Skolkovo Institute of Science and Technology

Automatic Noise and Artifacts Removal from Biomedical Signals and Images Using Tensor Completion

Doctoral Thesis
by

Maame Gyamfua Asante-Mensah

Doctoral Program in Computational and Data Science and Engineering

Supervisor
Professor Andrzej Cichocki

Moscow - 2023

I hereby declare that the work presented in this thesis was carried out by myself at Skolkovo Institute of Science and Technology, Moscow, except where due acknowledgement is made, and has not been submitted for any other degree.

Candidate (Maame Gyamfua Asante-Mensah)

Supervisor (Professor Andrzej Cichocki)

Abstract

The removal of noise and artifacts from images and signals (especially in biomedical signals and neuroimages) is very important in various applications and an active research area. In particular, in biomedical applications, the reduction of noise and enhancement of the resolution of neuroimages are significant pre-processing tasks.

This research investigates new methods and approaches that allow us to remove huge noise (outliers) and various artifacts in MRI and EEG signals using low-rank (typically high-order) tensor decomposition techniques and formulate the problem as a constrained optimization problem that is solved by modified ADMM with minimization of the tensor nuclear norm and imposing additional sparsity or smoothness constraints. The corrupted noise or artifact samples of signals and pixels or voxels of images or videos are identified, and novel techniques used in matrix or tensor completion are applied.

This thesis explores and compares various block Hankel folding strategies and tensorization techniques in order to transform low-order tensors (vectors, matrices, and 3D tensors) into higher-order block Hankel tensors before applying tensor completion via tensor train (TT) and tensor ring (TR) low-rank decompositions. This hankelization step ensures the low rank property of the incomplete data and also helps in improving the accuracy of reconstruction of images and signals corrupted by large noise. Moreover, the thesis applied convolutional neural network as one of the major Hankelization steps in tensorization.

Furthermore, the thesis investigates tensor cross approximation models and associated algorithms in order to process corrupted, noisy data quickly and efficiently. The thesis has three main contributions:

1. The first developed approach exploits a sparse representation for tensor ring cores using dictionary learning. The thesis proposed and implemented an efficient procedure that allows for the transformation of available data into higher-order tensors, and the core tensors are estimated using tensor ring decomposition. Then, using an over complete discrete cosine dictionary, sparse codes are generated. The sparse codes, together with the estimated core ten-

sors, are optimized using the ALS (Alternating Least Squares) gradient descent algorithm.

2. In the second alternative approach, the thesis investigates the 3D tensor completion task by computing the modified CUR approximations with smoothing constraints for the underlying data tensor. In each iteration of our algorithm, a CUR approximation of the underlying data tensor is computed, followed by the mask operator. The developed algorithms have low complexity and are very fast. For structurally missing data components or a high missing rate, the algorithm incorporates an efficient smooth variant of the developed tensor CUR algorithm, which first makes the sampled components smooth and then the CUR algorithm is applied.

3. In addition, this research also tackles the problem of motion artifacts in diffusion-weighted MRI images. The developed algorithm employs the tubal norm of the tensor for reconstructing MRI images from a few samples of MRI data in the Fourier transform.

Extensive computer simulations performed using all three strategies show the effectiveness and superiority of the methods used in the research.

Publications

1. Maame G Asante-Mensah, Salman Ahmadi-Asl, and Andrzej Cichocki. Matrix and tensor completion using tensor ring decomposition with sparse representation. *Machine Learning: Science and Technology*, 2(3):035008, 2021. **Scopus, Q1. First author.**

   Author's contributions in the paper include: idea, design, implementation, theoretical analysis, experiments, draft, and revisions.

2. Salman Ahmadi-Asl, Stanislav Abukhovich, Maame G Asante-Mensah, Andrzej Cichocki, Anh Huy Phan, Tohishisa Tanaka, and Ivan Oseledets. Randomized algorithms for computation of tucker decomposition and higher order svd (hosvd). *IEEE Access*, 9:28684–28706, 2021. **Scopus, Q1. First three authors equally contributed.**

   Author's contributions in the paper include: design, implementation, experiments, draft, and revisions.

3. Salman Ahmadi-Asl, Andrzej Cichocki, Anh Huy Phan, Maame G Asante-Mensah, Farid Mousavi, Ivan Oseledets, and Toshihisa Tanaka. Randomized algorithms for fast computation of low-rank tensor ring model. *Machine Learning: Science and Technology*, 2020. **Scopus, Q1.**

   Author's contributions in the paper include: implementation, experiments and draft.

4. Maame G Asante-Mensah and Andrzej Cichocki. Medical image de-noising using deep networks. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 315–319. IEEE, 2018. **Scopus, conference proceedings, h=25.**

   Author's contributions in the paper include: idea, design, implementation, theoretical analysis, experiments, draft, and revisions.

5. Salman Ahmadi-Asl, Maame G Asante-Mensah, Andrzej Cichocki, Anh Huy Phan, Ivan Oseledets, and Jun Wang. Fast cross tensor approximation for

image and video completion. *Signal Processing*, page 109121, 2023. **Scopus, Q1. First two authors equally contributed**.

Author's contributions in the paper include: theoretical analysis, implementation, experiments, and draft.

6. Maame G Asante-Mensah, Anh Huy Phan, Salman Ahmadi-Asl, Andrzej Cichocki, Image Reconstruction using Superpixel Clustering and Tensor Completion, https://arxiv.org/pdf/2305.09564.pdf. **Accepted: Signal Processing Journal, Elsevier. Scopus, Q1. First author.**

   Author's contributions in the paper include: design, implementation, theoretical analysis, experiments, draft, and revisions.

Thesis Outline

Significance: This thesis is devoted to the development of new methods and associated algorithms for removing or reducing noise and artifacts from signals (time series), images (represented by matrices and tensors), and videos (represented by 3D or 4D tensors), especially biomedical signals and neuroimages. Such data can be naturally represented by 3D/4D tensors (e.g., space-time-frequency) or even tensorized (i.e., vector matrices and 3D tensors are transformed to higher-order tensors). Such tensors can be decomposed into factor matrices and core tensors using tensor decomposition and tensor networks.

The thesis focused on data that is corrupted by large noise (outliers) and postulated that not all the data samples are corrupted by noise, but the majority of them are (typically up to about 95% of pixels can be corrupted by noise). In the thesis, it is also assumed that corrupted samples or pixels can be identified automatically (that is, their positions in time and/or space) by special software that was recently developed. The thesis solves these tasks using a matrix and tensor completion approach. The tensor completion helps to model and represent incomplete multidimensional data in the form of factor matrices and/or core tensors and reconstruct the missing elements via low-rank approximation with additional constraints imposed on factors like sparsity and smoothness. Although various tensor completion techniques have already been developed, to date there are still certain scenarios where the state-of-the arts algorithms are slow with high complexity or provide poor performance when the number of missing elements is large.

In other scenarios, the available algorithms to date are only able to recover certain types of missing data, such as randomly missing pixels. Motivated by the limitations of existing tensor completion techniques, the thesis developed new or alternative algorithms by exploiting and adopting different optimization strategies and constraints. The main advantage of our proposed methods is the ability to recover different kinds of missing data, such as structurally missing pixels e.g., data with whole rows or columns missing or missing blocks/patches. Also, images with scratched surfaces and images with randomly missing pixels can be recovered. The

algorithms are also computationally efficient compared to most current methods and are able to work on different types of tensor data.

The primary goal of the thesis is to develop efficient algorithms for tensor completion tasks that are capable of removing artifacts and noise from data, particularly MRI and EEG data. The proposed methods are comparable to state-of-the art algorithms and, in general, do not suffer from limitations that occur in existing methods.

Novelty: Three new methods based on matrix and tensor network decompositions are proposed in this work. The first method uses dictionary learning and sparse coding for updating the core tensors of a tensor ring decomposition, where noisy corrupted data is transformed into a higher-order tensor through block Hankelization. The problem is solved using the ADMM and gradient descent algorithms. The Hankelized high-order tensor is first decomposed into tensor ring cores. Then an over-complete discrete cosine dictionary is applied to each core tensor to learn the latent features and produce a sparse vector. The core tensors, together with the sparse codes, are then updated using the gradient descent algorithm until optimal accuracy is reached. The proposed method provides higher performance compared to other completion methods.

The second method proposed uses the matrix cross approximation methods and extends them to the tensor case. The thesis applied the Tucker decomposition model with the CUR sampling method to the fibers of the tensor. A smoothing technique is applied to the sampled incomplete fibers of each unfolding tensor before tensor completion is performed. This novel approach improves the performance of the completion results as compared to performing the completion without the smoothing strategy. Also, the use of tubes of tensor reduces the running time of the method, thereby providing a faster means of reconstructing huge amounts of incomplete data.

The third method proposed in this thesis performs the reconstruction of motion artifacts from MRI data using Hankelization and tensor tubal norm. A few samples of MRI data are sampled in the Fourier domain, and reconstruction is performed. The results are comparable to other methods used for reconstructing diffusion-weighted MRI data.

Acknowledgments

I would like to thank my supervisor, Prof. Andrzej Cichocki, for providing guidance and feedback on this project. I would also like to express my gratitude to Dr. Salman Ahmadi-Asl for the joint scientific collaboration and deep discussions about the thesis. I am thankful to Prof. Anh Huy Phan, Prof. Evgeny Burnaev, and Prof. Ivan Oseledets for their advice and feedback on this work. Lastly, I am grateful to my husband for his encouragement.

Contents

List of Figures

List of Tables

List of Abbreviations

ADMM Alternating Direction Method of Multipliers.

ALS Alternating Least Squares.

APG Accelerated Proximal Gradient.

BCI Brain Computer Interface.

CAD Computer Aided Diagnosis.

CNN Convolutional Neural Network.

CPD Canonical Polyadic Decomposition.

CT Computer Tomography.

CTA Cross Tensor Approximation.

CUR ColUmn Row Matrix Approximation.

DIP Deep Image Prior.

DNN Deep Neural Network.

DWI Diffusion Weighted Imaging.

EEG Electroencephalogram.

HOOI Higher Order Orthogonal Iteration.

HOSVD Higher Order SVD.

ICA Independent Component Analysis.

LRTC Low Rank Tensor Completion.

MDT Multi-way Delay Embedding.

MMES Manifold Modelling in Embedded Space.

MRI Magnetic Resonance Imagine.

NRMSE Normalised Root Mean Square Error.

ODCT Over-complete Discrete Cosine Transform.

PCA Principle Component Analysis.

PET Positron Emission Tomography.

PSNR Peak Signal-to-Noise Ratio.

RMSE Root Mean Square Error.

RSE Relative Squared Error.

SSIM Stuctured Similarity index Measurements.

SVD Singular Value Decomposition.

SVT Singular Value Thresholding.

t-SVD tensor-Singular Value Decomposition.

TC Tensor Completion.

TN Tensor Network.

TNN Tensor Nuclear Norm.

TR Tensor Ring.

TRLRF Tensor Ring Low Rank Factors.

TT Tensor Train.

Chapter 1

Introduction and Problem Statement

As healthcare and medical sciences advance rapidly, there is an increasing dependence on medical imaging due to its clarity that illustrates current and developing conditions in a patient's body. However, there are several challenges with the data acquisition processes on biomedical data, such as the movement of patients before the image capture, technical difficulties (poor calibration, environmental factors, etc.) of equipment, and image corruption during the transfer stage. These have given rise to the implementation of algorithms to solve/correct these problems. Incomplete or corrupted data can lead to miscalculations and misdiagnoses of medical conditions, which subsequently have a pronounced health, financial and reputational impact on all stakeholders. Biomedical data denoising is a classical problem in computer vision and has been studied extensively. Many researchers have taken up the task of finding more effective methods to eliminate these artifacts in medical image data due to the significant advantages it brings to medical care for correct diagnosis and treatment of patients. Most papers in the medical field address the reconstruction of images using transform coding-based methods such as wavelet transforms, ICA, PCA, Machine learning, etc [14, 15, 16, 17, 18, 19, 20, 21, 22]. However, in recent times, low-rank matrix approximations have been employed to solve these problems in biomedical signal data, particularly MRI and EEG data. Consequently, the idea and use of tensors have surpassed theoretical performance limits and opened up new avenues for real-world applications [23]. Tensors accomplish this by taking advantage of the underlying latent data structures [6, 24].

1.1 Thesis Statement

This research focuses on the completion of images and signals (especially biomedical signals and neuroimages) using tensor completion techniques. A pre-processing step of tensorizing the data into higher order tensors using different Hankel folding methods is adopted. This pre-processing step aids in improving the performance of the algorithms. To improve the performance of the results, our proposed algorithms make use of sparsity and smoothness in the data.

1.2 Layout of Thesis

After the brief introduction, Chapter 1 continues with preliminaries and notations, discussions on the problem statement, an introduction to the data used and an overview of some related works. The thesis continues with discussions on tensor decomposition, in Chapter 2. Chapter 3 discusses some proposed Hankelization procedures used for tensor completion problems. Consequently, the proposed tensor completion methods are discussed and evaluated in Chapters 4, 5 and 6. The last chapter, Chapter 7 presents a conclusion of the work and future research directions.

1.3 Preliminaries and notations

Notations adopted in the work by [12] are used all through this research. The order of a tensor is defined as number of its' dimensions, in some literature the order is also referred to as modes. Scalars are denoted by standard lowercase letters, e.g. x. Vectors or 1st-order tensors are denoted by boldface lower case letters, e.g. $\mathbf{x}$. A matrix or second order tensor is represented by boldface capital letter, e.g. $\mathbf{X}$ and a higher order tensor is also denoted by bold underlined capital letter, e.g. $\underline{\mathbf{X}}$. The $(i_1, i_2, \ldots, i_N)$th element of an Nth-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by $x_{i_1,i_2,\ldots,i_N} = \underline{\mathbf{X}}(i_1, i_2, \ldots, i_N)$. The multi-index is defined as $\overline{i_1 i_2 \ldots i_N} = i_N + (i_{N-1} - 1)\, I_N + \cdots + (i_1 - 1)\, I_2 I_3 \cdots I_N$.

A typical n-mode matricization [6] also known as mode-n unfolding of a tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is given by $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times I_1 \cdots I_{n-1} I_{n+1} \cdots I_N}$. Refer to figure 1-1 for

illustration on tensor unfolding. Transforming a vector, matrix, or a low-order to a higher-order tensor is known as tensorization or matrix folding. Tensorization is the inverse operation for matricization. Slices are sub-tensors generated by fixing all indices except two, and they are thus matrices. For example, for a tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, the slices $\underline{\mathbf{X}}(:,:,i)$, $i = 1, 2, \ldots, I_3$, are called frontal slices and are denoted as $\mathbf{X}^{(i)}$. Refer to figure 1-2 for illustration. A fiber is denoted as $\mathbf{X}(i,:,j), \mathbf{X}(:,j,k)$ and $\mathbf{X}(i,j,:)$. It is defined by fixing all but one index. Fibers in third order tensors are called rows, columns and tubes respectively. The inner product of two tensors $\underline{\mathbf{X}}$, $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as $\langle \underline{\mathbf{X}}, \underline{\mathbf{Y}} \rangle = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} a_{i_1,\ldots,i_N} b_{i_1,\ldots,i_N}$. The trace of $\underline{\mathbf{X}}$ is defined as $tr(\underline{\mathbf{X}}) = \sum_{i=1}^{I_3} tr(\mathbf{X}^{(i)})$ where $tr(.)$ is the matrix trace. The induced Frobenius norm of a tensor is denoted by $\|\underline{\mathbf{X}}\|_F = \sqrt{\langle \underline{\mathbf{X}}, \underline{\mathbf{X}} \rangle}$. Refer to Table 1.1 for common tensor operations and symbols.

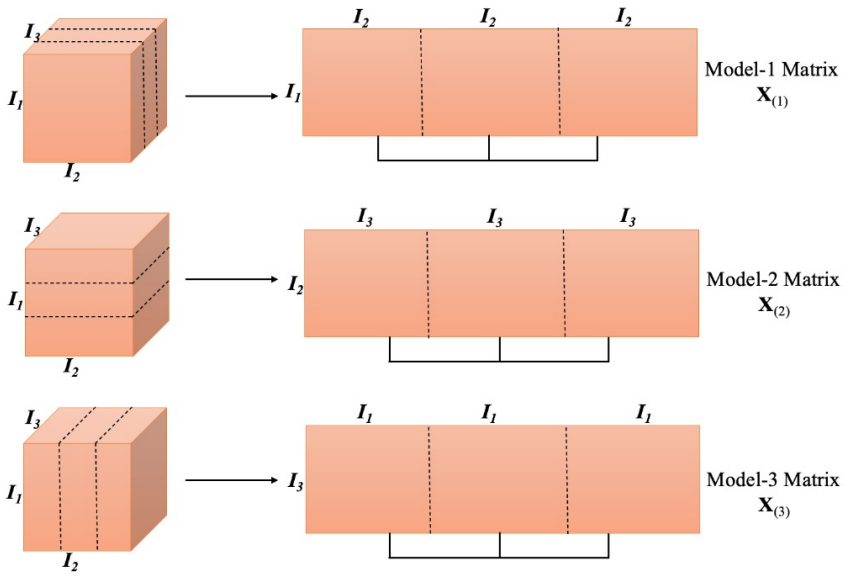


Figure 1-1: Mode-n unfolding of a tensor

Definition 1. (t-product) The t-product of two tensors $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and $\underline{\mathbf{Y}} \in \mathbb{R}^{I_2 \times I_4 \times I_3}$, is given by $\underline{\mathbf{C}} \in \mathbb{R}^{I_1 \times I_4 \times I_3}$. This is defined as

$$\underline{\mathbf{C}} = \underline{\mathbf{X}} * \underline{\mathbf{Y}} = \text{fold}\left(\text{circ}\left(\underline{\mathbf{X}}\right) . \text{unfold}\left(\underline{\mathbf{Y}}\right)\right), \tag{1.1}$$

Table 1.1: Matrix and tensor symbols and notations.

Notation/symbol	Meaning
$x, \mathbf{x}, \mathbf{X}$	scalar, vector and matrix
$\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$	an N^{th} order tensor of size $I_1 \times I_2 \times \cdots \times I_N$
$x_{i_1,i_2,\ldots,i_N} = \underline{\mathbf{X}}(i_1, i_2, \ldots, i_N)$	$(i_1, i_2, \ldots, i_N)^{th}$ entry of tensor $\underline{\mathbf{X}}$
$\mathbf{X}^T, \mathbf{X}^{-1}, \mathbf{X}^\dagger$	transpose, inverse and Moore–Penrose pseudo-inverse of a matrix $\mathbf{X}$
$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_R]$	matrix with R column vectors $a_r \in \mathbb{R}^I$, with entries a_{ir}
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{A}^{(n)}, \mathbf{B}^{(n)}, \mathbf{U}^{(n)}$	component (factor) matrices
$\underline{\mathbf{G}}, \underline{\mathbf{G}}^{(n)}, \underline{\mathbf{X}}^{(n)}$	core tensors
$\underline{\mathbf{X}}_{(n)} \in \mathbb{R}^{I_n \times I_1 \cdots I_{n-1} I_{n+1} \cdots I_N}$	mode-n matricization of $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$
$R, (R_1, \ldots, R_N)$	tensor rank R and its corresponding multi-linear rank
$\underline{\mathbf{X}}_{<n>} \in \mathbb{R}^{I_1 I_2 \cdots I_n \times I_{n+1} \cdots I_N}$	mode-$(1, \ldots, n)$ matricization of $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$
$\underline{\mathbf{X}}(:, i_2, i_3, \ldots, i_N) \in \mathbb{R}^{I_1}$	mode-1 fiber of a tensor $\underline{\mathbf{X}}$ obtained by fixing all indices but one (a vector)
$\underline{\mathbf{X}}(:, :, i_3, \ldots, i_N) \in \mathbb{R}^{I_1 \times I_2}$	slice (matrix) of a tensor $\underline{\mathbf{X}}$ obtained by fixing all indices but two
$\underline{\mathbf{X}}(:, :, :, i_4, \ldots, i_N)$	sub-tensor of $\underline{\mathbf{X}}$, obtained by fixing several indices
$< \underline{\mathbf{X}}, \underline{\mathbf{Y}} >$	inner product of two tensor $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$ with same size
$\underline{\mathbf{X}} \times_n \mathbf{A}$	mode-n product of $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and matrix $\mathbf{A} \in \mathbb{R}^{J \times I_n}$
$\circ$	outer product
$\odot$, $\otimes$	Khatri–Rao product, Kronecker product
$tr(.)$	trace of a square matrix
$diag(.)$	diagonal matrix

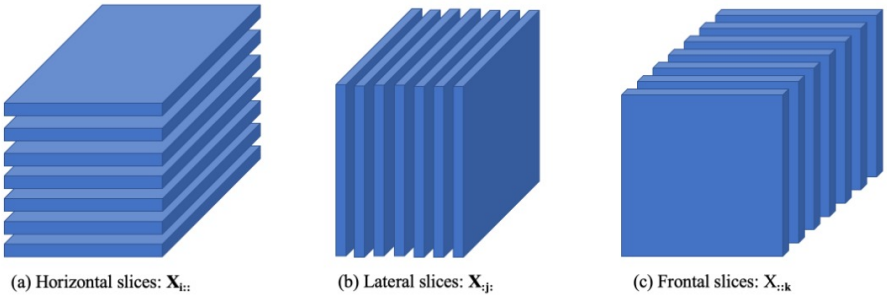


Figure 1-2: Slices from a 3rd order tensor [6]

where

$$\operatorname{circ}(\underline{\mathbf{X}}) = \begin{bmatrix} \underline{\mathbf{X}}(:,:,1) & \underline{\mathbf{X}}(:,:,I_3) & \cdots & \underline{\mathbf{X}}(:,:,2) \\ \underline{\mathbf{X}}(:,:,2) & \underline{\mathbf{X}}(:,:,1) & \cdots & \underline{\mathbf{X}}(:,:,3) \\ \vdots & \vdots & \ddots & \vdots \\ \underline{\mathbf{X}}(:,:,I_3) & \underline{\mathbf{X}}(:,:,I_3-1) & \cdots & \underline{\mathbf{X}}(:,:,1) \end{bmatrix},$$

and

$$\operatorname{unfold}(\underline{\mathbf{Y}}) = \begin{bmatrix} \underline{\mathbf{Y}}(:,:,1) \\ \underline{\mathbf{Y}}(:,:,2) \\ \vdots \\ \underline{\mathbf{Y}}(:,:,I_3) \end{bmatrix}, \quad \underline{\mathbf{Y}} = \operatorname{fold}(\operatorname{unfold}(\underline{\mathbf{Y}})).$$

It can be seen that the t-product operation (1.1) is equivalent to the circular convolution operator, and can therefore be easily computed through the Fast Fourier Transform (FFT). To be precise, all tubes from the two tensors $\underline{\mathbf{X}}$, $\underline{\mathbf{Y}}$ are transformed into the frequency domain, then the frontal slices of the spectral tensors are multiplied. we then apply the Inverse Fast Fourier Transform (IFFT) to all the tubes in the last tensor. The t-product can also be written in the Fourier domain as follows:

$$\underline{\mathbf{C}} = \underline{\mathbf{X}} * \underline{\mathbf{Y}} \equiv \widehat{\mathbf{C}} = \widehat{\mathbf{X}}\widehat{\mathbf{Y}}, \tag{1.2}$$

where $\widehat{\mathbf{X}}, \widehat{\mathbf{Y}}$ and $\widehat{\mathbf{C}}$ are block diagonal matrices defined as follows:

$$\widehat{\mathbf{X}} = bdiag(\underline{\widehat{\mathbf{X}}}) = \begin{bmatrix} \widehat{\mathbf{X}}^{(1)} & & 0 \\ & \widehat{\mathbf{X}}^{(2)} & \\ & & \ddots \\ & 0 & & \widehat{\mathbf{X}}^{(I_3)} \end{bmatrix},$$

where $\widehat{\mathbf{X}}^{(1)}$ is the a matrix computed by applying the fast Fourier transform. The operator $bdiag(.)$ maps the tensor $\underline{\widehat{\mathbf{X}}}$ to the block diagonal matrix $\widehat{\mathbf{X}}$. The procedure for t-product in the Fourier domain is summarized in Algorithm 1. The Matlab function $\text{fft}\left(\underline{\mathbf{X}}, N, DIM\right)$ is used in computing fast Fourier transform across the dimension DIM. Where DIM represents the dimension to perform the transform and N is the N-point fft where N can be fixed as the length of the data. The inverse Fourier transform is obtained using $\text{ifft}\left(\underline{\widehat{\mathbf{X}}}, N, DIM\right)$. The command $\text{fft}\left(\underline{\mathbf{X}}, [], 3\right)$ applies the fft operation across the frontal slice of $\underline{\mathbf{X}}$.

Algorithm 1: t-product in the Fourier domain [25]

Input : Two data tensors $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, $\underline{\mathbf{Y}} \in \mathbb{R}^{I_2 \times I_4 \times I_3}$
Output: t-product $\underline{\mathbf{C}} = \underline{\mathbf{X}} * \underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_4 \times I_3}$

$\underline{\widehat{\mathbf{X}}} = \text{fft}\left(\underline{\mathbf{X}}, [], 3\right)$
$\underline{\widehat{\mathbf{Y}}} = \text{fft}\left(\underline{\mathbf{Y}}, [], 3\right)$
for $i = 1, 2, \ldots, I_3$ **do**
  $\underline{\widehat{\mathbf{C}}}\left(:, :, i\right) = \underline{\widehat{\mathbf{X}}}\left(:, :, i\right) \underline{\widehat{\mathbf{Y}}}\left(:, :, i\right)$
end
$\underline{\mathbf{C}} = \text{ifft}\left(\underline{\widehat{\mathbf{C}}}, [], 3\right)$

Definition 2. (Transpose) The conjugate transpose of tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is denoted by $\underline{\mathbf{X}}^T \in \mathbb{R}^{I_2 \times I_1 \times I_3}$. It is obtained by applying transpose to all the frontal slices and then reversing the order of the transposed frontal slices from the second through to the last frontal slice.

Definition 3. (Identity tensor) An identity tensor $\underline{\mathbf{I}} \in \mathbb{R}^{I_1 \times I_1 \times I_3}$ is a tensor with first frontal slice being an identity matrix of size $I_1 \times I_1$, and all other frontal slices being equal to zero.

Definition 4. (Orthogonal tensor) A tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_1 \times I_3}$ is orthogonal if $\underline{\mathbf{X}}^T * \underline{\mathbf{X}} = \underline{\mathbf{X}} * \underline{\mathbf{X}}^T = \underline{\mathbf{I}}$ is satisfied.

Definition 5. (f-diagonal tensor) An f-diagonal tensor is a tensor with all of its frontal slices being diagonal.

Definition 6. (t-SVD) A tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, can be decomposed as

$$\underline{\mathbf{X}} = \underline{\mathbf{U}} * \underline{\mathbf{S}} * \underline{\mathbf{V}}^T,$$

where $\underline{\mathbf{U}} \in \mathbb{R}^{I_1 \times I_1 \times I_3}$, $\underline{\mathbf{V}} \in \mathbb{R}^{I_2 \times I_2 \times I_3}$ are orthogonal tensors, and tensor $\underline{\mathbf{S}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is f-diagonal.

Algorithm 2: t-SVD [26]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and target tubal rank R
Output: $\underline{\mathbf{U}}_R \in \mathbb{R}^{I_1 \times R \times I_3}$, $\underline{\mathbf{S}}_R \in \mathbb{R}^{R \times R \times I_3}$, $\underline{\mathbf{V}}_R \in \mathbb{R}^{I_2 \times R \times I_3}$

$\widehat{\underline{\mathbf{X}}} = \text{fft}\left(\underline{\mathbf{X}}, [], 3\right)$
for $i = 1, 2, \ldots, I_3$ **do**
  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{svd}\left(\widehat{\underline{\mathbf{X}}}(:, :, i), R\right)$
  $\widehat{\underline{\mathbf{U}}}\,(:, :, i) = \mathbf{U}$
  $\widehat{\underline{\mathbf{S}}}\,(:, :, i) = \mathbf{S}$
  $\widehat{\underline{\mathbf{V}}}\,(:, :, i) = \mathbf{V}$
end
$\underline{\mathbf{U}} = \text{ifft}\left(\widehat{\underline{\mathbf{U}}}, [], 3\right)$, $\underline{\mathbf{S}} = \text{ifft}\left(\widehat{\underline{\mathbf{S}}}, [], 3\right)$, $\underline{\mathbf{V}} = \text{ifft}\left(\widehat{\underline{\mathbf{V}}}, [], 3\right)$

The t-SVD can be obtained using the SVD of frontal slices of the original data tensor in the Fourier domain. The algorithm for computing the t-SVD for tensors is outlined in Algorithm 2. So, for the t-SVD of $\underline{\mathbf{X}}$, we have:

$$\widehat{\mathbf{X}}^{(i)} = \widehat{\mathbf{U}}^{(i)} * \widehat{\mathbf{S}}^{(i)} * (\widehat{\mathbf{V}}^{(i)})^T, \;\; i = 1, 2, \ldots, I_3\,, \tag{1.3}$$

where $\widehat{\mathbf{X}}^{(i)}$ is the i-th frontal slice of the tensor $\underline{\mathbf{X}}$ in the Fourier domain, i.e., $\widehat{\mathbf{X}}^{(i)} = \widehat{\mathbf{X}}(:, :, i)$.

Definition 7. (tensor tubal rank and tensor nuclear norm) [27] The tensor tubal rank of a third order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is defined as the maximum rank among

all frontal slices of an f-diagonal tensor $\underline{\mathbf{S}}$, i.e., max $rank(\underline{\mathbf{S}}^{(i)})$. It is the number of non-zero tubes of $\underline{\mathbf{S}}$, where $\underline{\mathbf{S}}$ is from the t-SVD of $\underline{\mathbf{X}} = \underline{\mathbf{U}} * \underline{\mathbf{S}} * \underline{\mathbf{V}}^T$

Additionally, the tensor nuclear norm $\|\underline{\mathbf{X}}\|_*$ is defined as the sum of the singular values in all frontal slices of $\underline{\mathbf{S}}$, i.e.,

$$\|\underline{\mathbf{X}}\|_* = tr(\underline{\mathbf{S}}) = \sum_{i=1}^{I_3} tr(\mathbf{S}^{(i)}), \tag{1.4}$$

where $\underline{\mathbf{S}}^{(i)}$ are defined in (1.3).

It is shown in [27] that the trace of tensor product $(\underline{\mathbf{X}} * \underline{\mathbf{Y}})$ equals to the trace of the product of $\widehat{\mathbf{X}}^{(1)}$ and $\widehat{\mathbf{Y}}^{(1)}$,that is

$$tr(\underline{\mathbf{X}} * \underline{\mathbf{Y}}) = tr(\widehat{\mathbf{X}}^{(1)}\widehat{\mathbf{Y}}^{(1)}). \tag{1.5}$$

Therefore, we can conclude as shown in [27] that the tensor nuclear norm defined in (1.4) can be simplified as

$$\|\underline{\mathbf{X}}\|_* = tr(\underline{\mathbf{S}}) = tr(\widehat{\mathbf{S}}^{(1)}) = \left\|(\widehat{\mathbf{X}}^{(1)})\right\|_*. \tag{1.6}$$

The work by [28] proposed the tubal nuclear norm minimization approach based on t-SVD and defined as the sum of nuclear norms of all frontal slices in the Fourier domain and proved to be convex envelope to the tensor tubal rank.

Definition 8. (Multi-rank) A multi-rank is a vector $[s_1, s_2, s_3, \ldots, s_n]$ where s_i is the rank of $\underline{\mathbf{X}}(:,:,i)$.

Definition 9. (Tensor singular value thresholding) The singular value thresholding (SVT) [27] operator $\underline{\mathbf{D}}_\beta(.)$ is performed on each frontal slice of the f-diagonal tensor $\widehat{\underline{\mathbf{S}}}$. That is,

$$\underline{\mathbf{D}}_\beta(\underline{\mathbf{X}}) = \underline{\mathbf{U}} * \underline{\mathbf{D}}_\beta(\underline{\mathbf{S}}) * \underline{\mathbf{V}}^T, \tag{1.7}$$

where $\underline{\mathbf{D}}_\beta(\underline{\mathbf{S}})$ is the inverse FFT of $\underline{\mathbf{D}}_\beta(\widehat{\underline{\mathbf{S}}})$ and $\underline{\mathbf{D}}_\beta(\widehat{\underline{\mathbf{S}}}^{(i)}) = diag(max\{\sigma_t - \beta, 0\}_{1\leq t\leq R})$, $i = 1, \ldots, I_3$, $\beta > 0$ and R is the rank of $\widehat{\mathbf{S}}^{(i)}$.

1.4 Problem Statement

With the increasing number of information processing tasks, experts seek to recover signals of interest from incomplete or highly corrupted data. The problem of corruption commonly arises in computational imaging, computer vision, and other fields when either large parts of the data get corrupted or could not be captured due to some constraints. In this regard, compressive sensing, machine learning, neural networks and other related areas have raised and addressed fundamental questions concerning the amount and the content of data that are sufficient for signal recovery [29, 30, 31, 32]. Despite a lot of recent progress, there are still certain challenges with visual representation of the resultant data. This work focuses specifically on the tensor completion approach for solving the problem of incomplete data. Tensor factorization/decomposition has evolved into a powerful tool for multidimensional data representation and analysis. Low rank tensor factorization is seen as a higher order generalization of low-rank matrix factorization, both of which have been used for image and video representation and reconstruction from compressive measurements. Tensor completion uses the techniques of tensor factorization in estimating missing pixels from incomplete or noisy data with the help other optimization methods and constraints to achieve a complete data useful for specific data analysis. Most tensor completion methods, exploit some *a priori* information in order to impose specific constraints such as smoothness, nonnegativity and sparsity. Some real-world data are often naturally non-negative, examples are images or medical data. In such cases, non-negativity constraints are added to the method in order to increase interpretability and uniqueness of tensor decomposition solutions. Moreover, in addition to noise, many datasets contain instances of data that are highly inconsistent, referred to as outliers in data distribution. The robust PCA techniques [33, 34] can be used to remove the effect of outliers and make the decomposition method more robust by separating the approximation tensor into low-rank tensor plus very sparse tensors to capture the different outlier patterns. The tensor completion problem can be solved by the minimization of a suitable cost or objective function. The minimization can be subject to some constraints and regularization terms. The task

of tensor completion is solved by completing an N^{th}-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ from its known entries given by a specific index set. In general, the problem is solved by minimizing the following optimization problem [35, 36]

$$\begin{aligned} &\min_{\underline{\mathbf{X}}} f(\underline{\mathbf{X}}), \\ &s.t.\, \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}), \end{aligned} \tag{1.8}$$

where $\underline{\mathbf{X}}$ is the estimated recovered tensor with full entries and $\underline{\mathbf{Y}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the observed incomplete data with observation index tensor $\underline{\mathbf{\Omega}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, defined as:

$$\underline{\mathbf{\Omega}}(i_1, i_2, \ldots, i_N) = \begin{cases} 1 & \text{if } x_{i_1,i_2,\ldots,i_N} \text{ is known,} \\ 0 & \text{if } x_{i_1,i_2,\ldots,i_N} \text{ is unknown,} \end{cases}$$

then the complement becomes:

$$\underline{\mathbf{\Omega}}^{\perp}(i_1, i_2, \ldots, i_N) = \begin{cases} 0 & \text{if } x_{i_1,i_2,\ldots,i_N} \text{ is known,} \\ 1 & \text{if } x_{i_1,i_2,\ldots,i_N} \text{ is unknown.} \end{cases}$$

The operator $\mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}})$ projects the data tensor $\underline{\mathbf{X}}$ onto the observation index tensor $\underline{\mathbf{\Omega}}$ and is defined as

$$\mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \begin{cases} x_{i_1,i_2,\ldots,i_N} & (i_1, i_2, \ldots, i_N) \in \underline{\mathbf{\Omega}}, \\ 0 & \text{Otherwise.} \end{cases}$$

The function $f(.)$ is a scalar cost function, with (prior) structural assumptions,

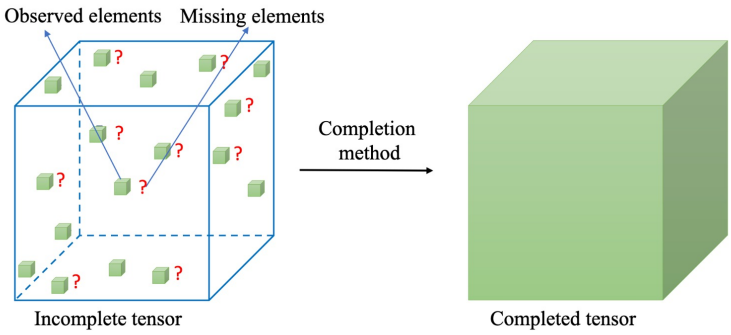


Figure 1-3: An illustration of tensor completion [7]

and in general non-convex and non-differentiable. For noisy and insistent tensor data, it is often more convenient to consider "inexact" tensor completion, which can be formulated, in general, as the following optimization problem (for detailed information see [35]):

$$\min_{\underline{\mathbf{X}}} f(\underline{\mathbf{X}}) + \tfrac{\lambda}{2} \left\| \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}} - \underline{\mathbf{Y}}) \right\|_F^2, \tag{1.9}$$

where $\lambda > 0$ is penalty parameter which is difficult to determine. In general, the task of tensor completion can be termed as an incorrectly defined problem because it can have multiple solutions. For example, a simple trivial solution can be obtained in a such way that all values of the recovered data tensor $\underline{\mathbf{X}}$ equal to those in $\underline{\mathbf{Y}}$, in nonzero positions of the index tensor Ω and in zero positions are generated as zero-mean random noise. To overcome this problem, we need some prior assumptions. The key and very popular assumption is the low-rank tensor assumption. In general, we can formulate the following minimum tensor rank optimization problem:

$$\begin{aligned} &\min_{\underline{\mathbf{X}}} \quad rank\,(\underline{\mathbf{X}}), \\ &s.t. \;\; \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}). \end{aligned} \tag{1.10}$$

Here, the rank of the tensor could have different definitions and forms depending on the type of decomposition being used. Also, determining the rank of a tensor is an NP hard problem [37], and various alternatives for tensor rank have been proposed in the open literature [38, 39, 30, 13]. In [38], a weighted summation of nuclear norms of mode-n matricized versions of tensor, called the Tucker norm, was used as an alternative for tensor rank. The nuclear norm of canonical matricized versions of a tensor, called tensor train rank, was used as an alternative to tensor rank in [39]. Using tensor train rank instead of the Tucker rank, results in a more balanced matrices and consequently better completion algorithm. The work by [30], exploited n-rank of data tensor for tensor completion. In addition, the rank of a tensor can be replaced by the rank of its core tensors derived from the decomposition. Minimizing the summation of nuclear norms of matricized versions of core tensors was proposed in [13]. Another alternative called latent low rank has been also used in [40]. The

Low Rank Tensor Completion (LRTC) model for the multi-linear (Tucker) rank can be formulated as follows:

$$\begin{aligned} \min_{\mathbf{X}_{(n)}} \quad & \sum_{n=1}^{N} \alpha_n rank\left(\mathbf{X}_{(n)}\right), \\ s.t. \quad & \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}), \end{aligned} \tag{1.11}$$

where α_n are nonnegative weights for rank $(\underline{\mathbf{X}}_{(n)})$ obtained by matricization of the data tensor along one single mode. For easier optimization, the rank minimization is usually relaxed to the following nuclear-norm minimization :

$$\begin{aligned} \min_{\{\underline{\mathbf{X}}\}} \quad & \sum_{n=1}^{N} \alpha_n \left\|\mathbf{X}_{(n)}\right\|_*, \\ s.t. \quad & \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}), \end{aligned} \tag{1.12}$$

where $\|.\|_*$ denotes the nuclear norm regularization in the form of a sum of the singular values of the matrix. That is:

$$\|\mathbf{X}\|_* = \sum_{i=1}^{min(I_1,I_2)} \sigma_i.$$

For the TT ranks with more balanced unfolded matrices, the LRTC model is formulated as

$$\begin{aligned} \min_{\{\underline{\mathbf{X}}\}} \quad & \sum_{n=1}^{N} \alpha_n \|\mathbf{X}_{<n>}\|_*, \\ s.t. \quad & \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}). \end{aligned} \tag{1.13}$$

Despite the fact that these unfolding matrices cannot be optimized independently due to their multi-linear correlations, this convex relaxation allows us to solve the completion problem without pre-defining the tensor rank, which makes it more tractable in practice. In the case of Gaussian noise, an equivalent problem could be written as

$$\arg\min_{\underline{\mathbf{X}}} \ \frac{\lambda}{2} \left\|\mathbf{P}_{\underline{\mathbf{\Omega}}}\left(\underline{\mathbf{X}} - \underline{\mathbf{Y}}\right)\right\|_F^2 + \sum_{n=1}^{N} \alpha_n \|\mathbf{X}_{(n)}\|_* , \tag{1.14}$$

where the operator $\mathbf{P}_{\underline{\mathbf{\Omega}}}(.)$ projects the observation index tensor $\underline{\mathbf{\Omega}}$ unto the estimated tensor.

A more popular approach for solving the above problem is to apply splitting

method [30, 41, 38, 42], example, Alternating Direction Method of Multipliers (ADMM). Based on ADMM method, the completion problem defined in Equation 1.14 could be reformulated by introducing auxiliary matrices $\mathbf{Z}_{(n)}, n = 1, ..., N$ as

$$\begin{aligned} &\arg\min_{\underline{\mathbf{X}}} \ \tfrac{\lambda}{2} \left\| \mathbf{P}_{\underline{\mathbf{\Omega}}} \left(\underline{\mathbf{X}} - \underline{\mathbf{Y}} \right) \right\|_F^2 + \sum_{n=1}^{N} \alpha_n \|\mathbf{Z}_{(n)}\|_*, \\ &s.t \ \mathbf{Z}_{(n)} = \mathbf{X}_{(n)}, n = 1, \ldots, N. \end{aligned} \tag{1.15}$$

An augmented Lagrangian objective function from the above equation could be derived as

$$\mathcal{L}_n = \tfrac{\lambda}{2} \| \mathbf{P}_{\underline{\mathbf{\Omega}}} \left(\underline{\mathbf{X}} - \mathbf{Y} \right) \|_F^2 + \sum_{n=1}^{n} (\alpha_n \left\| \mathbf{Z}_n \right\| + \langle \mathbf{Y}_n, \mathbf{Z}_n - \mathbf{X}_n \rangle + \tfrac{\mu}{2} \| \mathbf{Z}_n - \mathbf{X}_{(n)} \|_F^2), \tag{1.16}$$

where $\mathbf{Y}_{(n)}$ are the Lagrange multipliers, $\mu > 0$ is a penalty parameter. Using the ADMM, the update scheme for $\mathbf{Z}_n$ and $\mathbf{Y}_n$ could be iteratively updated as follows

$$\begin{aligned} &\mathbf{Z}_n \leftarrow \mathbf{D}_\beta \big(\mathbf{X}_n^{k-1} + \tfrac{\mathbf{Y}_n}{\lambda} \big), \\ &\mathbf{X}_n \leftarrow \big(\mathbf{Z}_n - \tfrac{1}{\lambda} \mathbf{Y}^{k-1} \big)_{\Omega^\perp}, \\ &\mathbf{Y}_n \leftarrow \mathbf{Y}_n + \mu \big(\mathbf{Z_n} - \mathbf{X}_n \big). \end{aligned} \tag{1.17}$$

These solutions run until a stopping criterion is reached. $\beta > 0$ is constant and $\mathbf{D}_\beta(.)$ is the matrix singular value thresholding operation achieved by computing the singular value decomposition of a matrix $\mathbf{X}$:

$$\mathbf{D}_\beta(\mathbf{X}) = \mathbf{U} \, \mathbf{D}_\beta(\mathbf{S}) \, \mathbf{V}^T,$$

where $\mathbf{U}$ represent the right singular vectors and and $\mathbf{V}$ represent the left singular vectors. A popular soft thresholding operator as follows

$$\mathbf{D}_\beta(x) = sgn(x) max(|x| - \beta, 0).$$

Various optimization methods based on minimizing the multilinear (Tucker) norm were proposed in [38], including Simple Low Rank Tensor Completion (SiLRTC), which employs relaxation between different matrices resulted from the matri-

cization of different modes of the tensor. Another algorithm is called Fast Low-Rank Tensor Completion (FaLRTC), which uses a smoothing scheme for smoothing non-smooth cost function. Finally, the third algorithm, High accuracy Low-Rank Tensor Completion (HaLRTC), uses the ADMM described above for minimizing the nuclear norms of unfolding matrices. Recently, several algorithms based on minimizing a tensor train rank have been proposed in [39]. The first algorithm, called SiLRTC-TT, is similar to SiLRTC whereby tensor train rank has been used instead of Tucker rank. Another approach is based on parallel matrix factorization of the canonical matricized version of the tensor, referred to as the TMac-TT, has been also proposed in [39]. The minimization of tensor train rank, in addition to the sparsity assumption of mode-n matricized version of the tensor, has been used in [43]. In this thesis, it is assumed that the missing elements may be distributed both randomly and structurally (i.e., the whole blocks or tubes can be missing). In tensor decomposition based algorithms, an incomplete tensor is decomposed to its latent variables and then the reconstructed tensor and consequently the missing elements are estimated using these latent variables [29, 44, 45, 13, 46, 47, 48]. In these approaches, it is typically assumed that the ranks of latent variables (core tensors and factor matrices) are known in advance. A large variety of tensor completion algorithms based on different tensor decomposition models have been proposed. Particularly, in [29], an approach based on CANDECOMP/PARAFAC (CP) decomposition, called CP Weighted OPTimization (CP-WOPT) was proposed which has been recently extended for TT and TR models. A very efficient Smoothed PARAFAC Decomposition (PD) method called Smooth PARAFAC tensor Completion (SPC), was also proposed for CPD tensor completion in [49], where the ranks of latent variables can be estimated during the completion procedure. In [47], the Bayesian Tucker decomposition model was used for tensor completion with automatic estimation of multilinear rank. On the other hand, tensor decomposition based on tensor ring and tensor train are promising approaches for large scale and high-order tensor completion [46, 43, 32, 50, 51, 52]. The main motivation for using tensor train and tensor ring decomposition rather than other decomposition approaches lies with the fact that determining CP rank of a tensor is NP-hard. In addition, Tucker decomposi-

tion suffers from issues related to the curse of dimensionality. In contrast to CP and Tucker decompositions, the rank of tensor train and tensor ring can be determined relatively easily through truncated SVD. In addition, the storage costs of tensor ring and tensor train decompositions change linearly with the tensor dimension [50]. Recently, [43, 32, 13, 40] have developed several new algorithms, including Tensor Train Weighted OPTimization (TT-WOPT) and Tensor Train Stochastic Gradient Descent (TT-STD) for tensor completion based on tensor train decomposition. Furthermore, the approach in [50] applied tensor train decomposition accompanied by system identification techniques for elegant and efficient tensor completion. In this approach, the completed tensor was assumed to be the system to be identified. Several variations of Alternating Least Squares (ALS) algorithms for finding low-rank approximation of incomplete tensor with TT and TR format were proposed in [45, 32, 13]. The first algorithm for tensor completion using tensor ring decomposition, called Tensor Ring completion by Alternating Least Squares (TR-ALS) was proposed in [46]. In [13] an efficient algorithm called Tensor Ring Low-Rank Factors (TR-LRF) was developed. In this algorithm, the missing elements were estimated by applying rank minimization to the matrices obtained from the matricization of the core tensors. The Tensor Ring Weighted OPTimization (TR-WOPT) also has been developed in [32]. In this algorithm, the core tensors are updated using gradient based optimization in such a way that the observed elements of the estimated and incomplete tensors are the same. In tensor completion there are some basic assumptions that are taken into consideration to make the completion task work. One of the most important assumptions is the sampling assumption:

- Tensor completion is typically based on the random sampling assumption, which states that the partially observed entries are drawn at random from the original tensor. Bernoulli sampling and independent sampling with replacement are two random sampling approaches for tensors. Several other sampling assumptions, such as Gaussian measurements, Fourier measurements, and Adaptive sampling, are also used for the convenience of theoretical demonstration or the practicability in real-world applications.

1.5 Challenges with MRI and EEG data

Our human bodies communicate health information that indicates the state of our organs as well as our overall health. This data is collected by devices that measure various information such as muscle movement, eye blinks, brain activity, heart rate, blood pressure, and many others [53]. Advancement in medical imaging modalities such as magnetic resonance imaging (MRI), computed tomography (CT), and positron emission tomography (PET) and ultrasound allows scientist and radiologists to visualize the functions and underlying structure of the human organs. One main characteristic of biomedical data is that they are all time series comprising of both spatial and temporal dimensions. Signal and image processing tasks such as segmentation of organ structures, classification, image processing or analysis and Computer Aided Diagnosis (CAD) make use of such data.

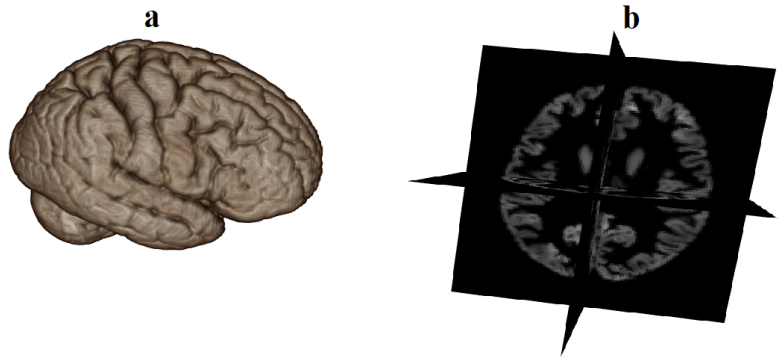


Figure 1-4: a. MRI brain image without a skull, b. 3D volumetric MRI brain data

MRI is a medical imaging technology that uses radio waves and a magnetic field to create detailed images of organs and tissues [54, 55]. It is a non-invasive technique, which provides good soft tissue contrast and is widely available in clinics. MRI has proven to be highly effective in diagnosing a number of conditions by showing the difference between normal and diseased soft tissues of the body. The MRI scan is more comfortable than any other scans for diagnosis [56, 57]. However, MRI just like the other biomedical signal is also susceptible to noise due to different acquisition techniques and movement of patient [58]. Figure 1-4 shows a volumetric illustration of MRI of the brain whiles Figure 1-5 depicts the various views of an MRI data.

EEG (Electroencephalogram) data represents electrical activities of the brain

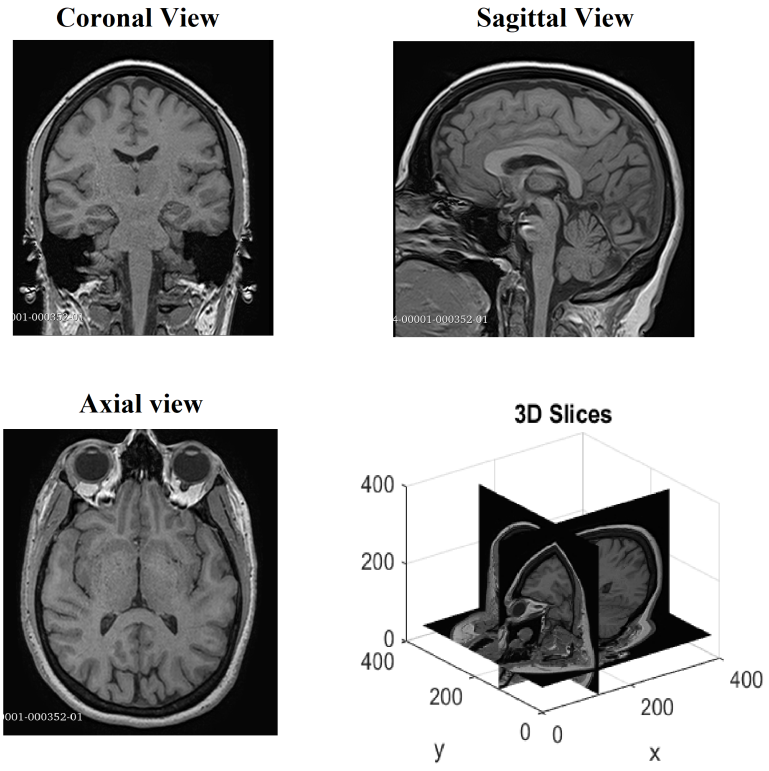
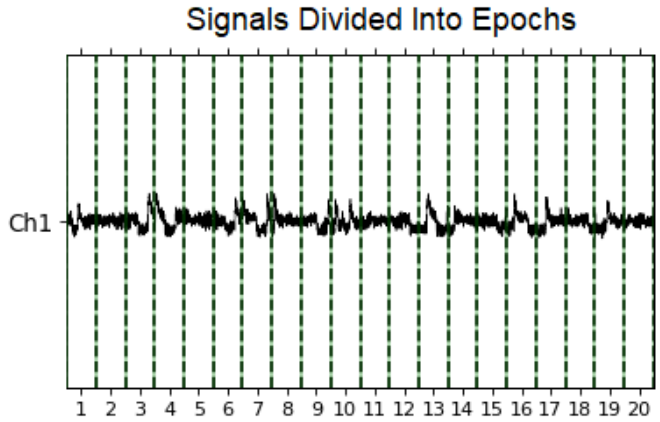


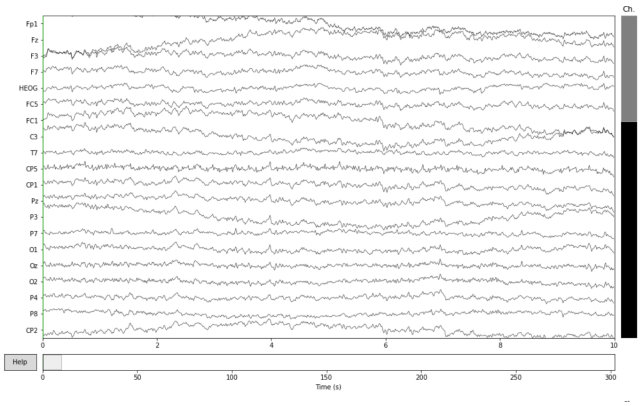
Figure 1-5: 3D Brain MRI with the three different views

that is recorded by placing several electrodes on a scalp. The signals provide the means of understanding the brain and how it works. EEG's are used extensively in neuro-science, cognitive science, psycho-physiological research and other medical fields. They can be used within a wide range of applications mostly in Brain Computer Interface (BCI) applications [59, 60]. As mentioned for MRI data, the main issue that affects the performance of such applications is the quality of the recorded EEG signal. Figure 1-6 illustrates a single channel EEG data divided into epochs for analysis and a multi-channel EEG data. Noisy Artifacts produced during the recording of the EEG signal impact directly on the quality of the acquired neural signal. Many methods have been proposed to remove artifacts from EEG recordings using different techniques [61, 62, 63, 64, 65, 66, 11, 67, 68]. The performance of BCI applications is dependent on the quality of the EEG signals. Refer to Figure 1-7 for visual illustration of EEG signal with EOG (eye movement) artifacts and its ground-truth.

A major issue with medical data is that it needs rigorous pre-processing before



(a) A typical single channel EEG data



(b) A multi-channel EEG data

Figure 1-6: EEG data

they can be used for any analysis and prediction work. Secondly, the artifacts in this type of data are independent since they have completely different generating mechanism and therefore requires different methods for tackling them [69]. Lastly, recording EEG or MRI data is expensive therefore discarding the whole data after some noise introduction is not an option [11]. As a result, there is a need to keep or preserve the available data and find ways of improving the data. Methods such as PCA (principal components analysis) and (ICA) independent component analysis [14, 70, 71, 72, 69, 73] have been proven to be successful in the recovering of such data without artifacts. However, there is no theoretical or experimental evidence that ICA or PCA are the only correct concept for extracting brain sources or isolating brain networks [74]. Figure 1-8 presents ICA sources recovered from multi-channel EEG data with artifacts.

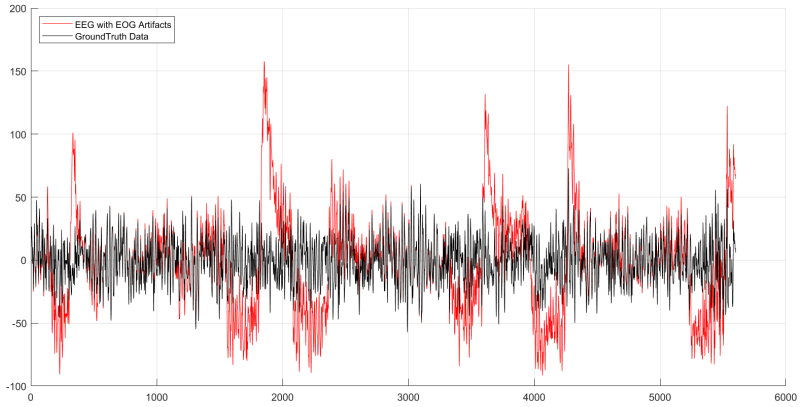


Figure 1-7: EEG data with artifact and its ground-truth data

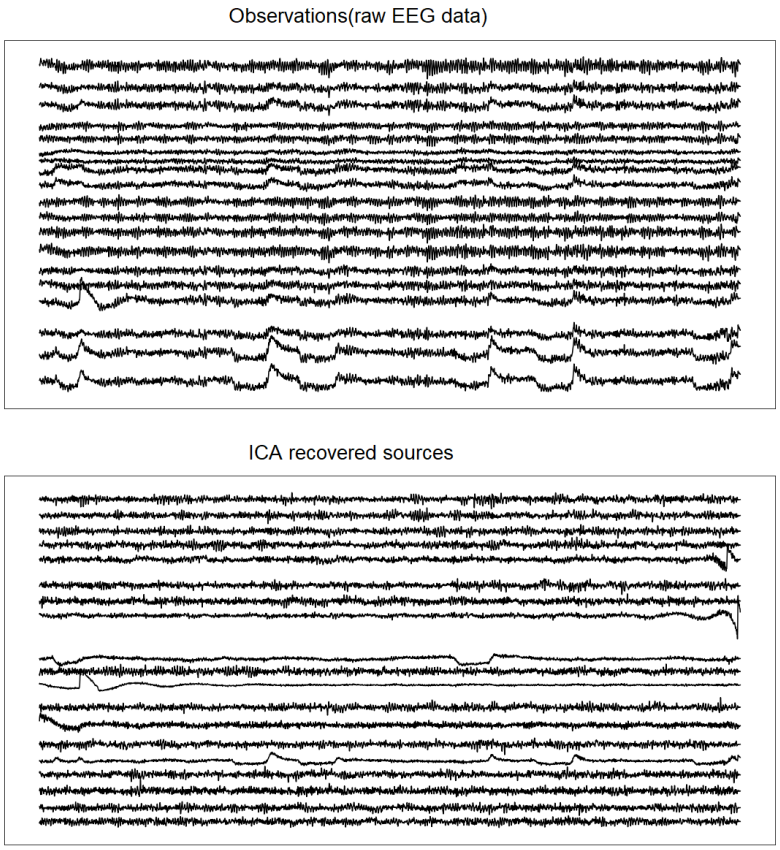


Figure 1-8: EEG data reconstruction using ICA

1.6 Challenges with completing Biomedical signals

Many algorithms [75, 76, 77, 64, 78, 79, 80, 11, 81, 82] have been developed to solve biomedical signals and its associated problems. Algorithms implementing matrix and tensor decomposition [26, 75, 83, 76, 11] have been recently proposed to

complete and recover these data from the original noisy input data. Some of these methods are based on low-rank matrix and tensor decomposition algorithms for the task of completing the missing pieces in the data [74, 75, 32, 13, 84, 85, 51, 86, 87]. For a comprehensive overview of such algorithms, refer to [24, 88, 89, 36, 90]. The major issue with completing medical data and time series data in general is the technicality in the structure of the data and the fact the there is no one method to fix all the problems [61, 91, 11, 92]. Most methods used in biomedical data reconstruction are tailored towards one type of data, mainly because of dimensionality issues and also MRI and biomedical data are different owing to their acquisition process. Different MRI data such as fMRI, dynamic weighted MRI, cardiac MRI, breast scans, require different techniques, data components and hyper-parameter tuning for reconstruction. In addition, due to dimensionality issues with biomedical data, it is very difficult to analyze the whole data. The memory capacity required to process such data becomes prohibitively expensive for most matrix and tensor completion methods [29, 93, 24]. Furthermore, some tensor decomposition techniques such as Tucker, TT and TR do not have unique solutions [86, 94]. There is therefore the need to find appropriate regularization to the optimization functions before reconstruction. Because the matrix and tensor completion problem is ill-conditioned there is always the task of finding appropriate constraints that work well for a particular data type [95, 96]. Constraints such as nonnegativity, sparseness, smoothness, decorrelation, statistical independence, etc have been adopted in various factorization models to improve the performance of the reconstruction process. Some methods also assume that all artifacts are very similar during a given recording session, which is an unrealistic assumption for artifacts caused by movements of the subject. Because this assumption does not hold completely, it becomes challenging to remove other artifacts in cases in which the regularity assumption is violated [97]. Also most matrix completion and reconstruction methods require the data to be reshaped into a long or tiny vector/Matrix, this process may undermine the natural spatial structure of MRI data [49, 22]. Solving these tasks efficiently can have enormous positive impact in neuro-imaging and neuro-physiological data for medicine and treatment of patients with physiological illness.

1.7 Related works

Various methods has been implemented for matrix and tensor completion using several optimization techniques. Lu *et al.* [98, 99] applies the Alternating Direction Method of Multipliers (ADMM) [41] as a solver for linearly constrained convex problems with separable objectives. They present the unified frameworks and convergence analysis using Gauss-Seidel ADMMs and Jacobian ADMMs. These frameworks tackle the objective function by minimizing their separable majorant surrogates to solve problems with non-separable objectives. ADMM is effective in dealing with large-scale problems and solve optimization problems that have multiple nonsmooth terms in the objective [100]. One of the early and popular works using tensor unfolding and applying nuclear norm to the resultant matrices was [100]. They extended the matrix nuclear norm to the tensor case in their simple SiLRTC, FaLRTC, HaLRTC algorithms. The HaLRTC algorithm used the ADMM optimization model for the completion task. The solution for HaLRTC can be accelerated depending on the hyper-parameters used.

The limitations with choosing the best rank number for completion introduced the methods that employed automatic rank selections and rank increments. A rank increment method is characterized by the fact that the tensor should be initialized with a lower rank approximation than its target rank [9]. A typical Canonical Polyadic Decomposition(CPD) with a rank too small will not be able to fit the data well whereas the big rank number may also result in over-fitting. Finding a way around this issue, Yokota *et al.* [49] adopts a Canonical Polyadic (CP) completion model with smoothness constraints on latent features for time series data. The smoothness constraint increases the minimum number of rank components although the variability and flexibility of the approximation model is decreased. This makes it hard to determine the upper bound of the tensor rank of the original tensor. Therefore a rank increment strategy is further used which allow to find optimal rank. The approach however, has difficulty is obtaining theoretical results on convergence which makes it difficult to analyse the model. This problem is attributed to the unit-norm constraint imposed on the model. On the same rank increment idea, the Tucker

decomposition for missing slices method [9] introduces a rank increment strategy on the factor matrices during the optimization process. This method provides a good solution for the completion of time series data. Rather than using the ADMM method, the Alternating Least Squares (ALS) optimization algorithm [95, 101] is used to improve the convergence characteristics during the optimization process. This strategy addresses the non-uniqueness of a solution for a particular tensor. Therefore, the effectiveness of the completion is dependent on the rank initialization. Following this and using the ALS optimization, the work in [52] implements a rank selection method using TR decomposition. The approach is known to find relatively close ranks for the core tensors automatically by gradually increasing the ranks in each iteration based on approximation error on each of the core tensors. The method has relatively lower storage costs and is particularly useful for tensors with inexact TT or TR structures. On the downside, one of the challenge with incremental strategies apart from the choice of rank update rules is how to append new entries to the core tensor at each iteration as the rank increases. Initializing new entries with zero does not affect the rank of the tensor which leaves the current estimate at a local minima. On the other hand appending arbitrary non-zero entries may destroy the similarity between the original and the factors computed at the previous iteration. To solve this an optimized procedure that updates the core tensors with vectors that belong to the null space of the current image estimate to provide a robust reconstruction is proposed in [102]. A TT and rank optimization approach by Bengua *et al.* [39] implements a nuclear norm minimization and multilinear matrix factorization based on SiLRTC [38]. The models referred to as SiLRTC-TT and tensor completion by parallel matrix factorization via Tensor Train (TMac-TT) makes it convenient to capture hidden information from tensors. However the method is based on the Block Coordinate Descent (BCD) algorithm to alternatively optimize a group of variables in the data. Experimenting on applying weights to tensor cores, the TT-WOPT [103] and TR-WOPT [32] for high dimension tensors applies the gradient-based optimization method to finding the optimal latent core tensors from the observed tensor data and recover the missing entries. The upside of using the weights in the optimization stage is that it helps in minimizing the distance

between weighted real data and weighted optimization objective. Nevertheless, the algorithms are sensitive to the setting of ranks so they use the same rank value for all their experiments. Also if the data is too large with high missing rate TT-WOPT has high computational cost and is not able to effectively recover missing entries.

The idea of sparseness in data and its advantage to completion has also been exploited by researchers for various tensor decomposition techniques [47, 104, 105]. The Sparse Tensor train Optimization (STTO) by Yuan *et al.* [43] uses their TT-WOPT method but with the introduction of sparse constraint on the incomplete data and is solved via gradient descent algorithm. The method works well if the data is transformed into a higher order form.

The use of Riemannian optimization techniques for low rank tensor completion has recently been adopted [106, 107, 108, 109]. The methods have been implemented using various tensor decomposition approach such as Tucker, CP and recently tensor ring decomposition. The method by [110] implements the tensor ring completion using pre-conditioning approach. Other method where a pre-conditioning with rank constraints achieves great performance for missing values in data has also been implemented that exploits the fundamental structures of symmetry, due to non-uniquenes of most tensor decompistion methods [111]. A new Riemannian metric or inner product is proposed that exploits the least-squares structure of the cost function and accounts for the structured symmetry present in the Tucker decomposition optimization criterion. Optimization methods such as gradient descent and conjugate gradient descent for solving the underlying data are used in these models. The frameworks proposed are computationally efficient for recovering missing data. However, the limitation in most of these methods is the time need to perform such operations and in some cases results are not comparable to other completion techniques.

The most pioneering work in recovering missing pixels using the Hankel folding technique is ALOHA [112], which has achieved desirable visual results in natural images. However, the patch-by-patch approach used in ALOHA simply treats each Hankel matrix independently, focusing only on local intra-patch information while failing to capture global inter-patch information. [113, 9] proposed the multi-way

delay-embedding transform (MDT) algorithm, which mathematically shifts the entire rows or columns of pixels into a much higher order tensor with Hankel structure. The MDT approach is again adopted for time series forecasting by representing the data as low-rank block Hankel tensors (BHT) [114] and incorporating the Autoregressive Integrated Moving Average (ARIMA) method for prediction. Wang *et al.* [115] applies the delay embedding to traffic data for traffic state estimation using sparse mobile sensor data. The Truncated nuclear norm (TNN) minimization technique is used with ADMM algorithm for solving such problem. Because of the Hankel nature of the input data, the size of the resulting tensor is much larger than the original, necessitating a large amount of storage memory, high computational cost for real time applications and preventing MDT from being used on ordinary PCs or laptops. A fast MDT approach [116] which uses circulant matrices for the computation of the MDT approximation is introduced to resolve the high computational cost problem. The works by Sedighin *et al.* [51] and Zheng *et al.* [117] extends the MDT method and incorporates the patch-by-patch Hankelization to the TC problem. The methods uses the ALS algorithm with TT and TR decomposition respectively. However, the algorithm focuses much on local intra-patch information. Furthermore, because each patch is treated independently, determining the subspace dimensions for tensor decomposition methods is difficult. Xu *et al.* [118] introduced clustering into the Hankelization game/scene by applying k-means clustering to 3D hankelized patches. The method known as Hankel tensor induced fast TNN (HFTNN), uses a randomized tensor SVD algorithm to recover the underlying missing data. As common to most Hankel methods, the computation cost is sacrificed and the method requires a pre-defined ranks. The computational cost of this algorithm is high because the input data is too large data and the data has to be transformed with specific kernel/window size to obtain good results.

Amid the various schemes for removing noise in EEG and MRI data, wavelet-based denoising techniques [71, 72, 119, 120, 121], blind source separation techniques (ICA, PCA and Non-negative Matrix Factorization) [122, 123, 18, 70, 124, 69, 73] and Machine learning approaches [125, 126] takes the central place in the signal processing field. ICA helps to optimize an objective function that approximates

independent component measures. EEG data recorded at multiple scalp sensors are linear sums of temporally independent components that arise from spatially fixed, distinct, or overlapping brain areas where propagation time delays are negligible, making ICA useful for blind source separation of EEG. Furthermore, the artifacts and EEG are distinct because they are generated by entirely different mechanisms [69]. Nevertheless, with ICA and its associated methods there is still a limited understanding of how the uncertainty in the variables affects the reconstructed EEG signal after the artifact components have been removed [127]. In recent times, tensor decomposition algorithms have also been adopted for removal and completion of EEG data [75, 26, 83, 76, 11]. These methods have proven to be effective in obtaining a low rank approximation of the data while still removing the unwanted parts. Though most of these methods are sensitive to rank selection and data missing ratio.

The idea for applying low rank tensor approximation for reconstructing MRI Diffusion weighted images with motion artifacts have been exploited by many researchers [128, 129, 130, 131, 132, 133]. The methods by [134, 135, 136, 137] uses a Hankel approach they term "Lifting" to generate higher order k-space sampled data and then applies low rank matrix completion for the task of reconstructing the MRI images from the under-sampled k-space data.

Deep learning approaches [138, 139, 140, 141, 142, 85, 143] for tensor completion and image reconstruction has also been vigorously investigated because of their strong ability to produce natural visual images from similar contextual samples. The works by [141] termed Deep Image Prior (DIP) uses an unsupervised learning approach for in-painting and other image reconstruction task. The network is able to take a noise input and perform completion using the noisy and incomplete data as a prior for optimization. The Manifold Modelling in Embedded Space (MMES) [143] method provided an interpretation to the DIP and convolutions where Hankelized image patches were used to optimize an auto-encoder network for reconstruction of incomplete noisy data. Diffusion weighted image reconstruction using Deep Learning approaches [144, 145] have also been investigated by [146, 147]. The model termed MoDL-MUSSELS is a generalized framework based on the existing MUSSELS algo-

rithm [137] that is implemented for correcting of phase errors in multi-shot diffusion weighted echo-planar MR images. However, while deep learning-based methods have achieved impressive results on complex occasions, they have their own limitations. On the one hand, updating network parameters requires a significant amount of time, and the hardware requirements are relatively high, limiting their widespread use in simple image processing tasks. The performance of most deep learning-based methods, on the other hand, is sensitive to the size and proportion of the missing parts [118].

These models and approaches demonstrate the feasibility and effectiveness of using various completion techniques to estimate missing data. However, there are still some challenges and limitations that need to be addressed. For example, most efficient neural network methods or models employ a deep learning based architecture. As a result, it requires massive amount of images and hardware resources to train the model. However, there is a shift from heavy computations and resources to methods that are able to perform reconstruction in a matter of minutes.

The addition of constraints to the optimization problem poses a significant challenge. Because the use of different constraints introduces new variants of the optimization problem. The question is what constraints to apply in order to produce the best approximation for the given data and parameters. The majority of constraints are applied based on assumed observations [36].

Parameterization and determining the best tensor network or structures for most datasets in order to provide a low-rank approximation of a sufficiently low-order with a relatively low computational complexity must also be addressed [12, 148].

Chapter 2

Tensor Decomposition Techniques

This section briefly present concepts that are essential for a better grasp of the material. We recommend the reader to read [6, 24, 12, 88] for a more thorough introduction to the fundamental of tensors and tensor decomposition methods. Tensors are generalizations of matrices and therefore treated as multi-dimensional data [89]. Tensor Decomposition is a generalization of the matrix factorization. Tensor decomposition factorizes a tensor into a low-rank tensor with the entries comparable to the original tensor with a minimum rank [24]. There are various approaches or techniques used to factorise or decompose a tensor. Tensor decom-

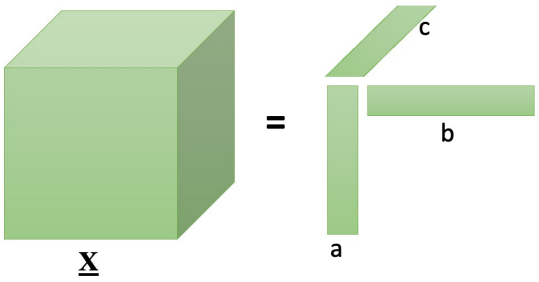


Figure 2-1: Tensor decomposition as a sum of rank 1 tensor

position techniques, such as those described in [6, 149, 24, 12], can be used to approximate low rank data structures while preserving the underlying data and features, which has the advantage of saving memory and processing time. The major factorization methods utilized under tensors are the Tucker [150] decomposition, the Canonical Polyadic (CP) [151] and Tensor networks (TNs). TNs decompose

higher-order tensors into sparsely interconnected small-scale low-order core tensors [152]. Tensor Train (TT) [153] and Tensor Ring (TR) [1] [154, 155, 156, 157] are two powerful and relatively simple tensor networks representing a higher order tensor as a train and ring (or chain) of third order core tensors. For a comprehensive study on tensor networks and applications in physics and machine learning, refer to [158, 159, 160, 161, 162, 163, 164, 165, 88, 166, 167].

2.1 Canonical Polyadic decomposition (CPD)

The Canonical Polyadic (CP) [151], [168] decomposition is considered as a rank decomposition where the tensor is decomposed into a sum of rank one tensors. Canoncial Polyadic/CANDECOMP PARAFAC (CP) was proposed by Hitchcock [151], the method was rediscovered in 1970 by Carroll et al [168] naming it Canonical Decomposition (CANDECOMP). The paper by Harshman et al [169] called it PARAllel FACtors (PARAFAC). PCA is expanded to higher-order tensors through CP decomposition. In actuality, PCA is a matrix-based CP decomposition. Because it is conceptually straightforward, CP decomposition is an appealing technique. The CP-rank is defined as smallest/minimum number of rank-1 tensors that generates the given tensor as their sum. The standard CP factorization is expressed/formalized by:

$$\min_{\hat{\underline{\mathbf{X}}}} \left\| \underline{\mathbf{X}} - \hat{\underline{\mathbf{X}}} \right\|, \quad \text{where} \quad \hat{\underline{\mathbf{X}}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \cdots \circ \mathbf{c}_r = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \tag{2.1}$$

where $\circ$ denotes the outer product of vectors and $[\![\ldots]\!]$ is a shorthand notation of CP factorization. $\{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$ are latent factor matrices corresponding to each of N modes, respectively. Where N is the number of dimensions of the tensor and R denotes the rank of a tensor. The rank of a tensor $\underline{\mathbf{X}}$ is defined as the smallest number of rank-one tensors that generate $\underline{\mathbf{X}}$ as their sum [6]. The CP model can be interpreted as a sum of R rank-one tensors, which is related to the definition of CP rank that is the smallest integer R for which the equation holds.

[1] It is also known as Tensor Chain (TC) decomposition.

However, for CP decomposition the best rank approximation may not exist [6]. Therefore the above minimization problem is generally an ill-posed one. However, incorporating a coherence constraint in the minimization problem helps to overcome this ill-posedness [170, 171]. Moreover, the CP decomposition of tensors with order higher than two, is often unique under mild assumptions, this constitutes one of the attractive properties of the CP decomposition [24]. Constraints such as orthogonality and non-negativity ensure the existence of the minimum of the optimization criterion used [172, 173].

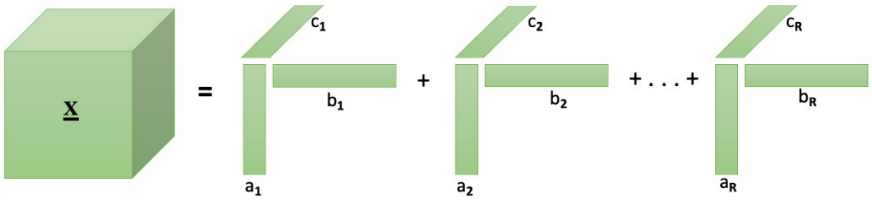


Figure 2-2: Graphical Representation of CP Decomposition

2.1.1 ALS algorithm for CP decomposition

A popular approach for computing the CP decomposition of a tensor, is the CP Alternating Least Squares (ALS) algorithm [168, 169]. This method is very efficient in many tensor decomposition applications. The main idea of this algorithm is to fix all factor matrices except for one in order to optimize for the non-fixed matrix, and then repeat this step for each matrix until some stopping criterion is met.

$$\begin{aligned} \mathbf{A} &\leftarrow \arg\min_{\hat{\mathbf{A}}} \|\mathbf{X}_{(1)} - \hat{\mathbf{A}}(\mathbf{C} \odot \mathbf{B})^T\| \\ \mathbf{B} &\leftarrow \arg\min_{\hat{\mathbf{B}}} \|\mathbf{X}_{(2)} - \hat{\mathbf{B}}(\mathbf{C} \odot \mathbf{A})^T\| \\ \mathbf{C} &\leftarrow \arg\min_{\hat{\mathbf{C}}} \|\mathbf{X}_{(3)} - \hat{\mathbf{C}}(\mathbf{B} \odot \mathbf{A})^T\| \end{aligned} \tag{2.2}$$

This minimization problem above can be solved optimally by:

$$\begin{aligned}\hat{\mathbf{A}} &= \mathbf{X}_{(1)}[(\mathbf{C} \odot \mathbf{B})^T]^\dagger &&= \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B})(\mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B})^\dagger \\ \hat{\mathbf{B}} &= \mathbf{X}_{(2)}[(\mathbf{C} \odot \mathbf{A})^T]^\dagger &&= \mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A})(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A})^\dagger \\ \hat{\mathbf{C}} &= \mathbf{X}_{(3)}[(\mathbf{B} \odot \mathbf{A})^T]^\dagger &&= \mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A})(\mathbf{B}^T\mathbf{B} * \mathbf{A}^T\mathbf{A})^\dagger\end{aligned} \tag{2.3}$$

The CP-ALS algorithm requires that the rank used for approximation be passed in as an argument. Furthermore, while the ALS algorithm is simple to understand and implement, it may require several steps to converge and may fail to converge to a global optimum. This means that the initialization of the ALS algorithm has a significant impact on its performance [89]. Algorithm 3 presents a generalization of the method to the N^{th} order case. Many tensor completion algorithms have been proposed using CP decomposition. In [29], the CP weighted optimization (CP-WOPT) is proposed. The tensor completion problem is formulated as a weighted least squares (WLS) problem and an optimization algorithm is used to find the optimal CP factors. Another popular algorithm that uses CP decomposition is the Fully Bayesian CP Factorization (FBCP)[47]. This method employs a Bayesian probabilistic model to find the optimal CP factors and CP rank simultaneously.

Algorithm 3: CP ALS [6]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and ranks $[R_1, R_2, \ldots, R_N]$
Output: Approximate CP of the tensor $\underline{\mathbf{X}}$

Initialize $\mathbf{A}^{(n)} \in \mathbb{R}^{I \times R}$ for $n = 1, \ldots, N$
repeat
  for $i = 1, \ldots, N$ **do**
    $\mathbf{Y} \leftarrow \mathbf{A}^{(1)^T}\mathbf{A}^{(1)} * \cdots \mathbf{A}^{(n-1)^T}\mathbf{A}^{(n-1)} * \mathbf{A}^{(n+1)^T}\mathbf{A}^{(n+1)} * \cdots * \mathbf{A}^{(N)^T}\mathbf{A}^{(N)}$
    $\mathbf{A}^{(n)} \leftarrow \mathbf{X}^{(n)}\big(\mathbf{A}^{(N)} \odot \cdots \mathbf{A}^{(n+1)} \odot \mathbf{A}^{(n-1)} \odot \cdots \odot \mathbf{A}^{(1)}\big)\mathbf{Y}^\dagger$
    Perform normalization for the columns of $\mathbf{A}^{(n)}$
    Store norms as λ
  end
until stopping criteria is satisfied
return $\lambda, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}$

2.2 Tucker decomposition

Tucker [150] decomposition is a generalized SVD [174] for matrices. It decomposes a tensor into a core tensor and a set factor matrices which correspond to different core scaling along each mode of the tensor. Tucker decomposition was also proposed by Ledyard R.Tucker[150] in 1966. A 3-way Tucker decomposition of a tensor, $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$ is expressed as:

$$\begin{aligned} \min_{\hat{\underline{\mathbf{X}}}} \left\| \underline{\mathbf{X}} - \hat{\underline{\mathbf{X}}} \right\| \quad \text{where} \quad \hat{\underline{\mathbf{X}}} &= \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} \ \mathbf{a}_r \circ \mathbf{b}_r \circ \ldots \circ \ \mathbf{c}_r \\ &= \underline{\mathbf{G}} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} \\ &= [\![\underline{\mathbf{G}}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] \end{aligned} \tag{2.4}$$

where $\underline{\mathbf{G}} \in \mathbb{R}^{P \times Q \times R}$, $\mathbf{A} \in \mathbb{R}^{I \times P}$, $\mathbf{B} \in \mathbb{R}^{R \times Q}$, $\mathbf{C} \in \mathbb{R}^{K \times R}$. In Tucker framework, $\underline{\mathbf{G}}$ is the core tensor, which expresses how and to which extent different tensor elements interact with each other. The factor matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are often referred to as the principal component in the respective tensor mode. The matricized version of

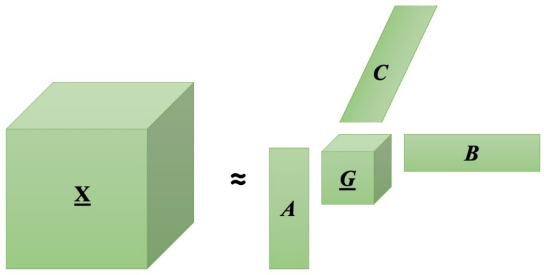


Figure 2-3: Graphical Illustration of Tucker Decomposition

the above tensor is given

$$\begin{aligned} \hat{\mathbf{X}}_{(1)} &= \mathbf{A}\mathbf{G}_{(1)}(\mathbf{C} \otimes \mathbf{B})^T \\ \hat{\mathbf{X}}_{(2)} &= \mathbf{B}\mathbf{G}_{(2)}(\mathbf{C} \otimes \mathbf{A})^T \\ \hat{\mathbf{X}}_{(3)} &= \mathbf{C}\mathbf{G}_{(3)}(\mathbf{B} \otimes \mathbf{A})^T \end{aligned} \tag{2.5}$$

In the general N-way case we get

$$\begin{aligned}
\hat{\underline{\mathbf{X}}} &= \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_n=1}^{R_N} g_{r_1 r_2 \cdots r_N} \ \mathbf{a}^{(1)}_{i_1 r_1} \mathbf{a}^{(2)}_{i_2 r_2} \cdots \mathbf{a}^{(N)}_{i_N r_N} \\
&= \underline{\mathbf{G}} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \cdots \times_N \mathbf{A}^{(N)} \\
&= [\![\underline{\mathbf{G}}; \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}]\!] \\
\hat{\mathbf{X}}_{(n)} &= \mathbf{A}^{(n)} \mathbf{G}_{(n)} \big(\mathbf{A}^{(N)} \otimes \cdots \otimes \mathbf{A}^{(n+1)} \otimes \mathbf{A}^{(n-1)} \otimes \cdots \mathbf{A}^{(1)} \big)^T
\end{aligned} \tag{2.6}$$

The introduction of Higher Order SVD (HOSVD) by [175] permits that for a given tensor $\underline{\mathbf{X}}$, we can easily find an exact Tucker decomposition of rank $(R_1, R_2, \ldots, R_N)$ where $R_n = rank_n(\underline{\mathbf{X}})$. It is shown that the HOSVD is a generalization of the matrix SVD. The HOSVD algorithm is able to efficiently compute the leading left singular vectors of $\mathbf{X}_{(n)}$. The decomposition is known as the truncated HOSVD when $R_n < rank_n(\underline{\mathbf{X}})$ for one or more n. In fact, the HOSVD's core tensor is all-orthogonal, which has implications for truncating the decomposition [6]. The method ins formalised in Algorithm 4

Algorithm 4: HOSVD [175]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and ranks $[R_1, R_2, \ldots, R_N]$
Output: Approximate HOSVD of the tensor $\underline{\mathbf{X}}$

```
for i = 1, , ..., N do
    A^(n) ← leading left singular vectors of X_(n)
end
G ← X ×_1 A^(1)^T ×_2 A^(2)^T ··· ×_N A^(N)^T
return G, A^(1), A^(2), ..., A^(N)
```

An alternative approach to computing the Tucker decomposition provided by De Lathauwer *et. al* [176] is the Higher Order Orthogonal Iteration (HOOI). The HOOI method is basically an ALS algorithm that uses the result of HOSVD on a tensor as a starting point for initializing the factor matrices. [6]. HOOI uses the SVD instead of eigen value decomposition for computing only the dominant singular vectors of $\mathbf{X}_{(n)}$. The method is effective when used with truncated HOSVD, since the successive application of the ALS algorithm allows for more accurate decompositions [6, 89]. Refer to Algorithm 5 for details.

Tucker decomposition is also a widely used tool for tensor completion. The

Algorithm 5: HOOI [176]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and ranks $[R_1, R_2, \ldots, R_N]$
Output: Approximate HOOI of the tensor $\underline{\mathbf{X}}$

Initialize factor matrices using HOSVD
repeat
  for $i = 1, \ldots, N$ **do**
    $\underline{\mathbf{Y}} \leftarrow \underline{\mathbf{X}} \times_1 \mathbf{A}^{(1)^T} \cdots \times_{n-1} \mathbf{A}^{(n-1)^T} \times_{n+1} \mathbf{A}^{(n+1)^T} \cdots \times_{(N)} \mathbf{A}^{(N)^T}$
    $\mathbf{A}^{(n)} \leftarrow$ leading left singular vectors of $\mathbf{Y}_{(n)}$
  end
until stopping criteria is satisfied
$\underline{\mathbf{G}} \leftarrow \underline{\mathbf{X}} \times_1 \mathbf{A}^{(1)^T} \times_2 \mathbf{A}^{(2)^T} \cdots \times_N \mathbf{A}^{(N)^T}$
return $\underline{\mathbf{G}}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(N)}$

algorithms by [38] (SiLRTC, FaLRTC, and HaLRTC) extend the nuclear norm regularization for matrix completion to tensor completion by minimizing the Tucker rank of the incomplete tensor. A Tucker low-n-rank tensor completion (TLnR) is proposed in [44] and their experimental results outperforms traditional nuclear norm minimization methods.

2.3 Tensor Train (TT) decomposition

Tensor networks (TNs) decompose higher-order tensors into sparsely interconnected small-scale low-order core tensors [152]. Tensor Train (TT) proposed by [153] is powerful and relatively simple tensor network used to represent a higher order tensor as a train of third order core tensors. The TT decomposition is also known as Matrix Product State (MPS) in quantum physics [158, 177]. It has several practical applications in machine learning [163, 178, 179, 180], compressing deep neural networks [181, 182, 183], tensor completion [46, 32, 39, 184], and hyperspectral image super-resolution [185, 186]. The memory storage requirement of this networks scale linearly with the order of the tensor so they break the *curse of dimensionality* which is a main bottleneck in handling high order data tensors [187]. A TT decomposition

of an $N^{th}-$ order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is written as

$$\begin{aligned} \underline{\mathbf{X}}\left(i_1, i_2, \ldots, i_N\right) \cong & \sum_{r_1, \ldots, r_{N-1}} \underline{\mathbf{G}}^{(1)}\left(i_1, r_1\right) \underline{\mathbf{G}}^{(2)}\left(r_1, i_1, r_2\right) \ldots \\ & \underline{\mathbf{G}}^{(N-1)}\left(r_{N-2}, i_{N-1}, r_{N-1}\right) \underline{\mathbf{G}}^{(N)}\left(r_{N-1}, i_N, r_0\right), \end{aligned} \tag{2.7}$$

and the N-tuple $(1, r_1, \ldots, r_{N-1}, 1)$ is called TT-ranks. It should be noted that, $\underline{\mathbf{G}}^{(1)}$ and $\underline{\mathbf{G}}^{(N)}$ are considered as matrices. The element-wise representation of the data tensor $\underline{\mathbf{X}}$ is represented as

$$x(i_1, i_2, \ldots, i_N) = \mathbf{G}_{i_1}^{(1)} \times \mathbf{G}_{i_2}^{(2)} \times \cdots \mathbf{G}_{i_N}^{(N)} \cong \left(\prod_{n=1}^{N} \mathbf{G}_{(i_n)}^{(n)} \right), \tag{2.8}$$

where $\mathbf{G}_{(i_n)}^{(n)} \in \mathbb{R}^{R_{n-1} \times R_n}$ is the i_n-th slice of the core tensor $\underline{\mathbf{G}}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$. where for $n = 1, \ldots, N$, $\underline{\mathbf{G}}^{(n)} \in \mathbb{R}^{r_{n-1} \times I_n \times r_n}$ and $r_0 = r_n = 1$. A shorthand notation [152] to express the relation between the approximated tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and the core tensors is as follows

$$\underline{\mathbf{X}} \cong \ll \underline{\mathbf{G}}^{(1)}, \underline{\mathbf{G}}^{(2)}, \ldots, \underline{\mathbf{G}}^{(N)} \gg$$

The notation $\ll . \gg$ is the operation that transform the core tensors to the approximated tensor. TT-format uses $\mathcal{O}(dnr^2)$ memory to store $\mathcal{O}(nd)$ elements. It should be taken into account that TT decomposition is most efficient with smaller ranks. TT decompositon has been applied to the task of tensor completion. The algorithm by [39] creates a low-TT-rank tensor completion where the nuclear norm regularizations are imposed on the more balanced unfoldings of the tensor. The assumption of low-rank imposed on the TT-ranks, improves performance of the method. TT-ALS is proposed in [188], employs the alternative least squares (ALS) method to find the TT decomposition factors to solve tensor completion problem. A gradient-based completion algorithms (TT-WOPT, TT-SGD) discussed in [103, 43, 40],seeks to find the TT decomposition by gradient descent method and simple cost functions.These methods are able to recover data with high missing rates.

2.4 Tensor Ring (TR) decomposition

Tensor Ring decomposition [2] (TR) [154, 155, 156, 157] similar to TT is a tensor network that represents a higher order tensor as a ring (or chain) of third order core tensors. The TT can be considered a special case of the TR decomposition. Let tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ be an Nth- order data. The TR decomposition represents the data tensor $\underline{\mathbf{X}}$ into a sequence of latent core tensors $\underline{\mathbf{G}}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$, $n = 1, 2, \ldots, N$. The element-wise representation of the data tensor $\underline{\mathbf{X}}$ in the TR format can be expressed as

$$\underline{\mathbf{X}}(i_1, i_2, \ldots, i_N) \cong \mathrm{Tr}\left(\prod_{n=1}^{N} \underline{\mathbf{G}}^{(n)}(i_n)\right), \tag{2.9}$$

where $\underline{\mathbf{G}}^{(n)}(i_n) \in \mathbb{R}^{R_{n-1} \times R_n}$ is the i_n-th lateral slice matrix of the core tensor $\underline{\mathbf{G}}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$. The expanded form of (2.9) is

$$\begin{aligned} \underline{\mathbf{X}}\left(i_1, i_2, \ldots, i_N\right) \cong \sum_{r_0=1}^{R_0} \cdots \sum_{r_{N-1}=1}^{R_{N-1}} \underline{\mathbf{G}}^{(1)}\left(r_0, i_1, r_1\right) \ldots \\ \underline{\mathbf{G}}^{(N)}\left(r_{N-1}, i_N, r_0\right), \end{aligned} \tag{2.10}$$

and the N-tuple $(R_0, R_1, \ldots, R_{N-1})$ is called TR-ranks. Note that in the TR decomposition, we have $R_0 = R_N$ and it is also shown in [157] that the TR-ranks satisfy $R_0 R_n \leq \mathrm{rank}\left(\mathbf{X}_{\langle n \rangle}\right)$ for $n = 1, 2, \ldots, N$. As shown for the TT case, the shorthand notation for the TR decomposition is similarly written as [152]

$$\underline{\mathbf{X}} \cong \ll \underline{\mathbf{G}}^{(1)}, \underline{\mathbf{G}}^{(2)}, \ldots, \underline{\mathbf{G}}^{(N)} \gg .$$

The TT decomposition is a special case of the TR decomposition for $R_0 = R_N = 1$. This means that the first and last cores in the TT decomposition are matrices and rest of them are $3rd-$order tensors. For graphical illustration of the TT and the TR decompositions, see Figures 2-4 (a)-(c). For a comprehensive overview on fast algorithms for TR decomposition, see [3].

[2] It is also known as Tensor Chain (TC).

Algorithm 6: TR-SVD algorithm [157].

Input : A tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, a prescribed tolerance ϵ, and an initial rank R_0 as a divisor of $\text{rank}_\delta\left(\mathbf{X}_{\langle 1 \rangle}\right)$

Output: Approximate representation of the tensor $\underline{\mathbf{X}}$ in the TR format $\widehat{\underline{\mathbf{X}}} = \ll \widehat{\underline{\mathbf{X}}}^{(1)}, \widehat{\underline{\mathbf{X}}}^{(2)}, \ldots, \widehat{\underline{\mathbf{X}}}^{(N)} \gg$, such that $\left\|\underline{\mathbf{X}} - \widehat{\underline{\mathbf{X}}}\right\|_F \leq \varepsilon \|\underline{\mathbf{X}}\|_F$ and the TR-ranks $\{R_0, R_1, \ldots, R_{N-1}\}$;

Compute $\delta = \frac{\varepsilon \|\underline{\mathbf{X}}\|_F}{\sqrt{N}}$;

$\mathbf{C} = \text{reshape}\left(\underline{\mathbf{X}}, \left[I_1, \frac{\text{numel}(\underline{\mathbf{X}})}{I_1}\right]\right)$;

$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}_\delta(\mathbf{C})$;

Set $R_0 R_1 = \text{rank}(\mathbf{S})$;

$\widehat{\underline{\mathbf{X}}}^{(1)} = \text{reshape}(\mathbf{U}, [R_0, I_1, R_1])$;

$\underline{\mathbf{C}} = \text{reshape}\left(\mathbf{S}\mathbf{V}^T, \left[R_1, \prod_{j=2}^{N} I_j, R_0\right]\right)$;

for $n = 2, \ldots, N-1$ **do**

  $\mathbf{C} = \text{reshape}\left(\underline{\mathbf{C}}, \left[R_{n-1} I_n, \frac{\text{numel}(\underline{\mathbf{C}})}{R_{n-1} I_n}\right]\right)$;

  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}_\delta(\mathbf{C})$;

  $R_n = \text{rank}(\mathbf{S})$;

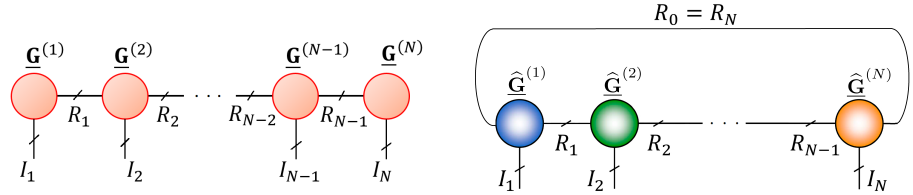
  $\widehat{\underline{\mathbf{X}}}^{(n)} = \text{reshape}(\mathbf{U}, [R_{n-1}, I_n, R_n])$;

  $\mathbf{C} = \mathbf{S}\mathbf{V}^T$;

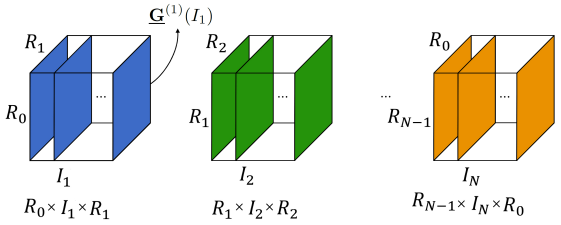
  $\underline{\mathbf{C}} = \text{reshape}\left(\mathbf{C}, \left[R_n, \prod_{j=n+1}^{N} I_j, R_0\right]\right)$

end

$\widehat{\underline{\mathbf{X}}}^{(N)} = \text{reshape}(\mathbf{C}, [R_{N-1}, I_N, R_0])$;



(a) Tensor train (TT) decomposition (b) Tensor ring (TR) decomposition



(c) Slice-wise representation of the tensor ring (TR) decomposition

Figure 2-4: Illustration of the tensor train (TT) and tensor ring (TR) decompositions.

2.5 Tubal Decomposition

Tubal tensor decomposition, commonly known as tensor-Singular Value Decomposition (t-SVD) is a special type of tensor decomposition representing a 3rd-order tensor as a product of three 3rd-order tensors where the middle tensor has nonzero components only in some tubes which are located in the diagonal parts of the tensor ([25, 26, 189, 190]), see Figure 2-5, for graphical illustration of the t-SVD an its truncated version. The generalization of the t-SVD to higher order tensors has been proposed in [191].

The number of nonzero tubes is called tubal rank. Unlike the Tucker decomposition or CPD, the truncated t-SVD gives the best approximation in the least-squares sense for any unitary invariant tensor norm [25]. To further explain the concept of t-SVD, if $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, then the t-SVD of the tensor $\underline{\mathbf{X}}$, admits the model $\underline{\mathbf{X}} = \underline{\mathbf{U}} * \underline{\mathbf{S}} * \underline{\mathbf{V}}^T$, where $\underline{\mathbf{U}} \in \mathbb{R}^{I_1 \times R \times I_3}$, $\underline{\mathbf{V}} \in \mathbb{R}^{R \times I_2 \times I_3}$ are orthogonal tensors and the tensor $\underline{\mathbf{S}} \in \mathbb{R}^{R \times R \times I_3}$ is f-diagonal ([25, 26]). Computing t-SVD of a tensor is performed in the Fourier domain. Here similar to the t-product, it is sufficient to consider only the truncated SVD of the first $\lceil \frac{I_3+1}{2} \rceil$ frontal slices $\widehat{\underline{\mathbf{X}}}(:,:,i)$, $i = 1, 2, \ldots, \lceil \frac{I_3+1}{2} \rceil$, (as described in [192]). More precisely, we first com-

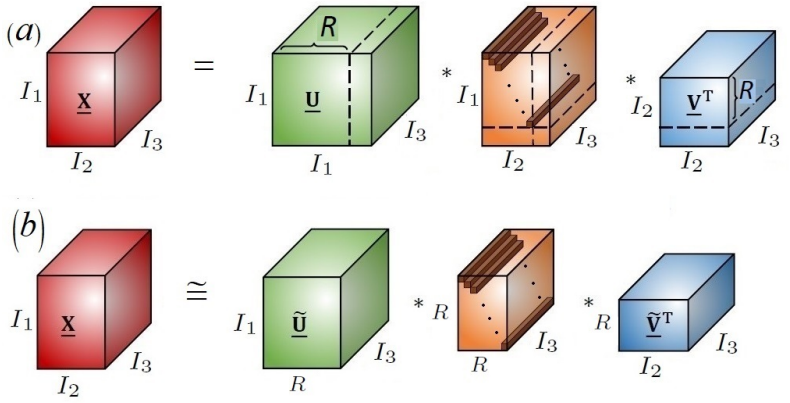


Figure 2-5: Illustration of (**a**) tensor-SVD (t-SVD) and (**b**) Truncated t-SVD for a 3rd-order tensor

Algorithm 7: Fast t-SVD

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and target tubal rank R;
Output: $\underline{\mathbf{U}}_R \in \mathbb{R}^{I_1 \times R \times I_3}$, $\underline{\mathbf{S}}_R \in \mathbb{R}^{R \times R \times I_3}$, $\underline{\mathbf{V}}_R \in \mathbb{R}^{I_2 \times R \times I_3}$;

$\widehat{\underline{\mathbf{X}}} = \text{fft}\left(\underline{\mathbf{X}}, [], 3\right)$;
for $i = 1, \ldots, \lceil \frac{I_3+1}{2} \rceil$ **do**
  $[\widehat{\mathbf{U}}(:,:,i), \widehat{\mathbf{S}}(:,:,i), \widehat{\mathbf{V}}(:,:,i)] = \text{truncated\_svd}\left(\widehat{\underline{\mathbf{X}}}(:,:,i), R\right)$;
end
for $i = \lceil \frac{I_3+1}{2} \rceil + 1, \ldots, I_3$ **do**
  $\widehat{\underline{\mathbf{U}}}\,(:,:,i) = \text{conj}(\widehat{\mathbf{U}}(:,:,I_3 - i + 2))$;
  $\widehat{\underline{\mathbf{S}}}\,(:,:,i) = \widehat{\mathbf{S}}(:,:,I_3 - i + 2)$;
  $\widehat{\underline{\mathbf{V}}}\,(:,:,i) = \text{conj}(\widehat{\mathbf{V}}(:,:,I_3 - i + 2))$;
end
$\underline{\mathbf{U}} = \text{ifft}\left(\widehat{\underline{\mathbf{U}}}, [], 3\right)$, $\underline{\mathbf{S}} = \text{ifft}\left(\widehat{\underline{\mathbf{S}}}, [], 3\right)$, $\underline{\mathbf{V}} = \text{ifft}\left(\widehat{\underline{\mathbf{V}}}, [], 3\right)$.

pute the truncated SVD of the first $\lceil \frac{I_3+1}{2} \rceil$ frontal slices $\widehat{\underline{\mathbf{X}}}(:,:,i)$, $i = 1, 2, \ldots, \lceil \frac{I_3+1}{2} \rceil$ as follows

$$\begin{aligned}
&[\widehat{\underline{\mathbf{U}}}(:,:,i), \widehat{\underline{\mathbf{S}}}(:,:,i), \widehat{\underline{\mathbf{V}}}(:,:,i)] = \\
&\text{truncated_svd}\left(\widehat{\underline{\mathbf{X}}}(:,:,i), R\right) \\
&i = 1, 2, \ldots, \lceil \frac{I_3+1}{2} \rceil,
\end{aligned}$$

and store $\widehat{\underline{\mathbf{U}}}(:,:,i)$, $\widehat{\underline{\mathbf{V}}}(:,:,i)$ and $\widehat{\underline{\mathbf{S}}}(:,:,i)$, for $i = 1, 2, \ldots, \lceil \frac{I_3+1}{2} \rceil$. Then, they are used to recover the remaining factors $\widehat{\underline{\mathbf{U}}}(:,:,i)$, $\widehat{\underline{\mathbf{S}}}(:,:,i)$, $\widehat{\underline{\mathbf{V}}}(:,:,i)$, $i = \lceil \frac{I_3+1}{2} \rceil, \ldots, I_3$ based on the following relations

$$\begin{aligned}
\widehat{\underline{\mathbf{U}}}(:,:,i) &= \widehat{\underline{\mathbf{U}}}(:,:,I_3 - i + 2), \\
\widehat{\underline{\mathbf{S}}}(:,:,i) &= \widehat{\underline{\mathbf{S}}}(:,:,I_3 - i + 2), \\
\widehat{\underline{\mathbf{V}}}(:,:,i) &= \widehat{\underline{\mathbf{V}}}(:,:,I_3 - i + 2), \\
&i = \lceil \frac{I_3+1}{2} \rceil + 1, \ldots, I_3.
\end{aligned} \tag{2.11}$$

Also, instead of using the discrete Fourier transform matrices, one can define the t-SVD according to an arbitrary unitary transform matrix. It is shown in [193] that this approach can provide t-SVD with lower tubal rank. The full algorithm for truncated t-SVD is summarized in Algorithm 7. Many tensor completion algorithms [99, 28, 98, 194, 192] have recently being adopting tubal SVD and its truncated variants for completion.

Chapter 3

Tensorization of data via Hankelization: Different Strategies

A Hankel matrix is a type of matrix that has all its elements along the skew-diagonals being constant. A Hankel tensor is a higher order Hankel matrix. A data $\underline{\mathbf{X}}$ is called a Hankel tensor if all components $x_{i_1,i_2,\ldots,i_N}$ for which the quantity $i_1 + i_2 + \cdots + i_N$ is fixed, are the same. *Hankelization* is the procedure of generating a Hankel matrix or tensor from a given vector, matrix or tensor. The concept of Hankel tensor has been introduced in several contexts. The first paper discussing the Hankel tensor is [195] which was concerned with phase retrieval as an important topic in signal processing. It defines the Hankelization process based on [1]. Hankelization is kind of tensorization in which a raw data tensor is transformed to a higher order tensor. It is mostly used as a pre-(data) processing technique in signal processing followed by other algorithms after which the processed data is returned back to the original format. In particular, this idea has been used for reconstructing data tensors with structured missing components (several sequential columns/row are removed) and has been shown to be a very promising approach for performing the mentioned task. Figure 3-1 shows an example of such application where the TR-ALS [46] and TRLRF [13] algorithms as two efficient tensor completion algorithms fail to provide promising reconstruction while the Hankelization technique with parameter $\tau = 10$ (to be defined later) gives quite good results when used as pre-processing step before

[1] The code "hankelize" used in tensorlab toolbox.

the tensor completion is performed.

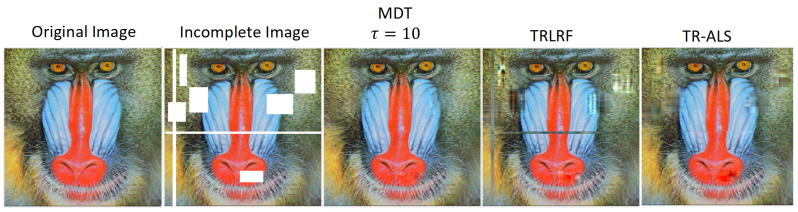


Figure 3-1: The effectiveness of using Hankelization approach as pre-processing step before completing images with structural missing pixels.

3.1 Generating a Hankel Matrix

A Hankel matrix can be generated from a vector as follows. Given a vector $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T \in \mathbb{R}^N$, and a window size τ, the operator $\mathcal{H}_\tau(\mathbf{x}) : \mathbb{R}^N \to \mathbb{R}^{\tau \times (N-\tau+1)}$ generates a Hankel matrix

$$\mathbf{X}_H = \mathcal{H}_\tau(\mathbf{x}) := \begin{pmatrix} x_1 & x_2 & \cdots & x_{N-\tau+1} \\ x_2 & x_3 & \cdots & x_{N-\tau+2} \\ \vdots & \vdots & \ddots & \vdots \\ x_\tau & x_{\tau+1} & \cdots & x_N \end{pmatrix} \in \mathbb{R}^{\tau \times (N-\tau+1)}.$$

The operator $\mathcal{H}_\tau(\mathbf{x})$ is referred to as a delay embedding transform of vector $\mathbf{x}$ with window size τ [9, 31]. Equally, the Hankelization procedure can be expressed by a duplication matrix which provides a relationship between a Hankel matrix $\mathbf{X}_H$ and a corresponding given vector $\mathbf{x}$. Example, let $\mathbf{X}_H$, be a Hankel matrix, then $\mathbf{T} \in \{0,1\}^{\tau(N-\tau+1) \times N}$, is called a duplication matrix [9, 143, 116] if

$$\text{vec}(\mathbf{X}_H) = \mathbf{T}\mathbf{x}. \tag{3.1}$$

The operator $\text{fold}_{(\mathrm{N},\tau)} : \mathbb{R}^{\tau(N-\tau+1)} \to \mathbb{R}^{\tau \times (N-\tau+1)}$ which returns back a vectorized form of a Hankel matrix to the original Hankel matrix is called folding operator, that is

$$\mathbf{X}_H = \text{fold}_{(\mathrm{N},\tau)}(\mathbf{T}\mathbf{x}). \tag{3.2}$$

A sample illustration of this definition for a vector of size 8 and window size $\tau = 5$ is presented in Figure 3-2. It is seen that the duplication matrices includes block of shifted identity matrices.

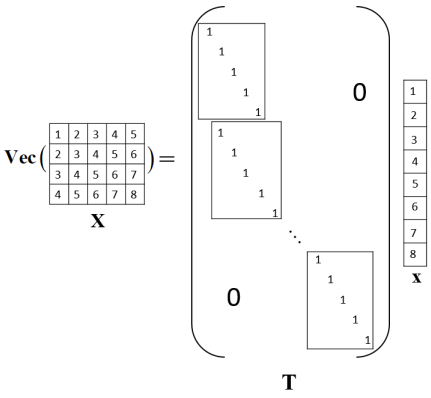


Figure 3-2: Duplication matrix for a vector of size 8 and window size $\tau = 5$.

3.1.1 Block Hankelization

A block Hankel matrix is defined similarly, where instead of the individual entries of a matrix, its blocks are repeated along the block skew-diagonals of the matrix. More precisely, from a set of matrices, we can generate a block Hankel matrix. The paper by [86, 51, 118] used this block Hankel method. See figure 3-3 for a step by step illustration of the block Hankelization procedure. Figure 3-4 further presents visual comparison on using different window sizes (τ) when performing tensor completion task with Hankel folding as a pre-processing step.

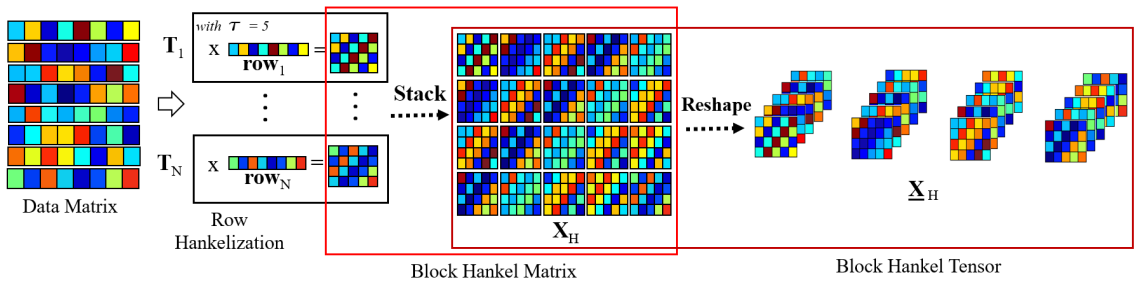


Figure 3-3: Illustration of block Hankelization of a matrix and construction of 4th-order tensor using row scanning. Similar tensor can be obtained by column scanning.

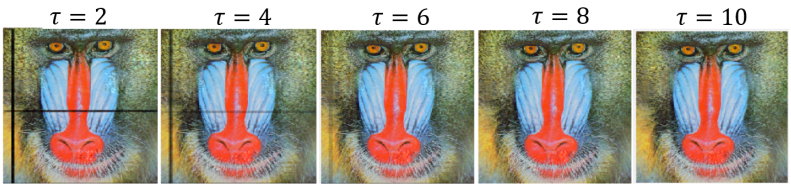


Figure 3-4: The Hankelization results for different parameters τ.

3.1.2 Row or Column *Hankelization*

The idea of row Hankelization of a data matrix to a higher order tensor was first proposed in [8]. In this work each row of the given matrix is Hankelized to a matrix and then a block Hankel matrix is produce after which is then reshaped to a block Hankel tensor, see Figure 3-5(a) for an illustrative example of such transformation. The procedure of column Hankelization procedure can be performed similarly by applying the aforementioned technique to the transpose of the underlying matrix. In [9], the column Hankelization is performed directly to a given matrix by using the so-called duplication matrices. The Hankelized tensors are called embedded space. So, the Hankelization introduced in [9] can be considered as a generalization of that proposed in [8]. It is worth mentioning that in the Hankelization framework introduced in [9, 8], each Hankel tensor is of even order, that is an Nth order tensor is transformed to a $2N$-th order tensor and based on this framework, a Hankel tensor of odd order is not defined. The main difference of this Hankelization procedures with the one in [10] is that given an N^{th} order tensor it is transformed into an $N + 1$ order tensor, therefore a Hankel tensor of odd order can be defined. Here, again the columns are Hankelized but the generated Hankel tensor is different from the one introduced in [9]. This approach perform the Hankelization procedure on each column and transform the column vector to a matrix. This resultant matrix is considered as a slice of the generated tensor, see Figure 3-5 (b) for a graphical sample of such Hankel tensors. Hankelization of the elements of a tensor can also be scanned in others ways such as zigzag shown in Figure 3-6.

3.1.3 Inverse Hankelization

A vector from which a Hankel matrix was constructed can be obtained through inverse Hankelization procedure [9] as follows

$$\mathbf{x} = \mathcal{H}_{\tau}^{-1}\left(\mathbf{X}_H\right) = \mathbf{T}^{\dagger}\mathrm{vec}\left(\mathbf{X}_H\right),$$

where $\mathbf{T}$ is a duplication matrix. The concept of delay embedding can be generalized to higher order tensors via multi-way delay embedding [9, 31, 143]. The intuition behind this generalization is that the classical delay embedding can be considered as a tensor-vector multiplication, i.e. $\mathbf{x}\times_1\mathbf{T} = \mathbf{T}\mathbf{x}$, where a vector (which is a first order tensor) uses only one duplication matrix $\mathbf{T}$. For a matrix $\mathbf{X} \in \mathbb{R}^{I_1\times I_2}$ (which is a second order tensor), two duplication matrices $\mathbf{T}_1 \in \{0,1\}^{\tau_1(I_1-\tau_1+1)\times I_1}$ and $\mathbf{T}_2 \in \{0,1\}^{\tau_2(I_2-\tau_2+1)\times I_2}$ are used. This gives us

$$\mathbf{Y} = \mathbf{X}\times_1\mathbf{T}_1\times_2\mathbf{T}_2 = \mathbf{T}_1\,\mathbf{X}\,\mathbf{T}_2^T \in \mathbb{R}^{\tau_1(I_1-\tau_1+1)\times\tau_1(I_2-\tau_2+1)},$$

where the operator $\mathrm{fold}_{(\mathrm{N},\tau)}\left(\mathbf{X}\times_1\mathbf{T}_1\times_2\mathbf{T}_2\right)$ reshapes the data matrix $\mathbf{Y}$ to a fourth order Hankel tensor of size $\tau_1 \times (I_1-\tau_1+1)\times\tau_1 \times (I_2-\tau_2+1)$.

For an Nth-order tensor, we should consider N duplication matrices $\mathbf{T}_n \in \{0,1\}^{\tau_n(I_n-\tau_n+1)\times I_n}$, $n = 1,2,\ldots,N$ and the multi-way embedded space is defined as [9]

$$\underline{\mathbf{X}}_H = \mathcal{H}_{\tau}\left(\underline{\mathbf{X}}\right) = \mathrm{fold}_{(I,\tau)}\left(\underline{\mathbf{X}}\times_1\mathbf{T}_1\cdots\times_N\mathbf{T}_N\right),$$

where $\mathrm{fold}_{(I,\tau)} : \mathbb{R}^{\tau_1(I_1-\tau_1+1)\times\cdots\times\tau_N(I_N-\tau_N+1)} \rightarrow \mathbb{R}^{\tau_1\times(I_1-\tau_1+1)\times\cdots\times\tau_N\times(I_N-\tau_N+1)}$ transforms an Nth-order tensor to an $2N$th-order tensor. The N-tuple $(\tau_1,\tau_2,\ldots,\tau_N)$ indicates the window size of different duplication matrices corresponding to different modes. Illustration for Hankelization of a matrix and transforming it to a 4th-order tensor is shown in Figure 3-5. The reverse procedure where the task is retrieving the original data tensor from the tensor $\underline{\mathbf{X}}_H$ was constructed, can be expressed as

$$\mathcal{H}_{\tau}^{-1}\left(\underline{\mathbf{X}}_H\right) = \mathrm{unfold}_{(\mathrm{I},\tau)}\left(\underline{\mathbf{X}}_H\right)\times_1\mathbf{T}_1^{\dagger}\cdots\times_N\mathbf{T}_N^{\dagger},$$

where $\text{unfold}_{(I,\tau)} = \text{fold}^{-1}_{(I,\tau)}$, and it basically transforms the $2N$th-order block Hankel tensor $\underline{\mathbf{X}}_H$ of size $\tau_1 \times (I_1 - \tau_1 + 1) \times \cdots \times \tau_N \times (I_N - \tau_N + 1)$ to an Nth-order tensor of size $\tau_1 (I_1 - \tau_1 + 1) \times \cdots \times \tau_N (I_N - \tau_N + 1)$.

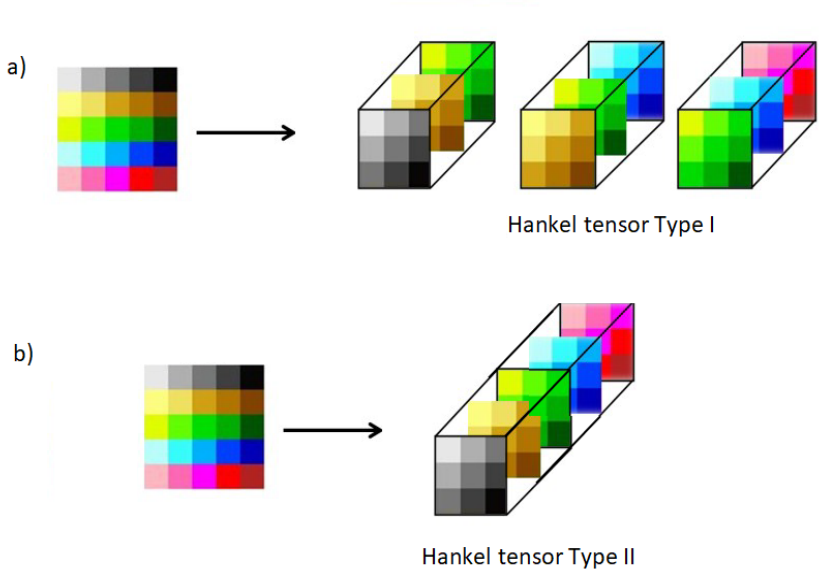


Figure 3-5: Two types of tensor Hankelization, a) Hankelization procedure introduced in [8, 9], b) Hankelization procedure introduced in [10].

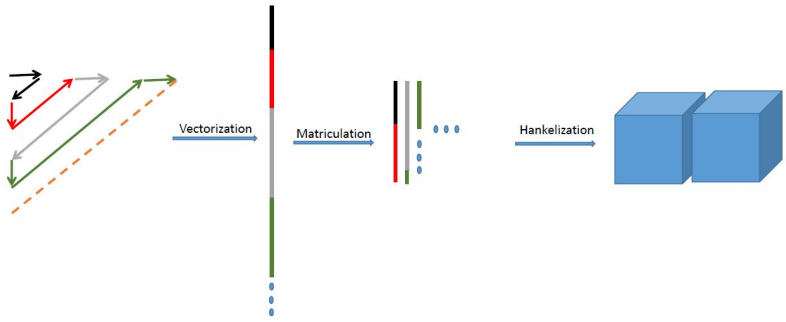


Figure 3-6: The procedure of zigzag Hankelization.

3.2 Perspectives on using Hankel data

The Hankelization procedure, also known as delay/shift embedding in the image processing community, is a technique that duplicates patches of an image with specified window sizes. It is well known that as a pre-processing step, a prior Hankelization

Figure 3-7: Completion of the "Barbara" and "Lena" image with structural missing parts using different types of Hankelization together with their corresponding PSNR and SSIM

procedure can capture the delay/shift-invariant features (e.g. non-local similarity) of signals/images [143]. This is an effective property for learning the hidden structure of images, allowing us to apply this technique in task completion [143, 9, 51]. Refer to figure 3-9 for some visual results using the delay embedding technique. In particular, it is experimentally shown that for incomplete data tensors with random missing pixels, this technique is quite efficient. Although, it is possible to use more sophisticated Hankelization, such as patch-based/block Hankelization [51, 196, 112], however we adopt the simple row Hankelization procedure and Hankelization using convolutional techniques due to its efficiency. Refer to table 3.1 for some comparison results. In our approach, each row of the frontal slices of the data tensor [2] is Hankelized using a given window size and a corresponding duplication matrix. Then, from these matrices, block Hankel matrices are constructed. Having computed all the block Hankel matrices, they are reshaped to higher order tensors. By column Hankelization, we can obtain similar or the same results. Refer to Figure 3-7 for visual results as well as PSNR and SSIM evaluation on different Hankel methods using the TT-WOPT [103] and TR-WOPT [32] completion algorithms on "Barbara" and "Lena" image. In the first row, we combine randomly missing pixels and structural missing pixels on the "Lena" image. For this case, we structurally removed 20% of pixels grouped in form of black big spots with some holes in the picture and continue

[2] In images, each frontal corresponds to RGB slices.

Figure 3-8: Reconstructing images from structural missing data using various Hankelization appraoches

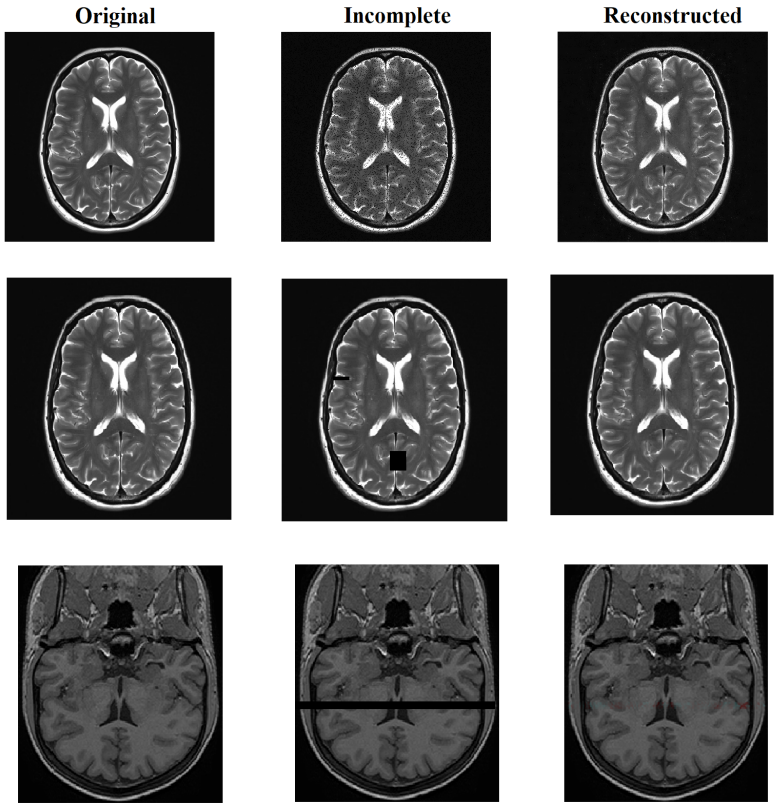
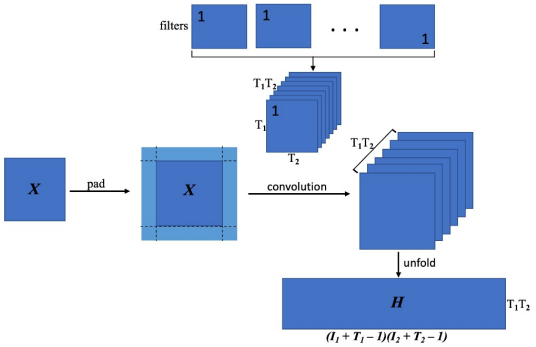


Figure 3-9: MRI reconstruction with MDT Hankelization on different missing scenarios

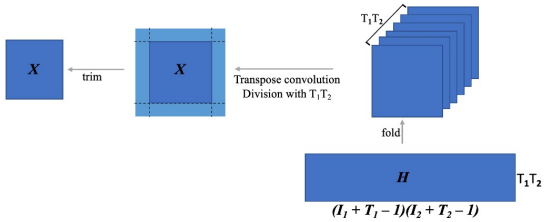
further by removing 50% pixels randomly from the image. The TR-WOPT method was used in this experiment. For "Barbara" image in the second row, about 40% of pixels were structurally removed in a form of small black spots creating holes in the image and the TT-WOPT algorithm was used. The images used in this experiment are of size $256 \times 256 \times 3$. The size of the Hankelized tensor depends on the Hankelization procedure used and for the column and row Hankelization, we use window size $\tau = [32, 32.1]$ and as a result a 6-th order tensor and size $32 \times 225 \times 32 \times 225 \times 1 \times 3$ is produced. In Figure 3-8, we consider "Facade" image with structural missing pixels. For the Hankel type II (illustrated in Figure 3-5), a tensor of order 4 and size $128 \times 129 \times 256 \times \times 3$ is generated. We apply our Tucker CUR algorithm (details in Chapter 5) for completing the mentioned images where the ranks are incrementally increased. The PSNR and SSIM of the different strategies are also given.

3.3 Convolutions as a Hankelization step

Convolutional neural networks have provided a lot of insights into image processing tasks. It has seen many applications in inpainting, denoising, recommender systems, segmentation, data interpolation and image prediction. Recently Yokota *et al.* [143] investigated the power of convolutional layers to performing Hankelization as a pre-processing step before using an auto-encoder for image in-painting. The Hankelized image patches generated using a convolutional neural network serves as a prior to an auto-encoder network. The method employs convolution as a form of delay embedding and a transformation step. In the same manner, in this work a convolutional layer is employed to perform a Hankelization step as a prior for the completion step. Table 3.1 shows the results and effectiveness of using convolutions as a Hankelization step prior to completion task.



(a) Hankelization Convolution



(b) Inverse Hankelization using transposed Convolutions

Figure 3-10: Illustration of Hankelization procedure using neural network Convolutions [9]

IMAGE	RESULTS	ORIGINAL	ROW/COLUMN [9]	PATCHBASED [51]	ZIGZAG	CONVOLUTION [143]
50% MISSING	PSNR	28.2641	30.6803	30.3937	30.0322	**37.6738**
	SSIM	0.8354	0.8708	0.8703	0.8300	**0.9634**
70% MISSING	PSNR	25.9434	28.03590	24.0412	26.5751	**34.6040**
	SSIM	0.7976	0.8031	0.7543	0.7961	**0.9121**

Table 3.1: Average PSNR and SSIM results for some natural images using various Hankel methods and the TRLRF algorithm [13] with rank 6 and tau/kernel size = 10

Chapter 4

Novel optimization method exploiting Sparsity and dictionary Learning

This chapter presents the first proposed method for this research. It explores the concept of sparsity on core tensors of a tensor ring decomposition taking advantage of dictionary learning and Hankel folding. The applicability and performance of many tensor completion methods are highly dependent on the probability distribution underlying the missing data components. The majority of matrix/tensor completion models rely on the assumption that the locations of missing entries are random and there should be some observable entries in each row or column [100, 39, 47, 157, 197]. Many of these algorithms struggle to recover the data tensor when a significant portion of it, such as several consecutive rows or columns of a frame is missing and only work well when the underlying missing components are distributed evenly. Therefore, recovering the incomplete entries would require more regularization than just what rank minimization could offer. For instance, if entire rows with missing data is filed with zeros, the result could yield the lowest rank; however, that is not the desired solution. The randomness assumption may be violated in many applications. For example, due to faulty sensors or surge problems with wireless transmissions, several rows or columns of an image may be missing. In addition, a number of fibers or slices may be missing from the MRI data. One or more signal units in

the latent tensor could be completely missing, and a video frame could be dropped during transmission. All these scenarios make the task of completion difficult. So, there is an urgent need to investigate methods to solve such challenging scenarios. Existing models that rely solely on the low-rank prior fail to account for structural missing. In general, tensor ring decomposition is not unique. Moreover, tensor completion is a very ill conditioned optimization problem, thus some regularization is necessary in order to obtain stable, relatively simple and meaningful representation. Sparsity constraints is one such regularization technique. In other words, sparse representations have two main purposes: They are a form of regularization that pushes as many parameters as possible to exact zeros. Furthermore, sparsity leads to simpler tensor models by learning what parameters can be dropped, lowering their total number. Also, sparse optimization methods are useful for signal processing problems that involve large, noisy, or incomplete data sets, where finding a simple and meaningful representation is challenging. Sparse representation has attracted great attention as it can allow us to find the characteristics of data in a high-dimensional space and therefore can be widely applied in engineering fields such as dictionary learning, signal reconstruction, image clustering, feature selection, and extraction. The method proposed by Yang et al. [198] introduce a sparsity prior on the columns to solve the failure of the low-rankness in regularizing structural matrix completion tasks. The method shows significant improvement over previous approaches in applications such as image inpainting, deraining, and interpolation of seismic data. In [94], the authors extend the matrix case to higher-dimensional tensors and consider structural missing along each mode using the TT framework with low rankness. Inspired by these successful methods, in our proposed model, the underlying tensor is regularized by a low-TR-rankness prior to exploiting the inter-fibers/slices correlations, and its fibers are regularized by a sparsity prior under dictionaries to exploit intra-fibers correlations. The work further explores the idea of Hankel folding [9, 31] as a pre-processing step in generating higher order tensors for images, videos and time series data. The proposed model is solved by an alternating direction method under the augmented Lagrangian multiplier framework.

4.1 Hankel folding with tensor ring decomposition

The idea of Hankelization was first proposed in [199] and recently generalized to tensors in [9] but so far, it has not exploited extensively for the TR representation with sparse constraints. The methods in [9, 51], employ the Tucker or the TT representations to the higher order Hankel tensor $\underline{\mathbf{X}}_H$ and then the original data tensor is reconstructed through inverse Hankelization. However, this method considers the optimization problem (1.9) with block Hankelized data tensor $\widehat{\underline{\mathbf{X}}}_H$ (described in chapter 3 and Figure 3-3) and exploit a TR decomposition as:

$$\widehat{\underline{\mathbf{X}}}_H \approx\ll \widehat{\underline{\mathbf{G}}}^{(1)}, \widehat{\underline{\mathbf{G}}}^{(2)}, \ldots, \widehat{\underline{\mathbf{G}}}^{(N)} \gg,$$

where the core tensors $\widehat{\underline{\mathbf{G}}}^{(n)}$, $n = 1, 2, \ldots, N$, are estimated using dictionary learning. Owing to this, we formulate an optimization problem in which the constrained core tensors $\widehat{\underline{\mathbf{G}}}^{(n)}$, $n = 1, 2, \ldots, N$, are updated sequentially to minimize the cost function in (1.9) and the implicit sparse representation constraint [94, 200], is imposed on the core tensors $\widehat{\underline{\mathbf{G}}}^{(n)}$ $(n = 1, 2, \ldots, N)$. Finally, the following constrained optimization problem is formulated:

$$\begin{aligned} \min_{\left\{\widehat{\underline{\mathbf{G}}}^{(1)}, \widehat{\underline{\mathbf{G}}}^{(2)}, \ldots, \widehat{\underline{\mathbf{G}}}^{(N)}\right\}} \quad & \left\| \mathbf{P}_{\Omega_H} \left(\widehat{\underline{\mathbf{X}}}_H - \ll \widehat{\underline{\mathbf{G}}}^{(1)}, \widehat{\underline{\mathbf{G}}}^{(2)}, \ldots, \widehat{\underline{\mathbf{G}}}^{(N)} \gg \right) \right\|_F^2 \\ & + \eta \sum_{n=1}^{N} \left\| \mathbf{C}^{(n)} \right\|_1 \\ s.t. \quad & \widehat{\mathbf{G}}_{(2)}^{(n)} = \mathbf{D}^{(n)} \mathbf{C}^{(n)}, \quad n = 1, 2, \ldots, N, \end{aligned} \tag{4.1}$$

where $\widehat{\mathbf{G}}_{(2)}^{(n)} \in \mathbb{R}^{I_n \times R_{n-1} R_n}$ is mode-2 matricization of the core tensor $\widehat{\underline{\mathbf{G}}}^{(n)} \in \mathbb{R}^{R_{n-1} \times I_n \times R_n}$, $\mathbf{D}^{(n)} \in \mathbb{R}^{I_n \times R_{n-1} R_n}$ is a pre-defined dictionary, $\mathbf{C}^{(n)} \in \mathbb{R}^{R_n R_{n-1} \times R_n R_{n-1}}$ is a sparse matrix and $\eta > 0$ is a regularization parameter. The method achieves good performance for η in range 0.05 to 0.3. It is important to note that while the optimization problem (4.1) is convex with respect to each core tensor $\widetilde{\underline{\mathbf{G}}}^{(n)}$, it is not convex with respect to the entire set of core tensors. Thus, there is no guarantee that the solution will eventually converge to a global minimum, though we have employed a sub-optimal approximation. In addition, unlike [94], where the sparse representa-

tion of unfolding matrices is used, we impose a sparsity representation constraint on the TR core tensors. Because TR core tensors are smaller in size than unfolding matrices, this can reduce the computational complexity of the algorithm while also improving performance. To compute the dictionary $\mathbf{D}^{(n)}$, various dictionary learning algorithms can be used such as Over-complete Discrete Cosine Transform (ODCT) dictionary [201, 201], [202], Online Dictionary Learning (ODL)[1] [203] and group sparse dictionary learning technique [200]. For the purpose of this work, the ODCT method [202, 204] was adopted because it is quite versatile and works for a broad range of data. The ODCT dictionary $\mathbf{D}^{(n)} \in \mathbb{R}^{I_n \times R_{n-1}R_n}$ is generated as follows

$$\mathbf{d}_{i,j}^{(n)} = \begin{cases} \frac{1}{\sqrt{I_n}}, & \text{if } j = 1. \\ \cos\left(\frac{(i-1)(j-1)\pi}{R_{n-1}R_n}\right), & \text{if } j > 1. \end{cases}$$

for $i = 1, 2, \ldots, I_n$ and $j = 1, 2, \ldots, R_{n-1}R_n$.

The algorithm proposed in [94], imposed the sparse representation constraints on the mode-n unfolding matrices while the nuclear norm minimization is applied on the n-unfolding matrices[2]. As a result, the proposed formulation (4.1) can be considered as a non-trivial extension and combination of techniques considered in [94] and [32], where on the one hand, this algorithm exploit l_1-norm instead of nuclear norm minimization [32] and on the other hand, similar to [94], the dictionary learning technique is employed. It is worth mentioning that ignoring the sparsity constraints, we come up to the algorithm proposed in [94]. The methods by [157, 32, 13], revealed the relation between a core tensor $\underline{\widehat{\mathbf{G}}}^{(n)}$ and the matricized tensor. From their works, they showed that:

$$\mathbf{X}_{H<n>} = \widehat{\mathbf{G}}_{(2)}^{(n)} (\widehat{\mathbf{G}}_{<2>}^{(\neq n)})^T, \quad \forall n, \tag{4.2}$$

where $\widehat{\mathbf{G}}^{(\neq n)}$ is a 3rd-order tensor of size $R_n \times \prod_{i=1, i\neq n}^{N} I_i \times R_{n-1}$ obtained by merging

[1]The MATLAB implementation of this dictionary can be found in https://github.com/kvdeepak/ksvd-denoising.

[2]Note that mode-n unfolding and n-unfolding matrices are related to the Tucker and the TT decompositions, respectively.

all other core tensors whose slice matrices are computed as

$$\underline{\widehat{\mathbf{G}}}^{(\neq n)}\left(\overline{i_{n+1}\dots i_N i_1\dots i_{n-1}}\right)=\prod_{j=n+1}^{N}\underline{\widehat{\mathbf{G}}}^{(j)}\left(i_j\right)\prod_{j=1}^{n-1}\underline{\widehat{\mathbf{G}}}^{(j)}\left(i_j\right).$$

For brevity of notation, in the rest of the paper we define $\widehat{\mathbf{G}}^{(n)} := \widehat{\mathbf{G}}^{(n)}_{(2)}$ and $\widehat{\mathbf{G}}^{(\neq n)} := \widehat{\mathbf{G}}^{(\neq n)}_{<2>}$. Using formula (4.2) and keeping in mind that the core tensors are independent, we can optimize the core tensors in optimization problem (4.1) independently while holding the other core tensors constant, as follows:

$$\begin{aligned}\min_{\{\widehat{\mathbf{G}}^{(n)}\}}\quad & \mathcal{H}\left(\widehat{\mathbf{G}}^{(n)}\right)+\eta\left\|\mathbf{C}^{(n)}\right\|_1\\ s.t.\quad & \widehat{\mathbf{G}}^{(n)}=\mathbf{D}^{(n)}\mathbf{C}^{(n)},\end{aligned}\tag{4.3}$$

for $n=1,2,\dots,N$, where

$$\mathcal{H}\left(\widehat{\mathbf{G}}^{(n)}\right)=\left\|\mathbf{P}_{\mathbf{\Omega}_{H\ \langle n\rangle}}\left(\widehat{\mathbf{X}}_{H\ \langle n\rangle}-\widehat{\mathbf{G}}^{(n)}\left(\widehat{\mathbf{G}}^{(\neq n)}\right)^T\right)\right\|_F^2.$$

The augmented Lagrangian function corresponding to the constrained optimization problem (4.3), can be constructed as

$$\begin{aligned}\mathcal{L}\left(\widehat{\mathbf{G}}^{(n)},\mathbf{C}^{(n)},\mathbf{B}\right)=\mathcal{H}\left(\widehat{\mathbf{G}}^{(n)}\right)+\eta\left\|\mathbf{C}^{(n)}\right\|_1+\left\langle\mathbf{B},\widehat{\mathbf{G}}^{(n)}-\mathbf{D}^{(n)}\mathbf{C}^{(n)}\right\rangle\\ +\tfrac{\lambda}{2}\left\|\widehat{\mathbf{G}}^{(n)}-\mathbf{D}^{(n)}\mathbf{C}^{(n)}\right\|_F^2,\end{aligned}\tag{4.4}$$

where $\mathbf{B}$ is a matrix representing the Lagrangian multipliers and λ is penalty parameter. The method works well for λ between 0.1 to 1.1. The ADMM [41] update rules for solving (4.1) are simply converted to simpler optimization problems using the Lagrangian function (4.4), which is given as follows:

$$\widehat{\mathbf{G}}^{(n)}_{k+1}=\min_{\widehat{\mathbf{G}}^{(n)}_k}\ \mathcal{H}\left(\widehat{\mathbf{G}}^{(n)}_k\right)+\frac{\lambda}{2}\left\|\widehat{\mathbf{G}}^{(n)}_k-\mathbf{D}^{(n)}\mathbf{C}^{(n)}_k+\mathbf{B}^{(k)}\right\|_F^2,\tag{4.5}$$

$$\mathbf{C}^{(n)}_{k+1}=\min_{\mathbf{C}^{(n)}_k}\ \left\|\mathbf{C}^{(n)}_k\right\|_1+\frac{\lambda}{2}\left\|\widehat{\mathbf{G}}^{(n)}_k-\mathbf{D}^{(n)}\mathbf{C}^{(n)}_k+\mathbf{B}^{(k)}\right\|_F^2,\tag{4.6}$$

$$\mathbf{B}^{(k+1)}=\mathbf{B}^{(k)}+\left(\widehat{\mathbf{G}}^{(n)}_{k+1}-\mathbf{D}^{(n)}\mathbf{C}^{(n)}_{k+1}\right).\tag{4.7}$$

It is worth noting that the ADMM algorithm is a versatile approach for solving

a wide range of optimization problems, including sparse inverse covariance selection, generalized and group lasso problems, and so on. For a comprehensive study of this technique and related problems, see [41]. Moreover, this strategy has been utilized for solving many tensor completion problems, please see [205, 94, 206, 207, 208], and for a comprehensive list of references see the review papers [36, 90] and the references therein.

4.1.1 Solving the Sub Optimization Problem (4.5)

The first optimization sub-problem (4.5) can be solved via the standard gradient descent method. To be more precise, the gradient of its objective function of (4.5) in k-th iteration is

$$\begin{aligned}\nabla_{\widehat{\mathbf{G}}_k^{(n)}} J = \mathbf{P}_{\mathbf{\Omega}_{H\ \langle n\rangle}} \left(\widehat{\mathbf{G}}_k^{(n)} \left(\widehat{\mathbf{G}}_k^{(\neq n)} \right)^T - \mathbf{X}_{H\ \langle n\rangle} \right) \widehat{\mathbf{G}}^{(\neq n)} \\ +\lambda \left(\widehat{\mathbf{G}}_k^{(n)} - \mathbf{D}^{(n)} \mathbf{C}_k^{(n)} + \mathbf{B}^{(k)} \right),\end{aligned} \tag{4.8}$$

where $J = \mathcal{L}\left(\widehat{\mathbf{G}}_k^{(n)}, \mathbf{C}^{(n)}, \mathbf{B}\right)$ and the first and second terms of (4.8) are the gradient of the first and second terms of the objective function of optimization problem (4.5), respectively. The gradient of the first term of (4.8) has been derived in [32] and the gradient of the second term can be derived by the following straightforward computations

$$\begin{aligned}\tfrac{\lambda}{2} \left\| \widehat{\mathbf{G}}_k^{(n)} - \mathbf{D}^{(n)} \mathbf{C}_k^{(n)} + \mathbf{B}^{(n)} \right\|_F^2 = \tfrac{\lambda}{2} \left(\left\| \widehat{\mathbf{G}}_k^{(n)} \right\|_F^2 + \left\| \mathbf{B}^{(n)} - \mathbf{D}^{(n)} \mathbf{C}_k^{(n)} \right\|_F^2 \right. \\ \left. +2\mathrm{Tr}\left(\widehat{\mathbf{G}}_k^{(n)\ T} \left(\mathbf{B}^{(n)} - \mathbf{D}^{(n)} \mathbf{C}_k^{(n)} \right) \right) \right),\end{aligned}$$

and taking the gradient with respect to $\mathbf{G}_k^{(n)}$.

4.1.2 Solving the Sub Optimization Problem (4.6)

In order to solve optimization problem (4.6), the Accelerated Proximal Gradient [209, 84] (APG) method is applied. Assuming that the $E\left(\mathbf{C}_k^{(n)}\right) = \left\| \mathbf{C}_k^{(n)} \right\|_1 +$

$F\left(\mathbf{C}_k^{(n)}\right)$, we have the following cost function:

$$F\left(\mathbf{C}_k^{(n)}\right) = \frac{\lambda}{2}\left\|\widehat{\mathbf{G}}_k^{(n)} + \mathbf{B}^{(k)} - \mathbf{D}^{(n)}\mathbf{C}_k^{(n)}\right\|_F^2.$$

The first step in the proximal gradient algorithms is considering an auxiliary proximal variable $\mathbf{Z}_k$, and computing a quadratic approximation of $E(\mathbf{C}_k^{(n)})$ around $\mathbf{Z}_k$ using the following equation

$$\begin{aligned} H\left(\mathbf{C}_k^{(n)}, \mathbf{Z}_k\right) = &\left(\left\|\mathbf{C}_k^{(n)}\right\|_1 + F\left(\mathbf{Z}_k\right)\right) + \left\langle \nabla_{\mathbf{C}_k} F\left(\mathbf{Z}_k\right), \mathbf{C}_k^{(n)} - \mathbf{Z}_k \right\rangle \\ &+ \tfrac{c}{2}\left\|\mathbf{C}_k^{(n)} - \mathbf{Z}_k\right\|_F^2, \end{aligned} \tag{4.9}$$

where the gradient $\nabla_{\mathbf{C}_k} F\left(\mathbf{Z}_k\right)$ is computed straightforwardly by expanding $F\left(\mathbf{Z}_k\right)$ and taking gradient with respect to $\mathbf{C}_k$ as

$$\nabla_{\mathbf{C}_k} F\left(\mathbf{Z}_k\right) = \lambda\left(\mathbf{D}^{(n)\,T}\mathbf{D}^{(n)}\,\mathbf{C}_k^{(n)} - \mathbf{D}^{(n)\,T}\left(\widehat{\mathbf{G}}_k^{(n)} + \mathbf{B}^{(k)}\right)\right). \tag{4.10}$$

The parameter $c > 0$ in (4.9) is a given parameter and in our simulation we set $c = \left\|\mathbf{D}^{(n)}\right\|_F^2$. Substituting $\mathbf{W}_k = \mathbf{Z}_k - \nabla_{\mathbf{C}_k} f\left(\mathbf{Z}_k\right)/c$ in (4.9), we obtain

$$H\left(\mathbf{C}_k^{(n)}, \mathbf{Z}_k\right) = T(\mathbf{C}_k^{(n)}, \mathbf{w}_k) - \frac{1}{2c}\left\|\nabla_{\mathbf{C}_k} f\left(\mathbf{Z}_k\right)\right\|_F^2, \tag{4.11}$$

where

$$T(\mathbf{C}_k^{(n)}, \mathbf{w}_k) = \left\|\mathbf{C}_k^{(n)}\right\|_1 + \frac{c}{2}\left\|\mathbf{C}_k^{(n)} - \mathbf{W}_k\right\|_F^2. \tag{4.12}$$

As a result, $T(\mathbf{C}_k^{(n)}, \mathbf{w}_k)$ is equivalent to $H\left(\mathbf{C}_k^{(n)}, \mathbf{Z}_k\right)$ up to a constant and it can be minimised instead of $H\left(\mathbf{C}_k^{(n)}, \mathbf{Z}_k\right)$. In the proximal gradient algorithms, instead of minimizing $E(\mathbf{C}_k^{(n)})$, which is a non differentiable optimization problem, the objective function $T(\mathbf{C}_k^{(n)}, \mathbf{Z}_k)$ is minimized for a sequence of proximal variables $\mathbf{Z}_k^j$. This leads to the following minimization problem which can be solved iteratively

$$\mathbf{C}_{k,j}^{(n+1)} = \underset{\mathbf{C}_k^{(n)}}{\arg\min}\ \left\|\mathbf{C}_k^{(n)}\right\|_1 + \frac{c}{2}\left\|\mathbf{C}_k^{(n)} - \mathbf{W}_k^{j+1}\right\|_F^2, \tag{4.13}$$

where $\mathbf{W}_k^{j+1} = \mathbf{Z}_k^j - \nabla_{\mathbf{C}_k} f\left(\mathbf{Z}_k^j\right)/c$. Finally, we obtain the following closed form solution

$$\mathbf{C}_{k,j+1}^{(n+1)} = \text{soft}\left(\mathbf{W}_k^{j+1}, \frac{1}{c}\right), \tag{4.14}$$

where the soft thresholding operator is defined as $\text{soft}(g,\tau) = \text{sign}(g)\max(|g|-\tau, 0)$. Note that in (4.13), the soft thresholding operator is applied in an element-wise manner to all components of the matrix $\mathbf{W}_k^{j+1}$. There are several possible options for updating the proximal variable $\mathbf{Z}_k^{j+1}$, but in this paper we used the accelerated one introduced in [209] as follows

$$\begin{cases} t^{k+1} = \frac{1+\sqrt{4t^{2j}+1}}{2}, \\ \mathbf{Z}_k^{j+1} = \mathbf{C}_{k,j}^{(n+1)} + \frac{t^k-1}{t^{k+1}}\left(\mathbf{C}_{k,j+1}^{(n+1)} - \mathbf{C}_{k,j}^{(n+1)}\right), \end{cases}$$

where $t^1 = 1$. The whole procedure is summarized in the pseudo-code of Algorithm 8. The stopping criterion which we used was that the relative error be less than a predefined tolerance or the maximum number of iterations is reached. The Matlab Poblano toolbox 1.1 [210] was used for solving the underlying nonlinear optimization problems.

4.2 Discussion on computational complexity

The majority of recent tensor completion algorithms use the nuclear norm minimization formulation, which necessitates multiple computations of the singular value decomposition (SVD). This increases their computational complexity while also slowing them down. To prevent this drawback, the proposed method does not use nuclear norm minimization therefore avoiding the SVD computation. For simplicity of expressions, let us assume $I_1 = I_2 = \ldots = I_N = I$ and $R_1 = R_2 = \ldots = R_N = R$, then the computational complexity of TR-ALS [46] and TRLRF [13, 206] are $\mathcal{O}(pNR^4I^N + NR^6)$ and $\mathcal{O}(NR^2I^N + NR^6)$, respectively, where p is constant between 0 and 1. The main operation in our algorithms is matrix-matrix multiplication in (4.8) where we need to multiply matrices $\widehat{\mathbf{G}}_k^{(n)} \in \mathbb{R}^{R^2 \times I_n}$ and $\widehat{\mathbf{G}}_k^{(\neq n)} \in \mathbb{R}^{\prod_{i\neq n} I_i \times R^2}$,

Algorithm 8: Algorithm for Tensor Ring Decomposition with Sparse Representation (TRDSR)

Input : An incomplete data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the observation index tensor $\underline{\mathbf{\Omega}}$, TR-ranks $(R_0, R_2, \ldots, R_{N-1})$, (for simplicity, we assume that $R_n = R \ \forall \ n$), a window size τ and regularization parameter $\lambda > 0$.

Output: Completed data tensor $\widehat{\underline{\mathbf{X}}}$

Hankelize the data tensor $\underline{\mathbf{X}}$ and $\underline{\mathbf{\Omega}}$ as $\underline{\mathbf{X}}_H$ and $\underline{\mathbf{\Omega}}_H$

Initialize the TR core tensors for the Hankelized tensor $\underline{\mathbf{X}}_H$ as $\widehat{\underline{\mathbf{G}}}^{(1)}, \widehat{\underline{\mathbf{G}}}^{(2)}, \ldots, \widehat{\underline{\mathbf{G}}}^{(N)}$ with Gaussian random tensors

while *A stopping criterion is not satisfied* **do**

  for $n = 1, 2, \ldots, N$ **do**

    Compute dictionary $\mathbf{D}^{(n)}$ for mode-2 matricization of the core tensor $\widehat{\underline{\mathbf{G}}}^{(n)}$

    Solve optimization Problem (4.3) for $\widehat{\mathbf{G}}^{(n)}$ and $\mathbf{C}^{(n)}$ using the update ADMM rules (4.5)-(4.7)

  end

  Compute $\widehat{\underline{\mathbf{X}}}_H = \ll \widehat{\underline{\mathbf{G}}}^{(1)}, \widehat{\underline{\mathbf{G}}}^{(2)}, \ldots, \widehat{\underline{\mathbf{G}}}^{(N)} \gg$

  Compute $\widehat{\underline{\mathbf{X}}}_H = \mathbf{P}_{\underline{\mathbf{\Omega}}_H}\left(\widehat{\underline{\mathbf{X}}}_H\right) + \mathbf{P}_{\underline{\mathbf{\Omega}}_H^{\perp}}\left(\widehat{\underline{\mathbf{X}}}_H\right)$

  De-Hankelize $\widehat{\underline{\mathbf{X}}}_H$

end

which costs $\mathcal{O}(R^2 I^N)$. This shows the proposed algorithm has lower complexity in comparison to others.

4.3 Experiments

This section presents the evaluation of the proposed algorithm using MRI images and some EEG data. The Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index measure (SSIM) and Normalised Root Mean Square Error (NRMSE) are used to evaluate and compare the performance of various algorithms. The NRMSE is computed as :

$$\text{NRMSE} = \sqrt{\frac{\langle(\widehat{\underline{\mathbf{X}}} - \underline{\mathbf{X}})^2\rangle}{\langle\underline{\mathbf{X}}^2\rangle}},$$

where $\underline{\mathbf{X}}$ and $\widehat{\underline{\mathbf{X}}}$ are the original data tensor with all observations and the reconstructed data tensor respectively. $\langle . \rangle$ is an expectation operator. The NRMSE was calculated based on all the entries of the data. The PSNR is defined as

$$\text{PSNR} = 10 \log_{10} \left(255^2/\text{MSE}\right),$$

where

$$\text{MSE} = \left\|\widehat{\underline{\mathbf{X}}} - \underline{\mathbf{X}}\right\|_F^2 / \text{num}\left(\underline{\mathbf{X}}\right),$$

and "num(.)" denotes the number of components of the tensor $\underline{\mathbf{X}}$ whiles the SSIM is defined as

$$\text{SSIM} = \frac{(2\mu_{\underline{\mathbf{X}}}\mu_{\widehat{\underline{\mathbf{X}}}} + c_1)(2\sigma_{\underline{\mathbf{X}}\widehat{\underline{\mathbf{X}}}} + c_2)}{(\mu^2_{\underline{\mathbf{X}}} + \mu^2_{\widehat{\underline{\mathbf{X}}}} + c_1)(\sigma^2_{\underline{\mathbf{X}}} + \sigma^2_{\widehat{\underline{\mathbf{X}}}} + c_2)}$$

where

$\mu_{\underline{\mathbf{X}}}, \mu_{\widehat{\underline{\mathbf{X}}}}$,$\sigma^2_{\underline{\mathbf{X}}}, \sigma^2_{\widehat{\underline{\mathbf{X}}}}$ are the mean and variances of $\underline{\mathbf{X}}$ and $\widehat{\underline{\mathbf{X}}}$ respectively. $\sigma_{\underline{\mathbf{X}}\widehat{\underline{\mathbf{X}}}}$ is the covariance $\underline{\mathbf{X}}$ between $\widehat{\underline{\mathbf{X}}}$ and $c_1 = (k_1 L)^2, c_2 = (k_2 L)^2$ are stabilization variables

4.3.1 Experiments on MRI, images and time series data

The brain imaging data used in this research include:

- MRI dataset One: Predictive Analytics Competition (PAC) dataset challenge

organized by the Translational Psychiatry Group at University Muenster, Germany (https://photon-ai.com/pac). The data consist of structural MRI data from a balanced MDD patients and healthy controls. The data is of size $121 \times 145 \times 121$

- MRI dataset Two: Resting state functional T2-weighted MRI EPI series and also structural T1-weighted MP-RAGE images from Skoltech Advanced Data Analytics in Science and Engineering group. The images are of size $384 \times 384 \times 352$. Some of the images were resized to $256 \times 256 \times 3$ for easy handling.

MRI data completion: In these sets of simulations, we attempted to use realistic missing ratios because very often, most actual corrupted MRI data have 30% to 60% corruption when measured. The incomplete data tensor used from MRI dataset one was Hankelized into an 5th-order tensor of size $64 \times 193 \times 64 \times 193 \times 3$ using the window sizes $\tau = (64, 64, 3)$. Pertaining to the MRI from dataset two, we used a window size $\tau = (32, 32, 4)$ and Hankelized the 3rd-order tensor into a 6th-order tensor of size $32 \times 90 \times 32 \times 90 \times 4 \times 7$.

In the first experiments, images from MRI dataset two was used. The performance of our algorithm was compared with some low rank completion algorithms using different missing rates and different structural missing types. The algorithms used were TRLRF [13], MDT [9], TRAR [52], TRALS[46], and TRWOPT [32]. In order to provide a fair comparison with other algorithms, the hyper-parameters were tuned as stated in each paper to make performance as best as possible. The reconstructed images are displayed in Figure 4-1 and Figure 4-2. Figure 4-1 presents results on degraded image with both random missing pixels and structural missing data. In the structural case some rows and columns in the MRI slices are removed and then about 50% of the rest of the pixels are also randomly removed. The results in Figure 4-2 shows the reconstruction from approximately 40% random missing pixels. The results are compared with other completion algorithms. It is seen that our proposed method, in general, outperforms the other algorithms in these two cases.

Using other MRI data from dataset two, 60% of random pixels are removed and PSNR of the slices from the reconstruction are compared. This experiment showed

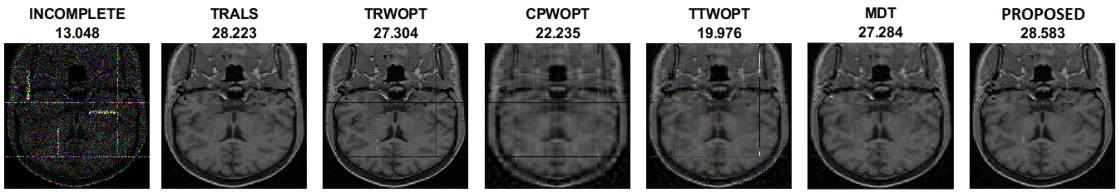


Figure 4-1: Visual and PSNR comparison of MRI image reconstructed using different tensor completion algorithms on structured missing pixels.

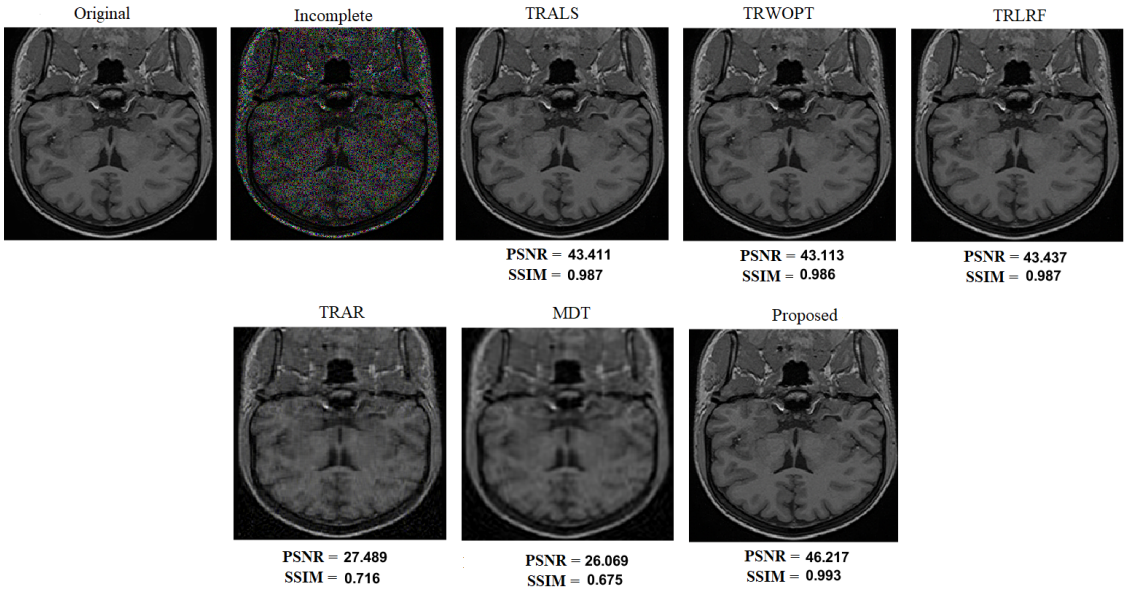


Figure 4-2: Comparison PSNR results with other algorithms on MRI Dataset Two with 40% of random pixels missing

that the algorithm is able to reconstruct the data entirely, since the PSNR results from the slices are comparable. Visual and PSNR results can be found in Figure 4-3. We present the PSNR results of the popular TRLRF [13] method compared with our proposed method in another experiment. In this scenario 50% of the pixels are removed and after reconstruction results for some selected slices are shown in 4-4. In addition, the plots in Figure 4-5 also shows the SSIM and PSNR results from this experiment.

In a similar manner, 40% and 60% of pixels are randomly removed from the MRI data from dataset one. The PSNR of some selected slices from the reconstructed images are presented. Refer to Figures 4-6 and 4-7 view the results. Results obtained confirms that our proposed method works well on the different slices of the entire data and also works on different MRI datasets.

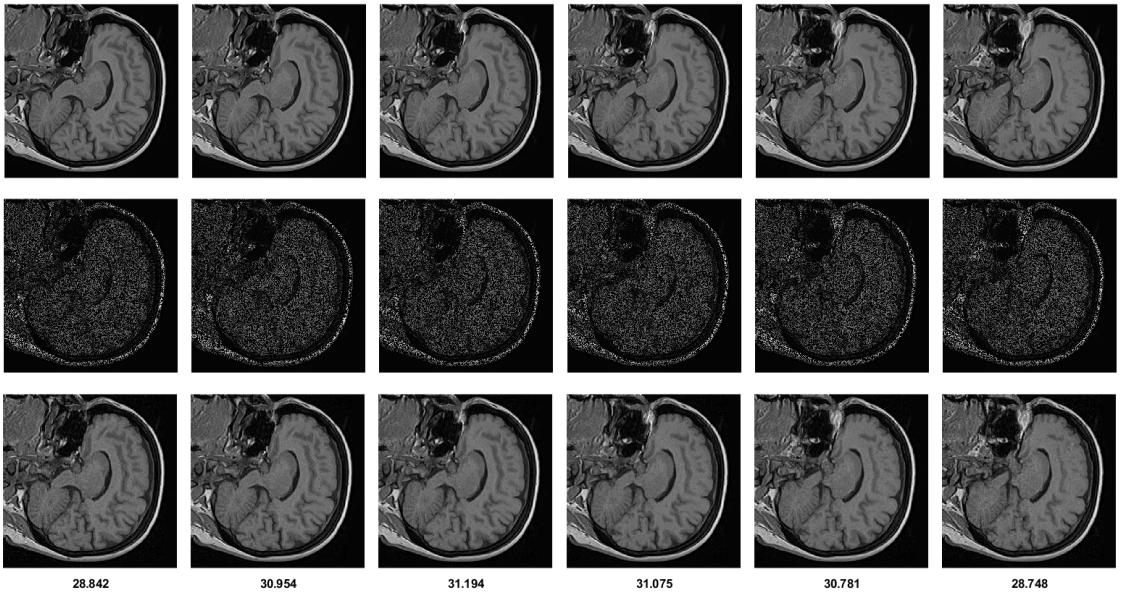


Figure 4-3: PSNR results on MRI Dataset Two with 60% of random pixels missing

In the second experiment, we applied a scratching degradation on MRI slices (first row in Figure 4-8) from dataset two, in the second row, we removed 10% of pixels and 10% of columns and rows randomly (second row in Figure 4-8), in the third row, we removed 40% of pixels and added some spots in the image (third row in Figure 4-8), and lastly, we removed 40% columns and rows (fourth row in Figure 4-8). The recovered images together with indicated corresponding PSNR index are reported in Figure 4-8. Our computer experiments indicate that the proposed algorithm also provides good results for recovering real-life medical images.

MRI data Denoising: To demonstrate the effectiveness of our method for denoising task, the proposed algorithm was further tested with noisy MRI images where we compared our method with two neural network based architectures. MRI images are of size $164 \times 164 \times 3$. During the experiments, frames of the MRI data were degraded with Poisson and Salt and Pepper Noise. The salt and pepper noise was added with a noise density of 0.1 to all the frames. The PSNR and SSIM of the reconstructed results are compared with the Deep Image Prior [141] and the Unet model [211]. Figure 4-9 presents the visual and quantitative results of this denoising task. The results show that our proposed method is comparable to some deep learning architectures.

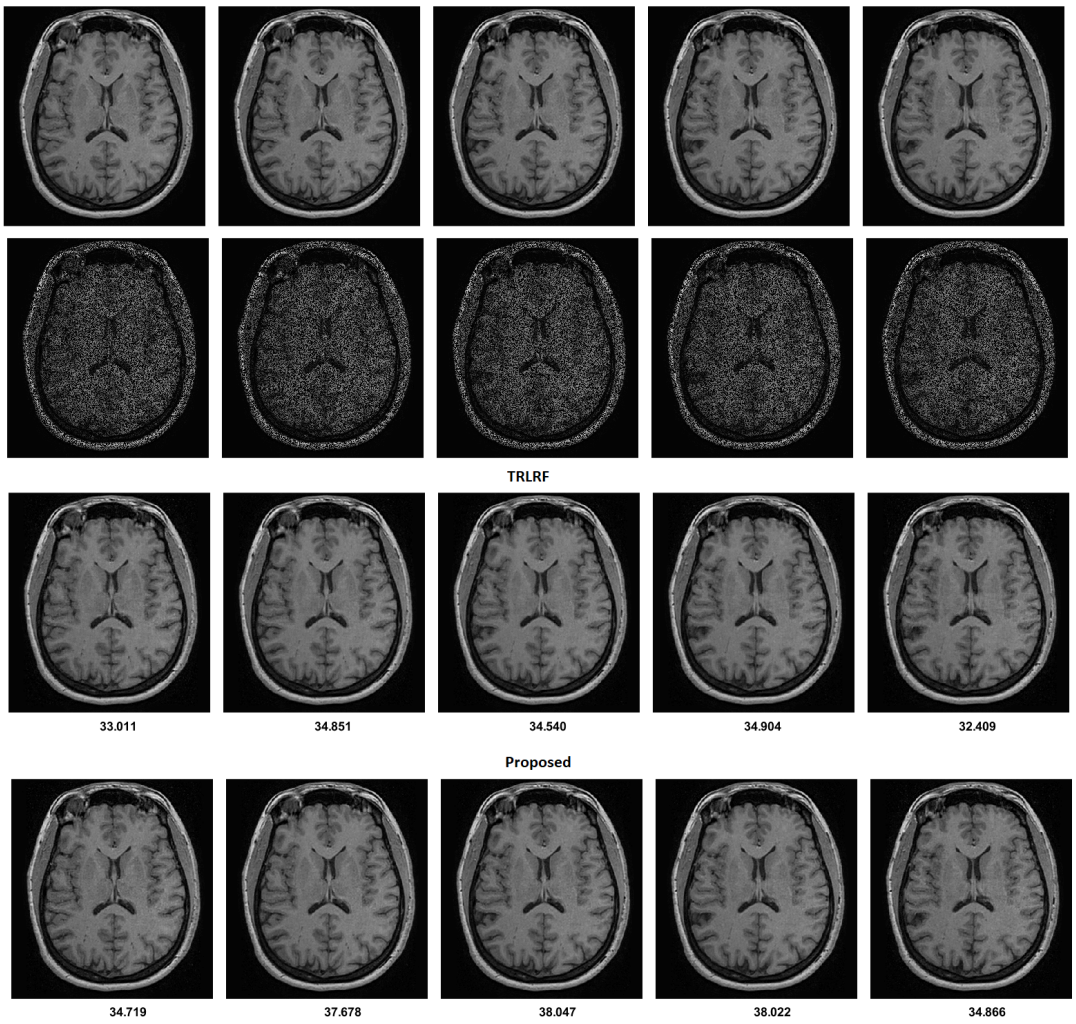
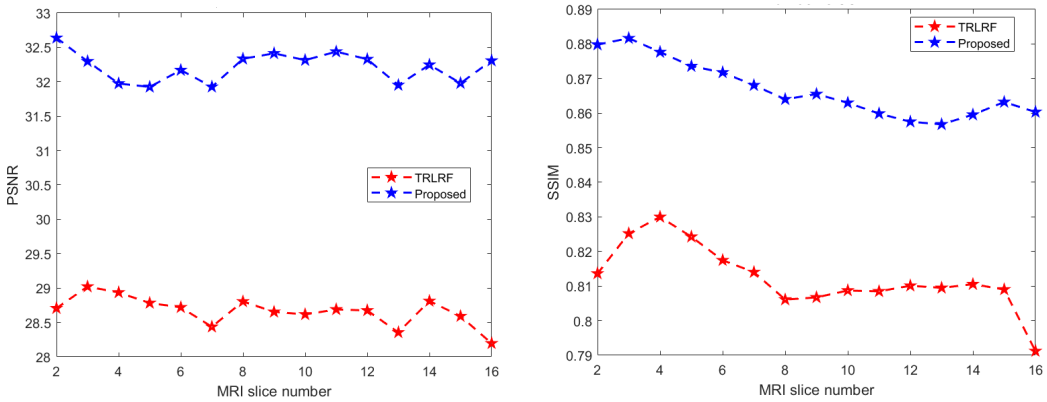


Figure 4-4: Visual comparison and PSNR results of the proposed method with TRLRF on MRI Dataset Two with 50% of random pixels missing

Image completion: The algorithm was also applied to natural color images of size $256 \times 256 \times 3$ using various missing scenarios. In this experiment we show that the proposed method also works for natural images not only biomedical MRI data. With the window size $\tau = (252, 252, 3)$, the incomplete images were first Hankelized into a 5th-order tensor of size $252 \times 5 \times 252 \times 5 \times 3$ and represented in TR format before applying the completion algorithms. Using the "Lena" image in the first simulation, we perform degradation by removing several rows and columns and then also randomly removing about 30% of the pixels. In another simulation, square holes were created in the image to render the image as corrupted. Then our tensor completion is perform to reconstruct these corrupted images. Figure 4-10



(a) PSNR comparison of MRI slices reconstruction

(b) SSIM comparison of MRI slices reconstruction

Figure 4-5: Comparison of PSNR and SSIM of proposed method with baseline method TRLRF on 50% of missing pixels

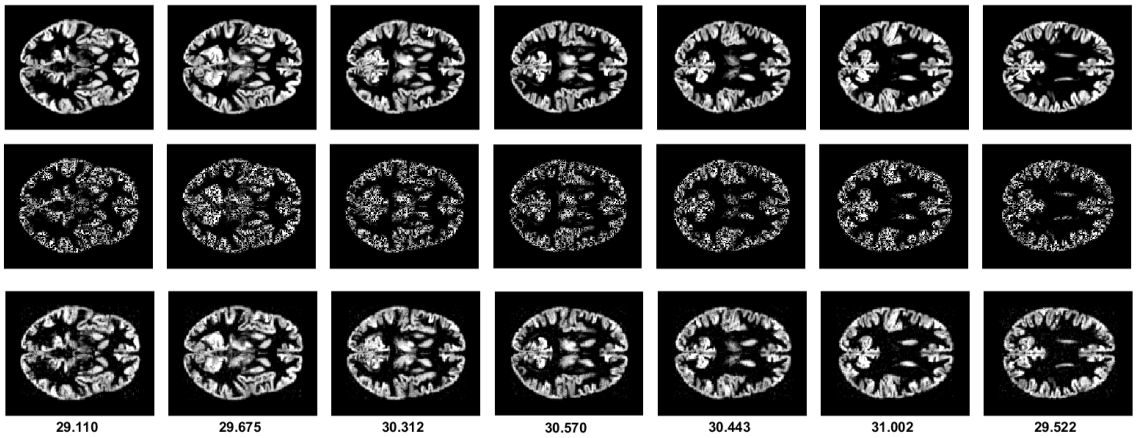


Figure 4-6: PSNR results on MRI Dataset One with 40% of random pixels missing

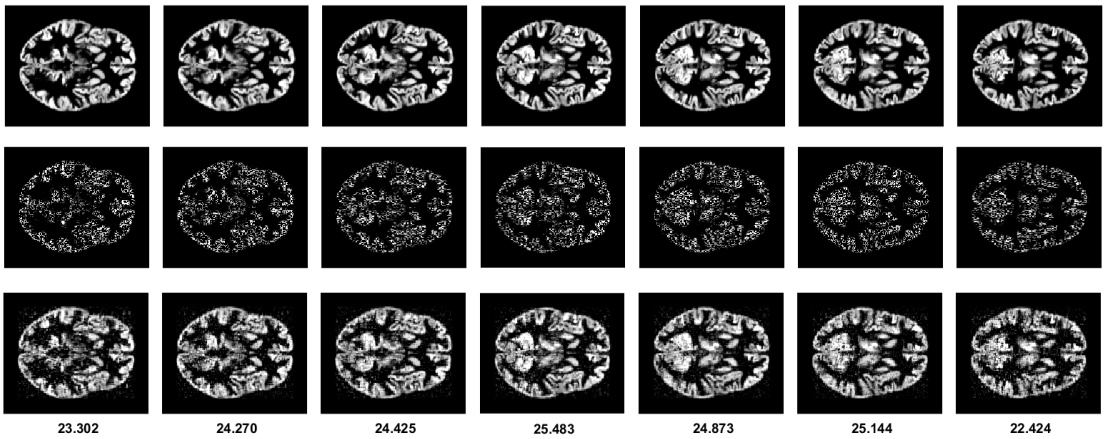


Figure 4-7: PSNR results on MRI Dataset One with 60% of random pixels missing

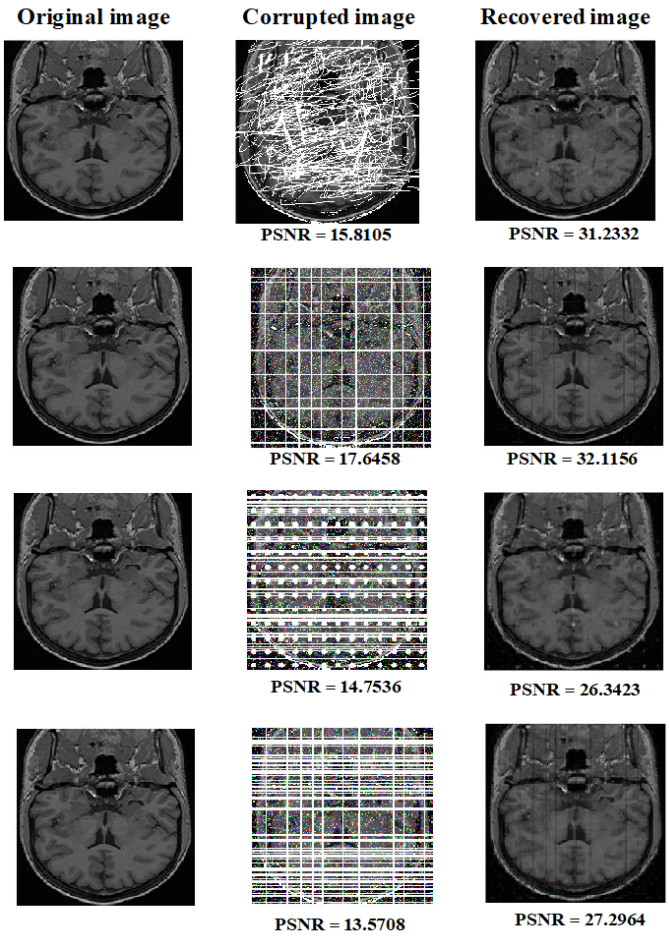


Figure 4-8: Performance of the proposed algorithm in recovering an MRI image with different types of structured missing pixels.

shows the PSNR and SSIM results of our method compared with other tensor ring completion methods. Finally in another experiment, round white holes were created in the "Lena" image both structurally and randomly. The PSNR results of our method compared with other tensor completion algorithms are given in 4-11.

The proposed method is examined in terms of robustness to TR rank selection. To this end, we considered an image with 90% random missing pixels and compared the RSE. The results are reported in Figure 4-12(a) which show that our proposed (TRDSR) algorithm achieves the lowest RSE compared to the other completion methods and also it is relatively insensitive to selected TR ranks. One of the main challenging task in tensor completion problem is a good selection of the rank because a larger rank does not always necessarily provides better results. This is mainly

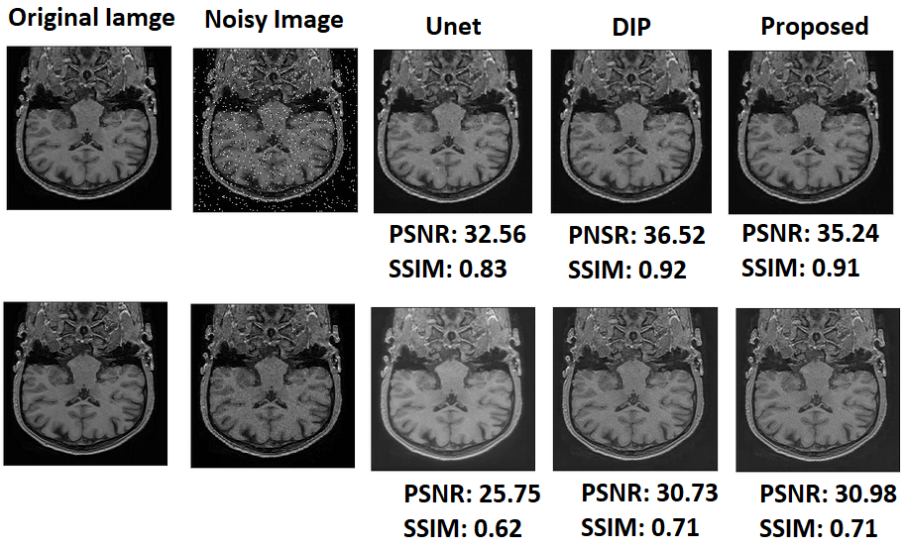


Figure 4-9: Performance of the proposed algorithm in recovering an MRI image with noisy artifacts. Top: Reconstruction of Noisy frame with Salt and Pepper. Bottom: Reconstruction of Noisy Frame with Poison Noise.

because of the over-fitting problem where the model is not selected appropriately. Because of this issue, it is of interest to have a tensor completion algorithm which is less sensitive to rank selection. In these simulations, we show that different from other algorithms, ours is less sensitive to the choice of TR-rank and it is more stable to various TR-rank. The running time of our method is also compared with other existing methods. The results are reported in Figure 4-12(b). It is seen that our algorithm is the fastest algorithm while at the same time, it provides better performance. Table 4.1 also shows the PSNR and SSIM results obtained from the reconstruction of various corrupted scenarios using our method compared with other tensor completion algorithms on other images.

Time series data completion: The performance of the proposed algorithm is compared with the state of the art MDT [9] and SSA [212] algorithms. In figure 4-13, we show the reconstruction of time series data. In the first sub-figure, the performance of the method is compared with other algorithms where parts of the signals are removed. Furthermore, the last 30 and 60 samples of the time series have also been taken off. It should be noted that these incomplete time series were noise free that is no noise was added to them. Only the last 500 elements of the signals are shown for better visualization. The results confirmed that the

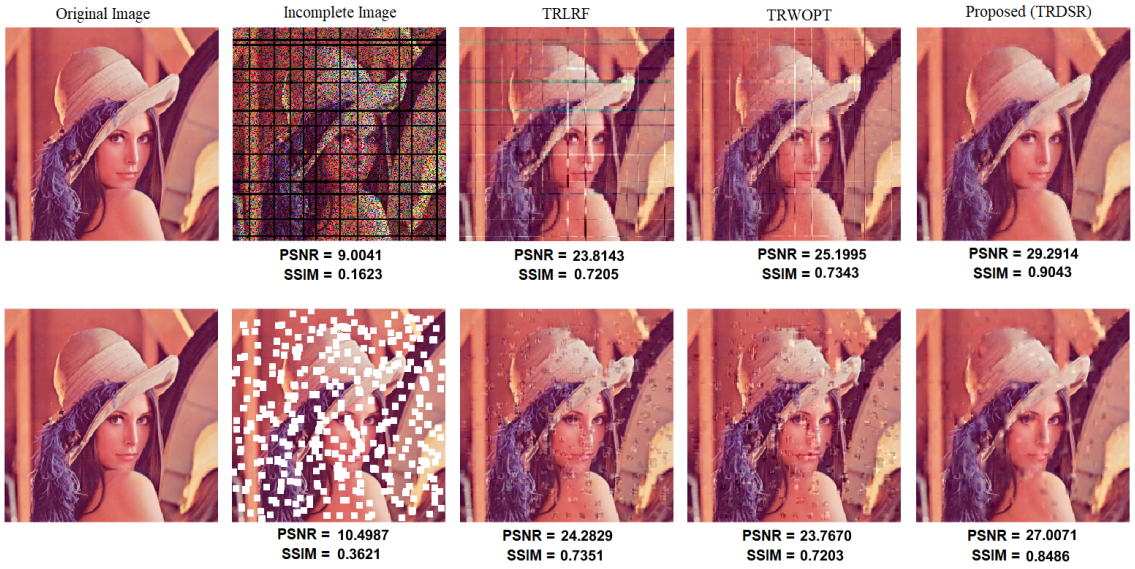
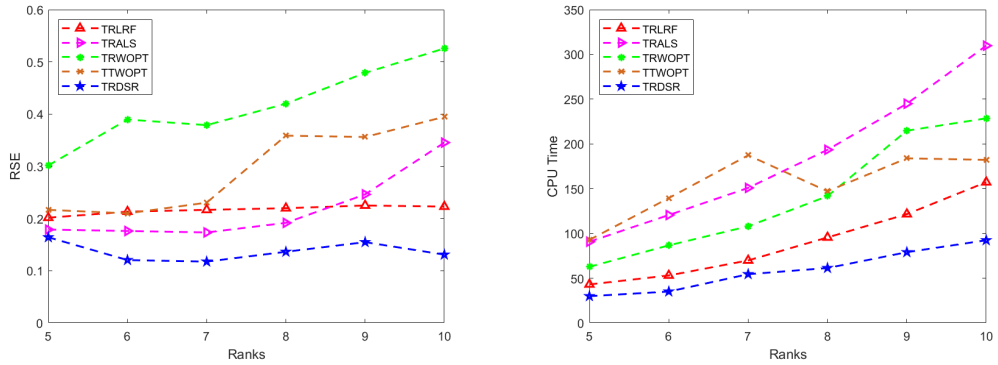


Figure 4-10: Performance of the proposed algorithm in recovering an structural missing pixels on "Lena" image

Figure 4-11: Performance of the proposed algorithm in recovering structural missing pixels on "Lena" image

proposed method also has ability to forecast future samples. The second sub-figure corresponds reconstruction of artificial time series and speech signal, respectively. Hankelization has been applied for using MDT. The window size (τ) is equal to 100 for the first and 300 for the second row. The missing rate for the speech signal is 74% is whiles for artificial time series the rate is 53%. For evaluation, the Normalized Root Mean Square Error (NRMSE) was utilized and shown beneath each figure in brackets. The results of the reconstruction and evaluation confirmed the better performance of the proposed approach.



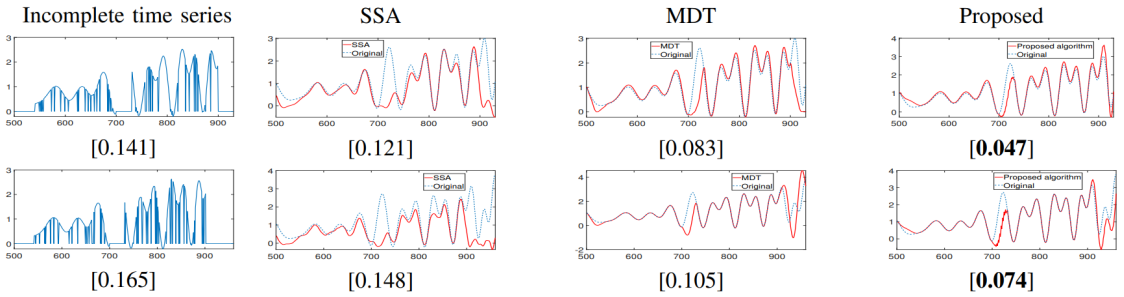
(a) Comparison of RSE vs TR Ranks R

(b) Comparison of CPU Time vs TR Ranks (in seconds)

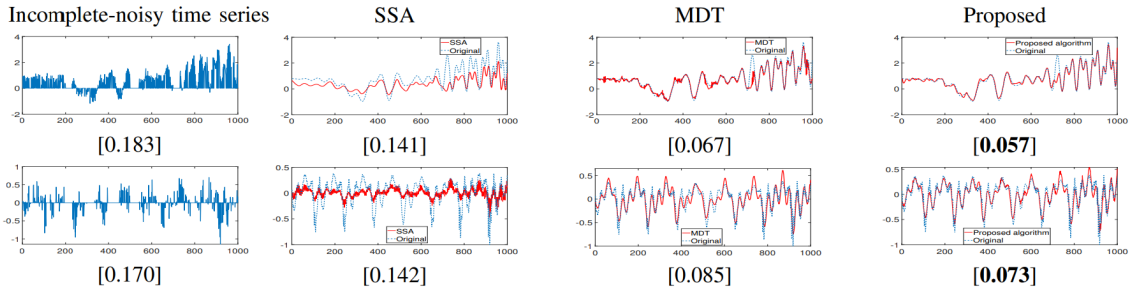
Figure 4-12: Comparison of RSE and CPU time with TR Ranks of some tensor completion algorithms with 90% missing components.

Table 4.1: Performance comparison using PSNR and SSIM for natural Images.

IMAGE	Results	TRLRF	MDT	TRALS	TRWOPT	TTWOPT	CPWOPT	SPC-TV	Proposed (TRDSR)
Lena	PSNR	29.63	27.65	30.19	28.39	23.80	21.45	27.11	**31.49**
	SSIM	0.9156	0.8233	0.9024	0.9048	0.8130	0.6057	0.9049	**0.9422**
Barbara	PSNR	18.76	19.34	19.59	10.63	17.53	10.71	16.96	**24.12**
Lines	SSIM	0.5251	0.4700	0.7300	0.2525	0.4619	0.1242	0.5199	**0.7702**
House	PSNR	32.67	32.41	31.66	30.49	27.78	22.73	30.73	**33.95**
	SSIM	0.9186	0.8696	0.9050	0.8411	0.8601	0.6287	0.9075	**0.9353**
Barbara	PSNR	29.98	26.77	29.67	28.70	28.64	21.05	30.29	**31.84**
White Circles	SSIM	0.9200	0.7813	0.9177	0.8999	0.9126	0.5713	0.9279	**0.9456**
Occluded	PSNR	22.56	19.91	16.18	20.699	15.22	22.14	20.06	**23.24**
Windows	SSIM	0.7300	0.6258	0.5721	0.6560	0.5732	0.6704	0.7041	**0.7425**
Lena	PSNR	32.89	32.03	33.95	32.18	30.00	21.73	34.34	**36.71**
Scratched	SSIM	0.9159	0.9236	0.9390	0.9011	0.8932	0.6090	0.9513	**0.9677**
Peppers	PSNR	17.85	20.11	21.33	20.52	13.88	16.50	20.06	**22.19**
90% Missing	SSIM	0.2772	0.5619	0.5056	0.4689	0.1569	0.2617	**0.6370**	0.5506
Giant	PSNR	16.71	19.58	18.35	18.37	13.48	16.38	19.09	**20.32**
90% Missing	SSIM	0.3110	0.4631	0.3986	0.3756	0.1550	0.2531	0.4797	**0.5272**



(a) Time series reconstruction with several last entries of the original time series removed. First row: the last 30 entries removed Second row: the last 60 elements of the time series removed



(b) Time series reconstruction from incomplete and noisy time series. First row: artificial time series reconstruction second row: reconstruction of speech signal

Figure 4-13: Time series Reconstruction

4.4 Experiments on EEG Data

- EEG dataset One: EEG recordings in which artifact-free EEG signals are contaminated with ocular artifacts artificially. The significance of this dataset is that it contains the pre-contamination EEG signals, allowing the brain signals underlying the EOG artifacts to be identified and the performance of each artifact rejection technique to be objectively evaluated. Using a realistic contamination model, the first EEG dataset was artificially contaminated with EOG artifacts. It is only concerned with EOG artifacts and employs a realistic model for the contamination of artifact-free EEGs rather than a random procedure [213]. The size of the EEG data in this set is of size 19×5600.

- EEG dataset Two: Multichannel EEG signal from the "NUI Maynooth EEG" dataset. The data was recorded according to the method by Sweeney *et. al.* [214] and used in the paper [215]. (Sampling Frequencies : 2048 Hz) and (Accelerometer : 200 Hz). Each EEG trial is of size 11×5600. The whole set consist of 23 trials.

- The EEG dataset Three: The Laboratory for Advanced Brain Signal Processing, BSI-RIKEN, Japan, provided the EEG recordings in collaboration with Shanghai Jiao Tong University, China. This dataset included 5 healthy subjects. The cue-based BCI paradigm included two motor imagery tasks: left hand (LH) and right hand (RH) movement imagination (RH). We used data from session 6 of the user A (i.e.,dataset SubA_s6). The EEG signals were recorded using g.tec (g.USBamp). The EEG signals were bandpass filtered between 2Hz and 30Hz at a sample rate of 256Hz, followed by a 50Hz notch filter. The size of the EEG data is $6 \times 768 \times 150$.

4.4.1 EEG data completion

The experiments in this section are designed to assess EEG completion as a method of dealing with missing or corrupted samples. Experiments are conducted using real EEG tensor datasets. The data is then artificially corrupted with missing entries.

That is, a mask is used to generate missing data in the recordings. The missing entries in the time series are assumed to be continuous by default. The first step is to tensorize any original 2 dimensional EEG signals into a 3 dimensional EEG tensor. EEG signals can be represented as a 3-dimensional tensor of shape channel $\times$ time $\times$ trials. Secondly, we simulate an incomplete EEG data by performing an operation to create missing entries using approaches adopted in [11, 216]. The selection of missing entries are performed in two ways:

1. Select missing data entries randomly. This mask has no predefined structure [65, 216]. See Figure 4-14(a) for illustration

2. Selecting data with missing channels in random trials. This scenario is more realistic in BCI applications. This can happen, for example, if an electrode has an incorrect impedance value or becomes completely detached during the experiment for any reason [11]. Refer to Figure 4-14(b) for illustration.

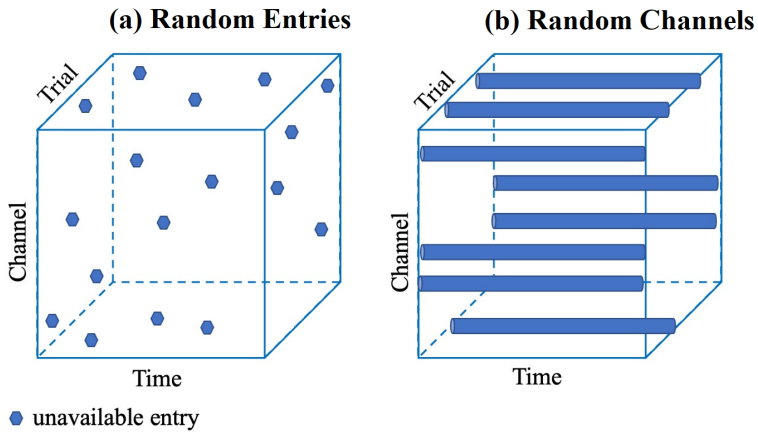


Figure 4-14: EEG missing data scenarios [11]

Simulating missing entries randomly: To generate random missing entries in $\underline{\mathbf{X}}$. A binary mask $\underline{\mathbf{Q}}$ is defined. Each entry in the mask $\underline{\mathbf{Q}}$ is either 0 or 1. Indexes with values corresponding to 1 will be preserved whiles index values with 0 are deleted. Therefore, the incomplete tensor is $\underline{\mathbf{Y}} = \underline{\mathbf{X}} * \underline{\mathbf{Q}}$, where $*$ denotes the element-wise product. The tensor completion algorithm is then applied on $\underline{\mathbf{Y}}$ to recover the information of the missing entries. Figure 4-15 provides an illustration of generating incomplete data with random entries.

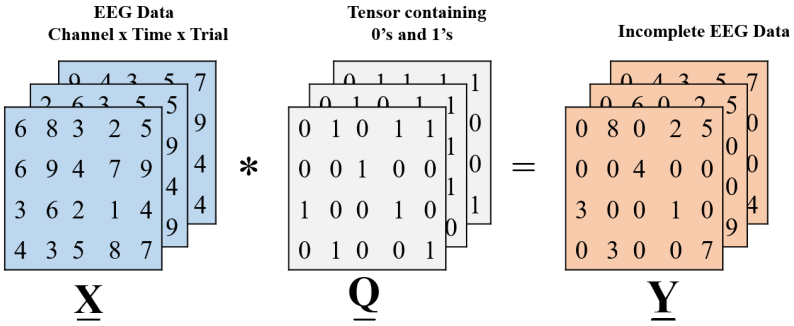


Figure 4-15: Generating incomplete EEG data with random entries

Experiments with missing or corrupted data

To evaluate the performance the tensor completion algorithm on EEG data. We compare with other tensor completion algorithms. The methods used are TRLRF [13], MDT [9], SPC [49], TRAR[51], CPWOPT [29], TRALS [46] and TRWOPT [32]. The algorithms were chosen because they represent various solutions to the tensor completion problem. We performed Hankelization for our TRDSR and MDT. The data from set one and two was reshaped into 3rd order tensors of size $140 \times 40 \times 19$ and $140 \times 40 \times 11$ respectively. Then, we perform Hankel folding using window size $\tau = (120, 35, 7)$ to generate a 6th-order tensor of size $120 \times 21 \times 35 \times 6 \times 7 \times 13$ and $120 \times 21 \times 35 \times 6 \times 7 \times 5$. The incomplete EEG from set three was Hankelized into an 5th-order tensor of size $760 \times 9 \times 154 \times 6 \times 6$ using the window sizes $\tau = (760, 150, 6)$. It should be noted that the τ values can be changed or a much smaller window size can be used however, we used this window size because of memory constraints. For EEG simulation, ranks between 8 and 20 was used for all methods that required rank tuning. MDT [9] and SPC [49] does not require rank setting. All other parameters were set according to method specifications. To compare the correctness of the proposed tensor completion algorithms, the NRMSE of the original and reconstructed error is computed computed. The NRMSE was calculated based on all the entries of the data.

EEG Experiment 1: Simulations are performed using different amounts of missing samples. Specifically, we considered 1%, 5%, 10%, 15%, 20%, 30% and 40% of missing samples on the whole EEG data tensor from dataset three. The missing entries are by generated randomly along all the channels in the EEG tensor without

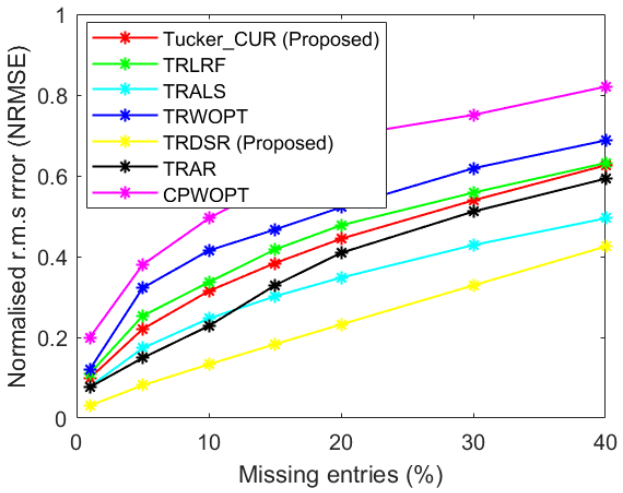


Figure 4-16: NRMSE results on different missing entries of EEG data

any restriction about the structure of the missed values (as shown in Figure 4-14a and Figure 4-15).

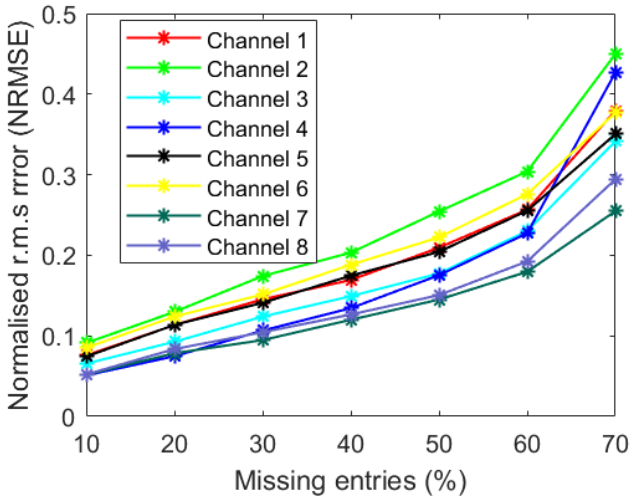


Figure 4-17: NRMSE results on different missing ratios using selected channels of EEG data

Despite the fact that this is not a more realistic scenario, the results show that all tensor completion algorithms can recover missing entries of EEG data with high accuracy. We report the NRMSE values obtained in Figure 4-16. Similarly, we report channel by channel NRMSE results of EEG data from dataset one with random missing ratio from 10% to 70% using our proposed TRDSR algorithm. Eight (8) channels were selected from the reconstruction and results displayed. Results are

shown in Figure 4-17.

Figure 4-18 also present results on selected trials from the EEG dataset three. The data was randomly corrupted with missing ratios from 10% to 80%. The tensor completion was performed on each trail data of size 11×5600. The results show that the method is able to perform similarly across all trails in the data set. In general,

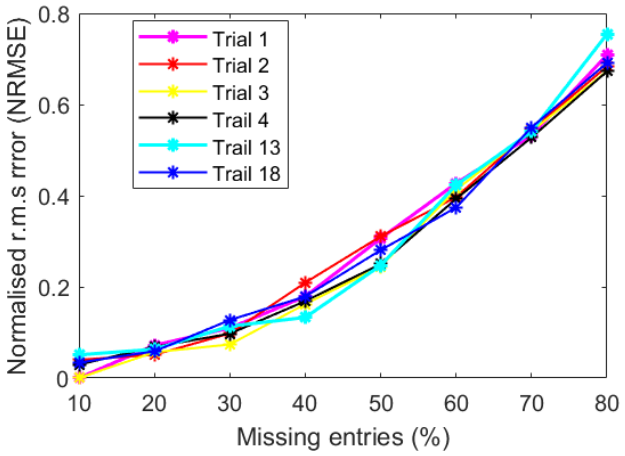


Figure 4-18: NRMSE results on different missing ratios using selected trials from dataset two

from these results it can be concluded that for the case of having random missing entries the reconstruction is more difficult especially as the missing ratio increases as can be seen with the high values of the error.

EEG Experiment 2: The second type of simulation include experiments for a more realistic scenario in which complete trials are corrupted or missing. Our results show that the proposed tensor completion algorithms are still applicable in more difficult scenarios. Figure 4-19 compares the NRMSE of our proposed TRDSR with SPC, MDT, TRAR algorithms which provided the best results. In Figure 4-22, the reconstructed results and spectrum are shown for a selected part of the EEG tensor. Figure 4-22(a), shows the reconstruction of the time series with missing data. The missing entries are contained within the two red lines in the figure with approximately 100 samples removed. Figure 4-22(b) shows the spectrograms of these reconstructed time series for further comparison and easier analysis.

EEG Experiment 3: Similarly, Figure 4-20 presents visual results of reconstructed signals using the tensor completion algorithm after removing artifact with

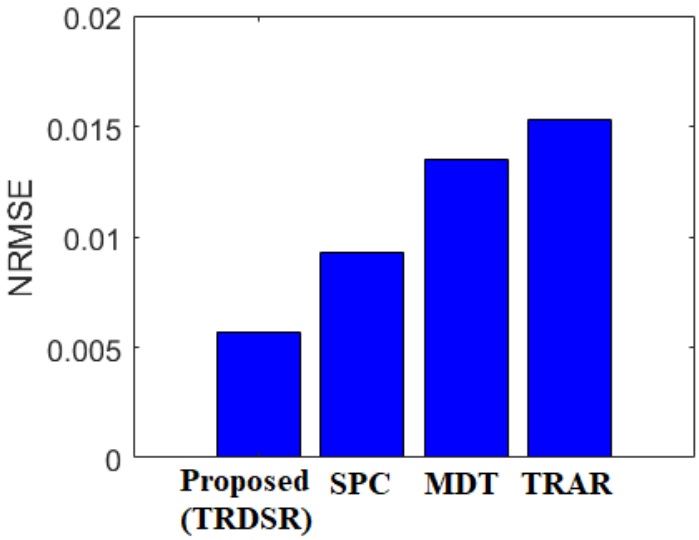


Figure 4-19: NRMSE results on EEG data with some channels removed

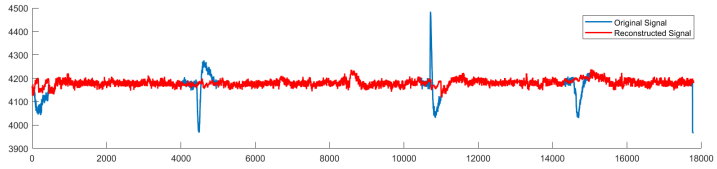


Figure 4-20: Single channel EEG data reconstruction after artifact detection

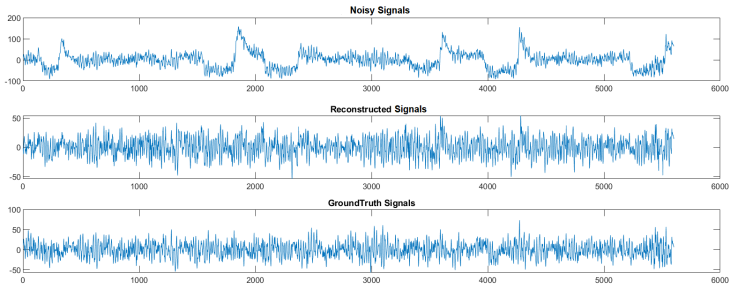


Figure 4-21: EEG data reconstruction after artifact detection

high frequency. We consider real EEG data with artifacts where the affected part is removed and attempt to reconstruct the removed section using our proposed TRDSR Algorithm. Figure 4-21 also shows the reconstruction results of one channel in a multi-channel EEG data with artifact correction. The corrupted data is taken from EEG dataset one.

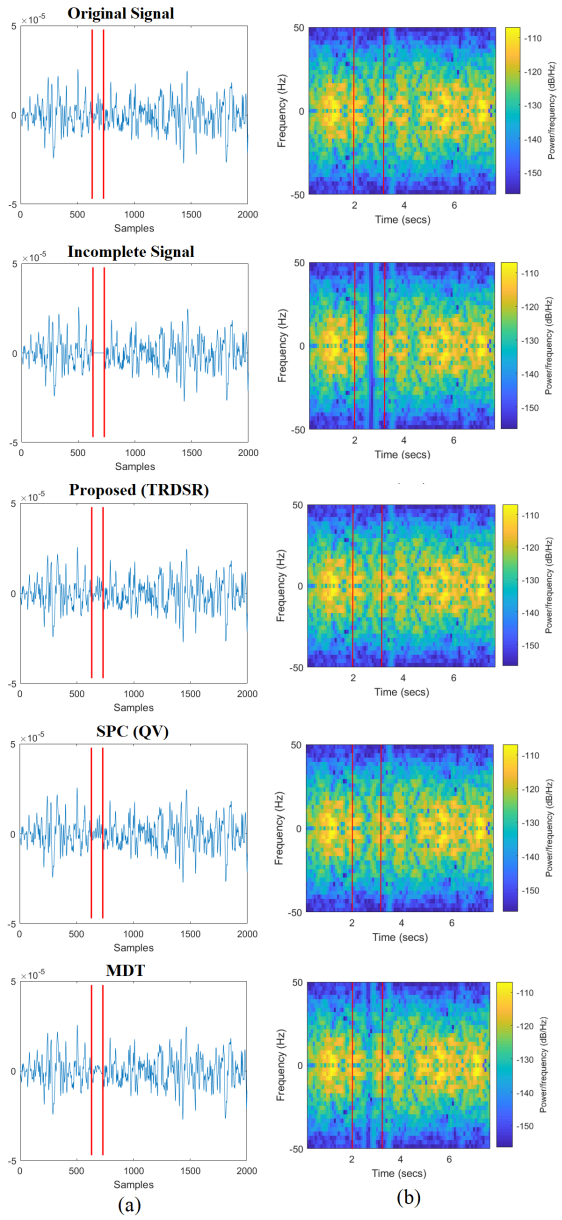


Figure 4-22: Spectrogram comparison on the missing entries of an EEG tensor with 6 Channels. (a) Completion effects in the time domain; (b) completion effects in frequency domain

Chapter 5

Cross tensor approximation exploiting smoothness

This chapter introduces us to the concept of cross approximation or CUR and its application to tensor completion. The ColUmn-Row (CUR) or cross-skeleton method was introduced to the field of numerical linear/multi-linear algebra for computation of fast low-rank matrix/tensor approximation using large-scale data matrices or tensors. A CUR matrix approximation is a set of three matrices (C, U, R) that, when multiplied together, closely approximate a given matrix. The main advantages for adopting CUR algorithms are:

- Fast low tensor rank approximation,
- Data interpretation concerns,
- Higher compression ratio.

In addition to the above advantages, for the case of matrices, the SVD of sparse matrices does not provide sparse factor matrices, whereas the cross-skeleton approximation does, resulting in a more compact data representation. Although CUR approaches are commonly employed for low-rank matrix/tensor approximation and compression, we are employing them to solve the completeness problem. The methods by [217] and [218] are the only algorithms that apply the CUR approximation approaches for the completion job. These methods, on the other hand, are purely

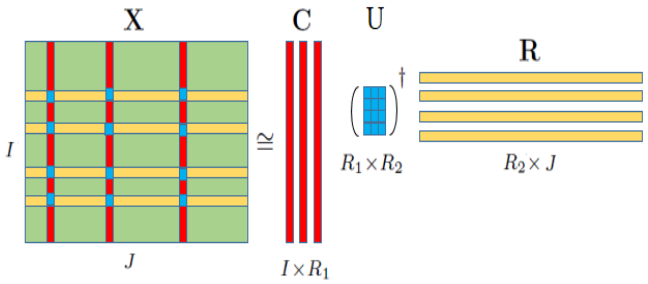


Figure 5-1: Illustration of cross decomposition for low-rank matrix approximation $\mathbf{X} \cong \mathbf{CUR}$, where $\mathbf{U} = \mathbf{W}^{+}$ [12].

theoretical and cover more complicated procedures. The method proposed in this research provides a simple and easy solution to implement. It can be implemented in few lines of code. Also, because they utilise CUR methods, they have a lower computational complexity than other completion techniques. This method is the first one using the tensor CUR algorithm developed in [219] and [220], for the tensor completion task.

5.1 Cross Matrix Approximation (CMA) and Matrix Column Selection

The skeleton or cross approximation method of computing a low-rank approximation of a given matrix based on a portion of its individual columns and rows was first developed in [221]. This framework is known as cross approximation because it uses a matrix that intersects the columns and rows during the approximation phase. The selection of a column or row can be done in a randomized or deterministic manner. Algorithms based on The Maxvol [222, 223], Cross2D [224, 225] and Discrete Empirical Interpolatory Method (DEIM) [226, 227] are known to have such selections. If the columns or rows are selected randomly, this framework is often known as randomized CUR approximation. Compared with conventional algorithms, such as SVD, the CUR approach requires less memory usage and floating-point operations. It can also preserve the structure of the original data matrix, such as non-negativity, sparsity, and smoothness.

The CUR problem is formally formulated as follows: Let $\mathbf{X} \in \mathbb{R}^{I \times J}$ be a given

matrix and $\mathbf{C} \in \mathbb{R}^{I \times R_1}$, $\mathbf{R} \in \mathbb{R}^{R_2 \times J}$ are the selected columns and rows, respectively, and the intersection matrix is $\mathbf{W} \in \mathbb{R}^{R_1 \times R_2}$, see Figure 5-1. It's low-rank cross approximation is computed as:

$$\mathbf{X} \cong \mathbf{CUR} = \mathbf{U} \times_1 \mathbf{C} \times_2 \mathbf{R}^T, \tag{5.1}$$

where $\mathbf{U} \in \mathbb{R}^{R_1 \times R_2}$ should be computed to yield the smallest error. The best middle matrix $\mathbf{U}$ in the least-squares sense is $\mathbf{U} = \mathbf{C}^+\mathbf{X}\mathbf{R}^+$ because

$$\mathbf{C}^\dagger \mathbf{X} \mathbf{R}^\dagger = \underset{\mathbf{U} \in \mathbb{R}^{R1 \times R2}}{\arg\min} \; \|\mathbf{X} - \mathbf{U} \times_1 \mathbf{C} \times_2 \mathbf{R}\|_F.$$

The approximation computed in the above formulation is exact if $\text{rank}\,(\mathbf{X}) \leq \min\{R_1, R_2\}$ [221].

In order to compute the middle matrix in formulation (5.1), the whole matrix $\mathbf{X}$ is needed. This approach can be simplified by using the Moore-Penrose of the intersection matrix $\mathbf{W}$ and computing the approximation $\mathbf{X} \cong \mathbf{C}\mathbf{W}^\dagger\mathbf{R}$. However, there are instability issues when computing the Moore-Penrose pseudoinverse of such a matrix especially when the matrix is ill-conditioned. Due to this, a well conditioned intersection matrix should be selected. It is also known that the quality of this intersection matrix depends on the module of the determinant called *matrix volume*. To be precise, a set of columns and rows should be selected whose intersection matrix has as much volume as possible. This clearly is an NP-hard problem since we need to check the volume of all possible intersection matrices produced by different selections of columns on rows. However, algorithms implemented in [228, 229, 230, 231] provide a heuristic approach for computing suboptimal solutions. If a given matrix $\mathbf{X}$ is positive semi-definite, then the CUR approximation is called Nyström method, i.e., $\mathbf{C} = \mathbf{R}$, and has several applications in machine learning and data science [232, 233, 234, 235]. In this research, this variant is not considered as the images/videos do not have symmetric structures.

A special case of the CMA where only columns are sampled is called matrix column selection, *interpolative matrix decompositions*, and in some contexts, it is also referred to as *CY decomposition*. This problem is known as column (feature)

selection in the field of machine learning and data analysis [236, 237, 238, 239, 240, 241, 242]. To illustrate, let $\mathbf{X} \in \mathbb{R}^{I\times J}$ and $\mathbf{C} \in \mathbb{R}^{I\times R}$ be the number of selected columns. Then the matrix column selection is formulated as follows:

$$\mathbf{X} \cong \mathbf{C}\mathbf{Y}, \tag{5.2}$$

where $\mathbf{C} \in \mathbb{R}^{I\times R}$ is a matrix containing the selected columns and $\mathbf{Y} \in \mathbb{R}^{R\times J}$ should be computed in such a way that the approximate error should be as small as possible. The best solution to the problem (5.2) in the least-squares (LS) sense is $\mathbf{Y} = \mathbf{C}^{+}\mathbf{X}$, and if $\text{rank}(\mathbf{X}) = R$, then the approximation is exact, i.e., $\mathbf{X} = \mathbf{C}\mathbf{C}^{+}\mathbf{X}$.

5.2 Cross Tensor Approximation(CTA)

CTA is a fast low-rank tensor approximation method that is a generalization of cross/skeleton matrix and CUR matrix approximation. The Skeleton or Cross approximation (CMA) computes a low-rank approximation using a portion of individual columns and rows. Deep learning methods [243], signal processing [244, 245], scientific computing [246, 247, 248] and machine learning [219, 249, 250] have all applied these methods in some applications. The next subsequent sections, explains how the CMA can be generalized to the tensor case. In general, there are three main categories under which the CTA techniques are generalized to tensors:

- Fiber selection,
- Fiber-Slice selection,
- Slice selection,

The first category is concerned with the Tucker decomposition while the last one is related to the tubal SVD. The second category is a different model and will be discussed in detail later.

5.2.1 CTA based on fiber selection

A straightforward generalization of the CMA to tensors are proposed in the works by [251, 252], and [253]. Similar to the CMA where parts of columns and rows of a given matrix are sampled, a set of fibers (along different modes) are selected, and the goal is to compute a Tucker approximation based on these sampled fibers. For example, for 3rd-order tensors, columns, rows, and tubes are sampled. The method proposed in [220] adopts a randomization approach, while those proposed in [252, 253] are deterministic. These methods are considered to be the starting points of generalizing CMA to tensors.

For noiseless data tensor, the existence of an exact Tucker model whose factor matrices are taken from the fibers of the original data tensor is obvious. To be more precise, let $\underline{\mathbf{X}}$ be an Nth-order tensor of size $I_1 \times I_2 \times \cdots \times I_N$ and of Tucker rank $(R_1, R_2, \ldots, R_N)$. Now, if we generate any full-rank factor matrices $\mathbf{A}_n \in \mathbb{R}^{I_n \times R_n}$, $n = 1, 2, \ldots, N$, by sampling fibers in each mode and computing the core tensor as

$$\underline{\mathbf{S}} = \underline{\mathbf{X}} \times_1 \mathbf{A}_1^+ \times_2 \mathbf{A}_2^+ \cdots \times_N \mathbf{A}_N^+ \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}, \tag{5.3}$$

then the obtained Tucker decomposition has the exact Tucker rank $(R_1, R_2, \ldots, R_N)$. So an exact Tucker decomposition of the tensor $\underline{\mathbf{X}}$ whose factor matrices are taken from the original data tensor is computed. For noisy tensors, similar to the CMA, the accuracy of approximation quite depends on the number of selected fibers and also the list of sampled fibers. The idea of sampling fibers and considering them as the factor matrices, first was proposed in [220]. In the first step, the factor matrices are generated, after which the core tensor is computed through (5.3) as described above. This is summarized in Algorithm 9.

The Fast Sampling Tucker Decomposition (FSTD) algorithm [253] is another approach for CTA. In this method, first an intersection subtensor $\underline{\mathbf{W}}$ is produced by sampling some indices in different modes and then the fibers are selected. In a more precise way, let $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ be a given 3rd-order tensor and $\mathcal{I}_1 \subseteq I_1$, $\mathcal{I}_2 \subseteq I_2$, $\mathcal{I}_3 \subseteq I_3$, be subsets of indices I_1, I_2, I_3 where, $|\mathcal{I}| = P_1$, $|\mathcal{I}_2| = P_2$, $|\mathcal{I}_3| = P_3$

Algorithm 9: Tucker CUR Algorithm [220]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, positive integer numbers $R_n,\ n = 1, 2, \ldots, N$

Output: Tucker approximation $\underline{\mathbf{X}} \cong [\![\underline{\mathbf{S}}; \mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N]\!]$

1 **for** $n = 1, 2, \ldots, N$ **do**

2 Sample R_n fibers in n-th mode and generate approximate factor matrix $\mathbf{A}_n \in \mathbb{R}^{I_n \times R_n}$

3 **end**

4 Compute the core tensor $\underline{\mathbf{S}} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}$ as in (5.19)

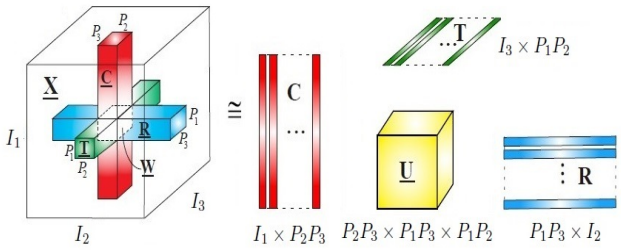


Figure 5-2: Illustration of the CTA (fiber selection version) for a 3rd-order low-rank tensor. For simplicity of presentation, we assume that all fibers build up to block sub-tensors, [12].

and the core intersection subtensor is denoted by $\underline{\mathbf{W}} \in \mathbb{R}^{P_1 \times P_2 \times P_3}$.

The question is how to compute an approximate Tucker decomposition for the tensor $\underline{\mathbf{X}}$ based on the intersection subtensor $\underline{\mathbf{W}}$?. Motivated by the fact that

$$\mathbf{U} = \mathbf{W} \times_1 \mathbf{W}_{(1)}^{+} \times_2 \mathbf{W}_{(2)}^{+} = \mathbf{W}^{+}\mathbf{W}\mathbf{W}^{+} = \mathbf{W}^{+}, \tag{5.4}$$

which is used as the middle matrix in the CMA, it is proposed in [253] to compute

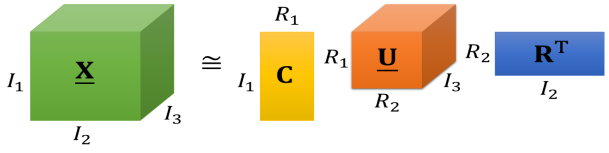


Figure 5-3: llustration of the Tucker-2 CUR approximation.

the approximate core tensor in the Tucker decomposition as

$$\begin{aligned} \underline{\mathbf{U}} &= \underline{\mathbf{W}} \times_1 \mathbf{W}_{(1)}^{+} \times_2 \mathbf{W}_{(2)}^{+} \times_3 \mathbf{W}_{(3)}^{+} \\ &\equiv \left[\!\left[\underline{\mathbf{W}}; \mathbf{W}_{(1)}^{+}, \mathbf{W}_{(2)}^{+}, \mathbf{W}_{(3)}^{+} \right]\!\right]. \end{aligned} \tag{5.5}$$

This is a direct generalization of (5.4) to 3rd-order tensors. In view of (5.5), the core tensor $\underline{\mathbf{U}}$ of the Tucker approximation is of size $P_2P_3 \times P_1P_3 \times P_1P_2$ and as a result, P_2P_3 columns, P_1P_3 rows, and P_1P_2 tubes should be sampled. The selection has to be done in an appropriate way, see Figure 5-2 for details. It is shown in [253] that the corresponding factor matrices $\mathbf{A}_1 \in \mathbb{R}^{I_1 \times P_2P_3}$, $\mathbf{A}_2 \in \mathbb{R}^{I_2 \times P_1P_3}$, $\mathbf{A}_3 \in \mathbb{R}^{I_3 \times P_1P_2}$ are the subsampled matrices from the unfolding matrices $\mathbf{X}_{(1)}(:, \mathcal{I}_2, \mathcal{I}_3)$, $\mathbf{X}_{(2)}(\mathcal{I}_1, :, \mathcal{I}_3)$ and $\mathbf{X}_{(3)}(\mathcal{I}_1, \mathcal{I}_2, :)$ respectively and CTA approximation can be found as

$$\begin{aligned} \underline{\mathbf{X}} &\cong [\![\underline{\mathbf{U}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!] \\ &\equiv \left[\!\left[\underline{\mathbf{W}}; \underbrace{\mathbf{A}_1\mathbf{W}_{(1)}^{+}}_{\widetilde{\mathbf{C}}_1}, \underbrace{\mathbf{A}_2\mathbf{W}_{(2)}^{+}}_{\widetilde{\mathbf{C}}_2}, \underbrace{\mathbf{A}_3\mathbf{W}_{(3)}^{+}}_{\widetilde{\mathbf{C}}_3} \right]\!\right]. \end{aligned} \tag{5.6}$$

The procedure of this approach is summarized as follows

- Consider indices $\mathcal{I}_n \in [I_n]$, $n = 1, 2, 3$ and produce the intersection subtensor $\underline{\mathbf{W}}$ and corresponding sampled columns, rows, and tubes $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3$.
- Compute the Tucker approximation (5.6).

The algorithm is referred to as Fast Sampling Tucker Decomposition (FSTD) and summarized in Algorithm 10 [253]. The work by [254] provides a similar approach to this method where the similarity lies in the number of selected fibers for each mode.

5.2.2 CTA Based on Slice-Fiber Selection

The work by [255] provides an alternate approach to CTA. The idea is motivated by some applications used in hyperspectral imaging, medical image analysis and consumer recommender system analysis. This is because for most of these data, one

Algorithm 10: Fast Sampling Tucker Decomposition (FSTD) algorithm for 3rd-order tensors [253]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, indices $\mathcal{I}_n \subseteq [I_n],\ n = 1, 2, 3$
Output: Tucker approximation $\underline{\mathbf{X}} \cong [\![\underline{\mathbf{U}}; \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3]\!]$
1 Generate the intersection subtensor $\underline{\mathbf{W}} = \underline{\mathbf{U}}(\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3)$
2 Generate the subsampled matrices $\mathbf{A}_1 = \mathbf{X}_{(1)}(:, \mathcal{I}_2, \mathcal{I}_3),\ \mathbf{A}_2 = \mathbf{X}_{(2)}(\mathcal{I}_1, :, \mathcal{I}_3)$ and $\mathbf{A}_3 = \mathbf{X}_{(3)}(\mathcal{I}_1, \mathcal{I}_2, :)$
3 $\underline{\mathbf{X}} \cong [\![\underline{\mathbf{W}}, \mathbf{A}_1\mathbf{W}_1^+, \mathbf{A}_2\mathbf{W}_2^+, \mathbf{A}_3\mathbf{W}_3^+]\!]$

of the modes is qualitatively different from the others. This section introduces a brief description of the idea for 3rd-order tensors. Let $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ be a given data tensor, and without loss of generality, we assume that the last mode is qualitatively different from the others. More precisely, the last mode has a larger dimension compared to the other modes, and some frontal slices and tubes are selected.

Given prior probability distributions for sampling frontal slices as $\{p_i\}_{i=1}^{I_3}$ and tubes as $\{q_j\}_{j=1}^{I_1 I_2}$, in the first step, some frontal slices, say L_1, are sampled and they are stored in $\underline{\mathbf{C}} \in \mathbb{R}^{I_1 \times I_2 \times L_1}$. In the second step, we sample some tubes, say $L_2 = R_1 R_2$, and store them in $\underline{\mathbf{R}} \in \mathbb{R}^{R_1 \times R_2 \times I_3}$, or a matrix $\mathbf{R} \in \mathbb{R}^{L_2 \times I_3}$ (see Figure 5-4 (a)).

The CTA is then defined as (see Figure 5-4 (b))

$$\underline{\mathbf{X}} \cong \underline{\mathbf{C}} \times_3 (\mathbf{U}\mathbf{R})^T, \tag{5.7}$$

where the tensor $\underline{\mathbf{C}} \in \mathbb{R}^{I_1 \times I_2 \times L_1}$ and the matrix $\mathbf{R} \in \mathbb{R}^{L_2 \times I_3}$ contain the sampled frontal slices and tubes, respectively. The matrix $\mathbf{U} \in \mathbb{R}^{L_1 \times L_2}$ is defined as

$$\mathbf{U} = \mathbf{D}_1(\mathbf{D}_2\mathbf{W}\mathbf{D}_1)^+\mathbf{D}_2 \in \mathbb{R}^{L_1 \times L_2},$$

$$\mathbf{W} = \text{reshape}(\underline{\mathbf{W}}, [L_2, L_1]),$$

where $\mathbf{D}_1 \in \mathbb{R}^{L_1 \times L_1}$ and $\mathbf{D}_2 \in \mathbb{R}^{L_2 \times L_2}$ are scaling diagonal matrices corresponding

to the slice and fiber sampling, respectively, and defined as follows

$$(\mathbf{D}_1)_{tt} = \frac{1}{\sqrt{L_1 p_{i_t}}}, \quad t = 1, 2, \ldots, L_1,$$
$$(\mathbf{D}_2)_{tt} = \frac{1}{\sqrt{L_2 q_{i_t}}}, \quad t = 1, 2, \ldots, L_2,$$

where $\{p_i\}_{i=1}^{I_3}$ are $\{q_j\}_{j=1}^{I_1 I_2}$ are probability distributions under which the frontal slices and fibers are sampled. This procedure is summarized in Algorithm 11. The length-squared probability distributions are defined as follows

$$\begin{aligned} p_i &= \frac{\|\underline{\mathbf{X}}(:,:,i_3)\|_F^2}{\|\underline{\mathbf{X}}\|_F^2}, \quad i_3 = 1, 2, \ldots, I_3, \\ q_j &= \frac{\underline{\mathbf{X}}(j_1, j_2, :)}{\|\underline{\mathbf{X}}\|_F^2}, \quad j_1, j_2 \in J_1, J_2. \end{aligned} \tag{5.8}$$

where J_1 and J_2 are subsets of the indices I_1 and I_2, are used in [255] for selecting the slices/tubes.

Remark 1. As discussed in [256], the model (5.7) can be considered as a special case of (5.6) with $\mathbf{A}_1 = \mathbf{W}_1$ and $\mathbf{A}_2 = \mathbf{W}_2$.

Algorithm 11: Slice-tube CUR algorithm [255]

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, a probability distribution $\{p_i\}_{i=1}^{I_3}$, a probability distribution $\{q_j\}_{j=1}^{I_1 I_2}$ and positive integers L_1 and L_2

Output: A tensor $\underline{\mathbf{C}}$ of size $I_1 \times I_2 \times L_1$, a matrix $\mathbf{U}$ of size $L_1 \times L_2$ and matrix $\mathbf{R}$ of size $L_2 \times I_3$

1 Select L_1 frontal slices of tensor $\underline{\mathbf{X}}$ i.i.d. trials according to $\{p_i\}_{i=1}^{I_3}$ and produce tensor $\underline{\mathbf{C}} \in \mathbb{R}^{I_1 \times I_2 \times L_1}$;

2 Generate diagonal scaling matrix D_1 of size $L_1 \times L_1$, where $(\mathbf{D}_1)_{tt} = \frac{1}{\sqrt{L_1 p_{i_t}}}$ for $t = 1, 2, \ldots, L_1$

3 Select L_2 tubes of tensor $\underline{\mathbf{X}}$ in L_2 i.i.d. trials according to $\{q_j\}_{j=1}^{I_1 I_2}$ and produce unfolding matrix $\mathbf{R} \in \mathbb{R}^{L_2 \times I_3}$

4 Generate diagonal scaling matrix of size $L_2 \times L_2$, where $(\mathbf{D}_2)_{tt} = \frac{1}{\sqrt{l_2 q_{j_t}}}$ for $t = 1, 2, \ldots, L_2$

5 Compute the 3rd-order tensor intersecting the sampled tubes and frontal slices as $\underline{\mathbf{W}} \in \mathbb{R}^{R_1 \times R_2 \times L_2}$, $R_1 R_2 = L_1$

6 Generate matrix $\mathbf{W} = \text{reshape}(\underline{\mathbf{W}}, L_2, L_1)$

7 Define matrix $\mathbf{U} = \mathbf{D}_1(\mathbf{D}_2\mathbf{W}\mathbf{D}_1)^{+}\mathbf{D}_2$

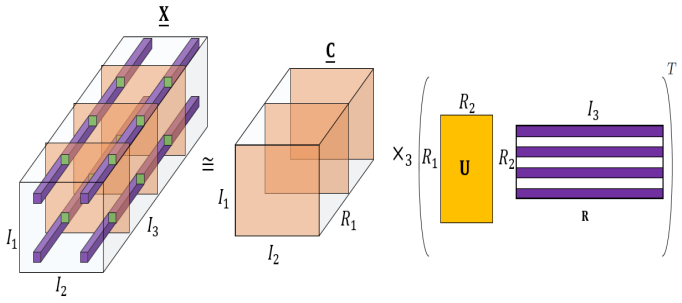


Figure 5-4: llustration of the CTA based on frontal slice and tube selection.

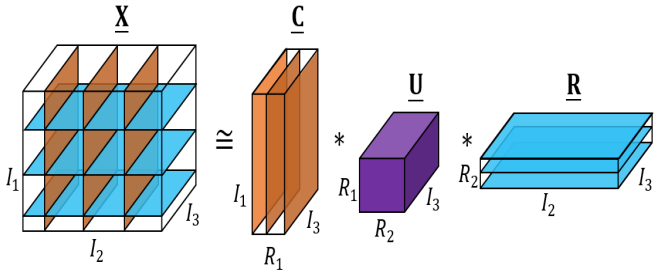


Figure 5-5: Illustration of the tubal CTA for a 3rd-order tensor.

5.2.3 CTA based on tubal product (t-product)

In this approach, the tensor variants of the CMA and matrix column selection are called the tubal CTA and lateral slice selection respectively. To be precise, let $\underline{\mathbf{X}}$ be a given 3rd-order tensor. The tubal cross approximation based on t-product is formulated as follows:

$$\underline{\mathbf{X}} \cong \underline{\mathbf{C}} * \underline{\mathbf{U}} * \underline{\mathbf{R}}, \tag{5.9}$$

where $*$ stands for the t-product [25], $\underline{\mathbf{C}} \in \mathbb{R}^{I_1 \times L_1 \times I_3}$ and $\underline{\mathbf{R}} \in \mathbb{R}^{L_2 \times I_2 \times I_3}$ are some sampled lateral and horizontal slices of the original tensor $\underline{\mathbf{X}}$ respectively and the middle tensor $\underline{\mathbf{U}} \in \mathbb{R}^{L_1 \times L_2 \times I_3}$ is computed in such a way that the approximation (5.9) should be as small as possible, see Figure 5-5 for graphical illustration concerning the tubal CTA.

Note that similar to other CTA algorithms discussed so far, the procedure of both lateral and horizontal slices sampling can be performed based on prior probability distributions[1]. The probability distributions used in [257] are uniform and

[1]Here different various probability distributions (with/without replacement) can be used but

nonuniform (length-squared and leverage scores) distributions. Let us first consider the tubal CTA. Similar to the CMA, the best solution for the middle tensor $\underline{\mathbf{U}}$ in the least-squares sense becomes :

$$\underline{\mathbf{U}} = \underline{\mathbf{C}}^{+} * \underline{\mathbf{X}} * \underline{\mathbf{R}}^{+}, \tag{5.10}$$

because

$$\underline{\mathbf{C}}^{+} * \underline{\mathbf{X}} * \underline{\mathbf{R}}^{+} = \underset{\underline{\mathbf{U}} \in \mathbb{R}^{L \times I_2 \times I_3}}{\operatorname{argmin}} \|\underline{\mathbf{X}} - \underline{\mathbf{C}} * \underline{\mathbf{U}} * \underline{\mathbf{U}}\|_F.$$

This is a straightforward generalization of the CMA [221] to tensors. Formula (5.10) can be computed in the Fourier domain and these computations are summarized in Algorithm 12. However, it is clear that (5.10) needs to pass the data tensor $\underline{\mathbf{X}}$ once again and this is of less practical interest for very large-scale data tensors, especially when the data tensors do not fit into the memory and communication between memory and disk is expensive [258]. To solve this problem, the MP pseudoinverse of the intersection subtensor $\underline{\mathbf{W}} \in \mathbb{R}^{L_2 \times L_1 \times I_3}$, which is obtained based on intersecting the sampled horizontal and lateral slices, should be approximated as

$$\underline{\mathbf{U}} = \underline{\mathbf{C}} * \underline{\mathbf{W}}^{+} * \underline{\mathbf{R}}.$$

It is not difficult to see that the tensor $\underline{\mathbf{W}}$ consists of some tubes of the original data tensor $\underline{\mathbf{X}}$.

Also the tensor lateral slice selection based on the t-product is formulated as follows

$$\underline{\mathbf{X}} \cong \underline{\mathbf{C}} * \underline{\mathbf{Y}}, \tag{5.11}$$

where $\underline{\mathbf{C}} \in \mathbb{R}^{I_1 \times L \times I_3}$ is a part of lateral slices of the tensor $\underline{\mathbf{X}}$ and the tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{L \times I_2 \times I_3}$ is computed in such a way that the reconstruction error (5.11) is as small as possible [257]. The best solution for the tensor $\underline{\mathbf{Y}}$ is

$$\underline{\mathbf{Y}} = \underline{\mathbf{C}}^{+} * \underline{\mathbf{X}},$$

we will not go through the theoretical details.

which provides the best approximation in a least-squares sense, that is

$$\left\|\underline{\mathbf{X}} - \underline{\mathbf{C}} * \left(\underline{\mathbf{C}}^{\dagger} * \underline{\mathbf{X}}\right)\right\|_F = \min_{\underline{\mathbf{Y}} \in \mathbb{R}^{L \times I_2 \times I_3}} \|\underline{\mathbf{X}} - \underline{\mathbf{C}} * \underline{\mathbf{Y}}\|_F ,$$

through the projection approximation $\underline{\mathbf{X}} \cong \underline{\mathbf{C}} * \underline{\mathbf{C}}^{+} * \underline{\mathbf{X}}$.

Algorithm 12: Tubal CUR algorithm

Input : A data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, positive numbers L_1, L_2, probability distributions p_i, $i = 1, 2, \ldots, I_2$ and p'_i, $i = 1, 2, \ldots, I_1$

Output: Tubal CTA $\underline{\mathbf{X}} \cong \underline{\mathbf{Z}} = \underline{\mathbf{C}} * \underline{\mathbf{U}} * \underline{\mathbf{R}}$

Select L_1 lateral slices of the tensor $\underline{\mathbf{X}}$ based on probability distributions p_i, $i = 1, 2, \ldots, I_2$ and set tensor $\underline{\mathbf{C}}$

Select L_2 horizontal slices of the tensor $\underline{\mathbf{X}}$ based on probability distributions p'_i, $i = 1, 2, \ldots, I_2$ and set tensor $\underline{\mathbf{R}}$

$\widehat{\underline{\mathbf{X}}} = \text{fft}\left(\underline{\mathbf{X}}, [], 3\right)$ $\widehat{\underline{\mathbf{C}}} = \text{fft}\left(\underline{\mathbf{C}}, [], 3\right)$, $\widehat{\underline{\mathbf{R}}} = \text{fft}\left(\underline{\mathbf{R}}, [], 3\right)$

for $i = 1, 2, \ldots, I_3$ **do**

  $\widehat{\underline{\mathbf{U}}}\left(:, :, i\right) = \widehat{\underline{\mathbf{C}}}\left(:, :, i\right)^{+} \widehat{\underline{\mathbf{X}}}\left(:, :, i\right) \widehat{\underline{\mathbf{R}}}\left(:, :, i\right)^{+}$

end

for $i = 1, 2, \ldots, I_3$ **do**

  $\widehat{\underline{\mathbf{Z}}}\left(:, :, i\right) = \widehat{\underline{\mathbf{C}}}\left(:, :, i\right) \widehat{\underline{\mathbf{U}}}\left(:, :, i\right) \widehat{\underline{\mathbf{R}}}\left(:, :, i\right)$

end

$\underline{\mathbf{Z}} = \text{ifft}\left(\widehat{\underline{\mathbf{Z}}}, [], 3\right)$, $\underline{\mathbf{U}} = \text{ifft}\left(\widehat{\underline{\mathbf{U}}}, [], 3\right)$

5.3 Tensor CUR for image completion

The proposed algorithm for the tensor completion task is described in this section. An important step in this technique is the CUR approximation of the underlying data tensors. To begin, a completion algorithm with rank minimization is stated as follows:

$$\begin{aligned} \min_{\underline{\mathbf{X}}} \quad & \text{rank}\left(\underline{\mathbf{X}}\right) , \\ \text{s.t.} \quad & \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}) , \end{aligned} \tag{5.12}$$

where $\underline{\mathbf{Y}}$ is the incompleted data tensor with only observed components with corresponding indices $\underline{\mathbf{\Omega}}$ and $\underline{\mathbf{X}}$ is the unknown data tensor that needs to be determined. The rank mentioned in formulations (5.12) can be CPD rank, TT/TR(TC) rank, Tucker rank, etc, and corresponding minimization problem is considered. In the case of Tucker rank which is an N-tuple $(R_1, R_2, \ldots, R_N)$, the following problem is

formulated as

$$\begin{aligned} \min_{\underline{\mathbf{X}}} \quad & \textstyle\sum_{n=1}^{N} R_n \,, \\ \text{s.t.} \quad & \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}.(\underline{\mathbf{Y}}) \,. \end{aligned} \tag{5.13}$$

The rank minimization problem in general is an NP-hard problem and a convex surrogate of it usually is replaced [36, 90]. The tensor decomposition formulation is a more efficient alternative to the nuclear norm minimization approach. The tensor decomposition formulation (5.14) is written as follows

$$\begin{aligned} \min_{\underline{\mathbf{X}}} \quad & \|\mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) - \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}})\|_F^2 \,, \\ \text{s.t.} \quad & rank(\underline{\mathbf{X}}) = R \,, \end{aligned} \tag{5.14}$$

where the unknown tensor $\underline{\mathbf{X}}$ has low tensor rank representation. Here again, different kinds of tensor ranks and associated tensor decompositions can be considered. Using an auxiliary variable $\underline{\mathbf{C}}$, the optimization problem (5.14) can be solved more conveniently by the following reformulation

$$\begin{aligned} \min_{\underline{\mathbf{X}},\underline{\mathbf{C}}} \quad & \|\mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}) - \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{C}})\|_F^2 \,, \\ \text{s.t.} \quad & rank(\underline{\mathbf{X}}) = R \,, \\ \text{and} \quad & \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{C}}) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}) \,. \end{aligned} \tag{5.15}$$

In this case, we can alternatively solve optimization problem (5.14) over variables $\underline{\mathbf{X}}$ and $\underline{\mathbf{C}}$. Therefore, the minimization problem can be approximated by the following iterative procedure

$$\underline{\mathbf{X}}^{(n)} = \mathcal{L}(\underline{\mathbf{C}}^{(n)}) \,, \tag{5.16}$$

$$\underline{\mathbf{C}}^{(n+1)} = \underline{\mathbf{\Omega}} \circledast \underline{\mathbf{Y}} + (\underline{\mathbf{1}} - \underline{\mathbf{\Omega}}) \circledast \underline{\mathbf{X}}^{(n)} \,, \tag{5.17}$$

where $\mathcal{L}$ is an operator which computes a low-rank tensor approximation of the data tensor $\underline{\mathbf{X}}^{(n)}$ and $\underline{\mathbf{1}}$ is a tensor whose all components are equal to one. Note that equation (5.16) solves the minimization problem (5.15) over $\underline{\mathbf{X}}$ for fixed variable $\underline{\mathbf{C}}$. Also equation (5.17) solves the minimization problem (5.15) over $\underline{\mathbf{C}}$ for fixed variable $\underline{\mathbf{X}}$. This algorithm consists of two main steps, *low tensor approximation* (5.16) and *Masking computation* (5.17). It starts from the initial incomplete data tensor

$\underline{\mathbf{X}}^{(0)}$ with the corresponding index set $\underline{\mathbf{\Omega}}$ and sequentially improves the approximate solution till some stopping criterion is satisfied or the maximum number of iterations is reached. Note that the term $\underline{\mathbf{\Omega}} \circledast \underline{\mathbf{Y}}^{(n)}$ is not required to be computed at each iteration as it is equal to the initial data tensor $\underline{\mathbf{X}}^{(0)}$. In general, any type of tensor decomposition can be utilized for low rank approximation in (5.16). For example, in [259], the Tucker decomposition is exploited at each iteration while in [29] and [46] the CPD and the TR/TC decomposition are used.

The proposed algorithm is summarized in Algorithm 13. The formulation is quite general and can incorporated into different tensor/matrix factorization cases. For instance, in the case of matrices, a matrix CUR algorithm selects some columns and rows ($\mathbf{C} \in \mathbb{R}^{I_1 \times R}$ and $\mathbf{R} \in \mathbb{R}^{R \times I_2}$) and computes the middle matrix as $\mathbf{U} = \mathbf{C}^{\dagger}\mathbf{X}\mathbf{R}^{\dagger} \in \mathbb{R}^{R \times R}$ and a low CUR approximation $\mathbf{X} \cong \mathbf{C}\mathbf{U}\mathbf{R}$ is also computed.

For the tensor case, different tensor decompositions can be utilized, such as Tucker decomposition [260, 150], tubal decomposition [25, 26], tensor CUR approximation [255]. To be more precise, let us consider the Tucker decomposition case. Here, in the first stage, the factor matrices $\mathbf{C}_n \in \mathbb{R}^{I_n \times R_n}$ are computed by sampling the columns of n-unfolding matrices (or n-mode fibers) after which the core tensor is computed as follows

$$\underline{\mathbf{S}} = \underline{\mathbf{X}} \times_1 \mathbf{C}_1^{\dagger} \times_2 \mathbf{C}_2^{\dagger} \cdots \times_N \mathbf{C}_N^{\dagger} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}. \tag{5.18}$$

Then, a low CUR Tucker approximation is computed at each iteration as follows

$$\underline{\mathbf{X}} = \underline{\mathbf{S}} \times_1 \mathbf{C}_1 \times_2 \mathbf{C}_2 \cdots \times_N \mathbf{C}_N. \tag{5.19}$$

Other kinds of CUR Tucker approximations such as FSTD or adaptive fiber sampling [253], Cross3D [252] can be utilized. In all our experiments, the sampling procedure is performed randomly, and as a result, the proposed algorithm can be considered a randomized tensor completion algorithm. It is also possible to exploit heuristic ones as it is a special case of our algorithm. However, they are slower than the randomized ones.

The simulations show that in the Tucker CUR and tubal CUR, the middle matrix

should be computed very carefully and accurately otherwise, the approximation scheme will be unstable, and the results will be very poor. In all our experiments for the Tucker CUR and tubal CUR, we have utilized formula (5.19) and (5.10), respectively.

Algorithm 13: Tensor CUR algorithm for tensor completion.

Input : An incomplete data tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, Tensor Rank $\mathbf{R}$, the set of observed components $\mathbf{\Omega}$, error bound ε and MaxIter.

Output: Completed data tensor $\underline{\mathbf{X}}^*$

$\underline{\mathbf{X}}^{(0)} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is a random tensor;

$\underline{\mathbf{Y}}^{(0)} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is a zero tensor;

for $n = 0, 1, 2, \ldots$ **do**

  $\underline{\mathbf{Y}}^{(n+1)} \leftarrow$Compute CUR approximation of the data tensor $\underline{\mathbf{X}}^{(n)}$ with rank R

  $\underline{\mathbf{X}}^{(n+1)} \leftarrow \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}^{(n+1)}) + \mathbf{P}_{\underline{\mathbf{\Omega}}^{\perp}}(\underline{\mathbf{Y}}^{(n+1)})$

  if $\frac{\|\underline{\mathbf{X}}^{(n+1)} - \underline{\mathbf{X}}^{(n)}\|_F}{\|\underline{\mathbf{X}}^{(n+1)}\|_F} < \varepsilon$ *or* $n >$ MaxIter **then**

    $\underline{\mathbf{X}}^* = \underline{\mathbf{X}}^{(n+1)}$ and **break**

  end

end

5.3.1 Multi-stage CUR for tensor completion

The tensor completion algorithm initializes a random data tensor with zero mean and unit variance, which is then updated sequentially to reconstruct the incomplete data tensor [2]. The CUR approximation of the underlying data tensor is created at each iteration, following which the mask operator is applied to keep fixed the observed elements of the underlying tensor. The operator $\mathcal{L}$ in (5.16) can be replaced by any of the tensor CUR algorithms described above. The approximation computed in the first stage is denoted by $\underline{\mathbf{X}}^{(1)}$ is a single-stage CUR approximation. This Single-stage CUR approximation does not work well in practice and so there is a need for a multi-stage CUR approximation by concatenating several single-stage CUR approximation. To be more specific, the above-mentioned procedure is repeated by computing a CUR approximation of the tensor $\underline{\mathbf{X}}^{(1)}$ and applying the mask operator

[2] We experimentally confirmed that the same results are achieved if the algorithm starts with the incomplete data tensor.

at each stage. This procedure is continued till a maximum number of iteration is reached or a stopping criterion is satisfied. This method also can be interpreted as a combination of a sequence of CUR approximation (linear operator) and mask functions (nonlinear operator). For a graphical illustration on the proposed approach see 5-6. Figure 5-7, shows that multistage CUR approximation provides better results. It can be seen that an incomplete image with PSNR 18.7717 was recorded after 100 iterations, a PSNR of 25.7841 was recorded at 200 iterations while for 500 iterations a PSNR of 31.3282 was achieved on the reconstructed MRI image. It was also experimentally found that if the selected fibers are smoothed in the tensor CUR approximation, the reconstruction performance of the algorithm is totally improved. In particular, this was more visible in difficult scenarios, such as high missing ratio or structural missing patterns.

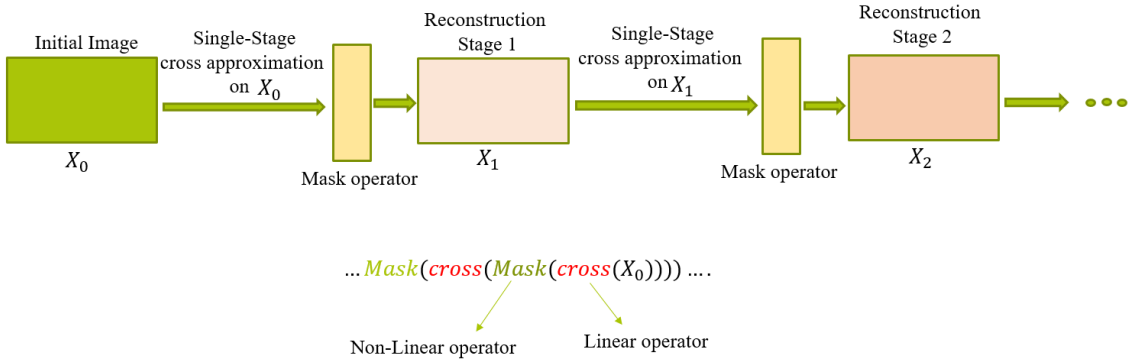


Figure 5-6: The procedure of the proposed algorithm as a multi-stage CUR approximation followed by mask operator.

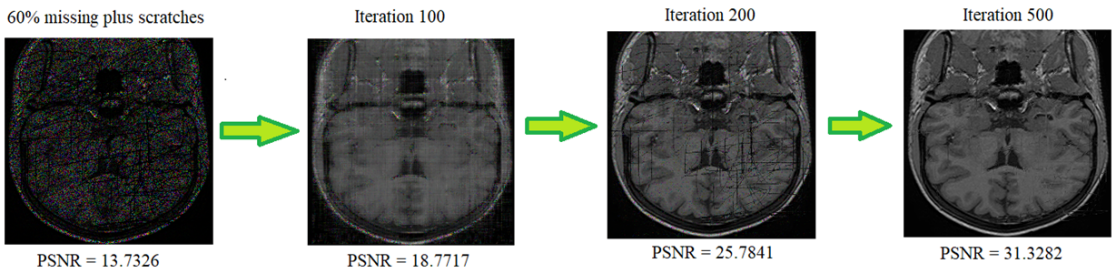


Figure 5-7: The proposed approach for reconstructing an MRI with structured missing.a

5.3.2 Smoothing techniques

The underlying elements of data tensors change smoothly or continuously in many applications. Examples of such data are images. It is therefore natural to consider the smoothness constraint when working on the optimization problems which are associated with the mentioned data tensors. There are various techniques and approaches to make the elements of a data tensor smooth such as low-pass filtering, moving averaging, locally estimated scatterplot smoothing (LOESS) and its weighed variant (LWOESS), the robust LOESS (RLOESS) and robust LWOESS (RLWOESS) etc., see [261, 262, 263, 264] for a comprehensive study of such smooth functions. As discussed in Section 5.1, the CMA methods can be generalized to tensor by sampling fibers, slice-fibers and slices. Firstly, we apply smoothness to the selected fibers and slices after their selections and then compute the corresponding low CTA. It is experimentally confirmed that this idea always provides better results than algorithm with no smoothness applied. More importantly, this can be seen in the situation with high missing ratio of elements or structural missing components, e.g., sequential columns and rows, the basic Algorithm 9 using Tucker CUR may not work properly while its smooth variant works perfectly.

That is, in the simulations, it was discovered that for difficult incomplete data tensors, such as removing sequential columns and rows or high missing ratio (95%), the tensor CUR algorithms may have a problem in reconstructing the incomplete data tensors. This is because the selected fibers of the incomplete data tensors have many zero components, and their components do not change continuously (smoothly) as expected in the case of images/videos. To overcome this difficulty, we propose to first perform smoothing on the selected fibers and then compute the CTA, refer to Figure 5-8 for illustration on the difference between the structure of smooth signal and non-smooth ones. Different types of smoothing techniques were used in the simulations including moving average, LOESS, LWOESS, LOESS, RLOESS, RLWOESS and Savitzky-Golay. Generally the smoothing technique significantly improves the Tucker CUR and tubal CUR algorithms with almost the same running time compared to the non-smooth version. The quality of the different smoothing strategies are compared in Figures 5-10 and 5-11, using some examples for the Tucker

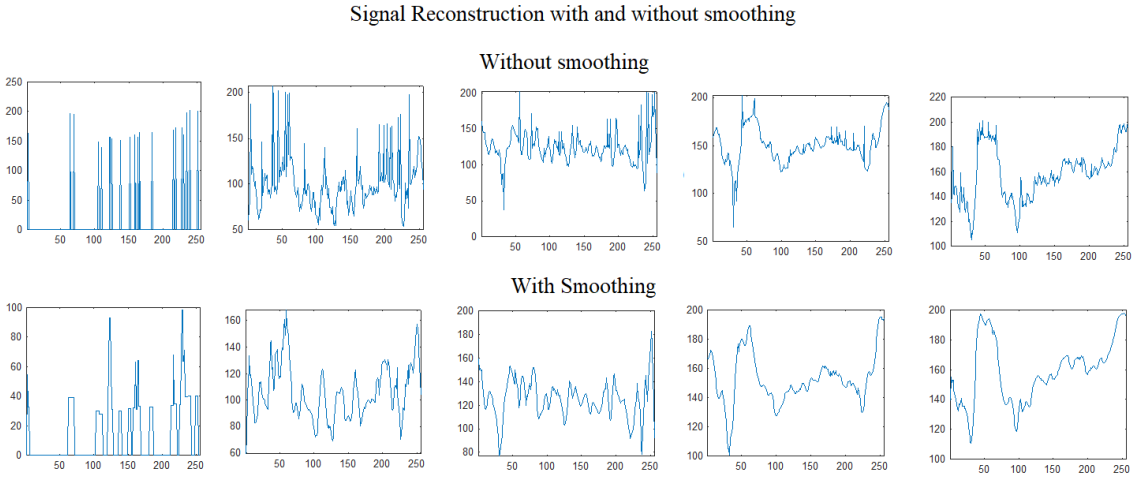


Figure 5-8: Illustration of the difference between the smooth fibers and non-smooth ones. The results are for the column fiber $\underline{\mathbf{X}}(:, 50, 1)$ using the Tucker CUR and the iterations 1, 10, 20, 30, 40, (from the left to the right).

and tubal decompositions respectively.

In our experiment, the Moving Average (MA) which is a special case of FIR (Finite Impulse Response) filter provided sufficiently good performance and simultaneously low complexity and shortest running time. In fact, the moving average method is the simplest type of such techniques used widely in the signal community as finite impulse response (low-pass) filter. It smooths out the elements of given data points, by replacing the elements of the data with a sequence of averages of different subsets of the given data points. Refer to Figure 5-11(a)-(b) for PSNR results using tubal CUR decomposition where we compare different smoothing techniques on the degraded "Baboon" and "Peppers" images shown in Figure 5-9. We can see that for this example, the moving average smoothing technique provided better recovery results than other smoothing approaches. Also, in Figure 5-8, the difference between the structure of one fiber of the "Peppers" image (fiber $\underline{\mathbf{X}}(:, 50, 1)$) with 95% missing ratio and its smooth forms during several iterations is shown. We used the MATLAB, the function "*smooth*" and it supports all the above-mentioned smoothing methods. The moving average technique with a span of 5 was utilized in all our experiments and it provided quite good and promising results. It is worth mentioning that inappropriate selection of the span parameter in moving average may lead to blur image output.

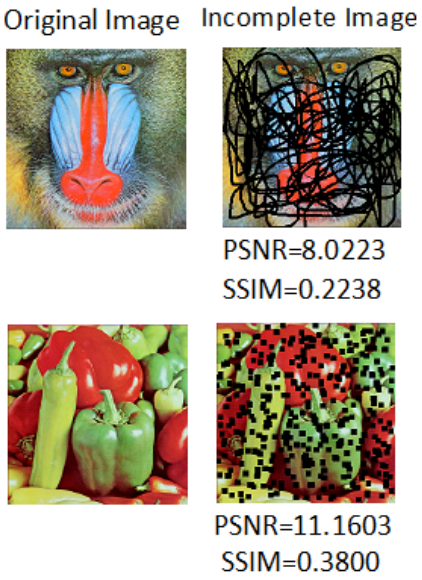


Figure 5-9: The original and the observed images.

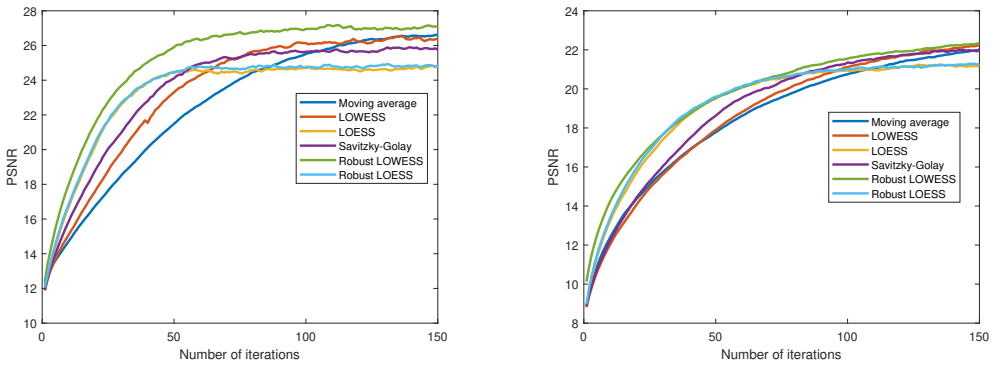
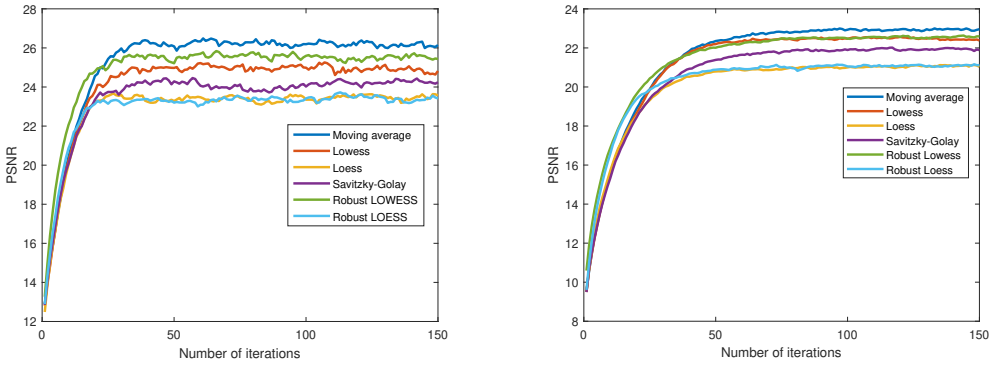


Figure 5-10: Comparing the performance of different smoothing strategies using the Tucker CUR approximation for reconstructing images with structured missing data.



(a) (b)

Figure 5-11: Comparing the performance of different smoothing strategies for reconstructing images using the tubal CUR approximation, a) incompleted "Pepper" in Figure 5-9 b) incomplete "Baboon" in Figure 5-9. 40 lateral/horizontal slices were used

5.3.3 Computational complexity

In this section, we make a brief comparison between the complexities of the algorithms. Let $\in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, and for the simplicity, assume $I_1 = I_2 = \ldots = I_N = I$ and $R_1 = R_2 = \ldots = R_N = R$, then the computational complexities of the TR-ALS, TRLRF and TR-WOPT per iteration are $\mathcal{O}(PNR^4I^N + NR^6)$, $\mathcal{O}(NR^2I^N + N^R6)$ and $\mathcal{O}(2NI^NR^2)$, respectively, where P is the total number of observations. The computational complexity of Tucker CUR is $\mathcal{O}(I^NR)$ which is lower than other completion algorithms. Our experimental results also confirm this. In our experiments, the TR-WOPT required more iterations for convergence and this is why in Figure 5-18, the TR-WOPT has high running time.

5.4 Experiments on MRI, images, times series and Video data

We examine the proposed CUR algorithms on image/video and MRI data completion. In all this experiments the sampling of fibers or slices were performed without replacement. The PSNR and SSIM is also used here to compare the performance of different completion algorithms.

MRI data completion: The first experiments show that the proposed Tucker CUR algorithm is applicable to medical images. To this end, we applied the Tucker CUR completion algorithm developed by us for the MRI dataset[3]. The original dataset is a 3rd order tensor of size $256 \times 256 \times 21$, see Figure 5-12 for some sample slices of the data tensor. 40% of the pixels of the original data tensor was uniformly removed and the Tucker CUR algorithm was applied with $R_1 = R_2 = 160,\ R_3 = 21$. The reconstructed images (also images with missing pixels) and their corresponding PSNR and SSIM are reported in Figure 5-12. We also apply the idea of smoothing the fibers in the first and second modes and the PSNR and SSIM comparisons are reported in Figure 5-14. In another experiment, MRI images of size $256 \times 256 \times 21$ was degraded using structural missing patterns. In the first instance a scratching pattern as seen in experiments in previous chapter was applied, whiles several rows and columns were removed in the second image before the CUR tensor completion was performed. Refer to Figure 5-13 for results on PSNR and SSIM. All the results shown in these experiments point to the fact that the proposed algorithm especially the algorithm equipped with smoothing idea is a promising approach for completing the medical images.

Image completion: The second experiment compares the Tucker CUR and tubal CUR with its smooth variants using images of size $256 \times 256 \times 3$. The benchmark images used in this simulations are "Baboon", "House", "Peppers", "Lena" and "Facade". Different kinds of missing components both random and structured where considered and the visual results are shown in Table 5.1. The reconstructed images were evaluated using PSNR and SSIM respectively. The running time of these methods are also shown in Table 5.1. The image "Lena" was used in a similar experiment where we applied a scratching degradation to render the image incomplete. Tubal CUR, FSTD, Slice-tube CUR, Tucker CUR and their smooth variants were applied on resultant image. The reconstructed images with PSNR and SSIM are displayed in Figure 5-15. Furthermore, we conduct more experiments using Tucker CUR compared with TRLRF [13], TR-WOPT [32], TR-ALS [46] and SPC

[3]From the https://www.kaggle.com/malekmechergui/mri-data-set, we have used the data s02_t1w_hires_defaced_MNI.nii.

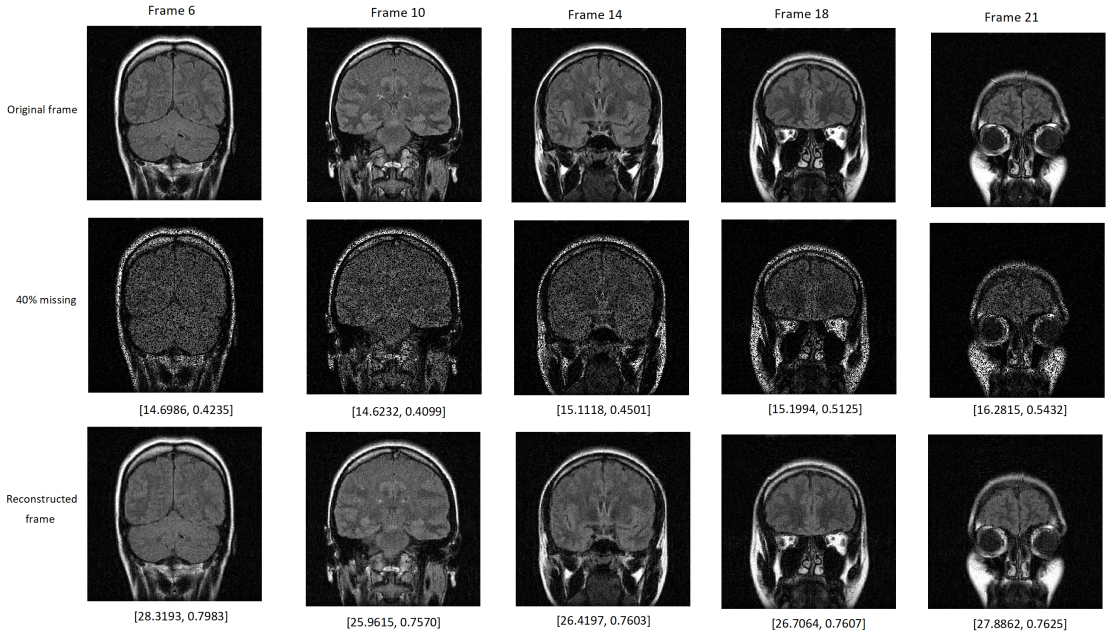


Figure 5-12: The reconstructed frame numbers (6-10-14-18-21) using the Tucker CUR algorithm for the MRI brain dataset.

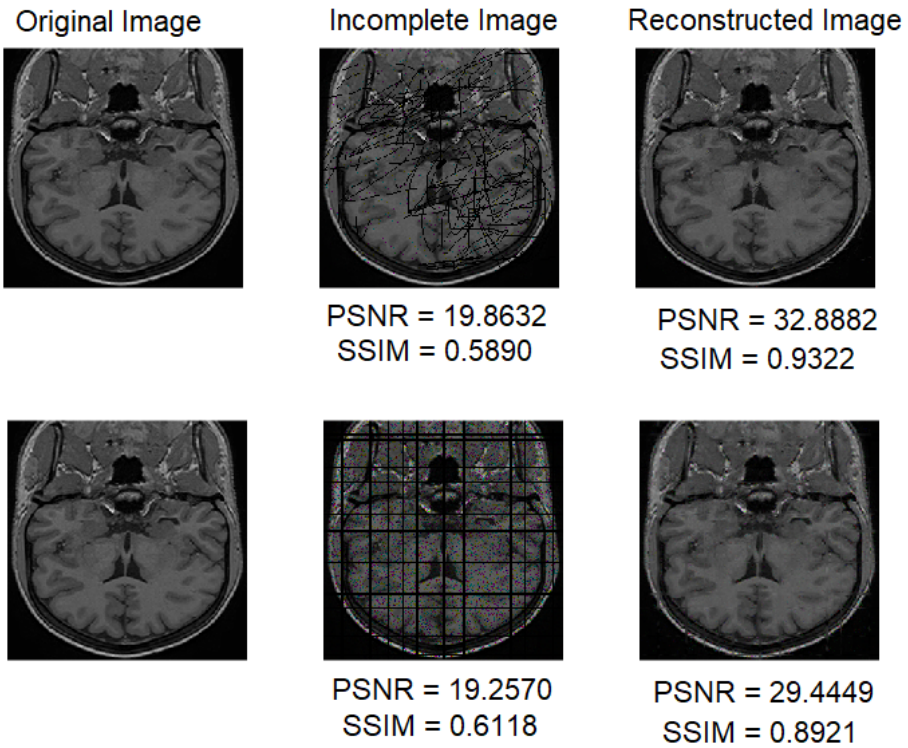


Figure 5-13: Performance of CUR on structured missing data

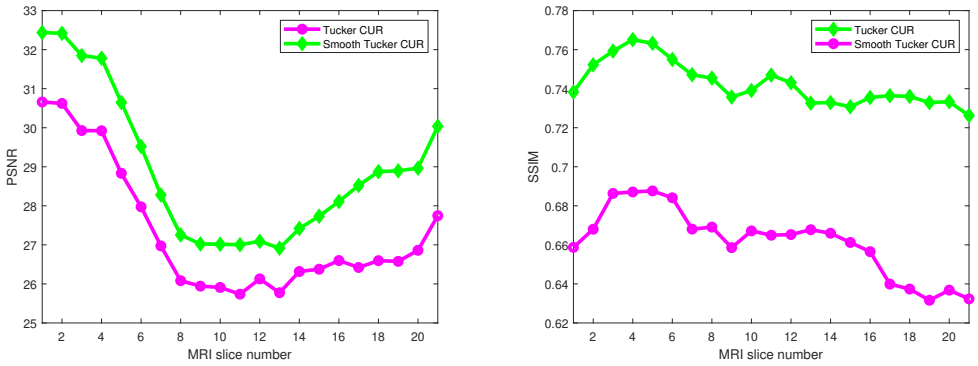


Figure 5-14: The comparison of Tucker CUR and Smooth Tucker CUR for the MRI dataset a) The PSNR comparison, b) The SSIM comparison.

Table 5.1: The PSNR and SSIM of the initial incomplete images and those achieved by the Tucker CUR, tubal CUR and their smooth variants with corresponding running time (in second) for images with 95% missing components. The Tucker rank $(37, 37, 3)$ and tubal rank $R_1 = 25$, $R_2 = 25$ were used.

Image	Incomplete image [PSNR, SSIM]	Tucker CUR [PSNR, SSIM, Time]	Smooth Tucker CUR [PSNR, SSIM, Time]	Tubal CUR [PSNR, SSIM, Time]	Smooth Tubal CUR [PSNR, SSIM, Time]
Lena	[5.3439, 0.0077]	[16.1314, 0.7696, 2.15]	[**22.1557**, **0.9011**, 4.11]	[16.7514 0.7356, 3.26]	[21.4871, 0.8919, 4.97]
House	[4.8375, 0.0042]	[18.2095, 0.6122, 2.17]	[**22.6334**, **0.8168**, 3.15]	[17.9430, 0.4930, 3.48]	[21.9302 0.7923, 5.26]
Facade	[5.9693, 0.0045]	[21.5613, 0.6044, 1.64]	[**22.5591**, **0.6445**, 3.57]	[22.3742, 0.6061, 3.23]	[20.0580, 0.4271, 5.22]
Peppers	[6.1378, 0.0088]	[14.7233, 0.6678, 1.52]	[**20.6701**, **0.8751**, 2.32]	[14.4540, 0.6202, 5.36]	[20.0312, 0.8580, 4.620]
Baboon	[5.5815, 0.0064]	[16.5684, 0.4111, 2.05]	[**19.8792**, **0.5763**, 3.38]	[16.0691, 0.3464, 3.20]	[19.5935, 0.5624, 4.82]

[49]. These methods are among the state-of-the-art algorithms for the tensor completion task. The reconstructed images using different completion algorithms along with their corresponding PSNR and SSIM and running times are reported in Figure 5-16. Detailed information regarding each of the parameters used in the simulations for TRLRF, TR-WOPT, TR-ALS and SPC are provided in Table 5.2. For an RGB image, we have only three frontal slices (RGB channels) and it is expected to select pixels from all three channels otherwise we completely loose one of the colors which leads unsatisfactory results. On the other hand, selecting all slices means using all elements of the data tensor, which defeats the purpose of sampling, which is to use only a part of the elements, not all of them. In this sense, for Algorithm 11 to be effective, we reshaped the images to a 3rd order tensor of size $64 \times 64 \times 48$ and then select some slices and tubes. For the Tucker CUR case, we considered Tucker rank of $(70, 70, 3)$ in which we select 70 columns, 70 rows and 3 channels. For the

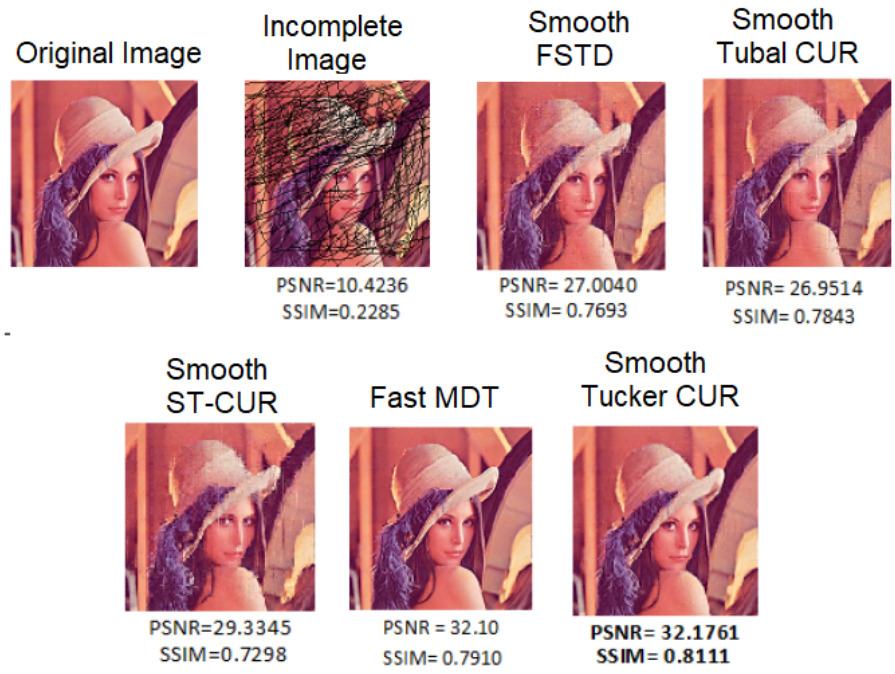


Figure 5-15: Experimental results using "Lena" image on different CUR methods

tubal CUR, we considered 40 lateral and 40 horizontal slices and for the slice-tube CUR, we considered 24 frontal slices and 1200 tubes.

Figure 5-17 compares the PSNR and the CPU time for the different completion algorithms. For the Fast MDT algorithm [116], we considered the delay embedded of $\tau = (32, 32, 1)$, tolerance $10^{(-4)}$, with 1000 as the maximum number of iterations. It can be seen that the proposed CUR algorithms are more scalable for achieving higher performance than the other completion algorithms.

Video completion: In the last experiment for this method, we consider "Akiyo" and "News" grayscale video datasets[4] each of which is a 3rd order tensor of size $176 \times 144 \times 300$. Although our algorithms are scalable to the number of frames but in order to compare with other completion algorithm we only consider the first 30 frames of the video and consider a 3rd order tensor of size $176 \times 144 \times 30$. In the next step we remove randomly 70% pixels of the videos and apply the completion algorithms to complete them. For Tucker CUR, we considered a Tucker rank of $(90, 90, 20)$, for the tubal CUR, we selected 40 lateral and horizontal slices and for

[4] http://trace.eas.asu.edu/yuv/

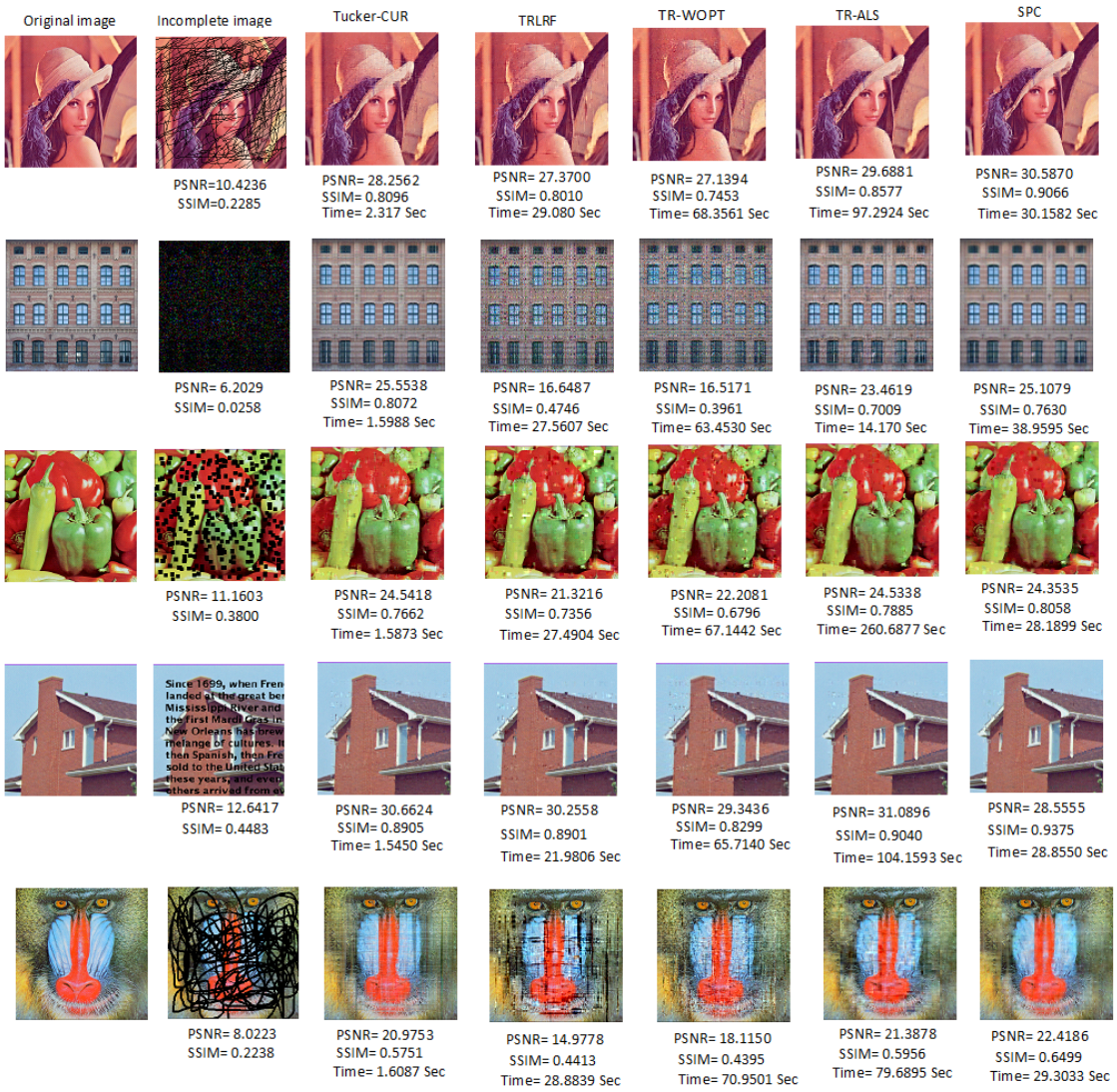


Figure 5-16: The reconstructed images using different tensor completion algorithms.

Table 5.2: The parameters of different completion algorithms used in our experiments for images/videos completion.

Images				
	TRLRF	TR-WOPT	TR-ALS	SPC
Size	$(256, 256, 3)$	$(256, 256, 3)$	$(4, 4, 16, 4, 4, 16, 3)$	$(256, 256, 3)$
TR-Rank	$(5, 5, 5)$	$(5, 5, 5)$	$(10, 10, \ldots, 10)$	Type:Quadratic
Max Iteration	300	100	10	150
Videos				
	TRLRF	TR-WOPT	TR-ALS	SPC
Size	$(176, 144, 30)$	$(176, 144, 30)$	$(16, 4, 4, 6, 33, 15)$	$(176, 144, 30)$
TR-Rank	$(5, 5, 5)$	$(5, 5, 5)$	$(10, 10, \ldots, 10)$	Type:Quadratic
Max Iteration	300	100	10	150

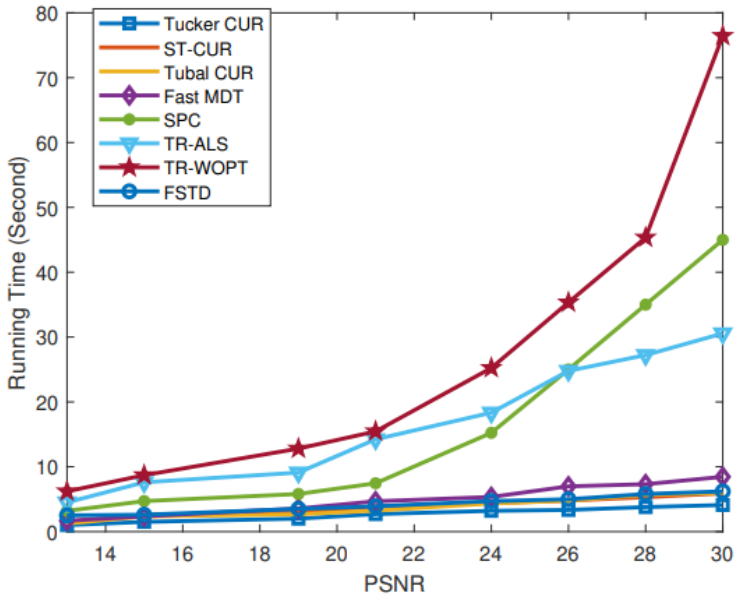


Figure 5-17: Comparing the performance and running times of different completion algorithms for the "House" image.

the Slice-tube CUR algorithm, we selected 15 frontal slices and 1500 tubes. The running time of all completion algorithms for recovering the "Akiyo" and "News" videos with 70% missing components is reported in Figure 5-18. The results show that the Tucker CUR algorithms need less running time for recovering the mentioned video with almost the same performance of other completion algorithms. Same parameters are applied on the News video data set and 70% pixels are removed.

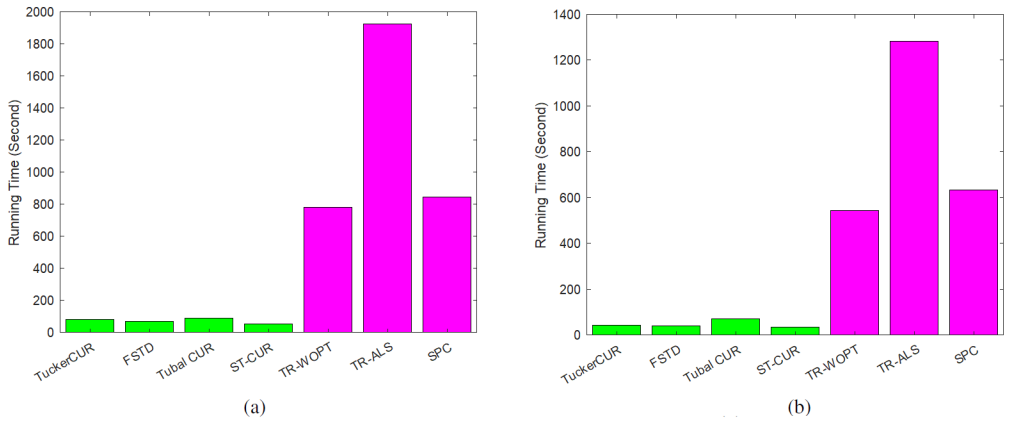


Figure 5-18: The running time comparison of different completion algorithms for a) News video dataset, and b) Akiyo video dataset.

Chapter 6

Low rank tubal decomposition for MRI motion artifact correction

Outliers in k-space measurements cause a variety of MR image artifacts. When a subject moves during scanning, motion artifacts are frequently observed [265, 266]. Some of the k-space data lose their integrity from the neighboring k-space data due to movements during the k-space scanning, which can also be considered as outliers [132]. Various methods have been implemented to correct motion artifacts in MRI images [267, 268, 269, 270, 271, 272, 273]. Recently, inspired by the duality property between the low-rank structure of a Hankel matrix in k-space and the sparsity in the image domain, motion artifact reconstruction models using low rank Hankel matrix and other optimization techniques have been proposed [133, 132, 136, 137, 274, 136]. These approaches have proven to work very well in reconstructing MRI data from sampled sparse measurements. But the downside to most of these methods is it that they do not use the data in its natural tensor form. And so the global correlation and information in the data is lost during the data transformation stages. As an extension of these methods, in this work, we take advantage of the original structure of the medical data so as not to lose important information. We propose a tensor completion algorithm which is effective in the reconstruction of motion artifact in MRI Diffusion weighted images and Dynamic MRI images. Based on the tensor-tensor product (t-product) and tensor singular value decomposition (t-SVD) [26, 192], Zhang *et al.* [28] formulated a low-tubal rank structure for a tensor

and proposed a new tensor nuclear norm (TNN), which has been proven to be the tightest convex relaxation for the tensor case. The t-SVD model is relatively simple and its associated learning algorithm is quite efficient. It is known that unlike the Tucker and Canonical Polyadic Decompositions, the t-SVD has similar properties to the classical SVD. More precisely, the truncated t-SVD provides the best low tubal rank approximation in the least-squares sense for any invariant tensor norm [25]. This was one motivation to use the t-SVD in our formulation. On the other hand, experimental results reported in the literature show the efficiency and performance of the t-SVD in many applications, such as tensor completion and tensor denoising. It is worth mentioning that several fast algorithms have been proposed to decompose a tensor into the t-SVD format and this facilitates the utilization of the t-SVD for real-world big data processing. The rest of this chapter tackles, a Hankel tensor-based reconstruction method for motion artifacts. The method first converts the sampled k-space data into a Hankel tensor which helps to better reveal the relationship in the data. Then a low rank tensor completion algorithm is used to reconstruct the acquired image.

Figure 6-1: Illustration of framework for MRI motion data reconstruction

6.1 Proposed Low rank tubal method

Even though some MR images, are already sparse in their pixel representation, most complex images are rarely sparse and only have a sparse representation in some transform domain, like their wavelet coefficients or their spatial finite differences [275]. Based on this observation, Lustig *et al.* [276] proposed the first optimiza-

tion method for compressed sensing MRI data using the spatial domain wavelet transform. More specifically, the problem is formulated as

$$\begin{aligned} &\min_x \|\Psi \mathbf{x}\|_1, \\ &s.t \|\mathbf{FSx} - \mathbf{y}\| < \epsilon, \end{aligned} \tag{6.1}$$

where $\|.\|_1$ is the l_1 norm, Ψ is some spatial transformation operation, $\mathbf{F}$ is the Fourier transform operator, $\mathbf{S}$ is down-sampling pattern or the sensitivity data maps, $\mathbf{y}$ is the downsampled acquired k-space measurements and ϵ is some noise level .

However, this optimization algorithms are computational expensive. Therefore advanced formulation for MRI compressed sensing was invented. One of the simplest approaches for motion reconstruction is the Sensitivity encoding method (SENSE) [277]. This approach explicitly utilizes the estimated coil maps to obtain an augmented compressed sensing problem. Based on the SENSE method, a general optimization problem for MRI reconstruction is formulated as:

$$\min_x \sum_{t=1}^{T} \|\mathbf{FSx}_t - \mathbf{y}_t\| + \lambda \|\Psi(\mathbf{X})\|_*. \tag{6.2}$$

Ψ is some transformation operation, which is the Hankelization procedure in our case, λ is the regularization parameter, and $\|.\|_*$ denotes the nuclear norm regularization in the form of a sum of the singular values of the tensor.

For simplicity, we make the following changes to the above equation:

- We represent $\mathbf{FS}$ as $\mathbf{A}$, which denotes the downsampled Fourier transform as can be seen in [275]
- The sampled/aquired signals $\mathbf{y}$ are concatenated to form a matrix $\mathbf{Y}$
- Similarly we tranform the estimated signal $\mathbf{x}_1, \ldots, \mathbf{x}_T$ into a matrix $\mathbf{X}$.

This results in a new equation as follows:

$$\min_{\{\mathbf{X}\}} \quad \|\mathbf{P}_\Omega (\mathbf{AX} - \mathbf{Y})\|_F^2 + \lambda \|\Psi \mathbf{X}\|_* , \tag{6.3}$$

where $\mathbf{P}_\Omega$ denote the projection of the k-space sampling index. Applying the Han-

kelization method to the equation 6.3, we convert it into a tensor case problem. Therefore we reformulate the above problem and generalize the completion model for the matrix case to higher order tensors by solving the following optimization problem:

$$\min_{\{\underline{\mathbf{X}}_H\}} \quad \tfrac{1}{2} \left\| \mathbf{P}_{\underline{\Omega}} \left(\underline{\mathbf{X}}_H \times_2 \mathbf{A} - \underline{\mathbf{Y}}_H \right) \right\|_F^2 + \lambda \left\| \underline{\mathbf{X}}_H \right\|_* , \tag{6.4}$$

where $\underline{\mathbf{X}}_H \times_2$ is the mode 2 unfolding of the tensor $\underline{\mathbf{X}}$. Again to further simplify our expression and notations, $\underline{\mathbf{X}}_H \times_2 \mathbf{A}$ will be represented by $\underline{\mathbf{X}}_H$. Therefore our equation (6.4) above becomes:

$$\min_{\{\underline{\mathbf{X}}_H\}} \quad \tfrac{1}{2} \left\| \mathbf{P}_{\underline{\Omega}} \left(\underline{\mathbf{X}}_H - \underline{\mathbf{Y}}_H \right) \right\|_F^2 + \lambda \left\| \underline{\mathbf{X}}_H \right\|_* . \tag{6.5}$$

The tensor completion algorithm in this section is solved using the tensor singular singular decomposition model. The motivation for using this new formulation is comparing the performance of the matrix and the tensor variants in reconstructing the images, this formulation enables one to take advantage of the inherent multi-dimensionality of data [278]. Inspired by the results achieved by the nuclear norm minimization of matrices for recovering data matrices with missing values, Zhang *et al.*[28] proposed the tubal nuclear norm minimization approach based on t-SVD, defined as the sum of nuclear norms of all frontal slices in the Fourier domain and proved to be convex envelope to the tensor tubal rank. More precisely, the model of rank minimization based tensor completion is formulated as follows

$$\min_{\underline{\mathbf{X}}} \quad \left\| \mathcal{P}_{\underline{\Omega}} \left(\underline{\mathbf{X}}_H - \underline{\mathbf{Y}}_H \right) \right\|_F^2 + \text{tubal rank} \left(\underline{\mathbf{X}}_H \right) , \tag{6.6}$$

Similar to the matrix case, minimizing the tubal tensor rank is NP hard because it includes the matrix case as a special case. The matrix trace norm was generalized to the tensor case based on the t-product in [279, 192, 27, 28]. We use the one introduced in [27, 192] which has been shown to provide superior results compared to the others and to be faster because of using only the information of the first slice in the Fourier domain. As a result, the following optimization problem is formulated

the following optimization problem

$$\min_{\underline{\mathbf{X}}} \quad \tfrac{1}{2} \left\| \mathcal{P}_{\underline{\Omega}} \left(\underline{\mathbf{X}}_H - \underline{\mathbf{Y}}_H \right) \right\|_F^2 + \lambda \left\| \underline{\mathbf{X}}_H \right\|_* , \tag{6.7}$$

where $\|.\|_*$ is the tubal nuclear norm. It should be noted that, the truncated tubal nuclear norm [27] shown in Algorithm 7 can also be used in the formulation (6.7). Similar tensor completion formulation is used in [118] but here we have used unitary transform matrices instead of discrete Fourier transform matrix that is used in the traditional tensor SVD and has shown to provide better results [280].

This optimization function is solved via the augmented Lagrangian method. Using ADMM [41], we transform the equation into a constrained one by introducing an auxiliary tensor $\underline{\mathbf{Z}}_H$ with same size as the tensor $\underline{\mathbf{X}}_H$:

$$\begin{aligned} \min_{\{\underline{\mathbf{X}}_H, \underline{\mathbf{Z}}_H\}} \quad & \tfrac{1}{2} \left\| \mathbf{P}_{\underline{\Omega}} \left(\underline{\mathbf{Z}}_H - \underline{\mathbf{Y}}_H \right) \right\|_F^2 + \lambda \left\| \underline{\mathbf{X}}_H \right\|_* , \\ s.t. \quad & \underline{\mathbf{X}}_H = \underline{\mathbf{Z}}_H . \end{aligned} \tag{6.8}$$

Our model utilizing the ADMM algorithm [41] because it has been shown to have fast convergence and good performance. The augmented Lagrangian function corresponding to the constrained optimization problem (6.8), can be constructed as

$$\mathcal{L} \left(\underline{\mathbf{X}}_H, \underline{\mathbf{Z}}_H, \underline{\mathbf{T}} \right) = \lambda \left\| \underline{\mathbf{X}}_H \right\|_* + \frac{1}{2} \left\| \mathbf{P}_{\underline{\Omega}} \left(\underline{\mathbf{Z}}_H - \underline{\mathbf{Y}}_H \right) \right\|_F^2 + \langle \underline{\mathbf{T}}, \underline{\mathbf{X}}_H - \underline{\mathbf{Z}}_H \rangle + \frac{\mu}{2} \left\| \underline{\mathbf{X}}_H - \underline{\mathbf{Z}}_H \right\|_F^2 , \tag{6.9}$$

where $\underline{\mathbf{T}}_H$ is a tensor representing the Lagrangian multipliers and μ is penalty parameter. While the method works well for μ between 0.1 to 1.1. We set $\lambda = 0.5$ in our simulations.

The Lagrangian function (6.9), and the ADMM [41] solution is converted to simpler sub-problems. The sub-problem for $\underline{\mathbf{X}}_H$ is updated as:

$$\underline{\mathbf{X}}_H^{k+1} = \min_{\underline{\mathbf{X}}_H} \ \mathcal{L} \left(\underline{\mathbf{X}}_H, \underline{\mathbf{Z}}_H^k, \underline{\mathbf{T}}^k, \mu^k \right) . \tag{6.10}$$

Problem (6.10) is solved via

$$\underline{\mathbf{X}}_H^{k+1} = \min_{\underline{\mathbf{X}}_H} \left(\lambda \left\| \underline{\mathbf{X}}_H \right\|_* + \frac{\mu_k}{2} \left\| \underline{\mathbf{X}}_H - \underline{\mathbf{Z}}_H^k + \frac{\underline{\mathbf{T}}^k}{\mu^k} \right\|_F^2 \right). \tag{6.11}$$

Its closed form solution can be written as:

$$\underline{\mathbf{X}}_H^{k+1} = \underline{\mathbf{D}}_\beta \left(\underline{\mathbf{Z}}_H^k - \frac{\underline{\mathbf{T}}^k}{\mu^k} \right), \tag{6.12}$$

where $\underline{\mathbf{D}}_\beta(.)$ is the tensor singular value thresholding operation. The value for β is derived from $\frac{\lambda}{\mu^k}$ and is updated at each iteration. The augmented Lagragian function w.r.t $\underline{\mathbf{Z}}$ can be updated using the equation:

$$\mathbf{Z}_H^{k+1} = \min_{\mathbf{Z}_H} \ \mathcal{L}\left(\underline{\mathbf{X}}_H^{k+1}, \underline{\mathbf{Z}}_H, \underline{\mathbf{T}}^k, \mu^k \right). \tag{6.13}$$

Sub-problem (6.13) for $\underline{\mathbf{Z}}$ can be solved through:

$$\mathbf{Z}_H^{k+1} = \min_{\underline{\mathbf{Z}}_H} \ \frac{1}{2} \left\| \mathbf{P}_{\underline{\Omega}} \left(\underline{\mathbf{Z}}_H - \underline{\mathbf{Y}}_H \right) \right\|_F^2 + \frac{\mu_k}{2} \left\| \underline{\mathbf{X}}_H^{k+1} - \underline{\mathbf{Z}}_H + \frac{\underline{\mathbf{T}}^k}{\mu^k} \right\|_F^2. \tag{6.14}$$

The closed form solution for problem (6.14) is also solved through:

$$\underline{\mathbf{Z}}_H^{k+1} = \mathbf{P}_{\underline{\Omega}}^{\perp} \left(\underline{\mathbf{X}}_H^{k+1} + \frac{\underline{\mathbf{T}}^k}{\mu^k} \right) + \underline{\mathbf{Y}}_H. \tag{6.15}$$

The solution for $\underline{\mathbf{T}}$ is also converted to a simpler optimization as:

$$\underline{\mathbf{T}}^{k+1} = \min_{\mathbf{T}} \ \mathcal{L}\left(\underline{\mathbf{X}}^{k+1}, \underline{\mathbf{Z}}^{k+1}, \underline{\mathbf{T}}, \mu^k \right). \tag{6.16}$$

The sub-problem (6.16) for $\underline{\mathbf{T}}$ can be solved through:

$$\underline{\mathbf{T}}^{k+1} = \underline{\mathbf{T}}^k + \mu^k \left(\underline{\mathbf{X}}_H^{k+1} - \underline{\mathbf{Z}}_H^{k+1} \right). \tag{6.17}$$

The Lagrangian parameter μ is solved through the equation:

$$\mu^{k+1} = \min \ \left(\mu^k, \mu_{max} \right). \tag{6.18}$$

Algorithm 14: Algorithm for low rank tensor completion

Input : Sampled data $\underline{\mathbf{Y}}_H \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ in Hankel form, the observation index tensor $\underline{\mathbf{\Omega}}$, and regularization parameter $\lambda > 0$, $\mu = 1e-4$ $\underline{\mathbf{Z}}_H^0 = \underline{\mathbf{X}}_H^0 = \underline{\mathbf{T}}_H^0 = 0$, $\mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{X}}_H) = \mathbf{P}_{\underline{\mathbf{\Omega}}}(\underline{\mathbf{Y}}_H)$, $K = 300$ is maximum iteration, $tol = 1e-5$.

Output: Recovered data tensor $\underline{\mathbf{X}}_H$

while *A stopping criterion is not satisfied* **do**
- Update $\underline{\mathbf{X}}_H^{k+1}$ with equation (6.12)
- Update $\underline{\mathbf{Z}}_H^{k+1}$ with equation (6.15)
- Update $\underline{\mathbf{T}}^{k+1}$ with equation (6.17)
- Update μ^{k+1} with equation (6.18)
- Check convergence conditions
- $\left\|\underline{\mathbf{X}}_H^{k+1} - \underline{\mathbf{X}}_H^k\right\|_* \leq tol$
- **Return** $\underline{\mathbf{X}}_H$

end

where μ_{max} is a given fixed number and μ is the parameter obtained from the previous iteration. Refer to Algorithm 14 for steps to performing the tensor completion and Algorithm 15 for the full algorithm.

Algorithm 15: Algorithm for MRI motion reconstruction

Input : MRI data with motion artifact

Output: Reconstructed MRI data $\mathbf{X}$

1. **Select** 40% of data in k-space from each channel
2. **Apply** sensitivity map to generate the down-sampled signal $\mathbf{y}_t$
3. **Concatenate** acquired signals $\mathbf{y}_1, \ldots, \mathbf{y}_T$ to form $\mathbf{Y}$
4. **Perform** Hankel folding on $\mathbf{Y}$ to generate $\underline{\mathbf{Y}}_H$
5. **Use** Algorithm 14 to complete the data
6. **Perform** inverse Hankel folding of $\underline{\mathbf{X}}_H$ to obtain $\mathbf{X}$

6.2 Experiments

MRI data used for this part of the research consist of:

- DWI one: A four-shot brain DWI data and corresponding sensitivity map of size $248 \times 244 \times 8$ with 8 coils.
- DWI Two: A 4 shots accelerated DWI data and corresponding sensitivity map of size $248 \times 244 \times 32$ with 32 coils.

- Dynamic MRI: Under sampled Cartesian cardiac perfusion MRI data of size $128 \times 128 \times 40$ with 12 coils.

Motion Artifact Reconstruction. We compared the performance of the algorithm with some MRI multi-shot diffusion weighted imaging techniques that adopts low rank completion for reconstruction. The algorithms which we used were SENSE [277], MUSSELS [136], Shot-LLR [137]. For fair comparison with these algorithms, we used the hyper-parameters as stated and used in each code to make performance as best as possible. In the experiments, it was realised that setting the penalty parameter $\lambda = 1e - 8$, achieves better performance and was used in all our simulations. The initial regularization parameter μ was set to $\mu = 1e - 4$. Figures 6-3 and 6-2 show reconstruction results. From the results obtained it can be seen that our method is comparable to other existing methods for motion reconstruction. The curve in Figure 6-4 presents results using RMSE to evaluate some of the methods used with our proposed method. It can be seen that our method converges faster than other methods and also has the lowest RMSE values. In another simulation, we reconstructed images from undersampled Dynamic MRI data. The experiments were performed using different sampling ratios and compared with other methods such as k-t SLR [269], FTVNNR [281], SRTPCA [273], BCS [270]. The results obtained from our method is comparable to the methods used and in some cases performs better. Figure 6-6 show the PSNR and RMSE comparison of the results and a visual representation of the results with 40% sampled data is also shown in Figure 6-5.

In addition, experiments are conducted for denoising MRI perfusion data. Noise of standard deviation $\sigma = 10^{-2}$ is added to the data. A visual and PSNR comparison of our result with the DIP [141] is presented in Figure 6-7. All hyper-parameters for denoising using DIP was used as stated in the paper[1]. For the proposed method, a block based approach where 8×8 patches of the noisy data was reconstructed in parallel using the t-SVD approach was adopted. Our method yields less blurry profile than the DIP method.

[1]https://dmitryulyanov.github.io/deep_image_prior

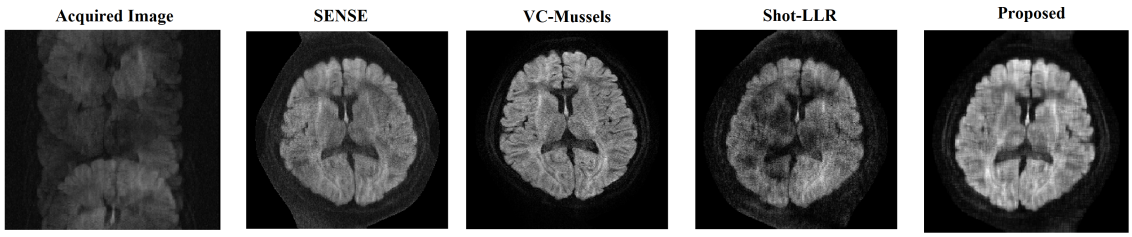


Figure 6-2: Reconstruction of Multi-shot DWI from 55% of k-space sampled data using samples from DWI one.

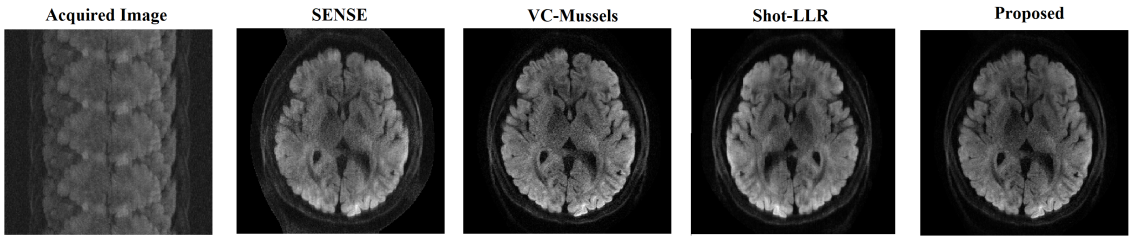


Figure 6-3: Reconstruction of Multi-shot DWI from 45% of k-space sampled data using samples from DWI Two.

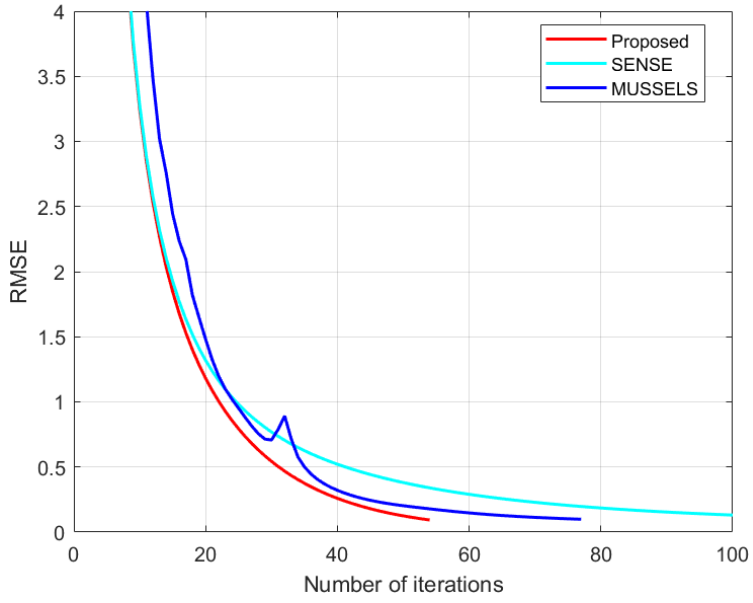


Figure 6-4: Convergence results from the algorithm using RMSE.

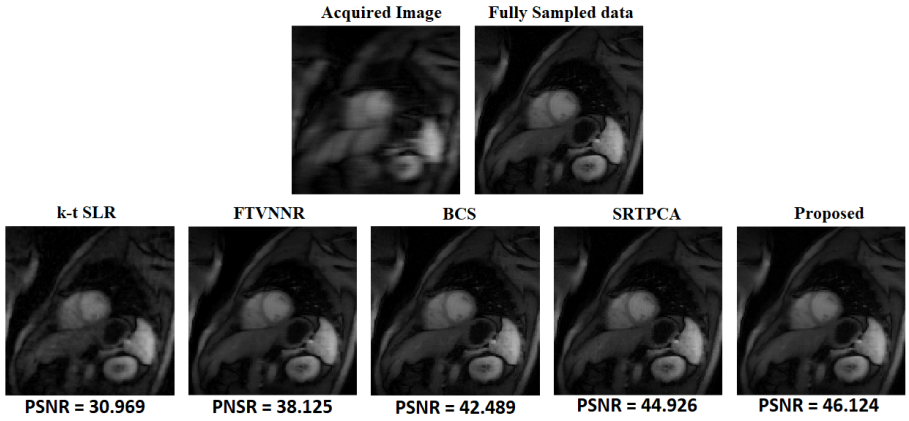
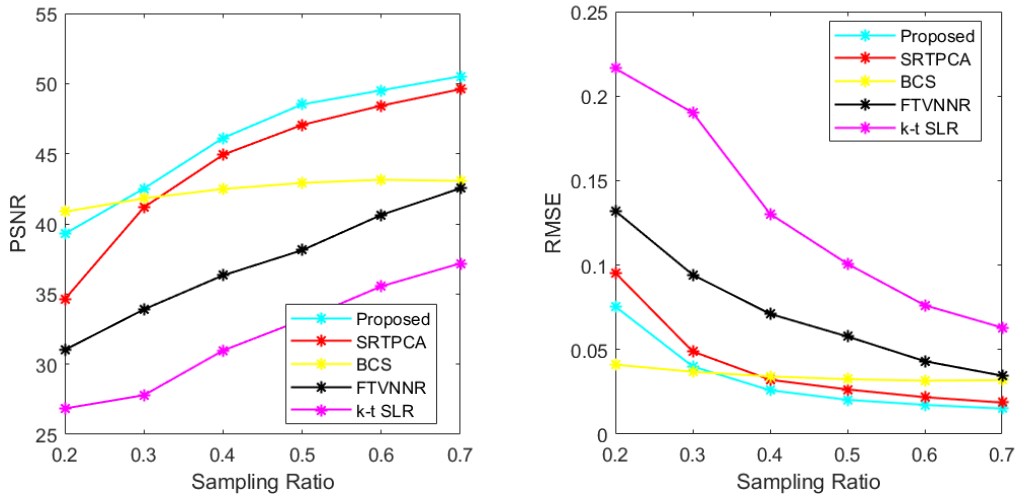


Figure 6-5: PSNR comparison on $20th$ frame of a cardiac perfusion MRI with 40% sampling of k-space on the Dynamic MRI data.



(a) Comparison of PSNR vs Sampling ratio (b) RMSE vs Sampling ratio

Figure 6-6: Comparison of PSNR and RMSE with different sampling ratios on other MRI reconstruction methods.

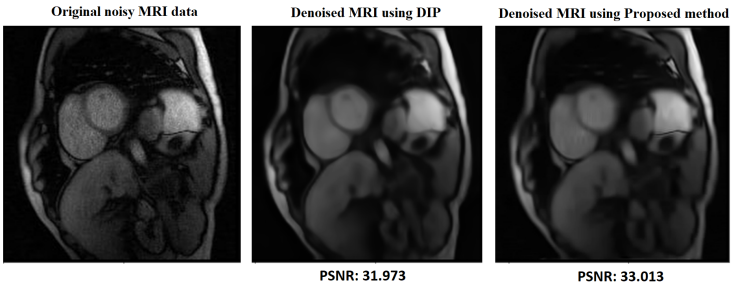


Figure 6-7: Denoised result of a frame from a cardiac MRI with noise using Deep Image Prior (DIP) and Proposed method

Chapter 7

Conclusion and Future work

The removal of noise in data is critical in various medical signals and imaging applications. For signals in medical imaging, in particular, the reduction of noise is a significant pre-processing task. One common approach used in the signal processing community is performing Hankel folding on the data. The Hankel folding is capable of capturing some delay or shift-invariant structure in the data. Therefore, in cases where the elements in some continuous slices are missing in the data, performing this delay embedding or Hankel folding provides a solution to this problem. Several variants of the Hankel tensors have been introduced in recent years. The first part of this work investigated their similarities and differences with particular emphasis on their advantages when used as a pre-processing step in the tensor completion task.

Furthermore, the second part of this thesis looked at three tensor completion algorithms for reconstructing MRI and EEG data. These methods also work well for natural RGB images. Low rank, sparsity and smoothness constraints were exploited in these techniques. In the first method, a novel and efficient tensor completion algorithm was proposed for recovering data tensors with random and/or structured missing entries. The main idea is to Hankelize the incomplete data tensor in order to obtain high-order tensors prior to learning the core tensors of the tensor ring representation with sparse constraints. The core tensors are learned using an efficient ODCT dictionary and sparse representation techniques. The ADMM algorithm and APG approaches were adopted to solve the underlying optimization problems.

Simulations show that our proposed method is able to recover incomplete data tensors with different types of structured and random missing elements. The algorithm exhibits higher performance in comparison to many existing tensor decomposition methods, while providing lower computational cost in comparison to other tensor completion approaches.

The second method proposes a general framework to use the tensor cross approximation based algorithms for reconstructing incomplete data tensors. The algorithms are simple and easy to implement with low computational complexity. For cases of data tensor with structural missing components or high missing rate, an efficient smooth variant of the developed tensor CUR algorithm which performs an initial smoothing on the sampled fibers before applying the CUR algorithms is implemented. The efficiency and performance of the method is exhibited with extensive simulations on MRI datasets and other images/videos data using different missing rates and patterns. The proposed tensor CUR-based algorithms have low computational complexity and are faster than some state-of-the-art algorithms for completing data tensors with execution time being 100 times faster in some cases.

Finally, we adopt an algorithm for reconstructing MRI motion artifacts using low-rank tubal decomposition and tensor completion. The technique takes advantage of the tensor nuclear norm and "Hankelization" to reconstruct the MRI in the Fourier domain. It takes a few samples of the k-space data for reconstruction. Experiments show the method is comparable to other MRI motion reconstruction techniques.

The completion results are analyzed with PSNR, SSIM and NRMSE using other tensor completion algorithms like MDT [9], TRLRF [13] TRWOPT [32], Tucker CUR [253], SPC [49] and CPWOPT [29]. In most cases our method outperforms some of these algorithms for EEG and MRI data reconstructions and also for natural images. Our methods also perform well for other natural images.

7.1 Future Works

This research mostly focused on optimization problems which work on a single image at a time. With the advent of sophisticated deep learning technology, it is now known that it is possible to extract more meaningful features from images by using a larger sample size rather than a single image. So, the main gap in the thesis is learning on multiple images, which could potentially lead to improved recovery performance. Furthermore, the tensor completion problem can be expressed as a convex or non-convex optimization problem. The convex formulation has a high computational complexity, whereas non-convex variants are frequently faster. To the best of my knowledge, no papers on non-convex optimization problems for tensor completion problems have been published that guarantee convergence to the global minimum. A collaborative optimization problem could be used to solve this problem.

It is established with experiments that Hankelization improves the performance of reconstruction of signals and images owing to the fact that the generated Hankel data has more entries for the completion algorithm to use. However, it is time-consuming and computationally expensive to run. It is therefore, imperative to explore other techniques for convolutions as a Hankelization step to help reduce the memory and time costs in running the algorithm.

Bibliography

[1] Maame G Asante-Mensah, Salman Ahmadi-Asl, and Andrzej Cichocki. Matrix and tensor completion using tensor ring decomposition with sparse representation. *Machine Learning: Science and Technology*, 2(3):035008, 2021.

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

[3] Salman Ahmadi-Asl, Andrzej Cichocki, Anh Huy Phan, Maame G Asante-Mensah, Farid Mousavi, Ivan Oseledets, and Toshihisa Tanaka. Randomized algorithms for fast computation of low-rank tensor ring model. *Machine Learning: Science and Technology*, 2020.

[4] Maame G Asante-Mensah and Andrzej Cichocki. Medical image de-noising using deep networks. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 315–319. IEEE, 2018.

[5] Salman Ahmadi-Asl, Maame G Asante-Mensah, Andrzej Cichocki, Anh Huy Phan, Ivan Oseledets, and Jun Wang. Fast cross tensor approximation for image and video completion. *Signal Processing*, page 109121, 2023.

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

[7] Yu-Bang Zheng, Ting-Zhu Huang, Xi-Le Zhao, Qibin Zhao, and Tai-Xiang Jiang. Fully-connected tensor network decomposition and its application to higher-order tensor completion. In *Proc. AAAI*, volume 35, pages 11071–11078, 2021.

[8] Stewart Trickett, Lynn Burroughs, and Andrew Milton. Interpolation using hankel tensor completion. In *SEG Technical Program Expanded Abstracts 2013*, pages 3634–3638. Society of Exploration Geophysicists, 2013.

[9] Tatsuya Yokota, Burak Erem, Seyhmus Guler, Simon K Warfield, and Hidekata Hontani. Missing slice recovery for tensors using a low-rank model in embedded space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8251–8259, 2018.

[10] Jean-Michel Papy, Lieven De Lathauwer, and Sabine Van Huffel. Exponential data fitting using multilinear algebra: the single-channel and multi-channel case. *Numerical linear algebra with applications*, 12(8):809–826, 2005.

[11] Jordi Sole-Casals, Cesar Federico Caiafa, Qibin Zhao, and Andrzej Cichocki. Brain-computer interface with corrupted eeg data: a tensor completion approach. *Cognitive Computation*, 10(6):1062–1074, 2018.

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

[13] Longhao Yuan, Chao Li, Danilo Mandic, Jianting Cao, and Qibin Zhao. Tensor ring decomposition with rank minimization on latent space: An efficient approach for tensor completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9151–9158, 2019.

[14] Andrzej Cichocki, Scott C Douglas, and Shun-ichi Amari. Robust techniques for independent component analysis (ica) with noisy data. *Neurocomputing*, 22(1-3):113–129, 1998.

[15] Md Rakibul Mowla, Siew-Cheok Ng, Muhammad SA Zilany, and Raveendran Paramesran. Artifacts-matched blind source separation and wavelet transform for multichannel eeg denoising. *Biomedical Signal Processing and Control*, 22:111–118, 2015.

[16] Andrzej Cichocki, Hyekyoung Lee, Yong-Deok Kim, and Seungjin Choi. Non-negative matrix factorization with α-divergence. *Pattern Recognition Letters*, 29(9):1433–1440, 2008.

[17] Sergiy Vorobyov and Andrzej Cichocki. Blind noise reduction for multisensory signals using ica and subspace filtering, with application to eeg analysis. *Biological Cybernetics*, 86(4):293–303, 2002.

[18] Seungjin Choi, Andrzej Cichocki, Hyung-Min Park, and Soo-Young Lee. Blind source separation and independent component analysis: A review. *Neural Information Processing-Letters and Reviews*, 6(1):1–57, 2005.

[19] Roman Rosipal, Mark Girolami, Leonard J Trejo, and Andrzej Cichocki. Kernel pca for feature extraction and de-noising in nonlinear regression. *Neural Computing & Applications*, 10(3):231–243, 2001.

[20] Abd-Krim Seghouane and Andrzej Cichocki. Bayesian estimation of the number of principal components. *Signal Processing*, 87(3):562–568, 2007.

[21] Andrzej Cichocki. Blind signal processing methods for analyzing multichannel brain signals. *System*, 1:2, 2004.

[22] Xiaowen Xu, Qiang Wu, Shuo Wang, Ju Liu, Jiande Sun, and Andrzej Cichocki. Whole brain fmri pattern analysis based on tensor neural network. *IEEE Access*, 6:29297–29305, 2018.

[23] Giuseppe G Calvi, Bruno Scalzo Dees, and Danilo P Mandic. Tight lower bound on the tensor rank based on the maximally square unfolding. *arXiv preprint arXiv:1909.05831*, 2019.

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

[25] Misha E Kilmer and Carla D Martin. Factorization strategies for third-order tensors. *Linear Algebra and its Applications*, 435(3):641–658, 2011.

[26] Misha E Kilmer, Karen Braman, Ning Hao, and Randy C Hoover. Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging. *SIAM Journal on Matrix Analysis and Applications*, 34(1):148–172, 2013.

[27] Shengke Xue, Wenyuan Qiu, Fan Liu, and Xinyu Jin. Low-rank tensor completion by truncated nuclear norm regularization. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 2600–2605. IEEE, 2018.

[28] Zemin Zhang and Shuchin Aeron. Exact tensor completion using t-svd. *IEEE Transactions on Signal Processing*, 65(6):1511–1526, 2016.

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

[30] Silvia Gandy, Benjamin Recht, and Isao Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 27(2):025010, 2011.

[31] Tatsuya Yokota and Hidekata Hontani. Simultaneous visual data completion and denoising based on tensor rank and total variation minimization and its primal-dual splitting algorithm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3732–3740, 2017.

[32] Longhao Yuan, Jianting Cao, Xuyang Zhao, Qiang Wu, and Qibin Zhao. Higher-dimension tensor completion via low-rank tensor ring decomposition. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1071–1076. IEEE, 2018.

[33] Stanislaw Osowski, Andrzej Majkowski, and Andrzej Cichocki. Robust pca neural networks for random noise reduction of the data. In *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 3397–3400. IEEE, 1997.

[34] Wenfei Cao, Kaidong Wang, Guodong Han, Jing Yao, and Andrzej Cichocki. A robust pca approach with noise structure learning and spatial–spectral low-rank modeling for hyperspectral image restoration. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(10):3863–3879, 2018.

[35] Tatsuya Yokota and Hidekata Hontani. Simultaneous tensor completion and denoising by noise inequality constrained convex optimization. *IEEE Access*, 7:15669–15682, 2019.

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

[37] Hillar CJ Lim LH. Most tensor problems are np-hard. *J. ACM*, 60(6):45, 2013.

[38] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2013.

[39] Johann A Bengua, Ho N Phien, Hoang Duong Tuan, and Minh N Do. Efficient tensor completion for color image and video recovery: Low-rank tensor train. *IEEE Transactions on Image Processing*, 26(5):2466–2479, 2017.

[40] Longhao Yuan, Chao Li, Danilo Mandic, Jianting Cao, and Qibin Zhao. Rank minimization on tensor ring: A new paradigm in scalable tensor decomposition and completion. *arXiv preprint arXiv:1805.08468*, 2018.

[41] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[42] Ryota Tomioka, Kohei Hayashi, and Hisashi Kashima. Estimation of low-rank tensors via convex optimization. *arXiv preprint arXiv:1010.0789*, 2010.

[43] Longhao Yuan, Qibin Zhao, and Jianting Cao. High-order tensor completion for data recovery via sparse tensor-train optimization. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 1258–1262. IEEE, 2018.

[44] Marko Filipović and Ante Jukić. Tucker factorization with missing data with application to low-n-rank tensor completion. *Multidimensional systems and signal processing*, 26(3):677–692, 2015.

[45] Lars Grasedyck, Melanie Kluge, and Sebastian Kramer. Variants of alternating least squares tensor completion in the tensor train format. *SIAM Journal on Scientific Computing*, 37(5):A2424–A2450, 2015.

[46] Wenqi Wang, Vaneet Aggarwal, and Shuchin Aeron. Efficient low rank tensor ring completion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5697–5705, 2017.

[47] Qibin Zhao, Liqing Zhang, and Andrzej Cichocki. Bayesian cp factorization of incomplete tensors with automatic rank determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1751–1763, 2015.

[48] Tatsuya Yokota and Andrzej Cichocki. Tensor completion via functional smooth component deflation. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 2514–2518. IEEE, 2016.

[49] Tatsuya Yokota, Qibin Zhao, and Andrzej Cichocki. Smooth Parafac decomposition for tensor completion. *IEEE Transactions on Signal Processing*, 64(20):5423–5436, 2016.

[50] Ching-Yun Ko, Kim Batselier, Lucas Daniel, Wenjian Yu, and Ngai Wong. Fast and accurate tensor completion with total variation regularized tensor trains. *IEEE Transactions on Image Processing*, 29:6918–6931, 2020.

[51] Farnaz Sedighin, Andrzej Cichocki, Tatsuya Yokota, and Qiquan Shi. Matrix and tensor completion in multiway delay embedded space using tensor train, with application to signal reconstruction. *IEEE Signal Processing Letters*, 27:810–814, 2020.

[52] Farnaz Sedighin, Andrzej Cichocki, and Anh-Huy Phan. Adaptive rank selection for tensor ring decomposition. *IEEE Journal of Selected Topics in Signal Processing*, 15(3):454–463, 2021.

[53] Nilanjan Dey, Amira S Ashour, Waleed S Mohamed, and Nhu Gia Nguyen. Biomedical signals. In *Acoustic Sensors for Biomedical Applications*, pages 7–20. Springer, 2019.

[54] Hany Kasban, MAM El-Bendary, and DH Salama. A comparative study of medical imaging techniques. *International Journal of Information Science and Intelligent System*, 4(2):37–58, 2015.

[55] Michael A Ibrahim and AB Dublin. Magnetic resonance imaging (mri) gadolinium. 2018.

[56] N Gopinath. Tumor detection in prostate organ using canny edge detection technique. *Int. J. Pure Appl. Math*, 118:211–217, 2018.

[57] A Naveen and T Velmurugan. Identification of calcification in mri brain images by k-means algorithm. *Indian Journal of Science and Technology*, 8(29):1, 2015.

[58] Zhi-Pei Liang and Paul C Lauterbur. *Principles of magnetic resonance imaging*. SPIE Optical Engineering Press Bellingham, 2000.

[59] Phattarapong Sawangjai, Supanida Hompoonsup, Pitshaporn Leelaarporn, Supavit Kongwudhikunakorn, and Theerawit Wilaiprasitporn. Consumer grade eeg measuring sensors as research tools: A review. *IEEE Sensors Journal*, 20(8):3996–4024, 2019.

[60] Yu Zhang, Guoxu Zhou, Qibin Zhao, Jing Jin, Xingyu Wang, and Andrzej Cichocki. Spatial-temporal discriminant analysis for erp-based brain-computer interface. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 21(2):233–243, 2013.

[61] A Garcés Correa, Eric Laciar, HD Patiño, and ME Valentinuzzi. Artifact removal from eeg signals using adaptive filters in cascade. In *Journal of Physics: Conference Series*, volume 90, page 012081. IOP Publishing, 2007.

[62] Irene Winkler, Stefan Haufe, and Michael Tangermann. Automatic classification of artifactual ica-components for artifact removal in eeg signals. *Behavioral and brain functions*, 7(1):1–15, 2011.

[63] Nadia Mammone, Fabio La Foresta, and Francesco Carlo Morabito. Automatic artifact rejection from multichannel scalp eeg by wavelet ica. *IEEE Sensors Journal*, 12(3):533–542, 2011.

[64] Jose Antonio Urigüen and Begoña Garcia-Zapirain. Eeg artifact removal—state-of-the-art and guidelines. *Journal of neural engineering*, 12(3):031001, 2015.

[65] Yu Zhang, Qibin Zhao, Guoxu Zhou, Jing Jin, Xingyu Wang, and Andrzej Cichocki. Removal of eeg artifacts for bci applications using fully bayesian tensor completion. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 819–823. IEEE, 2016.

[66] Banghua Yang, Kaiwen Duan, Chengcheng Fan, Chenxiao Hu, and Jinlong Wang. Automatic ocular artifacts removal in eeg using deep learning. *Biomedical Signal Processing and Control*, 43:148–158, 2018.

[67] Najmeh Mashhadi, Abolfazl Zargari Khuzani, Morteza Heidari, and Donya Khaledyan. Deep learning denoising for eog artifacts removal from eeg signals. In *2020 IEEE Global Humanitarian Technology Conference (GHTC)*, pages 1–6. IEEE, 2020.

[68] Sangmin S Lee, Kiwon Lee, and Guiyeom Kang. Eeg artifact removal by bayesian deep learning & ica. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 932–935. IEEE, 2020.

[69] Weidong Zhou and Jean Gotman. Automatic removal of eye movement artifacts from the eeg using ica and the dipole model. *Progress in Natural Science*, 19(9):1165–1170, 2009.

[70] Andrzej Cichocki, Sergei L Shishkin, Toshimitsu Musha, Zbigniew Leonowicz, Takashi Asada, and Takayoshi Kurachi. Eeg filtering based on blind source separation (bss) for early detection of alzheimer's disease. *Clinical Neurophysiology*, 116(3):729–737, 2005.

[71] Christopher G Thomas, Richard A Harshman, and Ravi S Menon. Noise reduction in bold-based fmri using component analysis. *Neuroimage*, 17(3):1521–1537, 2002.

[72] Carrie A Joyce, Irina F Gorodnitsky, and Marta Kutas. Automatic removal of eye movement and blink artifacts from eeg data using blind component separation. *Psychophysiology*, 41(2):313–325, 2004.

[73] G Geetha and SN Geethalakshmi. Artifact removal from eeg using spatially constrained independent component analysis and wavelet denoising with otsu's thresholding technique. *Procedia Engineering*, 30:1064–1071, 2012.

[74] Andrzej Cichocki. Tensor decompositions: new concepts in brain data analysis? *Journal of the Society of Instrument and Control Engineers*, 50(7):507–516, 2011.

[75] Yu Zhang, Guoxu Zhou, Jing Jin, Qibin Zhao, Xingyu Wang, and Andrzej Cichocki. Sparse bayesian classification of eeg for brain–computer interface. *IEEE transactions on neural networks and learning systems*, 27(11):2256–2267, 2015.

[76] Yu Zhang, Guoxu Zhou, Jing Jin, Yangsong Zhang, Xingyu Wang, and Andrzej Cichocki. Sparse bayesian multiway canonical correlation analysis for eeg pattern recognition. *Neurocomputing*, 225:103–110, 2017.

[77] Ahmadreza Baghaie and Zeyun Yu. An optimization method for slice interpolation of medical images. *arXiv preprint arXiv:1402.0936*, 2014.

[78] Hosna Ghandeharion and Abbas Erfanian. A fully automatic ocular artifact suppression from eeg data using higher order statistics: Improved performance by wavelet analysis. *Medical engineering & physics*, 32(7):720–729, 2010.

[79] Shady Mohamed, Sherif Haggag, Saeid Nahavandi, and Omar Haggag. Towards automated quality assessment measure for eeg signals. *Neurocomputing*, 237:281–290, 2017.

[80] Shankha Sanyal, Sayan Nag, Archi Banerjee, Ranjan Sengupta, and Dipak Ghosh. Music of brain and music on brain: a novel eeg sonification approach. *Cognitive Neurodynamics*, 13(1):13–31, 2019.

[81] Sergey Pavlov, Alexey Artemov, Maksim Sharaev, Alexander Bernstein, and Evgeny Burnaev. Weakly supervised fine tuning approach for brain tumor segmentation problem. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1600–1605. IEEE, 2019.

[82] Ekaterina Kondrateva, Marina Pominova, Elena Popova, Maxim Sharaev, Alexander Bernstein, and Evgeny Burnaev. Domain shift in computer vision models for mri data analysis: an overview. In *Thirteenth International Conference on Machine Vision*, volume 11605, pages 126–133. SPIE, 2021.

[83] Gaochao Cui, Lihua Gui, Qibin Zhao, Andrzej Cichocki, and Jianting Cao. Bayesian cp factorization of incomplete tensor for eeg signal application. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 2170–2173. IEEE, 2016.

[84] Longhao Yuan, Qibin Zhao, Lihua Gui, and Jianting Cao. High-order tensor completion via gradient-based optimization under tensor train format. *Signal Processing: Image Communication*, 73:53–61, 2019.

[85] Xuhui Yang, Yong Xu, Yuhui Quan, and Hui Ji. Image denoising via sequential ensemble learning. *IEEE Transactions on Image Processing*, 29:5038–5049, 2020.

[86] Maame G Asante-Mensah, Salman Ahmadi-Asl, and Andrzej Cichocki. Matrix and tensor completion using tensor ring decomposition with sparse representation. *Machine Learning: Science and Technology*, 2(3):035008, 2021.

[87] Farnaz Sedighin and Andrzej Cichocki. Image completion in embedded space using multistage tensor ring decomposition. *Frontiers in Artificial Intelligence*, 4, 2021.

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

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

[90] Zhen Long, Yipeng Liu, Longxi Chen, and Ce Zhu. Low rank tensor completion for multiway visual data. *Signal Processing*, 155:301–316, 2019.

[91] Kevin T Sweeney, Tomás E Ward, and Seán F McLoone. Artifact removal in physiological signals—practices and possibilities. *IEEE transactions on information technology in biomedicine*, 16(3):488–500, 2012.

[92] Fabien Lotte, Laurent Bougrain, Andrzej Cichocki, Maureen Clerc, Marco Congedo, Alain Rakotomamonjy, and Florian Yger. A review of classification algorithms for eeg-based brain–computer interfaces: a 10 year update. *Journal of neural engineering*, 15(3):031005, 2018.

[93] Cesar F Caiafa and Andrzej Cichocki. Multidimensional compressed sensing and their applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(6):355–380, 2013.

[94] Jingyu Yang, Yuyuan Zhu, Kun Li, Jiaoru Yang, and Chunping Hou. Tensor completion from structurally-missing entries by low-TT-rankness and fiber-wise sparsity. *IEEE Journal of Selected Topics in Signal Processing*, 12(6):1420–1434, 2018.

[95] Andrzej Cichocki and Rafal Zdunek. Regularized alternating least squares algorithms for non-negative matrix/tensor factorization. In *International Symposium on Neural Networks*, pages 793–802. Springer, 2007.

[96] Guoxu Zhou, Andrzej Cichocki, and Shengli Xie. Fast nonnegative matrix/tensor factorization based on low-rank approximation. *IEEE Transactions on Signal Processing*, 60(6):2928–2940, 2012.

[97] Wakako Nakamura, Kimitaka Anami, Takeyuki Mori, Osamu Saitoh, Andrzej Cichocki, and Shun-ichi Amari. Removal of ballistocardiogram artifacts from simultaneously recorded eeg and fmri data using independent component analysis. *IEEE Transactions on Biomedical Engineering*, 53(7):1294–1308, 2006.

[98] Canyi Lu, Jiashi Feng, Shuicheng Yan, and Zhouchen Lin. A unified alternating direction method of multipliers by majorization minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):527—–541, 2018.

[99] Canyi Lu. *A Library of ADMM for Sparse and Low-rank Optimization*. National University of Singapore, June 2016. https://github.com/canyilu/LibADMM.

[100] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2012.

[101] Lieven De Lathauwer and Dimitri Nion. Decompositions of a higher-order tensor in block terms—part iii: Alternating least squares algorithms. *SIAM journal on Matrix Analysis and Applications*, 30(3):1067–1083, 2008.

[102] M Salman Asif and Ashley Prater-Bennette. Low-rank tensor ring model for completing missing visual data. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5415–5419. IEEE, 2020.

[103] Longhao Yuan, Qibin Zhao, and Jianting Cao. Completion of high order tensor data with missing entries via tensor-train decomposition. In *International Conference on Neural Information Processing*, pages 222–229. Springer, 2017.

[104] Xiawei Guo, Quanming Yao, and James Kwok. Efficient sparse low-rank tensor completion using the frank-wolfe algorithm. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[105] Jiandong Zhao, Yuan Gao, Jinjin Tang, Lingxi Zhu, and Jiaqi Ma. Highway travel time prediction using sparse tensor completion tactics and-nearest

neighbor pattern matching method. *Journal of Advanced Transportation*, 2018, 2018.

[106] Daniel Kressner, Michael Steinlechner, and Bart Vandereycken. Low-rank tensor completion by riemannian optimization. *BIT Numerical Mathematics*, 54:447–468, 2014.

[107] Hiroyuki Kasai and Bamdev Mishra. Riemannian preconditioning for tensor completion. *arXiv preprint arXiv:1506.02159*, 2015.

[108] Madhav Nimishakavi, Pratik Kumar Jawanpuria, and Bamdev Mishra. A dual framework for low-rank tensor completion. *Advances in Neural Information Processing Systems*, 31, 2018.

[109] Ziyue Li, Nurettin Dorukhan Sergin, Hao Yan, Chen Zhang, and Fugee Tsung. Tensor completion for weakly-dependent data on graph for metro passenger flow prediction. In *proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4804–4810, 2020.

[110] Bin Gao, Renfeng Peng, and Ya-xiang Yuan. Riemannian preconditioned algorithms for tensor completion via tensor ring decomposition. *arXiv preprint arXiv:2302.14456*, 2023.

[111] Hiroyuki Kasai and Bamdev Mishra. Low-rank tensor completion: a riemannian manifold preconditioning approach. In *International conference on machine learning*, pages 1012–1021. PMLR, 2016.

[112] Kyong Hwan Jin and Jong Chul Ye. Annihilating filter-based low-rank hankel matrix approach for image inpainting. *IEEE Transactions on Image Processing*, 24(11):3498–3511, 2015.

[113] Tatsuya Yokota and Hidekata Hontani. Tensor completion with shift-invariant cosine bases. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1325–1333. IEEE, 2018.

[114] Qiquan Shi, Jiaming Yin, Jiajun Cai, Andrzej Cichocki, Tatsuya Yokota, Lei Chen, Mingxuan Yuan, and Jia Zeng. Block hankel tensor arima for multiple short time series forecasting. *arXiv preprint arXiv:2002.12135*, 2020.

[115] Xudong Wang, Yuankai Wu, Dingyi Zhuang, and Lijun Sun. Low-rank hankel tensor completion for traffic speed estimation. *arXiv preprint arXiv:2105.11335*, 2021.

[116] Ryuki Yamamoto, Hidekata Hontani, Akira Imakura, and Tatsuya Yokota. Fast algorithm for low-rank tensor completion in delay-embedded space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2058–2066, 2022.

[117] Jianwei Zheng, Mengjie Qin, Honghui Xu, Yuchao Feng, Peijun Chen, and Shengyong Chen. Tensor completion using patch-wise high order hankelization and randomized tensor ring initialization. *Engineering Applications of Artificial Intelligence*, 106:104472, 2021.

[118] Honghui Xu, Jianwei Zheng, Xiaomin Yao, Yuchao Feng, and Shengyong Chen. Fast tensor nuclear norm for structured low-rank visual inpainting. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(2):538–552, 2021.

[119] MCE Rosas-Orea, M Hernandez-Diaz, V Alarcon-Aquino, and Luis Gerardo Guerrero-Ojeda. A comparative simulation study of wavelet based denoising algorithms. In *15th International Conference on Electronics, Communications and Computers (CONIELECOMP'05)*, pages 125–130. IEEE, 2005.

[120] Edson Estrada, Homer Nazeran, Gustavo Sierra, Farideh Ebrahimi, and S Kamaledin Setarehdan. Wavelet-based eeg denoising for automatic sleep stage classification. In *CONIELECOMP 2011, 21st International Conference on Electrical Communications and Computers*, pages 295–298. IEEE, 2011.

[121] Burhan Ergen. *Signal and image denoising using wavelet transform*. InTech London, UK, 2012.

[122] Adel Belouchrani and Moeness G Amin. Blind source separation based on time-frequency signal representations. *IEEE Transactions on Signal Processing*, 46(11):2888–2897, 1998.

[123] Andrzej Cichocki and Shun-ichi Amari. *Adaptive blind signal and image processing: learning algorithms and applications*. John Wiley & Sons, 2002.

[124] Pierre Comon and Christian Jutten. *Handbook of Blind Source Separation: Independent component analysis and applications*. Academic press, 2010.

[125] Guo-Duo Zhang, Xu-Hong Yang, Hang Xu, Dong-Qing Lu, and Yong-Xiao Liu. Image denoising based on support vector machine. In *2012 Spring Congress on Engineering and Technology*, pages 1–4. IEEE, 2012.

[126] Prabhpreet Kaur, Gurvinder Singh, and Parminder Kaur. A review of denoising medical images using machine learning approaches. *Current medical imaging*, 14(5):675–685, 2018.

[127] Matthew B Pontifex, Kathryn L Gwizdala, Andrew C Parks, Martin Billinger, and Clemens Brunner. Variability of ica decomposition may impact eeg signals when used to remove eyeblink artifacts. *Psychophysiology*, 54(3):386–398, 2017.

[128] Nan-kuei Chen, Arnaud Guidon, Hing-Chiu Chang, and Allen W Song. A robust multi-shot scan strategy for high-resolution diffusion weighted mri enabled by multiplexed sensitivity-encoding (muse). *Neuroimage*, 72:41–47, 2013.

[129] Mei-Lan Chu, Hing-Chiu Chang, Hsiao-Wen Chung, Trong-Kha Truong, Mustafa R Bashir, and Nan-kuei Chen. Pocs-based reconstruction of multiplexed sensitivity encoded mri (pocsmuse): a general algorithm for reducing motion-related artifacts. *Magnetic resonance in medicine*, 74(5):1336–1348, 2015.

[130] José V Manjón, Pierrick Coupé, and Antonio Buades. Mri noise estimation and denoising using non-local pca. *Medical image analysis*, 22(1):35–47, 2015.

[131] Rodrigo A Lobos, Tae Hyung Kim, W Scott Hoge, and Justin P Haldar. Navigator-free epi ghost correction with structured low-rank matrix models: New theory and methods. *IEEE transactions on medical imaging*, 37(11):2390–2402, 2018.

[132] Kyong Hwan Jin, Ji-Yong Um, Dongwook Lee, Juyoung Lee, Sung-Hong Park, and Jong Chul Ye. Mri artifact correction using sparse+ low-rank decomposition of annihilating filter-based hankel matrix. *Magnetic resonance in medicine*, 78(1):327–340, 2017.

[133] Dongwook Lee, Kyong Hwan Jin, Eung Yeop Kim, Sung-Hong Park, and Jong Chul Ye. Acceleration of mr parameter mapping using annihilating filter-based low rank hankel matrix (aloha). *Magnetic resonance in medicine*, 76(6):1848–1864, 2016.

[134] Merry Mani, Hemant Kumar Aggarwal, Vincent Magnotta, and Mathews Jacob. Improved mussels reconstruction for high-resolution multi-shot diffusion weighted imaging. *Magnetic resonance in medicine*, 83(6):2253–2263, 2020.

[135] Yuxin Hu, Xiaole Wang, Qiyuan Tian, Grant Yang, Bruce Daniel, Jennifer McNab, and Brian Hargreaves. Multi-shot diffusion-weighted mri reconstruction with magnitude-based spatial-angular locally low-rank regularization (spa-llr). *Magnetic resonance in medicine*, 83(5):1596–1607, 2020.

[136] Yuxin Hu, Evan G Levine, Qiyuan Tian, Catherine J Moran, Xiaole Wang, Valentina Taviani, Shreyas S Vasanawala, Jennifer A McNab, Bruce A Daniel, and Brian L Hargreaves. Motion-robust reconstruction of multishot diffusion-weighted images without phase estimation through locally low-rank regularization. *Magnetic resonance in medicine*, 81(2):1181–1190, 2019.

[137] Merry Mani, Mathews Jacob, Douglas Kelley, and Vincent Magnotta. Multi-shot sensitivity-encoded diffusion data recovery using structured low-rank matrix completion (mussels). *Magnetic resonance in medicine*, 78(2):494–507, 2017.

[138] HuiMing Li. Deep learning for image denoising. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 7(3):171–180, 2014.

[139] Kyong Hwan Jin, Michael T McCann, Emmanuel Froustey, and Michael Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017.

[140] Maame G Asante-Mensah and Andrzej Cichocki. Medical image de-noising using deep networks. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 315–319. IEEE, 2018.

[141] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018.

[142] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. It takes (only) two: Adversarial generator-encoder networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[143] Tatsuya Yokota, Hidekata Hontani, Qibin Zhao, and Andrzej Cichocki. Manifold modeling in embedded space: an interpretable alternative to deep image prior. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[144] Jian Sun, Huibin Li, Zongben Xu, et al. Deep admm-net for compressive sensing mri. *Advances in neural information processing systems*, 29, 2016.

[145] Juyoung Lee, Yoseob Han, Jae-Kyun Ryu, Jang-Yeon Park, and Jong Chul Ye. k-space deep learning for reference-free epi ghost correction. *Magnetic Resonance in Medicine*, 82(6):2299–2313, 2019.

[146] Hemant K Aggarwal, Merry P Mani, and Mathews Jacob. Modl: Model-based deep learning architecture for inverse problems. *IEEE transactions on medical imaging*, 38(2):394–405, 2018.

[147] Hemant K Aggarwal, Merry P Mani, and Mathews Jacob. Modl-mussels: model-based deep learning for multishot sensitivity-encoded diffusion mri. *IEEE transactions on medical imaging*, 39(4):1268–1277, 2019.

[148] Dmitry V Savostyanov, SV Dolgov, JM Werner, and Ilya Kuprov. Exact nmr simulation of protein-size spin systems using tensor train formalism. *Physical Review B*, 90(8):085139, 2014.

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

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

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

[152] Andrzej Cichocki, Namgil Lee, Ivan V Oseledets, A-H Phan, Qibin Zhao, and Danilo P Mandic. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 perspectives and challenges. *Foundations and Trends® in Machine Learning*, 9(4-5):249–429, 2016.

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

[154] Boris N Khoromskij. O (dlog n)-quantics approximation of n-d tensors in high-dimensional numerical modeling. *Constructive Approximation*, 34(2):257–280, 2011.

[155] Mike Espig, Wolfgang Hackbusch, Stefan Handschuh, and Reinhold Schneider. Optimization problems in contracted tensor networks. *Computing and Visualization in Science*, 14(6):271–285, 2011.

[156] Mike Espig, Kishore Kumar Naraparaju, and Jan Schneider. A note on tensor chain approximation. *Computing and Visualization in Science*, 15(6):331–344, 2012.

[157] Qibin Zhao, Guoxu Zhou, Shengli Xie, Liqing Zhang, and Andrzej Cichocki. Tensor ring decomposition. *arXiv preprint arXiv:1606.05535*, 2016.

[158] Steven R White. Density matrix formulation for quantum renormalization groups. *Physical Review Letters*, 69(19):2863, 1992.

[159] Steven R. White. Density-matrix algorithms for quantum renormalization groups. *Physical Review B*, 48(14):10345, 1993.

[160] Ulrich Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Annals of Physics*, 326(1):96–192, 2011.

[161] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014.

[162] Jacob C Bridgeman and Christopher T Chubb. Hand-waving and interpretive dance: an introductory course on tensor networks. *Journal of Physics A: Mathematical and Theoretical*, 50(22):223001, 2017.

[163] Giacomo Torlai, Christopher J Wood, Atithi Acharya, Giuseppe Carleo, Juan Carrasquilla, and Leandro Aolita. Quantum process tomography with unsupervised learning and tensor networks. *arXiv preprint arXiv:2006.02424*, 2020.

[164] Jacob Biamonte and Ville Bergholm. Tensor networks in a nutshell. *arXiv preprint arXiv:1708.00006*, 2017.

[165] Jacob Biamonte, Andrey Kardashin, and Alexey Uvarov. Quantum machine learning tensor network states. *arXiv preprint arXiv:1804.02398*, 2018.

[166] Andrzej Cichocki. Era of big data processing: A new approach via tensor networks and tensor decompositions. *arXiv preprint arXiv:1403.2048*, 2014.

[167] Justin Reyes and Miles Stoudenmire. A multi-scale tensor network architecture for classification and regression. *arXiv preprint arXiv:2001.08286*, 2020.

[168] J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[169] Richard A Harshman et al. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[170] Pierre Comon. Tensors: a brief introduction. *IEEE Signal Processing Magazine*, 31(3):44–53, 2014.

[171] Marouane Nazih, Khalid Minaoui, Elaheh Sobhani, and Pierre Comon. Computation of low-rank tensor approximation under existence constraint via a forward-backward algorithm. *Signal Processing*, 188:108178, 2021.

[172] Wim P Krijnen, Theo K Dijkstra, and Alwin Stegeman. On the non-existence of optimal solutions and the occurrence of "degeneracy" in the candecomp/-parafac model. *Psychometrika*, 73(3):431–439, 2008.

[173] MIKAEL Sørensen, Lieven De Lathauwer, Pierre Comon, Sylvie Icart, and Luc Deneire. Canonical polyadic decomposition with orthogonality constraints. *SIAM J. Matrix Anal. Appl*, 33(4):1190–1213, 2012.

[174] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Linear algebra*, pages 134–151. Springer, 1971.

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

[176] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[177] Peter Pippan, Steven R White, and Hans Gerd Evertz. Efficient matrix-product state method for periodic boundary conditions. *Physical Review B*, 81(8):081103, 2010.

[178] Yinchong Yang, Denis Krompass, and Volker Tresp. Tensor-train recurrent neural networks for video classification. *arXiv preprint arXiv:1707.01786*, 2017.

[179] Maxim Kuznetsov, Daniil Polykovskiy, Dmitry P Vetrov, and Alex Zhebrak. A prior of a googol gaussians: a tensor ring induced prior for generative models. In *Advances in Neural Information Processing Systems*, pages 4102–4112, 2019.

[180] Edwin Stoudenmire and David J Schwab. Supervised learning with tensor networks. In *Advances in Neural Information Processing Systems*, pages 4799–4807, 2016.

[181] Alexander Novikov, Dmitrii Podoprikhin, Anton Osokin, and Dmitry P Vetrov. Tensorizing neural networks. In *Advances in Neural Information Processing Systems*, pages 442–450, 2015.

[182] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Compressing recurrent neural network with tensor train. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 4451–4458. IEEE, 2017.

[183] Yu Pan, Jing Xu, Maolin Wang, Jinmian Ye, Fei Wang, Kun Bai, and Zenglin Xu. Compressing recurrent neural networks with tensor ring for action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4683–4690, 2019.

[184] Qibin Zhao, Masashi Sugiyama, Longhao Yuan, and Andrzej Cichocki. Learning efficient tensor representations with ring-structured networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8608–8612. IEEE, 2019.

[185] Wei He, Yong Chen, Naoto Yokoya, Chao Li, and Qibin Zhao. Hyperspectral super-resolution via coupled tensor ring factorization. *arXiv preprint arXiv:2001.01547*, 2020.

[186] Renwei Dian, Shutao Li, and Leyuan Fang. Learning a low tensor-train rank representation for hyperspectral image super-resolution. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9):2672–2683, 2019.

[187] Richard E Bellman. *Adaptive Control Processes: A Guided Tour*, volume 2045. Princeton University Press, 2015.

[188] Wenqi Wang, Vaneet Aggarwal, and Shuchin Aeron. Tensor completion by alternating minimization under the tensor train (tt) model. *arXiv preprint arXiv:1609.05587*.

[189] Karen Braman. Third-order tensors as linear operators on a space of matrices. *Linear Algebra and its Applications*, 433(7):1241–1253, 2010.

[190] David F Gleich, Chen Greif, and James M Varah. The power and arnoldi methods in an algebra of circulants. *Numerical Linear Algebra with Applications*, 20(5):809–831, 2013.

[191] Carla D Martin, Richard Shafer, and Betsy LaRue. An order-p tensor factorization with applications in imaging. *SIAM Journal on Scientific Computing*, 35(1):A474–A490, 2013.

[192] Canyi Lu, Jiashi Feng, Yudong Chen, Wei Liu, Zhouchen Lin, and Shuicheng Yan. Tensor robust principal component analysis with a new tensor nuclear norm. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):925–938, 2019.

[193] Tai-Xiang Jiang, Michael K Ng, Xi-Le Zhao, and Ting-Zhu Huang. Framelet representation of tensor nuclear norm for third-order tensor completion. *IEEE Transactions on Image Processing*, 29:7233–7244, 2020.

[194] Canyi Lu, Jiashi Feng, Zhouchen Lin, and Shuicheng Yan. Exact low tubal rank tensor recovery from gaussian measurements. *arXiv preprint arXiv:1806.02511*, 2018.

[195] Otto Debals and Lieven De Lathauwer. Stochastic and deterministic tensorization for blind signal separation. In *International Conference on Latent Variable Analysis and Signal Separation*, pages 3–13. Springer, 2015.

[196] Jong Chul Ye, Jong Min Kim, Kyong Hwan Jin, and Kiryung Lee. Compressive sampling using annihilating filter-based low-rank interpolation. *IEEE Transactions on Information Theory*, 63(2):777–801, 2016.

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

[198] Jingyu Yang, Xuemeng Yang, Xinchen Ye, and Chunping Hou. Reconstruction of structurally-incomplete matrices with reweighted low-rank and sparsity priors. *IEEE Transactions on Image Processing*, 26(3):1158–1172, 2016.

[199] Yuxin Chen and Yuejie Chi. Spectral compressed sensing via structured matrix completion. *arXiv preprint arXiv:1304.4610*, 2013.

[200] Jian Zhang, Debin Zhao, and Wen Gao. Group-based sparse representation for image restoration. *IEEE Transactions on Image Processing*, 23(8):3336–3351, 2014.

[201] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[202] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[203] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 689–696, 2009.

[204] Shuting Cai, Zhao Kang, Ming Yang, Xiaoming Xiong, Chong Peng, and Mingqing Xiao. Image denoising via improved dictionary learning with global structure and local similarity preservations. *Symmetry*, 10(5):167, 2018.

[205] Huyan Huang, Yipeng Liu, Zhen Long, and Ce Zhu. Robust low-rank tensor ring completion. *IEEE Transactions on Computational Imaging*, 6:1117–1126, 2020.

[206] Longhao Yuan, Chao Li, Jianting Cao, and Qibin Zhao. Rank minimization on tensor ring: an efficient approach for tensor decomposition and completion. *Machine Learning*, 109(3):603–622, 2020.

[207] Xi-Le Zhao, Xin Nie, Yu-Bang Zheng, Teng-Yu Ji, and Ting-Zhu Huang. Low-rank tensor completion via tensor nuclear norm with hybrid smooth regularization. *IEEE Access*, 7:131888–131901, 2019.

[208] Meng Ding, Ting-Zhu Huang, Xi-Le Zhao, and Tian-Hui Ma. Tensor completion via nonconvex tensor ring rank minimization with guaranteed convergence. *arXiv preprint arXiv:2005.09674*, 2020.

[209] Kim-Chuan Toh and Sangwoon Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of optimization*, 6(615-640):15, 2010.

[210] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Poblano v1. 0: A matlab toolbox for gradient-based optimization. *Sandia National Laboratories, Tech. Rep. SAND2010-1422*, 2010.

[211] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[212] Hossein Hassani and Dimitrios Thomakos. A review on singular spectrum analysis for economic and financial time series. *Statistics and its Interface*, 3(3):377–397, 2010.

[213] Manousos A Klados and Panagiotis D Bamidis. A semi-simulated eeg/eog dataset for the comparison of eog artifact rejection techniques. *Data in brief*, 8:1004–1006, 2016.

[214] Kevin T Sweeney, Hasan Ayaz, Tomás E Ward, Meltem Izzetoglu, Seán F McLoone, and Banu Onaral. A methodology for validating artifact removal techniques for physiological signals. *IEEE transactions on information technology in biomedicine*, 16(5):918–926, 2012.

[215] Kevin T Sweeney, Seán F McLoone, and Tomas E Ward. The use of ensemble empirical mode decomposition with canonical correlation analysis as a novel artifact removal technique. *IEEE transactions on biomedical engineering*, 60(1):97–105, 2012.

[216] Feng Duan, Hao Jia, ZhiWen Zhang, Fan Feng, Ying Tan, YangYang Dai, Andrzej Cichocki, ZhengLu Yang, Cesar F Caiafa, Zhe Sun, et al. On the robustness of eeg tensor completion methods. *Science China Technological Sciences*, 64(9):1828–1842, 2021.

[217] Miao Xu, Rong Jin, and Zhi-Hua Zhou. Cur algorithm for partially observed matrices. In *International Conference on Machine Learning*, pages 1412–1421. PMLR, 2015.

[218] Lele Wang, Kun Xie, Thabo Semong, and Huibin Zhou. Missing data recovery based on tensor-cur decomposition. *IEEE Access*, 6:532–544, 2017.

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

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

[221] Sergei A Goreinov, Eugene E Tyrtyshnikov, and Nickolai L Zamarashkin. A theory of pseudoskeleton approximations. *Linear algebra and its applications*, 261(1-3):1–21, 1997.

[222] Sergei A Goreinov, Ivan V Oseledets, Dimitry V Savostyanov, Eugene E Tyrtyshnikov, and Nikolay L Zamarashkin. How to find a good submatrix. In *Matrix Methods: Theory, Algorithms And Applications: Dedicated to the Memory of Gene Golub*, pages 247–256. World Scientific, 2010.

[223] Sergei A Goreinov and Eugene E Tyrtyshnikov. The maximal-volume concept in approximation by low-rank matrices. *Contemporary Mathematics*, 280:47–52, 2001.

[224] DV Savostyanov. *Polilinear approximation of matrices and integral equations*. PhD thesis, PhD thesis, INM RAS, Moscow, 2006.(in Russian), 2006.

[225] Eugene Tyrtyshnikov. Incomplete cross approximation in the mosaic-skeleton method. *Computing*, 64(4):367–380, 2000.

[226] Saifon Chaturantabut and Danny C Sorensen. Discrete empirical interpolation for nonlinear model reduction. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 4316–4321. IEEE, 2009.

[227] Danny C Sorensen and Mark Embree. A DEIM induced CUR factorization. *SIAM Journal on Scientific Computing*, 38(3):A1454–A1482, 2016.

[228] David Anderson, Simon Du, Michael Mahoney, Christopher Melgaard, Kunming Wu, and Ming Gu. Spectral gap error bounds for improving CUR matrix decomposition and the nyström method. In *Artificial Intelligence and Statistics*, pages 19–27, 2015.

[229] A Yu Mikhalev and Ivan V Oseledets. Iterative representing set selection for nested cross approximation. *Numerical Linear Algebra with Applications*, 23(2):230–248, 2016.

[230] MV Rakhuba and Ivan V Oseledets. Fast multidimensional convolution in low-rank tensor formats via cross approximation. *SIAM Journal on Scientific Computing*, 37(2):A565–A582, 2015.

[231] Shusen Wang and Zhihua Zhang. Improving CUR matrix decomposition and the nyström approximation via adaptive sampling. *The Journal of Machine Learning Research*, 14(1):2729–2769, 2013.

[232] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling techniques for the Nyström method method. In *Artificial Intelligence and Statistics*, pages 304–311, 2009.

[233] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the nyström method. *Journal of Machine Learning Research*, 13(Apr):981–1006, 2012.

[234] Vin D Silva and Joshua B Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in neural information processing systems*, pages 721–728, 2003.

[235] Christopher KI Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

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

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

[238] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

[239] Amit Deshpande and Santosh Vempala. Adaptive sampling and fast low-rank matrix approximation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 292–303. Springer, 2006.

[240] Amit Deshpande, Luis Rademacher, Santosh Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(1):225–247, 2006.

[241] Amit Deshpande and Luis Rademacher. Efficient volume sampling for row/-column subset selection. In *2010 ieee 51st annual symposium on foundations of computer science*, pages 329–338. IEEE, 2010.

[242] Venkatesan Guruswami and Ali Kemal Sinop. Optimal column-based low-rank matrix reconstruction. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1207–1214. SIAM, 2012.

[243] An Mai, Loc Tran, Linh Tran, and Nguyen Trinh. Vgg deep neural network compression via SVD and CUR decomposition techniques. In *2020 7th NAFOSTED Conference on Information and Computer Science (NICS)*, pages 118–123. IEEE, 2020.

[244] Emily P Hendryx, Béatrice M Rivière, Danny C Sorensen, and Craig G Rusin. Finding representative electrocardiogram beat morphologies with CUR. *Journal of biomedical informatics*, 77:97–110, 2018.

[245] HanQin Cai, Keaton Hamm, Longxiu Huang, Jiaqi Li, and Tao Wang. Rapid robust principal component analysis: CUR accelerated inexact low rank estimation. *IEEE Signal Processing Letters*, 2020.

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

[247] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast Monte Carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on computing*, 36(1):158–183, 2006.

[248] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36(1):184–206, 2006.

[249] Changsheng Li, Xiangfeng Wang, Weishan Dong, Junchi Yan, Qingshan Liu, and Hongyuan Zha. Joint active learning with feature selection via CUR matrix decomposition. *IEEE transactions on pattern analysis and machine intelligence*, 41(6):1382–1396, 2018.

[250] Akram Aldroubi, Keaton Hamm, Ahmet Bugra Koku, and Ali Sekmen. CUR decompositions, similarity matrices, and subspace clustering. *Frontiers in Applied Mathematics and Statistics*, 4:65, 2019.

[251] Eleni Drinea, Petros Drineas, and Patrick Huggins. A randomized singular value decomposition algorithm for image processing applications. In *Proceedings of the 8th panhellenic conference on informatics*, pages 278–288. Citeseer, 2001.

[252] Ivan V Oseledets, DV Savostianov, and Eugene E Tyrtyshnikov. Tucker dimensionality reduction of three-dimensional arrays in linear time. *SIAM Journal on Matrix Analysis and Applications*, 30(3):939–956, 2008.

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

[254] Shmuel Friedland, V Mehrmann, A Miedlar, and M Nkengla. Fast low rank approximations of matrices and tensors. *The Electronic Journal of Linear Algebra*, 22, 2011.

[255] Michael W Mahoney, Mauro Maggioni, and Petros Drineas. Tensor-CUR decompositions for tensor-based data. *SIAM Journal on Matrix Analysis and Applications*, 30(3):957–987, 2008.

[256] HanQin Cai, Keaton Hamm, Longxiu Huang, and Deanna Needell. Mode-wise tensor decompositions: Multi-dimensional generalizations of CUR decompositions. *arXiv preprint arXiv:2103.11037*, 2021.

[257] Davoud Ataee Tarzanagh and George Michailidis. Fast randomized algorithms for t-product based tensor operations and decompositions with applications to imaging data. *SIAM Journal on Imaging Sciences*, 11(4):2629–2664, 2018.

[258] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[259] Zisen Fang, Xiaowei Yang, Le Han, and Xiaolan Liu. A sequentially truncated higher order singular value decomposition-based algorithm for tensor completion. *IEEE transactions on cybernetics*, 49(5):1956–1967, 2018.

[260] Ledyard R Tucker et al. The extension of factor analysis to three-dimensional matrices. *Contributions to mathematical psychology*, 110119, 1964.

[261] Abraham Savitzky and Marcel JE Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

[262] William S Cleveland. Lowess: A program for smoothing scatterplots by robust locally weighted regression. *American Statistician*, 35(1):54, 1981.

[263] William S Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836, 1979.

[264] Rao Veerabhadra Garimella. A simple introduction to moving least squares and local regression estimation. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2017.

[265] Matt A Bernstein, Kevin F King, and Xiaohong Joe Zhou. *Handbook of MRI pulse sequences*. Elsevier, 2004.

[266] Yi Wang, Roger C Grimm, Joel P Felmlee, Sephen J Riederer, and Richard L Ehman. Algorithms for extracting motion information from navigator echoes. *Magnetic resonance in medicine*, 36(1):117–123, 1996.

[267] Mark Hedley, HONG Yan, and Dov Rosenfeld. Motion artifact correction in mri using generalized projections. *IEEE transactions on medical imaging*, 10(1):40–46, 1991.

[268] Alfred Stadler, Wolfgang Schima, Ahmed Ba-Ssalamah, Joachim Kettenbach, and Edith Eisenhuber. Artifacts in body mr imaging: their appearance and how to eliminate them. *European radiology*, 17(5):1242–1255, 2007.

[269] Sajan Goud Lingala, Yue Hu, Edward DiBella, and Mathews Jacob. Accelerated dynamic mri exploiting sparsity and low-rank structure: kt slr. *IEEE transactions on medical imaging*, 30(5):1042–1054, 2011.

[270] Sajan Goud Lingala and Mathews Jacob. Blind compressive sensing dynamic mri. *IEEE transactions on medical imaging*, 32(6):1132–1145, 2013.

[271] Benjamin Trémoulhéac, Nikolaos Dikaios, David Atkinson, and Simon R Arridge. Dynamic mr image reconstruction–separation from undersampled (

$$\{$$

$\backslash$bf k$\}$, t)$-spacevialow-rankplussparseprior.IEEEtransactionsonmedicalimaging, 33(8) : 1689 - -1701, 2014.$

[272] Ben A Duffy, Wenlu Zhang, Haoteng Tang, Lu Zhao, Meng Law, Arthur W Toga, and Hosung Kim. Retrospective correction of motion artifact affected structural mri images using deep learning of simulated motion. 2018.

[273] Yipeng Liu, Tengteng Liu, Jiani Liu, and Ce Zhu. Smooth robust tensor principal component analysis for compressed sensing of dynamic mri. *Pattern Recognition*, 102:107252, 2020.

[274] Yuxin Hu and Brian A Hargreaves. Multi-shot diffusion-weighted mri reconstruction using unrolled network with u-net as priors, January 26 2021. US Patent 10,901,059.

[275] Jong Chul Ye. Compressed sensing mri: a review from signal processing perspective. *BMC Biomedical Engineering*, 1(1):1–17, 2019.

[276] Michael Lustig, David Donoho, and John M Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[277] Klaas P Pruessmann, Markus Weiger, Markus B Scheidegger, and Peter Boesiger. Sense: sensitivity encoding for fast mri. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 42(5):952–962, 1999.

[278] Marco Signoretto, Raf Van de Plas, Bart De Moor, and Johan AK Suykens. Tensor versus matrix completion: A comparison with application to spectral data. *IEEE Signal Processing Letters*, 18(7):403–406, 2011.

[279] Canyi Lu, Xi Peng, and Yunchao Wei. Low-rank tensor completion with a new tensor nuclear norm induced by invertible linear transforms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5996–6004, 2019.

[280] Guangjing Song, Michael K Ng, and Xiongjun Zhang. Robust tensor completion using transformed tensor singular value decomposition. *Numerical Linear Algebra with Applications*, 27(3):e2299, 2020.

[281] Jiawen Yao, Zheng Xu, Xiaolei Huang, and Junzhou Huang. An efficient algorithm for dynamic mri using low-rank and total variation regularizations. *Medical image analysis*, 44:14–27, 2018.