

**UNICAMP**

**UNIVERSIDADE ESTADUAL DE  
CAMPINAS**

**Instituto de Matemática, Estatística e  
Computação Científica**

**JOÃO LUIZ SANTOS GOMES**

**Noções distintas de posto para tensores e  
implicações práticas envolvendo o problema de  
completamento**

Campinas

2023

João Luiz Santos Gomes

## **Noções distintas de posto para tensores e implicações práticas envolvendo o problema de completamento**

Dissertação apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Matemática Aplicada e Computacional.

Orientadora: Sandra Augusta Santos

Este trabalho corresponde à versão final da Dissertação defendida pelo aluno João Luiz Santos Gomes e orientada pela Profa. Dra. Sandra Augusta Santos.

Campinas  
2023

A ficha catalográfica deverá ser solicitada online via <http://www.sbu.unicamp.br/sbu/elaboracao-de-ficha-catalografica/>

A folha de aprovação será fornecida pela Secretaria de Pós-Graduação

# Agradecimentos

Agradeço imensamente à minha orientadora Sandra, por toda paciência, carinho e atenção. Obrigado pelos conselhos e conversas que tivemos durante este período. Certamente aprendi muito ao seu lado e você é um grande exemplo de ser humano e profissional para mim. Também agradeço aos professores Cristiano, Renato e Ricardo pelos valiosos comentários feitos durante meu exame de qualificação que enriqueceram este trabalho.

À minha mãe Verônica, por estar sempre me apoiando aonde eu fosse. Ao meu pai Edinei por sempre estar ao meu lado acreditando em mim. Ao meu irmão Guilherme, com quem aprendi muito. À minha companheira Amanda, que me ajudou sempre a prosseguir nos momentos difíceis. Aos meus tios Luiz e Isabel por bons conselhos e ensinamentos. Às minhas avós Rosa e Chica e aos meus avôs João (*in memorian*) e Luiz (*in memorian*), com quem eu gostaria de compartilhar minhas conquistas e histórias recentes.

Agradeço aos meus amigos Vinícius, Pedro e Felipe, pessoas com as quais compartilhei dos melhores momentos na universidade. Nossas conversas, brincadeiras e jogatinas me fizeram muito feliz. Certamente vocês fizeram parte deste trabalho. Também agradeço à minha amiga Giovanna por toda a ajuda e apoio. Agradeço aos meus amigos de Sorocaba, Nicole, Felipe, Vitor e Roberto, por todo encorajamento.

Aos meus professores do ensino fundamental, médio e superior, em especial à professora Márcia Ruggiero, com quem troquei muitas ideias e aprendi muito.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 - Processo 88887.610062/2021-0.

# Resumo

Este trabalho tem como objetivo apresentar conceitos introdutórios no que diz respeito à álgebra linear computacional de tensores. Em específico, exibimos noções de posto para tensores e formas de representação a partir do produto tensorial de fatores de menor dimensão. Em especial, focamos nossos estudos na decomposição *Tensor Train* (TT) e no chamado posto-TT, o qual não sofre das desvantagens que as fatorações clássicas possuem, pois o número de parâmetros estimados cresce linearmente com a ordem. Este fato qualifica a decomposição TT como uma poderosa ferramenta para tratar problemas de grandes dimensões. Além disso, aplicamos as noções de posto mencionadas ao problema de completamento tensorial, o qual pode ser resolvido por técnicas de otimização aplicadas a modelos apropriados. Neste caso, a escolha da noção de posto e o ajuste adequado dos parâmetros são fundamentais para obtermos uma solução de boa qualidade. Por fim, investigamos aproximações de baixo posto-TT com o uso da proposta de método apresentada neste trabalho. Contando com a solução de subproblemas a partir de métodos baseados no vetor gradiente, propusemos uma forma de reconstruir o tensor com entradas faltantes, por meio do aumento dinâmico do posto-TT.

**Palavras-chave:** posto tensorial, decomposição tensorial, tensor train, completamento tensorial, completamento de alta ordem.

# Abstract

This work aims to present introductory concepts regarding computational linear algebra of tensors. Specifically, we recall notions of rank for tensors and forms of representation based on the tensor product of lower-dimensional factors. In particular, we focus our studies on the *Tensor Train* (TT) decomposition and on the so-called TT-rank, which does not suffer from the drawbacks that classical factorizations have, as the number of estimated parameters grows linearly with the order. This fact qualifies the TT decomposition as a powerful tool to deal with high-dimensional problems. Furthermore, we apply the related notions of rank to address the so-called *tensor completion problem*, which can be solved by optimization techniques applied to an appropriate model. In this case, the choice of the rank definition and adequate parameters are fundamental to the quality of the solution that approximates the original tensor. Finally, we investigated low TT-rank approximations using the proposed method presented in this work. Relying on the solution of subproblems of methods based on the gradient vector, we proposed a way to reconstruct the tensor with missing inputs, by means of the dynamic increase of the TT-rank.

**Keywords:** tensor rank, tensor decomposition, tensor train, tensor completion, higher-order completion problem.

# Listas de ilustrações

Figura 1 – Fatiamento de um tensor de ordem-3 . . . . .	20
Figura 2 – Fibras de um tensor de ordem-3 . . . . .	20
Figura 3 – Ilustração de desdobramento de modo-2 . . . . .	22
Figura 4 – Decomposição em tensores de posto-1 de ordem 3 . . . . .	25
Figura 5 – Decomposição de Tucker de ordem 3. . . . .	33
Figura 6 – Decomposição HOSVD truncada de ordem 3 . . . . .	34
Figura 7 – Figura colorida . . . . .	37
Figura 8 – Imagem original e duas reconstruções. . . . .	37
Figura 9 – Representação gráfica da decomposição TT de um tensor . . . . .	39
Figura 10 – Passo a passo da decomposição TT-SVD de um tensor. . . . .	42
Figura 11 – Ilustração para reconstrução de imagem minimizando o posto. . . . .	60
Figura 12 – Procedimento de <i>ket augmentation</i> . . . . .	73
Figura 13 – Ilustração para o processo de KA em uma imagem de dimensões $256 \times 256 \times 3$ . . . . .	73
Figura 14 – Figuras utilizadas nos Experimentos III e IV. . . . .	82
Figura 15 – Legenda que acompanha os resultados dos experimentos. . . . .	85
Figura 16 – Resultados para RSE obtidos pelo Experimento I. . . . .	86
Figura 17 – Resultados para RSE obtidos pelo Experimento II. . . . .	87
Figura 18 – Resultados para RSE e PSNR obtidos pelo Experimento III. . . . .	88
Figura 19 – Resultados para a imagem <i>peppers</i> sem <i>Ket Augmentation</i> . . . . .	88
Figura 20 – Resultados para a imagem <i>woman</i> sem <i>Ket Augmentation</i> . . . . .	89
Figura 21 – Resultados para RSE e PSNR obtidos pelo Experimento IV. . . . .	90
Figura 22 – Resultados para a imagem <i>peppers</i> com <i>Ket Augmentation</i> . . . . .	91
Figura 23 – Resultados para a imagem <i>peppers</i> com <i>Ket Augmentation</i> . . . . .	91
Figura 24 – Reconstrução do sinal unidimensional com TT-WOPT-DU. . . . .	94
Figura 25 – Reconstrução do sinal gerado por $z_1$ com TT-WOPT-DU. . . . .	96
Figura 26 – Reconstrução do sinal gerado por $z_2$ com TT-WOPT-DU. . . . .	96
Figura 27 – Visualização das fatias geradas pelos dados sísmicos simulados. . . . .	98
Figura 28 – Visualização da reconstrução produzida pelo TT-WOPT-DU com $\rho = 0.01$ . . . . .	99
Figura 29 – Novas figuras a serem consideradas para o completamento. . . . .	99
Figura 30 – Imagens reconstruídas pelo método TT-WOPT-DU. . . . .	100
Figura 31 – Valores singulares dos desdobramentos de $\mathcal{X}$ . . . . .	101
Figura 32 – Valores singulares do desdobramento de $\mathcal{Y}$ . . . . .	102

# Listas de tabelas

Tabela 1	– Tempo necessário para recompressão de $\mathcal{A}$ no formato TT de acordo com $N$ .	51
Tabela 2	– Resultados da integração utilizando o produto por contração.	54
Tabela 3	– Resultados da integração utilizando a representação TT.	54
Tabela 4	– Panorama das estratégias consideradas neste trabalho, com as respectivas nomenclaturas, referências e principais características.	67
Tabela 5	– RSE das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional $f$ amostrado em 4096 pontos.	92
Tabela 6	– Posto-TT das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional $f$ amostrado em 4096 pontos com 30% e 70% de dados faltantes.	93
Tabela 7	– RSE das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional $f$ amostrado em 531441 pontos.	94
Tabela 8	– Posto-TT das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional $f$ amostrado em 531441 pontos com 70% e 90% de dados faltantes.	94
Tabela 9	– Resultados para reconstrução do sinal bidimensional gerado por $z_1$ .	95
Tabela 10	– Resultados para reconstrução do sinal bidimensional gerado por $z_2$ .	95
Tabela 11	– Resultados para a reconstrução do tensor gerado pelo sinal multidimensional.	97
Tabela 12	– Resultados obtidos pelo método TT-WOPT-DU com $\rho = 0.01$ para reconstrução de imagens.	100

# Lista de abreviaturas e siglas

SVD	<i>Singular Value Decomposition</i>
PCA	<i>Principal Component Analysis</i>
RSE	<i>Root Squared Error</i>
PSNR	<i>Peak Signal-to-Noise Ratio</i>
PARAFAC	<i>Parallel Factor Decomposition</i>
ALS	<i>Alternate Least Squares</i>
HOSVD	<i>High Order Singular Value Decomposition</i>
TT	<i>Tensor Train</i>
TT-SVD	<i>Tensor Train Singular Value Decomposition</i>
SVT	<i>Singular Value Thresholding</i>
TNNR	<i>Truncated Nuclear Norm Regularization</i>
ADMM	<i>Alternating Direction Method of Multipliers</i>
SiLRTC	<i>Simple Low Rank Tensor Completion</i>
HaLRTC	<i>High Accuracy Low Rank Tensor Completion</i>
KA	<i>Ket Augmentation</i>
SiLRTC-TT	<i>Simple Low Rank Tensor Completion via Tensor Train</i>
TMac	<i>Tensor Completion by Parallel Matrix Factorization</i>
TMac-TT	<i>Tensor Completion by Parallel Matrix Factorization via Tensor Train</i>
TT-WOPT	<i>Tensor Train Weighted Optimization</i>
TT-WOPT-DU	<i>Tensor Train Weighted Optimization with Dynamic Updating of the TT-rank</i>

# Listas de símbolos

$\emptyset$	Conjunto vazio
$\mathbb{R}$	Conjunto dos números reais
$\mathbb{R}^n$	Conjunto dos vetores reais de dimensão $n$
$\mathbb{R}^{m \times n}$	Conjunto das matrizes reais de dimensões $m \times n$
$\mathbb{R}^{I_1 \times I_2 \times \dots \times I_n}$	Conjunto dos tensores reais de dimensões $I_1 \times I_2 \times \dots \times I_n$
$\mathbb{C}$	Conjunto dos números complexos
$\mathbb{C}^n$	Conjunto dos vetores complexos de dimensão $n$
$\mathbb{C}^{m \times n}$	Conjunto das matrizes complexas de dimensões $m \times n$
$\mathbb{C}^{I_1 \times I_2 \times \dots \times I_n}$	Conjunto dos tensores complexos de dimensões $I_1 \times I_2 \times \dots \times I_n$
$\mathbf{I}_{n \times n}$	Matriz identidade de ordem $n$
$\mathbf{A}^{-1}$	Inversa de $\mathbf{A}$
$\cdot^T$	Transposto
$\cdot^\dagger$	Pseudoinversa
$\circ$	Produto externo ou tensorial
$\otimes$	Produto de Kronecker
$\odot$	Produto de Khatri-Rao
$*$	Produto de Hadamard
$\langle \cdot, \cdot \rangle$	Produto interno
$\ \cdot\ _F$	Norma de Frobenius
$\ \cdot\ _*$	Norma nuclear
$\ \cdot\ _p$	Norma nuclear truncada

$\text{tr}$	Traço matricial
$\det$	Determinante matricial
$\mathcal{A}_{(n)}$	Desdobramento de modo- $n$ do tensor $\mathcal{A}$ na noção de posto-multilinear
$\mathcal{A}_{<n>}$	Desdobramento de modo- $n$ do tensor $\mathcal{A}$ na noção de posto-TT
$\times_n$	Produto de modo- $n$
$\Delta(\mathcal{X})$	Hiperdeterminante de um tensor $\mathcal{X}$ de dimensões $2 \times 2 \times 2$
$\sigma_n(\mathbf{A})$	$n$ -ésimo valor singular da matriz $\mathbf{A}$
$\varepsilon$	Tolerância utilizada em diversos métodos
$\varepsilon_\Gamma(\mathcal{X})$	Erro relativo da aproximação $\mathcal{X}$ calculado sobre o conjunto $\Gamma$
$\delta$	Limitante da SVD truncada
$A \leftarrow B$	Atribuição de $B$ em $A$
$\mathcal{I}, \mathcal{J}$	Conjuntos de índices
$\Omega, \Gamma$	Conjuntos de índices de entradas conhecidas
$\Omega^C$	Conjunto de índices de entradas desconhecidas
$\mathcal{P}_\Omega$	Projetor ortogonal sobre o conjunto $\Omega$
$\mathcal{D}_\tau$	Operador de <i>Soft Shrinkage</i> com parâmetro $\tau$
$\partial h$	Conjunto subdiferencial da função $h$
$\mathcal{L}$	Lagrangiano
$\mathcal{L}_\beta$	Lagrangiano aumentado com parâmetro de penalização $\beta$
$\boldsymbol{\alpha}$	Vetor de pesos de uma combinação convexa
$\lambda$	Parâmetro do método TMac-TT
$\gamma, \beta$	Parâmetros dos métodos SiLRTC, SiLRTC-TT e HaLRTC
$\rho, \eta$	Parâmetros de aceite do incremento do posto-TT

# **Lista de Algoritmos**

Algoritmo 1 – PARAFAC VIA ALS . . . . .	30
Algoritmo 2 – HOSVD . . . . .	35
Algoritmo 3 – TT-SVD . . . . .	48
Algoritmo 4 – TTNR-ADMM . . . . .	65
Algoritmo 5 – SiLRTC . . . . .	69
Algoritmo 6 – HALRTC . . . . .	71
Algoritmo 7 – SiLRTC-TT . . . . .	75
Algoritmo 8 – TMAC-TT . . . . .	77
Algoritmo 9 – TT-WOPT . . . . .	78
Algoritmo 10 – TT-WOPT-DU . . . . .	80
Algoritmo 11 – APROXIMAÇÃO CRUZADA PARA DECOMPOSIÇÃO ESQUELETAL . . .	115

# Sumário

<b>Introdução</b>	<b>15</b>
<b>1 Decomposições Tensoriais</b>	<b>18</b>
1.1 Álgebra Multilinear Básica	19
1.2 Decomposições Clássicas	27
1.3 Decomposição <i>Tensor Train</i>	38
<b>2 Completamento Tensorial</b>	<b>58</b>
2.1 Completamento de Matrizes	59
2.2 Completamento de Ordem Superior	66
<b>3 Experimentos Numéricos</b>	<b>81</b>
3.1 Detalhes de implementação e escolhas dos parâmetros	82
3.2 Resultados	85
3.3 Experimentos envolvendo TT-WOPT-DU	90
<b>4 Considerações Finais</b>	<b>103</b>
<b>Referências</b>	<b>106</b>
<b>APÊNDICE A Produtos Matriciais</b>	<b>111</b>
<b>APÊNDICE B Decomposição Esqueletal</b>	<b>113</b>

# Introdução

Na era do *Big Data*, existem diversas maneiras de lidar e organizar os dados. As formas mais comuns de trabalho são feitas com base na área de álgebra matricial. Para esta, décadas de desenvolvimento matemático e algorítmico se passaram para que fossem atingidos resultados significativos através da Decomposição em Valores Singulares (SVD) e da Análise de Componentes Principais (PCA). Além disto, refinamentos foram feitos e resultados mais profundos foram obtidos de maneira que diversas áreas como, processamento de imagens, análise numérica, neurociência, análise de sinais, aprendizado de máquina, entre outras, se beneficiaram.

À medida que estruturas de dados tornaram-se cada vez maiores e mais complexas, tornou-se necessário ampliar os graus de liberdade, isto é, a análise matricial, com apenas dois graus ou índices, passou a ser um fator limitante. Em diversas aplicações, os bancos de dados possuem mais de dois graus de liberdade, por exemplo, a concentração de uma substância com coordenadas  $(x, y)$  variando no tempo  $t$ . É possível realizar a abordagem matricialmente reestruturando o conjunto de dados. Todavia, tal mudança pode não fazer sentido, por exemplo, quando as grandezas envolvidas possuem diferentes unidades de medida. Para lidar com estas questões, o estudo de objetos multidimensionais chamados tensores se tornou necessário.

O ramo da álgebra multilinear ou tensorial já é bem conhecido entre os matemáticos e físicos, para os quais o objetivo é compreender como as grandezas físicas são modificadas através de mudanças de base, onde os tensores representam transformações multilineares entre espaços vetoriais. Em espaços de dimensão finita, estas transformações podem ser escritas através de um *array*-multidimensional.

Neste trabalho, iremos considerar um tensor de ordem  $N$ , um *array* de  $N$  modos ou graus, que é elemento de um produto tensorial de  $N$  espaços vetoriais. Assim como na álgebra matricial, queremos estudar possíveis decomposições tensoriais de maneira a extrair informações importantes dos dados. Para isso, será necessário definir operações importantes entre tensores para abordar desde as decomposições mais clássicas até as mais recentes. Com o propósito de enxergar as diferenças e sutilezas inerentes a este tema, focaremos nossos estudos em tensores de ordem 3. Em comparação com as matrizes, que podem ser vistas como tensores de ordem 2, o acréscimo de apenas uma ordem já é

suficiente para percebermos a necessidade de novos conceitos para acomodar a extensão introduzida.

O estudo acerca das decomposições tensoriais se inicia em 1927 com os trabalhos de Hitchcock [24] e até hoje recebe muita atenção, principalmente depois das ideias propostas por Tucker em 1963 [49] e Carroll e Chang em 1970 [7]. Algumas destas ideias serão expostas aqui como decomposições clássicas e servem de base para ferramentas mais aperfeiçoadas. O primeiro conceito de posto tensorial é abordado em conjunto com a decomposição em fatores paralelos (PARAFAC), que embora seja ótima em termos de consumo de memória, é numericamente instável [29]. Outra representação obtida pela decomposição SVD de alta ordem (HOSVD), é capaz de caracterizar o tensor utilizando algoritmos robustos e não iterativos [31]. Em 2011, Oseledets publicou um trabalho sobre a chamada representação *Tensor Train* (TT) de tensores, sendo esta mais flexível e robusta em diversas situações [39]. Deste modo, muitos problemas no ramo da álgebra linear computacional puderam ser abordados através das decomposições tensoriais, como equações diferenciais parciais de alta ordem [28, 43] e solução de sistemas lineares de grandes dimensões [40]. Outras aplicações numéricas envolvendo tensores podem ser encontradas em [9, 10, 14, 36, 50].

No caso deste trabalho, tratamos do chamado problema de completamento de tensores e possíveis métodos existentes que buscam resolvê-lo. Usualmente, conjuntos de dados possuem entradas faltantes pelas mais diversas razões e é desejável inferir quais são os valores desconhecidos. Supondo que o tensor original seja de baixo posto, é possível, em certas condições, recuperá-lo com boa precisão. A aproximação é encontrada a partir da formulação de um problema de otimização. Aplicações envolvendo completamento de tensores podem ser encontradas em [2, 34, 35, 46].

Um dos grandes interesses deste trabalho diz respeito ao método de completamento via otimização ponderada (TT-WOPT) [55], o qual se baseia na noção de posto *Tensor Train*. O algoritmo tem como ideia procurar os fatores da decomposição TT de um tensor, mas sem calculá-los explicitamente. Para isso, é possível fazer uso do vetor gradiente e de métodos de busca baseados em tal direção. Uma das propostas deste trabalho é testar o método considerando diferentes estruturas de dados, por exemplo, valores gerados por um sinal ou imagens. Entretanto, é necessário que o posto-TT associado à decomposição, seja inicializado previamente, algo que pode afetar a flexibilidade e a utilização do método quando modificamos a instância do problema, pois o posto, em geral, não é conhecido. Desse modo, baseando-se no trabalho de Steinlechner [46], propusemos uma atualização dinâmica do vetor de posto-TT a fim de contornar essa dificuldade. Esta proposta será denominada *Tensor Train Weighted Optimization with Dynamic Updating of the TT-rank* (TT-WOPT-DU)

Cada um dos métodos inseridos neste trabalho foi construído sobre uma noção

de posto e extensões dos casos matriciais. Para isso, no Capítulo 1, é desenvolvida a álgebra linear tensorial básica necessária para compreensão dos algoritmos e das noções clássicas de posto tensorial e posto-multilinear. Ainda, apresentamos a decomposição TT e demonstramos teoremas que garantem a melhor aproximação de posto-TT fixo. Cálculo de soma, produto interno, norma e outras operações neste formato foram detalhadas. Por fim, descrevemos outra forma de estimar os núcleos-TT através da representação TT-*Cross*. A ideia é substituir o uso de decomposições SVD de matrizes associadas aos tensores, por suas decomposições esqueletais. Com o intuito de auxiliar o leitor na compreensão das ideias expostas, muitos exemplos são apresentados.

No Capítulo 2, descrevemos inicialmente o problema de completamento para o caso matricial. Dada uma matriz com entradas faltantes, gostaríamos de estimá-las a partir do conjunto de entradas conhecidas. A abordagem é feita resolvendo um problema de otimização envolvendo o posto, que é capaz de capturar as possíveis correlações entre as entradas da matriz. Contudo, devido à sua natureza combinatória, resolvemos uma formulação envolvendo a norma nuclear: uma função convexa. Ainda, o operador de *soft thresholding* é definido e este servirá de base para os métodos que minimizam esta função. Todas estas ideias são estendidas na apresentação do problema considerando tensores de ordens superiores junto a uma diversidade de métodos da literatura. Uma tabela é exibida, resumindo os principais métodos encontrados na literatura. Por fim, apresentamos o processo de *Ket Augmentation* (KA), que mapeia as entradas de um tensor de baixa ordem para outro de ordem maior sem alterar o número de elementos. Este procedimento, devido ao maior número de núcleos-TT, aprimora o desempenho dos métodos baseados na noção de posto-TT. Ademais, também introduzimos o método TT-WOPT-DU.

No Capítulo 3, levamos em conta estruturas de dados sintéticas e reais. Para esta última, utilizamos imagens coloridas de dimensões  $256 \times 256$ . Em cada uma das instâncias, a descrição dos parâmetros dos métodos e suas possíveis escolhas foram apresentadas detalhadamente. Comparamos os métodos apresentados e exibimos os resultados. Além disso, realizamos testes com método TT-WOPT-DU. Por fim, o Capítulo 4 contém as conclusões finais e propostas futuras.

Existe uma ampla diversidade de pacotes computacionais para o manuseio eficiente de tensores, inclusive no formato TT. Neste trabalho, foram utilizados pacotes da linguagem Python e Julia com o intuito de introduzir ao leitor tais ferramentas. As implementações dos métodos que geraram os resultados desta dissertação foram feitas em Julia e encontram-se em um repositório do Github <sup>1</sup>.

---

<sup>1</sup> Disponível em [https://github.com/Joaoluiz87/completamento\\_dissertacao\\_mestrado](https://github.com/Joaoluiz87/completamento_dissertacao_mestrado)

# Capítulo 1

## Decomposições Tensoriais

Em aplicações envolvendo um conjunto de dados organizado em uma matriz, é possível representar as informações utilizando, por exemplo, a decomposição em valores singulares (SVD). Deste modo, os dados são reescritos como uma soma de matrizes de posto 1 e o número de parcelas é o posto da matriz original. Todavia, é possível truncar esta representação utilizando menos parcelas e, desse modo, obtemos uma aproximação do conjunto de dados a partir de menos parâmetros armazenados. Ocasionalmente, esta matriz pode ser bem aproximada por uma estrutura de baixo posto e assim, por meio desta, encontramos informações úteis que estavam ocultas anteriormente. Desse modo, o posto de uma matriz carrega consigo a quantidade de parâmetros e a memória computacional necessários para representar todas as entradas. Assim, no estudo de conjunto de dados de grandes dimensões, a aproximação de baixo posto torna-se necessária.

Para o caso tensorial, à medida que a ordem  $N$  do tensor aumenta, o número de entradas cresce exponencialmente. Esta dificuldade é conhecida na literatura como *maldição da dimensionalidade* [40] e impede o armazenamento explícito do tensor. Logo, tal como no caso matricial, desejamos representar os tensores por meio de um esquema de baixo posto, caso seja possível.

Neste capítulo introdutório, vamos expor as operações básicas da álgebra tensorial e as fatorações tensoriais clássicas como a decomposição por fatores paralelos (PARAFAC) e a SVD de alta ordem (HOSVD) ou Tucker. Em seguida, exibimos a decomposição *Tensor Train* (TT), uma ferramenta extremamente importante para estudarmos problemas de ordens mais altas. A notação usada aqui será baseada em [29]. Diversos exemplos permeiam o texto para ilustrar a apresentação das ideias.

Em essência, as decomposições tensoriais têm como base a representação do tensor a partir de tensores menores por meio do produto tensorial [29]. Este pode ser pensado como uma operação cujo resultado é um elemento de um espaço de maior dimensão. Como exemplo, recordamos que o produto externo entre dois vetores  $\mathbf{u} \in \mathbb{R}^m$  e  $\mathbf{v} \in \mathbb{R}^n$ , é

dado pela matriz  $\mathbf{u}\mathbf{v}^T \in \mathbb{R}^{m \times n}$ . Então, podemos considerar

$$\text{span}\{\mathbf{u}\mathbf{v}^T : \mathbf{u} \in \mathbb{R}^m, \mathbf{v} \in \mathbb{R}^n\} \subset \mathbb{R}^{m \times n},$$

o espaço gerado por todas as matrizes  $m \times n$  obtidas dessa maneira e denominá-lo espaço tensorial que, neste caso, é de ordem 2, pois seus elementos são tensores de ordem 2. O produto externo é um caso particular de produto tensorial [22, Remark 1.3]. Neste sentido, é possível considerar o produto tensorial de  $N$  vetores e obter um espaço tensorial de ordem  $N$ , cuja estrutura depende dos espaços vetoriais que o geraram. Assim, como no ramo da álgebra linear, as matrizes representam mapas lineares entre espaços vetoriais de dimensão finita, os tensores podem ser entendidos como mapas multilineares [22, Chapter 3.1.4]. A seguir, exibimos definições mais precisas envolvendo esses conceitos.

## 1.1 Álgebra Multilinear Básica

**Definição 1.1 (Tensor de ordem  $N$ )** [31, Definition 2.1.1] *Sejam  $V_1, V_2, \dots, V_N$  espaços vetoriais de dimensões finitas dadas, respectivamente, por  $I_1, I_2, \dots, I_N$ . Considere  $N$  vetores  $\mathbf{v}_i \in V_i$  ( $i = 1, \dots, N$ ). Definimos o produto tensorial  $\mathbf{v}_1 \circ \mathbf{v}_2 \circ \dots \circ \mathbf{v}_N$  pelo seguinte mapa multilinear*

$$(\mathbf{v}_1 \circ \mathbf{v}_2 \circ \dots \circ \mathbf{v}_N)(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N) = \langle \mathbf{v}_1, \mathbf{x}_1 \rangle_{V_1} \langle \mathbf{v}_2, \mathbf{x}_2 \rangle_{V_2} \cdots \langle \mathbf{v}_N, \mathbf{x}_N \rangle_{V_N}$$

onde  $\mathbf{x}_i \in V_i$  e  $\langle \cdot, \cdot \rangle_{V_i}$  denota o produto interno em  $V_i$  para  $i = 1, \dots, N$ . O espaço gerado por todos os elementos  $\mathbf{v}_1 \circ \mathbf{v}_2 \circ \dots \circ \mathbf{v}_N$  é chamado espaço tensorial e um elemento  $\mathcal{A}$  deste espaço é chamado tensor de ordem  $N$  sobre  $V_1 \times V_2 \times \dots \times V_N$ .

A partir da Definição 1.1, tomamos  $V_i = \mathbb{R}^{I_i}$  e o espaço obtido é o de tensores  $(I_1 \times I_2 \times \dots \times I_N)$  com entradas reais que será denotado por  $\mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ . No caso de entradas complexas a notação deste espaço será  $\mathbb{C}^{I_1 \times I_2 \times \dots \times I_N}$ .

Neste trabalho, escalares serão denotados por letras minúsculas ( $a$ ); vetores por letras minúsculas em negrito ( $\mathbf{a}$ ); matrizes por letras maiúsculas em negrito ( $\mathbf{A}$ ) e tensores de alta ordem por letras caligráficas ( $\mathcal{A}$ ). As entradas de um tensor serão denotadas a partir dos índices que o definem, i.e,

$$a_{i_1, i_2, \dots, i_N} = [\mathcal{A}]_{i_1, i_2, \dots, i_N}$$

Fixando um subconjunto de índices do tensor, *subarrays* são formados. No caso de ordem-3, fatias bidimensionais são definidas fixando um dos índices (cf. Figura 1 no Exemplo 1.1). Fixando dois dos índices, definimos as fibras ou vetores de modo- $n$  (cf. Figura 2 no Exemplo 1.1).

## Exemplo 1.1

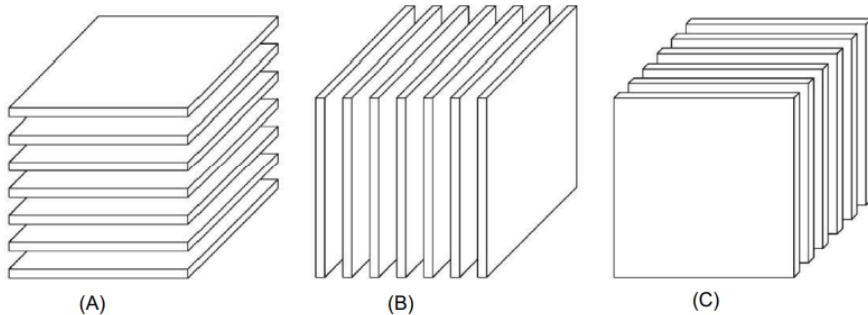


Figura 1 – Fatiamento de um tensor de ordem-3: (A) Horizontal ( $\mathbf{A}_{i,:,:}$ ), (B) Lateral ( $\mathbf{A}_{:,j,:}$ ) e (C) Frontal ( $\mathbf{A}_{:,:,k}$ ). Retirado de [29, Fig. 2.2].

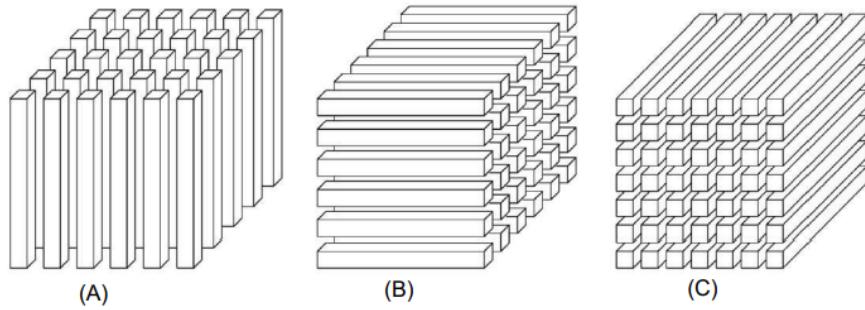


Figura 2 – Fibras de um tensor de ordem-3: (A) Modo-1 (linhas,  $\mathbf{A}_{:,j,k}$ ), (B) Modo-2 (columnas,  $\mathbf{A}_{i,:,k}$ ) e (C) Modo-3 (tubos,  $\mathbf{A}_{i,j,:}$ ). Retirado de [29, Fig. 2.1].

Assim como os vetores e matrizes, os tensores com entradas reais ou complexas formam um espaço vetorial de modo que a definição de soma e multiplicação por escalar é feita por componentes. Em seguida definimos outras operações importantes para nossos estudos.

**Definição 1.2 (Produto Externo)** Sejam  $\mathcal{A} \in \mathbb{C}^{I_1 \times \dots \times I_N}$  e  $\mathcal{B} \in \mathbb{C}^{J_1 \times \dots \times J_M}$ . O produto externo entre  $\mathcal{A}$  e  $\mathcal{B}$  é definido por

$$[\mathcal{A} \circ \mathcal{B}]_{i_1, i_2, \dots, i_N, j_1, j_2, \dots, j_M} = a_{i_1, i_2, \dots, i_N} b_{j_1, j_2, \dots, j_M}$$

para todos os valores de índices.

Observe que a partir da operação de produto externo, o resultado é um tensor cuja ordem é a soma da ordem dos tensores de entrada. Note também que esta definição é uma forma específica de produto tensorial. O Exemplo 1.2 ilustra a operação de produto externo entre três vetores

### Exemplo 1.2

Suponha que um tensor  $\mathcal{A}$  de ordem 3 possa ser escrito como produto externo de três vetores  $\mathbf{u}^{(1)}, \mathbf{u}^{(2)}$  e  $\mathbf{u}^{(3)}$ . Segue que

$$a_{i,j,k} = u_i^{(1)} u_j^{(2)} u_k^{(3)} \quad (1.1)$$

para todos os valores de índices. Tensores que podem ser escritos dessa forma são chamados de tensores de posto 1. Com efeito, seja  $\mathcal{A} \in \mathbb{R}^{2 \times 2 \times 2}$ , cujas fatias frontais são dadas por

$$\mathbf{A}_{::,1} = \begin{pmatrix} 1 & 3 \\ 2 & 6 \end{pmatrix} \quad \text{e} \quad \mathbf{A}_{::,2} = \begin{pmatrix} -2 & -6 \\ -4 & -12 \end{pmatrix}.$$

Observe que tomindo

$$\mathbf{u}^{(1)} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad \mathbf{u}^{(2)} = \begin{pmatrix} 1 \\ 3 \end{pmatrix}, \quad \mathbf{u}^{(3)} = \begin{pmatrix} 1 \\ -2 \end{pmatrix},$$

por meio da operação de produto externo (1.1), recuperamos o tensor original.

**Definição 1.3 (Produto Interno)** Sejam  $\mathcal{A}, \mathcal{B} \in \mathbb{C}^{I_1 \times \dots \times I_N}$ . O produto interno entre  $\mathcal{A}$  e  $\mathcal{B}$  é definido por

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} a_{i_1, i_2, \dots, i_N} \bar{b}_{i_1, i_2, \dots, i_N}$$

onde a barra indica o conjugado. A norma de Frobenius é dada por

$$\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}.$$

**Definição 1.4 (Desdobramento Matricial [29])** Dado  $\mathcal{A} \in \mathcal{C}^{I_1 \times \dots \times I_N}$ , seu desdobramento ou matricização de modo- $n$  é o mapeamento da entrada  $(i_1, i_2, \dots, i_N)$  do tensor para a entrada  $(i_n, j)$  de uma matriz denominada por  $\mathbf{A}_{(n)}$ , de modo que

$$j = 1 + \sum_{\substack{k=1 \\ k \neq n}}^N (i_k - 1) J_k \quad \text{onde} \quad J_k = \prod_{\substack{m=1 \\ m \neq n}}^{k-1} I_m.$$

Dessa forma, escrevemos

$$[\mathbf{A}_{(n)}]_{j_1, j_2} = a_{j_1, j_2}$$

onde  $j_1 = i_n$  e  $j_2 = \overline{i_1 i_2 \cdots i_{n-1} i_{n+1} \cdots i_N}$ , em que utilizamos a notação de [8, expressão (2.1)] dada por

$$\overline{i_1 i_2 \cdots i_N} = i_1 + (i_2 - 1) I_1 + (i_3 - 1) I_1 I_2 + \dots + (i_N - 1) I_1 I_2 \cdots I_{N-1}. \quad (1.2)$$

De maneira mais simples, segundo [29], as colunas da matriz gerada pelo desdobramento são os vetores de modo- $n$  do tensor. Em muitas aplicações com múltiplos graus de liberdade

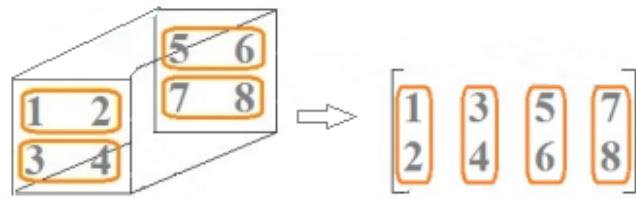


Figura 3 – Ilustração de desdobramento de modo-2 de um tensor  $2 \times 2 \times 2$ . Retirado de [44, Figure 1.7]

é possível empregar as propriedades da decomposição SVD a partir do desdobramento do tensor do modelo. A Figura 3 ilustra o desdobramento de modo-2 de um tensor  $2 \times 2 \times 2$ , e o Exemplo 1.3 apresenta os três desdobramentos para um tensor  $3 \times 4 \times 2$ .

### Exemplo 1.3

Seja  $\mathcal{A} \in \mathbb{R}^{3 \times 4 \times 2}$ , cujas fatias frontais são

$$\mathbf{A}_{(:,:,1)} = \begin{pmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{pmatrix} \quad \text{e} \quad \mathbf{A}_{(:,:,2)} = \begin{pmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{pmatrix}.$$

Segue então que

$$\mathbf{A}_{(1)} = \begin{pmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{pmatrix},$$

$$\mathbf{A}_{(2)} = \begin{pmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{pmatrix},$$

$$\mathbf{A}_{(3)} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 \end{pmatrix}.$$

Existem outras maneiras de realizar o desdobramento. Por exemplo, em [12, Fig. 1], o índice que caminha pelas fibras de modo-3 é mais rápido que o índice que percorre as fibras de modo-2. Neste caso, o desdobramento de modo-1 para o tensor do Exemplo 1.3 é

$$\mathbf{A}_{(1)} = \begin{pmatrix} 1 & 13 & 4 & 16 & 7 & 19 & 10 & 22 \\ 2 & 14 & 5 & 17 & 8 & 20 & 11 & 23 \\ 3 & 15 & 6 & 18 & 9 & 21 & 12 & 24 \end{pmatrix}.$$

O desdobramento de modo- $k$  de um tensor  $\mathcal{A}$  também pode ser definido pela matriz

$$\mathbf{A}_{\langle k \rangle} \in \mathbb{R}^{(I_1 \cdots I_k) \times (I_{k+1} \cdots I_N)}$$

cujas entradas são dadas por

$$[\mathbf{A}_{\langle k \rangle}]_{\overline{i_1 i_2 \cdots i_k}, \overline{i_{k+1} \cdots i_N}} = a_{i_1, i_2, \dots, i_N},$$

com base na notação (1.2). Desta definição, segue imediatamente que para  $k = N$ , o tensor será transformado em um vetor. Além disso,

$$\mathbf{A}_{\langle 1 \rangle} = \mathbf{A}_{(1)} \quad \text{e} \quad \mathbf{A}_{\langle N-1 \rangle} = \mathbf{A}_{(N)}^T,$$

conforme ilustrado no Exemplo 1.4.

#### Exemplo 1.4

Considere o tensor  $\mathcal{A} \in \mathbb{R}^{2 \times 3 \times 2}$ , cujas fatias frontais são

$$\mathbf{A}_{\cdot, :, 1} = \begin{pmatrix} a & c & e \\ b & d & f \end{pmatrix} \quad \text{e} \quad \mathbf{A}_{\cdot, :, 2} = \begin{pmatrix} g & i & k \\ h & j & l \end{pmatrix}.$$

Temos então

$$\mathbf{A}_{\langle 1 \rangle} = \begin{pmatrix} a & c & e & g & i & k \\ b & d & f & h & j & l \end{pmatrix} = \mathbf{A}_{(1)},$$

$$\mathbf{A}_{\langle 2 \rangle} = \begin{pmatrix} a & g \\ b & h \\ c & i \\ d & j \\ e & k \\ f & l \end{pmatrix},$$

$$\mathbf{A}_{\langle 3 \rangle} = (a \ b \ c \ d \ e \ f \ g \ h \ i \ j \ k \ l)^T.$$

Em geral, as possíveis permutações dos índices não influenciam no resultado final desde que os cálculos se mantenham consistentes. Veremos neste trabalho que operações com tensores podem ser reescritas a partir de seu desdobramento matricial e serem compreendidas de forma mais simples e direta.

Também definimos o respectivo operador de reconstrução  $fold_i$  tal que

$$fold_i \left\{ \mathbf{A}_{(i)} \right\} = \mathcal{A} \quad \text{ou} \quad fold_i \left\{ \mathbf{A}_{\langle i \rangle} \right\} = \mathcal{A}.$$

## Produto Tensor-Matriz

**Definição 1.5** O produto de modo- $n$  de um tensor  $\mathcal{A} \in \mathbb{C}^{I_1 \times \dots \times I_N}$  e uma matriz  $\mathbf{U} \in \mathbb{C}^{J \times I_n}$ , denotado por  $\mathcal{A} \times_n \mathbf{U}$ , de dimensões  $(I_1 \times I_2 \times \dots \times J \dots \times I_N)$  é tal que

$$(\mathcal{A} \times_n \mathbf{U})_{i_1, i_2, \dots, j, \dots, i_N} = \sum_{i_n=1}^{I_n} a_{i_1, i_2, \dots, i_n, \dots, i_N} u_{j, i_n}.$$

É possível expressar o produto tensor-matriz em termos do tensor desmembrado:

$$\mathcal{B} = \mathcal{A} \times_n \mathbf{U} \Leftrightarrow \mathbf{B}_{(n)} = \mathbf{U} \mathbf{A}_{(n)}.$$

A ordem de múltiplos produtos entre tensor e matrizes em diferentes modos é irrelevante: sejam  $\mathbf{U} \in \mathbb{C}^{J \times I_n}$  e  $\mathbf{V} \in \mathbb{C}^{K \times I_m}$ , então

$$\begin{aligned} \mathcal{A} \times_n \mathbf{U} \times_m \mathbf{V} &= \sum_{i_m=1}^{I_m} \left( \sum_{i_n=1}^{I_n} a_{i_1, \dots, i_n, \dots, i_m, \dots, i_N} u_{j, i_n} \right) v_{k, i_m} \\ &= \sum_{i_m=1}^{I_m} \sum_{i_n=1}^{I_n} a_{i_1, \dots, i_n, \dots, i_m, \dots, i_N} u_{j, i_n} v_{k, i_m} \\ &= \sum_{i_n=1}^{I_n} \left( \sum_{i_m=1}^{I_m} a_{i_1, \dots, i_n, \dots, i_m, \dots, i_N} v_{k, i_m} \right) u_{j, i_n} \\ &= \mathcal{A} \times_m \mathbf{V} \times_n \mathbf{U}. \end{aligned}$$

No caso em que o produto é feito no mesmo modo ( $I_n = I_m$ ), segue que

$$\mathcal{A} \times_n \mathbf{U} \times_n \mathbf{V} = \mathcal{A} \times_n (\mathbf{V} \mathbf{U}).$$

Uma releitura do produto matricial por meio da notação para o produto tensor-matriz é feita no Exemplo 1.5

### Exemplo 1.5

A partir da notação do produto tensor-matriz, é possível reinterpretar o produto entre matrizes. Para as matrizes  $\mathbf{U} \in \mathbb{R}^{J_1 \times I_1}$ ,  $\mathbf{V} \in \mathbb{R}^{J_2 \times I_2}$  e  $\mathbf{S} \in \mathbb{R}^{I_1 \times I_2}$ , segue que

$$\mathbf{U} \mathbf{S} \mathbf{V}^T = \mathbf{S} \times_1 \mathbf{U} \times_2 \mathbf{V}.$$

Neste sentido, a notação permite generalizar a transposição matricial para ordens superiores. As colunas de  $\mathbf{U}$  estão associadas ao espaço gerado pelos vetores de modo-1 de  $\mathbf{S}$ , e as colunas de  $\mathbf{V}$  estão associadas ao espaço de modo-2 de  $\mathbf{S}$ .

## Posto de um Tensor

No que diz respeito ao posto ou *rank* de um tensor, existem diversas maneiras de defini-lo e, consequentemente, vários algoritmos de decomposição que se baseiam nestas definições.

**Definição 1.6 (Tensor de posto 1)** Um tensor de ordem  $N$  possui posto 1 se puder ser escrito como produto externo de  $N$  vetores

**Definição 1.7 (Posto)** O posto de um tensor  $\mathcal{A}$  de ordem  $N$  denotado por  $R = \text{posto}(\mathcal{A})$  é o número mínimo de tensores de posto 1 cuja soma compõe  $\mathcal{A}$ .

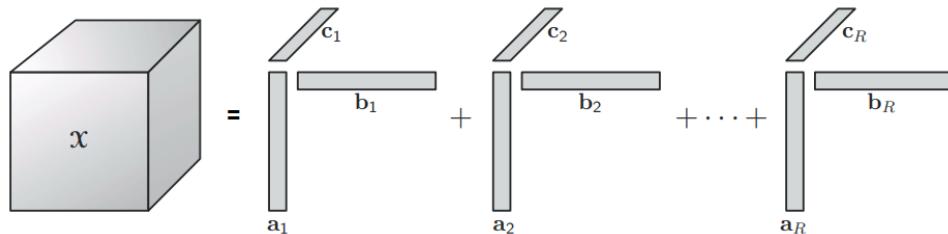


Figura 4 – Decomposição em tensores de posto-1 de ordem 3. Retirado de [29, Fig. 3.1].

**Definição 1.8 (Posto Multilinear [31])** Dado um tensor  $\mathcal{A}$  de ordem  $N$ , o posto multilinear é um vetor  $N$  dimensional cuja  $n$ -ésima entrada,  $R_n = \text{posto}(\mathcal{A}_{(n)})$  é a dimensão do subespaço gerado pelos vetores do modo  $n$ .

Observe que esta última definição generaliza a de posto coluna ou posto linha de uma matriz. No entanto, veremos que diferente do caso matricial, as entradas do posto multidimensional não necessariamente são iguais. Ainda, da definição segue que  $R_k \leq R$ , para  $k = 1, \dots, N$ .

O Exemplo 1.6 ilustra essas ideias, e foi extraído de [31, Example 2.6.5].

### Exemplo 1.6

Seja  $\mathcal{A} \in \mathbb{R}^{2 \times 2 \times 2}$ , cujas fatias frontais são dadas por

$$\mathbf{A}_{:,:,1} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \text{e} \quad \mathbf{A}_{:,:,2} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}.$$

Note que o posto multilinear é  $(2, 2, 2)$ . Agora definindo

$$\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{e} \quad \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

segue que  $\text{posto}(\mathcal{A}) \leq 3$ , pois

$$\mathcal{A} = \mathbf{e}_2 \circ \mathbf{e}_1 \circ \mathbf{e}_1 + \mathbf{e}_1 \circ \mathbf{e}_2 \circ \mathbf{e}_1 + \mathbf{e}_1 \circ \mathbf{e}_1 \circ \mathbf{e}_2.$$

No caso matricial, o posto pode ser obtido diretamente por meio da decomposição SVD. No entanto, para ordens superiores, não existe algoritmo que forneça este tipo de informação para um determinado tensor, pois este problema é do tipo NP-difícil [26]. A decomposição em fatores paralelos (PARAFAC) que veremos adiante será utilizada para computarmos uma aproximação para o posto de um tensor específico.

Fixadas as dimensões do tensor e variando suas entradas, qualquer valor de posto que ocorra com probabilidade não nula, é chamado de posto típico. Por exemplo, dado um tensor  $2 \times 2 \times 2$  de entradas reais, seu posto típico é  $\{2, 3\}$ , isto é, o conjunto de tensores com tais dimensões e que possuem posto igual a 2 ou 3 tem medida de Lebesgue não nula (ocorrem quase-sempre) (veja [30]). O valor de posto igual a 1 pode ocorrer, mas com probabilidade zero. Discutimos sobre estes valores no Exemplo 1.8. Valores de posto típico para tensores de ordem 3 de diversas dimensões são apresentados em [29, Table 3.3].

Vale a pena mencionar o critério baseado em hiperdeterminantes para determinação do posto de tensores  $2 \times 2 \times 2$ . Seja  $\mathcal{X}$  um tensor destas dimensões. Seguindo [30], definimos seu hiperdeterminante como

$$\begin{aligned} \Delta(\mathcal{X}) := & x_{1,1,1}^2 x_{2,2,2}^2 + x_{1,1,2}^2 x_{2,2,1}^2 + x_{1,2,1}^2 x_{2,1,2}^2 + x_{1,2,2}^2 x_{2,1,1}^2 \\ & - 2(x_{1,1,1} x_{1,1,2} x_{2,2,1} x_{2,2,2} + x_{1,1,1} x_{1,2,1} x_{2,1,2} x_{2,2,2} + x_{1,1,1} x_{1,2,2} x_{2,1,1} x_{2,2,2} \\ & + x_{1,1,2} x_{1,2,1} x_{2,1,2} x_{2,2,1} + x_{1,1,2} x_{1,2,2} x_{2,2,1} x_{2,1,1} + x_{1,2,1} x_{1,2,2} x_{2,1,2} x_{2,1,1}) \\ & + 4(x_{1,1,1} x_{1,2,2} x_{2,1,2} x_{2,2,1} + x_{1,1,2} x_{1,2,1} x_{2,1,1} x_{2,2,2}) \end{aligned} \quad (1.3)$$

O critério definido em [30] nos diz que  $\Delta(\mathcal{X}) > 0$  significa que o posto de  $\mathcal{X}$  é 2, enquanto para  $\Delta(\mathcal{X}) < 0$  o tensor é de posto 3. Caso o hiperdeterminante seja zero, o posto não é determinado precisamente, podendo ser 0, 1 ou 3. Em [47], é mostrado que se uma das fatias frontais for não singular e  $\Delta(\mathcal{X}) = 0$ , então o tensor é de posto 3, mas pode ser aproximado arbitrariamente por um tensor cujo posto é 2.

A partir do critério baseado em hiperdeterminantes, é possível mostrar que tomando um tensor  $\mathcal{A}$   $2 \times 2 \times 2$  com entradas obtidas de uma distribuição normal com média 0 e variância 1, a probabilidade de  $\text{posto}(\mathcal{A}) = 2$  é 79% e de  $\text{posto}(\mathcal{A}) = 3$  é 21%, aproximadamente. O Exemplo 1.7 ilustra, computacionalmente, esse resultado.

### Exemplo 1.7

Considere o seguinte código escrito em Python:

```
import numpy as np
from ferramentas import hyperdet
n = 1000000
tol = 10**(-8)
```

```

posto2 = 0
posto3 = 0
outro = 0
for i in range(n):
    X = np.random.randn(2,2,2)
    delta = hyperdet(X)
    if delta>tol:
        posto2 += 1
    elif delta<-tol:
        posto3 +=1
    else:
        outro += 1
print("posto2 = {:.2f}, posto3 = {:.2f}, outro =
 {:.2f}".format(posto2*100/n, posto3*100/n, outro*100/n))

posto2 = 78.53, posto3 = 21.47, outro = 0.00

```

A saída exibida pelo código corrobora os experimentos feitos por Kruskal em [30].

## 1.2 Decomposições Clássicas

A seguir estudaremos as decomposições clássicas para tensores de ordens superiores. Neste trabalho, para que possamos manusear com mais facilidade as estruturas e entender as sutilezas dos métodos, iremos focar apenas em tensores de ordem 3 com entradas reais.

### Decomposição em Fatores Paralelos (PARAFAC)

**Definição 1.9 (PARAFAC)** A decomposição em fatores paralelos de um tensor  $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$  é a combinação linear em termos de  $R$  componentes de posto-1:

$$\mathcal{X} = \sum_{r=1}^R \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \quad (1.4)$$

onde  $\mathbf{a}_r \in \mathbb{R}^I$ ,  $\mathbf{b}_r \in \mathbb{R}^J$  e  $\mathbf{c}_r \in \mathbb{R}^K$  para  $r = 1, \dots, R$ .

Dependendo do contexto, pode ser interessante normalizar os fatores agregando os escalares em um vetor  $\boldsymbol{\lambda} \in \mathbb{R}^R$  de modo que

$$\mathcal{X} = \sum_{r=1}^R \lambda_r \hat{\mathbf{a}}_r \circ \hat{\mathbf{b}}_r \circ \hat{\mathbf{c}}_r. \quad (1.5)$$

Alojando os vetores  $\mathbf{a}_r$ ,  $\mathbf{b}_r$  e  $\mathbf{c}_r$  nas colunas das matrizes  $\mathbf{A} \in \mathbb{R}^{I \times R}$ ,  $\mathbf{B} \in \mathbb{R}^{J \times R}$  e  $\mathbf{C} \in \mathbb{R}^{K \times R}$ , respectivamente, a equação (1.4) pode ser reescrita em termos das fatias frontais de  $\mathcal{X}$ , isto é,

$$\mathbf{X}_{(:,:,k)} = c_{k,1}\mathbf{a}_1\mathbf{b}_1^T + c_{k,2}\mathbf{a}_2\mathbf{b}_2^T + \dots + c_{k,R}\mathbf{a}_R\mathbf{b}_R^T \quad (1.6)$$

ou então

$$\mathbf{X}_{(:,:,k)} = \mathbf{A}\mathbf{D}_k\mathbf{B}^T,$$

com  $\mathbf{D}_k = \text{diag}(c_{k,1}, c_{k,2}, \dots, c_{k,R})$ . Observe que cada fatia frontal é escrita em termos das mesmas componentes  $\mathbf{A}$  e  $\mathbf{B}$ , mas com pesos diferentes. Todas as fatias são expressas por perfis paralelos e proporcionais, como em (1.6).

Outra forma de expressar a decomposição é através do produto de Khatri-Rao (cf. Definição A.2). Neste caso, os desdobramentos do tensor podem ser escritos como

$$\mathbf{X}_{(1)} = \mathbf{A}(\mathbf{C} \odot \mathbf{B})^T, \quad (1.7)$$

$$\mathbf{X}_{(2)} = \mathbf{B}(\mathbf{C} \odot \mathbf{A})^T, \quad (1.8)$$

$$\mathbf{X}_{(3)} = \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T. \quad (1.9)$$

Dado um tensor  $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$  com  $\text{posto}(\mathcal{X}) = R$ , dizemos que sua decomposição PARAFAC é única se os fatores  $\mathbf{A}$ ,  $\mathbf{B}$  e  $\mathbf{C}$  são únicos a menos do reordenamento de suas colunas e escalamento. Neste sentido, assumindo  $I = J = K$  e dada  $\mathbf{P} \in \mathbb{R}^{I \times I}$  uma matriz de permutação, segue que os fatores  $\mathbf{PA}$ ,  $\mathbf{PB}$  e  $\mathbf{PC}$  geram o mesmo tensor. Ainda

$$\mathcal{X} = \sum_{r=1}^R (\alpha_r \mathbf{a}_r) \circ (\beta_r \mathbf{b}_r) \circ (\gamma_r \mathbf{c}_r),$$

onde  $\alpha_r \beta_r \gamma_r = 1$ , para  $r = 1, \dots, R$ . Para enunciar o resultado de unicidade é necessário definir o chamado posto de Kruskal ou  $k$ -posto [30].

**Definição 1.10 ( $k$ -posto)** O  $k$ -posto de uma matriz  $\mathbf{A}$ , denotado por  $k_{\mathbf{A}}$ , é definido como o maior número  $k$  tal que o arranjo de quaisquer  $k$  colunas de  $\mathbf{A}$  seja linearmente independente.

**Teorema 1.1** Seja  $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$  com  $\text{posto}(\mathcal{X}) = R$  e  $\mathbf{A}$ ,  $\mathbf{B}$ ,  $\mathbf{C}$  seus fatores da decomposição PARAFAC. A condição suficiente para unicidade é que

$$k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \geq 2R + 2.$$

Prova: Veja [30, Theorem 4a].

## Computando a Decomposição PARAFAC

Como mencionado anteriormente, não existem algoritmos finitos para cálculo do posto de um tensor qualquer. Consequentemente, um problema que enfrentamos é: como escolher o número de componentes de posto 1 cuja soma aproxima nosso tensor de interesse? O procedimento que vamos adotar é tomar  $R = 1, 2, 3, \dots$ , e escolher aquele que nos dá a igualdade em (1.4). No entanto, os métodos que dispomos para o cálculo da decomposição são iterativos. Neste caso, dado  $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ , para um  $R$  fixado, obtemos uma aproximação  $\hat{\mathcal{X}}$  e é verificado o erro relativo

$$\frac{\|\mathcal{X} - \hat{\mathcal{X}}\|_F}{\|\mathcal{X}\|_F}.$$

Caso esteja abaixo de uma certa tolerância, tomamos  $R$  como o posto do tensor  $\mathcal{X}$ . No entanto, esta abordagem apresenta algumas dificuldades, por exemplo, considere o tensor de posto 3:

$$\mathcal{X} = \mathbf{a}_2 \circ \mathbf{b}_1 \circ \mathbf{c}_1 + \mathbf{a}_1 \circ \mathbf{b}_2 \circ \mathbf{c}_1 + \mathbf{a}_1 \circ \mathbf{b}_1 \circ \mathbf{c}_2$$

com  $\mathbf{A} \in \mathbb{R}^{I \times 2}$ ,  $\mathbf{B} \in \mathbb{R}^{J \times 2}$  e  $\mathbf{C} \in \mathbb{R}^{K \times 2}$ . Este pode ser aproximado arbitrariamente pelo tensor de posto 2

$$\mathcal{Y} = \beta \left( \mathbf{a}_1 + \frac{1}{\beta} \mathbf{a}_2 \right) \circ \left( \mathbf{b}_1 + \frac{1}{\beta} \mathbf{b}_2 \right) \circ \left( \mathbf{c}_1 + \frac{1}{\beta} \mathbf{c}_2 \right) - \beta \mathbf{a}_1 \mathbf{b}_1 \mathbf{c}_1,$$

de maneira que

$$\|\mathcal{X} - \mathcal{Y}\|_F = \frac{1}{\beta} \|\mathbf{a}_2 \circ \mathbf{b}_2 \circ \mathbf{c}_1 + \mathbf{a}_2 \circ \mathbf{b}_1 \circ \mathbf{c}_2 + \mathbf{a}_1 \circ \mathbf{b}_2 \circ \mathbf{c}_2\|_F + \frac{1}{\beta} \mathbf{a}_2 \mathbf{b}_2 \mathbf{c}_2.$$

Este é um exemplo de um tensor de posto 2 que está arbitrariamente próximo de um tensor de posto 3. Esta dificuldade é chamada de degenerescência e é tratada em [29, § 3.3]. Note que, a partir disso, o conjunto dos tensores de posto 2 não é fechado. Em outras palavras, a decomposição em  $R$  fatores paralelos não necessariamente é a melhor aproximação de posto  $R$  para um tensor dado.

O chamado posto típico de uma classe de tensores nos traz informações sobre quão bem podemos aproximar um dado tensor. Por exemplo, em [29] é dito que um tensor  $2 \times 2 \times 2$  com entradas reais possui posto típico 2 ou 3. Assim, em geral, 2 ou 3 fatores serão suficientes para obter uma boa aproximação do tensor.

Para o cálculo dos fatores da decomposição, o objetivo é resolver o problema de quadrados mínimos

$$\min_{\mathbf{A}, \mathbf{B}, \mathbf{C}} \left\| \mathcal{X} - \sum_{r=1}^R \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \right\|_F^2. \quad (1.10)$$

Assumindo que o número de componentes está fixo, o método mais tradicional implementado em diversas bibliotecas de linguagens de programação é o chamado Quadrados

Mínimos Alternados (ALS, do inglês *Alternate Least Squares*) [29]. A ideia é, a partir de matrizes iniciais  $\mathbf{B}$  e  $\mathbf{C}$ , atualizarmos  $\mathbf{A}$ ; fixamos  $\mathbf{A}$  e  $\mathbf{C}$  e atualizarmos  $\mathbf{B}$ ; fixamos  $\mathbf{A}$  e  $\mathbf{B}$  e atualizarmos  $\mathbf{C}$ . Para estimar  $\mathbf{A}$  é necessário resolver o problema

$$\min_{\hat{\mathbf{A}}} \left\| \mathbf{X}_{(1)} - \hat{\mathbf{A}}(\mathbf{C} \odot \mathbf{B})^T \right\|_F^2,$$

cuja solução é escrita a partir da pseudo-inversa de Moore-Penrose [19],

$$\mathbf{A} = \mathbf{X}_{(1)} \left[ (\mathbf{C} \odot \mathbf{B})^T \right]^\dagger,$$

ou então, das propriedades do produto de Khatri-Rao (cf. Definição A.2),

$$\mathbf{A} = \mathbf{X}_{(1)} (\mathbf{C} \odot \mathbf{B}) (\mathbf{C}^T \mathbf{C} * \mathbf{B}^T \mathbf{B})^\dagger,$$

onde o operador  $*$  é o produto de Hadamard (cf. Definição A.3).

Note que a pseudo-inversa é calculada sobre uma matriz  $R \times R$ . De forma análoga, são computadas as estimativas de  $\mathbf{B}$  e  $\mathbf{C}$  e seguimos iterativamente até que um critério de parada seja satisfeito, o qual pode ser: pequena variação na função objetivo, pequena variação nos fatores, atingir o valor mínimo da função objetivo ou atingir o número máximo de iterações. O pseudocódigo é apresentado no Algoritmo 1.

---

**Algoritmo 1 – PARAFAC via ALS**


---

**Entrada:**  $\mathcal{X}, R$

initialize  $\mathbf{B} \in \mathbb{R}^{J \times R}$  e  $\mathbf{C} \in \mathbb{R}^{K \times R}$

**enquanto** critério não é satisfeito **faça**

$$\mathbf{A} = \mathbf{X}_{(1)} (\mathbf{C} \odot \mathbf{B}) (\mathbf{C}^T \mathbf{C} * \mathbf{B}^T \mathbf{B})^\dagger$$

$$\mathbf{B} = \mathbf{X}_{(2)} (\mathbf{C} \odot \mathbf{A}) (\mathbf{C}^T \mathbf{C} * \mathbf{A}^T \mathbf{A})^\dagger$$

$$\mathbf{C} = \mathbf{X}_{(3)} (\mathbf{B} \odot \mathbf{A}) (\mathbf{B}^T \mathbf{B} * \mathbf{A}^T \mathbf{A})^\dagger$$

Normalize as colunas de  $\mathbf{A}, \mathbf{B}, \mathbf{C}$  e guarde suas normas no vetor  $\lambda$ .

**fim**

**retorna**  $\lambda, \mathbf{A}, \mathbf{B}, \mathbf{C}$

---

É importante notar que cada matriz estimada é a solução de um problema de quadrados mínimos clássico e, a cada iteração, a função objetivo correspondente melhora ou não. Neste sentido, ao longo das iterações, o erro é não crescente e como cada função objetivo é limitada inferiormente, podemos dizer que teremos convergência do método. No entanto, este fato não implica na convergência na sequência de matrizes gerada, mas em situações práticas geralmente é o caso [45]. Portanto, se há pouca variação na função objetivo a cada iteração, podemos parar o método na expectativa de termos encontrado uma boa solução. No entanto, o problema pode possuir diversos mínimos locais e as soluções encontradas podem ser diferentes para inicializações diferentes.

Por sua simplicidade de implementação, diversas técnicas de aperfeiçoamento da solução encontrada pelo método e de melhora para sua velocidade de convergência são

mencionadas em [29, Section 3.4]. Por exemplo, existe uma forte influência da aproximação inicial sobre a qualidade da solução. Os fatores podem ser inicializados aleatoriamente ou tomando o  $n$ -ésimo fator como os  $R$  primeiros vetores singulares de  $\mathbf{X}_{(n)}$ .

No Exemplo 1.8 ilustramos o funcionamento do método. Vale ressaltar que os testes foram realizados em linguagem Python com a biblioteca `tensorly.py`, que contém todos os meios necessários de manuseio com tensores. As aproximações iniciais foram todas aleatórias com entradas uniformemente distribuídas entre 0 e 1. Caso entre uma iteração e outra o decréscimo seja menor que  $10^{-6}$ , paramos o método ou executamos o laço até que seja atingido o número máximo de 100 iterações.

### Exemplo 1.8

(a) Seja  $\mathcal{X} \in \mathbb{R}^{4 \times 3 \times 2}$  cujas fatias frontais são

$$\mathbf{X}_{(:,:,1)} = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \\ 9 & 10 & 11 \end{pmatrix} \quad \text{e} \quad \mathbf{X}_{(:,:,2)} = \begin{pmatrix} 12 & 13 & 14 \\ 15 & 16 & 17 \\ 18 & 19 & 18 \\ 21 & 22 & 23 \end{pmatrix}.$$

De acordo com [29], o tensor dado possui posto típico igual a 4, ou seja, é esperado que 4 componentes sejam suficientes para obter uma boa aproximação do tensor. Tomando  $R = 2$ , obtemos um erro relativo de 0.0064 atingindo o número máximo de iterações. Já com  $R = 3$ , o erro foi  $3.9257 \times 10^{-8}$  em 32 iterações. Considerando  $R = 4$ , o método convergiu para uma solução com erro igual a zero em 1 iteração, como esperávamos.

Uma característica observada nos testes, quando tomamos  $R$  maior que os valores de posto típico, foi que o método converge para uma solução cujo erro é pequeno, mas pior que para  $R = 4$ . Caso o número de componentes seja uma grandeza de interesse, é importante verificar os valores de posto típico previamente, verificar o erro relativo e o número de iterações do método.

(b) Seja  $\mathcal{X} \in \mathbb{R}^{2 \times 2 \times 2}$  cujas fatias frontais são

$$\mathbf{X}_{(:,:,1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{e} \quad \mathbf{X}_{(:,:,2)} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

Com base no critério por hiperdeterminante, segue que  $\Delta(X) = -4$  de modo que o posto é igual a 3, com uma decomposição possível é

$$\mathcal{X} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \circ \begin{pmatrix} 0 \\ 1 \end{pmatrix} \circ \begin{pmatrix} 2 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \end{pmatrix} \circ \begin{pmatrix} 1 \\ 1 \end{pmatrix} \circ \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} \circ \begin{pmatrix} 1 \\ -1 \end{pmatrix} \circ \begin{pmatrix} 1/2 \\ -1/2 \end{pmatrix}.$$

No entanto, o método ALS possui grandes dificuldades em encontrar uma aproximação de posto igual a 3 para este tensor. De fato, inicializamos o método com 10000 diferentes aproximações iniciais e nenhuma solução encontrada obteve erro relativo da ordem de  $10^{-4}$  ou menor. Todavia, considerando aproximações de posto igual a 4, aproximadamente 99% das inicializações resultaram em soluções com erro relativo menor que  $10^{-10}$ . Uma delas exibimos abaixo:

$$\begin{aligned}\boldsymbol{\lambda} &= (106.9476 \quad 102.2455 \quad 4.8815 \quad 1.7364), \\ \mathbf{A} &= \begin{pmatrix} 0.9189 & -0.9283 & -0.8231 & 0.1593 \\ -0.3945 & 0.3719 & 0.5678 & 0.9872 \end{pmatrix}, \\ \mathbf{B} &= \begin{pmatrix} 0.6954 & 0.7188 & 0.05376 & 0.5594 \\ 0.7186 & 0.6952 & 0.9986 & 0.8289 \end{pmatrix}, \\ \mathbf{C} &= \begin{pmatrix} 0.05773 & 0.0432 & 0.3331 & 0.4827 \\ 0.9983 & 0.9991 & 0.9429 & 0.8758 \end{pmatrix}.\end{aligned}$$

Assim, fica claro que o método ALS apresenta problemas de convergência para o mínimo global. O valor de posto típico nos fornece um limitante inferior para a escolha de posto da aproximação. Incrementando o posto, o método tem maior liberdade para minimizar (1.10). No entanto, também aumentamos o custo computacional da execução..

## Generalização da decomposição SVD

Primeiramente vamos recordar a decomposição em valores singulares (SVD) para matrizes. Sejam  $\mathbf{A} \in \mathbb{R}^{m \times n}$  com  $\text{posto}(\mathbf{A}) = r \leq \min\{m, n\}$  e  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$  seus valores singulares. Então existem matrizes ortogonais  $\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_m] \in \mathbb{R}^{m \times m}$ ,  $\mathbf{V} = [\mathbf{v}_1 \cdots \mathbf{v}_n] \in \mathbb{R}^{n \times n}$  e matriz diagonal  $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times n}$ , onde

$$\boldsymbol{\Sigma} = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_r, 0, \dots, 0),$$

de modo que

$$\mathbf{A} = \mathbf{U} \boldsymbol{\Sigma} \mathbf{V}^T = \sum_{k=1}^r \sigma_k \mathbf{u}_k \mathbf{v}_k^T.$$

Caso os valores singulares sejam todos distintos, a unicidade é assegurada a menos de trocas de sinal nas componentes dos vetores singulares. Ainda, segue da decomposição uma propriedade extremamente útil: a matriz

$$\mathbf{A}_s = \sum_{k=1}^s \sigma_k \mathbf{u}_k \mathbf{v}_k^T \quad (s < r)$$

é a melhor aproximação de  $\mathbf{A}$  na norma Euclidiana ou de Frobenius entre todas as matrizes de posto  $s$ . Este resultado é conhecido como *Teorema de Eckart-Young* [48, Theorem 5.9].

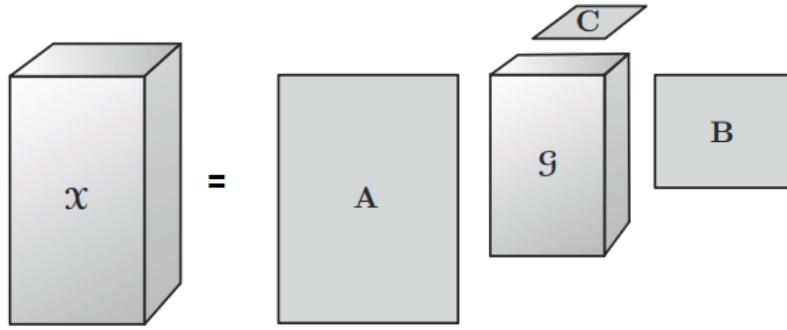


Figura 5 – Decomposição de Tucker de ordem 3. Retirado de [29, Fig 4.1].

O erro cometido por esta aproximação é:

$$\|\mathbf{A} - \mathbf{A}_s\|_F^2 = \sigma_{s+1}^2 + \dots + \sigma_r^2.$$

Desse modo, podemos introduzir um parâmetro  $\delta$  na decomposição para determinar a precisão que desejamos alcançar com a aproximação de baixo posto. Este procedimento pode ser útil, pois especificando uma precisão relativa, o resultado da fatoração é uma matriz  $A_r$  de posto  $r$  que satisfaz  $\|\mathbf{A} - \mathbf{A}_r\|_F^2 \leq \delta \|\mathbf{A}\|_F^2$ . Este procedimento é denominado SVD $_\delta$ . Em diversas aplicações, é desejado aproximar uma matriz de dados por estruturas de posto pequeno, ou então realizar compressão dos dados ou de uma figura. A decomposição SVD é o melhor caminho neste sentido.

Assim, é interessante pensar em generalizações de alta ordem da decomposição SVD para compreensão de estruturas de dados mais complicadas. Uma das abordagens mais clássicas é a decomposição de Tucker estudada em [49] e a SVD de alta ordem (HOSVD) estudada por Lathauwer *et al.* em [12].

### Decomposição de Tucker/HOSVD

Essencialmente, a decomposição de Tucker decompõe um tensor em um tensor núcleo cujos modos são transformados por matrizes. Dado  $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$  com posto multidimensional  $(P, Q, R)$ , segue que

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = \sum_{p=1}^P \sum_{q=1}^Q \sum_{r=1}^R g_{p,q,r} \mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_r,$$

onde  $\mathbf{A} \in \mathbb{R}^{I \times P}$ ,  $\mathbf{B} \in \mathbb{R}^{J \times Q}$  e  $\mathbf{C} \in \mathbb{R}^{K \times R}$  são as matrizes dos fatores dos modos, conforme ilustrado na Figura 5. O tensor  $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$  é chamado de tensor núcleo ou tensor *core* e suas entradas indicam as diversas interações entre os fatores. O caso em que os fatores são matrizes ortogonais é estudado em [12], onde foi mostrado que é possível obter um análogo à decomposição SVD para alta ordem. Neste caso,  $\mathcal{X}$  pode ser expresso como

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}, \quad (1.11)$$

onde  $\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}$  são matrizes com colunas ortonormais entre si e  $\mathcal{G}$  é um tensor núcleo tal que suas fatias de um modo fixado são ortogonais entre si e as normas destes são tais que

$$\|\mathcal{G}_{i_n=1}\|_F \geq \|\mathcal{G}_{i_n=2}\|_F \geq \dots \geq \|\mathcal{G}_{i_n=I_n}\|_F \geq 0$$

para  $n = 1, 2, 3$ . Assim, as diferentes fatias de  $\mathcal{G}$  são escolhidas de modo a serem mutuamente ortogonais com norma decrescente e, caso sejam distintas, a decomposição é única a menos de um sinal. Além disso, a partir de (1.11), o posto multilinear de  $\mathcal{X}$  pode ser obtido, isto é, suas componentes são estabelecidas como

$$R_k = \text{posto}(\mathbf{U}^{(k)}), \quad k = 1, 2, 3.$$

Comparando a versão matricial e a tensorial da SVD, podemos encontrar diversas semelhanças. Em primeiro lugar, os vetores singulares à esquerda e à direita foram substituídos pelos vetores de modo  $n$ . Em segundo, o papel dos valores singulares é feito pelas normas das fatias do tensor núcleo que são todos positivos. Em terceiro, o tensor núcleo não é necessariamente diagonal; no entanto, simplificações podem ser aplicadas para que a maior parte das entradas sejam nulas (veja [45]). A generalização vem do fato que a decomposição SVD clássica é caso particular da HOSVD [12].

## Computando a HOSVD

Analogamente à decomposição PARAFAC, na inicialização do método é necessário explicitar as entradas do posto multidimensional. Dessa maneira, a decomposição gera matrizes que não são necessariamente de posto completo, de modo que (1.11) pode ser truncada como mostra a Figura 6.

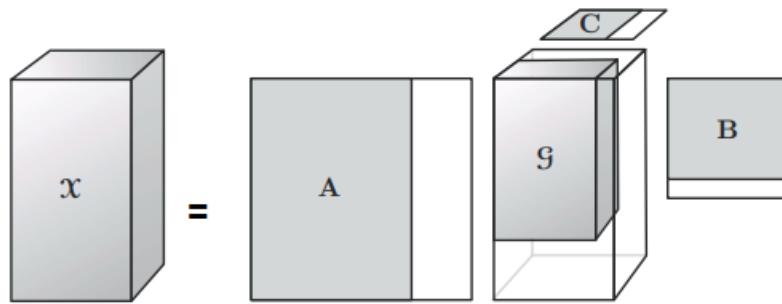


Figura 6 – Decomposição HOSVD truncada de ordem 3. Retirado de [29, Fig 4.2].

O cálculo da decomposição é feito a partir das decomposições SVD dos diferentes desdobramentos matriciais ( $\mathbf{X}_{(1)}, \mathbf{X}_{(2)}, \mathbf{X}_{(3)}$ ). As respectivas matrizes ortogonais que contêm os autovetores à esquerda ( $\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}$ ) são armazenadas e suas respectivas inversas são tomadas. Logo, o tensor núcleo é dado por

$$\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}^{(1)^T} \times_2 \mathbf{U}^{(2)^T} \times_3 \mathbf{U}^{(3)^T}.$$

Em termos de operações matriciais, a expressão pode ser reescrita com relação aos desdobramentos como

$$\mathbf{G}_{(1)} = \mathbf{U}^{(1)^T} \mathbf{X}_{(1)} (\mathbf{U}^{(2)} \otimes \mathbf{U}^{(3)}),$$

onde o operador  $\otimes$  é o produto de Kronecker (cf. Definição A.1). O pseudocódigo para o cálculo da decomposição é exibido no Algoritmo 2. No Exemplo 1.9 desenvolvemos os cálculos da decomposição HOSVD de um tensor  $2 \times 2 \times 2$ .

---

### Algoritmo 2 – HOSVD

---

**Entrada:**  $\mathcal{X}, (R_1, R_2, R_3)$

**para**  $n=1,2,3$  **faça**

$$\left| \begin{array}{l} \mathbf{X}_{(n)} = \mathbf{U} \Sigma \mathbf{V}^T \\ \text{Faça } \mathbf{U}^{(n)} \text{ receber as } R_n \text{ primeiras colunas de } \mathbf{U} \end{array} \right.$$

**fim**

$$\mathbf{G}_{(1)} = \mathbf{U}^{(1)^T} \mathbf{X}_{(1)} (\mathbf{U}^{(2)} \otimes \mathbf{U}^{(3)})$$

**retorna**  $\mathcal{G}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \mathbf{U}^{(3)}$

---

### Exemplo 1.9

Considere o tensor  $\mathcal{X} \in \mathbb{R}^{2 \times 2 \times 2}$  cujas fatias frontais são dadas por

$$\mathbf{X}_{::,1} = \begin{pmatrix} 1 & 0 \\ 1 & -1 \end{pmatrix} \quad \text{e} \quad \mathbf{X}_{::,2} = \begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix}.$$

Aqui utilizamos a biblioteca `TensorToolbox.jl` [41] para nos auxiliar nos cálculos. Desse modo, calculando os desdobramentos matriciais, temos

$$\mathbf{X}_{(1)} = \begin{pmatrix} 1 & 0 & -1 & 1 \\ -1 & 1 & 1 & 0 \end{pmatrix},$$

$$\mathbf{X}_{(2)} = \begin{pmatrix} 1 & 1 & -1 & 1 \\ 0 & -1 & 1 & 0 \end{pmatrix},$$

$$\mathbf{X}_{(3)} = \begin{pmatrix} 1 & 1 & 0 & -1 \\ -1 & 1 & 1 & 0 \end{pmatrix}.$$

Através das respectivas decomposições SVD, os vetores singulares são

$$\mathbf{U}^{(1)} = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathbf{U}^{(2)} = \begin{pmatrix} -0.8506 & 0.5257 \\ 0.5257 & 0.8506 \end{pmatrix} \quad \text{e} \quad \mathbf{U}^{(3)} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

e o tensor núcleo pode ser determinado por

$$\mathbf{G}_{(1)} = \begin{pmatrix} -0.5257 & -1.3764 & -0.8507 & -0.3249 \\ 0.8507 & 0.3249 & -0.5257 & -1.3764 \end{pmatrix}.$$

Como esperado, se fixamos um modo do tensor, o produto interno entre as duas fatias será zero para qualquer modo. Além disso, a norma dessas fatias são os valores singulares associados, que são dados por

$$\begin{aligned} \text{modo 1 : } & 1.7321, 1.7321 \\ \text{modo 2 : } & 2.0000, 1.4142 \\ \text{modo 3 : } & 1.7321, 1.7321. \end{aligned}$$

Os valores singulares são todos positivos, indicando que o posto multilinear de  $\mathcal{X}$  é  $(2, 2, 2)$ .

Destacamos que dado um tensor  $\mathcal{X}$ , a melhor aproximação de posto multilinear  $(P, Q, R)$  não necessariamente é obtida truncando a decomposição HOSVD, diferentemente do caso matricial. Considere o seguinte problema: dado  $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ , encontre um tensor  $\hat{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$  com posto multilinear  $(P, Q, R)$  que é solução do problema

$$\min_{\hat{\mathcal{X}}} \|\mathcal{X} - \hat{\mathcal{X}}\|_F^2.$$

É possível decompor  $\hat{\mathcal{X}}$  como

$$\hat{\mathcal{X}} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)},$$

com  $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$ ,  $\mathbf{U}^{(1)} \in \mathbb{R}^{I \times P}$ ,  $\mathbf{U}^{(2)} \in \mathbb{R}^{J \times Q}$  e  $\mathbf{U}^{(3)} \in \mathbb{R}^{K \times R}$  ortogonais. Assim, o objetivo é determinar estas matrizes e o núcleo que minimizam a função objetivo. Através da análise por multiplicadores de Lagrange, a abordagem é feita por uma extensão de alta ordem do algoritmo da Iteração QR para matrizes. O desenvolvimento e exemplificação do método são feitos com detalhes em [13].

No Exemplo 1.10 ilustramos uma aplicação para a decomposição HOSVD.

### Exemplo 1.10

A decomposição HOSVD pode ser utilizada para aplicações relacionadas à reconstrução de imagens. Sabe-se que uma imagem é formada por *pixels* dentro de um espectro RGB (*red*, *green*, *blue*) que carregam consigo valores entre 0 e 255. Para valores perto de 0, os *pixels* estão mais próximos da cor com intensidade máxima e para valores perto de 255 estão mais próximos da cor branca.

Considere a Figura 7, de dimensões  $427 \times 640$  *pixels*.



Figura 7 – Figura colorida retirada de <https://www.wikiaves.com.br/wiki/tucanucu>.

[H] Cada cor exibida por um *pixel* é uma combinação de cores do espectro RGB, de maneira que a figura acima, na realidade, pode ser separada em outras três figuras, uma em tons vermelhos, outra em tons verdes e outra em azuis. Neste sentido, podemos organizar essa estrutura de dados em um tensor  $\mathcal{X} \in \mathbb{R}^{427 \times 640 \times 3}$ . A partir da HOSVD, podemos reconstruir a imagem acima utilizando menos informações e memória.

Na Figura 8 exibimos duas decomposições, uma com posto multilinear pequeno e outra com posto maior

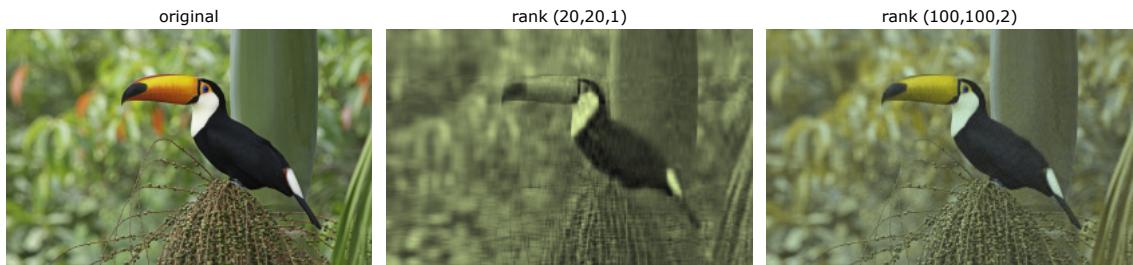


Figura 8 – Imagem original e duas reconstruções.

Observe que no primeiro caso, o tensor núcleo é de dimensão  $20 \times 20 \times 1$ , ou seja, poucas informações dos *pixels* e uma única cor foram empregadas. Já no segundo caso, com núcleo de dimensão  $100 \times 100 \times 2$ , além da quantidade de informações maior, duas cores do espectro poderiam ser utilizadas o que resultou em uma reconstrução de maior qualidade.

É importante ressaltar que ambas as decomposições acima não são as melhores aproximações do tensor original com posto multilinear  $(20, 20, 1)$  e  $(100, 100, 2)$ , como discutimos anteriormente. Todavia, é visível que as aproximações obtidas são bons pontos de partida para a reconstrução da imagem dada.

### 1.3 Decomposição *Tensor Train*

A aproximação de tensores de baixo posto é extremamente útil para a extração de informações relevantes de um volume massivo de dados. Como relembramos anteriormente, no caso matricial, o teorema de Eckart-Young [16] proporciona fundamento teórico para a existência e o cálculo desta aproximação via decomposição SVD.

A generalização deste problema para tensores de ordem maior pode ser abordada através das definições de posto e decomposições tensoriais apresentadas anteriormente. No caso da decomposição PARAFAC, somos capazes de fatorar um tensor  $I \times I \times \dots \times I$  de ordem  $N$  e posto  $R$  em tensores de posto 1. Computacionalmente, serão estimados por volta de  $N \cdot I \cdot R$  parâmetros por algum método iterativo. Todavia, existem diversos empecilhos que tornam o uso dessa decomposição inviável para tensores de ordens mais altas. Por exemplo, em geral, os métodos iterativos empregados não possuem garantia de convergência para a solução. Além disso, o conjunto de tensores de posto  $R$  não é fechado, tornando o problema muitas vezes mal-posto.

Ao tentarmos generalizar a decomposição SVD para o caso de alta ordem através da HOSVD, o cômputo dos fatores é feito sem necessidade de método iterativo. No entanto, considerando um tensor  $I \times I \times \dots \times I$  de ordem  $N$  e posto multilinear  $(R, R, \dots, R)$  serão estimados por volta de  $N \cdot I \cdot R + I^N$  parâmetros. Note que o número de dados armazenados cresce exponencialmente com a ordem do tensor. Neste sentido, no contexto do processamento massivo de dados, o uso da decomposição HOSVD é inviável.

Um dos maiores desafios no que diz respeito ao processamento de tensores de alta ordem é a chamada *maldição da dimensionalidade*. Identificado por Oseledets em [39], este problema refere-se ao crescimento exponencial do custo computacional necessário para armazenar e manipular tensores de ordens mais altas, como no caso da HOSVD. Para o caso da decomposição PARAFAC, os custos crescem linearmente com a ordem do tensor. No entanto, não é a melhor forma de abordar os problemas devido aos diversos empecilhos citados anteriormente.

Portanto, para os problemas com grandes conjuntos de dados, é desejável utilizar decomposições que não necessitem de métodos iterativos e que não sofram da maldição de dimensionalidade. Em [39], Oseledets apresenta a Decomposição *Tensor Train* (TT), como uma forma de fatorar um tensor de ordem  $N$  em um produto de tensores de ordem 3. A chave deste método são os múltiplos usos de desdobramentos e suas decomposições SVD aliados à reconstrução dos tensores de ordem 3 através de uma indexação coerente. Desse modo, é possível representar e operar com tensores de ordens mais altas neste novo formato utilizando menos recursos computacionais. Apresentaremos a seguir as definições e formas de computar a decomposição TT, bem como exibiremos exemplos para fixação das ideias.

### Representando Tensores no formato **Tensor Train**

**Definição 1.11 (Formato TT)** Sejam  $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$  e escalares  $r_0, \dots, r_N \in \mathbb{R}$ . Além disso, sejam  $\mathcal{G}^{(k)} \in \mathbb{R}^{r_{k-1} \times I_k \times r_k}$  tensores de ordem 3. Dizemos que  $\mathcal{A}$  está no formato Tensor Train (TT) se suas entradas são escritas como

$$a_{i_1, i_2, \dots, i_N} = \sum_{\alpha_0=1}^{r_0} \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_N=1}^{r_N} g_{\alpha_0, i_1, \alpha_1}^{(1)} g_{\alpha_1, i_2, \alpha_2}^{(2)} \cdots g_{\alpha_{N-1}, i_N, \alpha_N}^{(N)}, \quad (1.12)$$

ou então, matricialmente,

$$a_{i_1, i_2, \dots, i_N} = \mathbf{G}_{:, i_1, :}^{(1)} \mathbf{G}_{:, i_2, :}^{(2)} \cdots \mathbf{G}_{:, i_N, :}^{(N)}. \quad (1.13)$$

Denominamos os tensores  $\mathcal{G}^{(k)}$  como núcleos-TT e os números  $r_k$  como posto-TT para  $k = 0, \dots, N$ . Ainda, pela definição,  $r_0 = r_N = 1$ .

A representação acima pode ser visualizada a partir de um grafo onde os nós indicam os núcleos-TT e as arestas indicam seus respectivos índices. Nós ligados por arestas significa que existe um somatório sobre o índice exibido. O esquema gerado por esta representação gráfica se parece com um trem onde os nós são os vagões, justificando o nome dado à decomposição, conforme ilustrado na Figura 9. Note também que o primeiro e o último vagões são matrizes.

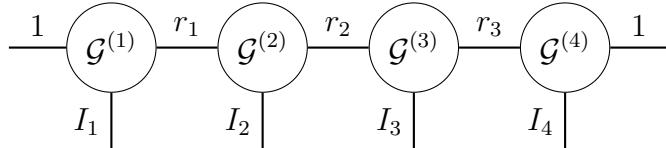


Figura 9 – Representação gráfica da decomposição TT de um tensor de ordem 4 com posto-TT igual a  $(r_1, r_2, r_3)$ .

Dado um tensor  $I \times I \times \dots \times I$  de ordem  $N$  e posto-TT limitado por  $R$ , serão armazenados por volta de  $(N - 2)IR^2 + 2IR$  parâmetros. Assim, o custo computacional não cresce exponencialmente com a ordem do tensor. Além disso, veremos que é possível calcular o formato TT (considerando uma tolerância) utilizando decomposições SVD de matrizes auxiliares, sendo desnecessário o uso de método iterativo. A partir da decomposição TT, a maldição de dimensionalidade deixará de ser um grande problema. A pergunta que fica é: como realizar operações importantes da álgebra linear, como soma de tensores, produto interno, norma e produto matriz-vetor, neste novo formato? Antes de respondê-la, vejamos um exemplo ilustrativo da decomposição TT (Exemplo 1.11), bem como resultados associados à construção propriamente dita.

### Exemplo 1.11

Este exemplo é inspirado nos slides criados por Anton Rodomanov<sup>1</sup>.

Considere  $\mathcal{A} \in \mathbb{R}^{3 \times 3 \times 3 \times 3}$  cujas entradas são dadas por

$$a_{i_1, i_2, i_3, i_4} = i_1 + i_2 + i_3 + i_4,$$

com  $i_1, i_2, i_3, i_4 \in \{1, 2, 3\}$ . Podemos escrever analiticamente a representação TT deste tensor. Da equação (1.13), segue que

$$a_{i_1, i_2, i_3, i_4} = \mathbf{G}_{:, i_1, :}^{(1)} \mathbf{G}_{:, i_2, :}^{(2)} \mathbf{G}_{:, i_3, :}^{(3)} \mathbf{G}_{:, i_4, :}^{(4)}$$

onde

$$a_{i_1, i_2, i_3, i_4} = \begin{pmatrix} i_1 & 1 \\ i_2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ i_3 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ i_4 \end{pmatrix}.$$

Então, agrupando todas as fatias, os núcleos-TT são

$$\begin{aligned} \mathcal{G}^{(1)} &= \left\{ \begin{pmatrix} 1 & 1 \\ 2 & 1 \end{pmatrix}, \begin{pmatrix} 3 & 1 \end{pmatrix} \right\}, \\ \mathcal{G}^{(2)} &= \left\{ \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 3 & 1 \end{pmatrix} \right\}, \\ \mathcal{G}^{(3)} &= \left\{ \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 3 & 1 \end{pmatrix} \right\}, \\ \mathcal{G}^{(4)} &= \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 3 \end{pmatrix} \right\}. \end{aligned}$$

O tensor  $\mathcal{A}$  possui 243 elementos. Já sua representação TT possui apenas 36 elementos com postos-TT  $(2, 2, 2)$ .

Para comparação, também calculamos a decomposição PARAFAC deste mesmo tensor através do método ALS com posto  $R = 8$ , obtendo erro de  $8.0927 \cdot 10^{-8}$ . Assim, ao total são necessários  $4 \cdot 3 \cdot 8 = 96$  parâmetros.

No caso da decomposição HOSVD com posto multilinear  $(2, 2, 2, 2)$ , o número de elementos necessários são  $2^4 = 16$  para o tensor núcleo e  $4 \cdot 3 \cdot 2 = 24$  para as matrizes. Assim, ao total são necessários 40 parâmetros para descrever o tensor  $\mathcal{A}$  por meio da decomposição HOSVD.

<sup>1</sup> Disponíveis em [https://bayesgroup.github.io/team/arodomanov/tt\\_hse16\\_slides.pdf](https://bayesgroup.github.io/team/arodomanov/tt_hse16_slides.pdf)

A decomposição TT e os postos-TT de um tensor podem ser obtidos através do teorema que demonstramos a seguir (cf. [39, Teorema 2.1]):

**Teorema 1.2** Considere um tensor  $\mathcal{A} \in \mathbb{R}^{I_1 \times \dots \times I_N}$ , cujos desdobramentos satisfazem  $\text{posto}(\mathbf{A}_{\langle k \rangle}) = r_k$ . Então  $\mathcal{A}$  pode ser escrito no formato TT com postos-TT não maiores que  $r_k$ .

Prova: Considere o desdoblamento de modo-1  $\mathbf{A}_{\langle 1 \rangle}$ . Pela decomposição SVD temos

$$\mathbf{A}_{\langle 1 \rangle} = \mathbf{U} \mathbf{S} \hat{\mathbf{V}}^T = \mathbf{U} \mathbf{V}^T, \quad (1.14)$$

onde  $\mathbf{U}$  e  $\mathbf{V}$  possuem  $r_1$  colunas orthonormais e  $\mathbf{V} = \hat{\mathbf{V}} \mathbf{S}$ . Em termos de entradas, resulta que

$$a_{i_1,j} = \sum_{\alpha_1=1}^{r_1} u_{i_1,\alpha_1} v_{\alpha_1,j}$$

com  $j := \overline{i_2 \dots i_N}$  (cf. notação definida em (1.2)). Tome  $\mathcal{G}_{1,:}^{(1)} = \mathbf{U}$  tal que  $\mathcal{G}^{(1)} \in \mathbb{R}^{r_0 \times I_1 \times r_1}$  e  $r_0 = 1$ . Em seguida, podemos reestruturar  $\mathbf{V}$  em um tensor  $\mathcal{V} \in \mathbb{R}^{r_1 I_2 \times I_3 \times \dots \times I_N}$  de ordem  $N - 1$  juntando  $\alpha_1$  e  $i_2$  em um único índice. Vamos mostrar que os desdoblamentos  $\mathbf{V}_{\langle k \rangle}$  possuem posto menores que ou iguais a  $r_k$  para  $k = 2, \dots, N$ . De (1.14), vale que  $\mathbf{V} = \mathbf{A}_{\langle 1 \rangle}^T \mathbf{U}$ . Então

$$v_{\alpha_1 i_2, i_3, \dots, i_N} = \sum_{i_1=1}^{I_1} a_{i_1, i_2, \dots, i_N} u_{i_1, \alpha_1}. \quad (1.15)$$

Do fato que  $\text{posto}(\mathbf{A}_{\langle k \rangle}) = r_k$ , como feito em (1.14), existem matrizes  $\mathbf{F}$  e  $\mathbf{G}$  com colunas orthonormais tais que  $\mathbf{A}_{\langle k \rangle} = \mathbf{F} \mathbf{G}^T$ . Assim, definindo  $j_1 := \overline{i_1 \dots i_k}$  e  $j_2 := \overline{i_{k+1} \dots i_N}$ , temos

$$a_{i_1, \dots, i_N} = a_{j_1, j_2} = \sum_{s=1}^{r_k} f_{j_1, s} g_{s, j_2}.$$

Desse modo, as entradas do  $k$ -ésimo desdoblamento de  $\mathcal{V}$  são dadas por

$$\begin{aligned} v_{\overline{(\alpha_1 i_2) \dots i_k}, \overline{i_{k+1} \dots i_N}} &= \sum_{i_1=1}^{I_1} a_{i_1, \dots, i_N} u_{i_1, \alpha_1} \\ &= \sum_{i_1=1}^{I_1} \sum_{s=1}^{r_k} f_{j_1, s} g_{s, j_2} u_{i_1, \alpha_1} \\ &= \sum_{s=1}^{r_k} h_{\alpha_1 i_2, \dots, i_k, s} g_{s, j_2} \end{aligned}$$

onde

$$h_{\alpha_1 i_2, \dots, i_k, s} = \sum_{i_1=1}^{I_1} f_{j_1, s} u_{i_1, \alpha_1}.$$

Observe que fomos capazes de escrever  $\mathbf{V}_{\langle k \rangle} = \mathbf{H} \mathbf{G}^T$  como em (1.14), o que significa que  $\text{posto}(\mathbf{V}_{\langle k \rangle}) \leq r_k$  e, portanto, podemos fazer a separação de índices aplicando a SVD em  $\mathbf{V}_{\langle 2 \rangle}$ . Assim, existem  $\mathbf{U}'$  e  $\mathbf{V}'$  com colunas orthonormais tais que

$$v_{\alpha_1 i_2, \dots, i_N} = \sum_{\alpha_2=1}^{r_2} u'_{\alpha_1 i_2, \alpha_2} v'_{\alpha_2 i_3, \dots, i_N} = \sum_{\alpha_2=1}^{r_2} g_{\alpha_1, i_2, \alpha_2} v'_{\alpha_2 i_3, i_4, \dots, i_N}$$

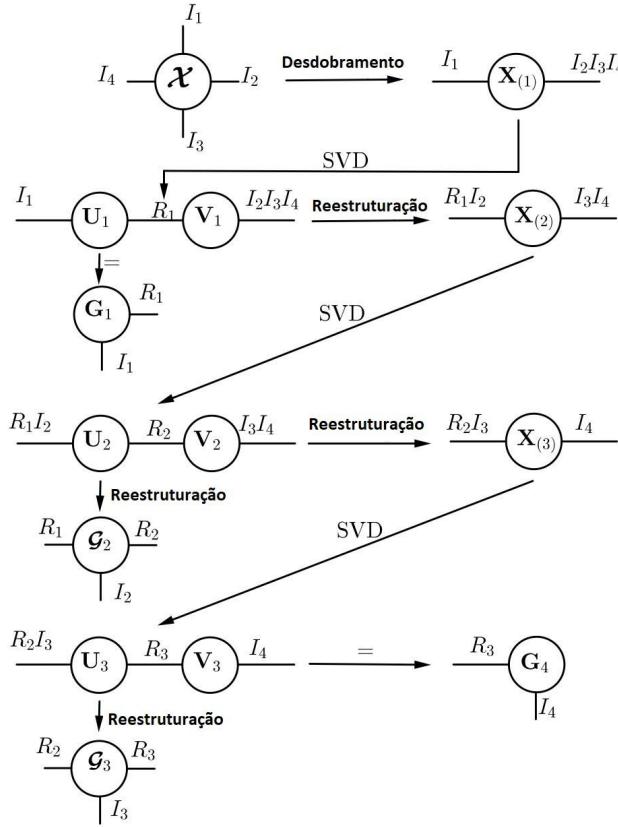


Figura 10 – Passo a passo da decomposição TT-SVD de um tensor  $\mathcal{X}$  de ordem 4 e postos-TT iguais a  $R_1, R_2$  e  $R_3$ . Retirado de [56, Fig 2.4]

onde  $\mathcal{G}^{(2)} \in \mathbb{R}^{r_1 \times I_2 \times r_2}$  é obtido reestruturando a matriz  $\mathbf{U}'$ , e  $\mathbf{V}'$  pode ser vista como um tensor de ordem  $N - 2$  juntando  $\alpha_2$  e  $i_3$  em um único índice. Neste sentido, prosseguimos de maneira análoga como feito anteriormente: separando os índices e realizando a reestruturação das matrizes resultantes da decomposição SVD. Ao final, obtemos todos os núcleos-TT e a representação TT de  $\mathcal{A}$ .  $\square$

A demonstração do Teorema 1.2 é construtiva e fornece a ideia de um algoritmo para o cálculo da fatoração TT através do uso de decomposições SVD. Este algoritmo será denominado TT-SVD. Esquematicamente, a Figura 10 nos permite acompanhar as etapas desse algoritmo para um tensor de ordem 4.

Para melhor compreensão das ideias, vamos determinar, no Exemplo 1.12, a decomposição TT de um tensor  $2 \times 2 \times 2$ .

### Exemplo 1.12

Considere  $\mathcal{A} \in \mathbb{R}^{2 \times 2 \times 2}$  cujas fatias frontais são

$$\mathbf{A}_{\cdot,\cdot,1} = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \quad \text{e} \quad \mathbf{A}_{\cdot,\cdot,2} = \begin{pmatrix} 5 & 7 \\ 6 & 8 \end{pmatrix}.$$

Prosseguiremos de maneira análoga à demonstração do Teorema 1.2. O primeiro

desdobramento é

$$\mathbf{M} = \begin{pmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \end{pmatrix}.$$

Computando sua decomposição SVD obtemos

$$\mathbf{U} = \begin{pmatrix} -0.64142 & -0.76719 \\ -0.76719 & 0.64142 \end{pmatrix},$$

$$\mathbf{S} = \begin{pmatrix} 14.26910 & 0 & 0 & 0 \\ 0 & 0.62683 & 0 & 0 \end{pmatrix},$$

$$\mathbf{V}^T = \begin{pmatrix} -0.15248 & -0.34992 & -0.54735 & -0.74479 \\ 0.82265 & 0.42137 & 0.02010 & -0.38117 \\ -0.39450 & 0.24280 & 0.69791 & -0.54620 \\ -0.37996 & 0.80066 & -0.46143 & 0.04074 \end{pmatrix}.$$

Pode-se concluir que  $r_1 = 2$  e  $\mathcal{G}_{1,:,:}^{(1)} = \mathbf{U}$ . Ainda, são necessárias as primeiras  $r_1$  colunas de  $\mathbf{V}$  e agora fazemos  $\mathbf{M} = (\mathbf{V}\mathbf{S})^T$ . Em seguida, reestruturamos a matriz  $\mathbf{M}$  para que tenha dimensões  $r_1 \cdot I_1 \times I_2 = (2 \cdot 2) \times 2$ . Portanto,

$$\mathbf{M} = \begin{pmatrix} -2.17580 & -7.81024 \\ 0.51566 & 0.01260 \\ -4.99302 & -10.62750 \\ 0.26413 & -0.23893 \end{pmatrix}.$$

Agora, aplicamos a decomposição SVD sobre a nova matriz  $\mathbf{M}$ :

$$\mathbf{U} = \begin{pmatrix} -0.56617 & -0.73328 & -0.34859 & -0.14229 \\ 0.01445 & -0.37623 & 0.88272 & -0.28116 \\ -0.82412 & 0.49993 & 0.25300 & 0.08230 \\ -0.00858 & -0.26613 & 0.18784 & 0.94542 \end{pmatrix},$$

$$\mathbf{S} = \begin{pmatrix} 14.22741 & 0 \\ 0 & 1.25733 \\ 0 & 0 \\ 0 & 0 \end{pmatrix},$$

$$\mathbf{V}^T = \begin{pmatrix} 0.37617 & 0.92655 \\ -0.92655 & 0.37617 \end{pmatrix}.$$

Portanto, segue que  $r_2 = 2$  e serão necessárias as primeiras  $r_2$  colunas de  $\mathbf{U}$  e  $\mathbf{V}$ . O núcleo-TT  $\mathcal{G}^{(2)}$  é obtido reestruturando  $\mathbf{U}$  em um tensor de dimensões  $r_1 \times I_2 \times r_2 = 2 \times 2 \times 2$ . Por fim, considere o tensor  $\mathcal{G}_{1,:,:}^{(3)} = \mathbf{V}\mathbf{S}$  como o último

núcleo-TT. Ao final, temos

$$\begin{aligned}\mathcal{G}^{(1)} &= \begin{pmatrix} -0.64142 & -0.76719 \\ -0.76719 & 0.64142 \end{pmatrix}, \\ \mathcal{G}_{:,1}^{(2)} &= \begin{pmatrix} -0.56617 & -0.82412 \\ 0.01445 & -0.00858 \end{pmatrix}, \quad \mathcal{G}_{:,2}^{(2)} = \begin{pmatrix} -0.73328 & 0.499928 \\ -0.37623 & -0.26613 \end{pmatrix}, \\ \mathcal{G}^{(3)} &= \begin{pmatrix} 5.35190 & -1.16498 \\ 13.18240 & 0.47297 \end{pmatrix}.\end{aligned}$$

O próximo teorema é uma extensão do Teorema 1.2 ao caso em que os desdobramentos  $\mathbf{A}_{<k>}$  podem ser aproximados por matrizes de posto baixo com uma certa precisão. Antes, será necessário demonstrar um resultado auxiliar.

**Lema 1.1** *Considere uma matriz  $\mathbf{A} \in \mathbb{R}^{m \times n}$  e sua decomposição SVD*

$$\mathbf{A} = \mathbf{U} \mathbf{S} \mathbf{V}^T,$$

onde  $\mathbf{U}$  e  $\mathbf{V}$  são ortogonais e  $\mathbf{S}$  é a matriz diagonal contendo todos os valores singulares de  $\mathbf{A}$ . Considere também a aproximação de posto  $r$  dada pela matriz

$$\mathbf{B} = \hat{\mathbf{U}} \hat{\Sigma} \hat{\mathbf{V}}^T$$

com erro dado pela matriz  $\mathbf{E}$  e

$$\Sigma = \begin{bmatrix} \hat{\Sigma} & \mathbf{0}_{r \times (n-r)} \\ \mathbf{0}_{(m-r) \times r} & \mathbf{0}_{(m-r) \times (n-r)} \end{bmatrix}$$

onde  $\hat{\Sigma}$  é diagonal contendo os  $r$  primeiros valores singulares de  $\mathbf{A}$ . As matrizes  $\hat{\mathbf{U}}$  e  $\hat{\mathbf{V}}$  são formadas pelas  $r$  primeiras colunas de  $\mathbf{U}$  e  $\mathbf{V}$  respectivamente. Então vale que

$$\hat{\mathbf{U}}^T \mathbf{E} = \mathbf{0}_{r \times n}.$$

Prova: Por hipótese,  $\mathbf{A} = \mathbf{B} + \mathbf{E}$ . Sendo assim

$$\begin{aligned}\hat{\mathbf{U}}^T \mathbf{E} &= \hat{\mathbf{U}}^T (\mathbf{A} - \mathbf{B}) \\ &= \hat{\mathbf{U}}^T \mathbf{U} \mathbf{S} \mathbf{V}^T - \hat{\mathbf{U}}^T \hat{\mathbf{U}} \hat{\Sigma} \hat{\mathbf{V}}^T.\end{aligned}$$

Note que  $\hat{\mathbf{U}}^T \mathbf{U} = [\mathbf{I}_{r \times r} \quad \mathbf{0}_{r \times (m-r)}]$ . Então

$$\hat{\mathbf{U}}^T \mathbf{E} = \hat{\mathbf{U}}^T \mathbf{U} \mathbf{S} \mathbf{V}^T - \hat{\mathbf{U}}^T \hat{\mathbf{U}} \hat{\Sigma} \hat{\mathbf{V}}^T = \hat{\Sigma} \hat{\mathbf{V}}^T - \hat{\Sigma} \hat{\mathbf{V}}^T = \mathbf{0}_{r \times n}.$$

□

**Teorema 1.3** Dado  $\mathcal{A} \in \mathbb{R}^{I_1 \times \dots \times I_N}$  existe uma decomposição TT, denotada por  $\mathcal{T} \in \mathbb{R}^{I_1 \times \dots \times I_N}$ , tal que

$$\|\mathcal{A} - \mathcal{T}\|_F^2 \leq \sum_{k=1}^{N-1} \varepsilon_k^2$$

onde

$$\varepsilon_k = \min_{\text{posto}(\mathbf{B}) \leq r_k} \|\mathbf{A}_{<k>} - \mathbf{B}\|_F.$$

Prova: A prova é feita por indução sobre a dimensão  $N$ . Para o caso matricial  $N = 2$ , a decomposição TT é dada por

$$t_{i_1, i_2} = \sum_{\alpha_1=1}^{r_1} g_{i_1, \alpha_1}^{(1)} g_{\alpha_1, i_2}^{(2)},$$

construída da seguinte maneira: considere a decomposição SVD de  $\mathbf{A}$ , isto é,  $\mathbf{A} = \mathbf{U} \mathbf{S} \mathbf{V}^T$  onde  $\mathbf{U}$  e  $\mathbf{V}$  são matrizes ortogonais e  $\mathbf{S}$  é matriz diagonal contendo os valores singulares  $\sigma_1 \geq \sigma_2 \geq \dots \geq 0$ . Tomamos  $\mathbf{G}^{(1)}$  e  $\mathbf{G}^{(2)}$  como as matrizes formadas pelas  $r_1$  primeiras colunas de  $\mathbf{U}$  e  $\mathbf{S} \mathbf{V}^T$ , respectivamente. Desse modo asseguramos que a distância  $\|\mathbf{A} - \mathbf{T}\|_F$  é a menor possível devido ao teorema de Eckart-Young. Agora considere  $N > 2$  arbitrário. Vamos tomar o desdobramento de modo-1 de  $\mathcal{A}$ . Sua decomposição SVD é dada por

$$\mathbf{A}_{<1>} = \mathbf{U} \mathbf{S} \mathbf{V}^T. \quad (1.16)$$

A melhor aproximação de posto  $r_1$  é a matriz

$$\mathbf{B}^{(1)} = \mathbf{U}^{(1)} \boldsymbol{\Sigma} \mathbf{V}^{(1)T}, \quad (1.17)$$

onde  $\boldsymbol{\Sigma} = \text{diag}(\sigma_1, \dots, \sigma_{r_1})$ . As matrizes  $\mathbf{U}^{(1)}$  e  $\mathbf{V}^{(1)}$  são formadas pelas  $r_1$  primeiras colunas de  $\mathbf{U}$  e  $\mathbf{V}$ , respectivamente. Portanto,

$$\mathbf{A}_{<1>} = \mathbf{B}_{<1>} + \mathbf{E}_{<1>} \quad \text{com} \quad \text{posto}(\mathbf{B}_{<1>}) \leq r_1 \quad \text{e} \quad \|\mathbf{E}_{<1>}\|_F = \varepsilon_1.$$

Observe que podemos considerar  $\mathbf{B}_{<1>}$  e  $\mathbf{E}_{<1>}$  como desdobramentos de tensores  $[\mathcal{B}]_{i_1, \dots, i_N}$  e  $[\mathcal{E}]_{i_1, \dots, i_N}$  respectivamente. Da equação (1.17), fazendo  $\hat{\mathbf{A}} = \boldsymbol{\Sigma} \mathbf{V}^{(1)T}$ , resulta que

$$b_{i_1, \overline{i_2 \dots i_N}} = \sum_{\alpha_1=1}^{r_1} u_{i_1, \alpha_1}^{(1)} \hat{a}_{\alpha_1, \overline{i_2 \dots i_N}}. \quad (1.18)$$

Concatenando os índices  $\alpha_1$  e  $i_2$  de  $\hat{\mathbf{A}}$  em um longo índice  $\alpha_1 i_2$ , podemos reescrever esta como um tensor de  $N - 1$  dimensões, isto é,

$$\hat{\mathcal{A}} = [\hat{\mathcal{A}}]_{\alpha_1 i_2, i_3, \dots, i_N}.$$

Pela hipótese de indução,  $\hat{\mathcal{A}}$  admite uma aproximação TT, denotada por  $\hat{\mathcal{T}}$ , cujas entradas são dadas por

$$\hat{t}_{\alpha_1 i_2, i_3, \dots, i_N} = \sum_{\alpha_2=1}^{r_2} \dots \sum_{\alpha_{N-1}=1}^{r_{N-1}} g_{\alpha_1 i_2, \alpha_2}^{(2)} g_{\alpha_2, i_3, \alpha_3}^{(3)} \dots g_{\alpha_{N-1}, i_N}^{(N-1)},$$

tal que

$$\|\hat{\mathcal{A}} - \hat{\mathcal{T}}\|_F^2 \leq \sum_{k=2}^{N-1} \hat{\varepsilon}_k^2$$

e

$$\hat{\varepsilon}_k = \min_{\text{posto}(\hat{\mathbf{B}}) \leq r_k} \|\hat{\mathbf{A}}_{<k>} - \hat{\mathbf{B}}\|_F.$$

Note que tomando  $g_{i_1, \alpha_1}^{(1)} = u_{i_1, \alpha_1}^{(1)}$  e separando os índices  $\alpha_1$  e  $i_2$ , encontramos a seguinte aproximação TT para  $\mathcal{A}$ , denotada por  $\mathcal{T}$ , tal que

$$t_{i_1, i_2, \dots, i_N} = \sum_{\alpha_1=1}^{r_1} \cdots \sum_{\alpha_{N-1}=1}^{r_{N-1}} g_{i_1, \alpha_1}^{(1)} g_{\alpha_1, i_2, \alpha_2}^{(2)} \cdots g_{\alpha_{N-1}, i_N}^{(N-1)}, \quad (1.19)$$

onde

$$\hat{\mathcal{T}} = \mathcal{T} \times_1 \mathbf{U}^{(1)} \Leftrightarrow \hat{\mathbf{T}}_{<1>} = \mathbf{U}^{(1)} \mathbf{T}_{<1>}. \quad (1.20)$$

Agora será necessário estimar a distância  $\|\mathcal{A} - \mathcal{T}\|_F$ . Note que

$$\|\mathcal{A} - \mathcal{T}\|_F^2 = \|(\mathcal{A} - \mathcal{B}) + (\mathcal{B} - \mathcal{T})\|_F^2 \leq \varepsilon_1^2 + \|\mathcal{B} - \mathcal{T}\|_F^2. \quad (1.21)$$

Para estimar  $\|\mathcal{B} - \mathcal{T}\|_F$ , invocamos o Lema 1.1 de modo que

$$\hat{\mathbf{A}} = \mathbf{U}^{(1)^T} \mathbf{B}_{<1>} = \mathbf{U}^{(1)^T} (\mathbf{A}_{<1>} - \mathbf{E}_{<1>}) = \mathbf{U}^{(1)^T} \mathbf{A}_{<1>}.$$

A partir daqui vamos buscar escrever  $\hat{\mathbf{A}}$  a partir de uma representação de baixo posto. Neste sentido, da expressão acima, resulta que

$$\hat{a}_{\alpha_1 i_2, i_3, \dots, i_N} = \sum_{i_1=1}^{I_1} u_{i_1, \alpha_1}^{(1)} a_{i_1, i_2, \dots, i_N}. \quad (1.22)$$

Considere que  $\mathbf{A}_{<k>} = \mathbf{B}^{(k)} + \mathbf{E}^{(k)}$  com  $\text{posto}(\mathbf{B}^{(k)}) \leq r_k$  e  $\|\mathbf{E}^{(k)}\|_F = \varepsilon_k$ . Observe que  $\mathbf{B}^{(k)}$  admite uma decomposição esqueletal (cf. Apêndice B)

$$\mathbf{B}^{(k)} = \mathbf{F}^{(k)} \mathbf{H}^{(k)^T}$$

tal que

$$a_{i_1, i_2, \dots, i_N} = a_{\overline{i_1 \dots i_k}, \overline{i_{k+1} \dots i_N}} = \sum_{s=1}^{r_k} f_{\overline{i_1 \dots i_k}, s}^{(k)} h_{s, \overline{i_{k+1} \dots i_N}}^{(k)} + e_{i_1, i_2, \dots, i_N}^{(k)}.$$

Substituindo na equação (1.22), segue que

$$\begin{aligned} \hat{a}_{\alpha_1 i_2, i_3, \dots, i_N} &= \sum_{i_1=1}^{I_1} u_{i_1, \alpha_1}^{(1)} \left( \sum_{s=1}^{r_k} f_{\overline{i_1 \dots i_k}, s}^{(k)} h_{s, \overline{i_{k+1} \dots i_N}}^{(k)} \right) + \sum_{i_1=1}^{I_1} u_{i_1, \alpha_1}^{(1)} e_{i_1, i_2, \dots, i_N}^{(k)} \\ &= \sum_{s=1}^{r_k} \left( \sum_{i_1=1}^{I_1} u_{i_1, \alpha_1}^{(1)} f_{\overline{i_1 \dots i_k}, s}^{(k)} \right) h_{s, \overline{i_{k+1} \dots i_N}}^{(k)} + r_{\alpha_1, \overline{i_2 \dots i_N}}^{(k)} \\ &= \sum_{s=1}^{r_k} p_{\alpha_1 \overline{i_2 \dots i_k}, s}^{(k)} h_{s, \overline{i_{k+1} \dots i_N}}^{(k)} + r_{\alpha_1, \overline{i_2 \dots i_N}}^{(k)} \\ &= q_{\alpha_1 \overline{i_2 \dots i_k}, \overline{i_{k+1} \dots i_N}}^{(k)} + r_{\alpha_1, \overline{i_2 \dots i_N}}^{(k)}, \end{aligned}$$

onde  $\mathbf{P}^{(k)} = \mathbf{U}^{(1)T} \mathbf{F}^{(k)}$ ,  $\mathbf{R}^{(k)} = \mathbf{U}^{(1)T} \mathbf{E}^{(k)}$  e  $\mathbf{Q}^{(k)} = \mathbf{P}^{(k)} \mathbf{H}^{(k)T}$ . Note que a matriz  $\mathbf{Q}^{(k)}$  está representada em uma forma esquelética e portanto seu posto não é maior que  $r_k$ . Desse modo, fomos capazes de escrever  $\hat{\mathbf{A}}$  como a soma de uma matriz de baixo posto  $\mathbf{Q}^{(k)}$  com outra que contém o erro cometido, representada pela matriz  $\mathbf{R}^{(k)}$ . Portanto, podemos reestruturar as matrizes  $\mathbf{Q}^{(k)}$  e  $\mathbf{R}^{(k)}$  como tensores  $\mathcal{Q}^{(k)}$  e  $\mathcal{R}^{(k)}$  de forma que

$$\hat{a}_{\alpha_1, i_2, \dots, i_N} = q_{\alpha_1, i_2, \dots, i_N}^{(k)} + r_{\alpha_1, i_2, \dots, i_N}^{(k)}.$$

Sabendo que  $\hat{\varepsilon}_k$  é a distância (na norma de Frobenius) entre  $\hat{\mathbf{A}}$  e sua melhor aproximação de posto  $r_k$ , temos que

$$\hat{\varepsilon}_k \leq \|\mathbf{R}^{(k)}\|_F = \|\mathbf{U}^{(1)T} \mathbf{E}^{(k)}\|_F \leq \|\mathbf{E}^{(k)}\|_F = \varepsilon_k,$$

pois a norma de Frobenius de uma matriz é invariante quando esta é pré-multiplicada por outra matriz com linhas ortonormais. Desse modo, imediatamente concluímos que

$$\|\hat{\mathcal{A}} - \hat{\mathcal{T}}\|_F^2 \leq \sum_{k=2}^{N-1} \hat{\varepsilon}_k^2 \leq \sum_{k=2}^{N-1} \varepsilon_k^2.$$

Assim, temos

$$\|\mathcal{B} - \mathcal{T}\|_F^2 = \|\mathbf{B}_{<1>} - \mathbf{T}_{<1>}\|_F^2 \stackrel{(1.18)-(1.20)}{=} \|\mathbf{U}^{(1)} \hat{\mathbf{A}} - \mathbf{U}^{(1)} \hat{\mathbf{T}}\|_F^2 \leq \sum_{k=2}^{N-1} \varepsilon_k^2.$$

Portanto, através da desigualdade em (1.21) resulta que

$$\|\mathcal{A} - \mathcal{T}\|_F^2 \leq \sum_{k=1}^{N-1} \varepsilon_k^2.$$

Pelo princípio da indução, está demonstrado o teorema para todo  $N \geq 2$ .  $\square$

**Corolário 1.1** Sejam  $\mathcal{A} \in \mathbb{R}^{I_1 \times \dots \times I_N}$  e limitantes para os postos-TT  $r_1, \dots, r_{N-1}$ . A melhor aproximação na norma de Frobenius com postos-TT limitados por  $r_k$  ( $k = 1, \dots, N-1$ ) existe e é denotada por  $\mathcal{A}'$ . A aproximação  $\mathcal{B}$  construída pelo algoritmo TT-SVD satisfaz

$$\|\mathcal{A} - \mathcal{B}\|_F \leq \sqrt{N-1} \|\mathcal{A} - \mathcal{A}'\|_F.$$

Prova: Denote por  $\mathbb{S}$  o conjunto de todos os tensores com postos-TT limitados por  $r_1, \dots, r_{N-1}$  e tome  $\varepsilon = \inf\{\|\mathcal{A} - \mathcal{B}\|_F, \mathcal{B} \in \mathbb{S}\}$ . Então, existe uma sequência  $(\mathcal{B}^{(n)})_{n \in \mathbb{N}}$  de decomposições TT tal que

$$\varepsilon \leq \|\mathcal{A} - \mathcal{B}^{(n)}\|_F \leq \varepsilon + \frac{1}{n}, \quad \forall n \in \mathbb{N}.$$

Em particular,  $\|\mathcal{A} - \mathcal{B}^{(n)}\|_F \leq \varepsilon + 1$ , para todo  $n \in \mathbb{N}$ . Logo, a sequência é limitada e existe uma subsequência convergente, para a qual, sem perda de generalidade e reordenando

os índices, temos  $\mathcal{B}^{(n)} \xrightarrow{n \in \mathbb{N}} \mathcal{A}'$ . É importante ressaltar que a convergência ocorre em cada entrada do tensor. Desse modo, vale que  $\mathcal{B}_{<k>}^{(n)} \xrightarrow{n \in \mathbb{N}} \mathcal{A}'_{<k>}$ , para  $k = 1, \dots, N - 1$ . Como o conjunto de matrizes com posto menor ou igual a  $r_k$  é fechado e  $\text{posto}(\mathcal{B}_{<k>}^{(n)}) \leq r_k$ , então  $\text{posto}(\mathcal{A}'_{<k>}) \leq r_k$  e  $\varepsilon = \|\mathcal{A} - \mathcal{A}'\|_F$ .

Cada desdobramento de  $\mathcal{A}$  pode ser aproximado com pelo menos a tolerância  $\varepsilon$  de forma que  $\varepsilon_k \leq \varepsilon$ . Então, do Teorema 1.3, temos

$$\|\mathcal{A} - \mathcal{B}\|_F \leq \sqrt{\sum_{k=1}^{N-1} \varepsilon_k^2} \leq \sqrt{N-1} \varepsilon = \sqrt{N-1} \|\mathcal{A} - \mathcal{A}'\|_F.$$

□

A partir do Teorema 1.3 e do Corolário 1.1, dada uma tolerância  $\varepsilon > 0$ , é possível truncar os valores singulares dos desdobramentos e calcular os postos-TT a partir de um limitante. De acordo com Oseledets [39], para que a aproximação da decomposição TT tenha precisão relativa  $\varepsilon$ , o limitante é escolhido como sendo  $\varepsilon(N-1)^{-1/2} \|\mathcal{A}\|_F$ . Finalmente, podemos exibir o pseudocódigo para a decomposição TT no Algoritmo 3.

---

### Algoritmo 3 – TT-SVD

---

**Entrada:**  $\mathcal{A}, \varepsilon > 0$

Compute o parâmetro  $\delta = \frac{\varepsilon}{\sqrt{N-1}} \|\mathcal{A}\|_F$ .

Faça  $\mathbf{X}_1 = \mathcal{A}_{(1)}$  e  $r_0 = 1$ .

**para**  $k = 1 : N - 1$  **faça**

Realize a SVD $_{\delta}$  de  $\mathbf{X}_k$  e obtenha  $\mathbf{U}_k, \mathbf{S}_k, \mathbf{V}_k$ .

Tome  $r_k$  como o número de colunas de  $\mathbf{U}_k$ .

Reestruture  $\mathbf{U}_k$  em um tensor de terceira ordem  $\mathcal{G}^{(k)} \in \mathbb{R}^{r_{k-1} \times I_k \times r_k}$ .

Tome  $X_{k+1}$  como a reestruturação de  $\mathbf{S}_k \mathbf{V}_k^T$  como uma matriz de dimensões

$r_k I_{k+1} \times \prod_{p=k+2}^N I_p$ .

**fim**

Reestruture  $\mathbf{X}_N$  em um tensor de terceira ordem  $\mathcal{G}^{(N)} \in \mathbb{R}^{r_{N-1} \times I_N \times 1}$ .

**retorna**  $\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \dots, \mathcal{G}^{(N)}$ .

---

### Representação TT-Cross

Em diversas aplicações, a construção de um tensor é feita entrada por entrada através de alguma relação para os índices. Considere, como ilustração, o Exemplo 1.11. Em tais casos, não é necessário armazenar completamente o tensor para operar com ele, reduzindo consideravelmente a memória necessária e permitindo trabalharmos com dimensões ainda maiores. Assim como é possível obter aproximações de matrizes através de sua decomposição esquelética (cf. (B.1) no Apêndice B), podemos estender este conceito para o algoritmo TT-SVD. De fato, se as dimensões do tensor são grandes, calcular a

decomposição SVD de seus desdobramentos pode ser inviável e, então, se torna necessário aproximá-los de outra forma. A seguir, apresentaremos o método TT-*Cross* desenvolvido por Oseledets e Tyrtshnikov em [39] a fim de explorar esta nova abordagem.

Considere um tensor  $\mathcal{A} \in \mathbb{R}^{I_1 \times \dots \times I_N}$  e assuma que seu primeiro desdoblamento  $\mathbf{A}_{<1>} = \mathbf{C}^{(1)} (\hat{\mathbf{A}}^{(1)})^{-1} \mathbf{R}^{(1)}$  possua decomposição esqueletal como em (B.1), onde  $\mathbf{C}^{(1)}$  é composta por  $r_1$  colunas de  $\mathbf{A}_{<1>}$  cujos índices compõem o conjunto

$$\mathcal{J}_1 = \{j_k^{(\alpha_1)} \mid \alpha_1 = 1, \dots, r_1, k = 2, \dots, N\}.$$

e  $\mathbf{R}^{(1)}$  é determinada por  $r_1$  linhas cujos índices compõem o conjunto

$$\mathcal{I}_1 = \{i_1^{(\alpha_1)} \mid \alpha_1 = 1, \dots, r_1\}.$$

Assim, temos que  $\mathbf{A}_{<1>} = \mathbf{C}^{(1)} (\hat{\mathbf{A}}^{(1)})^{-1} \mathbf{R}^{(1)}$  e definimos o primeiro núcleo-TT como  $\mathbf{G}_{1,:,:}^{(1)} = \mathbf{C}^{(1)} (\hat{\mathbf{A}}^{(1)})^{-1}$ . Em seguida, a matriz  $\mathbf{R}^{(1)}$  que possui  $r_1$  linhas de  $\mathbf{A}_{<1>}$  pode ser considerada como um tensor  $\mathcal{R}$  tal que

$$r_{\alpha_1, i_2, \dots, i_N} = a_{i_1^{(\alpha)}, i_2, \dots, i_N}.$$

Juntando  $\alpha_1$  e  $i_2$  em um longo único índice,  $\mathbf{R}^{(1)}$  é também um tensor de ordem  $N-1$ . Neste sentido, prosseguimos para obter os próximos núcleos-TT. Considere seu desdoblamento  $\mathbf{R}_{<2>}$  com entradas

$$\mathbf{R}_{<2>} = [\mathcal{R}]_{\alpha_1 i_2, \overline{i_3 \dots i_N}}.$$

A partir da decomposição esqueletal serão encontrados os novos núcleos-TT separando os índices e reestruturando as matrizes resultantes. A construção dos núcleos é exibida pelo teorema a seguir

**Teorema 1.4** [39, Theorem 3.1] *Seja  $\mathcal{A} \in \mathbb{R}^{I_1 \times \dots \times I_N}$  com postos-TT dados por*

$$r_k = \text{posto}(\mathbf{A}_{<k>}).$$

*Considere que os conjuntos de índices*

$$\mathcal{J}_k = \{j_l^{(\beta_1)} \mid \beta_1 = 1, \dots, r_k, l = k, \dots, N-1\}$$

*e*

$$\mathcal{I}_k = \{i_l^{(\alpha_1)} \mid \alpha_1 = 1, \dots, r_k, l = 1, \dots, k-1\}$$

*formam submatrizes não singulares de dimensões  $r_k \times r_k$  denotadas por  $\hat{\mathbf{A}}^{(k)}$ , onde*

$$\hat{a}_{\alpha_k, \beta_k}^{(k)} = a_{\overline{i_1^{(\alpha_k)} i_2^{(\alpha_k)} \dots i_{k-1}^{(\alpha_k)} i_k}, \overline{j_{k+1}^{(\beta_k)} \dots j_N^{(\beta_k)}}}.$$

*Então os tensores  $\mathcal{C}^{(k)} \in \mathbb{R}^{r_{k-1} \times I_k \times r_k}$ , com elementos*

$$c_{\alpha_k, i_k, \beta_k}^{(k)} = a_{i_1^{(\alpha_k)}, i_2^{(\alpha_k)}, \dots, i_{k-1}^{(\alpha_k)}, i_k, j_{k+1}^{(\beta_k)}, \dots, j_N^{(\beta_k)}}$$

definem núcleos

$$\begin{aligned}\hat{\mathcal{C}}^{(1)} &= \mathcal{C}^{(1)} \left( \hat{\mathbf{A}}^{(1)} \right)^{-1}, \\ \hat{\mathcal{C}}^{(k)} &= \mathcal{C}^{(k)} \times_3 \left( \hat{\mathbf{A}}^{(k)} \right)^{-1}, \quad k = 2, \dots, N-1, \\ \hat{\mathcal{C}}^{(N)} &= \mathcal{C}^{(N)},\end{aligned}$$

de forma que

$$a_{i_1, i_2, \dots, i_N} = \sum_{\alpha_1=1}^{r_1} \sum_{\alpha_2=1}^{r_2} \cdots \sum_{\alpha_{N-1}=1}^{r_{N-1}} \hat{c}_{i_1, \alpha_1}^{(1)} \hat{c}_{\alpha_1, i_2, \alpha_2}^{(2)} \cdots \hat{c}_{\alpha_{N-2}, i_{N-1}, \alpha_{N-1}}^{(N-1)} \hat{c}_{\alpha_{N-1}, i_N}^{(N)}.$$

Vale ressaltar que durante a demonstração do Teorema 1.2, os trechos em que utilizamos a decomposição SVD para representar os desdobramentos podem ser reescritos a partir de suas decomposições esqueletais. É importante levar em conta que a propriedade de ortogonalidade das matrizes foi utilizada durante a separação dos índices. No entanto, a demonstração pode ser mais geral, se utilizarmos a pseudoinversa das matrizes. Por exemplo, na expressão (1.15), se  $\mathbf{U}$  é um fator da representação esqueletal, não necessariamente possui colunas ortonormais e a expressão de sua pseudoinversa seria necessária para prosseguirmos com a demonstração. Por fim, desconhecemos a existência de informações sobre a qualidade da aproximação, como no Teorema 1.3, neste caso.

Operações matemáticas com tensores no formato TT produzem núcleos-TT com postos que não necessariamente são ótimos com respeito à precisão desejada. Por exemplo, o produto matriz-vetor amplia os postos-TT consideravelmente, aumentando o custo computacional de armazenamento. Neste sentido, o processo de recompressão, apresentado com detalhes em [39], é necessário na etapa de pós-processamento para reduzir os postos. Através da decomposição QR dos desdobramentos dos núcleos-TT, o processo de recompressão acontece a partir da SVD truncada sobre os novos núcleos-TT (veja [39, Algorithm 2]).

Também pode ser bastante útil representar tensores que já estão na forma PARAFAC para o formato TT. Por exemplo, seja  $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$  um tensor de posto  $R$ . Logo,

$$\mathcal{X} = \sum_{r=1}^R \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \cdots \circ \mathbf{a}_r^{(N)},$$

ou ainda,

$$x_{i_1 i_2, \dots, i_N} = \sum_{\alpha=1}^R a_{i_1, \alpha}^{(1)} a_{i_2, \alpha}^{(2)} \cdots a_{i_N, \alpha}^{(N)}.$$

Com o uso do delta de Kronecker, temos

$$x_{i_1, i_2, \dots, i_N} = \sum_{\alpha_1=1}^{r_1} \sum_{\alpha_2=1}^{r_2} \cdots \sum_{\alpha_{N-1}=1}^{r_{N-1}} a_{i_1, \alpha_1}^{(1)} \delta_{\alpha_1, \alpha_2} a_{i_2, \alpha_2}^{(2)} \delta_{\alpha_2, \alpha_3} \cdots \delta_{\alpha_{N-2}, \alpha_{N-1}} a_{i_N, \alpha_{N-1}}^{(N)}.$$

Assim, as fatias dos núcleos-TT,  $\mathcal{G}^{(k)} \in \mathbb{R}^{R \times I_k \times R}$ , são matrizes diagonais, isto é,  $\mathbf{G}_{:,i_k,:}^{(k)} = \text{diag}(a_{i_k,1}, \dots, a_{i_k,R}) \in \mathbb{R}^{R \times R}$  para  $k = 2, \dots, N - 1$ . Para  $k = 1$  temos  $\mathbf{G}^{(1)} = \mathbf{A}^{(1)} \in \mathbb{R}^{I_1 \times R}$  e para  $k = N$  temos  $\mathbf{G}^{(N)} = (\mathbf{A}^{(N)})^T \in \mathbb{R}^{R \times I_N}$ .

Com base nos experimentos de [39, Section 3.1], no Exemplo 1.13 exibimos a vantagem de se trabalhar com tensores cujas decomposições PARAFAC são conhecidas exatamente.

### Exemplo 1.13

Sejam  $a$  e  $b$  quaisquer vetores de  $\mathbb{R}^2$  e considere o tensor pertencente ao espaço gerado pelos  $N$  produtos externos abaixo:

$$\mathcal{A} = a \circ b \circ \cdots \circ b + b \circ a \circ b \circ \cdots \circ b + \dots + b \circ b \circ \cdots \circ a.$$

Como vimos anteriormente, podemos utilizar os fatores paralelos para construir a decomposição TT de  $\mathcal{A}$ . Os postos-TT, neste caso, serão iguais a  $\text{posto}(\mathcal{A}) = N$  e o processo de recompressão se torna necessário para reduzir a quantidade de memória necessária. Ao final, os postos-TT foram todos reduzidos a 2 e os tempos necessários para os cálculos são exibidos abaixo:

Tabela 1 – Tempo necessário para recompressão de  $\mathcal{A}$  no formato TT de acordo com  $N$ .

$N$	4	8	16	32	64	128
tempo (s)	0.00051	0.00083	0.00226	0.03515	0.22850	1.89331

## Operações Básicas no formato *Tensor Train*

### (i) Adição e multiplicação por escalar

Considere dois tensores  $\mathcal{A}$  e  $\mathcal{B}$  no formato TT:

$$a_{i_1, \dots, i_N} = \mathbf{G}_{:,i_1,:}^{(1)} \cdots \mathbf{G}_{:,i_N,:}^{(N)} \quad \text{e} \quad b_{i_1, \dots, i_N} = \mathbf{H}_{:,i_1,:}^{(1)} \cdots \mathbf{H}_{:,i_N,:}^{(N)}.$$

Os núcleos-TT da soma  $\mathcal{C} = \mathcal{A} + \mathcal{B}$  são definidos da seguinte forma:

$$\begin{aligned} \mathbf{C}_{:,i_k,:}^{(k)} &= \begin{bmatrix} \mathbf{G}_{:,i_k,:}^{(k)} & 0 \\ 0 & \mathbf{H}_{:,i_k,:}^{(k)} \end{bmatrix}, \quad k = 2, \dots, N - 1, \\ \mathbf{C}_{:,i_1,:}^{(1)} &= \begin{bmatrix} \mathbf{G}_{:,i_1,:}^{(1)} & \mathbf{H}_{:,i_1,:}^{(1)} \end{bmatrix} \quad \text{e} \quad \mathbf{C}_{:,i_N,:}^{(N)} = \begin{bmatrix} \mathbf{G}_{:,i_N,:}^{(N)} \\ \mathbf{H}_{:,i_N,:}^{(N)} \end{bmatrix}. \end{aligned}$$

A multiplicação por um escalar pode ser feita multiplicando-se um dos núcleos pelo escalar. Vale ressaltar que somando dois formatos TT, os postos se somam, tornando necessário o

uso da recompressão em processos em que muitas operações de adição são realizadas.

### (ii) Produto interno, norma e produto de Hadamard

Considere um tensor  $\mathcal{A}$  e vetores  $\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \dots, \mathbf{u}^N$ . Chamamos de contração multidimensional ao seguinte produto

$$W = \mathcal{A} \times_1 \mathbf{u}^{(1)} \times_2 \mathbf{u}^{(2)} \cdots \times_N \mathbf{u}^{(N)} = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} a_{i_1, i_2, \dots, i_N} u_{i_1}^{(1)} u_{i_2}^{(2)} \cdots u_{i_N}^{(N)}.$$

Suponha que  $\mathcal{A}$  esteja no formato TT,

$$a_{i_1, \dots, i_N} = \mathbf{G}_{:, i_1, :}^{(1)} \cdots \mathbf{G}_{:, i_N, :}^{(N)}$$

Então o produto pode ser reescrito como  $W = \mathbf{M}_1 \mathbf{M}_2 \cdots \mathbf{M}_N$ , onde

$$\mathbf{M}_k = \sum_{i_k=1}^{I_k} u_{i_k}^{(k)} \mathbf{G}_{:, i_k, :}^{(k)}$$

Note que  $\mathbf{M}_1$  e  $\mathbf{M}_N$  são vetores, de maneira que o produto resulta em um número real. Considerando um tensor de dimensões  $I \times I \times \cdots \times I$ , ordem  $N$  e postos-TT limitados por  $R$ , o número de operações realizadas é da ordem de  $NIR^2$ .

Agora considere o produto de Hadamard entre tensores  $N$ -dimensionais  $\mathcal{A}$  e  $\mathcal{B}$ . O resultado deste produto é um tensor  $\mathcal{C}$  cujas entradas são dadas por

$$c_{i_1, \dots, i_N} = a_{i_1, \dots, i_N} b_{i_1, \dots, i_N}.$$

Utilizando a representação TT, obtemos

$$c_{i_1, \dots, i_N} = \mathbf{G}_{:, i_1, :}^{(1)} \cdots \mathbf{G}_{:, i_N, :}^{(N)} \mathbf{H}_{:, i_1, :}^{(1)} \cdots \mathbf{H}_{:, i_N, :}^{(N)}$$

Da definição do produto de Kronecker (cf. Definição A.1) e de suas propriedades, segue

$$\begin{aligned} c_{i_1, \dots, i_N} &= a_{i_1, \dots, i_N} b_{i_1, \dots, i_N} \\ &= (\mathbf{G}_{:, i_1, :}^{(1)} \cdots \mathbf{G}_{:, i_N, :}^{(N)}) \otimes (\mathbf{H}_{:, i_1, :}^{(1)} \cdots \mathbf{H}_{:, i_N, :}^{(N)}) \\ &= (\mathbf{G}_{:, i_1, :}^{(1)} \otimes \mathbf{H}_{:, i_1, :}^{(1)}) \cdots (\mathbf{G}_{:, i_N, :}^{(N)} \otimes \mathbf{H}_{:, i_N, :}^{(N)}) \end{aligned}$$

e obtemos os núcleos-TT do produto de Hadamard. Deste modo, podemos calcular o produto interno entre dois tensores. Da definição,

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} a_{i_1, \dots, i_N} b_{i_1, \dots, i_N} = \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} c_{i_1, \dots, i_N} \cdot 1.$$

Portanto, o produto interno é o resultado de uma contração multidimensional com matrizes

$$\mathbf{M}_k = \mathbf{A}_{:, i_k, :}^{(i_k)} \otimes \mathbf{B}_{:, i_k, :}^{(i_k)}$$

e vetores  $\mathbf{u}^{(k)}$  com todas as entradas iguais a 1 para  $k = 1, \dots, N$ . A norma de Frobenius pode ser facilmente obtida fazendo  $\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$  (veja [39, Algorithm 4] para uma formalização do algoritmo para cálculo do produto interno). Neste caso, o número de operações realizadas é da ordem de  $NIR^3$ .

O Exemplo 1.14, inspirado em [38], encaminha uma aplicação da representação TT-Cross.

### Exemplo 1.14

Considere uma função  $f : \mathbb{R}^N \rightarrow \mathbb{R}$ , para a qual queremos calcular aproximadamente a integral

$$I(f) = \int_{[0,1]^N} f(x_1, \dots, x_N) dx_1 \cdots dx_N.$$

É necessário escolhermos um esquema de quadratura, isto é, precisamos determinar uma malha de pontos  $\{x_k\}_{k \in \mathbb{N}}$  para cada eixo e pesos associados a cada ponto. Lembramos do caso unidimensional: um esquema de  $n$  pontos satisfaz

$$\int_0^1 f(x) dx \approx \sum_{k=1}^n w_k f(x_k).$$

Aqui escolheremos o esquema de quadratura Gaussiana, ou seja, os pontos  $x_k$  da malha são as raízes do  $n$ -ésimo polinômio de Legendre  $P_n$  e

$$w_k = \frac{2}{(1 - x_k^2)(P'_n(x_k))^2}.$$

Deste modo, podemos generalizar para o caso multidimensional através do produto por contração. Neste sentido, a integral  $I(f)$  é aproximada por

$$I(f) \approx W = \sum_{k_1=1}^{K_1} \cdots \sum_{k_N=1}^{K_N} f(x_{k_1}, \dots, x_{k_N}) w_{k_1}^{(1)} \cdots w_{k_N}^{(N)},$$

onde  $K_1, \dots, K_N$  são as quantidades de pontos de quadratura em cada eixo e os pesos são os vetores  $\mathbf{w}^{(k)} \in \mathbb{R}^{K_k}$ . Os pontos sobre os quais a função  $f$  é avaliada podem ser armazenados em um tensor  $\mathcal{A}$  de dimensões  $K_1 \times \cdots \times K_N$ .

Assim, a integral é aproximada por um produto de contração. Se a quantidade de pontos em cada eixo for igual a  $K$ , então a quantidade de operações realizadas é da ordem de  $K^N$ . A maldição da dimensionalidade volta a aparecer e pode ser necessário utilizar da representação TT para realizar o produto. Considere especificamente a função

$$f(x_1, \dots, x_N) = \operatorname{sen}(x_1 + \dots + x_N).$$

A solução analítica, de acordo com [38], é

$$I(f) = \operatorname{Im} \left\{ \left( \frac{e^i - 1}{i} \right)^N \right\},$$

onde  $i = \sqrt{-1}$ . Vamos tomar  $N = [2, 5, 10, 20]$  e apenas 2 pontos de quadratura Gaussiana em cada direção. Além disso, a precisão da decomposição TT será de  $\varepsilon = 10^{-8}$ . Vale ressaltar que o manuseio dos tensores e a decomposição TT foram feitos através da biblioteca `TensorToolbox.jl` da linguagem Julia<sup>1</sup>. Os resultados são exibidos nas Tabelas 2 e 3.

Tabela 2 – Resultados da integração utilizando o produto por contração.

N	erro absoluto	tempo (ms)
2	$3.7 \cdot 10^{-4}$	0.04
5	$5.8 \cdot 10^{-4}$	0.12
10	$1.5 \cdot 10^{-3}$	0.13
20	$1.1 \cdot 10^{-3}$	45.51

Tabela 3 – Resultados da integração utilizando a representação TT.

N	erro absoluto	tempo (ms)
2	$3.7 \cdot 10^{-4}$	0.05
5	$5.8 \cdot 10^{-4}$	0.28
10	$1.5 \cdot 10^{-3}$	0.33
20	$1.1 \cdot 10^{-3}$	0.78

Observando as tabelas acima, podemos verificar que os erros absolutos são idênticos. Em [51, Theorem 1], se o erro cometido pela integração com produto por contração é  $\varepsilon_1$  e a precisão da fatoração TT é  $\varepsilon_2$ , então o erro cometido pela integração com a decomposição TT é  $N\varepsilon_1 + \varepsilon_2$ . Desse modo, como a precisão escolhida foi pequena, o erro absoluto das tabelas é praticamente o mesmo.

O tempo exibido no teste com a representação TT leva em conta o tempo para o cálculo desta decomposição. É notável que para dimensões pequenas como  $N = 2, 5$  e  $10$ , os tempos de execução foram pequenos e parecidos. Para  $N = 20$  vale lembrar que se prosseguirmos de maneira tradicional, são calculados  $2^{20}$  produtos. Neste sentido, a representação TT reduziu o tempo de execução expressivamente.

<sup>1</sup> <https://julialang.org/>

### (iii) Produto matriz-vetor

Dado um vetor  $\mathbf{v} \in \mathbb{R}^M$ , podemos representá-lo com um tensor cujo  $k$ -ésimo modo tem tamanho  $I_k$ , onde  $I_1 I_2 \cdots I_N = M$ . A decomposição TT deste tensor é a representação TT de  $\mathbf{v}$  cujos núcleos denotamos por  $\mathbf{v}^{(k)} \in \mathbb{R}^{p_{k-1} \times I_k \times p_k}$ . No caso de uma matriz quadrada  $\mathbf{A} \in \mathbb{R}^{M \times M}$ , seus elementos  $a_{\overline{i_1 \dots i_N}, \overline{j_1 \dots j_N}}$  são indicados por  $2N$  índices de forma que  $(i_1, \dots, i_N)$  indicam as linhas e  $(j_1, \dots, j_N)$  as colunas. De acordo com [39], a representação TT de  $\mathbf{A}$  possui núcleos-TT expressos por  $\mathcal{G}^{(k)} \in \mathbb{R}^{r_{k-1} \times I_k \times I_k \times r_k}$ , de forma que

$$a_{\overline{i_1 \dots i_N}, \overline{j_1 \dots j_N}} = \mathbf{G}_{:, i_1, j_1, :}^{(1)} \cdots \mathbf{G}_{:, i_N, j_N, :}^{(N)}$$

Seja  $\mathbf{B} = \mathbf{A}\mathbf{v}$ . Das propriedades do produto de Kronecker, sua representação TT é

$$\begin{aligned} b_{i_1, i_2, \dots, i_N} &= \sum_{j_1=1}^{I_1} \sum_{j_2=1}^{I_2} \cdots \sum_{j_N=1}^{I_N} a_{\overline{i_1 i_2 \dots i_N}, \overline{j_1 j_2 \dots j_N}} v_{\overline{j_1 j_2 \dots j_N}} \\ &= \sum_{j_1=1}^{I_1} \sum_{j_2=1}^{I_2} \cdots \sum_{j_N=1}^{I_N} \mathbf{G}_{:, i_1, j_1, :}^{(1)} \cdots \mathbf{G}_{:, i_N, j_N, :}^{(N)} \mathbf{v}_{:, j_1, :}^{(1)} \cdots \mathbf{v}_{:, j_N, :}^{(N)} \\ &= \sum_{j_1=1}^{I_1} \sum_{j_2=1}^{I_2} \cdots \sum_{j_N=1}^{I_N} (\mathbf{G}_{:, i_1, j_1, :}^{(1)} \cdots \mathbf{G}_{:, i_N, j_N, :}^{(N)}) \otimes (\mathbf{v}_{:, j_1, :}^{(1)} \cdots \mathbf{v}_{:, j_N, :}^{(N)}) \\ &= \sum_{j_1=1}^{I_1} \sum_{j_2=1}^{I_2} \cdots \sum_{j_N=1}^{I_N} (\mathbf{G}_{:, i_1, j_1, :}^{(1)} \otimes \mathbf{v}_{:, j_1, :}^{(1)}) \cdots (\mathbf{G}_{:, i_N, j_N, :}^{(N)} \otimes \mathbf{v}_{:, j_N, :}^{(N)}). \end{aligned}$$

Denote os núcleos-TT de  $\mathbf{B}$  por  $\mathcal{H}^{(k)} \in \mathbb{R}^{p_{k-1} r_{k-1} \times I_k \times I_k \times p_k r_k}$ , para  $k = 0, \dots, N$ . Matricialmente temos

$$\mathbf{H}_{:, i_k, :}^{(k)} = \sum_{j_k=1}^{I_k} \mathbf{G}_{:, i_k, j_k, :}^{(k)} \otimes \mathbf{v}_{:, j_k, :}^{(k)}$$

Os detalhes e a formalização do algoritmo são dados em [39, Section 4.3]. Os postos-TT de  $\mathbf{B}$  são o produto dos postos-TT de  $\mathbf{A}$  e  $\mathbf{v}$ , o que torna o uso do processo de recompressão mais que necessário para evitar a maldição da dimensionalidade.

Para concluirmos, demonstraremos uma propriedade que será importante em seções futuras. Dado  $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$  com núcleos-TT  $\mathcal{G}^{(k)} \in \mathbb{R}^{r_{k-1} \times I_k \times r_k}$ , defina

$$\mathcal{G}^{<n} \in \mathbb{R}^{I_1 \times \cdots \times I_{k-1} \times r_{k-1}} \quad \text{e} \quad \mathcal{G}^{>n} \in \mathbb{R}^{r_k \times I_{k+1} \times \cdots \times I_N}$$

onde

$$g_{i_1, \dots, i_{k-1}, \alpha_{k-1}}^{<k} := \sum_{\alpha_1=1}^{r_1} \sum_{\alpha_2=1}^{r_2} \cdots \sum_{\alpha_{k-2}=1}^{r_{k-2}} g_{1, i_1, \alpha_1}^{(1)} g_{\alpha_1, i_2, \alpha_2}^{(2)} \cdots g_{\alpha_{k-2}, i_{k-1}, \alpha_{k-1}}^{(k-1)}$$

e

$$g_{\alpha_k, i_{k+1}, \dots, i_N}^{>k} := \sum_{\alpha_{k+1}=1}^{r_{k+1}} \sum_{\alpha_{k+2}=1}^{r_{k+2}} \cdots \sum_{\alpha_{N-1}=1}^{r_{N-1}} g_{\alpha_k, i_{k+1}, \alpha_{k+1}}^{(k+1)} g_{\alpha_{k+1}, i_{k+2}, \alpha_{k+2}}^{(k+2)} \cdots g_{\alpha_{N-1}, i_N, 1}^{(N)}$$

Vamos mostrar que, para  $k = 1, \dots, N$ , vale [33, Table 4]

$$\mathbf{X}_{(k)} = \mathbf{G}_{(2)}^{(k)} (\mathbf{G}_{(1)}^{<k} \otimes \mathbf{G}_{(k)}^{>k}). \quad (1.23)$$

Pela decomposição *Tensor Train*, temos

$$\begin{aligned}
x_{i_k, \overline{i_1 \cdots i_{k-1} i_{k+1} \cdots i_N}} &= x_{i_1, i_2, \dots, i_N} \\
&= \sum_{\alpha_{k-1}=1}^{r_{k-1}} \sum_{\alpha_k=1}^{r_k} g_{i_1, \dots, i_k, \alpha_{k-1}}^{<k} g_{\alpha_{k-1}, i_k, \alpha_k}^{(k)} g_{\alpha_k, i_{k+1}, \dots, i_N}^{>k} \\
&= \sum_{\alpha_{k-1}=1}^{r_{k-1}} \sum_{\alpha_k=1}^{r_k} g_{\alpha_{k-1}, \overline{i_1 i_2 \cdots i_{k-1}}}^{<k} g_{i_k, \alpha_{k-1} \alpha_k}^{(k)} g_{\alpha_k, \overline{i_{k+1} \cdots i_N}}^{>k} \\
&= \sum_{\alpha_{k-1}=1}^{r_{k-1}} \sum_{\alpha_k=1}^{r_k} \left( g_{\alpha_{k-1}, \overline{i_1 i_2 \cdots i_{k-1}}}^{<k} \cdot g_{\alpha_k, \overline{i_{k+1} \cdots i_N}}^{>k} \right) g_{i_k, \overline{\alpha_{k-1} \alpha_k}}^{(k)}.
\end{aligned}$$

Observe que o resultado da operação entre parênteses é um elemento de uma matriz  $\mathbf{M}$  de dimensões  $r_{k-1} r_k \times I_1 \cdots I_{k-1} I_{k+1} \cdots I_N$ . Note que  $\mathbf{M} = \mathbf{G}_{(1)}^{>k} \otimes \mathbf{G}_{(k)}^{<k}$ . Concatenando os índices  $\alpha_{k-1}$  e  $\alpha_k$  em um longo índice  $\beta$ , temos

$$x_{i_k, \overline{i_1 \cdots i_{k-1} i_{k+1} \cdots i_N}} = \sum_{\beta=1}^{r_{k-1} r_k} g_{i_k, \beta}^{(k)} m_{\beta, \overline{i_1 i_2 \cdots i_{k-1} i_{k+1} \cdots i_N}}$$

que é uma multiplicação de matrizes, mostrando a validade da expressão (1.23).

## Ortogonalização dos núcleos-TT

A ortogonalização de núcleos-TT de um tensor é um procedimento essencial em métodos que envolvam a representação no formato TT [8, 46], devido à estabilidade proporcionada aos algoritmos numéricos.

Dado um tensor  $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ , definimos os desdobramentos à esquerda e à direita de um núcleo-TT  $\mathcal{G}^{(k)}$ , respectivamente como

$$\mathbf{G}_L^{(k)} := \mathbf{G}_{<2>}^{(k)} \in \mathbb{R}^{r_{k-1} I_k \times r_k} \quad \text{e} \quad \mathbf{G}_R^{(k)} := \mathbf{G}_{<1>}^{(k)} \in \mathbb{R}^{r_{k-1} \times I_k r_k}.$$

**Definição 1.12** Seja  $\mathcal{X}$  um tensor de ordem  $N$  com núcleos-TT  $\mathcal{G}^{(k)} \in \mathbb{R}^{r_{k-1} \times I_k \times r_k}$ ,  $k = 1, \dots, N$ . Considere também  $n \in \{1, \dots, N\}$ . Dizemos que  $\mathcal{X}$  é  $n$ -ortogonal se

$$\begin{aligned}
\mathbf{G}_L^{(k)^T} \mathbf{G}_L^{(k)} &= \mathbf{I}_{r_j}, \quad j = 1, \dots, n-1 \quad \text{e} \\
\mathbf{G}_R^{(k)} \mathbf{G}_R^{(k)^T} &= \mathbf{I}_{r_{j-1}}, \quad j = n+1, \dots, N.
\end{aligned}$$

Além disso,  $\mathcal{X}$  é ortogonal à esquerda se  $n = N$  e ortogonal à direita se  $n = 1$ .

Para realizar a ortogonalização dos núcleos-TT da esquerda para a direita, calculamos a decomposição QR do primeiro núcleo  $\mathbf{G}_L^{(1)} = \mathbf{Q}^{(1)} \mathbf{R}^{(1)}$ . Este será substituído pelo fator ortogonal e teremos

$$\mathcal{G}^{(2)} \leftarrow \mathcal{G}^{(2)} \times_1 \mathbf{R}^{(1)}.$$

Substituímos o segundo núcleo pelo fator ortogonal da fatoração QR do seu desdobramento à esquerda e seguimos dessa forma para os núcleos restantes. Por fim, as entradas de  $\mathcal{G}^{(N)}$  serão normalizadas. Em [8, Algorithm 14], é exibido um algoritmo eficiente para a ortogonalização dos núcleos. É importante ressaltar que este procedimento modifica a representação interna do tensor, que continua o mesmo. O Exemplo 1.15 ilustra essas ideias para um tensor  $3 \times 3 \times 3$ .

### Exemplo 1.15

Considere  $\mathcal{A} \in \mathbb{R}^{3 \times 3 \times 3}$  cujas entradas são dadas por

$$a_{i_1, i_2, i_3} = i_1 + i_2 + i_3$$

com  $i_1, i_2, i_3 \in \{1, 2, 3\}$ . Os núcleos-TT deste tensor são dados por

$$\begin{aligned}\mathcal{G}^{(1)} &= \left\{ \begin{pmatrix} 1 & 1 \end{pmatrix}, \begin{pmatrix} 2 & 1 \end{pmatrix}, \begin{pmatrix} 3 & 1 \end{pmatrix} \right\} \\ \mathcal{G}^{(2)} &= \left\{ \begin{pmatrix} 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \end{pmatrix} \right\} \\ \mathcal{G}^{(4)} &= \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 3 \end{pmatrix} \right\}.\end{aligned}$$

Efetuamos o processo de ortogonalização dos primeiros dois núcleos. Considere  $\mathbf{G}_L^{(1)}$  e sua fatoração QR dada por

$$\mathbf{G}_L^{(1)} = \begin{pmatrix} -0.2673 & 0.8729 \\ -0.5345 & 0.2182 \\ -0.8018 & -0.4364 \end{pmatrix} \begin{pmatrix} -3.7417 & -1.6036 \\ 0 & 0.6547 \end{pmatrix} = \mathbf{Q}^{(1)} \mathbf{R}^{(1)}.$$

Realizamos a atribuição ao segundo núcleo  $\mathcal{G}^{(2)} \leftarrow \mathcal{G}^{(2)} \times_1 \mathbf{R}^{(1)}$  e tomamos a sua fatoração QR:

$$\mathbf{G}_L^{(2)} = \begin{pmatrix} 0.4280 & 0.4448 \\ 0.0524 & -0.6052 \\ -0.5563 & 0.0 \\ 0.1048 & -0.4237 \\ -0.6847 & -0.4448 \\ 0.1572 & -0.2421 \end{pmatrix} \begin{pmatrix} 12.4900 & 2.8823 \\ 0.0 & -0.8321 \end{pmatrix} = \mathbf{Q}^{(2)} \mathbf{R}^{(2)}.$$

A atribuição ao segundo núcleo é  $\mathcal{G}^{(3)} \leftarrow \mathcal{G}^{(3)} \times_1 \mathbf{R}^{(2)}$ . Reestruturando  $\mathbf{Q}^{(1)}$  e  $\mathbf{Q}^{(2)}$  com as dimensões dos núcleos  $\mathcal{G}^{(1)}$  e  $\mathcal{G}^{(2)}$  respectivamente, obtemos a ortogonalidade à esquerda de acordo com a Definição 1.12.

# Capítulo 2

## Completamento Tensorial

Em diversos problemas práticos, existe a necessidade de se recuperar uma matriz a partir de uma amostra de suas entradas. Por exemplo, suponha que um formulário seja preenchido por diversas pessoas. As informações são organizadas em uma matriz onde as linhas são indexadas pelos indivíduos e as colunas pelos itens. É possível que várias questões sejam deixadas sem respostas, de modo que é desejável estimar as entradas faltantes. O desafio que temos em mãos é recuperar a matriz com todas as respostas a partir de uma amostra do que já foi respondido. Um dos exemplos mais conhecidos deste problema é o chamado *Netflix Prize* [23], uma competição aberta para encontrar o melhor algoritmo que fosse capaz de prever novas avaliações de assinantes, baseando-se apenas em avaliações conhecidas. Desse modo, é possível organizar este objetivo como um problema de completamento. Para mais aplicações, veja [34] e [3].

Usualmente em problemas reais, a matriz desejada pertence a um subespaço de posto pequeno. A maior parte das informações relevantes são representadas por um pequeno número de valores singulares o qual, usualmente, é menor que as dimensões da matriz. Neste sentido, iremos nos basear no trabalho de Candès e Recht [5], onde o problema de completamento é abordado pelo seguinte problema de otimização

$$\begin{cases} \min_{\mathbf{X}} \text{posto}(\mathbf{X}) \\ \text{s.a } \mathcal{P}_{\Omega}(\mathbf{X}) = \mathcal{P}_{\Omega}(\mathbf{T}) \end{cases}, \quad (2.1)$$

onde  $\mathbf{T}$  é a matriz original,  $\mathcal{P}_{\Omega}$  é o operador de projeção ortogonal sobre o conjunto de índices das entradas conhecidas, denotado por  $\Omega$ . Entretanto, devido à sua natureza combinatória, este problema é do tipo NP-difícil [5], sendo necessário o uso de algoritmos que aproximam o posto da matriz. Por este caminho, apresentamos nesta seção alguns dos algoritmos utilizados no completamento matricial. Desta maneira, seremos capazes de apresentar a extensão do problema para o caso tensorial de forma natural. Todavia, como mostrado em seções anteriores, a noção de posto para tensores não é única e, consequentemente, existem diversas formulações de algoritmos que buscam minimizar a noção de posto adotada.

A partir do problema matricial, vamos estudar o problema de completamento tensorial frente às noções de posto que vimos anteriormente, aprofundando naqueles que apresentam relação com a decomposição *Tensor Train* e o posto-TT, o qual, devido à sua alta versatilidade, tem mostrado ótimos resultados (veja [2] e [55]). As formulações serão testadas a partir de um problema de completamento de imagem que, originalmente, são tensores de terceira ordem. Neste caso, não será possível extrair toda as informações do posto-TT e uma técnica de mapeamento, chamada *Ket Augmentation*, associa as entradas da imagem para um tensor de ordem mais alta.

Na abordagem feita por Candès e Recht, conforme veremos a seguir, o completamento é feito resolvendo um problema de otimização cuja função objetivo é convexa e não diferenciável. Neste sentido, a fim de explorarmos as boas propriedades de abordagens para formulações suaves que empregam o vetor gradiente, também apresentamos métodos que fazem seu uso na resolução de um problema diferenciável.

## 2.1 Completamento de Matrizes

Em um primeiro momento, exibimos alguns dos principais métodos de completamento matricial os quais serão úteis para o entendimento do completamento dos tensores de mais alta ordem. Assim, considere  $\mathbf{T} \in \mathbb{R}^{m \times n}$  e um subconjunto de índices  $\Omega \subset \{1, \dots, m\} \times \{1, \dots, n\}$ . O projetor ortogonal  $\mathcal{P}_\Omega$  é tal que

$$\mathcal{P}_\Omega(\mathbf{X})_{i,j} = \begin{cases} x_{i,j}, & (i, j) \in \Omega \\ 0, & (i, j) \notin \Omega \end{cases}.$$

Como mencionado anteriormente, o problema (2.1) deve ser abordado de outra maneira devido à sua essência fortemente combinatorial. Em [17, Theorem 1], Fazel mostrou que a *norma nuclear*

$$\|\mathbf{X}\|_* := \sum_{i=1}^{\text{posto}(\mathbf{X})} \sigma_i(\mathbf{X})$$

é o envelope convexo da função posto. Ou seja, o problema (2.1) pode ser relaxado por meio de uma função convexa da seguinte maneira

$$\begin{cases} \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \|\mathbf{X}\|_* \\ \text{s.a } \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{T}) \end{cases}. \quad (2.2)$$

De fato, por ser uma norma (este resultado é provado nos slides de Andersen Ang<sup>1</sup>), a soma dos valores singulares de  $\mathbf{X}$  é uma função convexa. Neste sentido, o mínimo sobre o envelope convexo serve de limitante inferior para o mínimo do problema original. Candès e Tao [6] exibem diversas maneiras de se amostrar a matriz  $\mathbf{T}$ , podendo ser recuperada

<sup>1</sup> <https://angms.science/doc/LA/KyFanNorm.pdf>

perfeitamente com alta probabilidade desde que as entradas conhecidas estejam espalhadas uniformemente.

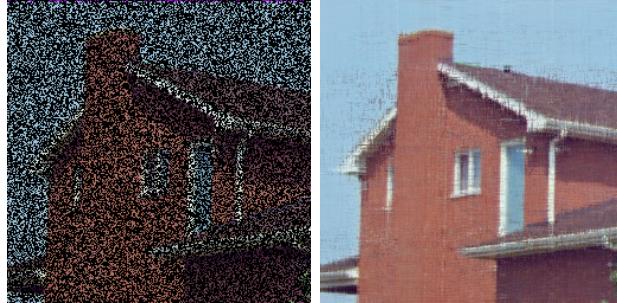


Figura 11 – Ilustração para a reconstrução de uma imagem por meio de um problema de minimização do posto.

Dada uma matriz  $\mathbf{X} \in \mathbb{R}^{m \times n}$  de posto  $r$ , considere a sua decomposição SVD

$$\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T, \quad \mathbf{S} = \text{diag}(\sigma_1, \dots, \sigma_r).$$

Os métodos de otimização que veremos, baseiam-se no seguinte operador de *shrinkage*:

$$\mathcal{D}_\tau(\mathbf{X}) := \mathbf{U}\mathcal{D}_\tau(\mathbf{S})\mathbf{V}^T, \quad \mathcal{D}_\tau(\mathbf{S}) = \text{diag}(\max(\sigma_1 - \tau, 0), \dots, \max(\sigma_r - \tau, 0)),$$

onde  $\tau \geq 0$ . Em palavras, este operador é semelhante ao operador de *soft-thresholding* para os valores singulares de  $\mathbf{X}$ . Caso estes estejam abaixo do parâmetro  $\tau$ , serão levados em zero e o posto de  $\mathcal{D}_\tau(\mathbf{X})$  será menor que  $r$ . Mostraremos que o operador de *shrinkage* é o operador proximal associado à norma nuclear (veja [1, Chapter 6] para mais informações sobre o operador proximal). Antes será necessário demonstrar um resultado auxiliar.

**Lema 2.1** *Seja  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  estritamente convexa. Então  $f$  não possui mais de um mínimo.*

Prova: Suponha, por contradição, que  $f$  possua um mínimo local em  $\mathbf{x}_1$  e outro em  $\mathbf{x}_2$ , com  $\mathbf{x}_1 \neq \mathbf{x}_2$ . Sem perda de generalidade, considere que

$$f(\mathbf{x}_1) \leq f(\mathbf{x}_2).$$

Por definição,

$$f(t\mathbf{x}_1 + (1-t)\mathbf{x}_2) < tf(\mathbf{x}_1) + (1-t)f(\mathbf{x}_2), \quad 0 < t < 1.$$

Como  $t > 0$ , vale que  $tf(\mathbf{x}_1) \leq tf(\mathbf{x}_2)$ . Desse modo

$$\begin{aligned} f(t\mathbf{x}_1 + (1-t)\mathbf{x}_2) &< tf(\mathbf{x}_1) + (1-t)f(\mathbf{x}_2) \\ &\leq tf(\mathbf{x}_2) + (1-t)f(\mathbf{x}_2) \\ &\leq f(\mathbf{x}_2). \end{aligned}$$

Portanto, tomando  $t$  suficientemente pequeno, chegamos em um absurdo, pois  $\mathbf{x}_2$  é minimizador local. Logo  $\mathbf{x}_1 = \mathbf{x}_2$ .  $\square$

Agora podemos demonstrar o seguinte teorema:

**Teorema 2.1** *Sejam  $\tau \geq 0$  e  $\mathbf{Y} \in \mathbb{R}^{m \times n}$ . O operador de shrinkage  $\mathcal{D}_\tau$  satisfaz*

$$\mathcal{D}_\tau(\mathbf{Y}) = \arg \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \left\{ \tau \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X} - \mathbf{Y}\|_F^2 \right\}.$$

Prova: Defina  $h : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$  dada por  $h(\mathbf{X}) = \tau \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X} - \mathbf{Y}\|_F^2$ . Observe que  $h$  é estritamente convexa e, pelo Lema 2.1, possui um único minimizador. Mostraremos que este único minimizador é  $\mathcal{D}_\tau(\mathbf{Y})$ . Vale ressaltar que a norma nuclear não é diferenciável, de forma que é necessário recorrermos ao conceito de subdiferencial. De acordo com [1, Theorem 3.63], segue que

$$\mathbf{x}^* \in \arg \min_{\mathbf{x}} h(\mathbf{x}) \Leftrightarrow \mathbf{0} \in \partial h(\mathbf{x}^*).$$

Aplicando as regras de cálculo subdiferencial:

$$\mathbf{0} \in \partial h(\mathbf{x}^*) \Leftrightarrow \mathbf{0} \in \mathbf{X}^* - \mathbf{Y} + \tau \partial \|\mathbf{X}^*\|_*.$$

Em [52] é demonstrado que o subdiferencial da norma nuclear é o conjunto dado por

$$\partial \|\mathbf{X}\|_* = \{\mathbf{U}\mathbf{V}^T + \mathbf{W} \mid \mathbf{W} \in \mathbb{R}^{m \times n}, \mathbf{U}^T \mathbf{W} = \mathbf{0}, \mathbf{WV} = \mathbf{0}, \|\mathbf{W}\|_2 \leq 1\},$$

em que  $\mathbf{U}$  e  $\mathbf{V}$  são, respectivamente, as matrizes cujas colunas contêm os valores singulares à esquerda e à direita da matriz  $\mathbf{X}$ . Assim, considere a decomposição SVD de  $\mathbf{Y}$  como  $\mathbf{Y} = \hat{\mathbf{U}}\hat{\mathbf{S}}\hat{\mathbf{V}}^T + \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$  onde a primeira parcela está associada aos valores singulares maiores que  $\tau$  e a segunda associada aos valores singulares menores ou iguais a  $\tau$ . Note que

$$\mathcal{D}_\tau(\mathbf{Y}) = \hat{\mathbf{U}}(\hat{\mathbf{S}} - \tau \mathbf{I})\hat{\mathbf{V}}^T.$$

Então

$$\begin{aligned} \mathbf{Y} - \mathcal{D}_\tau(\mathbf{Y}) &= \hat{\mathbf{U}}\hat{\mathbf{S}}\hat{\mathbf{V}}^T + \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \hat{\mathbf{U}}(\hat{\mathbf{S}} - \tau \mathbf{I})\hat{\mathbf{V}}^T \\ &= \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T + \tau \hat{\mathbf{U}}\hat{\mathbf{V}}^T. \end{aligned}$$

Desse modo,

$$\mathbf{Y} - \mathcal{D}_\tau(\mathbf{Y}) = \tau(\hat{\mathbf{U}}\hat{\mathbf{V}}^T + \mathbf{W}) \quad \text{e} \quad \mathbf{W} = \frac{1}{\tau} \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T.$$

Observe que  $\hat{\mathbf{U}}^T \mathbf{W} = \mathbf{0}$  e  $\mathbf{W}\hat{\mathbf{V}} = \mathbf{0}$ . Ainda, como os valores singulares de  $\mathbf{W}$  são limitados superiormente por  $\tau$ , segue que  $\|\mathbf{W}\|_2 \leq 1$ . Podemos concluir que  $\mathbf{Y} - \mathcal{D}_\tau(\mathbf{X}^*) \in \tau \partial \|\mathbf{X}^*\|_*$ , como desejávamos.  $\square$

### Método **Singular Value Thresholding (SVT)**

Considere o seguinte problema de otimização:

$$\begin{cases} \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \tau \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X}\|_F^2 \\ \text{s.a } \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{T}) \end{cases}. \quad (2.3)$$

Note que para  $\tau$  suficientemente grande, recuperamos o problema (2.2). Considerando a variável dual  $\mathbf{Y} \in \mathbb{R}^{m \times n}$ , o Lagrangiano é definido por

$$\mathcal{L}(\mathbf{X}, \mathbf{Y}) = \tau \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X}\|_F^2 + \langle \mathbf{Y}, \mathcal{P}_\Omega(\mathbf{X}) - \mathcal{P}_\Omega(\mathbf{T}) \rangle.$$

A resolução do problema é feita minimizando o Lagrangiano. Neste caso, em um primeiro momento, é feita a minimização para a variável primal e, em seguida, maximização para a dual, prosseguindo iterativamente. Este processo segue da validade da dualidade forte e pelo fato do par ótimo  $(\mathbf{X}^*, \mathbf{Y}^*)$  ser um ponto de sela do Lagrangiano. Assim, prosseguindo para  $\mathbf{X}$  e denotando o traço de  $\mathbf{M}$  por  $\text{tr}(\mathbf{M})$ , temos

$$\begin{aligned} \mathbf{X} &= \arg \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \tau \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X}\|_F^2 + \langle \mathbf{Y}, \mathcal{P}_\Omega(\mathbf{X} - \mathbf{T}) \rangle \\ &= \arg \min_{\mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{T})} \tau \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X}\|_F^2 + \langle \mathbf{Y}, \mathbf{X} \rangle \quad [\text{tr}(\mathbf{Y}^T \mathcal{P}_\Omega(\mathbf{X})) = \text{tr}(\mathcal{P}_\Omega(\mathbf{Y})^T \mathbf{X})] \\ &= \arg \min_{\mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{T})} \tau \|\mathbf{X}\|_* + \frac{1}{2} \|\mathbf{X} - \mathbf{Y}\|_F^2 \quad [\text{Completando quadrados}] \\ &= \mathcal{D}_\tau(\mathcal{P}_\Omega(\mathbf{Y})). \quad [\text{Teorema 2.1}] \end{aligned}$$

Note que, como o projetor é um operador linear, a diferença das projeções de  $\mathbf{X}$  e  $\mathbf{T}$  é igual à projeção da diferença  $\mathbf{X} - \mathbf{T}$ . Para o caso da variável dual, temos

$$\mathbf{Y} = \arg \max_{\mathbf{Y} \in \mathbb{R}^{m \times n}} \mathcal{L}(\mathbf{X}, \mathbf{Y}) = \mathcal{P}_\Omega(\mathbf{T} - \mathbf{X}).$$

Cabe ressaltar que pelas características do projetor,  $\mathbf{Y} = \mathcal{P}_\Omega(\mathbf{Y})$ . Desta maneira, fixado  $\tau > 0$ , uma sequência de tamanhos de passo  $\{t_k\}_{k \in \mathbb{N}}$  positivos e dado  $\mathbf{Y}^0$ , o método é definido por

$$\begin{cases} \mathbf{X}^{k+1} = \mathcal{D}_\tau(\mathbf{Y}^k) \\ \mathbf{Y}^{k+1} = \mathbf{Y}^k + t_k \mathcal{P}_\Omega(\mathbf{T} - \mathbf{X}^{k+1}) \end{cases}$$

para todo  $k \geq 0$ . A função objetivo do problema é estritamente convexa e as restrições são lineares e convexas. Deste modo, o método é do tipo gradiente proximal dual e sua convergência é assegurada [1, Chapter 12]. Para mais detalhes sobre a convergência e implementação, veja [4].

## Regularização da Norma Nuclear Truncada (TNNR)

Quando consideramos a minimização da norma nuclear, nos deparamos com o seguinte contratempo: todos os valores singulares são minimizados simultaneamente, possuindo diferentes contribuições. Como sabemos, todos os valores singulares não-nulos possuem igual participação na definição do posto. Pode-se estudar o caso em que buscamos otimizar uma quantidade menor de valores singulares. Em [25], o método de regularização da norma nuclear truncada (TNNR) é proposto. Supondo que a matriz do problema seja de posto  $r$ , a abordagem é feita considerando os  $p$ -ésimos valores singulares e minimizando os  $r - p$  restantes. A solução é obtida com auxílio do método de direções alternadas e multiplicadores (ADMM) [1, Chapter 15].

Seja  $\mathbf{X} \in \mathbb{R}^{m \times n}$ . A norma nuclear truncada é definida (cf. [25]) como

$$\|\mathbf{X}\|_p := \sum_{i=p+1}^{\min\{m,n\}} \sigma_i(\mathbf{X}).$$

A formulação do problema de completamento é dada por

$$\begin{cases} \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \|\mathbf{X}\|_p \\ \text{s.a } \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{T}) \end{cases}. \quad (2.4)$$

Para qualquer  $r \leq \min\{m, n\}$  e quaisquer matrizes  $\mathbf{A} \in \mathbb{R}^{r \times m}$  e  $\mathbf{B} \in \mathbb{R}^{r \times n}$ , com  $\mathbf{A}\mathbf{A}^T = I_{r \times r} = \mathbf{B}\mathbf{B}^T$ , vale que [25, Theorem 3.1]

$$\text{tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) \leq \sum_{i=1}^r \sigma_i(\mathbf{X}). \quad (2.5)$$

Agora considere a decomposição SVD de  $\mathbf{X}$ . As colunas de  $\mathbf{U}$  são formadas pelos vetores  $\mathbf{u}_1, \dots, \mathbf{u}_m$  e as colunas  $\mathbf{V}$ , pelos vetores  $\mathbf{v}_1, \dots, \mathbf{v}_n$ . Tome

$$\mathbf{A} = (\mathbf{u}_1, \dots, \mathbf{u}_r)^T \quad \text{e} \quad \mathbf{B} = (\mathbf{v}_1, \dots, \mathbf{v}_r)^T.$$

Então

$$\begin{aligned} \text{tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) &= \text{tr}\left((\mathbf{u}_1, \dots, \mathbf{u}_r)^T \mathbf{U} \mathbf{S} \mathbf{V}^T (\mathbf{v}_1, \dots, \mathbf{v}_r)\right) \\ &= \text{tr}\left((\mathbf{I}_{r \times r} \ \mathbf{0}_{r \times (n-r)}) \mathbf{S} \begin{pmatrix} \mathbf{I}_{r \times r} \\ \mathbf{0}_{(n-r) \times r} \end{pmatrix}\right) \\ &= \text{tr}(\text{diag}(\sigma_1, \dots, \sigma_r)) \\ &= \sum_{i=1}^r \sigma_i(\mathbf{X}). \end{aligned}$$

Combinando a igualdade acima com (2.5), temos

$$\max_{\mathbf{A}\mathbf{A}^T = I, \mathbf{B}\mathbf{B}^T = I} \text{tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) = \sum_{i=1}^r \sigma_i(\mathbf{X})$$

e consequentemente

$$\|\mathbf{X}\|_* - \max_{\mathbf{A}\mathbf{A}^T = I, \mathbf{B}\mathbf{B}^T = I} \text{tr}(\mathbf{A}\mathbf{X}\mathbf{B}^T) = \|\mathbf{X}\|_p.$$

Como tiraremos proveito do método ADMM, o problema (2.4) pode ser reescrito da seguinte forma

$$\begin{cases} \min_{\mathbf{X}, \mathbf{W} \in \mathbb{R}^{m \times n}} \|\mathbf{X}\|_* - \text{tr}(\mathbf{A}\mathbf{W}\mathbf{B}^T) \\ \text{s.a } \mathbf{X} = \mathbf{W}, \mathcal{P}_\Omega(\mathbf{W}) = \mathcal{P}_\Omega(\mathbf{T}) \end{cases}. \quad (2.6)$$

Considerando a variável dual  $\mathbf{Y} \in \mathbb{R}^{m \times n}$ , o Lagrangiano aumentado deste problema é

$$\mathcal{L}_\beta(\mathbf{X}, \mathbf{W}, \mathbf{Y}) = \|\mathbf{X}\|_* - \text{tr}(\mathbf{A}\mathbf{W}\mathbf{B}^T) + \langle \mathbf{X} - \mathbf{W}, \mathbf{Y} \rangle + \frac{\beta}{2} \|\mathbf{X} - \mathbf{W}\|_F^2.$$

onde  $\beta > 0$  é um parâmetro de penalização. As variáveis serão atualizadas de forma alternada minimizando o Lagrangiano aumentado. Computando  $\mathbf{X}^{k+1}$  temos

$$\begin{aligned} \mathbf{X}^{k+1} &= \arg \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \mathcal{L}_\beta(\mathbf{X}, \mathbf{W}^k, \mathbf{Y}^k) \\ &= \arg \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \|\mathbf{X}\|_* - \text{tr}(\mathbf{A}\mathbf{W}^k\mathbf{B}^T) + \langle \mathbf{X} - \mathbf{W}^k, \mathbf{Y}^k \rangle + \frac{\beta}{2} \|\mathbf{X} - \mathbf{W}^k\|_F^2 \\ &= \arg \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \|\mathbf{X}\|_* + \langle \mathbf{X} - \mathbf{W}^k, \mathbf{Y}^k \rangle + \frac{\beta}{2} \|\mathbf{X} - \mathbf{W}^k\|_F^2 \\ &= \arg \min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \|\mathbf{X}\|_* + \frac{\beta}{2} \left\| \mathbf{X} - \left( \mathbf{W}^k - \frac{1}{\beta} \mathbf{Y}^k \right) \right\|_F^2 \\ &= \mathcal{D}_{\frac{1}{\beta}} \left( \mathbf{W}^k - \frac{1}{\beta} \mathbf{Y}^k \right). \end{aligned}$$

Agora atualizamos a variável primal  $\mathbf{W}$ :

$$\begin{aligned} \mathbf{W}^{k+1} &= \arg \min_{\mathcal{P}_\Omega(\mathbf{W}) = \mathcal{P}_\Omega(\mathbf{T})} \mathcal{L}_\beta(\mathbf{X}^{k+1}, \mathbf{W}, \mathbf{Y}^k) \\ &= \arg \min_{\mathcal{P}_\Omega(\mathbf{W}) = \mathcal{P}_\Omega(\mathbf{T})} - \text{tr}(\mathbf{A}\mathbf{W}\mathbf{B}^T) + \langle \mathbf{X}^{k+1} - \mathbf{W}, \mathbf{Y}^k \rangle + \frac{\beta}{2} \|\mathbf{X}^{k+1} - \mathbf{W}\|_F^2 \\ &= \arg \min_{\mathcal{P}_\Omega(\mathbf{W}) = \mathcal{P}_\Omega(\mathbf{T})} - \langle \mathbf{W}, \mathbf{A}^T \mathbf{B} \rangle + \langle \mathbf{X}^{k+1} - \mathbf{W}, \mathbf{Y}^k \rangle + \frac{\beta}{2} \|\mathbf{X}^{k+1} - \mathbf{W}\|_F^2 \\ &= \arg \min_{\mathcal{P}_\Omega(\mathbf{W}) = \mathcal{P}_\Omega(\mathbf{T})} \langle \mathbf{X}^{k+1} - \mathbf{W}, \mathbf{A}^T \mathbf{B} \rangle + \langle \mathbf{X}^{k+1} - \mathbf{W}, \mathbf{Y}^k \rangle + \frac{\beta}{2} \|\mathbf{X}^{k+1} - \mathbf{W}\|_F^2 \\ &= \arg \min_{\mathcal{P}_\Omega(\mathbf{W}) = \mathcal{P}_\Omega(\mathbf{T})} \frac{\beta}{2} \left\| \mathbf{W} - \left( \mathbf{X}^{k+1} + \frac{1}{\beta} (\mathbf{A}^T \mathbf{B} + \mathbf{Y}^k) \right) \right\|_2^F \\ &= \mathcal{P}_{\Omega^C} \left( \mathbf{X}^{k+1} + \frac{1}{\beta} (\mathbf{A}^T \mathbf{B} + \mathbf{Y}^k) \right) + \mathcal{P}_\Omega(\mathbf{T}). \end{aligned}$$

Finalmente, atualizando a variável dual  $\mathbf{Y}$ , temos

$$\mathbf{Y}^{k+1} = \arg \min_{\mathbf{Y} \in \mathbb{R}^{m \times n}} \mathcal{L}_\beta(\mathbf{X}^{k+1}, \mathbf{W}^{k+1}, \mathbf{Y}) = \mathbf{Y}^k + \beta(\mathbf{X}^{k+1} - \mathbf{W}^{k+1}).$$

O procedimento do método é resumido no Algoritmo 4.

---

**Algoritmo 4 – TTNR-ADMM**


---

**Entrada:**  $\mathbf{X}^0 = \mathcal{P}_\Omega(\mathbf{T})$ ,  $\mathbf{W}^0 = \mathbf{X}^0$ ,  $\mathbf{Y}^0 = \mathbf{X}^0$ ,  $\beta > 0$ ,  $\varepsilon > 0$

$k = 0$

**enquanto**  $\|\mathbf{X}^{k+1} - \mathbf{X}^k\|_F > \varepsilon$  **faça**

$$\mathbf{X}^{k+1} = \mathcal{D}_{\frac{1}{\beta}} \left( \mathbf{W}^k - \frac{1}{\beta} \mathbf{Y}^k \right)$$

$$\mathbf{Z} = \mathbf{X}^{k+1} + \frac{1}{\beta} (\mathbf{A}^T \mathbf{B} + \mathbf{Y}^k)$$

$$\mathbf{W}^{k+1} = \mathcal{P}_{\Omega^C}(\mathbf{Z}) + \mathcal{P}_\Omega(\mathbf{T})$$

$$\mathbf{Y}^{k+1} = \mathbf{Y}^k + \beta(\mathbf{X}^{k+1} - \mathbf{W}^{k+1})$$

$$k = k + 1$$

**fim**

**retorna**  $\mathbf{X}^k$ .

---

A convergência do método é garantida pela convergência do ADMM [1, Theorem 15.4]. Note que o maior custo computacional está associado ao cálculo da SVD pelo operador de *shrinkage*. É possível fazer modificações acelerando as iterações. Para mais detalhes veja [25]. Outros métodos que envolvem o uso da norma nuclear podem ser encontrados em [34].

### Completamento via Decomposição Esqueletal

Com relação aos métodos e referências mencionados anteriormente, a abordagem feita ao problema de completamento com o uso da norma nuclear, apresenta algoritmos de simples compreensão, com bons resultados. No entanto, todos compartilham do uso do operador de *shrinkage*. Assim, a cada iteração, é necessário determinar a decomposição SVD da matriz vigente, de modo que estamos limitados a problemas de pequena a média escala.

Neste sentido, Wen *et al.* [53] apresentam a seguinte formulação

$$\begin{cases} \min_{\mathbf{U}, \mathbf{V}, \mathbf{X}} \|\mathbf{U}\mathbf{V}^T - \mathbf{X}\|_F^2 \\ \text{s.a } \mathcal{P}_\Omega(\mathbf{X}) = \mathcal{P}_\Omega(\mathbf{T}) \end{cases}, \quad (2.7)$$

onde  $\mathbf{U} \in \mathbb{R}^{m \times r}$ ,  $\mathbf{V} \in \mathbb{R}^{n \times r}$ ,  $\mathbf{X} \in \mathbb{R}^{m \times n}$  e  $r$  é o posto da matriz original  $\mathbf{T}$ . Como o posto não é conhecido, o valor de  $r$  é modificado dinamicamente durante as iterações. Dadas as

matrizes vigentes, cada variável é atualizada de maneira alternada, fixando-se as outras duas. Este procedimento é exibido no seguinte esquema iterativo para todo  $k \in \mathbb{N}$ :

$$\begin{aligned}\mathbf{U}^{k+1} &= \mathbf{X}^k (\mathbf{V}^k)^\dagger = \mathbf{X}^k (\mathbf{V}^k)^T \left( \mathbf{V}^k (\mathbf{V}^k)^T \right)^\dagger \\ \mathbf{V}^{k+1} &= (\mathbf{U}^{k+1})^\dagger \mathbf{X}^k = \left( (\mathbf{U}^{k+1})^T \mathbf{U}^{k+1} \right)^\dagger \left( (\mathbf{U}^{k+1})^T \mathbf{X}^k \right) \\ \mathbf{X}^{k+1} &= \mathcal{P}_{\Omega^C}(\mathbf{U}^{k+1} \mathbf{V}^{k+1}) + \mathcal{P}_\Omega(\mathbf{T}).\end{aligned}\quad (2.8)$$

É possível acelerar o método com o uso de técnicas de relaxações sucessivas extrapolando as componentes do ponto encontrado durante as iterações [53]. Uma estratégia para avaliar aproximadamente o posto  $r$  é inicializar a aproximação com um valor pequeno e apenas aumentá-lo quando o método está estagnado. Para outras informações e detalhes sobre convergência, veja [53].

## 2.2 Completamento de Ordem Superior

Nos problemas de completamento, é necessário o uso de ferramentas que possam capturar tanto informação global quanto local da estrutura de dados, bem como as possíveis correlações das entradas [35]. No caso matricial, o posto é uma ferramenta poderosa, como ilustrado na Figura 11. No entanto, a noção de posto para estruturas de alta ordem, e.g. uma imagem colorida, não é única. Nesta seção vamos estudar diversos algoritmos de completamento que se baseiam nas noções de posto PARAFAC, Tucker e *Tensor Train*. A Tabela 4 contém todas as informações pertinentes dos métodos apresentados neste trabalho.

Considere um tensor  $\mathcal{T} \in \mathbb{R}^{I_1 \times \dots \times I_N}$  e um subconjunto de índices  $\Omega \subset \{1, \dots, I_1\} \times \dots \times \{1, \dots, I_N\}$ . Generalizando o modelo (2.2), obtemos o seguinte problema de otimização

$$\begin{cases} \min_{\mathcal{X} \in \mathbb{R}^{I_1 \times \dots \times I_N}} \|\mathcal{X}\|_* \\ \text{s.a } \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases} . \quad (2.9)$$

No entanto, a norma nuclear para tensores precisa ser definida. Neste caso, a extensão para o caso de alta ordem pode ser dada pela seguinte combinação convexa

$$\|\mathcal{X}\|_* := \sum_{i=1}^N \alpha_i \|\mathbf{X}_{(i)}\|_*,$$

onde  $\sum_{i=1}^N \alpha_i = 1$  e  $\alpha_i \geq 0$ . Outra possibilidade seria

$$\|\mathcal{X}\|_* := \sum_{i=1}^{N-1} \alpha_i \|\mathbf{X}_{<i>}\|_*$$

Tabela 4 – Panorama das estratégias consideradas neste trabalho, com as respectivas nomenclaturas, referências e principais características.

Nomenclatura em inglês	Sigla	Referência	Nomenclatura adotada	Características principais
<i>Heuristic Algorithms</i>	PARAFAC e HOSVD	Liu et al. (2013) [35]	Algoritmos Heurísticos	Métodos básicos envolvendo as abordagens PARAFAC e HOSVD, para fins comparativos.
<i>Simple Low Rank Tensor Completion</i>	SILRTC	Liu et al. (2013) [2]	Método Simples de Completamento Tensorial de Baixo Posto	Utiliza a técnica de relaxação para as restrições e o método de descenso por blocos coordenados para obter a solução ótima.
<i>High Accuracy Low-Rank Tensor Completion</i>	HALRTC	Liu et al. (2013) [35]	Método de Completamento Tensorial de Alta Acurácia	Utiliza o método das direções alternadas e multiplicadores (ADMM) para o modelo em questão.
<i>Simple Low Rank Tensor Completion via Tensor Train</i>	SILRTC-TT	Bengua et al. (2017) [2]	Método Simples de Completamento Tensorial de Baixo Posto via Tensor Train	Similar ao SILRTC, mas considera a minimização do posto-TT ao invés do posto multilinear.
<i>Tensor Completion by Parallel Matrix Factorization</i>	TMac	Xu et al. (2015) [54]	Completamento Tensorial por Fatoração Matricial	Recupera um tensor de posto baixo por meio de fatorações simultâneas das matrizações de todos os modos do tensor de interesse em um algoritmo de otimização alternada.
<i>Tensor Completion by Parallel Matrix Factorization via Tensor Train</i>	TMac-TT	Bengua et al. (2017) [2]	Completamento Tensorial por Fatoração Matricial via Tensor Train	Similar ao TMac, mas considera a minimização do posto-TT ao invés do posto multilinear.
<i>Tensor Train Weighted Optimization</i>	TT-WOPT	Yuan et al. (2019) [55]	Otimização Ponderada via Tensor Train	Otimiza os fatores da decomposição TT por meio de uma estratégia envolvendo quadrados mínimos ponderados e algoritmos de primeira ordem.
Tensor Train Weighted Optimization with Dynamic Updating of the TT-rank	TT-WOPT-DU	Esta dissertação	Otimização Ponderada via Tensor Train com atualização dinâmica do posto-TT	Estratégia baseada no TT-WOPT com atualização dinâmica do posto-TT inspirada na proposta de [46].

com  $\sum_{i=1}^{N-1} \alpha_i = 1$  e  $\alpha_i \geq 0$ . Aqui, não é possível usar o envelope convexo do posto do tensor, pois, como vimos anteriormente, computar esta grandeza é um problema do tipo NP-difícil. Além disso, dado um número inteiro não negativo  $r$ , o conjunto de tensores de posto menor ou igual a  $r$  não é fechado (cf. [29, §3.3], ver também [22, §9.3]).

## Algoritmos Heurísticos

Em primeiro lugar, apresentamos dois algoritmos heurísticos básicos, seguindo [35], que servirão para estabelecer um comparativo. Uma abordagem inicial é utilizar a decomposição em fatores paralelos (PARAFAC) da seguinte maneira

$$\begin{cases} \min_{\mathcal{X}, \mathbf{U}^{(1)}, \dots, \mathbf{U}^{(N)}} \|\mathcal{X} - \mathbf{U}^{(1)} \circ \dots \circ \mathbf{U}^{(N)}\|_F^2 \\ \text{s.a } \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases}, \quad (2.10)$$

onde  $\mathbf{U}^{(k)} \in \mathbb{R}^{I_k \times r}$ ,  $k = 1, \dots, N$  e  $r$  denota o posto de  $\mathcal{X}$ . É possível prosseguir atualizando as variáveis de forma alternada. Primeiramente, tomamos  $\mathcal{X}$  como o tensor amostrado  $\mathcal{T}$  e calculamos a sua decomposição PARAFAC. Com os novos fatores, recalculamos  $\mathcal{X}$ , fazemos a projeção e repetimos o processo.

Outra heurística utilizada será com a decomposição HOSVD. Supondo que o posto multilinear de  $\mathcal{X}$  seja  $(r_1, \dots, r_N)$ , o problema é formulado como

$$\begin{cases} \min_{\mathcal{X}, \mathcal{G}, \mathbf{U}^{(1)}, \dots, \mathbf{U}^{(N)}} \|\mathcal{X} - \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \dots \times_N \mathbf{U}^{(N)}\|_F^2 \\ \text{s.a } \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases}, \quad (2.11)$$

onde  $\mathcal{G} \in \mathbb{R}^{r_1 \times \dots \times r_N}$  e  $\mathbf{U}^{(k)} \in \mathbb{R}^{I_k \times r_k}$  possui colunas ortonormais, para  $k = 1, \dots, N$ . Prosseguimos da mesma maneira que na heurística anterior.

## Método Simples de Completamento Tensorial de Baixo Posto (SiLRTC)

O problema (2.9) apresenta dificuldade para ser resolvido devido à conexão entre os modos, isto é, enquanto otimizamos a soma de múltiplas normas nucleares, as matrizes associadas aos modos compartilham as mesmas entradas e não podem ser minimizadas de forma independente [35]. É necessário separá-los introduzindo matrizes auxiliares  $\mathbf{M}^{(1)}, \dots, \mathbf{M}^{(N)}$ , obtendo a seguinte formulação equivalente

$$\begin{cases} \min_{\mathcal{X}, \mathbf{M}^{(1)}, \dots, \mathbf{M}^{(N)}} \sum_{i=1}^N \alpha_i \|\mathbf{M}^{(i)}\|_* \\ \text{s.a } \mathbf{X}_{(i)} = \mathbf{M}^{(i)}, \quad i = 1, \dots, N \\ \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases}. \quad (2.12)$$

Adicionalmente, pode-se relaxar as restrições dos modos, para alguns valores de  $\beta_i > 0$ ,  $i = 1, \dots, N$ :

$$\begin{cases} \min_{\mathcal{X}, \mathbf{M}^{(1)}, \dots, \mathbf{M}^{(N)}} \sum_{i=1}^N \alpha_i \|\mathbf{M}^{(i)}\|_* + \frac{\beta_i}{2} \|\mathbf{X}_{(i)} - \mathbf{M}^{(i)}\|_F^2 \\ \text{s.a } \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases}. \quad (2.13)$$

A solução é calculada a partir da otimização em blocos, como fizemos em outras formulações. Minimizamos um bloco de variáveis enquanto os outros estão fixados. Os blocos são divididos nas variáveis  $\mathcal{X}$  e  $\mathbf{M}^{(i)}$ ,  $i = 1, \dots, N$ . Para a variável  $\mathcal{X}$ , resolvemos o seguinte subproblema

$$\mathcal{X} = \arg \min_{\mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T})} \sum_{i=1}^N \frac{\beta_i}{2} \|\mathbf{X}_{(i)} - \mathbf{M}^{(i)}\|_F^2.$$

Observando que a função a ser minimizada é limitada inferiormente por zero, basta determinarmos  $\mathcal{X}$  cujo  $i$ -ésimo desdobramento seja  $\mathbf{M}^{(i)}$ . Neste caso, quando reconstruirmos os tensores a partir de  $\mathbf{M}^{(i)}$  no  $i$ -ésimo modo, devemos recuperar o tensor  $\mathcal{X}$ . Logo, segue que

$$\mathcal{X} = \mathcal{P}_{\Omega^C} \left( \frac{\sum_{i=1}^N \beta_i \text{fold}_i \{\mathbf{M}^{(i)}\}}{\sum_{i=1}^N \beta_i} \right) + \mathcal{P}_\Omega(\mathcal{T}). \quad (2.14)$$

No caso da matriz  $\mathbf{M}^{(i)}$ , apenas uma parcela de cada somatório é selecionada, resultando que

$$\mathbf{M}^{(i)} = \arg \min_{\mathbf{M}} \alpha_i \|\mathbf{M}\|_* + \frac{\beta_i}{2} \|\mathbf{X}_{(i)} - \mathbf{M}\|_F^2 = \mathcal{D}_{\frac{\alpha_i}{\beta_i}}(\mathbf{X}_{(i)}).$$

O termo diferenciável é convexo e o termo não diferenciável também é convexo e separável. Neste caso, o método do descenso coordenado por blocos tem convergência garantida para o mínimo global [1, Theorem 11.18]. O pseudocódigo que resume o método é exibido no Algoritmo 5:

---

#### Algoritmo 5 – SiLRTC

---

**Entrada:**  $\mathcal{X}^0 = \mathcal{P}_\Omega(\mathcal{T})$ ,  $\alpha_1, \dots, \alpha_N \geq 0$ , com  $\sum_{i=1}^N \alpha_i = 1$ ,  $\beta_1, \dots, \beta_N > 0$ ,  $\varepsilon > 0$

$k = 0$

**enquanto**  $\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F > \varepsilon$  **faça**

<b>para</b> $i = 1 : N$ <b>faça</b> $\quad \mid \mathbf{M}^{(i)} = \mathcal{D}_{\frac{\alpha_i}{\beta_i}}(\mathbf{X}_{(i)})$ <b> fim</b> Atualize $\mathcal{X}^k$ por (2.14). $k = k + 1$ <b>fim</b> <b>retorna</b> $\mathcal{X}^k$ .
---

---

## Método de Completamento Tensorial de Alta Acurácia (HaLRTC)

Baseado no método anterior, podemos explorar a estruturação do problema através do método de direções alternadas e multiplicadores (ADMM). De acordo com [1, Chapter 15], consideramos tensores auxiliares  $\mathcal{M}^{(i)}$ , cujo  $j$ -ésimo desdobramento  $\mathbf{M}_{(j)}^{(i)}$  é associado ao  $j$ -ésimo desdobramento de  $\mathcal{X}$  na função objetivo. A formulação é exibida abaixo:

$$\begin{cases} \min_{\mathcal{X}, \mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}} \sum_{i=1}^N \alpha_i \|\mathcal{M}_{(i)}^{(i)}\|_* \\ \text{s.a } \mathcal{X} = \mathcal{M}^{(i)}, \quad i = 1, \dots, N, \\ \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases} \quad (2.15)$$

onde  $\sum_{i=1}^N \alpha_i = 1$ ,  $\alpha_i \geq 0$  e  $\beta > 0$ . Considerando variáveis duais  $\mathcal{Y}^{(i)}$ , para  $i = 1, \dots, N$ , buscamos minimizar o Lagrangiano aumentado associado por blocos. Então,

$$\begin{aligned} \mathcal{L}_\beta(\mathcal{X}, \mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}, \mathcal{Y}^{(1)}, \dots, \mathcal{Y}^{(N)}) = \\ \sum_{i=1}^N \left( \alpha_i \|\mathbf{M}_{(i)}^{(i)}\|_* + \langle \mathcal{X} - \mathcal{M}^{(i)}, \mathcal{Y}^{(i)} \rangle + \frac{\beta}{2} \|\mathcal{X} - \mathcal{M}^{(i)}\|_F^2 \right). \end{aligned}$$

Atualizando o bloco dos tensores auxiliares  $\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}$ , segue que

$$\begin{aligned} \{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}\} &= \arg \min_{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}} \mathcal{L}_\beta(\mathcal{X}, \mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}, \mathcal{Y}^{(1)}, \dots, \mathcal{Y}^{(N)}) \\ &= \arg \min_{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}} \sum_{i=1}^N \left( \alpha_i \|\mathbf{M}_{(i)}^{(i)}\|_* + \langle \mathcal{X} - \mathcal{M}^{(i)}, \mathcal{Y}^{(i)} \rangle + \frac{\beta}{2} \|\mathcal{X} - \mathcal{M}^{(i)}\|_F^2 \right) \\ &= \arg \min_{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}} \sum_{i=1}^N \alpha_i \|\mathbf{M}_{(i)}^{(i)}\|_* - \langle \mathcal{M}^{(i)}, \mathcal{Y}^{(i)} \rangle + \frac{\beta}{2} \left( \|\mathcal{X}\|_F^2 + \|\mathcal{M}^{(i)}\|_F^2 - \langle \mathcal{M}^{(i)}, \mathcal{X} \rangle \right) \\ &= \arg \min_{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}} \sum_{i=1}^N \alpha_i \|\mathbf{M}_{(i)}^{(i)}\|_* - \frac{\beta}{2} \langle \mathcal{M}^{(i)}, \mathcal{X} + \frac{1}{\beta} \mathcal{Y}^{(i)} \rangle + \frac{\beta}{2} \|\mathcal{X}\|_F^2 + \|\mathcal{M}^{(i)}\|_F^2 \\ &= \arg \min_{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}} \sum_{i=1}^N \alpha_i \|\mathbf{M}_{(i)}^{(i)}\|_* + \frac{\beta}{2} \left\| \mathbf{M}_{(i)}^{(i)} - \left( \mathbf{X}_{(i)} + \frac{1}{\beta} \mathbf{Y}_{(i)}^{(i)} \right) \right\|_F^2. \end{aligned}$$

Novamente, fazemos uso do operador de *shrinkage* sobre a matriz  $\mathbf{X}_{(i)} + \frac{1}{\beta} \mathbf{Y}_{(i)}^{(i)}$  e reconstruímos o tensor a partir do  $i$ -ésimo modo. Desta forma, temos

$$\mathcal{M}^{(i)^{k+1}} = \text{fold}_i \left\{ \mathcal{D}_{\frac{\alpha_i}{\beta}} \left( \mathbf{X}_{(i)}^k + \frac{1}{\beta} \mathbf{Y}_{(i)}^{(i)^k} \right) \right\} \quad \forall k \in \mathbb{N}, \quad (2.16)$$

para  $i = 1, \dots, N$ . No caso da variável  $\mathcal{X}$ , a atualização é feita por

$$\begin{aligned} \mathcal{X} &= \arg \min_{\mathcal{P}_\Omega(\mathcal{X})=\mathcal{P}_\Omega(\mathcal{T})} \mathcal{L}_\beta(\mathcal{X}, \mathcal{M}^{(1)}, \dots, \mathcal{M}^{(N)}, \mathcal{Y}^{(1)}, \dots, \mathcal{Y}^{(N)}) \\ &= \arg \min_{\mathcal{P}_\Omega(\mathcal{X})=\mathcal{P}_\Omega(\mathcal{T})} \sum_{i=1}^N \langle \mathcal{X} - \mathcal{M}^{(i)}, \mathcal{Y}^{(i)} \rangle + \frac{\beta}{2} \|\mathcal{X} - \mathcal{M}^{(i)}\|_F^2. \end{aligned}$$

Desta maneira,

$$\mathcal{X}^{k+1} = \mathcal{P}_{\Omega^C} \left( \frac{1}{N} \left( \sum_{i=1}^N \mathcal{M}^{(i)k+1} - \frac{1}{\beta} \mathcal{Y}^k \right) \right) + \mathcal{P}_{\Omega}(\mathcal{T}), \quad \forall k \in \mathbb{N}. \quad (2.17)$$

A atualização das variáveis duais é dada por

$$\mathcal{Y}^{k+1} = \mathcal{Y}^k + \beta(\mathcal{X} - \mathcal{M}^{(i)}).$$

para  $i = 1, \dots, N$ . Para mais detalhes e informações sobre este método e possíveis alterações, veja [35]. O pseudocódigo que resume os processos é exibido no Algoritmo 6. A convergência do método HaLRTC segue da convergência do método ADMM [1, Theorem 15.4].

---

#### Algoritmo 6 – HALRTC

---

**Entrada:**  $\mathcal{X}^0 = \mathcal{P}_{\Omega}(\mathcal{T})$ ,  $\mathcal{Y}^{(1)} = \dots = \mathcal{Y}^{(N)} = 0$ ,  $\alpha_1, \dots, \alpha_N \geq 0$  com  $\sum_{n=1}^N \alpha_n = 1$ ,  
 $\beta > 0$ ,  $\varepsilon > 0$

$k = 0$

**enquanto**  $\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F > \varepsilon$  **faça**

**para**  $i = 1 : N$  **faça**

$\mathcal{M}^{(i)} = \text{fold}_i \left\{ \mathcal{D}_{\frac{\alpha_i}{\beta}} \left( \mathbf{X}_{(i)} - \frac{1}{\beta} \mathcal{Y}_{(i)}^{(i)} \right) \right\}$

**fim**

Atualize  $\mathcal{X}^k$  por (2.17).

$\mathcal{Y} = \mathcal{Y} + \beta(\mathcal{X} - \mathcal{M}^{(i)})$

$k = k + 1$

**fim**

**retorna**  $\mathcal{X}^k$ .

---

É importante notar que as formulações apresentadas até aqui levam em conta a minimização do posto multilinear, cujas componentes são os postos dos desdobramentos canônicos apresentados na Definição 1.4. Note que este esquema é desbalanceado, na medida que temos, para cada entrada do vetor de posto, um modo associado ao produto das componentes restantes. Neste sentido, o posto da matriz resultante do desdoblamento não é capaz de capturar a correlação entre as entradas do tensor com entradas faltantes. A minimização do posto requer o uso de um esquema balanceado e o posto multilinear pode não ser efetivo na resolução dos problemas de ordens mais altas. De fato, em [2], é mostrado que a função posto( $\mathbf{X}_{(n)}$ ) não é apropriada para capturar a correlação global do tensor. Na verdade, essa é capaz de capturar a correlação entre apenas um modo e o restante. Para superarmos este empecilho, um esquema de desdoblamento balanceado pode ser empregado através da minimização do posto-TT. Neste caso, os modos estão mais correlacionados pois o posto( $\mathbf{X}_{<n>}$ ) captura a correlação dos  $n$  primeiros modos com os restantes. No caso do problema de completamento de imagens, é introduzida uma técnica

chamada *Ket Augmentation* (KA) para representarmos um tensor de ordem baixa por outro de ordem maior sem alterar o número de entradas. O esquema de KA proporciona uma boa maneira de explorarmos o potencial do posto-TT na otimização de imagens coloridas [2]. Exibimos o esquema a seguir e, posteriormente, alguns métodos associados.

### **Ket Augmentation**

Nesta seção apresentamos o processo de *Ket Augmentation* (KA), que mapeia as entradas de um tensor de ordem baixa para outro de ordem maior sem alterar o número de elementos. A apresentação será feita para tensores de ordem 3 para melhor fixação das ideias. Dado  $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ , através do mapeamento, obtemos um tensor  $\tilde{\mathcal{T}} \in \mathbb{R}^{J_1 \times \dots \times J_N}$  tal que  $I_1 I_2 I_3 = \prod_{i=1}^N J_i$ . Em palavras, as entradas do tensor são endereçadas apropriadamente para uma estrutura de blocos, onde cada um destes é selecionado por índices [32]. A fim de simplificar a discussão, vamos considerar  $I_1 \times I_2 = 2^N \times 2^N$  o número de pixels de uma imagem e  $I_3 = 3$  o número de cores.

Primeiramente, começamos com um bloco de  $2^1 \times 2^1$  píxeis de cor  $j$  fixada. Defina um vetor  $\mathbf{e}_{i_1}$  e suponha que  $\mathbf{e}_1$  está associado ao canto superior esquerdo,  $\mathbf{e}_2$  ao superior direito,  $\mathbf{e}_3$  ao inferior esquerdo e  $\mathbf{e}_4$  ao inferior direito do bloco. Então, temos que

$$\mathcal{T} = \sum_{i_1=1}^4 c_{i_1,j} \mathbf{e}_{i_1} \in \mathbb{R}^{2^1 \times 2^1 \times 1}, \quad j = 1, 2, 3 \quad (2.18)$$

onde  $c_{i_1,j}$  é o valor do pixel associado à cor  $j$  e ao bloco  $\mathbf{e}_{i_1}$ . Agora, considerando a base canônica do  $\mathbb{R}^3$  para representar o espectro RGB, denotada por  $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$  segue

$$\mathcal{T} = \sum_{i_1=1}^4 \sum_{j=1}^3 c_{i_1,j} \mathbf{e}_{i_1} \circ \mathbf{u}_j \in \mathbb{R}^{2^1 \times 2^1 \times 3}. \quad (2.19)$$

Agora, consideramos um bloco maior denotado por  $i_2$  formado por 4 sub-blocos para cada cor  $j$  (veja Figura 12). Este novo bloco integra a seguinte relação

$$\mathcal{T} = \sum_{i_2=1}^4 \sum_{i_1=1}^4 \sum_{j=1}^3 c_{i_2,i_1,j} \mathbf{e}_{i_2} \circ \mathbf{e}_{i_1} \circ \mathbf{u}_j \in \mathbb{R}^{2^2 \times 2^2 \times 3}.$$

Após diversos passos, todo o tensor será coberto pela estrutura de blocos. Ao final, a imagem de tamanho  $2^N \times 2^N \times 3$ , é representada por um tensor de ordem  $N + 1$  de dimensões  $4 \times 4 \times \dots \times 4 \times 3$  dado por

$$\mathcal{T} = \sum_{i_N=1}^4 \dots \sum_{i_2=1}^4 \sum_{i_1=1}^4 \sum_{j=1}^3 c_{i_N, \dots, i_2, i_1, j} \mathbf{e}_{i_N} \circ \dots \circ \mathbf{e}_{i_2} \circ \mathbf{e}_{i_1} \circ \mathbf{u}_j \in \mathbb{R}^{2^N \times 2^N \times 3}.$$

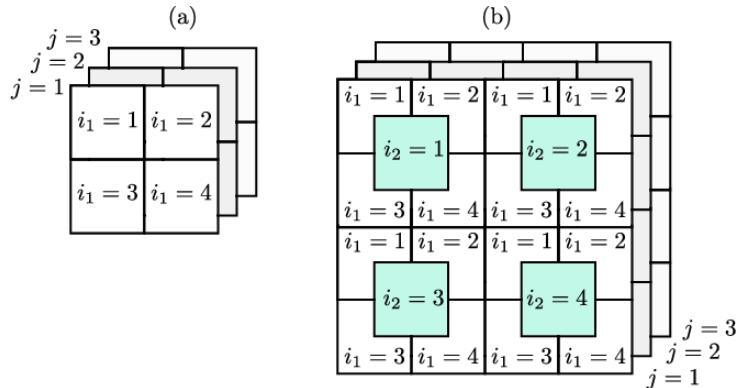


Figura 12 – Procedimento de mapeamento para uma estrutura de blocos. O bloco mais simples (a) é representado por (2.18), que junto a outros quatro formam o bloco (b), representado por (2.19). Retirada de [2, Fig. 1].

Para ajudar fixar as ideias, considere uma imagem de dimensões  $4^4 \times 4^4 \times 3$ . Através do procedimento de KA, será obtido um tensor de ordem 9 de dimensões  $4 \times 4 \times 3$ . Partindo do último índice  $i_N$ , dividimos a imagem em quatro blocos de dimensões  $128 \times 128$  organizados da maneira mostrada na Figura 13 e escolhemos um deles. Sua enumeração será o valor de  $i_N$ . Agora, dividimos o bloco vigente em outros quatro de dimensões  $64 \times 64$ , indexados pelo índice  $i_{N-1}$  e prosseguimos de maneira análoga para os índices restantes.

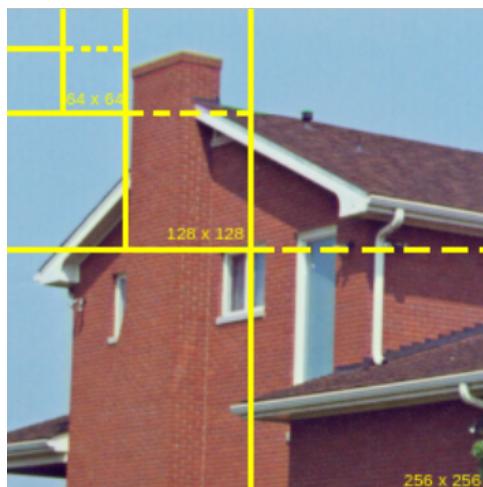


Figura 13 – Ilustração para o processo de KA em uma imagem de dimensões  $256 \times 256 \times 3$ . Baseada em [55, Fig. 3].

Esta forma de apresentação é adequada para o processamento de imagens, pois os *pixels* são reordenados em um tensor de alta ordem e, desse modo, as texturas (detalhes das imagens) são melhor assimiladas pelo posto-TT e sua flexibilidade [2].

No Exemplo 2.1 ilustramos a dinâmica do mapeamento para uma matriz  $4 \times 4$ .

**Exemplo 2.1**

Seja  $\mathbf{A} \in \mathbb{R}^{4 \times 4}$  dada por

$$\mathbf{A} = \begin{pmatrix} 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \\ 4 & 8 & 12 & 16 \end{pmatrix}.$$

Vamos considerar, as entradas  $a_{4,3} = 12$  e  $a_{1,3} = 9$ . Aplicando o processo de KA, dividimos a matriz em blocos da seguinte forma

$$\mathbf{A} = \left( \begin{array}{cc|cc} 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ \hline 3 & 7 & 11 & 15 \\ 4 & 8 & 12 & 16 \end{array} \right).$$

O elemento  $a_{1,3}$  será mapeado, primeiramente, para o bloco inferior esquerdo e, então, para o canto superior esquerdo. O elemento  $a_{4,3}$  será levado para o bloco inferior esquerdo e, então, para o canto inferior direito. Dessa maneira,

$$\text{KA}(\mathbf{A}) = \begin{pmatrix} 1 & 2 & 5 & 6 \\ 3 & 4 & 7 & 8 \\ 9 & 10 & 13 & 14 \\ 11 & 12 & 15 & 16 \end{pmatrix}.$$

**Método Simples de Completamento Tensorial via *Tensor Train* (SiLRTC-TT)**

Denominamos como método SiLRTC-TT a resolução do problema (2.13) modificado levando em conta a minimização do posto-TT ao invés do posto multilinear. Então, obtemos a seguinte formulação:

$$\begin{cases} \min_{\mathcal{X}, \mathbf{M}^{(1)}, \dots, \mathbf{M}^{(N-1)}} \sum_{i=1}^{N-1} \alpha_i \|\mathbf{M}^{(i)}\|_* + \frac{\beta_i}{2} \|\mathbf{X}_{<i>} - \mathbf{M}^{(i)}\|_F^2 \\ \text{s.a } \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases}. \quad (2.20)$$

O procedimento é análogo ao SiLRTC, que já descrevemos. Fazemos uso do descenso coordenado por blocos para otimizar um grupo de variáveis por vez enquanto os demais estão fixos. O algoritmo tem convergência garantida para uma solução única, pois a função objetivo é convexa e o termo não-diferenciável é separável [1, Theorem 11.18]. No Algoritmo 7, apresentamos o pseudocódigo que resume o método.

**Algoritmo 7 – SiLRTC-TT**


---

**Entrada:**  $\mathcal{X}^0 = \mathcal{P}_\Omega(\mathcal{T})$ ,  $\alpha_1, \dots, \alpha_{N-1} \geq 0$ , com  $\sum_{n=1}^{N-1} \alpha_n = 1$ ,  $\beta_1, \dots, \beta_{N-1} > 0$ ,  
 $\varepsilon > 0$

$k = 0$

**enquanto**  $\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F > \varepsilon$  **faça**

**para**  $i = 1 : N$  **faça**

$\mathbf{M}^{(i)} = \mathcal{D}_{\frac{\alpha_i}{\beta_i}}(\mathbf{X}_{<i>})$

**fim**

Atualize  $\mathcal{X}^k$  por (2.14).

$k = k + 1$

**fim**

**retorna**  $\mathcal{X}^k$ .

---

**Completamento Tensorial por Fatoração Matricial via *Tensor Train* (TMac-TT)**

Seja  $\mathcal{X} \in \mathbb{R}^{I_1 \times \dots \times I_N}$  com postos-TT iguais a  $r_1, \dots, r_{N-1}$ . Baseado no caso matricial, aplicamos uma decomposição de baixo posto para cada um dos desdobramentos  $\mathbf{X}_{(k)}$  escolhendo matrizes  $\mathbf{U}^{(k)} \in \mathbb{R}^{\prod_{j=1}^k I_j \times r_k}$  e  $\mathbf{V}^{(k)} \in \mathbb{R}^{r_k \times \prod_{j=k+1}^N I_j}$ . Dessa forma, estamos interessados na resolução do seguinte problema:

$$\begin{cases} \min_{\mathcal{X}, \{\mathbf{U}^{(k)}, \mathbf{V}^{(k)}\}_{k=1}^{N-1}} \sum_{k=1}^{N-1} \frac{\alpha_k}{2} \|\mathbf{U}^{(k)} \mathbf{V}^{(k)} - \mathbf{X}_{<k>}\|_F^2, \\ \text{s.a } \mathcal{P}_\Omega(\mathcal{X}) = \mathcal{P}_\Omega(\mathcal{T}) \end{cases}, \quad (2.21)$$

onde  $\sum_{k=1}^{N-1} \alpha_k = 1$  e  $\alpha_k \geq 0$  para  $k = 1, \dots, N-1$ . Como feito em [2], aplicamos o descenso coordenado por blocos minimizando um grupo de variáveis por vez. Observe que o problema é não linear e não convexo, portanto uma solução global não é garantida. Neste caso, focamos nossas atenções para o subproblema convexo

$$\min_{\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \mathcal{X}} \|\mathbf{U}^{(k)} \mathbf{V}^{(k)} - \mathbf{X}_{<k>}\|_F^2.$$

Note que este é um problema de quadrados mínimos convexo em cada uma das variáveis  $\mathbf{U}^{(k)}, \mathbf{V}^{(k)}$  e  $\mathcal{X}$ . A resolução é similar àquela feita na seção anterior para matrizes, e apresentada no esquema (2.8). Dessa maneira, para todo  $k \in \mathbb{N}$ , atualizamos as variáveis ciclicamente. Assim, temos

$$\mathbf{U}^{(n)^{k+1}} = \mathbf{X}_{<n>}^k \left( \mathbf{V}^{(n)^k} \right)^T \left( \mathbf{V}^{(n)^k} \left( \mathbf{V}^{(n)^k} \right)^T \right)^\dagger, \quad n = 1, \dots, N-1$$

$$\mathbf{V}^{(n)^{k+1}} = \left( \left( \mathbf{U}^{(n)^{k+1}} \right)^T \mathbf{U}^{(n)^{k+1}} \right)^\dagger \left( \mathbf{U}^{(n)^{k+1}} \right)^T \mathbf{X}_{<n>}^k, \quad n = 1, \dots, N-1$$

$$\mathbf{X}_{<n>}^{k+1} = \mathbf{U}^{(n)^{k+1}} (\mathbf{V}^{(n)^{k+1}})^T.$$

Após atualizarmos as variáveis para  $n = 1, \dots, N - 1$ , computamos o tensor  $\mathcal{X}^{k+1}$  por

$$\mathcal{X}^{k+1} = \mathcal{P}_{\Omega^C} \left( \sum_{n=1}^{N-1} \alpha_k \text{fold}_n \left\{ \mathbf{U}^{(n)^{k+1}} (\mathbf{V}^{(n)^{k+1}})^T \right\} \right) + \mathcal{P}_{\Omega}(\mathcal{T}). \quad (2.22)$$

Em [54, Lemma 3.1] é mostrado que a iteração para  $\mathbf{U}^{(n)}$  pode ser simplificada desconsiderando o cálculo da pseudoinversa. Ou seja, é necessário calcular apenas a expressão

$$\mathbf{U}^{(n)^{k+1}} = \mathbf{X}_{<n>}^k (\mathbf{V}^{(n)^k})^T$$

para  $n = 1, \dots, N - 1$ . Deste modo, o cálculo da pseudoinversa é feito uma única vez por modo. Supondo que os tensores têm baixo posto-TT, o cálculo da decomposição SVD é feito sobre uma matriz quadrada de pequenas dimensões, diferentemente dos métodos onde a norma nuclear é empregada.

O problema (2.21) necessita que uma estimativa dos postos-TT seja dada como entrada. Se estes forem fixos, é importante que a aproximação seja suficientemente boa para que o modelo funcione bem. Valores de posto muito pequenos podem ser insuficientes (*underfitting*): a solução encontrada apresentará um erro grande. Postos-TT sobreajustados (*overfitting*) podem causar grande desvio do tensor original e ruídos podem ser incorporados na solução. Neste trabalho, será utilizado um esquema de adaptação dinâmica dos postos-TT. Neste propósito, a cada iteração calculamos a distância entre as soluções da iteração corrente e da anterior. À medida que esta distância pouco varia, notamos que o método está estagnando e é preciso recalcular os valores de posto-TT.

Para obter os valores iniciais de posto-TT, cada  $r_k$  é limitado escolhendo os valores singulares que satisfazem

$$\frac{\sigma_j^{(i)}}{\sigma_1^{(i)}} > \lambda, \quad \text{para } j = 1, \dots, r_k, \quad (2.23)$$

onde  $\lambda > 0$  e  $\sigma_j^{(i)}$  é o  $j$ -ésimo valor singular do  $i$ -ésimo desdobramento de  $\mathcal{X}$ . No Algoritmo 8, resumimos as etapas do método.

Fixados os parâmetros  $\alpha_k$  e  $r_k$ , para  $k = 1, \dots, N - 1$ , foi demonstrado em [54, Theorem 4.1] que a sequência de tensores  $\{\mathcal{X}^k\}_{k \in \mathbb{N}}$  gerada pelo método converge para um ponto estacionário quando consideramos a definição de posto multilinear. Observando a demonstração deste teorema, existe a possibilidade desta ser estendida para o caso da definição de posto-TT.

**Algoritmo 8 – TMAC-TT**


---

**Entrada:**  $\mathcal{X}^0 = \mathcal{P}_\Omega(\mathcal{T})$ ,  $\alpha_1, \dots, \alpha_{N-1} \geq 0$ , com  $\sum_{n=1}^{N-1} \alpha_n = 1$ ,  $r_1, \dots, r_{N-1}$ ,  $\lambda > 0$ ,  $\varepsilon > 0$

**para**  $n = 1 : N - 1$  **faça**

- | Estime o posto-TT inicial  $r_n$  de  $\mathbf{X}_{<n>}$  utilizando os valores singulares que satisfazem (2.23)
- | Calcule uma aproximação inicial de  $\mathbf{U}^{(n)}$  e  $\mathbf{V}^{(n)}$  de postos iguais à  $r_n$

**fim**

$k = 0$

**enquanto**  $\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F > \varepsilon$  **faça**

**para**  $n = 1 : N - 1$  **faça**

- | Calcule o  $n$ -ésimo desdobramento  $\mathbf{X}_{\langle n \rangle}$
- |  $\mathbf{U}^{(n)k+1} = \mathbf{X}_{<n>}^k (\mathbf{V}^{(n)k})^T$
- |  $\mathbf{V}^{(n)k+1} = \left( (\mathbf{U}^{(n)k+1})^T \mathbf{U}^{(n)k+1} \right)^\dagger (\mathbf{U}^{(n)k+1})^T \mathbf{X}_{<n>}^k$
- |  $\mathbf{X}_{<n>}^{k+1} = \mathbf{U}^{(n)k+1} (\mathbf{V}^{(n)k+1})^T$

**fim**

Atualize  $\mathcal{X}^k$  por (2.22).

$k = k + 1$

**fim**

**retorna**  $\mathcal{X}^k$ .

---

**Otimização Ponderada via *Tensor Train* (TT-WOPT)**

Vamos definir o tensor parcialmente observado com entradas faltantes por  $\mathcal{Y} = \mathcal{P}_\Omega(\mathcal{T})$  e por  $\mathcal{X}$ , a representação TT dada por núcleos  $\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \dots, \mathcal{G}^{(N)}$ . Estamos interessados em encontrar uma aproximação dos núcleos-TT de  $\mathcal{Y}$ . Assim, temos a seguinte função objetivo a ser minimizada [55]:

$$f(\mathcal{G}^{(1)}, \mathcal{G}^{(2)}, \dots, \mathcal{G}^{(N)}) = \frac{1}{2} \|\mathcal{Y} - \mathcal{P}_\Omega(\mathcal{X})\|_F^2. \quad (2.24)$$

Observe que este é um problema não linear de otimização associado aos núcleos-TT. A abordagem será feita tirando proveito da diferenciabilidade da função objetivo e de métodos bem estabelecidos baseados no vetor gradiente. Para calcularmos as derivadas parciais de  $f$ , lembre-se da equação (1.23) dada por

$$\mathbf{X}_{(k)} = \mathbf{G}_{(2)}^{(k)} (\mathbf{G}_{(1)}^{<k} \otimes \mathbf{G}_{(k)}^{>k}), \quad k = 1, \dots, N.$$

O valor da função objetivo é o mesmo considerando desdobramentos de  $\mathcal{X}$ . Neste caso, é possível calcular as derivadas parciais de  $f$  com relação aos desdobramentos do segundo modo dos núcleos. A partir das regras de derivação [42], segue que

$$\frac{\partial f}{\partial \mathbf{G}_{(2)}^{(k)}} = (\mathbf{X}_{(k)} - \mathbf{Y}_{(k)}) (\mathbf{G}_{(1)}^{<k} \otimes \mathbf{G}_{(k)}^{>k})^T, \quad k = 1, \dots, N. \quad (2.25)$$

Desse modo, é possível atualizar os núcleos  $\mathcal{G}^{(k)}$  a partir do gradiente de  $f$ . Uma manobra feita em nossos testes consiste em considerar o gradiente como um vetor que armazena as matrizes  $\mathbf{G}_{(2)}^{(k)}$  e reestruturá-lo em um novo *array* cujas entradas são as matrizes reorganizadas em um longo vetor. Com esta mudança, pode-se usar os métodos em sua forma vetorial, o que é bastante conveniente.

Para um tensor  $I \times I \times \dots \times I$  de ordem  $N$  e postos-TT iguais a  $r$ , a complexidade computacional é da ordem de  $I^N + I^{N-1}r^2$  [55]. Assim, o uso de técnicas como *Ket Augmentation* pode aumentar o tempo de execução do método consideravelmente. O Algoritmo 9 resume o método.

---

**Algoritmo 9 – TT-WOPT**


---

**Entrada:**  $\mathcal{Y} = \mathcal{P}_\Omega(\mathcal{T})$ , postos-TT  $r_1, \dots, r_{N-1}$ , inicialização dos núcleos-TT  $\mathcal{G}^{(1)}, \dots, \mathcal{G}^{(N)}$  e  $\varepsilon > 0$ .

$k = 0$

**enquanto**  $\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F > \varepsilon$  **faça**

**para**  $n = 1 : N$  **faça**

| Compute a  $n$ -ésima entrada do gradiente de acordo com (2.25).

**fim**

Atualize os núcleos  $\mathcal{G}^{(1)}, \dots, \mathcal{G}^{(N)}$  utilizando o método da máxima descida.

$k = k + 1$

**fim**

**retorna**  $\mathcal{G}^{(1)}, \dots, \mathcal{G}^{(N)}$ .

---

Como parâmetro, o posto-TT deve ser fixado para que as dimensões dos núcleos estejam bem definidas. Essa escolha é crucial para um bom desempenho do método, isto é, se as entradas do vetor forem pequenas, o tensor original será subestimado. Caso sejam muito grandes, a complexidade e a quantidade de parâmetros armazenados acabam inviabilizando o método. No entanto, em [55] foi escolhido o mesmo valor para todas as entradas do vetor de postos e não fica claro o porquê. Além desta abordagem ser pouco flexível quando modificamos a instância do problema, e.g., trocar o tensor e o número de entradas conhecidas, dificilmente todas as entradas do posto-TT serão iguais.

Para contornarmos a dificuldade mencionada anteriormente, com base nas ideias de Steinlechner em [46], propusemos uma atualização dinâmica do posto-TT junto ao método TT-WOPT. Primeiramente, inicializamos o TT-WOPT com posto-TT igual a  $[1, 1, \dots, 1]$ . Em seguida, incrementamos uma das entradas em uma unidade e executamos o método novamente. Se houve uma diminuição considerável do erro, seguimos com a mudança. Caso contrário, não realizamos o incremento, partimos para a próxima entrada e seguimos dessa maneira até atingirmos um valor máximo de posto.

Seja  $r_n$  a  $n$ -ésima entrada do posto-TT. Os núcleos-TT associados a esse valor

são  $\mathcal{G}^{(n)} \in \mathbb{R}^{r_{n-1} \times I_n \times r_n}$  e  $\mathcal{G}^{(n+1)} \in \mathbb{R}^{r_n \times I_{n+1} \times r_{n+1}}$ . Para incrementarmos  $r_n$  em uma unidade, consideramos os desdobramentos

$$\mathbf{G}_L^{(n)} \in \mathbb{R}^{r_{n-1} I_n \times r_n} \quad \text{e} \quad \mathbf{G}_R^{(n+1)} \in \mathbb{R}^{r_n \times I_{n+1} r_{n+1}}.$$

e realizamos as atribuições

$$\mathbf{G}_L^{(n)} \leftarrow (\mathbf{G}_L^{(n)} \quad \mathbf{w}_n) \quad \text{e} \quad \mathbf{G}_R^{(n+1)} \leftarrow \begin{pmatrix} \mathbf{G}_R^{(n+1)} \\ \mathbf{w}_{n+1}^T \end{pmatrix}, \quad (2.26)$$

onde  $\mathbf{w}_n \in \mathbb{R}^{r_{n-1} I_n}$  e  $\mathbf{w}_{n+1} \in \mathbb{R}^{r_{n+1} I_{n+1}}$  são vetores com entradas aleatórias retiradas de uma distribuição normal padrão e de magnitude pequena, por exemplo  $10^{-8}$ . Dessa forma, estes novos elementos pouco interferem na representação TT da solução corrente. Contudo, para mantermos a estabilidade numérica do método, ortogonalizamos os núcleos da esquerda para direita como definido no final da Seção 1.3.

O acréscimo das entradas do posto-TT deve ser feito a partir de algum critério que envolva o erro da aproximação. Considere

$$\Gamma \subset \{1, \dots, I_1\} \times \cdots \times \{1, \dots, I_N\}, \quad (2.27)$$

onde  $\Gamma \cap \Omega = \emptyset$ . Este novo conjunto de índices será utilizado para verificarmos a qualidade da aproximação em cada iteração. Defina também o erro

$$\varepsilon_\Gamma(\mathcal{X}) := \frac{\|\mathcal{P}_\Gamma(\mathcal{T}) - \mathcal{P}_\Gamma(\mathcal{X})\|_F}{\|\mathcal{P}_\Gamma(\mathcal{T})\|_F}.$$

Sejam  $\mathcal{X}^k$  a aproximação na iteração corrente e  $\bar{\mathcal{X}}$  a aproximação com o incremento no posto-TT. Vamos considerar que a atribuição  $\mathcal{X}^{k+1} = \bar{\mathcal{X}}$  não é feita levando em conta duas condições: quando

$$\frac{|\varepsilon_\Gamma(\mathcal{X}^k) - \varepsilon_\Gamma(\bar{\mathcal{X}})|}{\varepsilon_\Gamma(\mathcal{X}^k)} < \eta \quad (2.28)$$

for satisfeito para  $\eta > 0$ , ou quando

$$\varepsilon_\Gamma(\bar{\mathcal{X}}) - \varepsilon_\Gamma(\mathcal{X}^k) > \rho \varepsilon_\Gamma(\mathcal{X}^k) \quad (2.29)$$

for satisfeito para  $\rho > 0$ . Em (2.28), tem-se em vista o ganho na qualidade da solução e, se este não for suficiente, o incremento não é aceito. Após um certo número de iterações ou um decréscimo considerável da função objetivo, pode-se considerar refinar o parâmetro  $\eta$ . Já a condição (2.29), é utilizada em [46]. Neste caso, os passos que diminuem o erro, sempre são aceitos, por mais que o decréscimo seja pequeno. Todavia, a condição também aceita uma folga para passos que acarretam em aumento do erro, controlado pelo parâmetro  $\rho$ . No Algoritmo 10, exibimos o TT-WOPT com as mudanças propostas. Este novo método será denominado *Tensor Weighted Optimization with Dynamical Updating of TT-rank* (TT-WOPT-DU).

---

**Algoritmo 10 – TT-WOPT-DU**

---

**Entrada:**  $\mathcal{Y} = \mathcal{P}_\Omega(\mathcal{T})$ , postoTT =  $[1, 1, \dots, 1]$ ,  $\Gamma, \rho, \eta, \varepsilon_{DU} > 0$ ,  $r_{\max}$ .

Incialize os núcleos-TT  $\mathcal{G}^{(1)}, \dots, \mathcal{G}^{(N)}$ .

**para**  $k = 2, \dots, r_{\max}$  **faça**

**para**  $j = 1, \dots, N - 1$  **faça**

Se  $\varepsilon_\Gamma(\mathcal{X}^k) < \varepsilon_{DU}$ , **break**.

**se**  $r_j \geq r_{\max}$  **então**

| Avance para a próxima entrada do postoTT.

**senão**

| postoTT =  $[r_1, \dots, r_j + 1, \dots, r_{N-1}]$

| Atualize os núcleos-TT de acordo com (2.26) e inicie o TT-WOPT.

| Se (2.28) ou (2.29) for satisfeita, não aceite a atualização do posto-TT.

| Caso contrário, aceite o passo.

**fim**

**fim**

**fim**

**retorna**  $\mathcal{G}^{(1)}, \dots, \mathcal{G}^{(N)}$ .

---

Vale ressaltar que o número de sub-iterações feitas na otimização dos núcleos-TT pode ser reduzido alterando-se o número de iterações do TT-WOPT e, dessa maneira, evitar que tenhamos iterações muito longas. A execução é terminada quando  $\varepsilon_\Gamma(\mathcal{X}^k) < \varepsilon_{DU}$ . Existe uma sutileza do método com relação ao aumento dos postos-TT e a fatoração QR dos desdobramentos dos núcleos-TT. Para que o incremento seja possível, é necessário que os desdobramentos sejam matrizes cujo número de colunas seja menor ou igual ao número de linhas. Seja  $\mathbf{G}_L^{(n)} \in \mathbb{R}^{r_{n-1}I_n \times r_n + 1}$  o  $n$ -ésimo núcleo-TT após o incremento de  $r_n$ . Caso  $r_n + 1 > r_{n-1}I_n$ , o fator  $Q$  apresentará  $r_n$  colunas e o incremento do posto será perdido durante o processo de ortogonalização dos núcleos-TT. Neste sentido, não permitimos o aumento do posto-TT quando o método se depara com este obstáculo e seguimos para a próxima entrada do vetor.

# Capítulo 3

## Experimentos Numéricos

Neste capítulo, diversos experimentos com diferentes estruturas de dados foram realizados a fim de comparar os métodos exibidos anteriormente para o problema de completamento. As heurísticas e os métodos SiLRTC, HaLRTC, SiLRTC-TT, TMac-TT e TT-WOPT, foram implementados em *Julia* 1.7.2. É importante ressaltar que foi utilizada a biblioteca disponível em [41] para operarmos com tensores de forma eficiente.

Em cada experimento, geramos um tensor  $\mathcal{T}$  a partir de algum procedimento e  $\mathcal{W}$ , de mesmas dimensões, que serve de máscara. As entradas de  $\mathcal{W}$  admitem dois valores: 0, se a entrada é desconhecida, e 1, se for conhecida. Realizando o produto de Hadamard  $\mathcal{W} * \mathcal{T}$  (cf. Definição A.3), obtemos o tensor de entradas desconhecidas. Sendo  $mr$  (de *missing rate*), a taxa de dados faltantes e  $n$ , o número de elementos de  $\mathcal{T}$ , a quantidade de entradas desconhecidas será dada por  $n mr$ . O tensor  $\mathcal{W}$  foi construído da seguinte forma: consideramos o número de entradas do tensor dado por  $p = I_1 I_2 \cdots I_N$  e utilizamos a rotina `randperm` da linguagem *Julia* para realizar uma permutação aleatória de um vetor com entradas  $(1, 2, 3, \dots, p)$ . Definida a taxa de dados faltantes  $mr$ , tomamos as  $k$  primeiras entradas do vetor permutado, onde  $k$  é o inteiro mais próximo de  $(1 - mr)p$ . Estas entradas representam os índices linearizados do conjunto  $\Omega$  das entradas conhecidas. Utilizando a linearização dos índices de  $\mathcal{W}$ , fixamos o valor 1 nas entradas que estão em  $\Omega$ . Este procedimento simula a amostragem uniforme de  $p$  pontos sem reposição.

A princípio foram realizados quatro experimentos com diferentes estruturas para testar a capacidade dos métodos. Os tensores utilizados são especificados abaixo:

- **Experimento I.** Considere 5 tensores de ordem 3 de dimensões  $r \times 10 \times r$  com entradas retiradas de uma distribuição normal padrão. Estes tensores compõem os núcleos-TT de um tensor de dimensões  $10 \times 10 \times 10 \times 10 \times 10$  e de posto-TT com entradas iguais a  $r \in \{4, 8, 12, 16\}$ ;
- **Experimento II.** Geramos os fatores de uma decomposição HOSVD de um tensor de ordem 4 de dimensões  $30 \times 30 \times 30 \times 30$  e de posto multilinear com entradas

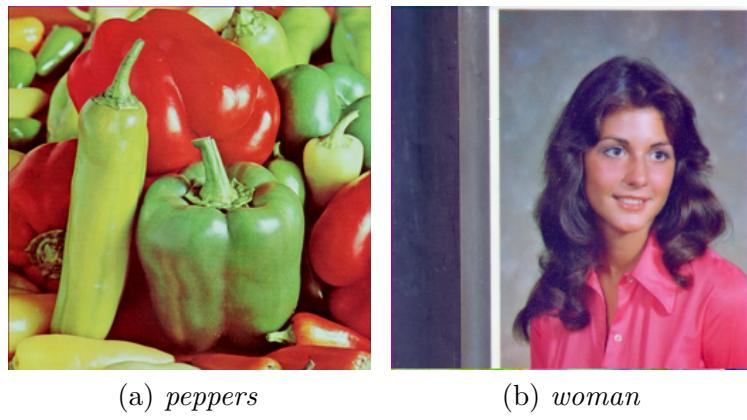


Figura 14 – Figuras utilizadas nos Experimentos III e IV.

iguais a  $r \in \{4, 8, 12, 16\}$ ;:

- **Experimento III.** Foram utilizadas duas imagens (*peppers* e *woman*<sup>1</sup>) com dimensões  $256 \times 256 \times 3$ .
- **Experimento IV.** Cada imagem do experimento anterior foi reestruturada, por *Ket Augmentation*, em um tensor de ordem 9 de dimensões  $4 \times 4 \times 3$ ;

Todos os métodos foram executados com as mesmas taxas  $mr \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$  e interrompemos quando a sequência de aproximações satisfez

$$\frac{\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F}{\|\mathcal{P}_\Omega(\mathcal{T})\|_F} < \varepsilon = 10^{-4}$$

ou quando foram atingidas 1000 iterações. Escolhas similares foram feitas em [2].

### 3.1 Detalhes de implementação e escolhas dos parâmetros

Nesta seção disponibilizamos maiores detalhes a respeito das implementações dos métodos e da escolha dos parâmetros. Todos os testes foram realizados em uma máquina pessoal com processador IntelCore i5-8250U com 8 núcleos 1.60GHz e 8Gb de memória RAM.

Os métodos baseados na minimização da norma nuclear dependem do vetor de pesos da combinação convexa  $\boldsymbol{\alpha}$  e de um vetor  $\boldsymbol{\beta}$  associado às penalizações quadráticas. Prosseguindo de maneira semelhante a [2], para os métodos que envolvem o desdobramento associado ao posto multilinear, escolhemos

$$\alpha_k = \frac{I_k}{\sum_{j=1}^N I_j}, \quad k = 1, \dots, N. \quad (3.1)$$

<sup>1</sup> Retiradas de <https://sipi.usc.edu/database/database.php>

Note que esta escolha favorece os modos de maiores dimensões, pois estes carregam maior informação acerca das entradas. No caso dos desdobramentos associados ao posto-TT, temos

$$\alpha_k = \frac{\delta_k}{\sum_{j=1}^{N-1} \delta_j}, \quad \text{com} \quad \delta_k = \min \left\{ \prod_{j=1}^k I_j, \prod_{j=k+1}^N I_j \right\}, \quad k = 1, \dots, N-1. \quad (3.2)$$

Neste caso, os modos cujos desdobramentos possuem um equilíbrio maior entre o número de linhas e colunas carregam os maiores pesos. Por fim, escolhemos  $\beta = \gamma \alpha$ , onde  $\gamma$  é uma constante positiva. Abaixo, comentamos sobre as escolhas para cada método:

- Heurísticas: A partir da aproximação inicial, os fatores da decomposição PARAFAC são calculados pelo método de Quadrados Mínimos Alternados (ALS). Estes são inicializados aleatoriamente a partir de uma distribuição uniforme entre 0 e 1 e paramos quando a distância entre a aproximação corrente e a anterior ficou que  $10^{-4}$  ou foram atingidas 1000 iterações. No caso da decomposição HOSVD, prosseguimos como no Algoritmo 2.

Como parâmetros, devemos escolher o posto da aproximação. No caso da PARAFAC, devido ao mau condicionamento, não é possível saber a priori, um bom valor de posto. Então, com alguns testes, selecionamos valores tais que a complexidade e o número de parâmetros armazenados não fossem altos. Neste sentido, seguimos de maneira análoga com a escolha do posto multilinear. É preciso ser cauteloso, pois a maldição da dimensionalidade dificulta o uso do método para tensores de ordens mais altas.

- SiLRTC e SiLRTC-TT (Algoritmos 5 e 7): Os dois métodos possuem como parâmetros os vetores  $\alpha$  e  $\beta$  mencionados anteriormente. O primeiro leva em conta a noção de posto multilinear e o segundo explora a noção de posto-TT.
- HaLRTC (Algoritmo 6): O vetor de pesos da combinação convexa é obtido pela relação (3.1). A função objetivo leva em conta o termo de penalização do Lagrangiano aumentado. Como em [35], escolhemos valores pequenos para parâmetro  $\beta$ , como  $10^{-3}$  e  $10^{-4}$ .
- TMac-TT (Algoritmo 8): A inicialização do método se dá pela aproximação inicial  $\mathcal{X}^0 = \mathcal{P}_\Omega(\mathcal{T})$  e também uma aproximação do posto-TT para definirmos as matrizes  $\mathbf{U}^{(n)0}$  e  $\mathbf{V}^{(n)0}$ , para  $n = 1, \dots, N-1$ , cujas dimensões dependem deste vetor. Em um primeiro momento, o posto-TT é estimado considerando o posto dos desdobramentos  $\mathbf{X}_{\langle n \rangle}^0$  para  $n = 1, \dots, N-1$ . No entanto, estamos considerando uma aproximação de baixo posto. Neste sentido, dado um parâmetro  $\lambda \in (0, 1)$  e a aproximação vigente do posto-TT, denotada por  $\mathbf{r}$ , escolhemos apenas os valores singulares que satisfazem

$$\frac{\sigma_j^{(n)}}{\sigma_1^{(n)}} > \lambda, \quad j = 1, \dots, r_n, \quad n = 1, \dots, N-1, \quad (3.3)$$

onde  $\sigma_j^{(n)}$  é o  $j$ -ésimo valor singular do  $n$ -ésimo desdobramento da aproximação corrente. Dessa forma, procuramos levar em consideração apenas as informações mais importantes na reconstrução e os desdobramentos com baixo posto apresentarão mais valores singulares truncados.

Todavia, percebeu-se que esta aproximação inicial não seria suficiente para obtermos boas reconstruções. Neste sentido, seguindo as ideias apresentadas em [54], é possível incrementar as entradas do posto-TT e, então, aumentar as dimensões das matrizes  $\mathbf{U}^{(n)}$  e  $\mathbf{V}^{(n)}$ . Este procedimento foi feito quando verificamos estagnação do método. Com maior liberdade para representar os desdobramentos do tensor  $\mathcal{X}$ , espera-se que a qualidade da aproximação melhore. Neste sentido, considere a relação:

$$\left| 1 - \frac{\|\mathcal{X}^{k+1} - \mathcal{X}^k\|_F}{\|\mathcal{X}^k - \mathcal{X}^{k-1}\|_F} \right| < 0.1.$$

Quando válida, incrementamos em  $\Delta r$  todas as entradas do posto-TT até que se atinja um limitante  $r_{\max}$ . Após alguns testes preliminares, escolhemos  $\Delta r = 3$  e  $r_{\max} = 30$ . Por fim, para realizar a atualização (2.22), escolhemos os pesos de acordo com (3.2).

- TT-WOPT (Algoritmo 9) A solução é encontrada a partir de um método de otimização que leva em conta a direção do vetor gradiente. Assim, para que fôssemos capazes de reproduzir os resultados de [55], utilizamos um *solver* de otimização em Julia chamado `Optim.jl`<sup>2</sup>. Neste caso, escolhemos o método de gradientes conjugados com busca de Moré-Thuente para ajuste do tamanho de passo [37], com os mesmos parâmetros utilizados por [55]. Os núcleos-TT foram aproximados inicialmente com entradas retiradas de uma distribuição normal padrão e então normalizadas para controlar a estabilidade numérica.

Dada a solução produzida por um método, as métricas utilizadas para comparar sua qualidade foram o *Root Squared Error* (RSE) e *Peak Signal-to-Noise ratio* (PSNR), dadas por

$$\text{RSE}(\mathcal{X}) := \frac{\|\mathcal{X} - \mathcal{T}\|_F}{\|\mathcal{T}\|_F} \quad \text{e} \quad \text{PSNR}(\mathcal{X}) := 10 \log_{10} \left( \frac{\prod_{j=1}^N I_j}{\|\mathcal{T} - \mathcal{X}\|_F^2} \right).$$

Esta última é comumente utilizadas em aplicações que envolvem processamento de imagens, pois é uma expressão para a razão entre o maior valor do sinal e o maior ruído que distorce a qualidade da imagem. Neste sentido, o objetivo é conseguir valores maiores de PSNR e menores de RSE. Nos experimentos que envolvem a reconstrução de imagens, os pixels exteriores ao intervalo  $[0, 1]$  eventualmente obtidos, foram projetados em  $[0, 1]$  para que fosse possível exibi-las.

<sup>2</sup> <https://juliansolvers.github.io/Optim.jl/stable/>

Por fim, nos gráficos que exibiremos adiante, cada método será identificado pela legenda da Figura 15.

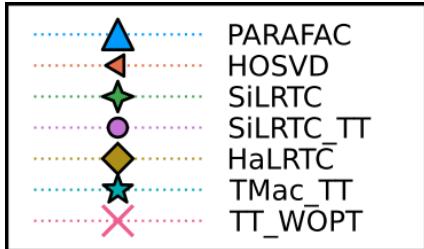


Figura 15 – Legenda que acompanha os resultados dos experimentos.

## 3.2 Resultados

### Experimento I

A ideia do primeiro experimento é reconstruir um tensor cuja estrutura é baseada na representação TT. Neste sentido, espera-se que os métodos baseados na noção de posto-TT tenham melhor desempenho quando comparados com aqueles baseados no posto multilinear ou posto PARAFAC. Os parâmetros são exibidos a seguir:

- PARAFAC: posto = 24;
- HOSVD: posto multilinear = [4, 4, 4, 4, 4];
- SiLRTC:  $\alpha = [0.2, 0.2, 0.2, 0.2, 0.2]$ ,  $\beta = 0.1\alpha$ ;
- HaLRTC:  $\alpha = [0.2, 0.2, 0.2, 0.2, 0.2]$ ,  $\beta = 0.01$ ;
- SiLRTC-TT:  $\alpha = [0.0454, 04545, 0.4545, 0.0454]$ ,  $\beta = 0.1\alpha$ ;
- TMac-TT:  $\alpha = [0.0454, 04545, 0.4545, 0.0454]$ ,  $\lambda = 0.41$ ;
- TT-WOPT: posto-TT =  $[r + 1, r + 1, r + 1, r + 1]$ .

Pelos resultados exibidos na Figura 16 percebe-se que os métodos baseados na noção de posto-TT apresentaram melhor desempenho quando comparado aos demais. Podemos entender que se o tensor possuir alguma representação TT de baixo posto, métodos como TT-WOPT podem ser utilizados para realizar o completamento. Com o aumento do posto e, consequentemente, o aumento da quantidade de entradas que precisam ser conhecidas, espera-se que os métodos tenham maior dificuldade para inferir as entradas corretamente. No caso do problema com 90% de entradas faltantes, observa-se grande dificuldade em encontrar uma boa solução, o que é esperado tendo em vista a dificuldade intrínseca do problema.

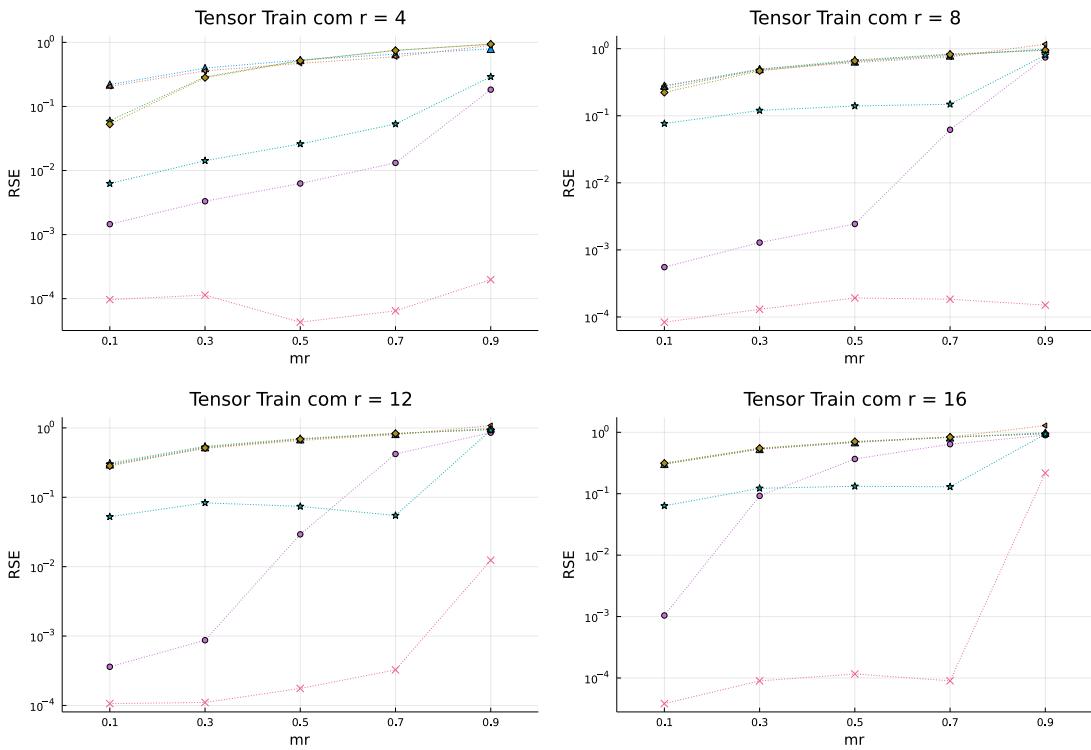


Figura 16 – Resultados para RSE obtidos pelo Experimento I (legenda na Figura 15).

## Experimento II

Trocando os papéis do Experimento I, estamos agora interessados em reconstruir um tensor baseado na estrutura do posto-multilinear. Abaixo apresentamos os parâmetros:

- PARAFAC: posto = 4;
- HOSVD: posto multilinear =  $[r + 1, r + 1, r + 1, r + 1]$ ;
- SiLRTC:  $\alpha = [0.25, 0.25, 0.25, 0.25]$ ,  $\beta = 0.01\alpha$ ;
- HaLRTC:  $\alpha = [0.25, 0.25, 0.25, 0.25]$ ,  $\beta = 0.001$ ;
- SiLRTC-TT:  $\alpha = [0.03125, 0.9375, 0.03125]$ ,  $\beta = 0.01\alpha$ ;
- TMac-TT:  $\alpha = [0.03125, 0.9375, 0.03125]$ ,  $\lambda = 0.415$ ;
- TT-WOPT: posto-TT = [12, 48, 12];

Conforme resultados exibidos na Figura 17, em um primeiro momento, observe que os métodos baseados no posto-multilinear, como HOSVD e HaLRTC, apresentaram desempenho melhor em comparação ao Experimento I. No entanto, para  $r = 4$ , alguns métodos baseados no posto-TT conseguiram capturar a correlação entre as entradas. Vale notar o caso do TT-WOPT que conseguiu obter uma boa aproximação mesmo para o problema com 90% de entradas desconhecidas e também o caso do TMac-TT, que conseguiu

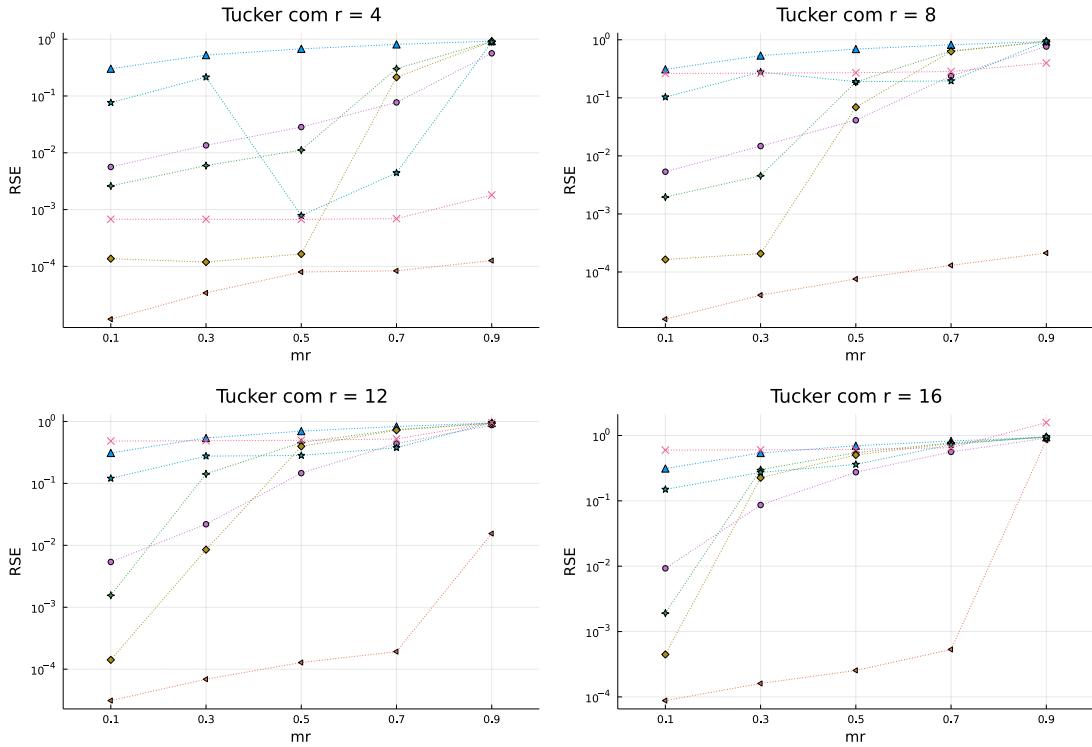


Figura 17 – Resultados para RSE obtidos pelo Experimento II (legenda na Figura 15).

encontrar boas aproximações para as taxas de 50% e 70%, e perdeu-se no problema com taxa igual a 30%.

### Experimento III

Aqui tratamos as imagens da Figura 14 como tensores de ordem 3. Neste caso, o posto-TT possui apenas duas entradas, de modo que existe pouca liberdade para os métodos baseados neste minimizarem a função objetivo. Os parâmetros são dados a seguir:

- PARAFAC: posto = 24
- HOSVD: posto multilinear = [24, 24, 3]
- SiLRTC:  $\alpha = [0.25, 0.25, 0.25, 0.25]$ ,  $\beta = 0.01\alpha$
- HaLRTC:  $\alpha = [0.25, 0.25, 0.25, 0.25]$ ,  $\beta = 0.001$
- SiLRTC-TT:  $\alpha = [0.9884, 0.0116]$ ,  $\beta = 0.01\alpha$
- TMac-TT:  $\alpha = [0.9884, 0.0116]$ ,  $\lambda = 0.415$
- TT-WOPT: posto-TT = [24, 3]

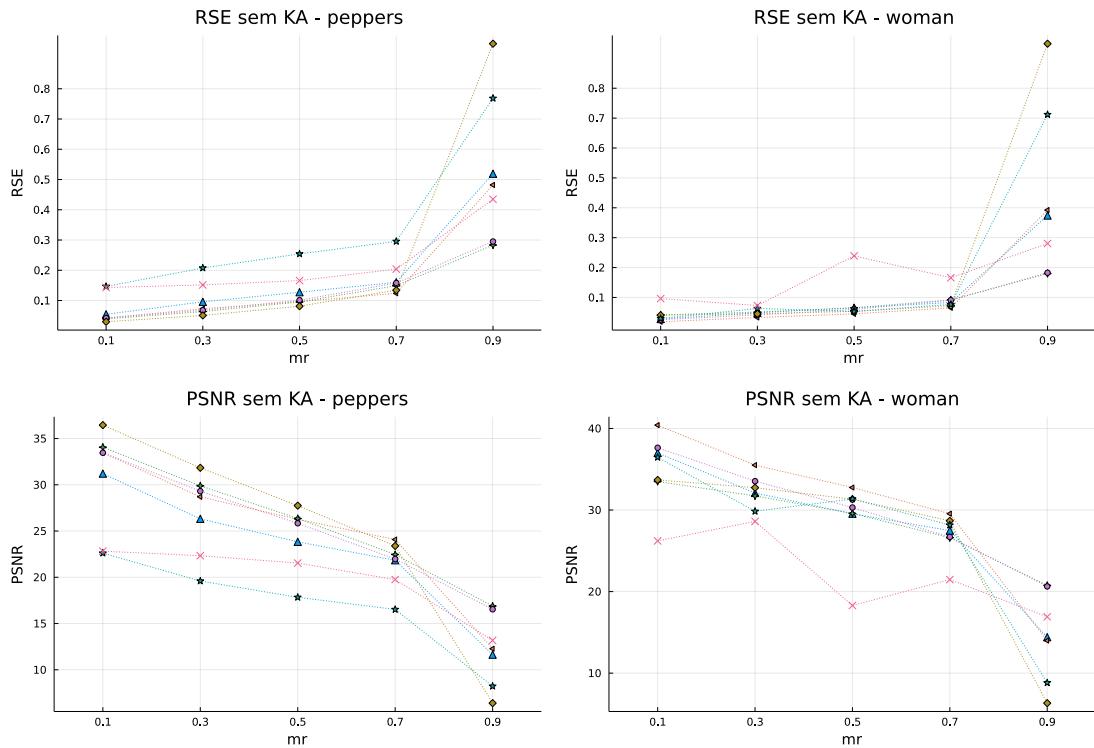


Figura 18 – Resultados para RSE e PSNR obtidos pelo Experimento III (legenda na Figura 15).

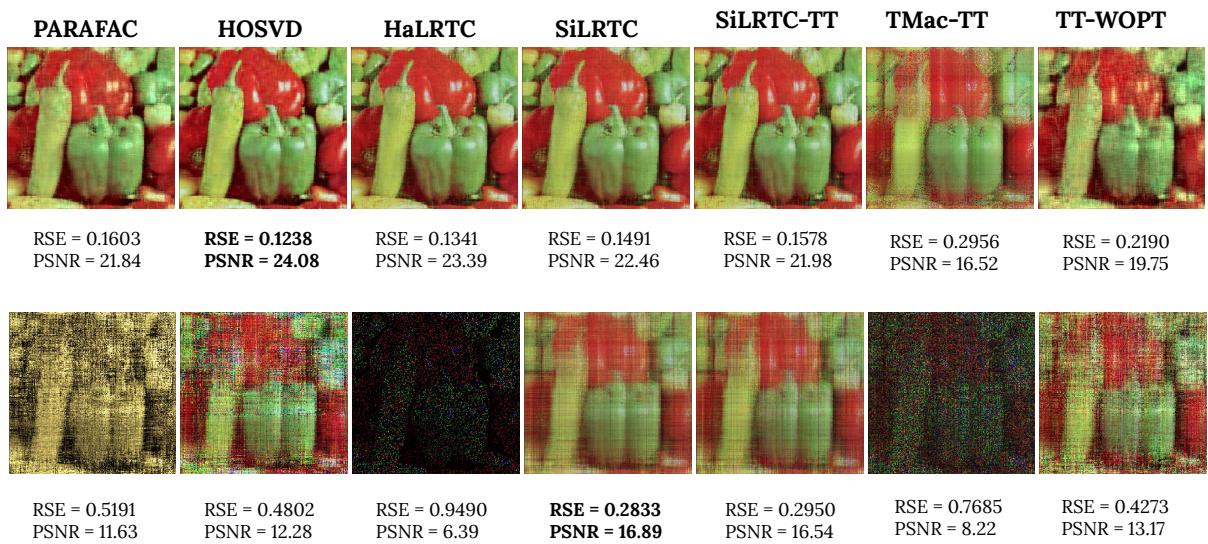


Figura 19 – Resultados para a imagem *peppers* sem *Ket Augmentation*. As soluções para o problema com 70% e 90% são exibidas na primeira e segunda linha, respectivamente.

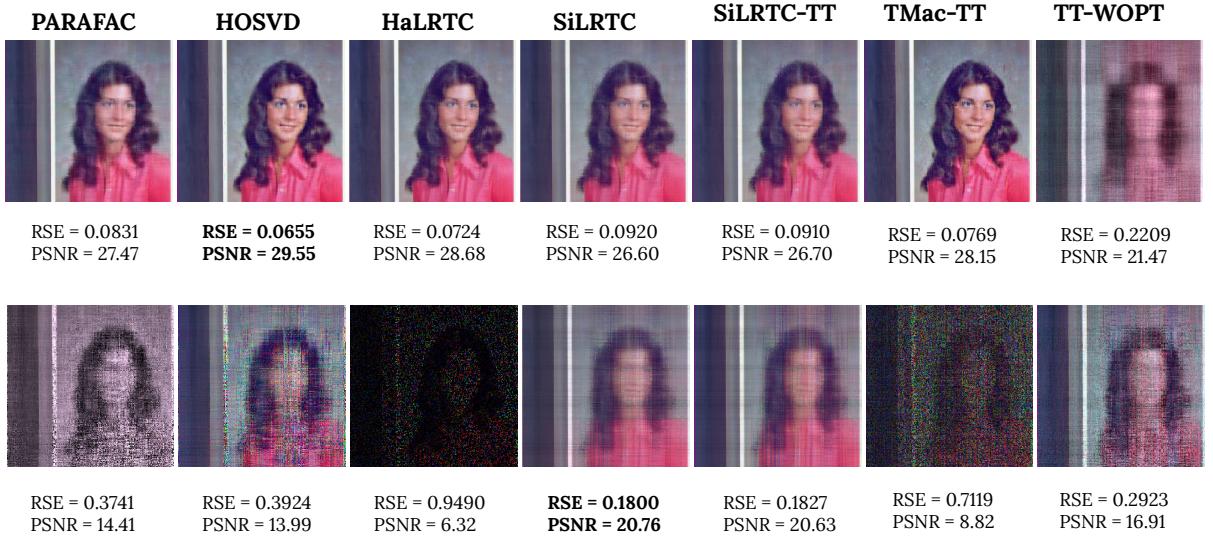


Figura 20 – Resultados para a imagem *woman* sem *Ket Augmentation*. As soluções para o problema com 70% e 90% são exibidas na primeira e segunda linha, respectivamente.

Na Figura 18 exibimos os resultados referentes às duas medidas de qualidades adotadas, e podemos perceber claramente que, à medida que  $mr$  cresce e a dificuldade do problema aumenta, o desempenho dos métodos piora.

Observa-se pelas Figuras 19 e 20 que os métodos baseados na noção do posto-multilinear apresentaram as melhores reconstruções. No entanto, mesmo com 70% de entradas faltantes, já é possível visualizar alguns detalhes das imagens.

## Experimento IV

Neste experimento, as imagens foram reestruturadas em tensores de ordem 9 pelo processo de *Ket Augmentation*. Utilizamos os mesmos tipos de figuras do Experimento III. Abaixo disponibilizamos os parâmetros.

- PARAFAC: posto = 24;
- HOSVD: posto multilinear = [4, 4, 4, 4, 4, 4, 4, 4, 3];
- SiLRTC:  $\alpha = [0.1143, 0.1143, \dots, 0.1143, 0.085]$ ,  $\beta = 0.01\alpha$ ;
- HaLRTC:  $\alpha = [0.1143, 0.1143, \dots, 0.1143, 0.085]$ ,  $\beta = 0.001$ ;
- SiLRTC-TT:  

$$\alpha = [0.0067, 0.0269, 0.1076, 0.4303, 0.3227, 0.0806, 0.0201, 0.0054], \beta = 0.01\alpha;$$

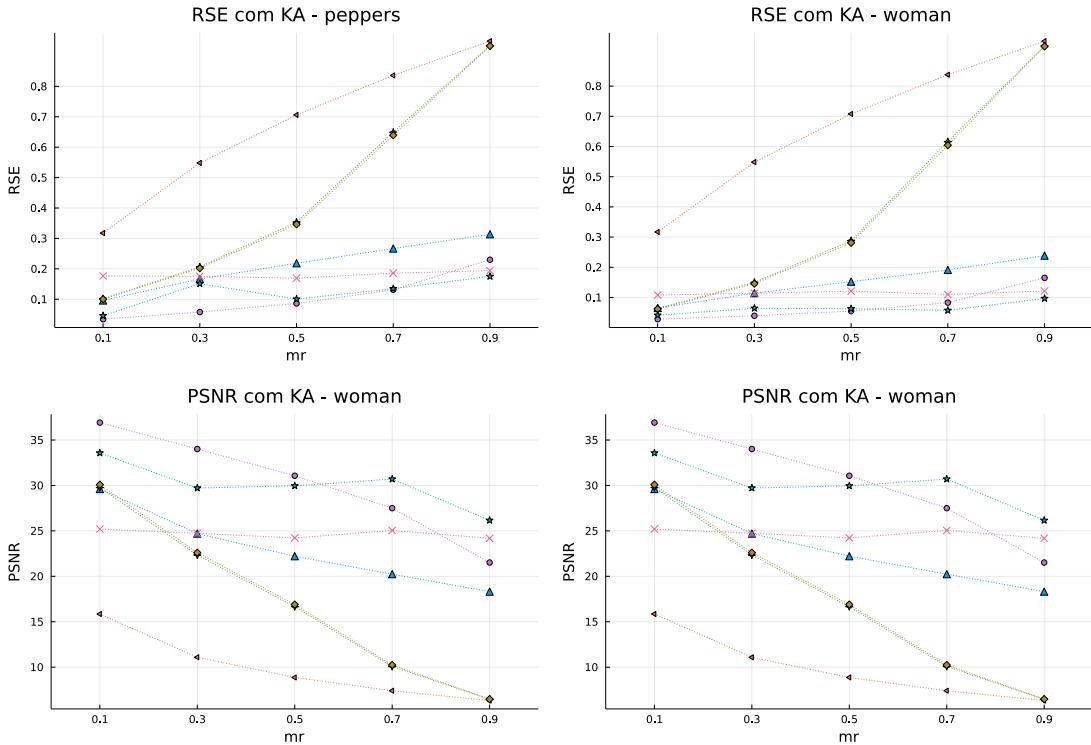


Figura 21 – Resultados para RSE e PSNR obtidos pelo Experimento IV (legenda na Figura 15).

- TMac-TT:

$$\boldsymbol{\alpha} = [0.0067, 0.0269, 0.1076, 0.4303, 0.3227, 0.0806, 0.0201, 0.0054], \lambda = 0.415;$$

- TT-WOPT: posto-TT = [16, 16, 16, 16, 16, 16, 16, 16, 16];

Conforme podemos acompanhar nas Figuras 21, 22 e 23, o aumento da ordem do tensor, com a manutenção do número de entradas, mostrou-se favorável aos métodos baseados no posto-TT, indicando que os métodos como TMac-TT e TT-WOPT têm melhor desempenho quando a instância envolve tensores de ordens mais altas. Além disso, a qualidade da melhor aproximação neste experimento é superior àquela do Experimento III, conforme acompanhamos os destaque em negrito das Figuras 19 e 22 e das Figuras 20 e 23 respectivamente.

### 3.3 Experimentos envolvendo TT-WOPT-DU

Com a implementação do método TT-WOPT com atualização dinâmica das entradas do posto-TT (TT-WOPT-DU) baseada no Algoritmo 10, foi possível realizar alguns experimentos cujos resultados forneceram informações interessantes acerca do método original. As escolhas sobre os parâmetros do TT-WOPT foram especificadas na Seção 3.1. Apenas alteramos o número de iterações realizadas pelo TT-WOPT para 500.

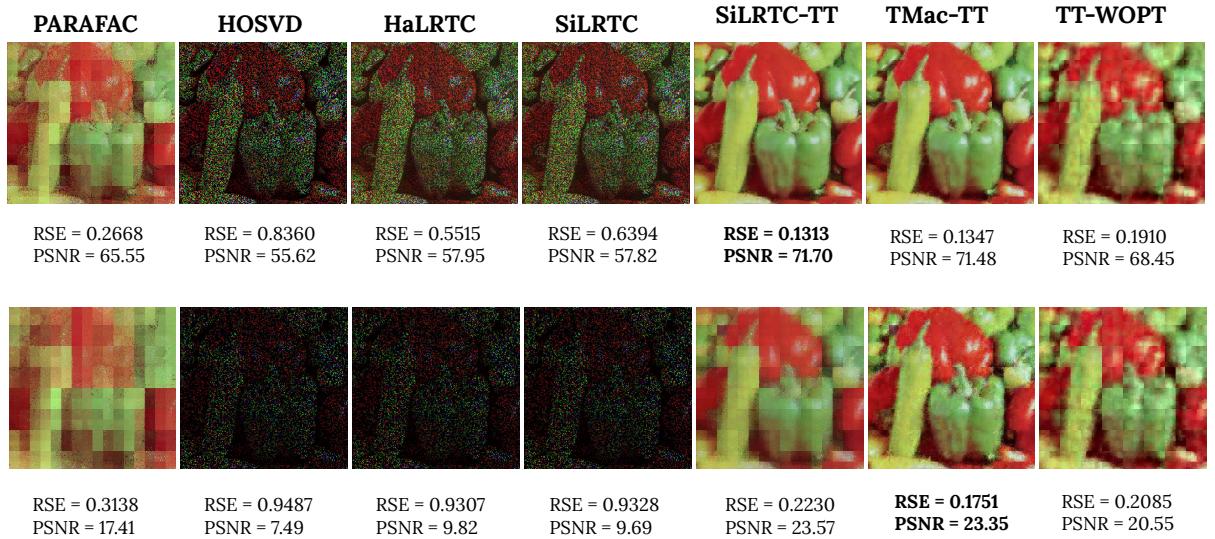


Figura 22 – Resultados para a imagem *peppers* com *Ket Augmentation*. As soluções para o problema com 70% e 90% são exibidas na primeira e segunda linha, respectivamente.

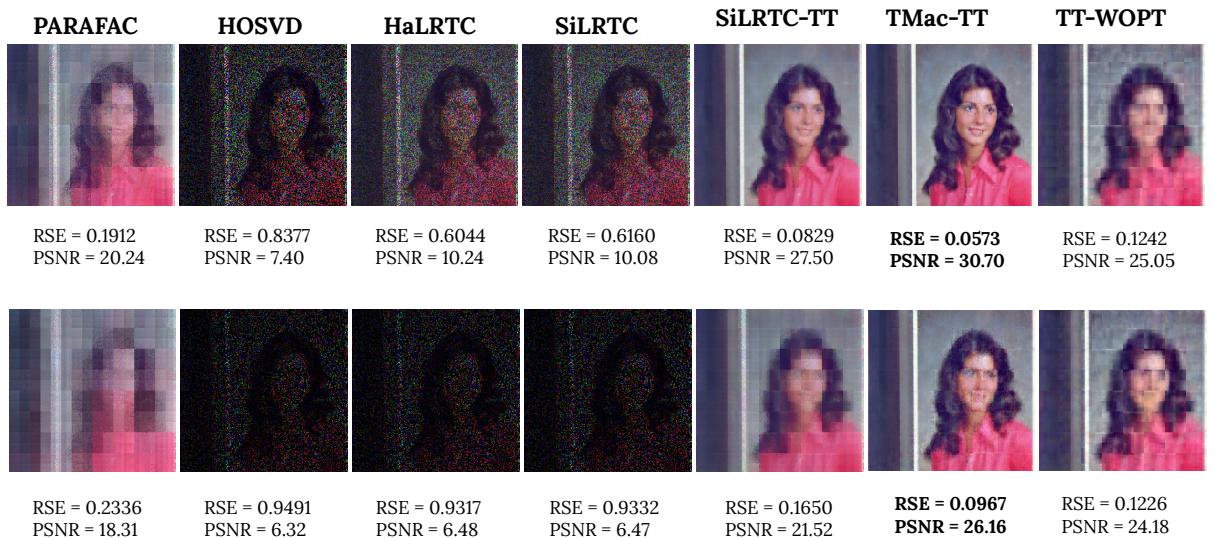


Figura 23 – Resultados para a imagem *peppers* com *Ket Augmentation*. As soluções para o problema com 70% e 90% são exibidas na primeira e segunda linha, respectivamente.

Tabela 5 – RSE das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional  $f$  amostrado em 4096 pontos.

$\rho = 1.0$				$\rho = 0.1$				$\rho = 0.01$			
mr	ordem_3	ordem_4	ordem_6	ordem_3	ordem_4	ordem_6	ordem_3	ordem_4	ordem_6	ordem_4	ordem_6
0.1	0.09030	<b>0.00004</b>	0.00146	0.05491	<b>0.00007</b>	0.00230	0.00494	<b>0.00008</b>	0.01100		
0.3	1.46619	<b>0.62920</b>	0.89462	0.50830	<b>0.00079</b>	0.08173	0.10300	<b>0.00020</b>	0.05739		
0.5	0.88062	<b>0.00110</b>	1.51688	0.35293	<b>0.00102</b>	0.00496	0.01031	<b>0.00084</b>	0.00604		
0.7	5.34928	<b>2.13870</b>	18.13957	0.10883	<b>0.00523</b>	0.64843	0.12279	<b>0.00882</b>	0.01600		
0.9	<b>6.67965</b>	33.60201	33.47548	1.05646	<b>1.01153</b>	1.34155	1.04246	<b>0.62289</b>	0.64392		

O conjunto de índices  $\Gamma$  (cf. (2.27)) foi escolhido da mesma maneira que o conjunto  $\Omega$ , mas com 20% de entradas conhecidas e tomamos  $\varepsilon_{DU} = 10^{-6}$ .

Realizamos o completamento de um tensor gerado a partir de sinais de diferentes dimensões. A ideia foi verificar, com o uso do método TT-WOPT-DU, que o posto-TT de funções trigonométricas, exponenciais e trigonométricas é baixo, como mostrado em [28]. Também verificamos o comportamento do método em instâncias envolvendo imagens e fizemos as comparações. Em todos os casos, apresentamos os resultados obtidos com a condição (2.29) e escolhas distintas do parâmetro de aceite  $\rho$ .

### Experimento com dados sintéticos - sinal unidimensional

Considere a função  $f : [-4\pi, 4\pi] \rightarrow \mathbb{R}$  dada por  $f(x) = \sin(x/4) \cos(x^2)$ . Amostramos em 4096 pontos igualmente espaçados do domínio, obtendo um *array* unidimensional. A ideia do experimento é reestruturar este vetor em tensores de diferentes ordens, mas mantendo o mesmo número de entradas. De acordo com [28], este tipo de tensorização de funções elementares possui posto-TT baixo. Verificamos este fato levando em conta três tensorizações com dimensões  $16 \times 16 \times 16$  (ordem 3),  $8 \times 8 \times 8 \times 8$  (ordem 4) e  $4 \times 4 \times 4 \times 4 \times 4 \times 4$  (ordem 6). Consideramos  $\rho \in \{1.0, 0.01, 0.001\}$ .

Pela Tabela 5, em um primeiro momento, nota-se que as instâncias de ordem 4 apresentaram os menores erros relativos, indicando que métodos baseados na noção de posto-TT não são favorecidos por instâncias de ordem 3, onde as outras noções de posto podem ser empregadas. Por fim, a reconstrução com 90% de dados faltantes não foi possível, indicando que, neste caso, devemos aumentar a ordem do tensor, aplicando  $f$  em mais pontos.

Com relação ao parâmetro  $\rho$ , observa-se que à medida que a taxa de entradas desconhecidas aumenta, este deve ser menor. Este fato é observado nas três formas de tensorização. Todavia, nota-se que a escolha  $\rho = 1$  já não produz boas aproximações como em [46]. Em um contexto em que não há dados em abundância ou quando o posto-TT não é tão pequeno, o método TT-WOPT-DU necessita de folgas menores, diferentemente do que foi proposto por Steinlechner. Neste caso, dentro do escopo da otimização Riemanniana, processos complexos de projeção e retração são executados para que o método esteja dentro

Tabela 6 – Posto-TT das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional  $f$  amostrado em 4096 pontos com 30% e 70% de dados faltantes.

mr = 0.3				mr = 0.7			
$\rho$	ordem_3	ordem_4	ordem_6	ordem_3	ordem_4	ordem_6	
1.0	[12, 15]	[8, 15, 13]	[4, 15, 15, 16, 15]	[15, 15]	[8, 14, 16]	[4, 15, 14, 16, 15]	
0.1	[8, 9]	[8, 8, 9]	[4, 11, 10, 12, 10]	[3, 14]	[8, 10, 16]	[4, 9, 5, 7, 11]	
0.01	[4, 11]	[8, 12, 16]	[4, 9, 9, 14, 9]	[2, 12]	[5, 6, 9]	[4, 6, 6, 16, 14]	

da variedade. Além disso, o tamanho de passo é ajustado por uma busca linear exata, de forma que o método proposto no artigo pode se utilizar de uma boa folga com  $\rho = 1$ . No caso do método TT-WOPT-DU, dependemos apenas da direção do gradiente e de um tamanho de passo. Assim, é necessário restringir o parâmetro  $\rho$ . Dessa forma, a escolha  $\rho = 0.01$  encontrou os melhores resultados. Considerando a taxa de entradas desconhecidas igual a 70%, na Figura 24 ilustramos os resultados obtidos com as reestruturações de ordem 4. Os postos-TT obtidos estão resumidos na Tabela 6. Observe que as entradas do vetor de postos não são iguais, diferentemente do que foi proposto em [55]. Nota-se que as instâncias onde existe uma grande quantidade de pontos conhecidos, o posto-TT encontrado é maior. Quando o método foi capaz de encontrar as entradas do vetor de postos associadas aos desdobramentos com os maiores valores singulares, obtemos os menores erros relativos. Isso foi possível, pois os incrementos desnecessários foram identificados pelo parâmetro  $\rho$ .

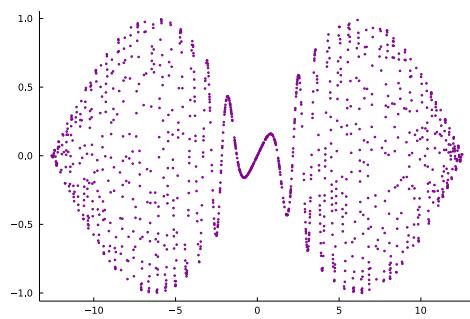
Para efeitos de comparação, consideramos amostrar o sinal gerado pela função  $f$  em 531441 pontos no mesmo intervalo. Trabalhamos com tensorizações de dimensões  $81 \times 81 \times 81$  (ordem 3),  $27 \times 27 \times 27 \times 27$  (ordem 4) e  $9 \times 9 \times 9 \times 9 \times 9 \times 9$  (ordem 6). Aqui, escolhemos  $r_{\max} = 8$ . Agora, com um número maior de entradas conhecidas, esperamos obter melhores aproximações durante a reconstrução do tensor. De acordo com a Tabela 7, as reconstruções encontradas apresentaram baixos valores de RSE, indicando que o conhecimento de um número maior de entradas é favorável ao completamento. Todavia, a abundância de pontos permite que os passos diminuam o erro com maior frequência, aumentando o posto-TT, conforme pode ser acompanhado na Tabela 8. Além disso, as aproximações são qualitativamente semelhantes, como podemos observar pela pouca variação dos erros relativos, de forma que é a melhor aproximação que o método pode encontrar com o limitante  $r_{\max}$  escolhido. Neste caso, para a tensorização de ordem 3, é suficiente para que a reconstrução seja de boa qualidade. Neste sentido, a maioria dos valores de posto-TT foram iguais a  $r_{\max} = 8$ .

Tabela 7 – RSE das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional  $f$  amostrado em 531441 pontos.

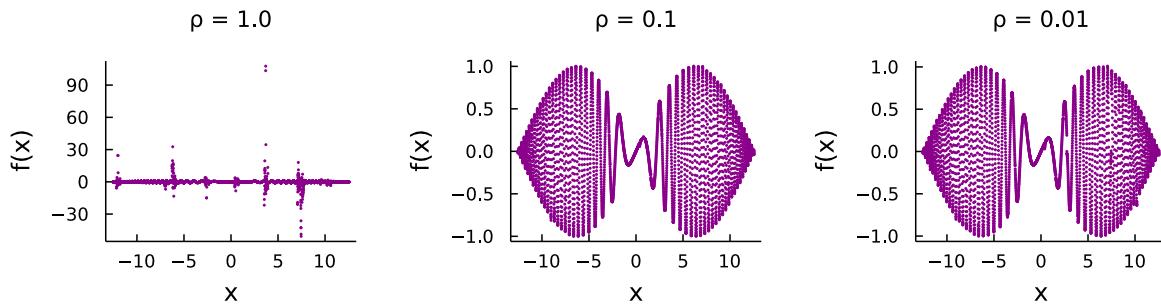
$\rho = 1.0$				$\rho = 0.1$			$\rho = 0.01$		
mr	ordem_3	ordem_4	ordem_6	ordem_3	ordem_4	ordem_6	ordem_3	ordem_4	ordem_6
0.1	<b>0.00019</b>	0.05866	0.07460	<b>0.00082</b>	0.06200	0.07377	<b>0.00018</b>	0.08137	0.07380
0.3	<b>0.00019</b>	0.08285	0.07010	<b>0.00018</b>	0.05971	0.09692	<b>0.00019</b>	0.05866	0.07227
0.5	<b>0.00019</b>	0.06097	0.07145	<b>0.00078</b>	0.05989	0.07288	<b>0.00018</b>	0.06389	0.07570
0.7	<b>0.00019</b>	0.07965	0.07886	<b>0.00019</b>	0.06447	0.07653	<b>0.00019</b>	0.06275	0.07482
0.9	<b>0.00084</b>	0.08167	0.09512	<b>0.00063</b>	0.11311	0.07635	<b>0.00104</b>	0.10245	0.07506

Tabela 8 – Posto-TT das aproximações encontradas pelo TT-WOPT-DU para a tensorização do sinal unidimensional  $f$  amostrado em 531441 pontos com 70% e 90% de dados faltantes.

$mr = 0.7$				$mr = 0.9$			
$\rho$	ordem_3	ordem_4	ordem_6	ordem_3	ordem_4	ordem_6	
1.0	[8, 8]	[8, 8, 8]	[8, 8, 8, 8, 8]	[8, 8]	[8, 8, 8]	[8, 8, 8, 8, 8]	
0.1	[8, 8]	[8, 8, 8]	[8, 8, 8, 6, 8]	[8, 8]	[7, 7, 8]	[6, 5, 5, 5, 8]	
0.01	[8, 8]	[6, 6, 8]	[8, 7, 7, 5, 8]	[7, 7]	[4, 4, 8]	[7, 7, 7, 4, 8]	



(a) Sinal unidimensional com 70% de entradas faltantes (amostrado em 4096 pontos).



(b) Reconstruções a partir da tensorização de ordem 4.

Figura 24 – Reconstrução do sinal unidimensional com TT-WOPT-DU.

Tabela 9 – Resultados para reconstrução do sinal bidimensional gerado por  $z_1$ .

	$\rho = 1.0$	$\rho = 0.1$	$\rho = 0.01$
RSE	$1.4 \cdot 10^{-8}$	$4.7 \cdot 10^{-7}$	$2.6 \cdot 10^{-7}$
posto-TT	[3, 3, 2, 2, 2]	[2, 2, 2, 2, 2]	[2, 2, 2, 2, 2]
tempo (s)	17.9	1.0	0.9

Tabela 10 – Resultados para reconstrução do sinal bidimensional gerado por  $z_2$ .

	$\rho = 1.0$	$\rho = 0.1$	$\rho = 0.01$
RSE	$3.5 \cdot 10^{-2}$	$2.4 \cdot 10^{-2}$	$2.1 \cdot 10^{-2}$
posto-TT	[4, 8, 8, 8, 8]	[4, 6, 7, 8, 8]	[4, 6, 6, 7, 6]
tempo (s)	40.4	22.7	16.7

### Experimento com dados sintéticos - sinal bidimensional

Neste experimento, vamos mostrar que um sinal gerado por uma função em  $\mathbb{R}^2$  também pode ser recuperado por meio do completamento. Assim como fizemos anteriormente, desejamos mostrar que tensorizações de funções tradicionais possuem posto-TT baixo. Dessa forma, podemos representá-las com poucos pontos.

Primeiramente, vamos considerar a função  $z_1 : [-\pi, \pi]^2 \rightarrow \mathbb{R}$  dada por  $z_1(x_1, x_2) = \cos(3x_1 + 3x_2)$ . Vamos amostrá-la em uma malha de  $64 \times 64 = 4096$  pontos igualmente espaçados no domínio obtendo um *array* bidimensional ou matriz. Esta matriz será mapeada em um tensor de ordem 6 de dimensões  $4 \times 4 \times 4 \times 4 \times 4 \times 4$  por *Ket Augmentation*. Escolhemos  $r_{\max} = 16$  e taxa de dados faltantes de 0.9. Os resultados são exibidos na Figura 25 e na Tabela 9. Pode-se perceber que o método foi capaz de capturar a estrutura de baixo posto. Ainda, devido ao tempo de execução, notamos que o parâmetro  $\rho = 1.0$  não foi a escolha mais eficiente para o método.

Seja  $z_2 : [-\pi, \pi]^2 \rightarrow \mathbb{R}$  dada por  $z_2(x_1, x_2) = 0.2x_1^2 - 0.1x_2^2 - \cos(2x_2) + \sin(3x_1)$ . Prosseguimos de maneira análoga ao que foi feito anteriormente. Os resultados são apresentados na Figura 26 e na Tabela 10. De acordo com [28], a soma de funções que geram tensores de posto-TT baixo, produz um tensor cujo posto-TT é limitado superiormente. Neste sentido, fomos capazes de reconstruir o tensor a partir de poucos pontos conhecidos. Novamente, podemos verificar que o parâmetro  $\rho = 0.01$  produziu os melhores resultados.

O método TT-WOPT-DU é inicializado com a escolha de um valor máximo para as entradas do posto-TT. Dessa maneira, em um contexto de posto-TT baixo, como os casos mostrados nessa seção, não é necessário escolher  $r_{\max}$  alto. Por vezes, altos valores deste parâmetro podem aumentar o tempo de execução de forma desnecessária. Neste sentido, entendemos que a escolha de  $r_{\max}$  está totalmente atrelada ao poderio computacional do usuário e o conhecimento da estrutura de dados utilizada.

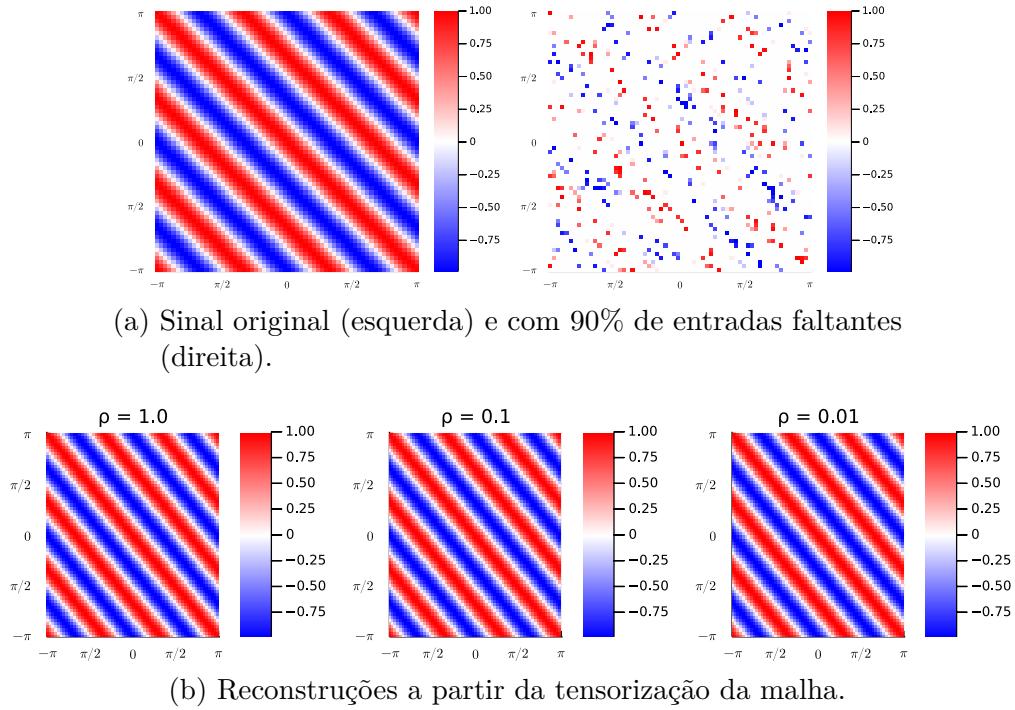
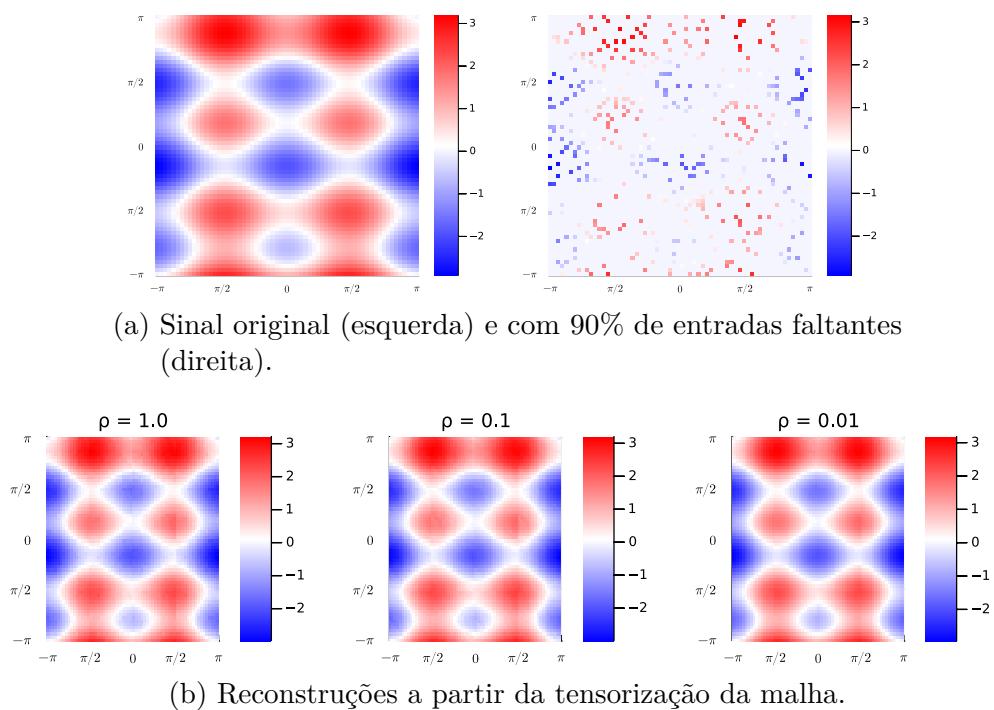
Figura 25 – Reconstrução do sinal gerado por  $z_1$  com TT-WOPT-DU.Figura 26 – Reconstrução do sinal gerado por  $z_2$  com TT-WOPT-DU.

Tabela 11 – Resultados para a reconstrução do tensor gerado pelo sinal multidimensional.

$\rho = 1.0$				$\rho = 0.1$				$\rho = 0.01$			
mr	RSE	posto-TT	tempo (s)	RSE	posto-TT	tempo (s)	RSE	posto-TT	tempo (s)		
0.90	$5.2 \cdot 10^{-4}$	[5, 5, 5]	110.1	$4.8 \cdot 10^{-4}$	[5, 5, 5]	97.1	$5.0 \cdot 10^{-4}$	[5, 5, 5]	90.1		
0.95	$1.5 \cdot 10^{-3}$	[5, 5, 5]	88.6	$1.5 \cdot 10^{-3}$	[4, 4, 4]	94.8	$1.7 \cdot 10^{-3}$	[4, 3, 3]	79.6		
0.99	$2.3 \cdot 10^0$	[5, 5, 5]	111.2	$1.0 \cdot 10^0$	[1, 1, 1]	91.3	$1.0 \cdot 10^0$	[1, 1, 1]	90.6		

### Experimento com dados sintéticos - sinal multidimensional

Com base em [46, Section 5.4.2], procuramos reconstruir um tensor de ordem 4 a partir da discretização de uma função multidimensional. Este experimento simula uma aplicação na área da sismologia quando consideramos dados sísmicos. Tais dados podem ser organizados em um tensor de ordem 5, onde duas dimensões são as coordenadas  $(x, y)$  das fontes do sinal (*source*), outras duas dimensões correspondem às coordenadas  $(x, y)$  dos receptores do sinal (*receiver*) e o tempo está representado na última dimensão. Dessa forma, fixando uma faixa de frequência após tomar a transformada de Fourier no tempo, obtemos uma fatia 4D do tensor (veja [11]). Reorganizando os dados, podemos exibir parte das informações em gráficos do tipo *source*  $\times$  *source* ou *receiver*  $\times$  *receiver* e considerar os valores de função para estes pontos. Nesse experimento simulado, assim como [46], trabalhamos com a função

$$f : [0, 1]^4 \rightarrow \mathbb{R}, \quad f(\mathbf{x}) = \exp(-\|\mathbf{x}\|_F),$$

e geramos uma malha de  $20^4 = 160000$  pontos igualmente espaçados sobre o domínio  $[0, 1]^4$ . Na Figura 27 é possível visualizar o sinal gerado por  $f$ . Também apresentamos o sinal com 95% dos pontos retirados uniformemente sem reposição juntamente com as respectivas curvas de nível e os pontos conhecidos. A reconstrução do tensor foi feita escolhendo posto-TT máximo  $r_{\max} = 5$ . Na Tabela 11 exibimos os resultados para taxas de 0.90, 0.95 e 0.99 de dados faltantes. Exibimos a aproximação gerada pelo método com  $\rho = 0.01$  na Figura 28. Observando os resultados, pode-se dizer que a reconstrução do sinal foi um sucesso. Todavia, não foi possível resolver a instância em que conhecemos 0.01% dos dados. Note que, assim como nos experimentos anteriores, a escolha  $\rho = 0.01$ , obteve ótimas aproximações com o menor posto-TT.

### Reconstrução de Imagens

Nos experimentos da Seção 3.1, percebeu-se que o método TT-WOPT encontrou dificuldades para reconstruir as imagens escolhidas. Agora com o incremento dinâmico do posto-TT, podemos procurar pelo melhor posto-TT para representar as imagens já reestruturadas por *Ket Augmentation*. Para isso, além das figuras *peppers* e *woman* (Figura 14), vamos considerar as imagens da Figura 29<sup>3</sup>. Fixamos a taxa de dados

<sup>3</sup> Retiradas de <https://sipi.usc.edu/database/database.php>

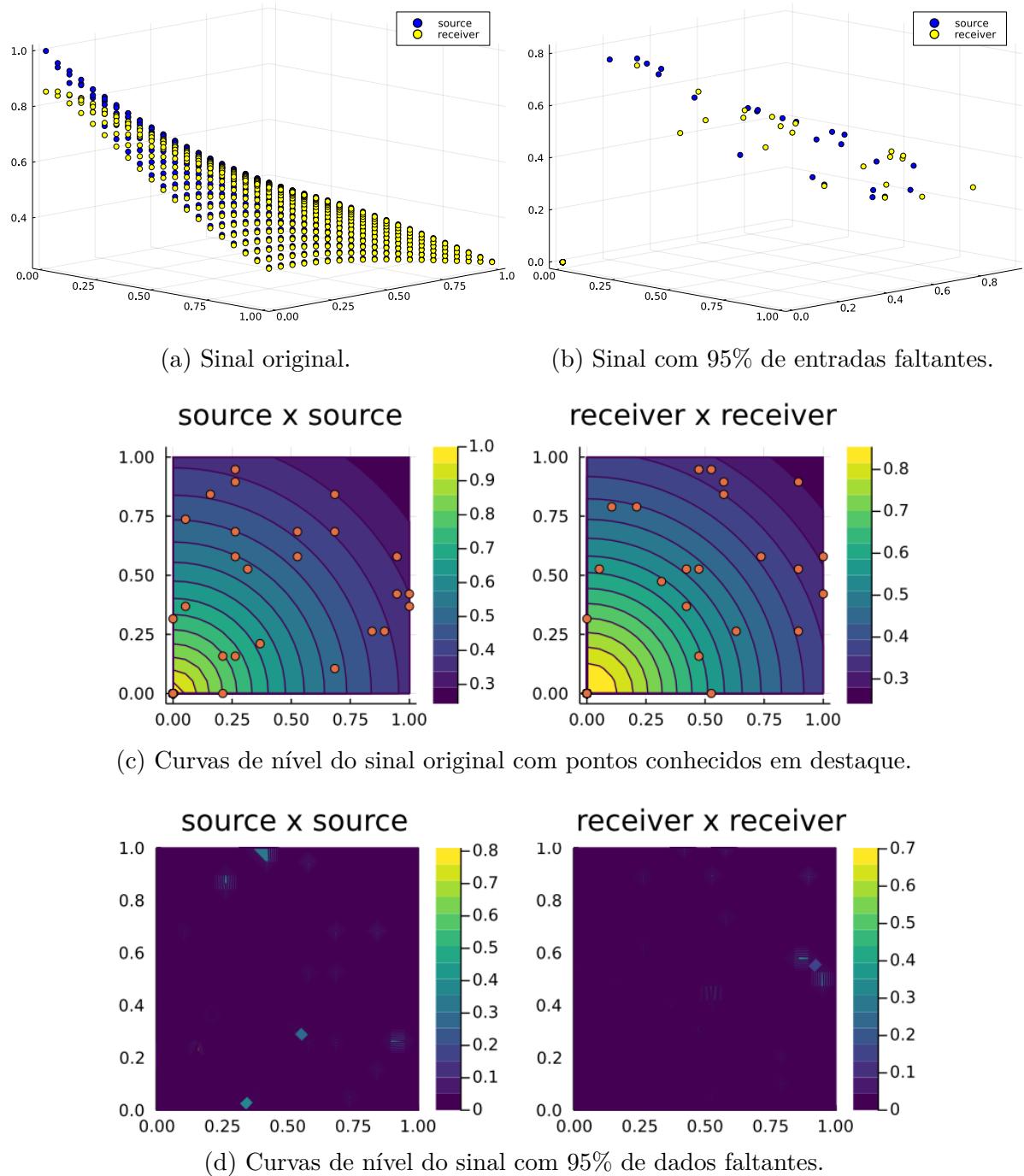
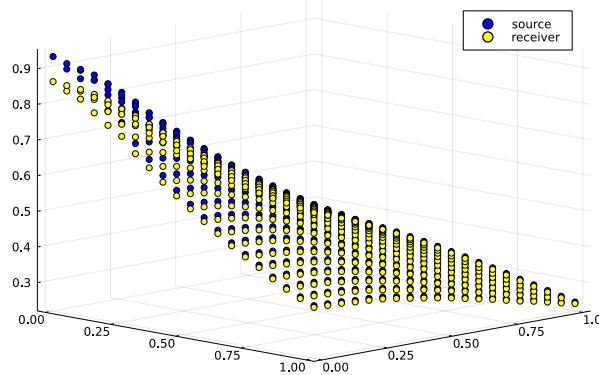
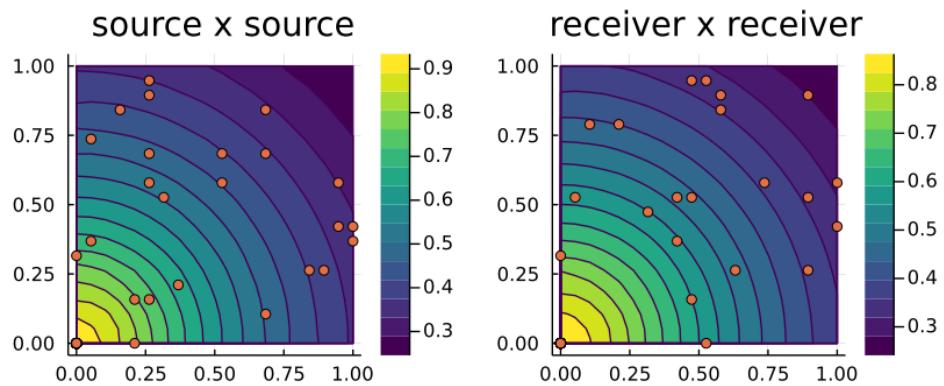


Figura 27 – Visualização das fatias geradas pelos dados sísmicos simulados.



(a) Reconstrução do sinal.



(b) Curvas de nível do sinal reconstruído com pontos conhecidos em destaque.

Figura 28 – Visualização da reconstrução produzida pelo TT-WOPT-DU com  $\rho = 0.01$ (a) *satellite*(b) *texture*

Figura 29 – Novas figuras a serem consideradas para o completamento.

faltantes igual a 50% e  $r_{\max} = 10$  para evitar tempo excessivo de execução devido à grande quantidade de pontos.

Observando os resultados de RSE e PSNR da Tabela 12, percebe-se que não houve melhora em relação aos valores do Experimento IV, que foram,  $RSE(peppers) = 0.16920$ ,  $PSNR(peppers) = 21.36976$ ,  $RSE(woman) = 0.12084$  e  $PSNR(woman) = 24.22259$ . Todavia, pela Figura 30, pouca diferença se verifica nas reconstruções anteriores das

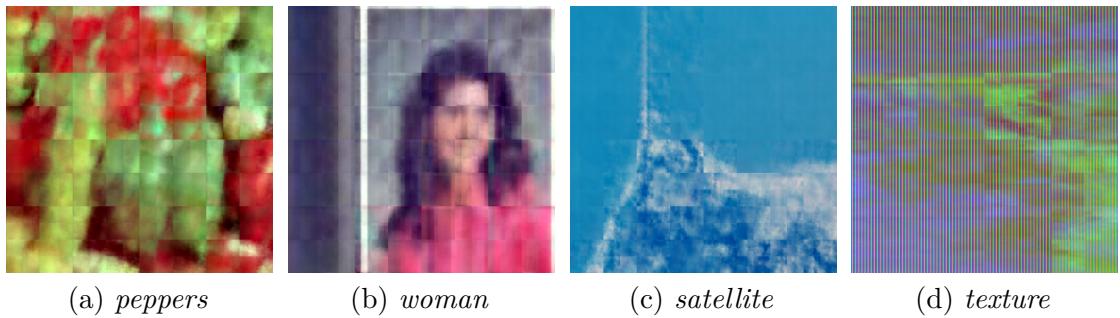


Figura 30 – Imagens reconstruídas pelo método TT-WOPT-DU.

Tabela 12 – Resultados obtidos pelo método TT-WOPT-DU com  $\rho = 0.01$  para reconstrução de imagens.

	peppers	woman	satellite	texture
RSE	0.19424	0.12876	0.12241	0.14539
PSNR	21.76363	23.67107	23.47025	22.22975
posto-TT	[4, 10, 10, 10, 10, 10, 10, 10]			
tempo (s)	13580	12009	12231	10288

Figuras 22 e 23, mesmo sendo obtidas com menos informações. Além disso, a reconstrução da imagem *satellite* apresenta um aspecto borrado, como já encontramos em outras reconstruções feitas pelo TT-WOPT. Já a imagem *texture*, não foi capaz de ser reconstruída, mesmo apresentando uma estrutura mais simples do que as outras. Neste sentido, nota-se a dificuldade do método TT-WOPT com instâncias envolvendo imagens. Pode-se justificar este fato pelo posto-TT das aproximações encontradas. De fato, todas as entradas do vetor de postos atingiram o valor  $r_{\max} = 10$ , indicando a necessidade em aumentar as dimensões dos núcleos-TT e, consequentemente, aumentar a complexidade computacional. A inviabilidade também pode ser observada pelo tempo utilizado pelo método. Todos estes fatos indicam que as imagens, após o processo de *Ket Augmentation*, não possuem posto-TT baixo. Tais dificuldades não foram encontradas na instância gerada pelas instâncias sintéticas dos experimentos anteriores. Os resultados para  $\rho = 1.0$  e  $\rho = 0.1$  foram semelhantes aos da Tabela 12, com diferenças pouco significativas.

Para buscar mais elementos para interpretar os resultados obtidos, faremos uma comparação entre os valores singulares de alguns dos desdobramentos dos tensores obtidos anteriormente. Assim, consideramos o tensor  $\mathcal{X}$  de ordem 6 e dimensões  $4 \times 4 \times 4 \times 4 \times 4 \times 4$ , gerado pela função  $f(x) = \sin(x/4) \cos(x^2)$ , bem como a imagem *satellite* com *Ket Augmentation* que denotaremos por  $\mathcal{Y}$ . Nas Figuras 31 e 32 exibimos os respectivos valores singulares em escala logarítmica.

Com base nas escalas verticais dos gráficos, podemos concluir que, de fato, os desdobramentos gerados pela função apresentam posto-TT baixo. No caso do desdobra-

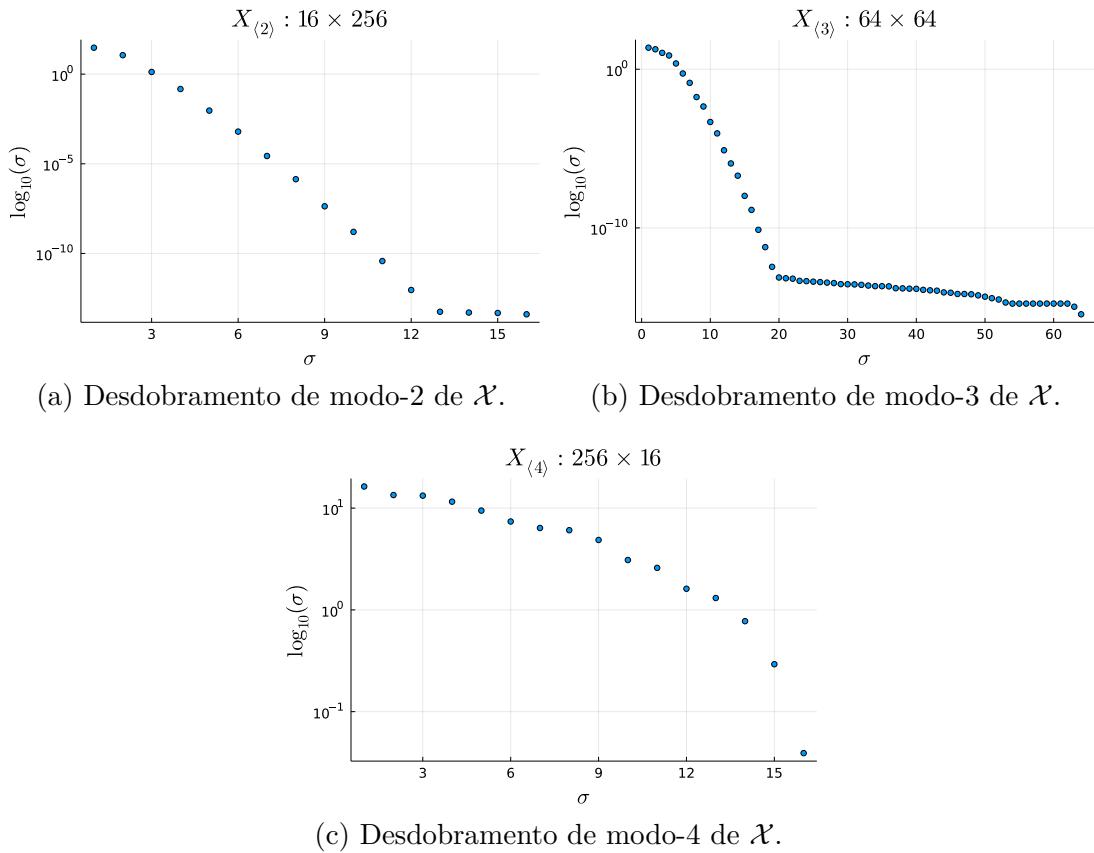


Figura 31 – Valores singulares dos desdobramentos de  $\mathcal{X}$  (função geradora do sinal unidimensional  $f$  com amostragem de 4096 pontos).

mento de modo-4, a informação necessária para representar o tensor está concentrada nos núcleos que compartilham desta entrada do posto. Por outro lado, também com base na escala vertical, verificamos que os desdobramentos da imagem *satellite* são todos de posto completo. Prosseguindo analogamente para as outras imagens, notamos o mesmo padrão de valores singulares dos desdobramentos. Neste sentido, o método TT-WOPT não é uma boa escolha para instâncias envolvendo imagens, pois estas apresentam posto-TT alto após o processo de *Ket Augmentation* e o desempenho do método está inteiramente ligado a este comportamento.

Nos casos em que não conhecemos exatamente os núcleos do tensor, como no caso da função  $\sin(x/4) \cos(x^2)$ , o método TT-WOPT-DU é uma ótima escolha para aproximarmos os núcleos-TT, sabendo da propriedade de baixo posto. Como a representação é a solução de um problema de completamento, pode-se amostrar uma função  $f$  em alguns pontos de uma malha e, então, obter a representação-TT de  $f$  na malha inteira. Esta é uma ideia parecida com o método TT-cross [38]. Em nosso caso, a aproximação é feita resolvendo um problema de otimização sobre os núcleos-TT enquanto o TT-cross realiza a decomposição esqueletal dos desdobramentos do tensor.

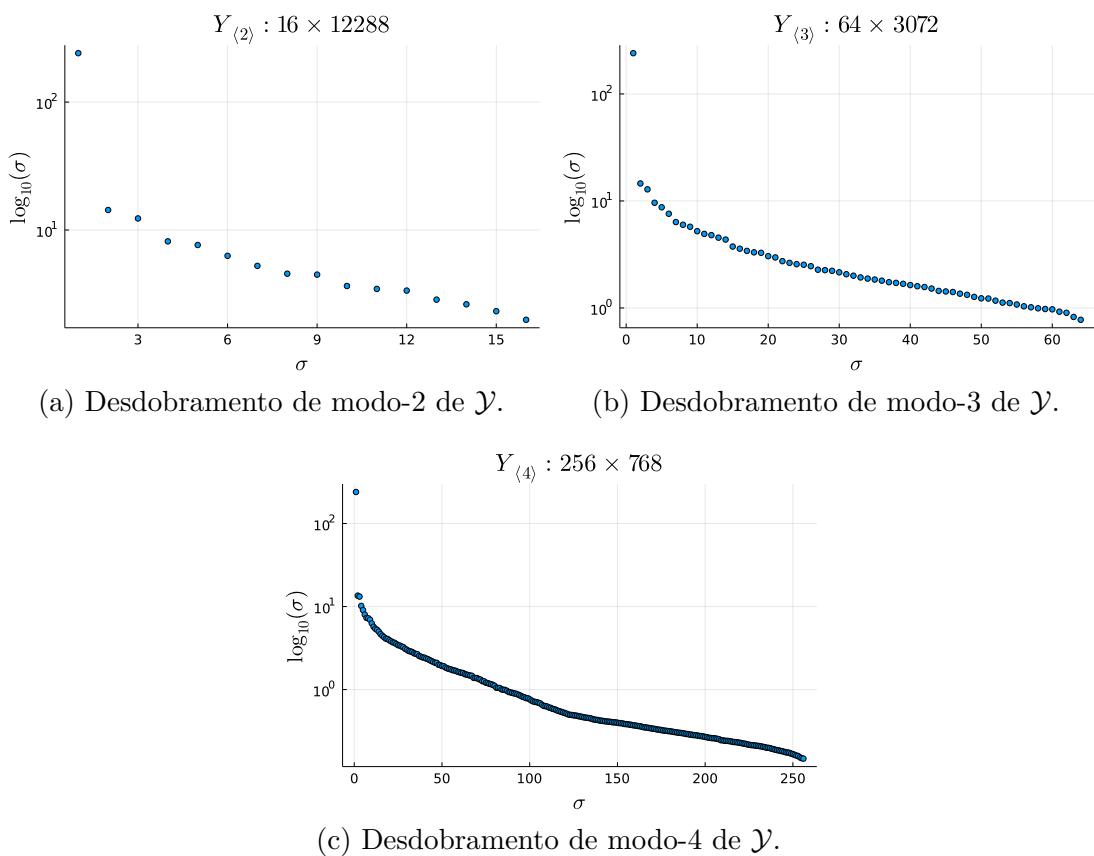


Figura 32 – Valores singulares dos desdobramentos de  $\mathcal{Y}$  (imagem *satellite*).

# Capítulo 4

## Considerações Finais

O intuito deste trabalho é introduzir os conceitos da álgebra linear computacional tensorial para aqueles que pouco conhecem sobre o assunto. Neste sentido, apresentamos as operações básicas da álgebra linear e as noções de posto clássicas que sempre permeiam os trabalhos sobre este tópico. Exibimos as primeiras noções de posto associados à decomposição em fatores paralelos (PARAFAC) e a SVD de alta ordem (HOSVD), bem como suas dificuldades. Ainda, apresentamos a decomposição *Tensor Train*: uma ferramenta potente, baseada na noção de posto-TT, capaz de representar grandes conjuntos de dados e de ordens mais altas com poucos parâmetros. Para que o leitor pudesse se familiarizar com as ideias e com as distintas noções de posto, apresentamos o problema de completamento tensorial junto a diversos métodos da literatura que visam solucioná-lo. Após a leitura dos exemplos e dos experimentos, esperamos que o leitor possa investir nos caminhos que apontamos na literatura, que envolvem sistemas lineares, cálculo de autovalores, resolução de equações diferenciais, completamento de estrutura de dados, representação de baixo-posto de operadores, entre outros. Além disso, os códigos utilizados para gerar os resultados do Capítulo 3 estão disponíveis em [https://github.com/Joaoluiz87/completamento\\_dissertacao\\_mestrado](https://github.com/Joaoluiz87/completamento_dissertacao_mestrado).

Para o problema de completamento, percebeu-se que existem poucos resultados teóricos acerca dos métodos e, principalmente, na escolha dos parâmetros. Verificamos que os métodos apresentados neste trabalho têm seu desempenho totalmente atrelado aos parâmetros. No caso de problemas com altas taxas de dados faltantes, o ajuste dos parâmetros deve ser extremamente fino e pode mudar quando modificamos a instância. Além da comparação entre diferentes algoritmos da literatura, parte de nossa atenção foi dada ao método *Tensor Train Weighted Optimization* (TT-WOPT), pela simplicidade de sua formulação bem como a possibilidade de utilizarmos o vetor gradiente da função objetivo. Contudo, é necessário definir, antes da inicialização, o posto-TT da aproximação, dificultando seu uso, pois geralmente não conhecemos esta informação. Motivados em solucionar esta dificuldade, propusemos uma maneira de atualizar dinamicamente as

entradas do posto-TT e o novo método foi denominado *Tensor Train Weighted Optimization with Dynamical Updating of TT-rank* (TT-WOPT-DU).

Os resultados mostraram que o método a ser escolhido para realizar o completamento depende da ordem do tensor considerado. Tensores de ordens mais baixas, como 3 ou 4, podem ter melhores reconstruções utilizando métodos baseados na noção de posto-multilinear. Para ordens superiores, métodos que se baseiam na noção de posto-TT, têm melhor desempenho. Técnicas que aumentem a ordem do tensor, mas que não modifiquem o número de entradas, podem ser empregadas para aprimorar o desempenho dos métodos baseados nesta noção de posto. Vimos que o remapeamento via *Ket Augmentation* melhorou consideravelmente a qualidade das soluções encontradas pelos métodos SiLRTC-TT, TMac-TT e TT-WOPT. Para este último, a escolha dos parâmetros não é trivial, como sugerido em seu artigo de criação.

Com relação ao método TT-WOPT-DU, diversos experimentos foram apresentados para atestarmos sua eficácia. Em um primeiro momento, mostramos como problemas de completamento unidimensionais e bidimensionais podem ser abordados pela filosofia do completamento tensorial. Em [28] é mostrado que tensores gerados por funções trigonométricas, exponenciais e polinomiais têm posto-TT baixo, de forma que este pode ser representado a partir de uma pequena amostra de pontos. Este fato foi verificado nos experimentos da Seção 3.3, onde fomos capazes de recuperar o sinal a partir de 10% de entradas conhecidas em diversos casos. Entretanto, considerando instâncias que envolvam imagens, como ilustrado na Figura 32, o método não é capaz de reconstruí-las com qualidade, pois seus desdobramentos apresentaram posto completo, após o mapeamento por *Ket Augmentation*. Dessa maneira, métodos que exploraram a noção de posto-TT de outras formas, como o método TMac-TT, podem ser mais eficientes neste contexto. Por fim, fomos capazes de mostrar que as entradas do posto-TT de um tensor nem sempre são iguais, hipótese de trabalho em [55].

O método TT-WOPT-DU foi baseado nas ideias propostas por Steinlechner em [46], de forma que o incremento dinâmico das entradas do posto-TT depende da escolha do parâmetro de aceite  $\rho$ . Além da escolha feita no artigo ( $\rho = 1$ ), os testes foram feitos considerando também  $\rho = 0.1$  e  $\rho = 0.01$ . Em todos os testes, a última prevaleceu sobre as outras em termos de produzir menores erros relativos, tempo de execução e acréscimos do posto-TT, diminuindo assim, o custo computacional. Como discutido na Seção 3.3, diversos passos complexos precisam ser executados no contexto de otimização Riemanniana, de maneira que a folga  $\rho = 1$  é suficiente para lidar com as dificuldades da otimização na variedade. Em nosso caso, foi preciso restringir esta folga, para que o método não degringolasse.

Vale ressaltar sobre as escolhas de parâmetros feitas para a execução do método TT-WOPT. Como mencionamos na Seção 2.2, o completamento é feito a partir de um

método de descida que depende do vetor gradiente. Os testes foram feitos utilizando o método de Gradientes Conjugados com busca linear de Moré-Thuente [37]. Além disso, para o TT-WOPT-DU, fixamos o número máximo de iterações em 500. Este número pode mudar durante a execução do método, sendo menor nas primeiras iterações e maior nas seguintes. Todas estas escolhas influenciam diretamente no tempo de execução do método, especialmente em um contexto de representação com posto-TT baixo.

Considerando trabalhos futuros, há interesse em melhorar a eficiência do método TT-WOPT-DU. Os resultados apresentados neste texto são frutos de uma primeira implementação e podem ser refinados. Neste trabalho, o aumento das entradas do posto-TT foi feito aumentando as dimensões dos núcleos-TT artificialmente, de maneira que o tensor gerado continuasse inalterado. Contudo, processos mais robustos podem ser estudados para este caso (veja, por exemplo, [15, §3.2]). Por mais que exista uma rica literatura em algoritmos TT de completamento para tensores e sua eficiência empírica, não há entendimento teórico para informações importantes como, quantidade de dados conhecidos, aproximação inicial e condições de convergência destes métodos, embora já existam certos resultados para os formatos PARAFAC e HOSVD. Em [46], a mesma função objetivo do método TT-WOPT é minimizada, mas sobre a variedade Riemanniana dos tensores de posto-TT fixo. Neste caso, garantias teóricas sobre inicialização e convergência foram fornecidas e espera-se que resultados semelhantes possam ser demonstrados para os métodos TT-WOPT ou TT-WOPT-DU.

Ademais, existe grande interesse de aproximação de tensores no contexto de equações diferenciais. Nestes casos, após o emprego de um esquema de discretização, a solução numérica geralmente é obtida por meio da resolução de um sistema linear. Neste sentido, a matriz, a solução e o vetor de fonte do sistema podem ser representados pelos seus núcleos-TT. A solução, portanto, é encontrada utilizando um esquema de quadrados mínimos alternados para cada um dos núcleos. Todavia, existem operadores e funções cuja representação TT e o posto-TT não são conhecidos. O método TT-WOPT-DU pode ser empregado neste contexto para obtermos uma representação com poucos parâmetros destes objetos. Veja [27] para uma aplicação destas ideias em um contexto de aplicação do método de elementos finitos.

# Referências

- [1] BECK, A. *First-Order Methods in Optimization*. Volume 25 of MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia(PA), 2017.
- [2] BENGUA, J. A., PHIEN, H. N., , TUAN, H. D., AND DO, M. N. Efficient tensor completion for color image and video recovery: Low-rank tensor train. *IEEE Transactions on Image Processing* 26 (2017), 2466–2479.
- [3] CABRAL, R., TORRE, F. D., COSTEIRA, J. P., AND BERNARDINO, A. Matrix completion for weakly-supervised multi-label image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37 (2015), 121–135.
- [4] CAI, J. F., CANDÈS, E. J., AND SHEN, Z. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization* 20 (2010), 1956–1982.
- [5] CANDÈS, E. J., AND RECHT, B. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics* 9 (2009), 717–772.
- [6] CANDÈS, E. J., AND TAO, T. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory* 56 (2010), 2053–2080.
- [7] CARROLL, J. D., AND CHANG, J. Analysis of individual differences in multidimensional scaling via an N-way generalization of “Eckart-Young” decomposition. *Psychometrika* 35 (1970), 283–319.
- [8] CICHOCKI, A., LEE, N., OSELEDETS, I., PHAN, A., ZHAO, Q., AND MANDIC, D. P. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions. *Foundations and Trends in Machine Learning* 9 (2016), 249–429.
- [9] CICHOCKI, A., MANDIC, D., DE LATHAUWER, L., ZHOU, G., ZHAO, Q., CAIAFA, C., AND PHAN, H. A. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine* 32, 2 (2015), 145–163.

- [10] DA COSTA, M. N., ATTUX, R., CICHOCKI, A., AND ROMANO, J. M. T. Tensor-train networks for learning predictive modeling of multidimensional data. Disponível em <https://arxiv.org/abs/2101.09184>., 2021.
- [11] DA SILVA, C., AND HERRMANN, F. J. Optimization on the Hierarchical Tucker manifold – applications to tensor completion. *Linear Algebra and its Applications* 481 (2015), 131–173.
- [12] DE LATHAUWER, L., DE MOOR, B., AND VANDEWALLE, J. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications* 21 (2000), 1253–1278.
- [13] DE LATHAUWER, L., DE MOOR, B., AND VANDEWALLE, J. On the best rank-1 and rank- $(r_1, r_2, \dots, r_n)$  approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications* 21 (2000), 1324–1342.
- [14] DE LATHAUWER, L., AND VANDEWALLE, J. Dimensionality reduction in higher-order signal processing and rank- $(r_1, r_2, \dots, r_n)$  reduction in multilinear algebra. *Linear Algebra and its Applications* 391 (2004), 31–55.
- [15] DOLGOV, S. V., AND SAVOSTYANOV, D. V. Alternating minimal energy methods for linear systems in higher dimensions. *SIAM Journal on Scientific Computing* 36, 5 (2014), A2248–A2271.
- [16] ECKART, C., AND YOUNG, G. The approximation of one matrix by another of lower rank. *Psychometrika* 1 (1936), 211–218.
- [17] FAZEL, M. Matrix Rank Minimization with Applications. PhD thesis, Stanford University, 2002.
- [18] FREDERIX, K., AND BAREL, M. V. Solving a large dense linear system by adaptive cross approximation. *Journal of Computational and Applied Mathematics* 234 (2010), 3181–3195.
- [19] GOLUB, G. H., AND LOAN, C. F. V. *Matrix Computations*, 4th ed. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2012.
- [20] GOREINOV, S., AND TYRITYSHNIKOV, E. The maximal-volume concept in approximation by low-rank matrices. *Contemporary Mathematics* 208 (01 2001), 47–52.
- [21] GOREINOV, S. A., OSELEDETS, I. V., SAVOSTYANOV, D. V., TYRITYSHNIKOV, E. E., AND ZAMARASHKIN, N. L. How to find a good submatrix. *Research Report 08-10, ICM HKBU, Kowloon Tong, Hong Kong* (2008), 08–10.

- [22] HACKBUSCH, W. *Tensor Spaces and Numerical Tensor Calculus*, 2nd edition ed. Springer Series in Computational Mathematics 56. Springer International Publishing, 2019.
- [23] HALLINAN, B.; STRIPHAS, T. Recommended for you: The Netflix Prize and the production of algorithmic culture. *New Media & Society* 18 (2016), 117–137.
- [24] HITCHCOCK, F. L. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics* 6 (1927), 164–189.
- [25] HU, Y., ZHANG, D., YE, J., LI, X., AND HE, X. Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2013), 2117–2130.
- [26] HÅSTAD, J. Tensor rank is NP-complete. *Journal of Algorithms* 11 (1990), 644–654.
- [27] KAZEEV, V., OSELEDETS, I., RAKHUBA, M., AND SCHWAB, C. QTT-finite-element approximation for multiscale problems I: model problems in one dimension. *Advances in Computational Mathematics* 43 (2017), 411–442.
- [28] KHOROMSKIJ, B. N. Tensor numerical methods for multidimensional PDEs: theoretical analysis and initial applications. *ESAIM: Proceedings and Surveys*. 48 (2015), 1–28.
- [29] KOLDA, T. G., AND BADER, B. W. Tensor decompositions and applications. *SIAM Review* 51 (2009), 455–500.
- [30] KRUSKAL, J. B. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications* 18 (1977), 95–138.
- [31] LATHAUWER, L. D. Signal Processing Based on Multilinear Algebra. PhD thesis, Katholieke Universiteit Leuven, 1997.
- [32] LATORRE, J. I. Image compression and entanglement. Disponível em <https://arxiv.org/abs/quant-ph/0510031>., 2005.
- [33] LEE, N., AND CICHOCKI, A. Fundamental tensor operations for large-scale data analysis using tensor network formats. *Multidimensional Systems and Signal Processing* 29 (2018), 921–960.
- [34] LI, X. P., HUANG, L., SO, H. C., AND ZHAO, B. A survey on matrix completion: Perspective of signal processing. Disponível em <https://arxiv.org/abs/1901.10885>., 2019.

- [35] LIU, J., MUSIALSKI, P., WONKA, P., AND YE, J. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2013), 208–220.
- [36] MIWAKEICHI, F., MONTES, E. M., SOSA, P. A. V., NISHIYAMA, N., MIZUHARA, H., AND YAMAGUCHI, Y. Decomposing EEG data into space-time-frequency components using parallel factor analysis. *NeuroImage* 22, 3 (2004), 1035–1045.
- [37] MORÉ, J. J.; THUENTE, D. J. Line search algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software* 20 (1994), 286–307.
- [38] OSELEDETS, I., AND TYRTYSHNIKOV, E. TT-cross approximation for multidimensional arrays. *Linear Algebra and its Applications* 432 (2010), 70–88.
- [39] OSELEDETS, I. V. Tensor-train decomposition. *SIAM Journal on Scientific Computing* 33 (2011), 2295–2317.
- [40] OSELEDETS, I. V., AND DOLGOV, S. V. Solution of linear systems and matrix inversion in the TT-format. *SIAM Journal on Scientific Computing* 34 (2012), A2718–A2739.
- [41] PERISA, L. Julia Tensor Toolbox. Disponível em <https://github.com/lanaperisa/TensorToolbox.jl>., 2021.
- [42] PETERSEN, K. B., AND PEDERSEN, M. S. The matrix cookbook. Disponível em <http://www2.compute.dtu.dk/pubdb/pubs/3274-full.html>., 2012.
- [43] RICHTER, L., SALLANDT, L., AND NÜSKEN, N. Solving high-dimensional parabolic PDEs using the tensor train format. Disponível em <https://arxiv.org/abs/2102.11830>., 2021.
- [44] ROVI, A. Analysis of  $2 \times 2 \times 2$  Tensor. Master’s thesis, Linkopings University, Sweden, 2010.
- [45] SMILDE, A., BRO, R., AND GELADI, P. *Multi-way Analysis: Applications in the Chemical Sciences*. Wiley, Chichester, England, 2004.
- [46] STEINLECHNER, M. Riemannian optimization for high-dimensional tensor completion. *SIAM Journal on Scientific Computing* 38, 5 (2016), S461–S484.
- [47] TEN BERGE, J. M. F. Kruskal’s polynomial for  $2 \times 2 \times 2$  arrays and a generalization to  $2 \times n \times n$  arrays. *Psychometrika* 56 (1991), 631–636.
- [48] TREFETHEN, L. N., AND BAU, D. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, 1997.

- [49] TUCKER, L., AND HARRIS, C. Implications of factor analysis of three way matrices for measurements of change. In *Problems in measuring change*. University of Wisconsin Press, Madison, 1963.
- [50] VASILESCU, M. A. O., AND TERZOPOULOS, D. Multilinear image analysis for facial recognition. *Object recognition supported by user interaction for service robots* 2 (2002), 511–514.
- [51] VYSOTSKY, L. I., SMIRNOV, A. V., AND TYRTYSHNIKOV, E. E. Tensor-train numerical integration of multivariate functions with singularities. *Lobachevskii Journal of Mathematics* 42, 7 (Jul 2021), 1608–1621.
- [52] WATSON, G. Characterization of the subdifferential of some matrix norms. *Linear Algebra and its Applications* 170 (1992), 33–45.
- [53] WEN, Z., YIN, W., AND ZHANG, Y. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation* 4 (2012), 333–361.
- [54] XU, Y., HAO, R., YIN, W., AND SU, Z. Parallel matrix factorization for low-rank tensor completion. *Inverse Problems and Imaging* 9, 2 (2015), 601–624.
- [55] YUAN, L., ZHAO, Q., GUI, L., AND CAO, J. High-order tensor completion via gradient-based optimization under tensor train format. *Signal Processing: Image Communication* (2019), 73:53–61.
- [56] ZNIYED, Y. *Breaking the curse of dimensionality based on tensor train: models and algorithms*. Thesis, Paris-Saclay, Oct. 2019.
- [57] CIVRIL, A., AND MAGDON-ISMAIL, M. On selecting a maximum volume submatrix of a matrix and related problems. *Theoretical Computer Science* 410 (2009), 4801–4811.

# APÊNDICE A

## Produtos Matriciais

**Definição A.1 (Produto de Kronecker)** O produto de Kronecker entre matrizes  $\mathbf{A} \in \mathbb{R}^{I \times J}$  e  $\mathbf{B} \in \mathbb{R}^{M \times N}$  é denotado por  $\mathbf{A} \otimes \mathbf{B}$ . O resultado desta operação é uma matriz de dimensões  $(IM) \times (JN)$  dada por

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \dots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \dots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \dots & a_{IJ}\mathbf{B} \end{pmatrix}.$$

**Definição A.2 (Produto de Khatri-Rao)** O produto de Khatri-Rao entre matrizes  $\mathbf{A} \in \mathbb{R}^{I \times K}$  e  $\mathbf{B} \in \mathbb{R}^{M \times K}$  é denotado por  $\mathbf{A} \odot \mathbf{B}$ . O resultado desta operação é uma matriz de dimensões  $(IM) \times K$  dada por

$$\mathbf{A} \odot \mathbf{B} = (\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \dots \quad \mathbf{a}_K \otimes \mathbf{b}_K).$$

**Definição A.3 (Produto de Hadamard)** O produto de Hadamard entre matrizes  $\mathbf{A} \in \mathbb{R}^{I \times J}$  e  $\mathbf{B} \in \mathbb{R}^{I \times J}$  é denotado por  $\mathbf{A} * \mathbf{B}$ . O resultado desta operação é uma matriz de dimensões  $I \times J$  dada por

$$\mathbf{A} * \mathbf{B} = \begin{pmatrix} a_{11}b_{11} & a_{12}b_{12} & \dots & a_{1J}b_{1J} \\ a_{21}b_{21} & a_{22}b_{22} & \dots & a_{2J}b_{2J} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}b_{I1} & a_{I2}b_{I2} & \dots & a_{IJ}b_{IJ} \end{pmatrix}.$$

Propriedades básicas dos produtos matriciais são exibidas abaixo. Mais informações podem ser encontradas em [19, Section 12.3].

1.  $(\mathbf{A} \otimes \mathbf{B})^T = \mathbf{A}^T \otimes \mathbf{B}^T$
2.  $(\mathbf{A} \otimes \mathbf{B})^\dagger = \mathbf{A}^\dagger \otimes \mathbf{B}^\dagger$

3.  $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$
4.  $\mathbf{A}(\mathbf{B} \otimes \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B})\mathbf{C}$
5.  $\text{posto}(\mathbf{A} \otimes \mathbf{B}) = \text{posto}(\mathbf{A}) \cdot \text{posto}(\mathbf{B})$
6.  $\det(\mathbf{A} \otimes \mathbf{B}) = \det(\mathbf{A})^I \cdot \det(\mathbf{B})^J \quad \mathbf{A} \in \mathbb{R}^{I \times I}, \mathbf{B} \in \mathbb{R}^{J \times J}$
7.  $\text{tr}(\mathbf{A} \otimes \mathbf{B}) = \text{tr}(\mathbf{A}) \cdot \text{tr}(\mathbf{B})$
8.  $\|\mathbf{A} \otimes \mathbf{B}\|_F = \|\mathbf{A}\|_F \cdot \|\mathbf{B}\|_F$
9.  $\|\mathbf{A} \otimes \mathbf{B}\|_2 = \|\mathbf{A}\|_2 \cdot \|\mathbf{B}\|_2$
10. Se  $\mathbf{A}$  e  $\mathbf{B}$  são ortogonais, então  $\mathbf{A} \otimes \mathbf{B}$  é ortogonal.
11.  $(\mathbf{A} \odot \mathbf{B}) \odot \mathbf{C} = \mathbf{A} \odot (\mathbf{B} \odot \mathbf{C})$
12.  $(\mathbf{A} \odot \mathbf{B})^T (\mathbf{A} \odot \mathbf{B}) = (\mathbf{A}^T \mathbf{A}) * (\mathbf{B}^T \mathbf{B})$
13.  $(\mathbf{A} \odot \mathbf{B})^\dagger = [(\mathbf{A}^T \mathbf{A}) * (\mathbf{B}^T \mathbf{B})]^\dagger (\mathbf{A} \odot \mathbf{B})^T$

onde  $\mathbf{A}^\dagger$  é a notação para pseudo-inversa de Moore-Penrose de  $\mathbf{A}$ . Mais informações podem ser encontradas em [19, Section 12.3].

## APÊNDICE B

### Decomposição Esqueletal

Em diversas aplicações que envolvem matrizes densas de grandes porte, o cálculo de uma aproximação de baixo posto através da decomposição SVD é inviável. Dada uma matriz  $\mathbf{A} \in \mathbb{R}^{m \times n}$  com  $\text{posto}(\mathbf{A}) = r$ , podemos representá-la da seguinte maneira

$$\mathbf{A} = \mathbf{C}\hat{\mathbf{A}}^{-1}\mathbf{R}, \quad (\text{B.1})$$

onde  $\mathbf{C} = \mathbf{A}_{\cdot, \mathcal{J}}$  são  $r$  colunas de  $\mathbf{A}$  e  $\mathbf{R} = \mathbf{A}_{\mathcal{I}, \cdot}$  são  $r$  linhas de  $\mathbf{A}$ . Os índices são dados pelos conjuntos  $\mathcal{I}$  e  $\mathcal{J}$  de forma que

$$\hat{\mathbf{A}} = \mathbf{A}_{\mathcal{I}, \mathcal{J}}$$

é a submatriz não singular formada pela interseção das linhas e colunas de  $\mathbf{A}$ . Denotando  $\mathbf{D} = (\hat{\mathbf{A}}^{-1}\mathbf{R})^T$ , obtemos a chamada decomposição esqueletal de  $\mathbf{A}$ :

$$\mathbf{A} = \mathbf{C}\mathbf{D}^T = \sum_{l=1}^r \mathbf{c}^{(l)}\mathbf{d}^{(l)T}.$$

O problema aqui é como escolher as linhas e colunas quando consideramos uma aproximação de baixo posto. Uma abordagem é selecionar as linhas e colunas de forma que a interseção tenha maior determinante em módulo (volume) possível. Neste sentido, segue que

$$\left\| \mathbf{A} - \mathbf{C}\hat{\mathbf{A}}^{-1}\mathbf{R} \right\|_C \leq (r+1)\sigma_{r+1}(\mathbf{A}),$$

onde  $\|\cdot\|_C$  denota o maior valor em módulo das entradas da matriz (veja [20]). Ainda, o  $(r+1)$ -ésimo valor singular de  $\mathbf{A}$  é denotado por  $\sigma_{r+1}(\mathbf{A})$ . No entanto, o problema de encontrar a submatriz de maior volume é do tipo NP-difícil [57] e é necessário prosseguirmos heuristicamente. De fato, apresentaremos a seguir um algoritmo guloso que busca maximizar, a cada iteração, o volume de uma submatriz de  $\mathbf{A}$ . A ideia será construir aproximações de  $\mathbf{A}$  com postos crescentes: começamos escolhendo aleatoriamente uma linha  $i_1$  e em seguida selecionamos a coluna  $j_1$  tal que

$$j_1 = \underset{j=1, \dots, m}{\text{argmax}} |a_{i_1, j}|.$$

e usamos  $\mathbf{A}^{(1)} = \mathbf{A}_{:,j_1} a_{i_1,j_1}^{-1} \mathbf{A}_{i_1,:}$  como primeira aproximação. Repetimos o processo para a matriz  $\mathbf{R}^{(1)} = \mathbf{A} - \mathbf{A}^{(1)}$ , em seguida para  $\mathbf{R}^{(2)} = \mathbf{A} - \mathbf{A}^{(1)} - \mathbf{A}^{(2)}$ , até que o último resíduo  $\mathbf{R}^{(r)} = \mathbf{A} - \mathbf{A}^{(1)} - \dots - \mathbf{A}^{(r)}$  tenha norma de Frobenius pequena suficiente ou atingirmos o  $r$  aproximações. Em todas as etapas armazenamos os índices das linhas no conjunto  $\mathcal{I}$  e os índices das colunas em  $\mathcal{J}$ . Denominamos este método de Aproximação Cruzada e os passos correspondentes estão descritos no Algoritmo 11. Uma maneira diferente mas muito semelhante à descrita anteriormente é feita a partir do método MaxVol estudado por Goreinov *et al.* em [21].

Outro critério de parada leva em conta outras entradas da matriz que não foram escolhidas pelo método. Suponha que o método esteja na etapa  $k$  e considere uma quantidade  $t$  fixa de entradas da matriz  $\mathbf{A}$  dadas por  $a_{i_l,j_l}$ , com  $i_l \in \{1, \dots, m\} \setminus \mathcal{I}$  e  $j_l \in \{1, \dots, n\} \setminus \mathcal{J}$ ,  $l = 1, \dots, t$ . Ao fim da etapa, uma nova representação esqueletal  $\mathbf{R}^{(k)}$  é construída. O critério de parada é satisfeito se, dada uma tolerância  $\tau > 0$ , temos

$$\frac{|a_{i_l,j_l} - r_{i_l,j_l}^{(k)}|}{a_{\max}} \leq \tau, \quad \forall l = 1, \dots, t,$$

onde  $a_{\max}$  é o máximo em módulo das entradas  $a_{i_l,j_l}$  para  $l = 1, \dots, t$ . Para uma descrição detalhada do algoritmo e do critério de parada descrito, veja [18].

---

**Algoritmo 11 – APROXIMAÇÃO CRUZADA PARA DECOMPOSIÇÃO ESQUELETAL**

---

**Entrada:**  $i_1^* \in \{1, \dots, m\}$ ,  $\mathcal{I} = \{\}$ ,  $\mathcal{J} = \{\}$ ,  $\tau > 0$

$\text{stop} = 0$

$k = 1$

**enquanto**  $\text{stop} = 0$  **faça**

$d_j^{(k)} = a_{i_k, j} - \sum_{l=1}^{k-1} c_{i_k}^{(l)} d_j^{(l)}$

$j_k = \operatorname{argmax}_{j=1, \dots, n} |d_j^{(k)}|$

pivo =  $d_{j_k}^{(k)}$

Tome  $\mathcal{J} = \mathcal{J} \cup \{j_k\}$ .

**se**  $|\text{pivo}| < \tau$  **então**

Escolha  $t$  linhas e colunas de  $\mathbf{R}^{(k-1)}$  aleatoriamente e tome o índice  $i_l$  do maior elemento em módulo desta.

**fim**

**senão**

Tome  $\mathcal{I} = \mathcal{I} \cup \{i_k\}$ .

$c_i^{(k)} = \frac{1}{\text{pivo}} \left( a_{i, j_k} - \sum_{l=1}^{k-1} c_i^{(l)} d_{j_k}^{(l)} \right)$

$i_{k+1} = \operatorname{argmax}_{i \in \{1, \dots, m\} \setminus \mathcal{I}} |c_i^{(k)}|$

Se o critério de parada é satisfeito:  $\text{stop} = 1$

$k = k + 1$

**fim**

**fim**

**retorna**  $\mathcal{I}$  e  $\mathcal{J}$ .

---