k$ or equivalently, the vectorized form of a rank-1 tensor $\mathcal{X} = \bigcirc_{k=1}^{d} \mathbf{x}_k$, then one can observe that

$$\begin{aligned}
\mathbf{\Phi x} &= \sqrt{\frac{N}{m_{\text{kron}}}}\, \mathbf{S} \bigotimes_{k=d}^{1} \left(\mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \right) \bigotimes_{k=d}^{1} \mathbf{x}_k \\
&= \sqrt{\frac{N}{m_{\text{kron}}}}\, \mathbf{S} \bigotimes_{k=d}^{1} \left(\mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \mathbf{x}_k \right) = \sqrt{\frac{N}{m_{\text{kron}}}}\, \mathbf{S}\, \text{vec} \left(\bigcirc_{k=d}^{1} \mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \mathbf{x}_k \right) \\
&= \sqrt{\frac{N}{m_{\text{kron}}}}\, \mathbf{S}\, \text{vec} \left[\left(\bigcirc_{k=d}^{1} \mathbf{x}_k \right) \bigtimes_{k=1}^{d} \left(\mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \right) \right].
\end{aligned} \tag{C.2}$$

It is easy to see that applying $\mathbf{\Phi}$ to $\mathbf{x}$ defined above is equivalent to performing a modewise fast JL embedding to a rank-1 tensor whose vectorized form is $\mathbf{x}$, meaning $\mathbf{x} = \text{vec}(\mathcal{X}) = \text{vec}(\bigcirc_{k=1}^{d} \mathbf{x}_k)$. The embedding applied to mode k is $\mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}}$. Next, the result is vectorized and the random restriction matrix $\mathbf{S}$ is applied followed by the factor $\sqrt{\frac{N}{m_{\text{kron}}}}$.

The computational cost will be low since if $\mathbf{y}_k := \mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \mathbf{x}_k$, then one can see that

$$\mathcal{Y} := \bigcirc_{k=1}^{d} \mathbf{y}_k = \bigcirc_{k=1}^{d} \mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \mathbf{x}_k = \mathcal{X} \bigtimes_{k=1}^{d} \left(\mathbf{F}_{n_k} \mathbf{D}_{\xi_{n_k}} \right)$$

and $\mathbf{y} := \text{vec}\,(\mathcal{Y}) = \bigotimes_{k=1}^{d} \mathbf{y}_k$. Therefore, when applying the random sampling matrix S, if one knows which indices in $\mathbf{y}$ are being picked, those indices can be translated to indices in $\mathbf{y}_k$, meaning

that calculations will only be done for those specific indices. Assume we are interested in finding element i_k of $\mathbf{y}_k$. For $i_k \in [n_k]$ and $k \in [d]$, we have that

$$\mathbf{y}_k(i_k) = \sum_{j=1}^{n_k} \left(\mathbf{F}_{n_k}\mathbf{D}_{\xi_{n_k}}\right)_{i_k,j} (\mathbf{x}_k)_j \,.$$

On the other hand,

$$\left(\mathbf{F}_{n_k}\mathbf{D}_{\xi_{n_k}}\right)_{i_k,j} = \left(\left(\mathbf{F}_{n_k}\right)_{i_k,:} * \mathbf{d}_{\xi_{n_k}}\right)_j$$

where $*$ is the elementwise multiplication and $\mathbf{d}_{\xi_{n_k}}$ denotes the diagonal of $\mathbf{D}_{\xi_{n_k}}$. Therefore,

$$\mathbf{y}_k(i_k) = \left\langle \left(\mathbf{F}_{n_k}\right)_{i_k,:} * \mathbf{d}_{\xi_{n_k}}, \mathbf{x}_k \right\rangle .$$

This means in each mode, all one needs is $\mathbf{d}_{\xi_{n_k}}$ and the $i_k{}^{\text{th}}$ row of $\mathbf{F}_{n_k}$. This significantly reduces the computational cost as in practice, it is the case that $m_{\text{kron}} \ll N$.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Alzheimer's disease neuroimaging initiative. http://adni.loni.usc.edu/.

[2] A mathematical introduction to fast and memory efficient algorithms for big data. https://users.math.msu.edu/users/iwenmark/Notes_Fall2020_Iwen_Classes.pdf.

[3] T. D. Ahle, M. Kapralov, J. B. Knudsen, R. Pagh, A. Velingker, D. P. Woodruff, and A. Zandieh. Oblivious sketching of high-degree polynomial kernels. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 141–160. SIAM, 2020.

[4] R. Bro and H. A. Kiers. A new efficient method for determining the number of components in parafac models. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 17(5):274–286, 2003.

[5] S. Foucart and H. Rauhut. *A mathematical introduction to compressive sensing*. Springer, 2013.

[6] Z. Hao, L. He, B. Chen, and X. Yang. A linear support higher-order tensor machine for classification. *IEEE Transactions on Image Processing*, 22(7):2911–2920, July 2013.

[7] L. He, C.-T. Lu, G. Ma, S. Wang, L. Shen, P. S. Yu, and A. B. Ragin. Kernelized support tensor machines. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1442–1451. JMLR. org, 2017.

[8] M. Iwen and B. Ong. A distributed and incremental SVD algorithm for agglomerative data analysis on large networks. *SIAM Journal on Matrix Analysis and Applications*, 37(4):1699–1718, 2016.

[9] M. A. Iwen, D. Needell, E. Rebrova, and A. Zare. Lower memory oblivious (tensor) subspace embeddings with fewer random bits: modewise methods for least squares. *SIAM Journal on Matrix Analysis and Applications*, 42(1):376–416, 2021.

[10] R. Jin, T. G. Kolda, and R. Ward. Faster Johnson–Lindenstrauss transforms via kronecker products. *arXiv preprint arXiv:1909.04801*, 2019.

[11] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to platt's smo algorithm for svm classifier design. *Neural computation*, 13(3):637–649, 2001.

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

[13] F. Krahmer and R. Ward. New and improved Johnson–Lindenstrauss embeddings via the restricted isometry property. *SIAM Journal on Mathematical Analysis*, 43(3):1269–1281, 2011.

[14] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. Mpca: Multilinear principal component analysis of tensor objects. *IEEE Transactions on Neural Networks*, 19(1):18–39, Jan 2008.

[15] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100), 1996.

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

[17] A. Ozdemir, A. Zare, M. A. Iwen, and S. Aviyente. Multiscale analysis for higher-order tensors. In *Wavelets and Sparsity XVIII*, volume 11138, page 1113808. International Society for Optics and Photonics, 2019.

[18] R. Pagh. Compressed matrix multiplication. *ACM Transactions on Computation Theory (TOCT)*, 5(3):1–17, 2013.

[19] N. Pham and R. Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–247, 2013.

[20] Y. Shi and A. Anandkumar. Higher-order count sketch: Dimensionality reduction that retains efficient tensor operations, 2019.

[21] N. D. Sidiropoulos and R. Bro. On the uniqueness of multilinear decomposition of n-way arrays. *Journal of Chemometrics*, 14(3):229–239, 2000.

[22] Y. Sun, Y. Guo, C. Luo, J. Tropp, and M. Udell. Low-rank tucker approximation of a tensor from streaming data. *SIAM Journal on Mathematics of Data Science*, 2(4):1123–1150, 2020.

[23] Y. Sun, Y. Guo, J. A. Tropp, and M. Udell. Tensor random projection for low memory dimension reduction. In *NeurIPS Workshop on Relational Representation Learning*, 2018.

[24] D. Tao, X. Li, X. Wu, W. Hu, and S. Maybank. Supervised tensor learning, knowledge and information systems. 2007.

[25] R. Vershynin. High-dimensional probability: An introduction with applications in data science. cambridge series in statistical and probabilistic mathematics, 2018.

[26] X. Liu and N. D. Sidiropoulos. Cramer-rao lower bounds for low-rank decomposition of multidimensional arrays. *IEEE Transactions on Signal Processing*, 49(9):2074–2086, Sep. 2001.

[27] A. Zare, A. Ozdemir, M. A. Iwen, and S. 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.