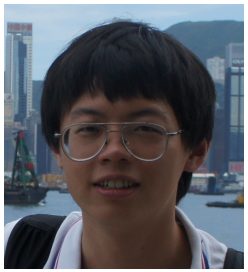
ICASSP 2017 Tutorial #12: Tensor Decomposition for Signal Processing and Machine Learning

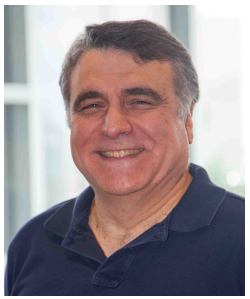
Presenters:
N.D. Sidiropoulos, L. De Lathauwer, X. Fu, E.E. Papalexakis

Sunday, March 5 2017

Main reference





N.D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E.E. Papalexakis, C. Faloutsos, "**Tensor Decomposition for Signal Processing and Machine Learning**", *IEEE Trans. on Signal Processing*, to appear in 2017 (overview paper 32 pages; + 12 pages supplement, plus code and demos; arXiv preprint 1607.01668v2).
https://arxiv.org/abs/1607.01668

Tensors?

- A **tensor** is an array whose elements are indexed by $(i, j, k, \ldots)$ – more than two indexes.
  - Purchase data: indexed by 'user', 'item', 'seller', 'time'
  - Movie rating data: indexed by 'user', 'movie', 'time'
  - MIMO radar: indexed by 'angle', 'range', 'Doppler', 'profiles'
  - ...
  - Tensor can also represent an *operator* – more later.
- A matrix is a *second-order* (AKA *two-way*) tensor, whose elements are indexed by (i, j).
- Three-way tensors are easy to visualize "shoe boxes".

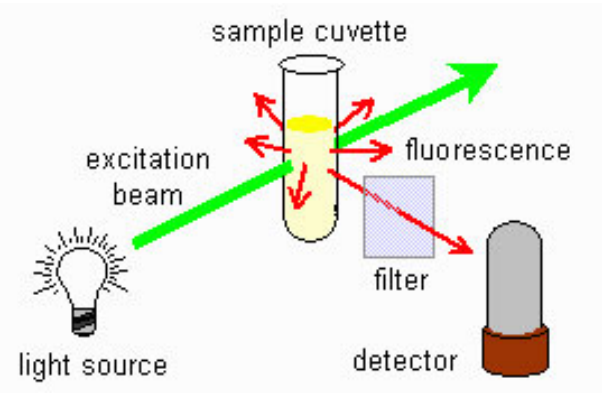
Why Are We Interested in Tensors?

- Tensor algebra has many similarities with matrix algebra, but also some **striking differences**.
- Determining matrix rank is easy (SVD), but determining higher-order tensor rank is NP-hard.
- Low-rank matrix decomposition is easy (SVD), but tensor decomposition is NP-hard.
- A matrix can have an infinite number of low-rank decompositions, but tensor decomposition is **essentially unique** under mild conditions.

Pertinent Applications

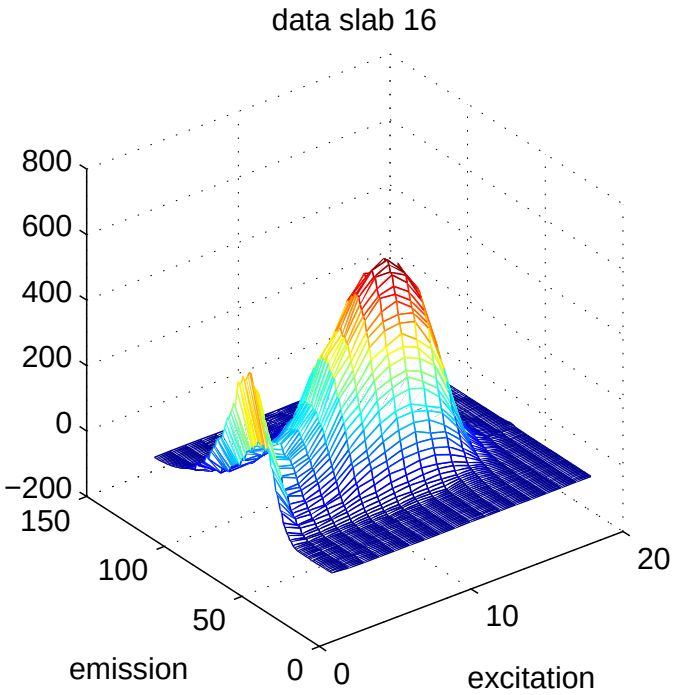
- Early developments: Psychometrics and Chemometrics
- Begining from the 1990s: Signal processing
  - communications, radar,
  - speech and audio,
  - biomedical
  - ...
- Starting in mid-2000's: Machine learning and data mining
  - clustering
  - dimensionality reduction
  - latent factor models (topic mining & community detection)
  - structured subspace learning
  - ...

Chemometrics - Fluorescence Spectroscopy



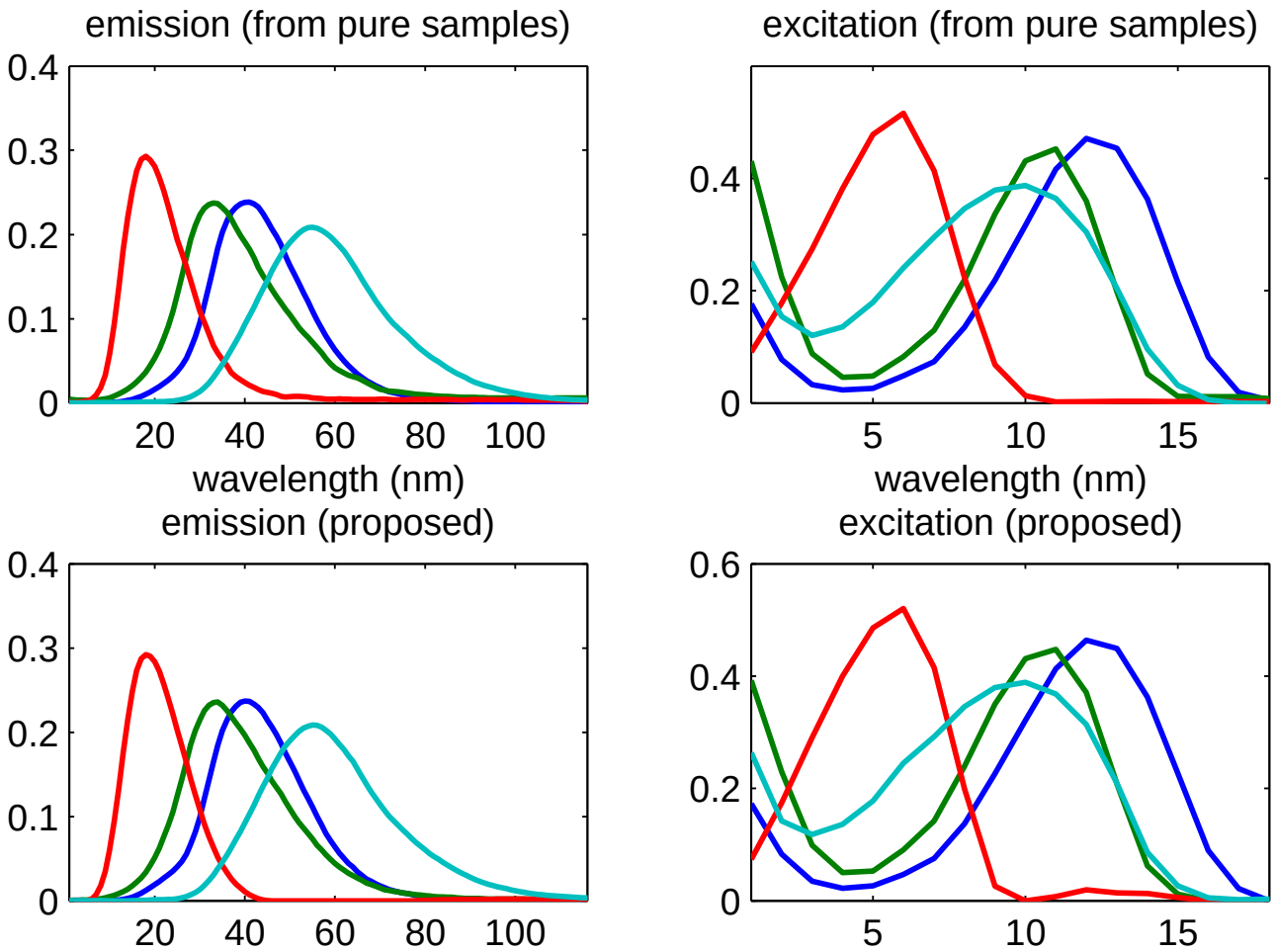
http://www.chromedia.org/

credit:



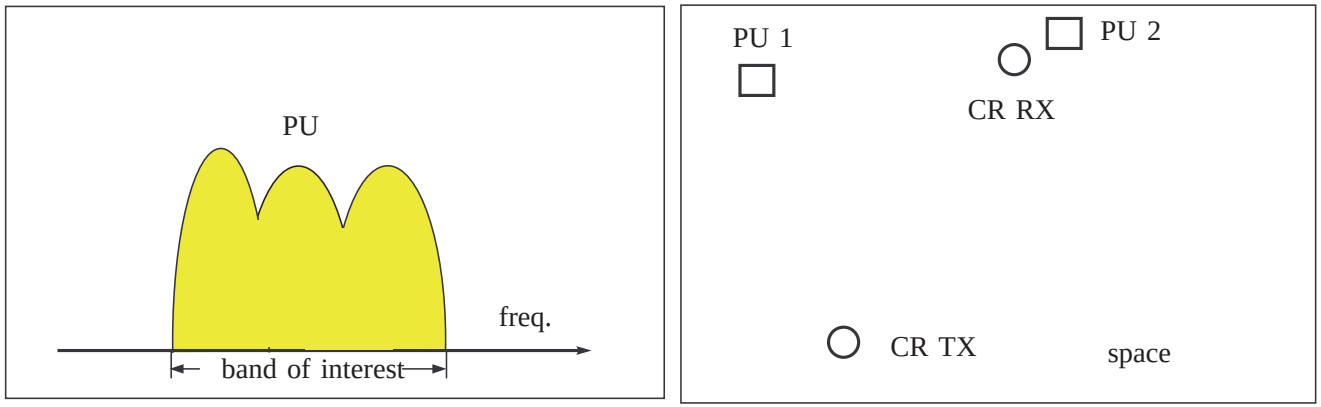
Chemometrics - Fluorescence Spectroscopy

- Resolved spectra of different materials.

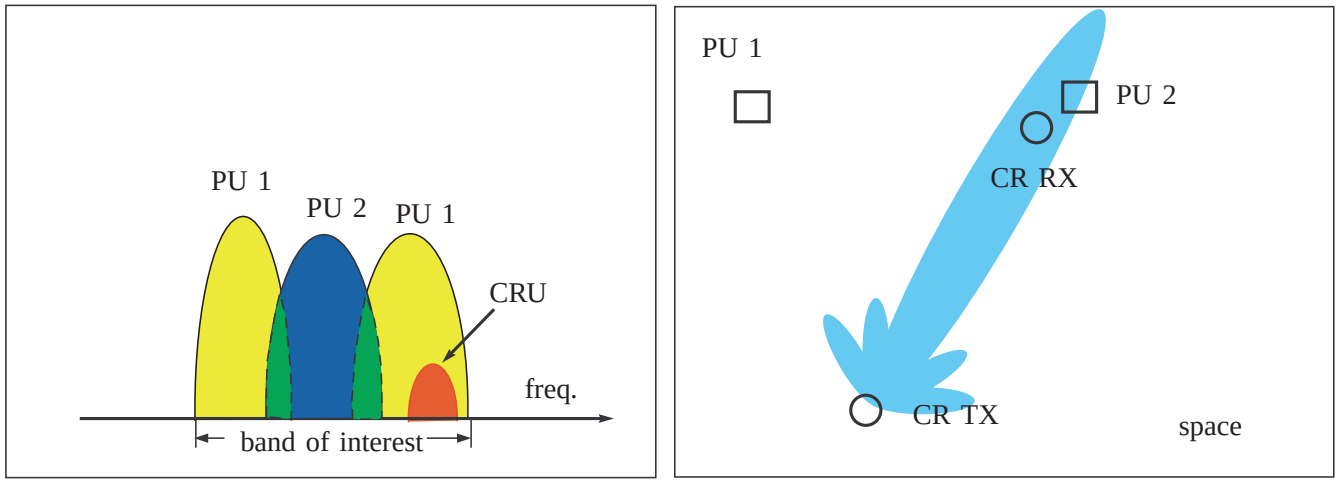


Signal Processing - Signal Separation

- Spectrum sensing in cognitive radio – the multiple transceiver case.

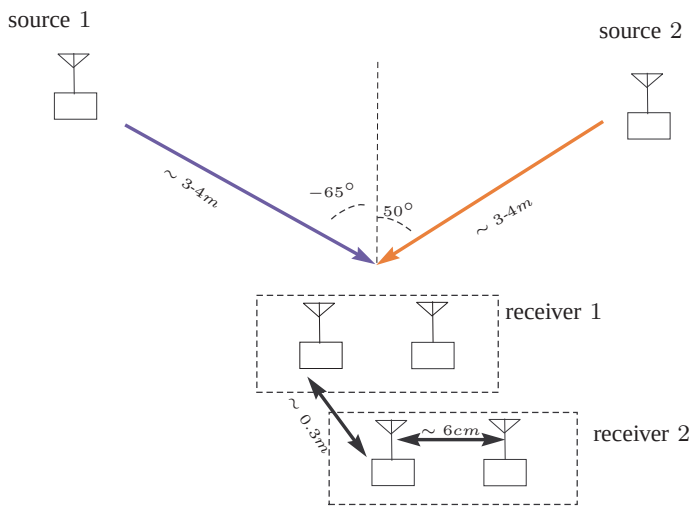


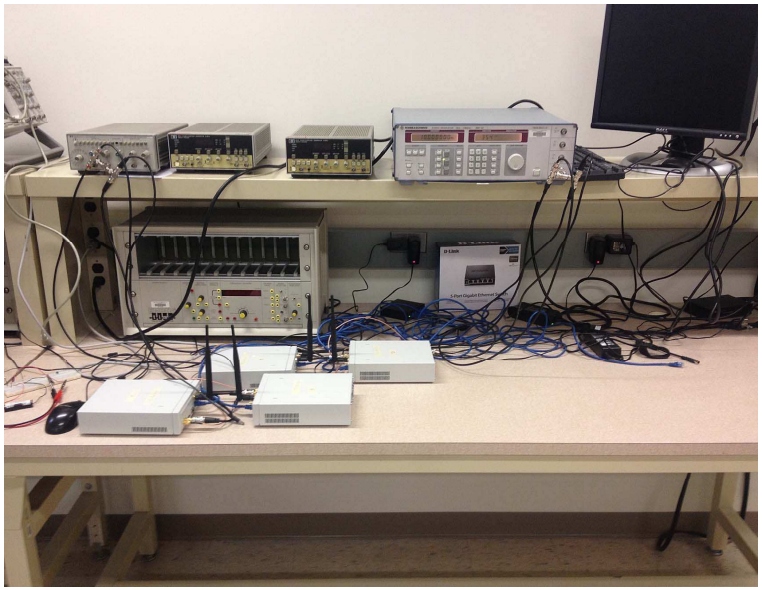
(a)



(b)

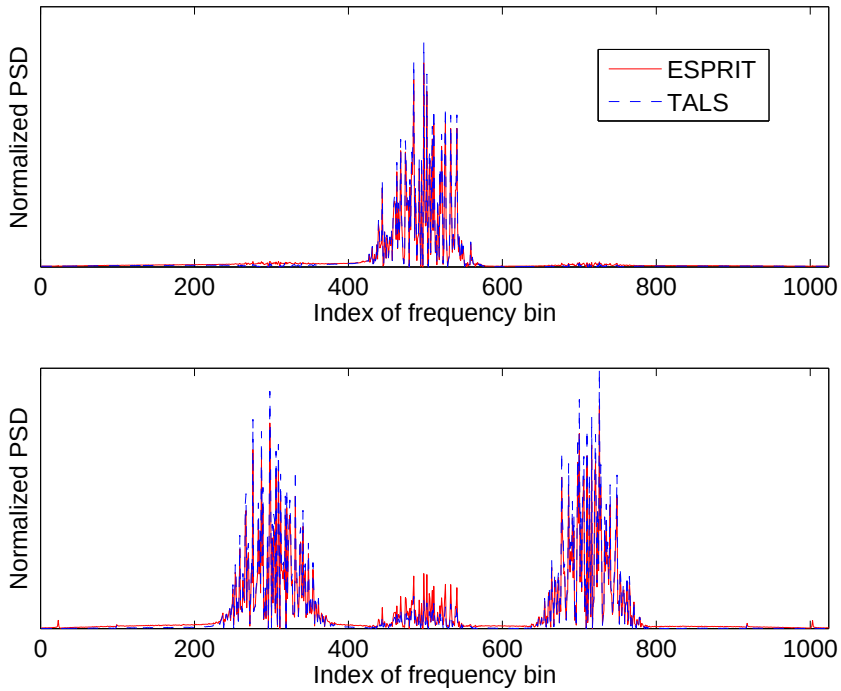
Signal Processing - Signal Separation





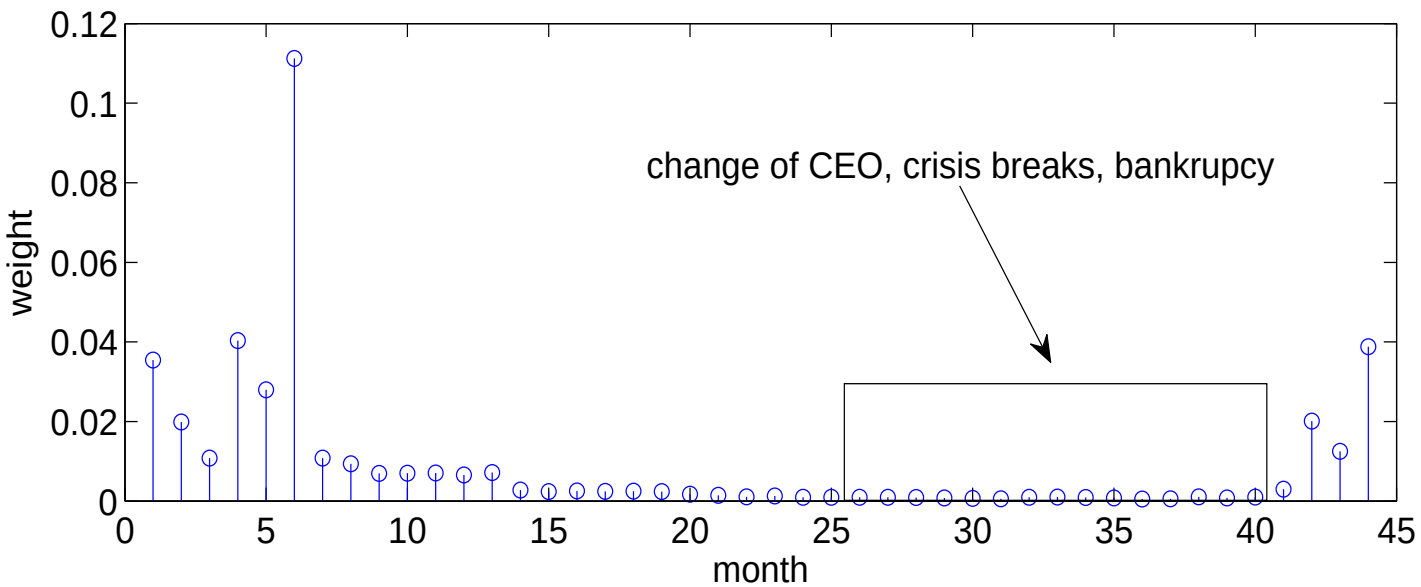
Signal Processing - Signal Separation

Algorithm	Measure	Avergae Result
ESPRIT	SMR	0.1572
	DOAs	$(-67.1438°, 47.3884°)$
TALS	SMR	0.1014
	DOAs	$(-67.1433°, 53.0449°)$
NMF	SMR	0.2537
	DOAs	-



Machine Learning - Social Network Co-clustering

- Email data from the Enron company.
- Indexed by ‘sender’, ‘receiver’, ‘month’ – a three-way tensor.



Machine Learning - Social Network Co-clustering

cluster 1 (Legal; 16 persons)	cluster 2 (Excecutive; 18 persons)	cluster 3 (Executive; 25 persons)
Brenda Whitehead, N/A	David Delainey, CEO ENA and Enron Energy Services	Andy Zipper , VP Enron Online
Dan Hyvl, N/A	Drew Fossum, VP Transwestern Pipeline Company (ETS)	Jeffrey Shankman, President Enron Global Markets
Debra Perlingiere, Legal Specialist ENA Legal	Elizabeth Sager, VP and Asst Legal Counsel ENA Legal	Barry Tycholiz, VP Marketing
Elizabeth Sager, VP and Asst Legal Counsel ENA Legal	James Steffes, VP Government Affairs	Richard Sanders, VP Enron Wholesale Services
Jeff Hodge, Asst General Counsel ENA Legal	Jeff Dasovich, Employee Government Relationship Executive	James Steffes, VP Government Affairs
Kay Mann, Lawyer	John Lavorato, CEO Enron America	Mark Haedicke, Managing Director ENA Legal
Louise Kitchen, President Enron Online	Kay Mann, Lawyer	Greg Whalley, President
Marie Heard, Senior Legal Specialist ENA Legal	Kevin Presto, VP East Power Trading	Jeff Dasovich, Employee Government Relationship Executive
Mark Haedicke, Managing Director ENA Legal	Margaret Carson, Employee Corporate and Environmental Policy	Jeffery Skilling, CEO
Mark Taylor , Manager Financial Trading Group ENA Legal	Mark Haedicke, Managing Director ENA Legal	Vince Kaminski, Manager Risk Management Head
Richard Sanders, VP Enron Wholesale Services	Philip Allen, VP West Desk Gas Trading	Steven Kean, VP Chief of Staff
Sara Shackleton, Employee ENA Legal	Richard Sanders, VP Enron Wholesale Services	Joannie Williamson, Executive Assistant
Stacy Dickson, Employee ENA Legal	Richard Shapiro , VP Regulatory Affairs	John Arnold, VP Financial Enron Online
Stephanie Panus, Senior Legal Specialist ENA Legal	Sally Beck, COO	John Lavorato, CEO Enron America
Susan Bailey, Legal Assistant ENA Legal	Shelley Corman, VP Regulatory Affairs	Jonathan McKa, Director Canada Gas Trading
Tana Jones, Employee Financial Trading Group ENA Legal	Steven Kean, VP Chief of Staff	Kenneth Lay, CEO
	Susan Scott, Employee Transwestern Pipeline Company (ETS)	Liz Taylor, Executive Assistant to Greg Whalley
	Vince Kaminski, Manager Risk Management Head	Louise Kitchen, President Enron Online
cluser 4 (Trading; 12 persons)	cluster 5 (Pipeline; 15 persons)	Michelle Cash, N/A
Chris Dorland, Manager	Bill Rapp, N/A	Mike McConnel, Executive VP Global Markets
Eric Bas, Trader Texas Desk Gas Trading	Darrell Schoolcraft, Employee Gas Control (ETS)	Kevin Presto, VP East Power Trading
Philip Allen, Manager	Drew Fossum, VP Transwestern Pipeline Company (ETS)	Richard Shapiro, VP Regulatory Affairs
Kam Keiser, Employee Gas	Kevin Hyatt, Director Asset Development TW Pipeline Business (ETS)	Rick Buy, Manager Chief Risk Management Officer
Mark Whitt, Director Marketing	Kimberly Watson, Employee Transwestern Pipeline Company (ETS)	Sally Beck, COO
Martin Cuilla, Manager Central Desk Gas Trading	Lindy Donoho, Employee Transwestern Pipeline Company (ETS)	Hunter Shively, VP Central Desk Gas Trading
Matthew Lenhart, Analyst West Desk Gas Trading	Lynn Blair, Employee Northern Natural Gas Pipeline (ETS)	
Michael Grigsby, Director West Desk Gas Trading	Mark McConnell, Employee Transwestern Pipeline Company (ETS)	
Monique Sanchez, Associate West Desk Gas Trader (EWS)	Michelle Lokay, Admin. Asst. Transwestern Pipeline Company (ETS)	
Susan Scott, Employee Transwestern Pipeline Company (ETS)	Rod Hayslett, VP Also CFO and Treasurer	
Jane Tholt, VP West Desk Gas Trading	Shelley Corman, VP Regulatory Affairs	
Philip Allen, VP West Desk Gas Trading	Stanley Horton, President Enron Gas Pipeline	
	Susan Scott, Employee Transwestern Pipeline Company (ETS)	
	Teb Lokey, Manager Regulatory Affairs	
	Tracy Geaccone, Manager (ETS)	

Goals of this course

- A comprehensive overview with sufficient technical depth.
- Required background: First graduate courses in linear algebra, probability & random vectors. A bit of optimization will help, but not strictly required.
- Sufficient technical details to allow graduate students to start tensor-related research; and practitioners to start developing tensor software.
- Proofs of and insights from special cases that convey the essence.
- Understand the basic (and very different) ways in which tensor decompositions are used in signal processing and machine learning.
- Various examples of how practical problems are formulated and solved as tensor decomposition problems.

Roadmap

- **Preliminaries**
- **Rank and rank decomposition** (CPD)
- **Uniqueness, demystified**
- **Other models**
  - Tucker, MLSVD
  - Tensor Trains, PARALIND / Block Component Decomposition, ... (brief)
- **Algorithms**
  - basic and constrained algorithms, factorization at scale
- **Applications in signal processing and machine learning**

Rank & Rank Decomposition - Matrices

- Consider an $I \times J$ matrix $\mathbf{X}$.
- colrank($\mathbf{X}$) $:=$ no. of linearly indep. columns of $\mathbf{X}$, i.e., dim(range($\mathbf{X}$)).
- colrank($\mathbf{X}$) is the minimum $k \in \mathbb{N}$ such that $\mathbf{X} = \mathbf{A}\mathbf{B}^T$, where $\mathbf{A}$ is an $I \times k$ basis of range($\mathbf{X}$), and $\mathbf{B}^T$ is $k \times J$.
- dim(range($\mathbf{X}^T$)), which is the minimum $\ell \in \mathbb{N}$ such that $\mathbf{X}^T = \mathbf{B}\mathbf{A}^T \Longleftrightarrow \mathbf{X} = \mathbf{A}\mathbf{B}^T$, where $\mathbf{B}$ is $J \times \ell$ and $\mathbf{A}^T$ is $\ell \times I$.

Rank & Rank Decomposition - Matrices

- $\mathbf{X} = \sum_{n=1}^{\ell} \mathbf{a}_n \mathbf{b}_n^T$, where $\mathbf{A} = [\mathbf{a}_1, \cdots, \mathbf{a}_\ell]$ and $\mathbf{B} = [\mathbf{b}_1, \cdots, \mathbf{b}_\ell]$.
  - $\text{rank}(\mathbf{X}) =$ minimum m such that $\mathbf{X} = \sum_{n=1}^{m} \mathbf{a}_n \mathbf{b}_n^T$.
- Easy to notice: $\text{colrank}(\mathbf{X}) = \text{rowrank}(\mathbf{X}) = \text{rank}(\mathbf{X})$.
- The three definitions actually coincide!
- Obviously, $\text{rank}(\mathbf{X}) \leq \min(I, J)$.

Low-Rank Matrix Approximation

- In practice, we observe $\mathbf{X} = \mathbf{L} + \mathbf{N}$.
  - $\mathbf{L} = \mathbf{A}\mathbf{B}^T$ is low-rank.
  - $\mathbf{N}$ represents noise and 'unmodeled dynamics'.
- In many cases we are interested in the $\mathbf{L}$ part:

$$\min_{\mathbf{L} \mid \text{rank}(\mathbf{L})=\ell} ||\mathbf{X} - \mathbf{L}||_F^2 \iff \min_{\mathbf{A}\in\mathbb{R}^{I\times\ell},\ \mathbf{B}\in\mathbb{R}^{J\times\ell}} ||\mathbf{X} - \mathbf{A}\mathbf{B}^T||_F^2.$$

- Solution: truncated SVD of $\mathbf{X}$.
  - $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, $\mathbf{L} = \mathbf{U}(:, 1:\ell)\mathbf{\Sigma}(1:\ell, 1:\ell)(\mathbf{V}(:, 1:\ell))^T$.
  - $\mathbf{A} = \mathbf{U}(:, 1:\ell)\mathbf{\Sigma}(1:\ell, 1:\ell)$, $\mathbf{B} = \mathbf{V}(:, 1:\ell)$.
- Even without noise, low-rank decomposition of $\mathbf{X}$ is non-unique:

$$\mathbf{A}\mathbf{B}^T = \mathbf{A}\mathbf{M}\mathbf{M}^{-1}\mathbf{B}^T = (\mathbf{A}\mathbf{M})(\mathbf{B}\mathbf{M}^{-T})^T,$$

holds for any non-singular $\mathbf{M}$.

Useful Products

- **Kronecker product** of $\mathbf{A}$ ($I \times K$) and $\mathbf{B}$ ($J \times L$) is the $IJ \times KL$ matrix

$$\mathbf{A} \otimes \mathbf{B} := \begin{bmatrix} \mathbf{BA}(1,1) & \mathbf{BA}(1,2) & \cdots & \mathbf{BA}(1,K) \\ \mathbf{BA}(2,1) & \mathbf{BA}(2,2) & \cdots & \mathbf{BA}(2,K) \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{BA}(I,1) & \mathbf{BA}(I,2) & \cdots & \mathbf{BA}(I,K) \end{bmatrix}$$

- Properties:
  - $\mathbf{b} \otimes \mathbf{a} = \text{vec}(\mathbf{a}\mathbf{b}^T)$.
  - $\text{vec}\left(\mathbf{A}\mathbf{M}\mathbf{B}^T\right) = (\mathbf{B} \otimes \mathbf{A})\,\text{vec}(\mathbf{M})$.

$$\begin{aligned} \text{vec}\left(\mathbf{A}\mathbf{M}\mathbf{B}^T\right) &= \text{vec}\left(\sum_{k=1}^{K}\sum_{\ell=1}^{L} \mathbf{A}(:,k)\mathbf{M}(k,\ell)(\mathbf{B}(:,\ell))^T\right) \\ &= \sum_{k=1}^{K}\sum_{\ell=1}^{L} \mathbf{M}(k,\ell)\mathbf{B}(:,\ell) \otimes \mathbf{A}(:,k) \\ &= (\mathbf{B} \otimes \mathbf{A})\,\text{vec}(\mathbf{M}). \end{aligned}$$

Useful Products - Cont'd

- The vectorization property is handy for computation:

$$\min_{\mathbf{M}} ||\mathbf{X} - \mathbf{A}\mathbf{M}\mathbf{B}^T||_F^2 \Longleftrightarrow \min_{\mathbf{m}} ||\text{vec}(\mathbf{X}) - (\mathbf{B} \otimes \mathbf{A})\mathbf{m}||_2^2,$$

  where $\mathbf{m} = \text{vec}(\mathbf{M})$.

- **Khatri-Rao Product**: $\mathbf{A} \odot \mathbf{B} := [\mathbf{a}_1 \otimes \mathbf{b}_1, \cdots \mathbf{a}_\ell \otimes \mathbf{b}_\ell]$.
- Define $\mathbf{D} = \text{Diag}(\mathbf{d})$ and $\mathbf{d} = \text{diag}(\mathbf{D})$. $\text{vec}\left(\mathbf{A}\mathbf{D}\mathbf{B}^T\right) = (\mathbf{B} \odot \mathbf{A})\,\mathbf{d}$, which is useful when dealing with the following

$$\min_{\mathbf{D}=\text{Diag}(\mathbf{d})} ||\mathbf{X} - \mathbf{A}\mathbf{D}\mathbf{B}^T||_F^2 \Longleftrightarrow \min_{\mathbf{d}} ||\text{vec}(\mathbf{X}) - (\mathbf{B} \odot \mathbf{A})\mathbf{d}||_2^2.$$

- Khatri–Rao product $\mathbf{B} \odot \mathbf{A}$ is a subset of columns from $\mathbf{B} \otimes \mathbf{A}$.

Useful Products - More Properties

- $(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C})$ (associative)
- Note though that $\mathbf{A} \otimes \mathbf{B} \neq \mathbf{B} \otimes \mathbf{A}$ (non-commutative).
- $(\mathbf{A} \otimes \mathbf{B})^T = \mathbf{A}^T \otimes \mathbf{B}^T$ (note order, unlike $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$).
- $(\mathbf{A} \otimes \mathbf{B})^* = \mathbf{A}^* \otimes \mathbf{B}^* \Longrightarrow (\mathbf{A} \otimes \mathbf{B})^H = \mathbf{A}^H \otimes \mathbf{B}^H$
  - *, ^{H} stand for conjugation and Hermitian transposition, respectively.
- $(\mathbf{A} \otimes \mathbf{B})(\mathbf{E} \otimes \mathbf{F}) = (\mathbf{AE} \otimes \mathbf{BF})$ (mixed product rule).
  - $(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$, for square $\mathbf{A}$, $\mathbf{B}$.
  - If $\mathbf{A} = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T$ and $\mathbf{B} = \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^T$, $\mathbf{A} \otimes \mathbf{B} = (\mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T) \otimes (\mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^T)$ $= (\mathbf{U}_1 \otimes \mathbf{U}_2)(\mathbf{\Sigma}_1 \otimes \mathbf{\Sigma}_2)(\mathbf{V}_1 \otimes \mathbf{V}_2)^T$.
  - $\text{rank}(\mathbf{A} \otimes \mathbf{B}) = \text{rank}(\mathbf{A})\text{rank}(\mathbf{B})$.
  - $\text{tr}(\mathbf{A} \otimes \mathbf{B}) = \text{tr}(\mathbf{A})\text{tr}(\mathbf{B})$, for square $\mathbf{A}$, $\mathbf{B}$.
  - $\det(\mathbf{A} \otimes \mathbf{B}) = \det(\mathbf{A})\det(\mathbf{B})$, for square $\mathbf{A}$, $\mathbf{B}$.

Useful Products - More Properties

- $(\mathbf{A} \odot \mathbf{B}) \odot \mathbf{C} = \mathbf{A} \odot (\mathbf{B} \odot \mathbf{C})$ (associative).
- $\mathbf{A} \odot \mathbf{B} \neq \mathbf{B} \odot \mathbf{A}$ (non-commutative).
- $(\mathbf{A} \otimes \mathbf{B})(\mathbf{E} \odot \mathbf{F}) = (\mathbf{AE}) \odot (\mathbf{BF})$ (mixed product rule).
- Heads-up: the mixed product rule plays an essential role in large-scale tensor computations.

Useful Products - Tensor Outer Product

- **Tensor (outer) product** of $\mathbf{a}$ $(I \times 1)$ and $\mathbf{b}$ $(J \times 1)$ is defined as the $I \times J$ matrix $\mathbf{a} \circledcirc \mathbf{b}$ with elements

$$(\mathbf{a} \circledcirc \mathbf{b})(i,j) = \mathbf{a}(i)\mathbf{b}(j)$$

- Note that $\mathbf{a} \circledcirc \mathbf{b} = \mathbf{a}\mathbf{b}^T$.
- $\mathbf{a} \circledcirc \mathbf{b} \circledcirc \mathbf{c}$ with elements $(\mathbf{a} \circledcirc \mathbf{b} \circledcirc \mathbf{c})(i,j,k) = \mathbf{a}(i)\mathbf{b}(j)\mathbf{c}(k)$.
- Naturally generalizes to three- and higher-way cases.

Tensor Rank

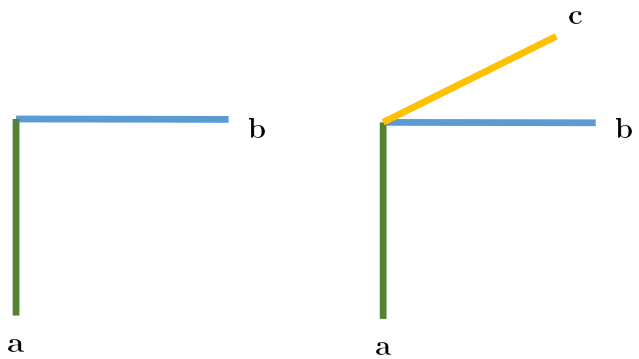


Figure: Schematic of a rank-1 matrix and tensor.

- A **rank-1 matrix X** of size $I \times J$ is an outer product of two vectors: $\mathbf{X}(i,j) = \mathbf{a}(i)\mathbf{b}(j), \forall i \in \{1, \cdots, I\}, j \in \{1, \cdots, J\}$; i.e.,

$$\mathbf{X} = \mathbf{a} \circledcirc \mathbf{b}.$$

- A **rank-1 third-order tensor X** of size $I \times J \times K$ is an outer product of three vectors: $\mathbf{X}(i,j,k) = \mathbf{a}(i)\mathbf{b}(j)\mathbf{c}(k)$; i.e.,

$$\mathbf{X} = \mathbf{a} \circledcirc \mathbf{b} \circledcirc \mathbf{c}.$$

Tensor Rank

- Rank of matrix $\mathbf{X}$ is the smallest F that $\mathbf{X} = \sum_{f=1}^{F} \mathbf{a}_f \circledcirc \mathbf{b}_f$ holds for some $\mathbf{a}_f$'s and $\mathbf{b}_f$'s.
- **Rank of tensor $\mathbf{X}$** is the minimum number of rank-1 tensors needed to produce $\mathbf{X}$ as their sum.
- A tensor of rank at most F can be written as

$$\mathbf{X} = \sum_{f=1}^{F} \mathbf{a}_f \circledcirc \mathbf{b}_f \circledcirc \mathbf{c}_f \Longleftrightarrow \mathbf{X}(i,j,k) = \sum_{f=1}^{F} \mathbf{a}_f(i)\mathbf{b}_f(j)\mathbf{c}_f(k)$$

- It is also customary to use $\mathbf{X}(i,j,k) = \sum_{f=1}^{F} a_{i,f} b_{j,f} c_{k,f}$.
- Let $\mathbf{A} := [\mathbf{a}_1, \cdots, \mathbf{a}_F]$, $\mathbf{B} := [\mathbf{b}_1, \cdots, \mathbf{b}_F]$, and $\mathbf{C} := [\mathbf{c}_1, \cdots, \mathbf{c}_F] \Rightarrow \mathbf{X}(i,j,k) = \sum_{f=1}^{F} \mathbf{A}(i,f)\mathbf{B}(j,f)\mathbf{C}(k,f)$.
- For brevity, $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$.

Tensor Rank - Illustration

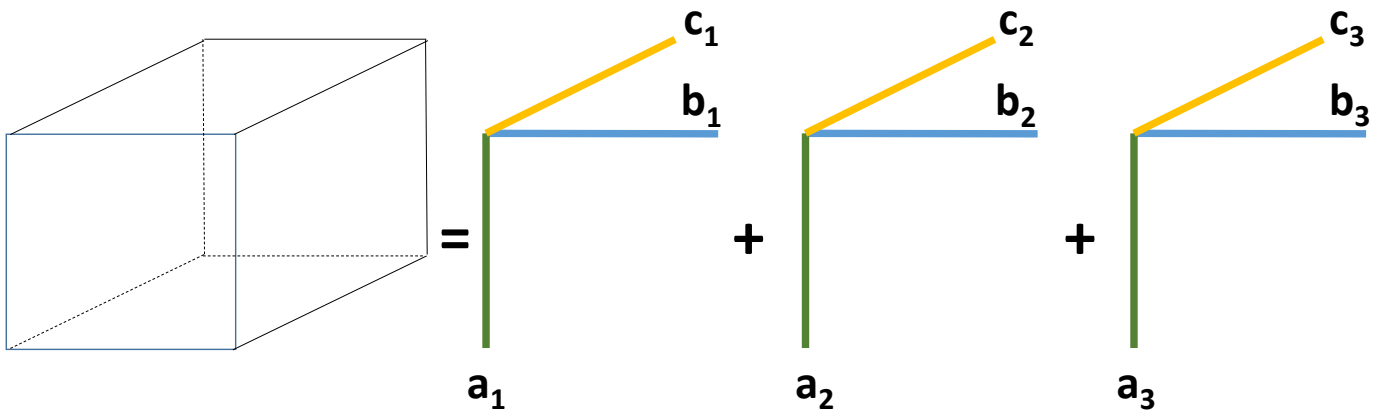


Figure: Schematic of tensor of rank three.

Slab Representations

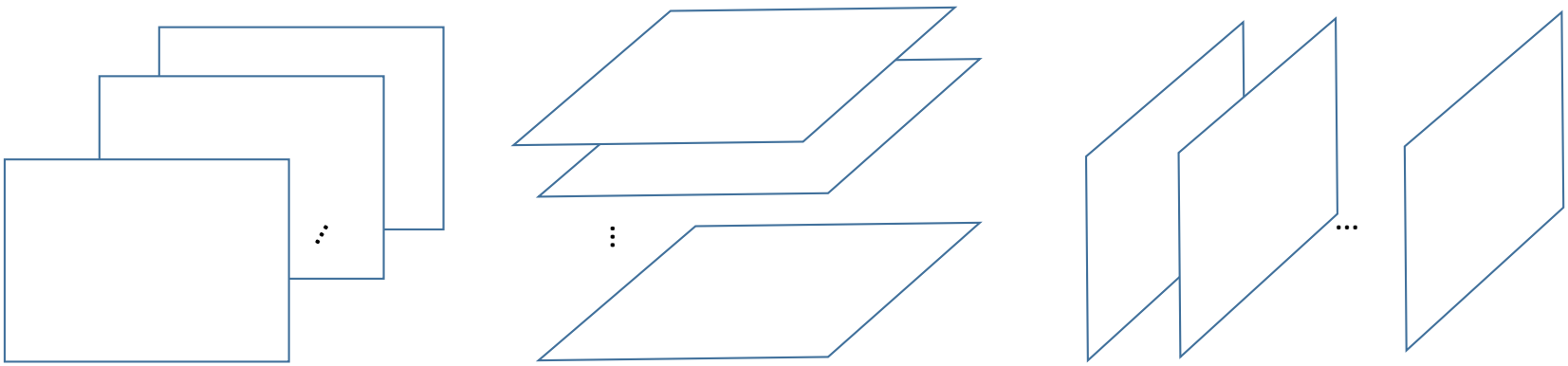


Figure: Slab views of a three-way tensor.

Slab Representations - Towards Rank Decomposition

- Let us look at the **frontal slab** $\mathbf{X}(:,:,1)$ of $\mathbf{X}$:

$$\mathbf{X}(i,j,1) = \sum_{f=1}^{F} \mathbf{a}_f(i)\mathbf{b}_f(j)\mathbf{c}_f(1) \Longrightarrow \mathbf{X}(:,:,1) = \sum_{f=1}^{F} \mathbf{a}_f\mathbf{b}_f^T\mathbf{c}_f(1) =$$

$$\mathbf{A}\text{Diag}([\mathbf{c}_1(1), \mathbf{c}_2(1), \cdots, \mathbf{c}_F(1)])\mathbf{B}^T = \mathbf{A}\text{Diag}(\mathbf{C}(1,:))\mathbf{B}^T.$$

- Denote $\text{D}_k(\mathbf{C}) := \text{Diag}(\mathbf{C}(k,:))$ for brevity. Hence, for any k,

$$\mathbf{X}(:,:,k) = \mathbf{A}\text{D}_k(\mathbf{C})\mathbf{B}^T, \quad \text{vec}(\mathbf{X}(:,:,k)) = (\mathbf{B} \odot \mathbf{A})(\mathbf{C}(k,:))^T,$$

- By parallel stacking, we obtain the matrix **unfolding**

$$\mathbf{X}_3 := [\text{vec}(\mathbf{X}(:,:,1)), \text{vec}(\mathbf{X}(:,:,2)), \cdots, \text{vec}(\mathbf{X}(:,:,K))] \rightarrow$$

$$\mathbf{X}_3 = (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T, \quad (IJ \times K).$$

Slab Representations - Towards Rank Decomposition

- In the same vain, we may consider **lateral slabs** , e.g.,

$$\mathbf{X}(:, j, :) = \mathbf{A}\mathrm{D}_j(\mathbf{B})\mathbf{C}^T \rightarrow \mathrm{vec}(\mathbf{X}(:, j, :)) = (\mathbf{C} \odot \mathbf{A})(\mathbf{B}(j, :))^T.$$

  Hence

$$\mathbf{X}_2 := [\mathrm{vec}(\mathbf{X}(:, 1, :)), \mathrm{vec}(\mathbf{X}(:, 2, :)), \cdots, \mathrm{vec}(\mathbf{X}(:, J, :))] \rightarrow$$

$$\mathbf{X}_2 = (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T, \quad (IK \times J),$$

- Similarly for the **horizontal slabs** $\mathbf{X}(i, :, :) = \mathbf{B}\mathrm{D}_i(\mathbf{A})\mathbf{C}^T$,

$$\mathbf{X}_1 := [\mathrm{vec}(\mathbf{X}(1, :, :)), \mathrm{vec}(\mathbf{X}(2, :, :)), \cdots, \mathrm{vec}(\mathbf{X}(I, :, :))] \rightarrow$$

$$\mathbf{X}_1 = (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T, \quad (KJ \times I).$$

Slab Representations

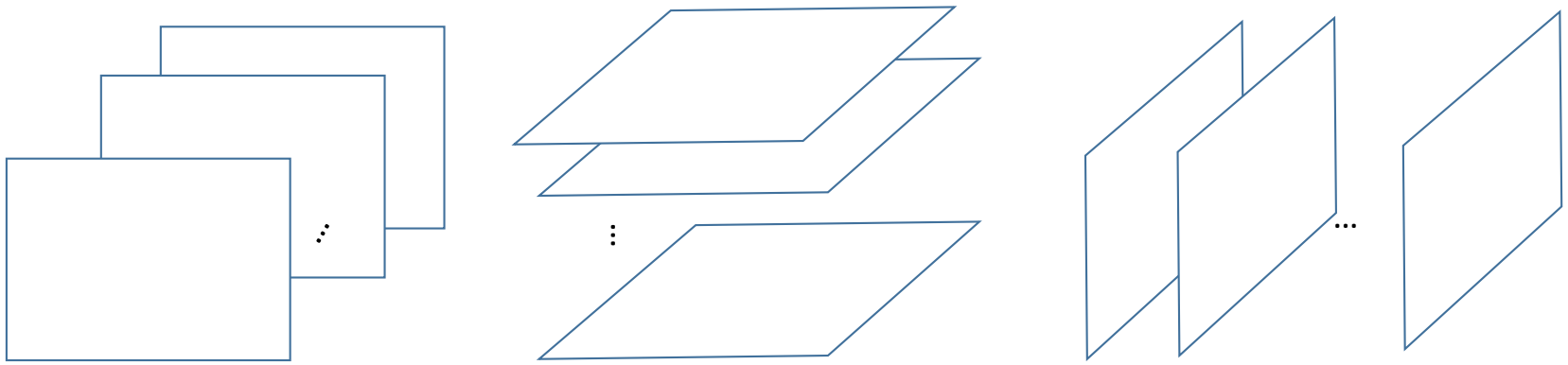


Figure: Slab views of a three-way tensor.

- **Frontal slabS**: $\mathbf{X}(:,:,k) = \mathbf{A}\mathrm{D}_k(\mathbf{C})\mathbf{B}^T$.
- **Horizontal slabs**: $\mathbf{X}(i,:,:) = \mathbf{B}\mathrm{D}_i(\mathbf{A})\mathbf{C}^T$.
- **Lateral slabs**: $\mathbf{X}(:,j,:) = \mathbf{A}\mathrm{D}_j(\mathbf{B})\mathbf{C}^T$.

Low-rank Tensor Approximation

- Adopting a least squares criterion, the problem is

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} ||\mathbf{X} - \sum_{f=1}^{F} \mathbf{a}_f \circledcirc \mathbf{b}_f \circledcirc \mathbf{c}_f||_F^2,$$

- Equivalently, we may consider

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} ||\mathbf{X}_1 - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T||_F^2.$$

- Alternating optimization:

$$\mathbf{A} \leftarrow \arg\min_{\mathbf{A}} ||\mathbf{X}_1 - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T||_F^2,$$

$$\mathbf{B} \leftarrow \arg\min_{\mathbf{B}} ||\mathbf{X}_2 - (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T||_F^2,$$

$$\mathbf{C} \leftarrow \arg\min_{\mathbf{C}} ||\mathbf{X}_3 - (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T||_F^2,$$

- The above is widely known as **Alternating Least Squares (ALS)**.

Bounds on Tensor Rank

- For an $I \times J$ matrix $\mathbf{X}$, we know that $\text{rank}(\mathbf{X}) \leq \min(I, J)$, and $\text{rank}(\mathbf{X}) = \min(I, J)$ almost surely.
- Considering $\mathbf{X} = \mathbf{A}\mathbf{B}^T$ where $\mathbf{A}$ is $I \times F$ and $\mathbf{B}$ is $J \times F$, the **number of unknowns**, or **degrees of freedom** (DoF) in the $\mathbf{A}\mathbf{B}^T$ model is $(I + J - 1)F$.
- The number of equations in $\mathbf{X} = \mathbf{A}\mathbf{B}^T$ is IJ, suggesting that F (at most) of order $\min(I, J)$ may be needed.
- What can we say about $I \times J \times K$ tensors $\mathbf{X}$?

Bounds on Tensor Rank

- For $\mathbf{X} = \sum_{f=1}^{F} \mathbf{a}_f \circledcirc \mathbf{b}_f \circledcirc \mathbf{c}_f$, the DoF is $(I + J + K - 2)F$.
- The number of equations is IJK.
- This suggests that

$$F \geq \lceil \frac{IJK}{I + J + K - 2} \rceil$$

  may be needed to describe an arbitrary tensor $\mathbf{X}$ of size $I \times J \times K$.
- Suggests a 3rd-order tensor's rank can potentially be $\min(IJ, JK, IK)$.
- In fact this turns out being sufficient as well.

Bounds on Tensor Rank - An Intuitive Way to See It

- Denote $\mathbf{X}(:,:,k) = \mathbf{A}_k \mathbf{B}_k^T$.
  - $\mathbf{X}(:,:,k)$ is of size $I \times J \Rightarrow \mathbf{A}_k$ and $\mathbf{B}_k$ have at most $\min(I,J)$ columns.
- Let $\mathbf{A} := [\mathbf{A}_1, \cdots, \mathbf{A}_K]$, $\mathbf{B} := [\mathbf{B}_1, \cdots, \mathbf{B}_K]$, and $\mathbf{C} := \mathbf{I}_{K\times K} \otimes \mathbf{1}_{1\times \min(I,J)}$, we can synthesize $\mathbf{X}$ as $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$.
  - The kth frontal slab looks like

$$\mathbf{X}(:,:,k) = [\mathbf{A}_1, \ldots, \mathbf{A}_K] \begin{bmatrix} \mathbf{0} & \ldots & \mathbf{0} \\ \vdots & \vdots & \vdots \\ \ldots & \mathbf{I}_{\min(I,J)} & \ldots \\ \vdots & \vdots & \vdots \\ \mathbf{0} & \ldots & \mathbf{0} \end{bmatrix} [\mathbf{B}_1, \ldots, \mathbf{B}_K]^T$$

  - implies at most $\min(IK, JK)$ columns in $\mathbf{A}, \mathbf{B}, \mathbf{C}$ to represent $\mathbf{X}$.
  - Using role symmetry, the rank upper bound is $\min(IK, JK, IJ)$.

A Lower Bound of Tensor Rank

- Concatenate the frontal slabs one next to each other

$$[\mathbf{X}(:,:,1)\cdots\mathbf{X}(:,:,K)] = \mathbf{A}\left[\mathbf{D}_k(\mathbf{C})\mathbf{B}^T\cdots\mathbf{D}_k(\mathbf{C})\mathbf{B}^T\right]$$

- F must be $\geq \dim(\text{range}([\mathbf{X}(:,:,1)\cdots\mathbf{X}(:,:,K)]))$.
- Define

$$R_1(\mathbf{X}) := \dim \text{colspan}(\mathbf{X}) := \dim \text{span}\left\{\mathbf{X}(:,j,k)\right\}_{\forall j,k},$$

$$R_2(\mathbf{X}) := \dim \text{rowspan}(\mathbf{X}) := \dim \text{span}\left\{\mathbf{X}(i,:,k)\right\}_{\forall i,k},$$

$$R_3(\mathbf{X}) := \dim \text{fiberspan}(\mathbf{X}) := \dim \text{span}\left\{\mathbf{X}(i,j,:)\right\}_{\forall i,j}.$$

- We have $\max(R_1, R_2, R_3) \leq F$.
- Combining with our previous argument on upper bound, we have

$$\max(R_1, R_2, R_3) \leq F \leq \min(R_1R_2, R_2R_3, R_1R_3).$$

Typical, Generic, and Border Rank

- Consider a $2 \times 2 \times 2$ tensor $\mathbf{X}$ whose elements are i.i.d., drawn from the standard normal distribution $\mathcal{N}(0,1)$.
- The rank of $\mathbf{X}$ *over the real field*, i.e., when we consider

$$\mathbf{X} = \sum_{f=1}^{F} \mathbf{a}_f \circledcirc \mathbf{b}_f \circledcirc \mathbf{c}_f, \quad \mathbf{a}_f \in \mathbb{R}^{2\times 1}, \mathbf{b}_f \in \mathbb{R}^{2\times 1}, \mathbf{c}_f \in \mathbb{R}^{2\times 1}, \forall f$$

  is

$$\text{rank}(\mathbf{X}) = \begin{cases} 2, & \text{with probability } \frac{\pi}{4} \\ 3, & \text{with probability } 1 - \frac{\pi}{4} \end{cases}$$

- The rank of the same $\mathbf{X}$ is 2 with probability 1 when decomposition over the complex field.
- As another example, for $\mathbf{X} = \text{randn}(3,3,2)$,

$$\text{rank}(\mathbf{X}) = \begin{cases} 3, & \text{with probability } \frac{1}{2} \\ 4, & \text{with probability } \frac{1}{2} \end{cases}, \quad \text{over } \mathbb{R};$$

$$3, \quad \text{with probability } 1, \quad \text{over } \mathbb{C}.$$

Typical & Generic Rank

- Consider the $2 \times 2 \times 2$ case and denote $\mathbf{S}_1 := \mathbf{X}(:,:,1)$ and $\mathbf{S}_2 := \mathbf{X}(:,:,2)$.
- For $\mathbf{X}$ to have $\text{rank}(\mathbf{X}) = 2$, we must be able to express these slabs as

$$\mathbf{S}_1 = \mathbf{A}\mathbf{D}_1(\mathbf{C})\mathbf{B}^T, \text{ and } \mathbf{S}_2 = \mathbf{A}\mathbf{D}_2(\mathbf{C})\mathbf{B}^T,$$

  for some 2×2 real or complex matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$.
- If $\mathbf{X} = \text{randn}(2,2,2)$, $\mathbf{S}_1$ and $\mathbf{S}_2$ are nonsingular almost surely.

Typical & Generic Rank

- $\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}_1(\mathbf{C})$, and $\mathbf{D}_2(\mathbf{C})$ must all be nonsingular too.
- Denoting $\tilde{\mathbf{A}} := \mathbf{A}\mathbf{D}_1(\mathbf{C})$, $\mathbf{D} := (\mathbf{D}_1(\mathbf{C}))^{-1}\mathbf{D}_2(\mathbf{C}) \Leftrightarrow \mathbf{B}^T = (\tilde{\mathbf{A}})^{-1}\mathbf{S}_1$, and $\mathbf{S}_2 = \tilde{\mathbf{A}}\mathbf{D}(\tilde{\mathbf{A}})^{-1}\mathbf{S}_1$, leading to

$$\mathbf{S}_2\mathbf{S}_1^{-1} = \tilde{\mathbf{A}}\mathbf{D}(\tilde{\mathbf{A}})^{-1}.$$

- For rank($\mathbf{X}$) $= 2$ over $\mathbb{R}$, the matrix $\mathbf{S}_2\mathbf{S}_1^{-1}$ should have two **real** eigenvalues.
- But complex conjugate eigenvalues do arise with positive probability.

Typical & Generic Rank

- We see that the rank of a tensor for decomposition over $\mathbb{R}$ is a random variable that can take more than one value with positive probability.
- These values are called **typical ranks**.
- For decomposition over $\mathbb{C}$ the situation is different: rank(randn(2,2,2)) $= 2$ with probability 1.
- When there is only one typical rank (that occurs with probability 1 then) we call it **generic rank**.

Border Rank

- Consider $\mathbf{X} = \mathbf{u} \circledcirc \mathbf{u} \circledcirc \mathbf{v} + \mathbf{u} \circledcirc \mathbf{v} \circledcirc \mathbf{u} + \mathbf{v} \circledcirc \mathbf{u} \circledcirc \mathbf{u}$, where $||\mathbf{u}|| = ||\mathbf{v}|| = 1$, with $|\mathbf{u}^T\mathbf{v}| \neq 1$.
- Consider

$$\begin{aligned}\mathbf{X}_n &= n(\mathbf{u} + \frac{1}{n}\mathbf{v}) \circledcirc (\mathbf{u} + \frac{1}{n}\mathbf{v}) \circledcirc (\mathbf{u} + \frac{1}{n}\mathbf{v}) - n\mathbf{u} \circledcirc \mathbf{u} \circledcirc \mathbf{u} \\ &= \mathbf{u} \circledcirc \mathbf{u} \circledcirc \mathbf{v} + \mathbf{u} \circledcirc \mathbf{v} \circledcirc \mathbf{u} + \mathbf{v} \circledcirc \mathbf{u} \circledcirc \mathbf{u} + \\ &+ \frac{1}{n}\mathbf{v} \circledcirc \mathbf{v} \circledcirc \mathbf{u} + + \frac{1}{n}\mathbf{u} \circledcirc \mathbf{v} \circledcirc \mathbf{v} + \frac{1}{n^2}\mathbf{v} \circledcirc \mathbf{v} \circledcirc \mathbf{v},\end{aligned}$$

  so $\mathbf{X}_n = \mathbf{X} +$ terms that vanish as $n \to \infty$.
- $\mathbf{X}$ has rank equal to 3, but **border rank** equal to 2 [Com14].
- The above example shows the following is ill-posed in general:

$$\min_{\{\mathbf{a}_f, \mathbf{b}_f, \mathbf{c}_f\}_{f=1}^F} \left|\left| \mathbf{X} - \sum_{f=1}^F \mathbf{a}_f \circledcirc \mathbf{b}_f \circledcirc \mathbf{c}_f \right|\right|_F^2.$$

Comments

- For a tensor of a given size, there is always one typical rank over $\mathbb{C}$, which is therefore generic.
- For $I_1 \times I_2 \times \cdots \times I_N$ tensors, this generic rank is the value

$$\left\lceil \frac{\prod_{n=1}^{N} I_n}{\sum_{n=1}^{N} I_n - N + 1} \right\rceil$$

except for the so-called defective cases:

- (i) $I_1 > \prod_{n=2}^{N} I_n - \sum_{n=2}^{N} (I_n - 1)$
- (ii) the third-order case of dimension $(4, 4, 3)$
- (iii) the third-order cases of dimension $(2p + 1, 2p + 1, 3)$, $p \in \mathbb{N}$
- (iv) the 4th-order cases of dimension $(p, p, 2, 2)$, $p \in \mathbb{N}$

Comments - Cont'd

- Typical rank may change when the tensor is constrained in some way; e.g., when the frontal slabs are symmetric.
- Consider, for example, a fully symmetric tensor, i.e., one such that $\mathbf{X}(i,j,k) = \mathbf{X}(i,k,j) = \mathbf{X}(j,i,k) = \mathbf{X}(j,k,i) = \mathbf{X}(k,i,j) = \mathbf{X}(k,j,i)$.
- Then the **symmetric rank** of N-way $\mathbf{X}$ over $\mathbb{C}$ is defined as the minimum R such that $\mathbf{X} = \sum_{r=1}^{R} \mathbf{a}_r \circledcirc \mathbf{a}_r \circledcirc \cdots \circledcirc \mathbf{a}_r$.
- It has been shown that this symmetric rank equals $\lceil \binom{I+N-1}{N} / I \rceil$ almost surely except in the defective cases $(N, I) = (3,5), (4,3), (4,4), (4,5)$, where it is 1 higher [AH95].
- Taking $N = 3$ as a special case, this formula gives $\frac{(I+1)(I+2)}{6}$.
- We also remark that constraints such as nonnegativity of a factor matrix can strongly affect rank.

Tensor Rank

- Maximal and typical ranks for decomp. over $\mathbb{R}$ [KB09, Lan12].

Table: Maximum attainable rank over $\mathbb{R}$.

Size	Maximum attainable rank over $\mathbb{R}$
$I \times J \times 2$	$\min(I, J) + \min(I, J, \lfloor \max(I, J)/2 \rfloor)$
$2 \times 2 \times 2$	3
$3 \times 3 \times 3$	5

Table: Typical rank over $\mathbb{R}$

Size	Typical ranks over $\mathbb{R}$
$I \times I \times 2$	$\{I, I+1\}$
$I \times J \times 2$, $I > J$	$\min(I, 2J)$
$I \times J \times K$, $I > JK$	JK

Table: Symmetry may affect typical rank.

Size	Typical ranks, $\mathbb{R}$ partial symmetry	Typical ranks, $\mathbb{R}$ no symmetry
$I \times I \times 2$	$\{I, I+1\}$	$\{I, I+1\}$
$9 \times 3 \times 3$	6	9

Tensor Rank in Practice

- Given a particular tensor $\mathbf{X}$, determining rank($\mathbf{X}$) is NP-hard [Hås90].
- Is this an issue in practical applications of tensor decomposition?
- In applications, one is really interested in fitting a model that has the "essential" or "meaningful" number of components – "signal rank".
- Determining the signal rank is challenging, even in the matrix case.
- There exist heuristics that can help.
- ... but, at the end of the day, the process generally involves some trial-and-error.

Tensors as Operators – Rank Matters

- Consider $\mathbf{M}_1 \times \mathbf{M}_2$, where $\mathbf{M}_1$ and $\mathbf{M}_2$ are both 2×2 matrices.
- A naive implementation of $\mathbf{P} = \mathbf{M}_1\mathbf{M}_2$ requires 8 multiplications.
- The rank of a tensor can give an upper bound of the number of multiplications that is needed.
- Define $[\text{vec}(\mathbf{P})]_k = \text{vec}(\mathbf{M}_1)^T\mathbf{X}_k\text{vec}(\mathbf{M}_2)$, e.g.,

$$\mathbf{X}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

then $\text{vec}(\mathbf{M}_1)^T\mathbf{X}_k\text{vec}(\mathbf{M}_2) = \mathbf{P}(1,1)$.

Tensors as Operators – Rank Matters

- Assume that $\mathbf{X}$ has a decomposition $\mathbf{X}_k = \mathbf{A}\mathbf{D}_k(\mathbf{C})\mathbf{B}^T$ with rank F.
- Any element of $\mathbf{P}$ can be written as $\text{vec}(\mathbf{M}_1)^T\mathbf{A}\mathbf{D}_k(\mathbf{C})\mathbf{B}^T\text{vec}(\mathbf{M}_2)$.
- $\mathbf{B}^T\text{vec}(\mathbf{M}_2)$ can be computed using F inner products, and the same is true for $\text{vec}(\mathbf{M}_1)^T\mathbf{A}$.
- If the elements of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ take values in $\{0, \pm 1\}$, then these inner products require no multiplication.
- Letting $\mathbf{u}^T := \text{vec}(\mathbf{M}_1)^T\mathbf{A}$ and $\mathbf{v} := \mathbf{B}^T\text{vec}(\mathbf{M}_2)$, it remains to compute $\mathbf{u}^T\mathbf{D}_k(\mathbf{C})\mathbf{v} = \sum_{f=1}^{F} \mathbf{u}(f)\mathbf{v}(f)\mathbf{C}(k,f)$, $\forall k \in \{1,2,3,4\}$.

Tensor as Operator – Rank Matters

- F multiplications to compute the products $\{\mathbf{u}(f)\mathbf{v}(f)\}_{f=1}^{F}$.
- The rest is all selections, additions, subtractions if $\mathbf{C}$ takes values in $\{0, \pm 1\}$.
- The rank of Strassen's $4 \times 4 \times 4$ tensor is 7, so $F = 7$ suffices.
- Contrast to the "naive" approach which entails $F = 8$ multiplications.

Are Matrix Unfoldings Rank-Revealing?

- Recall we have three unfoldings

$$\mathbf{X}_1 = (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T, \quad \mathbf{X}_2 = (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T, \quad \mathbf{X}_3 = (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T$$

- If $\text{rank}(\mathbf{C} \odot \mathbf{B}) = \text{rank}(\mathbf{A}) = F$, then $\text{rank}(\mathbf{X}_1) = F = \text{rank}(\mathbf{X})$.
- For this to happen it is *necessary* (but not sufficient) that $JK \geq F$, and $I \geq F$, so F has to be small: $F \leq \min(I, JK)$.
- It follows that $F \leq \max(\min(I, JK), \min(J, IK), \min(K, IJ))$ is necessary to have a rank-revealing matricization of the tensor.
- However, we know that the (perhaps unattainable) upper bound on $F = \text{rank}(\mathbf{X})$ is $F \leq \min(IJ, JK, IK)$.
- In more general (and more interesting cases) of tensor factorization,

$$F = \text{rank}(\mathbf{X}) \geq \max(\text{rank}(\mathbf{X}_1), \text{rank}(\mathbf{X}_2), \text{rank}(\mathbf{X}_3)).$$

Going to Higher-Order

- Let us start with 4-way tensors:

$$\mathbf{X}(i,j,k,\ell) = \sum_{f=1}^{F} \mathbf{a}_f(i)\mathbf{b}_f(j)\mathbf{c}_f(k)\mathbf{e}_f(\ell), \forall \begin{cases} i \in \{1,\cdots,I\} \\ j \in \{1,\cdots,J\} \\ k \in \{1,\cdots,K\} \\ \ell \in \{1,\cdots,L\} \end{cases}$$

$$\text{or, equivalently } \mathbf{X} = \sum_{f=1}^{F} \mathbf{a}_f \circledcirc \mathbf{b}_f \circledcirc \mathbf{c}_f \circledcirc \mathbf{e}_f.$$

- Upon defining $\mathbf{A} := [\mathbf{a}_1, \cdots, \mathbf{a}_F]$, $\mathbf{B} := [\mathbf{b}_1, \cdots, \mathbf{b}_F]$, $\mathbf{C} := [\mathbf{c}_1, \cdots, \mathbf{c}_F]$, $\mathbf{E} := [\mathbf{e}_1, \cdots, \mathbf{e}_F]$, we may also write

$$\mathbf{X}(i,j,k,\ell) = \sum_{f=1}^{F} \mathbf{A}(i,f)\mathbf{B}(j,f)\mathbf{C}(k,f)\mathbf{E}(\ell,f),$$

and we sometimes also use $\mathbf{X}(i,j,k,\ell) = \sum_{f=1}^{F} a_{i,f}b_{j,f}c_{k,f}e_{\ell,f}$.

Going to Higher-Order

- Now consider $\mathbf{X}(:,:,:,1)$, which is a third-order tensor:

$$\mathbf{X}(i,j,k,1) = \sum_{f=1}^{F} a_{i,f} b_{j,f} c_{k,f} e_{1,f}.$$

- let us vectorize $\mathbf{X}(:,:,:,1)$ into an $IJK \times 1$ vector

$$\text{vec}\left(\text{vec}\left(\mathbf{X}(:,:,:,1)\right)\right) = (\mathbf{C} \odot \mathbf{B} \odot \mathbf{A})(\mathbf{E}(1,:))^T.$$

- Stacking one next to each other the vectors corresponding to $\mathbf{X}(:,:,:,1)$, $\mathbf{X}(:,:,:,2)$, $\cdots$, $\mathbf{X}(:,:,:,L)$, we obtain $(\mathbf{C} \odot \mathbf{B} \odot \mathbf{A})\mathbf{E}^T$; and after one more vec$(\cdot)$ we get $(\mathbf{E} \odot \mathbf{C} \odot \mathbf{B} \odot \mathbf{A})\mathbf{1}$.

Going to Higher-Order

- $(\mathbf{E} \odot \mathbf{C} \odot \mathbf{B} \odot \mathbf{A})\mathbf{1} = ((\mathbf{E} \odot \mathbf{C}) \odot (\mathbf{B} \odot \mathbf{A}))\mathbf{1} = \text{vec}\left((\mathbf{B} \odot \mathbf{A})(\mathbf{E} \odot \mathbf{C})^T\right)$:
- "Balanced" matricization of the 4th-order tensor:

$$\mathbf{X}_b = (\mathbf{B} \odot \mathbf{A})(\mathbf{E} \odot \mathbf{C})^T.$$

- $\mathbf{X}_b$ is *rank-revealing* means $F \leq \min(IJK, IJL, IKL, JKL)$
- Looks better than the 3-order case? - but the rank upper bound is also much higher.
- In short: matricization can reveal tensor rank in low-rank cases only.

Going to Higher-Order

- For a general N-way tensor, we can write it in scalar form as

$$\mathbf{X}(i_1, \cdots, i_N) = \sum_{f=1}^{F} \mathbf{a}_f^{(1)}(i_1) \cdots \mathbf{a}_f^{(N)}(i_N) = \sum_{f=1}^{F} a_{i_1,f}^{(1)} \cdots a_{i_N,f}^{(N)},$$

  and in (combinatorially!) many different ways, including

$$\mathbf{X}_N = (\mathbf{A}_{N-1} \odot \cdots \odot \mathbf{A}_1)\mathbf{A}_N^T \rightarrow \text{vec}(\mathbf{X}_N) = (\mathbf{A}_N \odot \cdots \odot \mathbf{A}_1)\mathbf{1}.$$

- We sometimes also use the shorthand $\text{vec}(\mathbf{X}_N) = \left(\odot_{n=N}^{1} \mathbf{A}_n\right) \mathbf{1}$.

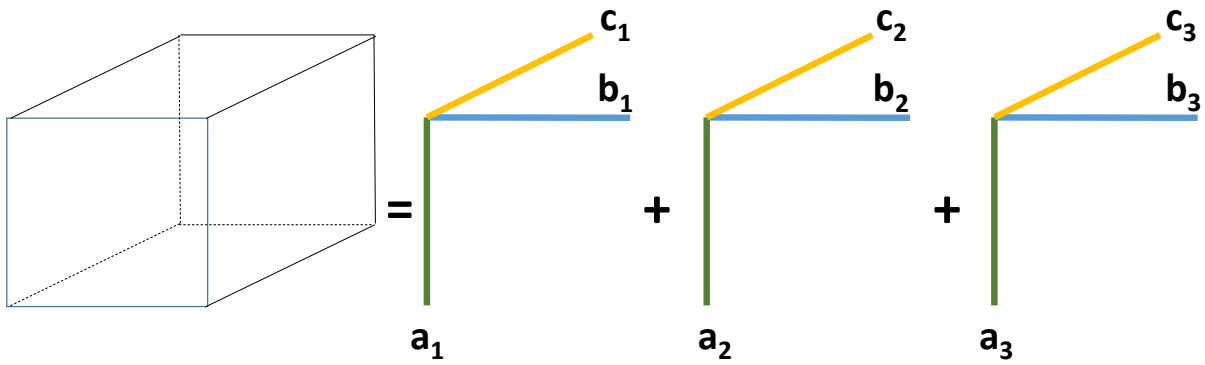
Reprise – what's coming next

- CPD: Uniqueness, demystified
- Tucker
- MLSVD: Properties, analogies, computation
- Links between CPD and Tucker, MLSVD
- Other models:
  - Tensor Trains
  - Hierarchical Tucker
  - PARALIND and Block-Term Decomposition
  - Coupled Decompositions

Reprise

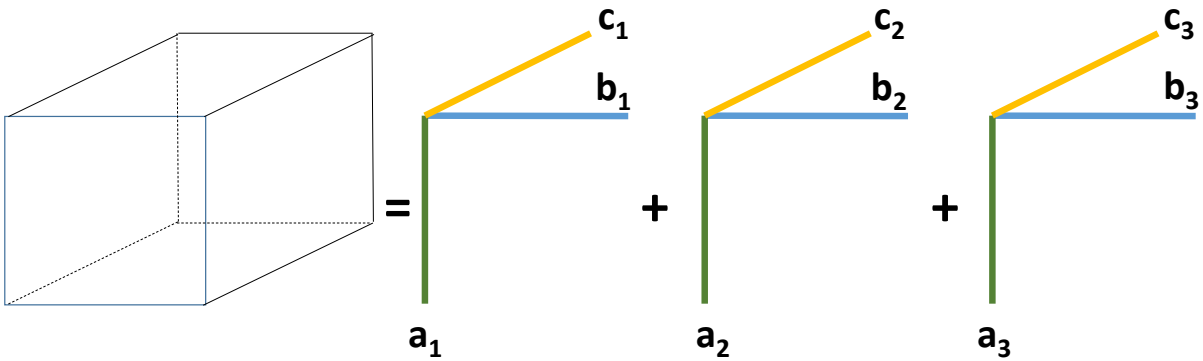
- Most significant advantage of tensors vs. matrices: low-rank tensor decomposition (essentially) unique for rank $\gg 1$, even rank $>$ #rows, columns, fibers.
- For matrices, only true for rank $= 1$ – not interesting, in most cases.
- But why tensors (of order ≥ 3) are so different may seem like a mystery ...
- Phase transition between second-order (matrices) and third- or higher-order tensors.
- Let's shed some light into this phenomenon.

Essential uniqueness



- Given tensor $\mathbf{X}$ of rank F, its CPD is *essentially unique* iff the F rank-1 terms in its decomposition (the outer products or "chicken feet") are unique;
- i.e., there is no other way to decompose $\mathbf{X}$ for the given number of terms.
- Can of course permute "chicken feet" without changing their sum $\rightarrow$ permutation ambiguity.
- Can scale $\mathbf{a}_1$ by α and counter-scale $\mathbf{b}_1$ (or $\mathbf{c}_1$) by $\frac{1}{\alpha}$.

Essential uniqueness



- If $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, with $\mathbf{A}: \ I \times F$, $\mathbf{B}: \ J \times F$, and $\mathbf{C}: \ K \times F$, then essential uniqueness means that $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are unique up to a common permutation and scaling / counter-scaling of columns.
- Meaning that if $\mathbf{X} = [\![\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}]\!]$, for some $\bar{\mathbf{A}}: \ I \times F$, $\bar{\mathbf{B}}: \ J \times F$, and $\bar{\mathbf{C}}: \ K \times F$, then there exists a permutation matrix $\mathbf{\Pi}$ and diagonal scaling matrices $\mathbf{\Lambda}_1, \mathbf{\Lambda}_2, \mathbf{\Lambda}_3$ such that

$$\bar{\mathbf{A}} = \mathbf{A\Pi\Lambda}_1, \ \bar{\mathbf{B}} = \mathbf{B\Pi\Lambda}_2, \ \bar{\mathbf{C}} = \mathbf{C\Pi\Lambda}_3, \ \mathbf{\Lambda}_1\mathbf{\Lambda}_2\mathbf{\Lambda}_3 = \mathbf{I}.$$

Simple proof of uniqueness

- Consider $I \times J \times 2$ tensor $\mathbf{X}$ of rank $F \leq \min(I, J)$.

$$\mathbf{X}^{(1)} = \mathbf{X}(:, :, 1) = \mathbf{A}\mathbf{D}_1(\mathbf{C})\mathbf{B}^T,$$

$$\mathbf{X}^{(2)} = \mathbf{X}(:, :, 2) = \mathbf{A}\mathbf{D}_2(\mathbf{C})\mathbf{B}^T,$$

- Assume, for the moment, no zero element on the diagonals.
- $\widetilde{\mathbf{A}} := \mathbf{A}\mathbf{D}_1(\mathbf{C})$, $\mathbf{D} := (\mathbf{D}_1(\mathbf{C}))^{-1}\mathbf{D}_2(\mathbf{C})$.
- Then, $\mathbf{X}^{(1)} = \widetilde{\mathbf{A}}\mathbf{B}^T$, $\mathbf{X}^{(2)} = \widetilde{\mathbf{A}}\mathbf{D}\mathbf{B}^T$, or

$$\begin{bmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{bmatrix} = \begin{bmatrix} \widetilde{\mathbf{A}} \\ \widetilde{\mathbf{A}}\mathbf{D} \end{bmatrix} \mathbf{B}^T.$$

Simple proof of uniqueness

- $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] = \text{svd}\left(\begin{bmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{bmatrix}\right)$, i.e., $\begin{bmatrix} \mathbf{X}^{(1)} \\ \mathbf{X}^{(2)} \end{bmatrix} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$
- Assuming $\text{rank}\left(\mathbf{X}^{(1)}\right) = \text{rank}\left(\mathbf{X}^{(2)}\right) = F$ ($\Rightarrow$ rank of all matrices is F) $\rightarrow$

$$\mathbf{U} = \begin{bmatrix} \mathbf{U}_1 \\ \mathbf{U}_2 \end{bmatrix} = \begin{bmatrix} \widetilde{\mathbf{A}} \\ \widetilde{\mathbf{A}}\mathbf{D} \end{bmatrix} \mathbf{M} = \begin{bmatrix} \widetilde{\mathbf{A}}\mathbf{M} \\ \widetilde{\mathbf{A}}\mathbf{D}\mathbf{M} \end{bmatrix},$$

  where matrix $\mathbf{M}$ is $F \times F$ nonsingular.
- Compute auto- and cross-correlation

$$\mathbf{R}_1 = \mathbf{U}_1^T\mathbf{U}_1 = \mathbf{M}^T\widetilde{\mathbf{A}}^T\widetilde{\mathbf{A}}\mathbf{M} =: \mathbf{Q}\mathbf{M},$$

$$\mathbf{R}_2 = \mathbf{U}_1^T\mathbf{U}_2 = \mathbf{M}^T\widetilde{\mathbf{A}}^T\widetilde{\mathbf{A}}\mathbf{D}\mathbf{M} = \mathbf{Q}\mathbf{D}\mathbf{M}.$$

- Both $\mathbf{R}_1$ and $\mathbf{R}_2$ are $F \times F$ nonsingular.

Simple proof of uniqueness

- What have we accomplished so far?
- Obtained equations involving square nonsingular matrices (instead of possibly tall, full column rank ones). Key step comes next:

$$\left(\mathbf{R}_1^{-1}\mathbf{R}_2\right)\mathbf{M}^{-1} = \mathbf{M}^{-1}\mathbf{D}$$

- $\mathbf{M}^{-1}$ holds eigenvectors of $\left(\mathbf{R}_1^{-1}\mathbf{R}_2\right)$, and $\mathbf{D}$ holds eigenvalues (assumed to be distinct, for the moment).
- $\exists$ freedom to scale eigenvectors (remain eigenvectors), and obviously one cannot recover the order of the columns of $\mathbf{M}^{-1}$.
- $\rightarrow$ permutation and scaling ambiguity in recovering $\mathbf{M}^{-1}$ from eigendecomposition of $\left(\mathbf{R}_1^{-1}\mathbf{R}_2\right)$.

Simple proof of uniqueness

- What we do recover is actually $\widetilde{\mathbf{M}}^{-1} = \mathbf{M}^{-1}\mathbf{\Pi}\mathbf{\Lambda}$, where $\mathbf{\Pi}$ is a permutation matrix and $\mathbf{\Lambda}$ is a nonsingular diagonal scaling matrix.
- If we use $\widetilde{\mathbf{M}}^{-1}$ to recover $\widetilde{\mathbf{A}}$ from equation $\mathbf{U}_1 = \widetilde{\mathbf{A}}\mathbf{M} \Rightarrow$ $\widetilde{\mathbf{A}} = \mathbf{U}_1\mathbf{M}^{-1}$, we will in fact recover $\widetilde{\mathbf{A}}\mathbf{\Pi}\mathbf{\Lambda}$.
- That is, $\widetilde{\mathbf{A}}$ up to the same column permutation and scaling that stem from the ambiguity in recovering $\mathbf{M}^{-1}$.
- Now easy to see that we can recover $\mathbf{B}$ and $\mathbf{C}$ by going back to the original equations for $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ and left-inverting $\mathbf{A}$.

Simple proof of uniqueness

- During the course of the derivation, made assumptions in passing:
  - that the slabs of $\mathbf{X}$ have rank $F = \text{rank}(\mathbf{X})$, and
  - that the eigenvalues in $\mathbf{D}$ are distinct ($\Rightarrow$ one row of $\mathbf{C}$ has no zero elements).
- Next we revisit those, starting from the last one, and show that they can be made WLOG.
- First note that $\left(\mathbf{R}_1^{-1}\mathbf{R}_2\right)$ is diagonalizable (i.e., has a full set of linearly independent eigenvectors) *by construction* under our working assumptions.
- If two or more of its eigenvalues are identical though, then linear combinations of the corresponding eigenvectors are also eigenvectors, corresponding to the same eigenvalue. Hence distinct eigenvalues (elements of $\mathbf{D}$) are *necessary* for uniqueness.

Simple proof of uniqueness

- Consider creating two random slab mixtures

$$\widetilde{\mathbf{X}}^{(1)} = \gamma_{1,1}\mathbf{X}^{(1)} + \gamma_{1,2}\mathbf{X}^{(2)} =$$

$$\mathbf{A}\left(\gamma_{1,1}\mathbf{D}_1(\mathbf{C}) + \gamma_{1,2}\mathbf{D}_2(\mathbf{C})\right)\mathbf{B}^T,$$

$$\widetilde{\mathbf{X}}^{(2)} = \gamma_{2,1}\mathbf{X}^{(1)} + \gamma_{2,2}\mathbf{X}^{(2)} =$$

$$\mathbf{A}\left(\gamma_{2,1}\mathbf{D}_1(\mathbf{C}) + \gamma_{2,2}\mathbf{D}_2(\mathbf{C})\right)\mathbf{B}^T.$$

- Net effect $\mathbf{C} \leftarrow \widetilde{\mathbf{C}} := \mathbf{\Gamma}\mathbf{C}$, and we draw $\mathbf{\Gamma} := \begin{bmatrix} \gamma_{1,1} & \gamma_{1,2} \\ \gamma_{2,1} & \gamma_{2,2} \end{bmatrix}$ from i.i.d. $\mathcal{U}[0,1]$.
- All elements of $\widetilde{\mathbf{C}} \neq 0$, rank $\left(\widetilde{\mathbf{X}}^{(1)}\right) =$ rank $\left(\widetilde{\mathbf{X}}^{(2)}\right) = F$, almost surely.
- Any two columns of $\widetilde{\mathbf{C}} = \mathbf{\Gamma} \times$ corresponding two columns of $\mathbf{C} \Rightarrow$ distinct ratios $\widetilde{\mathbf{C}}(1,:)/\widetilde{\mathbf{C}}(2,:)$ a.s. iff *any two* columns of $\mathbf{C}$ are linearly independent – $\mathbf{C}$ has Kruskal rank ≥ 2.

Simple proof of uniqueness

- We have proven

Theorem

Given $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, *with* $\mathbf{A}: I \times F$, $\mathbf{B}: J \times F$, *and* $\mathbf{C}: 2 \times F$, *if* $F > 1$ *it is necessary for uniqueness of* $\mathbf{A}$, $\mathbf{B}$ *that* $k_{\mathbf{C}} = 2$. *If, in addition* $r_{\mathbf{A}} = r_{\mathbf{B}} = F$, *then rank*$(\mathbf{X}) = F$ *and the decomposition of* $\mathbf{X}$ *is essentially unique.*

- For $K \geq 2$ slices, consider two random slice mixtures to obtain

Theorem

Given $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, *with* $\mathbf{A}: I \times F$, $\mathbf{B}: J \times F$, *and* $\mathbf{C}: K \times F$, *if* $F > 1$ *it is necessary for uniqueness of* $\mathbf{A}$, $\mathbf{B}$ *that* $k_{\mathbf{C}} \geq 2$. *If, in addition* $r_{\mathbf{A}} = r_{\mathbf{B}} = F$, *then rank*$(\mathbf{X}) = F$ *and the decomposition of* $\mathbf{X}$ *is essentially unique.*

Intermediate result conveying flavor of Kruskal's

Theorem

Given $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, *with* $\mathbf{A}: I \times F$, $\mathbf{B}: J \times F$, *and* $\mathbf{C}: K \times F$, *it is necessary for uniqueness of* $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ *that*

$$\min(r_{\mathbf{A}\odot\mathbf{B}}, r_{\mathbf{B}\odot\mathbf{C}}, r_{\mathbf{C}\odot\mathbf{A}}) = F. \tag{1}$$

If $F > 1$, *then it is also necessary that*

$$\min(k_{\mathbf{A}}, k_{\mathbf{B}}, k_{\mathbf{C}}) \geq 2. \tag{2}$$

If, in addition,

$$r_{\mathbf{C}} = F, \tag{3}$$

and

$$k_{\mathbf{A}} + k_{\mathbf{B}} \geq F + 2, \tag{4}$$

then the decomposition is essentially unique.

Intermediate result conveying flavor of Kruskal's

- Necessary conditions: consider

$$\mathbf{X}^{(JI\times K)} = (\mathbf{A} \odot \mathbf{B})\,\mathbf{C}^T, \tag{5}$$

$$\mathbf{X}^{(IK\times J)} = (\mathbf{C} \odot \mathbf{A})\mathbf{B}^T, \tag{6}$$

$$\mathbf{X}^{(KJ\times I)} = (\mathbf{B} \odot \mathbf{C})\mathbf{A}^T. \tag{7}$$

- Need the KRPs to be fcr, else can add a vector in the right null space of that Khatri-Rao product to any of the rows of the corresponding third matrix without affecting the data.
- Consider the $F = 2$ case: if one can mix two rank-1 factors [meaning: use two linear combinations of the two factors instead of the pure factors themselves] without affecting their contribution to the data, then the model is not unique, irrespective of the remaining factors.

Intermediate result conveying flavor of Kruskal's

- Consider $I \times 2$, $J \times 2$, $K \times 2$ case, assume WLOG that $k_{\mathbf{A}} = 1$.
- This means that the two columns of $\mathbf{A}$ are collinear, i.e.,

$$x_{i,j,k} = a_{i,1}(b_{j,1}c_{k,1} + \lambda b_{j,2}c_{k,2}),$$

  which implies that the i-th slab of $\mathbf{X}$ along the first mode is given by

$$\mathbf{X}_i = a_{i,1}\bar{\mathbf{B}}\mathbf{C}^T, \ i = 1, \cdots, I,$$

  where $\bar{\mathbf{B}} = [\mathbf{b}_1 \ \lambda\mathbf{b}_2]$, and $\mathbf{C} = [\mathbf{c}_1 \ \mathbf{c}_2]$.
- Therefore, slabs along the first mode are multiples of each other;
- $\mathbf{a}_1 = \left[a_{1,1}, \cdots, a_{I,1}\right]^T$ uniquely determined up to global scaling ($\mathbf{A}$ determined up to scaling of its columns)
- ... but $\exists$ linear transformation freedom in choosing $\mathbf{B}$ and $\mathbf{C}$.
- Tensor $\mathbf{X}$ comprises only copies of matrix $\mathbf{X}_1$; same as matrix decomposition.

Intermediate result conveying flavor of Kruskal's

- Sufficiency will be shown by contradiction.
- WLOG assume $r_{\mathbf{C}} = F$ (implies that $\mathbf{C}$ is tall or square), and $r_{\mathbf{A} \odot \mathbf{B}} = F$.
- Suffices to consider square $\mathbf{C}$, for otherwise $\mathbf{C}$ contains a square submatrix consisting of linearly independent rows.
- Amounts to discarding the remaining rows of $\mathbf{C}$, or, equivalently, dispensing with certain data slabs along third mode.
- Suffices to prove that the parameterization of $\mathbf{X}$ in terms of $\mathbf{A}$, $\mathbf{B}$, and the row-truncated $\mathbf{C}$ is unique based on part of the data. The uniqueness of the full $\mathbf{C}$ then follows trivially.
- Will use the following elementary fact, which is a very special case of the *Permutation Lemma* in [Kruskal, '77].

Simple case of Kruskal's Permutation Lemma

- Let w($\mathbf{v}$) denote the number of nonzero elements (the *weight*) of $\mathbf{v} \in \mathbb{C}^K$.
- Consider two $F \times F$ nonsingular matrices $\mathbf{C}$ and $\bar{\mathbf{C}}$. Suppose that

$$\text{w}(\mathbf{v}^T\mathbf{C}) = 1, \ \forall \mathbf{v} \ \mid \ \text{w}(\mathbf{v}^T\bar{\mathbf{C}}) = 1. \tag{8}$$

- Meaning: for all $\mathbf{v}$ such that w($\mathbf{v}^T\bar{\mathbf{C}}$) $= 1$, it holds that w($\mathbf{v}^T\mathbf{C}$) $= 1$ as well.
- It then follows that $\bar{\mathbf{C}} = \mathbf{C\Pi\Lambda}$, where $\mathbf{\Pi}$ is a permutation matrix, and $\mathbf{\Lambda}$ is a nonsingular diagonal scaling matrix.

Insight: simple case of Kruskal's Permutation Lemma

- For a proof, note that if condition (8) holds, then

$$\bar{\mathbf{C}}^{-1}\bar{\mathbf{C}} = \mathbf{I} \Longrightarrow \bar{\mathbf{C}}^{-1}\mathbf{C} = \mathbf{\Pi}^T\mathbf{D},$$

  where $\mathbf{D}$ is a nonsingular diagonal matrix, and we have used that the product $\bar{\mathbf{C}}^{-1}\mathbf{C}$ is full rank, and its rows have weight one.
- It then follows that

$$\mathbf{C} = \bar{\mathbf{C}}\mathbf{\Pi}^T\mathbf{D} \Longleftrightarrow \bar{\mathbf{C}} = \mathbf{C}\mathbf{D}^{-1}\mathbf{\Pi} = \mathbf{C}\mathbf{\Pi}\mathbf{\Lambda}.$$

Intermediate result conveying flavor of Kruskal's

- Suppose $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\mathbf{C}}]\!]$. From (5), it follows that

$$(\mathbf{A} \odot \mathbf{B})\,\mathbf{C}^T = \mathbf{X}^{(JI \times K)} = (\bar{\mathbf{A}} \odot \bar{\mathbf{B}})\,\bar{\mathbf{C}}^T. \tag{9}$$

- Since $r_{\mathbf{A}\odot\mathbf{B}} = r_{\mathbf{C}} = F$, it follows that $r_{\bar{\mathbf{A}}\odot\bar{\mathbf{B}}} = r_{\bar{\mathbf{C}}} = F$.
- Taking linear combinations of the slabs along the third mode,

$$\sum_{k=1}^{K} v_k \mathbf{X}(:,:,k) = \mathbf{A}\,diag(\mathbf{v}^T\mathbf{C})\mathbf{B}^T = \bar{\mathbf{A}}\,diag(\mathbf{v}^T\bar{\mathbf{C}})\bar{\mathbf{B}}^T, \tag{10}$$

for all $\mathbf{v} := [v_1, \cdots, v_F]^T \in \mathbb{C}^F$.

Intermediate result conveying flavor of Kruskal's

- The rank of a matrix product is always less than or equal to the rank of any factor, and thus

$$\mathrm{w}(\mathbf{v}^T\bar{\mathbf{C}}) = r_{diag(\mathbf{v}^T\bar{\mathbf{C}})} \geq r_{\bar{\mathbf{A}}diag(\mathbf{v}^T\bar{\mathbf{C}})\bar{\mathbf{B}}^T} = r_{\mathbf{A}diag(\mathbf{v}^T\mathbf{C})\mathbf{B}^T}. \tag{11}$$

- Assume $\mathrm{w}(\mathbf{v}^T\bar{\mathbf{C}}) = 1$; then (11) implies $r_{\mathbf{A}diag(\mathbf{v}^T\mathbf{C})\mathbf{B}^T} \leq 1$, and we wish to show that $\mathrm{w}(\mathbf{v}^T\mathbf{C}) = 1$.
- Use shorthand $\mathrm{w} := \mathrm{w}(\mathbf{v}^T\mathbf{C})$. Using Sylvester's inequality and the definition of k-rank:

$$r_{\mathbf{A}diag(\mathbf{v}^T\mathbf{C})\mathbf{B}^T} \geq \min(k_{\mathbf{A}}, \mathrm{w}) + \min(k_{\mathbf{B}}, \mathrm{w}) - \mathrm{w}.$$

Intermediate result conveying flavor of Kruskal's

- Hence

$$\min(k_{\mathbf{A}}, \mathrm{w}) + \min(k_{\mathbf{B}}, \mathrm{w}) - \mathrm{w} \leq 1. \tag{12}$$

- Consider cases:
  1. Case of $\mathrm{w} \leq \min(k_{\mathbf{A}}, k_{\mathbf{B}})$: then (12) implies $\mathrm{w} \leq 1$, hence $\mathrm{w} = 1$, because $\mathbf{C}$ is nonsingular and $\mathbf{v} \neq \mathbf{0}$;
  2. Case of $\min(k_{\mathbf{A}}, k_{\mathbf{B}}) \leq \mathrm{w} \leq \max(k_{\mathbf{A}}, k_{\mathbf{B}})$: then (12) implies $\min(k_{\mathbf{A}}, k_{\mathbf{B}}) \leq 1$, which contradicts (2), thereby excluding this range of w from consideration;
  3. Case of $\mathrm{w} \geq \max(k_{\mathbf{A}}, k_{\mathbf{B}})$: then (12) implies that $\mathrm{w} \geq k_{\mathbf{A}} + k_{\mathbf{B}} - 1$. Under (4), however, this yields another contradiction, as it requires that $\mathrm{w} \geq F + 1$, which is impossible since the maximum possible $\mathrm{w} = \mathrm{w}(\mathbf{v}^T\mathbf{C})$ is F.

Intermediate result conveying flavor of Kruskal's

- We conclude that, under (1)-(2) and (3)-(4), $\mathrm{w}(\mathbf{v}^T\bar{\mathbf{C}}) = 1$ implies $\mathrm{w}(\mathbf{v}^T\mathbf{C}) = 1$. From the elementary version of the Permutation Lemma, it follows that $\bar{\mathbf{C}} = \mathbf{C\Pi\Lambda}$.
- From (9) we now obtain

$$\left[(\mathbf{A} \odot \mathbf{B}) - (\bar{\mathbf{A}} \odot \bar{\mathbf{B}})\,\mathbf{\Lambda\Pi}^T\right]\mathbf{C}^T = \mathbf{0},$$

and since $\mathbf{C}$ is nonsingular,

$$(\bar{\mathbf{A}} \odot \bar{\mathbf{B}}) = (\mathbf{A} \odot \mathbf{B})\,\mathbf{\Pi\Lambda}^{-1}. \tag{13}$$

Intermediate result conveying flavor of Kruskal's

- It follows that, for every column $\mathbf{a}_f \otimes \mathbf{b}_f$ of $\mathbf{A} \odot \mathbf{B}$ there exists a unique column $\bar{\mathbf{a}}_{f'} \otimes \bar{\mathbf{b}}_{f'}$ of $\bar{\mathbf{A}} \odot \bar{\mathbf{B}}$ such that

$$\mathbf{a}_f \otimes \mathbf{b}_f = \bar{\mathbf{a}}_{f'} \otimes \bar{\mathbf{b}}_{f'} \lambda_{f'} .$$

- It only remains to account for uniqueness of the truncated rows of a possibly tall $\mathbf{C}$, but this is now obvious from (9), (13), and (1). This completes the proof.

Full rank in one mode

- Assume only one of the loading matrices is full column rank, instead of two.

Theorem (Jiang & Sidiropoulos, '04)

Given $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, *with* $\mathbf{A}: I \times F$, $\mathbf{B}: J \times F$, *and* $\mathbf{C}: K \times F$, *and assuming* $r_{\mathbf{C}} = F$, *it holds that the decomposition* $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$ *is essentially unique* $\Longleftrightarrow$ *nontrivial linear combinations of columns of* $\mathbf{A} \odot \mathbf{B}$ *cannot be written as* $\otimes$ *product of two vectors.*

- Despite its conceptual simplicity and appeal, the above condition is hard to check.
- In [Jiang & Sidiropoulos, '04] it is shown that it is possible to recast this condition as an equivalent criterion on the solutions of a system of quadratic equations – which is also hard to check, ...
- but serves as a stepping stone to easier conditions and even generalizations of the EVD-based computation.

Full rank in one mode: generic uniqueness

Theorem (Chiantini & Ottaviani, 2012, Domanov & De Lathauwer, 2015, Strassen, 1983)

Given $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, *with* $\mathbf{A}: I \times F$, $\mathbf{B}: J \times F$, *and* $\mathbf{C}: K \times F$, *let* $K \geq F$ *and* $\min(I, J) \geq 3$. *Then rank*$(\mathbf{X}) = F$ *and the decomposition of* $\mathbf{X}$ *is essentially unique, almost surely, if and only if* $(I-1)(J-1) \geq F$.

Kruskal

- Most well-known result covered by [Kruskal '77]:

Theorem (Kruskal '77)

Given $\mathbf{X} = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$, *with* $\mathbf{A}: I \times F$, $\mathbf{B}: J \times F$, *and* $\mathbf{C}: K \times F$, *if* $k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \geq 2F + 2$, *then* $\mathrm{rank}(\mathbf{X}) = F$ *and the decomposition of* $\mathbf{X}$ *is essentially unique.*

- Kruskal's condition is sharp, in the sense that there exist decompositions that are not unique as soon as F goes beyond the bound [Derksen, 2013].
- This *does not* mean that uniqueness is impossible beyond Kruskal's bound!
- Uniqueness well beyond Kruskal's bound, but *not always* – there exist counter-examples.

Generalization of Kruskal's theorem to tensors of any order

Theorem (Sidiropoulos & Bro, 2000)

Given $\mathbf{X} = [\![\mathbf{A}_1, \ldots, \mathbf{A}_N]\!]$, *with* $\mathbf{A}_n : \ I_n \times F$, *if* $\sum_{n=1}^{N} k_{\mathbf{A}_n} \geq 2F + N - 1$, *then the decomposition of* $\mathbf{X}$ *in terms of* $\{\mathbf{A}_n\}_{n=1}^{N}$ *is essentially unique.*

- This condition is sharp in the same sense as the $N = 3$ version is sharp [Derksen, 2013].

Is low-rank tensor decomposition always unique?

- One may wonder whether CPD is unique almost surely for any value of F strictly less than the generic rank?
- Cf. equations-versus-unknowns discussion.
- Proven for symmetric decompositions, with few exceptions: $(N, I; F) = (6, 3; 9), (4, 4; 9), (3, 6; 9)$ where there are two decompositions generically [Chiantini, 2016]
- For unsymmetric decompositions it has been verified for tensors up to 15000 entries that the only exceptions are $(I_1, \cdots, I_N; F) = (4, 4, 3; 5)$, $(4, 4, 4; 6)$, $(6, 6, 3; 8)$, $(p, p, 2, 2; 2p - 1)$ for $p \in \mathbb{N}$, $(2, 2, 2, 2, 2; 5)$, and the so-called unbalanced case $I_1 > \alpha$, $F \geq \alpha$, with $\alpha = \prod_{n=2}^{N} I_n - \sum_{n=2}^{N} (I_n - 1)$ [Chiantini, 2014].

Tucker and Multilinear SVD (MLSVD)

- Any $I \times J$ matrix $\mathbf{X}$ can be decomposed via SVD as $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U}^T\mathbf{U} = \mathbf{I} = \mathbf{U}\mathbf{U}^T$, $\mathbf{V}^T\mathbf{V} = \mathbf{I} = \mathbf{V}\mathbf{V}^T$, $\mathbf{\Sigma}(i,j) \geq 0$, $\mathbf{\Sigma}(i,j) > 0$ only when $j = i$ and $i \leq r_{\mathbf{X}}$, and $\mathbf{\Sigma}(i,i) \geq \mathbf{\Sigma}(i+1,i+1)$, $\forall i$.
- $\mathbf{U} := [\mathbf{u}_1, \cdots, \mathbf{u}_I]$, $\mathbf{V} := [\mathbf{v}_1, \cdots, \mathbf{v}_J]$, $\sigma_f := \mathbf{\Sigma}(f,f)$

$$\mathbf{X} = \mathbf{U}(:, 1:F)\mathbf{\Sigma}(1:F, 1:F)(\mathbf{V}(:, 1:F))^T = \sum_{f=1}^{F} \sigma_f \mathbf{u}_f \mathbf{v}_f^T$$

- Can we generalize the SVD to tensors, in a way that retains the many beautiful properties of matrix SVD?

Tucker and Multilinear SVD (MLSVD)

- Intuitively, introduce $K \times K$ matrix $\mathbf{W}$, $\mathbf{W}^T\mathbf{W} = \mathbf{I} = \mathbf{W}\mathbf{W}^T$, and a nonnegative $I \times J \times K$ *core tensor* $\mathbf{\Sigma}$ such that $\mathbf{\Sigma}(i, j, k) > 0$ only when $k = j = i$.

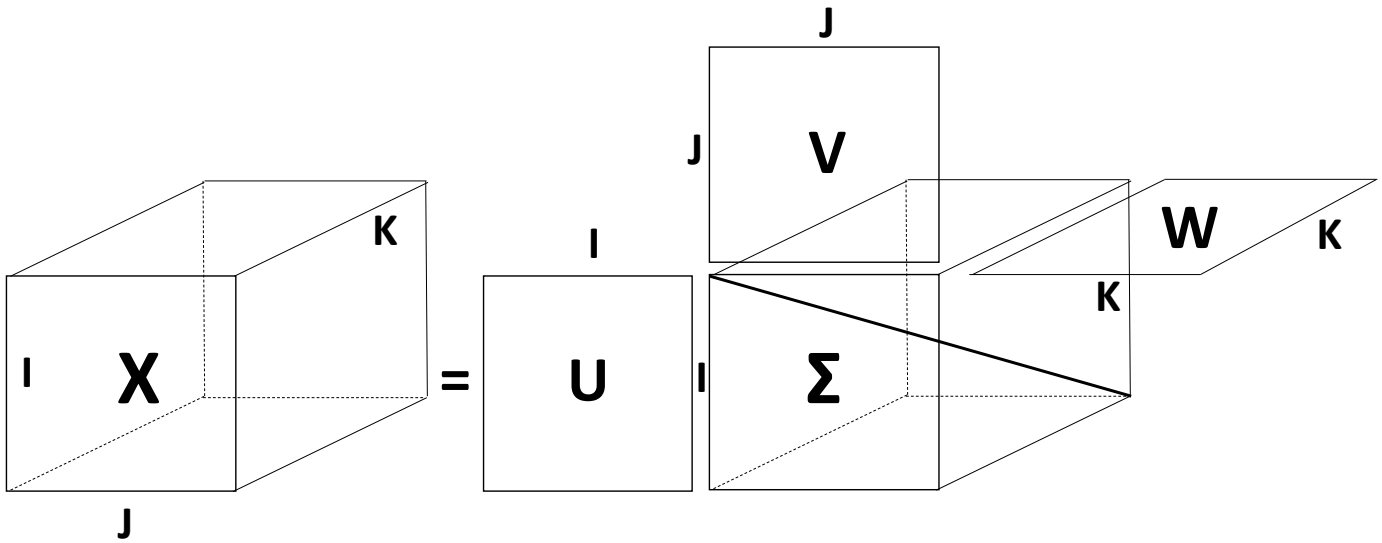


Figure: Diagonal tensor SVD?

Tucker and Multilinear SVD (MLSVD)

- Can we write an arbitrary tensor $\mathbf{X}$ in this way?
- Back-of-the-envelop calculation:
  - DoF in model $= I^2 + J^2 + K^2 + \min(I, J, K)$;
  - # equations = IJK
  - DoF $<$ # equations :-(
- Contrast: for matrices: $I^2 + J^2 + \min(I, J) > I^2 + J^2 > IJ$, always!
- More formal look: postulated model can be written out as

$$\sigma_1 \mathbf{u}_1 \circledcirc \mathbf{v}_1 \circledcirc \mathbf{w}_1 + \sigma_2 \mathbf{u}_2 \circledcirc \mathbf{v}_2 \circledcirc \mathbf{w}_2 + \cdots + \sigma_m \mathbf{u}_m \circledcirc \mathbf{v}_m \circledcirc \mathbf{w}_m,$$

  where $m := \min(I, J, K)$; so, a tensor of rank at most $\min(I, J, K)$, but we know that tensor rank can be (much) higher than that.
- Hence we certainly have to give up diagonality.

Tucker and Multilinear SVD (MLSVD)

- Consider instead a full (possibly dense, but ideally sparse) core tensor **G**

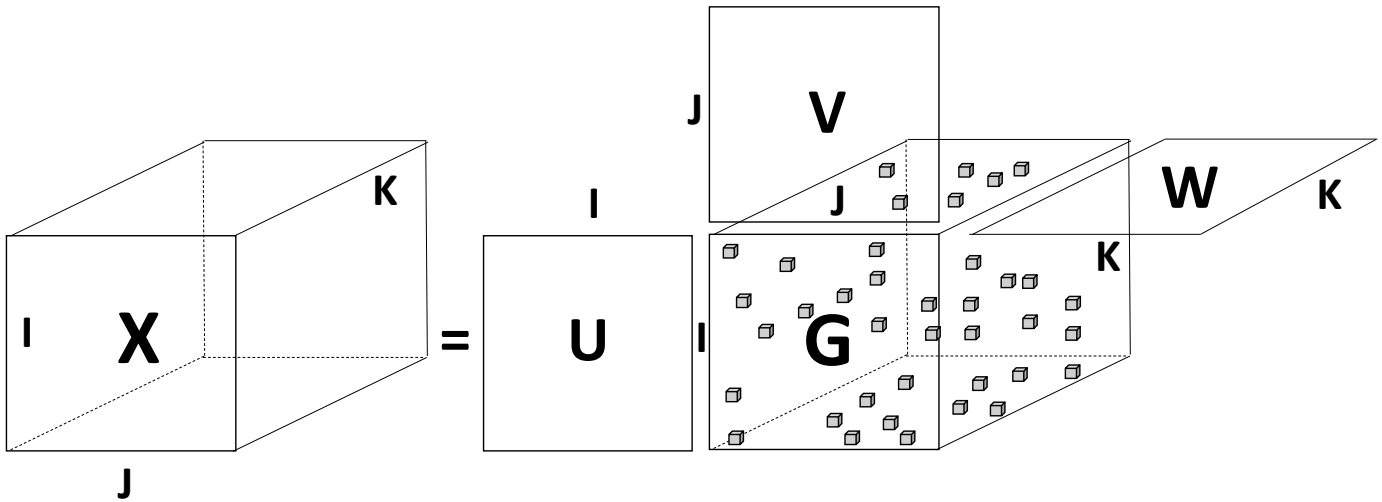


Figure: The Tucker model

Tucker and Multilinear SVD (MLSVD)

- Element-wise view

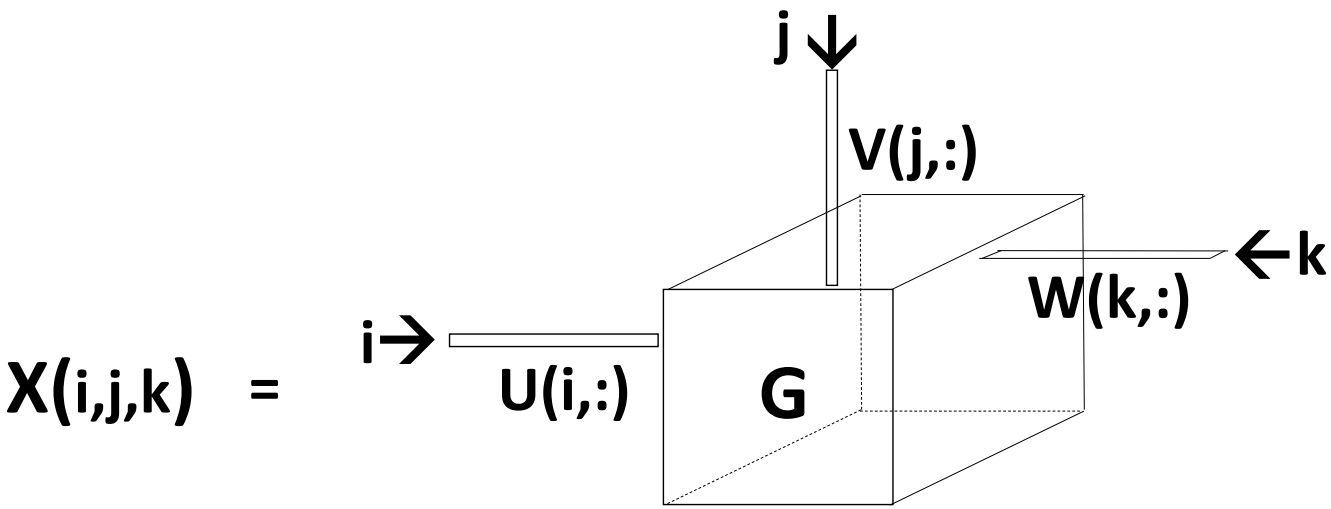


Figure: Element-wise view of the Tucker model

Tucker and Multilinear SVD (MLSVD)

- From the last figure $\rightarrow$

$$\mathbf{X}(i,j,k) = \sum_{\ell=1}^{I}\sum_{m=1}^{J}\sum_{n=1}^{K}\mathbf{G}(\ell,m,n)\mathbf{U}(i,\ell)\mathbf{V}(j,m)\mathbf{W}(k,n),$$

- or, equivalently,

$$\mathbf{X} = \sum_{\ell=1}^{I}\sum_{m=1}^{J}\sum_{n=1}^{K}\mathbf{G}(\ell,m,n)\mathbf{U}(:,\ell)\circledcirc\mathbf{V}(:,m)\circledcirc\mathbf{W}(:,n),$$

$$\text{or } \mathbf{X} = \sum_{\ell=1}^{I}\sum_{m=1}^{J}\sum_{n=1}^{K}\mathbf{G}(\ell,m,n)\mathbf{u}_\ell\circledcirc\mathbf{v}_m\circledcirc\mathbf{w}_n. \qquad (14)$$

- Here $\mathbf{u}_\ell := \mathbf{U}(:,\ell)$ and likewise for the $\mathbf{v}_m$, $\mathbf{w}_n$.

Tucker and Multilinear SVD (MLSVD)

$$\mathbf{X} = \sum_{\ell=1}^{I} \sum_{m=1}^{J} \sum_{n=1}^{K} \mathbf{G}(\ell, m, n) \mathbf{u}_\ell \circledcirc \mathbf{v}_m \circledcirc \mathbf{w}_n$$

- Note that each column of **U** interacts with every column of **V** and every column of **W** in this decomposition.
- The strength of this interaction is encoded in the corresponding element of **G**.
- Different from CPD, which only allows interactions between corresponding columns of $\mathbf{A}, \mathbf{B}, \mathbf{C}$, i.e., the only outer products that can appear in the CPD are of type $\mathbf{a}_f \circledcirc \mathbf{b}_f \circledcirc \mathbf{c}_f$.
- The *Tucker model* in (14) also allows "mixed" products of non-corresponding columns of **U**, **V**, **W**.

Tucker and Multilinear SVD (MLSVD)

- Note that *any* tensor $\mathbf{X}$ can be written in Tucker form (14), and a trivial way of doing so is to take $\mathbf{U} = \mathbf{I}_{I\times I}$, $\mathbf{V} = \mathbf{I}_{J\times J}$, $\mathbf{W} = \mathbf{I}_{K\times K}$, and $\mathbf{G} = \mathbf{X}$.
- Hence we may seek a possibly sparse $\mathbf{G}$, which could help reveal the underlying "essential" interactions between triples of columns of $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$.
- This is sometimes useful when one is interested in quasi-CPD models.
- The main interest in Tucker though is for finding subspaces and for tensor approximation purposes.

Tucker vs. CPD

- CPD appears to be a special case of the Tucker model, corresponding to $\mathbf{G}(\ell, m, n) = 0$ for all ℓ, m, n except possibly for $\ell = m = n$.
- However, when **U**, **V**, **W** are all square, such a restricted diagonal Tucker form can only model tensors up to rank $\min(I, J, K)$.
- If we allow "fat" (and therefore, clearly, non-orthogonal) **U**, **V**, **W** in Tucker though, it is possible to think of CPD as a special case of such a "blown-up" non-orthogonal Tucker model.

Tucker vs. CPD

- By similar token, if we allow column repetition in **A**, **B**, **C** for CPD, i.e.,
  - every column of **A** is repeated JK times, and we call the result **U**;
  - every column of **B** is repeated IK times, and we call the result **V**;
  - every column of **C** is repeated IJ times, and we call the result **W**,

  then it is possible to think of non-$\perp$ Tucker as a special case of CPD, *but*, repeated columns $\rightarrow$ k-ranks = 1 $\rightarrow$ very non-unique.
- In a nutshell,
  - both CPD and Tucker are sum-of-outer-products models;
  - one can argue that the most general form of one contains the other;
  - what distinguishes the two is uniqueness,
  - which is related but not tantamount to model parsimony ("minimality");
  - and modes of usage, which are quite different for the two models, as we will see.

MLSVD

- Tucker model in long vector form

$$\mathbf{x} := \text{vec}(\mathbf{X}) = (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})\,\mathbf{g},$$

  where $\mathbf{g} := \text{vec}(\mathbf{G})$; order of vectorization of $\mathbf{X}$ only affects order in which $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ appear in the Kronecker product chain, and the permutation of the elements of $\mathbf{g}$.

- From the properties of the Kronecker product, the expression above is the result of vectorization of matrix

$$\mathbf{X}_1 = (\mathbf{V} \otimes \mathbf{W})\,\mathbf{G}_1\mathbf{U}^T$$

  where the $KJ \times I$ matrix $\mathbf{X}_1$ contains all rows (mode-1 vectors) of tensor $\mathbf{X}$, and the $KJ \times I$ matrix $\mathbf{G}_1$ is a likewise reshaped form of the core tensor $\mathbf{G}$.

MLSVD

$$\mathbf{X}_1 = (\mathbf{V} \otimes \mathbf{W})\, \mathbf{G}_1 \mathbf{U}^T$$

- Notice we can linearly transform the columns of $\mathbf{U}$ and absorb the inverse transformation in $\mathbf{G}_1$, i.e.,

$$\mathbf{G}_1 \mathbf{U}^T = \mathbf{G}_1 \mathbf{M}^{-T} (\mathbf{U}\mathbf{M})^T,$$

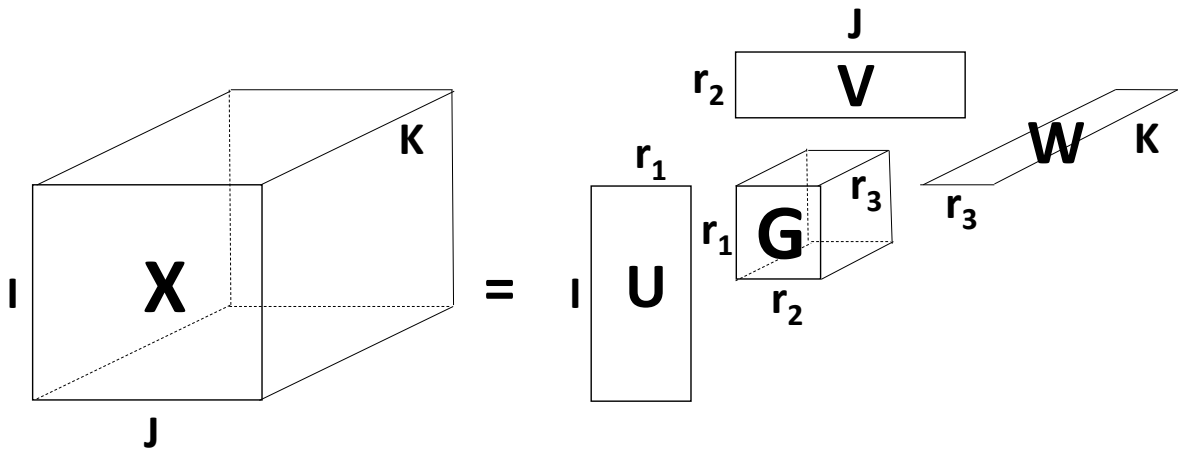
- Hence the Tucker model is not unique.
- $\mathbf{X}_1$ contains all rows of tensor $\mathbf{X}$; let r_1 denote the row-rank (mode-1 rank) of $\mathbf{X}$.
- WLOG can pick $\mathbf{U}$ to be an $I \times r_1$ orthonormal basis of the row-span of $\mathbf{X}$, and absorb the linear transformation in $\mathbf{G}$, which is thereby reduced from $I \times J \times K$ to $r_1 \times J \times K$.

MLSVD

- Doing the same for other two modes $\rightarrow$ WLOG the Tucker model can be written in compact form as

$$\mathbf{x} := \text{vec}(\mathbf{X}) = (\mathbf{U}_{r_1} \otimes \mathbf{V}_{r_2} \otimes \mathbf{W}_{r_3})\,\mathbf{g},$$

where $\mathbf{U}_{r_1}$ is $I \times r_1$, $\mathbf{V}_{r_2}$ is $J \times r_2$, $\mathbf{W}_{r_3}$ is $K \times r_3$, and $\mathbf{g} := \text{vec}(\mathbf{G})$ is $r_1 r_2 r_3 \times 1$ – the vectorization of the $r_1 \times r_2 \times r_3$ reduced-size core tensor $\mathbf{G}$.



MLSVD

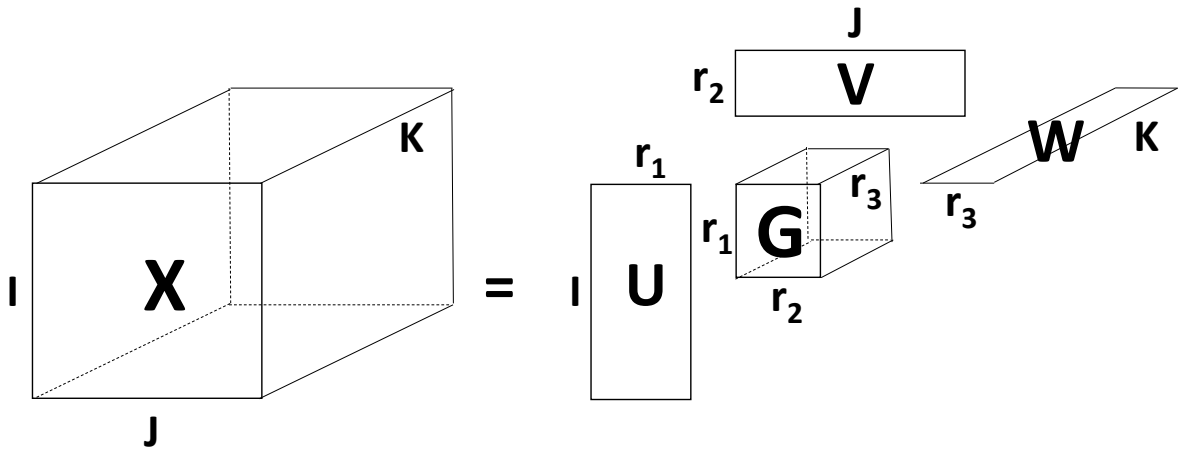


Figure: Compact (reduced) Tucker model: r_1, r_2, r_3 are the mode (row, column, fiber, resp.) ranks of $\mathbf{X}$.

- Drop subscripts from $\mathbf{U}_{r_1}$, $\mathbf{V}_{r_2}$, $\mathbf{W}_{r_3}$ for brevity.
- MLSVD (earlier name: HOSVD) = Tucker with orthonormal $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ chosen as the right singular vectors of the matrix unfoldings $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$, resp.

MLSVD

- Orthonormality of the columns of $\mathbf{U}_{r_1}$, $\mathbf{V}_{r_2}$, $\mathbf{W}_{r_3}$ implies orthonormality of the columns of their Kronecker product.
- This is because $(\mathbf{U}_{r_1} \otimes \mathbf{V}_{r_2})^T(\mathbf{U}_{r_1} \otimes \mathbf{V}_{r_2}) = (\mathbf{U}_{r_1}^T \otimes \mathbf{V}_{r_2}^T)(\mathbf{U}_{r_1} \otimes \mathbf{V}_{r_2}) = (\mathbf{U}_{r_1}^T\mathbf{U}_{r_1}) \otimes (\mathbf{V}_{r_2}^T\mathbf{V}_{r_2}) = \mathbf{I} \otimes \mathbf{I} = \mathbf{I}$.
- Recall that
  $\mathbf{x}_1 \perp \mathbf{x}_2 \Longleftrightarrow \mathbf{x}_1^T\mathbf{x}_2 = 0 \Longrightarrow ||\mathbf{x}_1 + \mathbf{x}_2||_2^2 = ||\mathbf{x}_1||_2^2 + ||\mathbf{x}_2||_2^2$.
- It follows that

$$||\mathbf{X}||_F^2 := \sum_{\forall\ i,j,k} |\mathbf{X}(i,j,k)|^2 = ||\mathbf{x}||_2^2 = ||\mathbf{g}||_2^2 = ||\mathbf{G}||_F^2,$$

where $\mathbf{x} = \text{vec}(\mathbf{X})$, and $\mathbf{g} = \text{vec}(\mathbf{G})$.

MLSVD

- If we drop certain outer products from the decomposition $\mathbf{x} = (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})\,\mathbf{g}$, or equivalently from (14), i.e., set the corresponding core elements to zero, then, by orthonormality

$$\left\|\mathbf{X} - \widehat{\mathbf{X}}\right\|_F^2 = \sum_{(\ell,m,n)\in\mathcal{D}} |\mathbf{G}(\ell, m, n)|^2,$$

  where $\mathcal{D}$ is the set of dropped core element indices.
- So, if we order the elements of $\mathbf{G}$ in order of decreasing magnitude, and discard the "tail", then $\widehat{\mathbf{X}}$ will be close to $\mathbf{X}$, and we can quantify the error without having to reconstruct $\mathbf{X}$, take the difference and evaluate the norm.

MLSVD

- With the matrix SVD in mind, feel tempted to drop entire columns of **U**, **V**, **W**.
- For matrix SVD, this corresponds to zeroing out small singular values on the diagonal of $\mathbf{\Sigma}$; per Eckart–Young $\rightarrow$ best low-rank approximation.
- Can we do the same for higher-order tensors?
- Can permute the slabs of **G** in any direction, and corresponding columns of **U**, **V**, **W** accordingly – cf. (14).
- Bring the frontal slab with the highest energy $||\mathbf{G}(:,:,n)||_F^2$ up front, then the one with second highest energy, etc.
- Likewise order the lateral slabs of the core *without changing the energy of the frontal slabs*; etc.
- $\rightarrow$ compact the energy of the core on its upper-left-front corner.

MLSVD

- We can then truncate the core, keeping only its upper-left-front dominant part of size $r_1' \times r_2' \times r_3'$, with $r_1' \leq r_1$, $r_2' \leq r_2$, and $r_3' \leq r_3$.
- The resulting approximation error can be readily bounded as

$$\left|\left|\mathbf{X} - \widehat{\mathbf{X}}\right|\right|_F^2 \leq \sum_{\ell=r_1'+1}^{r_1} ||\mathbf{G}(\ell,:,:)||_F^2 + \sum_{m=r_2'+1}^{r_2} ||\mathbf{G}(:,m,:)||_F^2 + \sum_{n=r_3'+1}^{r_3} ||\mathbf{G}(:,:,n)||_F^2 ,$$

  where we use $\leq$ as opposed to $=$ because dropped elements may be counted up to three times (in particular, the lower-right-back ones).
- One can of course compute the exact error of such a truncation strategy, but this involves instantiating $\mathbf{X} - \widehat{\mathbf{X}}$.

MLSVD

- Truncation in general *does not* yield the best approximation of $\mathbf{X}$ for the given (r_1', r_2', r_3').
- There is no exact equivalent of the Eckart–Young theorem for tensors of order higher than two [Kolda, 2013].
- The best low multilinear rank approximation problem for tensors is NP-hard.
- Despite this "bummer", much of the beauty of matrix SVD remains in MLSVD.

MLSVD

- Slabs of the core array $\mathbf{G}$ along each mode are orthogonal to each other, i.e., $(\text{vec}(\mathbf{G}(\ell,:,:)))^T \text{vec}(\mathbf{G}(\ell',:,:)) = 0$ for $\ell' \neq \ell$, and $||\mathbf{G}(\ell,:,:)||_F$ equals the ℓ-th singular value of $\mathbf{X}_1$.
- These properties generalize a property of matrix SVD, where "core matrix" of singular values $\mathbf{\Sigma}$ is diagonal $\rightarrow$ its rows are orthogonal to each other; same true for columns.
- Diagonality $\rightarrow$ orthogonality of one-lower-order slabs (sub-tensors of order one less than the original tensor).
- Converse is not true, e.g., consider

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

- Core diagonality not possible in general for higher-order tensors: DoF vs. # equations ...

MLSVD

- ... *but* all-orthogonality of one-lower-order slabs of the core array, and interpretation of their Frobenius norms as singular values of a certain matrix view of the tensor come WLOG + WLOPT (see next property).
- Intuitively pleasing result, first pointed out by [De Lathauwer, 2000]; motivates the analogy to matrix SVD.
- Simply truncating slabs (or elements) of the full core will not give the best low multilinear rank approximation of $\mathbf{X}$ in the case of three- and higher-order tensors; but
- Error $||\mathbf{X} - \widehat{\mathbf{X}}||_F^2$ is in fact at most 3 times higher than the minimal error (N times higher in the N-th order case) [Grasedyck, 2012, Hackbusch 2012].

MLSVD

- Error bound is actually the proper generalization of the Eckart–Young theorem.
- In the matrix case, because of diagonality there is only one summation and equality instead of inequality.
- Simply truncating the MLSVD at sufficiently high (r_1', r_2', r_3') is often enough to obtain a good approximation in practice – we may control the error as we wish, so long as we pick high enough (r_1', r_2', r_3').

MLSVD

- If we are interested in the best possible approximation of $\mathbf{X}$ with mode ranks (r_1', r_2', r_3'), however, then we need the following (dropping the $'$s for brevity):
- Let $\left(\widehat{\mathbf{U}}, \widehat{\mathbf{V}}, \widehat{\mathbf{W}}, \widehat{\mathbf{G}}_1\right)$ be a solution to

$$\begin{aligned} \min_{(\mathbf{U},\mathbf{V},\mathbf{W},\mathbf{G}_1)} \quad & ||\mathbf{X}_1 - (\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1\mathbf{U}^T||_F^2 \\ \text{such that:} \quad & \mathbf{U} : I \times r_1,\ r_1 \leq I,\ \mathbf{U}^T\mathbf{U} = \mathbf{I} \\ & \mathbf{V} : J \times r_2,\ r_2 \leq J,\ \mathbf{V}^T\mathbf{V} = \mathbf{I} \\ & \mathbf{W} : K \times r_3,\ r_3 \leq K,\ \mathbf{W}^T\mathbf{W} = \mathbf{I} \\ & \mathbf{G}_1 : r_3 r_2 \times r_1 \end{aligned}$$

- Claim: Then $\widehat{\mathbf{G}}_1 = (\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T\mathbf{X}_1\widehat{\mathbf{U}}$; and $\curvearrowleft$

MLSVD

- Substituting the conditionally optimal $\mathbf{G}_1$, problem recast in "concentrated" form

$$\begin{aligned} \max_{(\mathbf{U},\mathbf{V},\mathbf{W})} \quad & ||(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1 \mathbf{U}||_F^2 \\ \text{such that:} \quad & \mathbf{U} : I \times r_1,\ r_1 \leq I,\ \mathbf{U}^T\mathbf{U} = \mathbf{I} \\ & \mathbf{V} : J \times r_2,\ r_2 \leq J,\ \mathbf{V}^T\mathbf{V} = \mathbf{I} \\ & \mathbf{W} : K \times r_3,\ r_3 \leq K,\ \mathbf{W}^T\mathbf{W} = \mathbf{I} \end{aligned}$$

- $\widehat{\mathbf{U}}$ = dominant r_1-dim. right subspace of $(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_1$;
- $\widehat{\mathbf{V}}$ = dominant r_2-dim. right subspace of $(\widehat{\mathbf{U}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_2$;
- $\widehat{\mathbf{W}}$ = dominant r_3-dim. right subspace of $(\widehat{\mathbf{U}} \otimes \widehat{\mathbf{V}})^T \mathbf{X}_3$;
- $\widehat{\mathbf{G}}_1$ has orthogonal columns; and
- $\left\{ ||\widehat{\mathbf{G}}_1(:,m)||_2^2 \right\}_{m=1}^{r_1} = r_1$ principal sing. vals of $(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_1$.

MLSVD: Proof

- (Note: Each column of $\widehat{\mathbf{G}}_1$ is a vectorized slab of the core array $\widehat{\mathbf{G}}$)
- $||\text{vec}(\mathbf{X}_1) - (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})\,\text{vec}(\mathbf{G}_1)||_2^2 = ||\mathbf{X}_1 - (\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1\mathbf{U}^T||_F^2$.
- Conditioned on (orthonormal) $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ the optimal $\mathbf{G}$ is given by $\text{vec}(\widehat{\mathbf{G}}_1) = (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})^T\,\text{vec}(\mathbf{X}_1)$.
- Therefore $\widehat{\mathbf{G}}_1 = (\mathbf{V} \otimes \mathbf{W})^T\mathbf{X}_1\mathbf{U}$.
- Consider $||\mathbf{X}_1 - (\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1\mathbf{U}^T||_F^2$, and define $\widetilde{\mathbf{X}}_1 := (\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1\mathbf{U}^T$;

MLSVD: Proof

- Use that
$||\mathbf{X}_1 - \widetilde{\mathbf{X}}_1||_F^2 = \text{Tr}((\mathbf{X}_1 - \widetilde{\mathbf{X}}_1)^T(\mathbf{X}_1 - \widetilde{\mathbf{X}}_1)) = ||\mathbf{X}_1||_F^2 + ||\widetilde{\mathbf{X}}_1||_F^2 - 2\text{Tr}(\mathbf{X}_1^T\widetilde{\mathbf{X}}_1)$.
- Orthonormality of $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W} \Rightarrow ||\widetilde{\mathbf{X}}_1||_F^2 = ||\mathbf{G}_1||_F^2$.
- Consider

$$-2\text{Tr}(\mathbf{X}_1^T\widetilde{\mathbf{X}}_1) = -2\text{Tr}(\mathbf{X}_1^T(\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1\mathbf{U}^T),$$

- Substitute $\mathbf{G}_1 = (\mathbf{V} \otimes \mathbf{W})^T\mathbf{X}_1\mathbf{U}$ to obtain

$$-2\text{Tr}(\mathbf{X}_1^T(\mathbf{V} \otimes \mathbf{W})(\mathbf{V} \otimes \mathbf{W})^T\mathbf{X}_1\mathbf{U}\mathbf{U}^T).$$

- Using property of trace to bring rightmost to the left,

$$-2\text{Tr}(\mathbf{U}^T\mathbf{X}_1^T(\mathbf{V} \otimes \mathbf{W})(\mathbf{V} \otimes \mathbf{W})^T\mathbf{X}_1\mathbf{U}) = -2\text{Tr}(\mathbf{G}_1^T\mathbf{G}_1) = -2||\mathbf{G}_1||_F^2.$$

MLSVD: Proof

- It follows $||\mathbf{X}_1 - (\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1\mathbf{U}^T||_F^2 = ||\mathbf{X}_1||_F^2 - ||\mathbf{G}_1||_F^2 \iff$
- maximize $||\mathbf{G}_1||_F^2 = ||(\mathbf{V} \otimes \mathbf{W})^T\mathbf{X}_1\mathbf{U}||_F^2$.
- So $\widehat{\mathbf{U}}$ is the dominant right subspace of $(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T\mathbf{X}_1$;
- take it to be the r_1 principal right singular vectors of $(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T\mathbf{X}_1$.
- Corresponding results for $\widehat{\mathbf{V}}$ and $\widehat{\mathbf{W}}$ by role symmetry.

MLSVD: Proof

- To show that $\widehat{\mathbf{G}}_1$ has orthogonal columns, let $\widehat{\mathbf{G}}_1 = [\widehat{\mathbf{g}}_{1,1}, \cdots, \widehat{\mathbf{g}}_{1,r_1}]$, $\widehat{\mathbf{U}} = [\widehat{\mathbf{u}}_1, \cdots, \widehat{\mathbf{u}}_{r_1}]$, and consider

$$\widehat{\mathbf{g}}_{1,m_1}^T \widehat{\mathbf{g}}_{1,m_2} = \widehat{\mathbf{u}}_{m_1}^T \mathbf{X}_1^T (\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_1 \widehat{\mathbf{u}}_{m_2}.$$

- Let $\mathbf{\Gamma \Sigma \tilde{U}}^T$ be the SVD of $(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_1$.
- Then $\tilde{\mathbf{U}} = [\widehat{\mathbf{U}}, \ \check{\mathbf{U}}]$, so

$$(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_1 \mathbf{u}_{m_2} = \gamma_{m_2} \sigma_{m_2}$$

MLSVD: Proof

- It follows, by virtue of orthonormality of left singular vectors of $(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_1$ (here $\delta(\cdot)$ is the Kronecker delta)

$$\widehat{\mathbf{g}}_{1,m_1}^T \widehat{\mathbf{g}}_{1,m_2} = \sigma_{m_1} \sigma_{m_2} \boldsymbol{\gamma}_{m_1}^T \boldsymbol{\gamma}_{m_2} = \sigma_{m_1} \sigma_{m_2} \delta(m_1 - m_2),$$

- By role symmetry, it follows that the slabs of $\widehat{\mathbf{G}}$ along any mode are likewise orthogonal.
- As byproduct of last equation, $||\widehat{\mathbf{G}}(:,:,m)||_F^2 = ||\widehat{\mathbf{G}}_1(:,m)||_2^2 = ||\widehat{\mathbf{g}}_{1,m}||_2^2 = \sigma_m^2$;
- that is, *the Frobenius norms of the lateral core slabs are the r_1 principal singular values of* $(\widehat{\mathbf{V}} \otimes \widehat{\mathbf{W}})^T \mathbf{X}_1$.

Best rank-1 tensor approximation: NP-hard :-(

- The best rank-1 tensor approximation problem over $\mathbb{R}$ is NP-hard [Hillar & Lim, 2013].
- So the best low multilinear rank approximation problem is also NP-hard (the best multilinear rank approximation with $(r_1, r_2, r_3) = (1, 1, 1)$ is the best rank-1 approximation).
- This is reflected in key limitation of MLSVD characterization:
  - gives explicit expressions that relate the sought **U**, **V**, **W**, and **G**, ...
  - but does not provide an explicit solution for any of them!
- On the other hand, characterization naturally suggests alternating least squares scheme $\curvearrowleft$

⊥-Tucker ALS

1. Initialize:
   - $\mathbf{U} = r_1$ principal right singular vectors of $\mathbf{X}_1$;
   - $\mathbf{V} = r_2$ principal right singular vectors of $\mathbf{X}_2$;
   - $\mathbf{W} = r_3$ principal right singular vectors of $\mathbf{X}_3$;
2. repeat:
   - $\mathbf{U} = r_1$ principal right sing. vec. of $(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1$;
   - $\mathbf{V} = r_2$ principal right sing. vec. of $(\mathbf{U} \otimes \mathbf{W})^T \mathbf{X}_2$;
   - $\mathbf{W} = r_3$ principal right sing. vec. of $(\mathbf{U} \otimes \mathbf{V})^T \mathbf{X}_3$;
   - until negligible change in $||(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1 \mathbf{U}||_F^2$.
3. $\mathbf{G}_1 = (\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1 \mathbf{U}$.

$\perp$-Tucker ALS

- First variant of Tucker-ALS goes back to [Kroonenberg & De Leeuw].
- Initialization in step 1) [together with step 3)] is (truncated) MLSVD. Not optimal, but helps in most cases.
- Each variable update is conditionally optimal $\rightarrow$ reward $||(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1 \mathbf{U}||_F^2$ is non-decreasing ($\Leftrightarrow$ cost $||\mathbf{X}_1 - (\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1 \mathbf{U}^T||_F^2$ is non-increasing)
- Also bounded from above (resp. below), thus convergence of the reward (cost) sequence is guaranteed.
- Conceptual similarity of the above algorithm with ALS for CPD.
- Using MLSVD with somewhat higher (r_1, r_2, r_3) can be computationally preferable to ALS.

Tucker for large-scale problems

- In the case of big data, even the computation of MLSVD may be prohibitive.
- Randomized projection approaches become more appealing.
- Draw the columns of **U**, **V**, **W** from the columns, rows, fibers of **X**: [Oseledets, 2008; Mahoney, 2008]. Drawback: no identifiability.
- "Completely" random projections: [Sidiropoulos *et al.*, 2012, 2014]: Identifiability!
- Krylov subspace methods offer an alternative for large-scale problems; see [Savas, 2013] for Tucker-type extensions.

Compression as preprocessing: CANDELINC

- Consider a tensor $\mathbf{X}$ in vectorized form, and corresponding CPD and orthogonal Tucker ($\perp$-Tucker) models

$$\mathbf{x} = (\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\mathbf{1} = (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})\mathbf{g}.$$

- Pre-multiplying with $(\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})^T = (\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T)$ and using the mixed-product rule for $\otimes$, $\odot$, we obtain

$$\mathbf{g} = \Big((\mathbf{U}^T\mathbf{A}) \odot (\mathbf{V}^T\mathbf{B}) \odot (\mathbf{W}^T\mathbf{C})\Big)\,\mathbf{1},$$

  i.e., the Tucker core array $\mathbf{G}$ (shown above in vectorized form $\mathbf{g}$) admits a CPD decomposition of rank($\mathbf{G}$) $\leq$ rank($\mathbf{X}$).

- Let $\big[\!\big[\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}}\big]\!\big]$ be a CPD of $\mathbf{G}$, i.e., $\mathbf{g} = (\tilde{\mathbf{A}} \odot \tilde{\mathbf{B}} \odot \tilde{\mathbf{C}})\mathbf{1}$. Then

$$\mathbf{x} = (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})\mathbf{g} = (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})(\tilde{\mathbf{A}} \odot \tilde{\mathbf{B}} \odot \tilde{\mathbf{C}})\mathbf{1} =$$

$$= \Big((\mathbf{U}\tilde{\mathbf{A}}) \odot (\mathbf{V}\tilde{\mathbf{B}}) \odot (\mathbf{W}\tilde{\mathbf{C}})\Big)\,\mathbf{1},$$

  by the mixed product rule.

Compression as preprocessing: CANDELINC

- Assuming that the CPD of $\mathbf{X}$ is essentially unique, it then follows that

$$\mathbf{A} = \mathbf{U}\tilde{\mathbf{A}}\mathbf{\Pi}\mathbf{\Lambda}_a, \ \mathbf{B} = \mathbf{V}\tilde{\mathbf{B}}\mathbf{\Pi}\mathbf{\Lambda}_b, \ \mathbf{C} = \mathbf{W}\tilde{\mathbf{C}}\mathbf{\Pi}\mathbf{\Lambda}_c,$$

  where $\mathbf{\Pi}$ is a permutation matrix and $\mathbf{\Lambda}_a\mathbf{\Lambda}_b\mathbf{\Lambda}_c = \mathbf{I}$.
- It follows that

$$\mathbf{U}^T\mathbf{A} = \tilde{\mathbf{A}}\mathbf{\Pi}\mathbf{\Lambda}_a, \ \mathbf{V}^T\mathbf{B} = \tilde{\mathbf{B}}\mathbf{\Pi}\mathbf{\Lambda}_b, \ \mathbf{W}^T\mathbf{C} = \tilde{\mathbf{C}}\mathbf{\Pi}\mathbf{\Lambda}_c,$$

  so that the CPD of $\mathbf{G}$ is essentially unique, and therefore $\text{rank}(\mathbf{G}) = \text{rank}(\mathbf{X})$.
- This suggests that an attractive way to compute the CPD of $\mathbf{X}$ is to first compress, compute the CPD of $\mathbf{G}$, and then "blow-up" the resulting factors, since $\mathbf{A} = \mathbf{U}\tilde{\mathbf{A}}$ (up to column permutation and scaling).
- Also shows that $\mathbf{A} = \mathbf{U}\mathbf{U}^T\mathbf{A}$, and likewise for the other two modes.

Compression as preprocessing: CANDELINC

- Caveat: discussion ↑ assumes *exact* CPD and ⊥-Tucker models;
- but also works for low-rank least-squares approximation, see *Candelinc* theorem of [Carroll *et al.*, 1980]; and [Bro & Andersson, 1998].
- Does not work for a constrained CPD (e.g. one or more factor matrices nonnegative, monotonic, sparse, . . .)
- In ALS can still exploit multi-linearity, to update $\mathbf{U}$ by solving a *constrained* and/or regularized linear least squares problem.
- For $\mathbf{G}$, we can use the vectorization property of the Kronecker product to bring it to the right, and then use a constrained or regularized linear least squares solver.
- By the mixed product rule, this last step entails pseudo-inversion of the $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ matrices, instead of their (much larger) Kronecker product.

Compression as preprocessing: Oblique Tucker

- This type of model is sometimes called *oblique* Tucker, to distinguish from *orthogonal* Tucker.
- More generally, one can fit the constrained CPD in the uncompressed space, but with $\mathbf{X}$ replaced by its parameter-efficient factorized representation.
- The structure of the latter may then be exploited to reduce the per iteration complexity; see [De Lathauwer, Tensorlab3, Asilomar 2016].

Other tensor models: Tensor Train (TT) and hierarchical Tucker (hTucker)

- $\perp$-Tucker/MLSVD approximation / compression limited to tensors of moderate order.
- Consider situation at order N and assume for simplicity that $r_1 = r_2 = \ldots = r_N = r > 1$.
- Then the core tensor $\mathbf{G}$ has r^N entries.
- This exponential dependence of the number of entries on the tensor order N is called the *Curse of Dimensionality*
- In such cases one may resort to a Tensor Train (TT) representation or a hierarchical Tucker (hTucker) decomposition instead [Oseledets, 2011], [Grasedyck, 2013].
- Caveat: approximation / compression only; no identifiability in general.

Tensor Train (TT) and hierarchical Tucker (hTucker)

- A TT of an N-th order tensor $\mathbf{X}$ is of the form

$$\mathbf{X}(i_1, i_2, \ldots, i_N) = \sum_{r_1 r_2 \ldots r_{N-1}} u^{(1)}_{i_1 r_1} u^{(2)}_{r_1 i_2 r_2} u^{(3)}_{r_2 i_3 r_3} \ldots u^{(N)}_{i_N r_{N-1}}, \qquad (15)$$

  in which one can see $\mathbf{U}^{(1)}$ as the locomotive and the next factors as the carriages.
- Each carriage "transports" one tensor dimension, and two consecutive carriages are connected through the summation over one common index.

Tensor Train (TT) and hierarchical Tucker (hTucker)

- Since every index appears at most twice and since there are no index cycles, the TT-format is “matrix-like”, i.e. a TT approximation can be computed using established techniques from numerical *linear* algebra, similarly to MLSVD.
- Number of entries is now $O(NIr^2)$, so the Curse of Dimensionality has been broken.
- hTucker is the extension in which the indices are organized in a binary tree.

PARALIND, Block Term Decomposition

- Possible to have unique decomposition in terms that are not even rank-1.
- Block Term Decompositions (BTD) [De Lathauwer, 2008, 2011] write a tensor as a sum of terms that have low multilinear rank.
- As in CPD, uniqueness of a BTD is up to a permutation of the terms. The scaling/counterscaling ambiguities within a rank-1 term generalize to the indeterminacies in a Tucker representation.
- Expanding the block terms into sums of rank-1 terms with repeated vectors yields a form that is known as PARALIND [Bro, Harshman, Sidiropoulos, Lundy, 2009].

Data Fusion: Coupled Decompositions

- Multiple data sets may be jointly analyzed by means of coupled decompositions of several matrices and/or tensors, possibly of different size.
- Harshman's PARAFAC2 is early variant, in which coupling was imposed through a shared covariance matrix.
- In coupled setting, particular decompositions may inherit uniqueness from other decompositions; in particular, the decomposition of a data matrix may become unique thanks to coupling [Sorensen & De Lathauwer, 2015].

Reprise – what's coming up next

Algorithms:

- Alternating Least Squares (ALS)
- Gradient Descent
- Quasi-Newton & Non-linear Least Squares
- Line Search
- Handling Missing Values
- Stochastic Gradient Descent (SGD)
- Imposing Constraints

Introduction to ALS

- ALS is the “workhorse” algorithm for tensor decompositions
- Special case of Block Coordinate Descent (BCD)
- Flexible and easy to derive
- No parameters to tune
- General ALS iteration:
  1. Fix all factors except for one
  2. Solve the linear LS estimation problem for that factor
  3. Cycle over all factors
- Today we will see ALS for CPD and Tucker

ALS for CPD

- Suppose we fix $\mathbf{A}, \mathbf{B}$, then update formula for $\mathbf{C}$ is:

$$\mathbf{C} \leftarrow \arg\min_{\mathbf{C}} ||\mathbf{X}_3 - (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T||_F^2,$$

- This is a simple linear Least Squares problem
- Solution :

$$\mathbf{C}^T \leftarrow (\mathbf{B} \odot \mathbf{A})^\dagger \mathbf{X}_3.$$

ALS for CPD (contd.)

- The pseudo-inverse

$$(\mathbf{B} \odot \mathbf{A})^{\dagger} = \Big[(\mathbf{B} \odot \mathbf{A})^{T}(\mathbf{B} \odot \mathbf{A})\Big]^{-1}(\mathbf{B} \odot \mathbf{A})^{T},$$

  can be simplified.
- We can show that

$$(\mathbf{B} \odot \mathbf{A})^{T}(\mathbf{B} \odot \mathbf{A}) = (\mathbf{B}^{T}\mathbf{B}) * (\mathbf{A}^{T}\mathbf{A}),$$

  This only involves the Hadamard product of $F \times F$ matrices,
  - Easy to invert for small ranks F
  - *But* note that in case of big sparse data, small F may not be enough
- Thus the update of $\mathbf{C}$ can be performed as

$$\mathbf{C}^{T} \leftarrow \Big((\mathbf{B}^{T}\mathbf{B}) * (\mathbf{A}^{T}\mathbf{A})\Big)^{-1}(\mathbf{B} \odot \mathbf{A})^{T}\mathbf{X}_{3}.$$

ALS for CPD (contd.)

- When F is small

$$(\mathbf{B} \odot \mathbf{A})^T \mathbf{X}_3$$

  is the bottleneck
- Notice that $\mathbf{B} \odot \mathbf{A}$ is $IJ \times F$, and $\mathbf{X}_3$ is $IJ \times K$.
- Brute-force computation of

$$(\mathbf{B} \odot \mathbf{A})^T \mathbf{X}_3$$

  needs IJF additional memory and flops to instantiate $\mathbf{B} \odot \mathbf{A}$, even though the result is only $F \times K$, and $IJKF$ flops to actually compute the product – but see [BK07, VMV15, PTC13].

ALS for CPD (contd.)

- When $\mathbf{X}$ is sparse,
  - NNZ($\mathbf{X}$) nonzero elements stored in a [i,j,k,value] list
  - Every nonzero element multiplies a column of $(\mathbf{B} \odot \mathbf{A})^T$
  - Result should be added to column k
  - The specific column can be generated on-the-fly with $F + 1$ flops, for an overall complexity of $(2F + 1)$NNZ($\mathbf{X}$), without requiring any additional memory (other than that needed to store running estimates of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and the data $\mathbf{X}$).
- When $\mathbf{X}$ is dense, the number of flops is inevitably of order $IJKF$, but still no additional memory is needed this way.

ALS for CPD (contd.)

- Computation can be parallelized in several ways
- See [BK07, KPHF12, RSSK14, CV14, SRSK15] for various resource-efficient algorithms for *matricized tensor times Khatri–Rao product* (MTTKRP) computations.

ALS for Tucker

As we saw previously, for $\perp$**-Tucker ALS** we have:

1. Initialize:
   - $\mathbf{U} = r_1$ principal right singular vectors of $\mathbf{X}_1$;
   - $\mathbf{V} = r_2$ principal right singular vectors of $\mathbf{X}_2$;
   - $\mathbf{W} = r_3$ principal right singular vectors of $\mathbf{X}_3$;
2. repeat:
   - $\mathbf{U} = r_1$ principal right sing. vec. of $(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1$;
   - $\mathbf{V} = r_2$ principal right sing. vec. of $(\mathbf{U} \otimes \mathbf{W})^T \mathbf{X}_2$;
   - $\mathbf{W} = r_3$ principal right sing. vec. of $(\mathbf{U} \otimes \mathbf{V})^T \mathbf{X}_3$;
   - until negligible change in $||(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1 \mathbf{U}||_F^2$.
3. $\mathbf{G}_1 = (\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1 \mathbf{U}$.

- Steps 1 and 3 correspond to truncated MLSVD
  - Not necessarily optimal, but very good initialization in most cases

ALS for Tucker (contd.)

- ALS for $\perp$-Tucker ALS also called Higher Order Orthogonal Iteration (HOOI)
- Each variable update is optimal conditioned on the rest of the variables,
  - Reward $||(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1 \mathbf{U}||_F^2$ is non-decreasing and bounded from above
  - Convergence of the reward (cost) sequence is guaranteed
- First variant of Tucker-ALS goes back to [Kro08].

ALS for Tucker (contd.)

For $\perp$-Tucker ALS,

- Need to compute products of type $(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1$
- Then compute the principal right singular vectors of resulting $r_2 r_3 \times I$ matrix
- Column-generation idea can be used here as well to avoid intermediate memory explosion and exploit sparsity in $\mathbf{X}$ when computing $(\mathbf{V} \otimes \mathbf{W})^T \mathbf{X}_1$.

ALS for Tucker (contd.)

For oblique Tucker ALS need to compute

- $((\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1)^\dagger \mathbf{X}_1$ for updating $\mathbf{U}$
- $\left(\mathbf{U}^\dagger \otimes \mathbf{V}^\dagger \otimes \mathbf{W}^\dagger\right) \mathbf{x}$ for updating $\mathbf{g} \leftrightarrow \mathbf{G}$
  - Requires pseudo-inverses of relatively small matrices,
  - But note that
$$((\mathbf{V} \otimes \mathbf{W})\mathbf{G}_1)^\dagger \neq \mathbf{G}_1^\dagger (\mathbf{V} \otimes \mathbf{W})^\dagger ,$$
  - Equality holds if $\mathbf{V} \otimes \mathbf{W}$ is full column rank *and* $\mathbf{G}_1$ is full row rank, which requires $r_2 r_3 \leq r_1$.

Gradient Descent

Consider the squared loss

$$\mathcal{L}(\mathbf{A},\mathbf{B},\mathbf{C}) := ||\mathbf{X}_1 - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T||_F^2 =$$

$$\text{tr}\left((\mathbf{X}_1 - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T)^T(\mathbf{X}_1 - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T)\right) = ||\mathbf{X}_1||_F^2 -$$

$$2\,\text{tr}\left(\mathbf{X}_1^T(\mathbf{C} \odot \mathbf{B})\mathbf{A}^T\right) + \text{tr}\left(\mathbf{A}(\mathbf{C} \odot \mathbf{B})^T(\mathbf{C} \odot \mathbf{B})\mathbf{A}^T\right).$$

- Recall that $(\mathbf{C} \odot \mathbf{B})^T(\mathbf{C} \odot \mathbf{B}) = (\mathbf{C}^T\mathbf{C}) * (\mathbf{B}^T\mathbf{B})$
- We may equivalently take the gradient of $-2\,\text{tr}\left(\mathbf{X}_1^T(\mathbf{C} \odot \mathbf{B})\mathbf{A}^T\right) + \text{tr}\left(\mathbf{A}(\mathbf{C}^T\mathbf{C}) * (\mathbf{B}^T\mathbf{B})\mathbf{A}^T\right)$.

Gradient Descent (contd.)

Arranging the gradient in the same format as $\mathbf{A}$, we have

$$\begin{aligned}\frac{\partial \mathcal{L}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial \mathbf{A}} &= -2\mathbf{X}_1^T(\mathbf{C}\odot\mathbf{B}) + 2\,\mathbf{A}\left[(\mathbf{C}^T\mathbf{C})*(\mathbf{B}^T\mathbf{B})\right] \\ &= -2\left(\mathbf{X}_1^T - \mathbf{A}(\mathbf{C}\odot\mathbf{B})^T\right)(\mathbf{C}\odot\mathbf{B}),\end{aligned}$$

Appealing to role symmetry, we likewise obtain

$$\begin{aligned}\frac{\partial \mathcal{L}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial \mathbf{B}} &= -2\mathbf{X}_2^T(\mathbf{C}\odot\mathbf{A}) + 2\,\mathbf{B}\left[(\mathbf{C}^T\mathbf{C})*(\mathbf{A}^T\mathbf{A})\right] \\ &= -2\left(\mathbf{X}_2^T - \mathbf{B}(\mathbf{C}\odot\mathbf{A})^T\right)(\mathbf{C}\odot\mathbf{A}),\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{L}(\mathbf{A},\mathbf{B},\mathbf{C})}{\partial \mathbf{C}} &= -2\mathbf{X}_3^T(\mathbf{B}\odot\mathbf{A}) + 2\,\mathbf{C}\left[(\mathbf{B}^T\mathbf{B})*(\mathbf{A}^T\mathbf{A})\right] \\ &= -2\left(\mathbf{X}_3^T - \mathbf{C}(\mathbf{B}\odot\mathbf{A})^T\right)(\mathbf{B}\odot\mathbf{A}).\end{aligned}$$

- With these gradient expressions at hand, we can employ any gradient-based algorithm for model fitting.

Gradient Descent (contd.)

Remark

The conditional least squares update for $\mathbf{A}$ *is*

$$\mathbf{A} \leftarrow \left[(\mathbf{C}^T\mathbf{C}) * (\mathbf{B}^T\mathbf{B})\right]^{-1} (\mathbf{C} \odot \mathbf{B})^T\mathbf{X}_1,$$

This means that:

- *Taking a gradient step or solving the LS sub-problem to (conditional) optimality involves computing the same quantities:* $(\mathbf{C}^T\mathbf{C}) * (\mathbf{B}^T\mathbf{B})$ *and* $(\mathbf{C} \odot \mathbf{B})^T\mathbf{X}_1$*.*
- *The only difference is that to take a gradient step you don't need to invert the* $F \times F$ *matrix* $(\mathbf{C}^T\mathbf{C}) * (\mathbf{B}^T\mathbf{B})$*.*
- *For small* F*, each gradient step is essentially as expensive as an ALS step.*

Quasi Newton & Non-linear Least Squares

- The well-known Newton descent algorithm uses a local quadratic approximation of the cost function $\mathcal{L}(\mathbf{A}, \mathbf{B}, \mathbf{C})$ to obtain a new step as the solution of the set of linear equations

$$\mathbf{H}\mathbf{p} = -\mathbf{g},$$

  in which $\mathbf{g}$ and $\mathbf{H}$ are the gradient and Hessian of $\mathcal{L}$, respectively.
- Computation of Hessian may be prohibitively expensive,
  - Resort to an approximation
  - Newton and Nonlinear Least Squares (NLS)

Quasi Newton & Non-linear Least Squares (contd.)

- Quasi-Newton methods such as Nonlinear Conjugate Gradients (NCG) and (limited memory) BFGS use a diagonal plus low-rank matrix approximation of the Hessian.
- In combination with line search or trust region globalization strategies for step size selection, quasi-Newton does guarantee convergence to a stationary point
  - Contrary to plain ALS, and its convergence is superlinear [SVD13, NW06]

Quasi Newton & Non-linear Least Squares (contd.)

- NLS methods such as Gauss–Newton and Levenberg–Marquardt start from a local linear approximation of the residual $\mathbf{X}_1 - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T$ to approximate the Hessian as $\mathcal{D}_{\boldsymbol{\theta}}\boldsymbol{\varphi}(\boldsymbol{\theta})^T \mathcal{D}_{\boldsymbol{\theta}}\boldsymbol{\varphi}(\boldsymbol{\theta})$, with $\mathcal{D}_{\boldsymbol{\theta}}\boldsymbol{\varphi}(\boldsymbol{\theta})$ the Jacobian matrix of $\boldsymbol{\varphi}(\boldsymbol{\theta})$ (where $\boldsymbol{\theta}$ is the parameter vector)
- The algebraic structure of $\mathcal{D}_{\boldsymbol{\theta}}\boldsymbol{\varphi}(\boldsymbol{\theta})^T \mathcal{D}_{\boldsymbol{\theta}}\boldsymbol{\varphi}(\boldsymbol{\theta})$ can be exploited to obtain a fast inexact NLS algorithm that has several favorable properties [TB06, SVD13].

Quasi Newton & Non-linear Least Squares (contd.)

- **Robustness**: NLS has been observed to be more robust for difficult decompositions than plain ALS [SVD13, TB06].
- **Parallelizability**: $\mathcal{D}_{\boldsymbol{\theta}}\boldsymbol{\varphi}(\boldsymbol{\theta})^T\mathcal{D}_{\boldsymbol{\theta}}\boldsymbol{\varphi}(\boldsymbol{\theta})$ can easily be split into smaller matrix-vector products (N^2 in the N-th order case)
  - Makes inexact NLS overall well-suited for parallel implementation.
  - Variants for low multilinear rank approximation are discussed in [IAVHDL11, SL10] and references therein.

Line Search

- An important issue in numerical optimization is the choice of step-size
- We cam exploit the multi-linearity of the cost function [RCH08]
- Suppose we have determined an update ("search") direction, say the negative gradient one.
- We seek to select the optimal step-size μ for the update

$$\begin{bmatrix} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{bmatrix} \leftarrow \begin{bmatrix} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{bmatrix} + \mu \begin{bmatrix} \boldsymbol{\Delta}_A \\ \boldsymbol{\Delta}_B \\ \boldsymbol{\Delta}_C \end{bmatrix},$$

and the goal is to

$$\min_{\mu} \left\| \mathbf{X}_1 - \left((\mathbf{C} + \mu \boldsymbol{\Delta}_C) \odot (\mathbf{B} + \mu \boldsymbol{\Delta}_B) \right) (\mathbf{A} + \mu \boldsymbol{\Delta}_A)^T \right\|_F^2 .$$

Line Search (contd.)

$$\min_{\mu} \left|\left| \mathbf{X}_1 - \left((\mathbf{C} + \mu \mathbf{\Delta}_C) \odot (\mathbf{B} + \mu \mathbf{\Delta}_B) \right) (\mathbf{A} + \mu \mathbf{\Delta}_A)^T \right|\right|_F^2 .$$

- Note that the above cost function is a polynomial of degree 6 in μ.
- We can determine the coefficients $c_0, \cdots, c_6$ of this polynomial by evaluating it for 7 different values of μ and solving

$$\begin{bmatrix} 1 & \mu_1 & \mu_1^2 & \cdots & \mu_1^6 \\ 1 & \mu_2 & \mu_2^2 & \cdots & \mu_2^6 \\ & & \vdots & & \\ 1 & \mu_7 & \mu_7^2 & \cdots & \mu_7^6 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_6 \end{bmatrix} = \begin{bmatrix} \ell_1 \\ \ell_2 \\ \vdots \\ \ell_7 \end{bmatrix},$$

where $\ell_1, \cdots, \ell_7$ are the corresponding loss values.

Line Search (contd.)

- Once the coefficients are determined, the derivative is the 5-th order polynomial $c_1 + 2c_2\mu + \cdots + 6c_6\mu^5$
- Use numerical root finding to evaluate the loss at its roots and pick the best μ.

This has a *drawback*:

- It requires 11 evaluations of the loss function.
- We can do half of that by working polynomial coeffs. out analytically
- **Bottom line**: optimal line search costs more than gradient computation *per se* (which roughly corresponds to 3 evaluations of the loss function)

- In practice, we typically use a small, or "good enough" μ.
- We resort to exact line search in more challenging cases (e.g.,"swamps")

Handling Missing Values

- Real data are never perfect
- Parts of data may be *missing* for various reasons
  - Faulty measurement sensors
  - Errors when storing data
  - Partial observation of the data (e.g., recommendation systems/Netflix prize)
  - ...
- Need to modify algorithms to handle missing values
- In the case of recommendation systems, also need to use decomposition to estimate missing values.

Handling Missing Values (contd.)

- Consider the **C**-update step in ALS,

$$\min_{\mathbf{C}} ||\mathbf{X}_3 - (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T||_F^2$$

- If there are missing elements in $\mathbf{X}$ (and so in $\mathbf{X}_3$), define the weight tensor

$$\mathbf{W}(i,j,k) = \begin{cases} 1, & \mathbf{X}(i,j,k): \text{ available} \\ 0, & \text{otherwise.} \end{cases},$$

- We now modify the update step as $\min_{\mathbf{C}} ||\mathbf{W}_3 * (\mathbf{X}_3 - (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T)||_F^2$ where matrix $\mathbf{W}_3$ is the matrix unfolding of tensor $\mathbf{W}$ obtained in the same way that matrix $\mathbf{X}_3$ is obtained by unfolding tensor $\mathbf{X}$

Handling Missing Values (contd.)

$$\min_{\mathbf{C}} ||\mathbf{W}_3 * (\mathbf{X}_3 - (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T)||_F^2$$

- Notice that the Hadamard operation applies to the product $((\mathbf{B} \odot \mathbf{A})\mathbf{C}^T)$, not to $(\mathbf{B} \odot \mathbf{A})$ –
  - This complicates things
- One may think of resorting to column-wise updates, but this does not work!
- Instead, if we perform *row-wise* updates on $\mathbf{C}$, then we have to deal with minimizing over $\mathbf{C}(k,:)$ the squared norm of vector

$$\text{Diag}(\mathbf{W}_3(:,k))\mathbf{X}_3(:,k) - \text{Diag}(\mathbf{W}_3(:,k))(\mathbf{B} \odot \mathbf{A})(\mathbf{C}(k,:))^T,$$

which is a simple linear least squares problem.

Handling Missing Values (contd.)

There are two basic alternatives to the above strategy for handling missing data.

Alternative 1

- Use derivative-based methods, such as (stochastic) gradient descent or Gauss-Newton
  - Derivatives are easy to compute, even in the presence of $\mathbf{W}$
- Stochastic gradient descent, computes gradient estimates by drawing only from the *observed values*
  - This bypasses element-wise multiplication by $\mathbf{W}$, stochastic gradient methods
  - Deals with missing data in a natural and effortless way.
  - Well known in the machine learning community, but seemingly under-appreciated in the signal processing community.

Handling Missing Values (contd.)

Alternative 2

- Use Expectation-Maximization to impute the missing values together with the estimation of the model parameters $\mathbf{A}, \mathbf{B}, \mathbf{C}$ [TB05].
- Initially impute misses with the average of the available entries (or any other reasonable estimate)

Efficiency Considerations

- For very big and sparse data, imputation is very inefficient in terms of memory, and is thus avoided
- As a short-cut in large-scale applications, one may deliberately use only part of the available entries when estimating a decomposition [VDSD14]

Stochastic Gradient Descent (SGD)

- Popular in the machine learning community for many types of convex non-convex problems
- In its simplest form:
  - SGD randomly picks a data point $\mathbf{X}(i, j, k)$ from the available ones,
  - takes a gradient step only for those model parameters that have an effect on $\mathbf{X}(i, j, k)$; that is, only the i-th row of $\mathbf{A}$, the j-th row of $\mathbf{B}$ and the k-th row of $\mathbf{C}$

Stochastic Gradient Descent (SGD) (contd.)

We have

$$\frac{\partial}{\partial \mathbf{A}(i,f)} \left(\mathbf{X}(i,j,k) - \sum_{f=1}^{F} \mathbf{A}(i,f)\mathbf{B}(j,f)\mathbf{C}(k,f) \right)^2 =$$

$$-2 \left(\mathbf{X}(i,j,k) - \sum_{f'=1}^{F} \mathbf{A}(i,f')\mathbf{B}(j,f')\mathbf{C}(k,f') \right) \times \mathbf{B}(j,f)\mathbf{C}(k,f),$$

so that

$$\frac{\partial}{\partial \mathbf{A}(i,:)} = -2 \left(\mathbf{X}(i,j,k) - \sum_{f=1}^{F} \mathbf{A}(i,f)\mathbf{B}(j,f)\mathbf{C}(k,f) \right) \times \left(\mathbf{B}(j,:) * \mathbf{C}(k,:) \right).$$

- $\mathbf{B}(j,:) * \mathbf{C}(k,:)$ is used once outside and once inside the parenthesis
- $2F$ multiplications for the update of $\mathbf{A}(i,:)$, and $6F$ for the SGD update of $\mathbf{A}(i,:)$, $\mathbf{B}(j,:)$, $\mathbf{C}(k,:)$.
- Very cheap, especially for random access memory

Stochastic Gradient Descent (SGD) (contd.)

Missing elements

- SGD naturally deals with missing elements
- They are simply never sampled to execute an update
- This has made SGD very popular in recommendation systems
  - Also because of efficiency in very large, sparse data

Stochastic Gradient Descent (SGD) (contd.)

One major drawback:

- Truly random disk access pattern is a bad idea
- Computation will be dominated from the disk I/O.
- **Solution**: Fetch blocks of data from secondary memory, and use intelligent caching strategies
  - updates involving (stemming from) $\mathbf{X}(i,j,k)$ and $\mathbf{X}(i',j',k')$ do not conflict with each other and can be executed in parallel, provided $i' \neq i,\ j' \neq j,\ k' \neq k$ – where all three $\neq$ must hold simultaneously
  - Max number of parallel updates is $\min(\{I_n\}_{n=1}^{N})$ in the general N-way case
  - See [BTK$^+$] for parallel SGD CPD algorithms

Stochastic Gradient Descent (SGD) (contd.)

Other Relevant Block Sampling Approaches:

- [VD]
  - Efficiently decompose TB-size tensors without resorting to parallel computation
  - Leverages CPD uniqueness of the sampled blocks to uniqueness of the CPD of the full tensor
- [PS12]
  - Randomized block-sampling approach for very sparse datasets
  - parallel CPD decomposition of multiple pseudo-randomly drawn sub-tensors, and combining the CPDs using *anchor rows*
  - Sampling is based on mode densities
  - Identifiability is guaranteed if the sub-tensors have unique CPD

Imposing Constraints

- We are often interested in imposing constraints on a CPD model
- Why do we need this? CPD is essentially unique after all!
- In the next few slides we will review reasons why constraints are useful

Imposing Constraints (contd.)

Reasons to impose constraints:

- **Restoring identifiability** in otherwise non-identifiable cases;
- **Improving estimation accuracy** in relatively challenging (low-SNR, and/or barely identifiable, and/or numerically ill-conditioned) cases;
- Ensuring **interpretability** of the results (e.g., power spectra cannot take negative values); and
- As a remedy against **ill-posedness**

There are many types of constraints that are relevant in many applications. We will review a representative list.

Imposing Constraints (contd.)

Symmetry or Hermitian (conjugate) symmetry:

- Require $\mathbf{B} = \mathbf{A}$, or $\mathbf{B} = \mathbf{A}^*$, leading to $\mathbf{X}(:,:,k) = \mathbf{A}\mathbf{D}_k(\mathbf{C})\mathbf{A}^T$ or $\mathbf{X}(:,:,k) = \mathbf{A}\mathbf{D}_k(\mathbf{C})\mathbf{A}^H$.
  - This is actually only *partial symmetry*
  - Corresponds to joint diagonalization of the frontal slabs, using a non-orthogonal and possibly fat diagonalizer $\mathbf{A}$
- Full symmetry: $\mathbf{C} = \mathbf{B} = \mathbf{A}$.
- Symmetric tensors arise when one considers higher-order statistics (HOS).

Imposing Constraints (contd.)

Real-valued parameters:

- When $\mathbf{X} \in \mathbb{R}^{I \times J \times K}$, complex-valued $\mathbf{A}, \mathbf{B}, \mathbf{C}$ make little sense,
- Sometimes arise because tensor rank is sensitive to the field over which the decomposition is taken
- Issue in some applications in Chemistry and Psychology

Imposing Constraints (contd.)

Element-wise non-negativity:

- $\mathbf{A} \geq 0$ and/or $\mathbf{B} \geq 0$, and/or $\mathbf{C} \geq 0$.
- When all three are in effect, the resulting problem is known as *non-negative tensor factorization* (NTF).
- Non-negativity can help restore uniqueness
  - Even non-negative matrix factorization (NMF) is unique under certain (stricter) conditions
- Example:
  - When $k_{\mathbf{C}} = 1$ CPD alone cannot be unique
  - But if NMF of $\mathbf{X}(:,:,k) = \mathbf{A}\mathbf{D}_k(\mathbf{C})\mathbf{B}^T$ is unique (this requires $F < \min(I, J)$ and a certain level of sparsity in $\mathbf{A}$ and $\mathbf{B}$)
  - non-negativity can still ensure essential uniqueness of $\mathbf{A}, \mathbf{B}, \mathbf{C}$!
- Applications:
  - Modeling power spectra
  - Modeling "sum-of-parts" representation (generally *interpretability*)

Imposing Constraints (contd.)

- **Orthogonality:** This e.g., may be the result of prewhitening [SDC$^+$12].
- **Probability simplex constraints:** $\mathbf{A}(i,:) \geq 0$, $\mathbf{A}(i,:)\mathbf{1} = 1$, $\forall\, i$, or $\mathbf{A}(:,f) \geq 0$, $\mathbf{1}^T\mathbf{A}(:,f) = 1$, $\forall\, f$, are useful when modeling allocations or probability distributions.
- **Linear constraints:** More general linear constraints on $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are also broadly used. Can be column-wise, row-wise, or matrix-wise, such as $\text{tr}(\mathbf{WA}) \leq b$.

Imposing Constraints (contd.)

- **Monotonicity and related constraints:** Useful in cases where we deal with, e.g., concentrations that are known to be decaying, or spectra that are known to have a single or few peaks (unimodality, oligo-modality [BS98]).
- **Sparsity:** In many cases we know (an upper bound on) the number of nonzero elements of $\mathbf{A}, \mathbf{B}, \mathbf{C}$, per column, row, or as a whole; or the number of nonzero columns or rows of $\mathbf{A}, \mathbf{B}, \mathbf{C}$ (group sparsity).
- **Smoothness:** Smoothness can be measured in different ways, but a simple one is in terms of convex quadratic inequalities such as

$$\left\| \begin{bmatrix} -1 & 1 & 0 & \cdots & & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & & \end{bmatrix} \mathbf{A} \right\|_F^2$$

Imposing Constraints (contd.)

Data model constraints:

- All the above constraints apply to the model parameters.
- We may also be interested in constraints on the reconstructed model of the data, e.g.,

$$(\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\mathbf{1} \geq 0, \text{ (element-wise)}, \ \mathbf{1}^T(\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\mathbf{1} = 1,$$

- If $\mathbf{X}$ models a joint probability distribution, or in $(\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\mathbf{1}$ being "smooth" in a suitable sense.

Constrained matrix/tensor factorization

Consider the (low-rank) matrix factorization problem:

$$\mathbf{Y} \approx \mathbf{W}\mathbf{H}^T$$

where the factor matrices $\mathbf{W}$ and $\mathbf{H}$ need to satisfy some constraints: non-negative, sparse, etc.

- Numerous applications: NMF, dictionary learning, ...
- Constraints help resolve the non-uniqueness of unconstrained matrix factorization;
- ... but also *complicate* the problem (even $\rightarrow$ NP-hard, e.g., SVD vs. NMF) and often require *custom* algorithm development.
- We need an algorithmic framework that is more efficient *and* more flexible wrt the types of constraints it can readily accommodate.

We also address tensor counterpart - already unique, but constraints $\rightarrow$ better estimates

$$\mathbf{Y}_{(1)} \approx (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T$$

Problem formulation

Starting with constrained matrix factorization, we formulate the problem as

$$\underset{\mathbf{W},\mathbf{H}}{\text{minimize}} \quad \frac{1}{2}\|\mathbf{Y} - \mathbf{W}\mathbf{H}^T\|_F^2 + r(\mathbf{W}) + r(\mathbf{H}),$$

and $r(\cdot)$ can take the value $+\infty$ to incorporate hard constraints.

- Easily becomes NP-hard when there are constraints.
- Popular method: alternating optimization (AO).
- Key to accelerate algorithm: solve the sub-problems efficiently.

We propose to solve the sub-problems using ADMM,

hence the name AO-ADMM [Huang, Sidiropoulos, Liavas '15]

ADMM: scaled form

Consider a convex optimization problem in the following form.

$$\begin{aligned} \underset{\mathbf{x},\mathbf{z}}{\text{minimize}} \quad & f(\mathbf{x}) + g(\mathbf{z}) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c}, \end{aligned}$$

The alternating direction method of multipliers [Boyd *et al.*, 2011]:

$$\begin{aligned} \mathbf{x} &\leftarrow \arg\min_{\mathbf{x}} f(\mathbf{x}) + (\rho/2)\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \\ \mathbf{z} &\leftarrow \arg\min_{\mathbf{z}} g(\mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} + \mathbf{u}\|_2^2, \\ \mathbf{u} &\leftarrow \mathbf{u} + (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}), \qquad \text{dual update} \end{aligned}$$

For convex problems, converges to the global solution.

ADMM inner-loop

Adopting the alternating optimization framework, reformulate the sub-problem for $\mathbf{H}$ in each AO iteration as

$$\begin{aligned} \underset{\mathbf{H},\tilde{\mathbf{H}}}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{Y} - \mathbf{W}\tilde{\mathbf{H}}\|_F^2 + r(\mathbf{H}) \\ \text{subject to} \quad & \mathbf{H} = \tilde{\mathbf{H}}^T. \end{aligned}$$

ADMM iterates:

$$\begin{aligned} \tilde{\mathbf{H}} &\leftarrow (\mathbf{W}^T\mathbf{W} + \rho\mathbf{I})^{-1}(\mathbf{W}^T\mathbf{Y} + \rho(\mathbf{H} + \mathbf{U})^T), \\ \mathbf{H} &\leftarrow \arg\min_{\mathbf{H}} r(\mathbf{H}) + \frac{\rho}{2}\|\mathbf{H} - \tilde{\mathbf{H}}^T + \mathbf{U}\|_F^2, \\ \mathbf{U} &\leftarrow \mathbf{U} + \mathbf{H} - \tilde{\mathbf{H}}^T. \end{aligned}$$

Inner convergence

Suppose $\mathbf{Y}$ is $m \times n$, $\mathbf{H}$ is $n \times k$.

ADMM for convex problems is well studied.

- Linear convergence. Theoretically, best convergence rate given by $\rho = \sigma_{\max}(\mathbf{W})\sigma_{\min}(\mathbf{W})$;
- Empirically, $\rho = \text{trace}(\mathbf{W}^T\mathbf{W})/k$ works almost as good (and much easier to obtain);
- Initialize $\mathbf{H}$ and $\mathbf{U}$ from the previous AO outer-loop, then optimality gap is bounded by the per-iteration improvement of an AO step;
- After ~ 10 AO outer-iterations, ADMM converges in only one inner-iteration.

Implementation: $\tilde{\mathbf{H}}$

$$\tilde{\mathbf{H}} \leftarrow (\mathbf{W}^T\mathbf{W} + \rho\mathbf{I})^{-1}(\mathbf{W}^T\mathbf{Y} + \rho(\mathbf{H} + \mathbf{U})^T)$$

- $\mathbf{W}^T\mathbf{Y}$ and $\mathbf{W}^T\mathbf{W}$ only need to be computed once, with complexity $\mathcal{O}(\text{nnz}(\mathbf{Y})k)$ and $\mathcal{O}(mk^2)$ respectively;
- Can cache the Cholesky decomposition ($\mathcal{O}(k^3)$) of $\mathbf{W}^T\mathbf{W} + \rho\mathbf{I} = \mathbf{L}\mathbf{L}^T$;
- Within one ADMM iteration, $\tilde{\mathbf{H}}$ is obtained from one forward substitution and one backward substitution, with complexity $\mathcal{O}(nk^2)$.

Implementation: $\mathbf{H}$

$$\mathbf{H} \leftarrow \arg\min_{\mathbf{H}} r_H(\mathbf{H}) + \frac{\rho}{2}\|\mathbf{H} - \tilde{\mathbf{H}}^T + \mathbf{U}\|_F^2$$

so-called proximity operator, in a lot of cases can be efficiently evaluated

- non-negative: $\mathbf{H} \geq 0$;
- l_1 regularization: $\lambda\|\mathbf{H}\|_1$ (soft thresholding);
- sum to one: $\mathbf{H1} = \mathbf{1}$ or $\mathbf{H}^T\mathbf{1} = \mathbf{1}$;
- smoothness regularization: $\lambda\|\mathbf{TH}\|_F^2$ where $\mathbf{T}$ is tri-diagonal (linear complexity using banded-system solver).

All with $\mathcal{O}(nk)$ complexity.

ADMM: summary

1: Initialize $\mathbf{H}$ and $\mathbf{U}$ from previous AO iteration
2: $\mathbf{G} = \mathbf{W}^T\mathbf{W}$, $\mathbf{F} = \mathbf{W}^T\mathbf{Y}$
3: $\rho = \text{trace}(\mathbf{G})/k$
4: Cholesky decomposition: $\mathbf{G} + \rho\mathbf{I} = \mathbf{L}\mathbf{L}^T$
5: **repeat**
6: $\quad \tilde{\mathbf{H}} \leftarrow \mathbf{L}^{-T}\mathbf{L}^{-1}(\mathbf{F} + \rho(\mathbf{H} + \mathbf{U}))$ by forward/backward substitution
7: $\quad \mathbf{H} \leftarrow \arg\min_{\mathbf{H}} r_H(\mathbf{H}) + \frac{\rho}{2}\|\mathbf{H} - \tilde{\mathbf{H}}^T + \mathbf{U}\|_F^2$
8: $\quad \mathbf{U} \leftarrow \mathbf{U} + \mathbf{H} - \tilde{\mathbf{H}}$
9: **until** convergence
10: **return H** and **U**.

- Most of the computations are done at line 2 and line 4;
- One iteration of ADMM has complexity = one ALS update;
- Only one matrix inversion is required (line 4).

Constrained PARAFAC

It is easy to extend from 2-way to higher-way data arrays.
For 3-way tensors, the update of $\mathbf{A}$ becomes

$$\underset{\mathbf{A}}{\text{minimize}} \quad \frac{1}{2}\|\mathbf{Y}_{(1)} - (\mathbf{C} \odot \mathbf{B})\mathbf{A}^T\|_F^2 + r(\mathbf{A})$$

Same sub-problem as the matrix case by letting $\mathbf{Y} = \mathbf{Y}_{(1)}$ and $\mathbf{W} = (\mathbf{C} \odot \mathbf{B})$, except that there are additional structures in $\mathbf{W}$.

- $\mathbf{W}^T\mathbf{W} = \mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B}$, element-wise product;
- $\mathbf{W}^T\mathbf{Y} = (\mathbf{C} \odot \mathbf{B})^T\mathbf{Y}$, usually the bottle-neck operation even in the unconstrained factorization case.

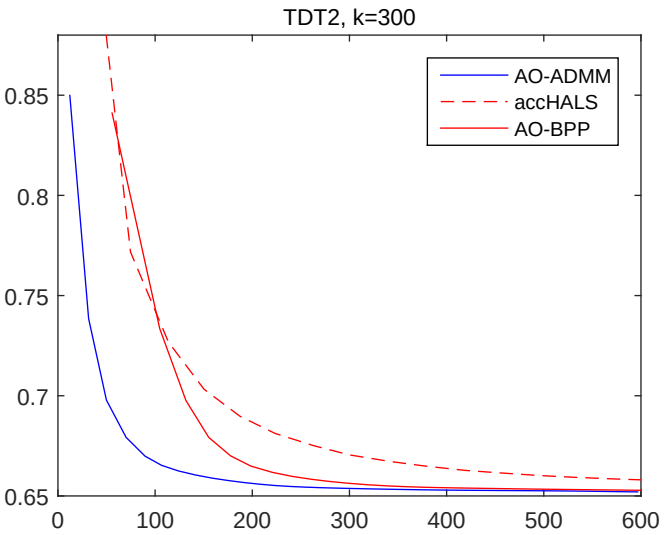
AO-ADMM: Reprise

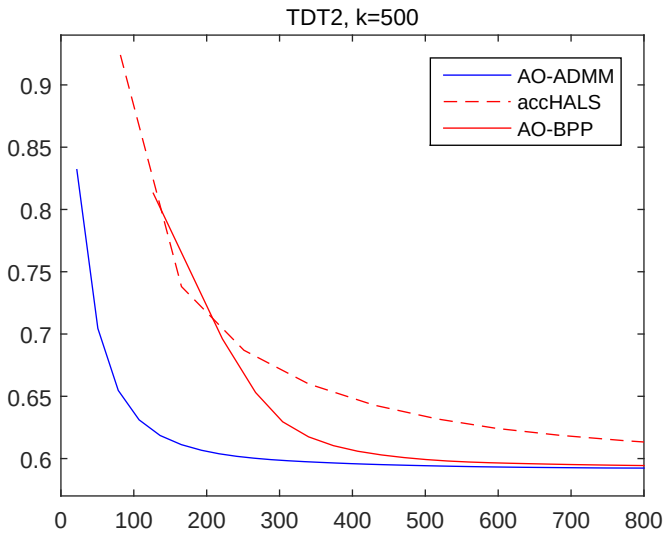
AO-ADMM can be viewed as a generalization of the workhorse ALS method:

- similar monotonic convergence property (due to AO);
- per-iteration complexity is almost the same;
- any smart implementation of ALS can easily be modified to handle constraints with the corresponding proximity operator: **plug-and-play!!**

Non-negative Matrix Factorization(NMF): real data

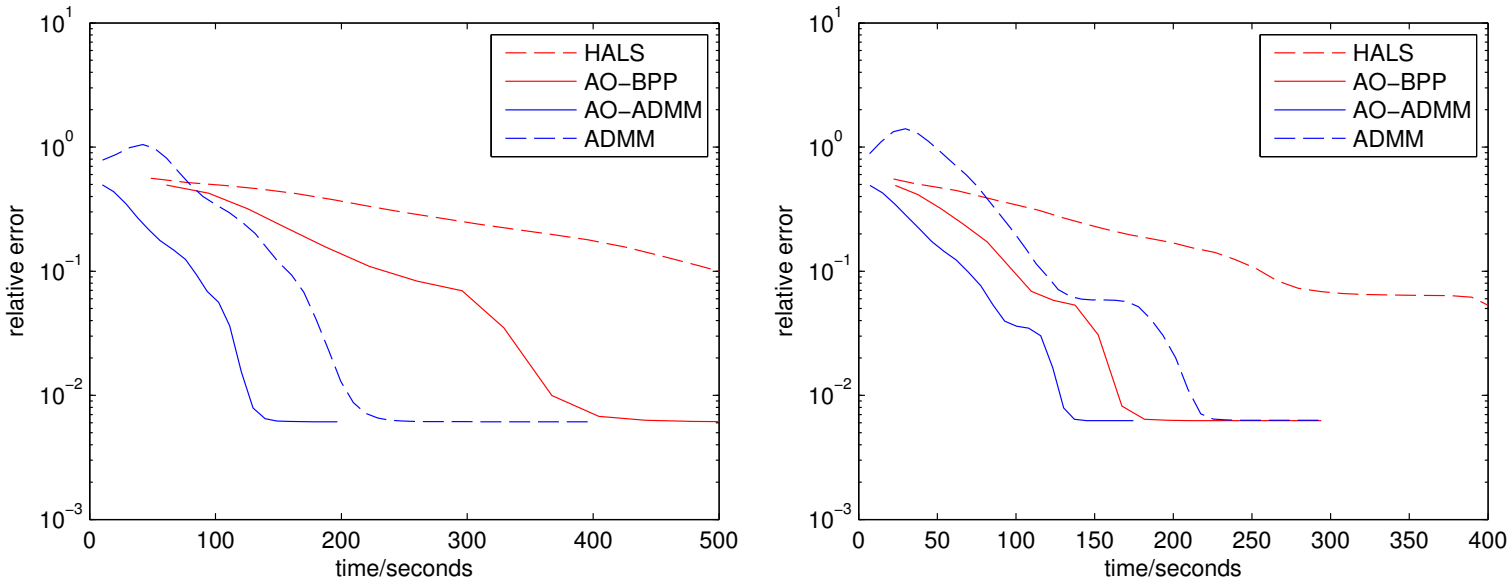
Sparse real data: Topic Detection and Tracking 2 (TDT2) text corpus, of size 10212×36771.





Non-negative PARAFAC: synthetic data

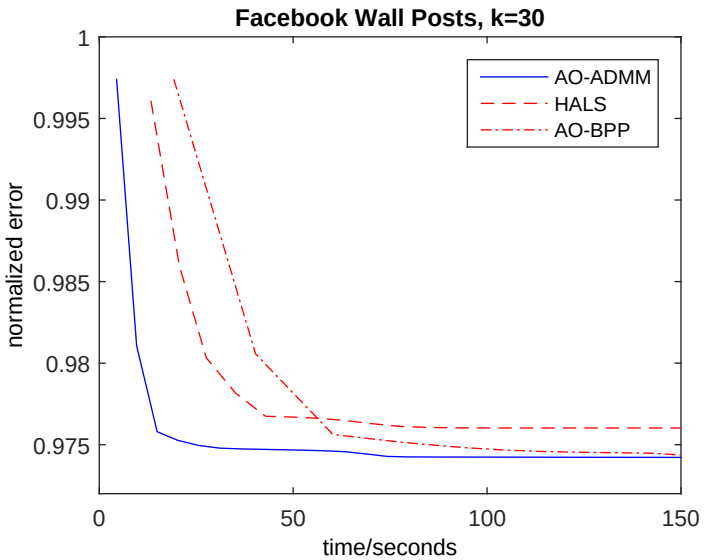
Synthetically generate the true factors randomly $\sim$ i.i.d. exponential elements, with 50% zeros. Then added Gaussian noise with variance 0.01.

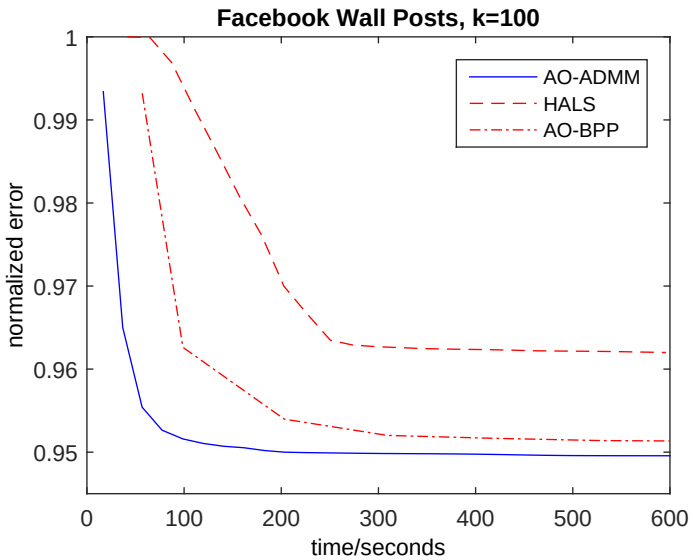


(a) $500 \times 500 \times 500$, rank 100 (b) $1000 \times 500 \times 200$, rank 100

Non-negative PARAFAC: real data

Sparse real data: Facebook Wall Posts, of size $46952 \times 46951 \times 1592$.
Used `tensor_toolbox`* to handle basic sparse tensor operations.





* B. W. Bader, T. G. Kolda, *et al.*, "Matlab tensor toolbox v2.6", Feb. 2015.

AO-ADMM: recap

- **Efficiency:** Uses ADMM to solve the subproblems, with computation caching, warm start, and good choice of ρ;
- **Flexibility:**
  - Can handle many constraints on the latent factors with element-wise complexity;
  - Can handle non-LS loss: missing values, l_1 fitting, K-L divergence...
- **Convergence:**
  - Monotonic decrease of the cost function;
  - Adopting block successive upperbound minimization (BSUM) as the AO framework, we can guarantee that every limit point is a stationary point.
- **Bottleneck:** MTTKRP - same as ALS. So ...

Back to ALS: MTTKRP

$$\mathbf{C} = \mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A})(\mathbf{B}^T\mathbf{B} * \mathbf{A}^T\mathbf{A})^\dagger$$
$$\mathbf{A} = \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B})(\mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B})^\dagger$$
$$\mathbf{B} = \mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A})(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A})^\dagger$$

- Computation and inversion of $(\mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B})$ relatively easy: relatively small $K \times F$, $J \times F$ matrices, invert $F \times F$ matrix, usually F is small
- Bottleneck is computing $\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B})$; likewise $\mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A})$, $\mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A})$
- Entire $\underline{\mathbf{X}}$ needs to be accessed for each computation in each ALS iteration, data transport costs
- Memory access pattern is different for the three computations, making efficient block caching very difficult
- 'Solution': replicate data three times in main (fast) memory :-(

MTTKRP - Prior work

- Kolda *et al*, 2008 compute $\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B})$ with $3F\ \mathrm{NNZ}$ flops using NNZ intermediate memory (on top of that required to store the tensor). Does not provision for efficient parallelization (accumulation step must be performed serially)
- Kang *et al*, 2012 compute $\mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B})$ with $5F\ \mathrm{NNZ}$ flops using $O(\max(J + \mathrm{NNZ}, K + \mathrm{NNZ}))$ intermediate memory. Parallel implementation.
- Choi *et al*, 2014 (DeFacTo): $2F(\mathrm{NNZ} + P)$ flops, using $(2\mathrm{NNZ} + I + 3P + 2)$ memory, where P is the number of non-empty J-mode fibers. Parallel implementation.
- Room for considerable improvements in terms of memory- and computation-efficiency, esp. for high-performance parallel computing architectures

[Ravindarn, Sidiropoulos, Smith, Karypis '14]: Suite of three agorithms

- Algorithm 1: **Output:** $\mathbf{M}_1 \leftarrow \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}) \in \mathbb{R}^{I \times F}$

  1: $\mathbf{M}_1 \leftarrow \mathbf{0}$
  2: **for** $k = 1, \ldots, K$ **do**
  3: $\quad \mathbf{M}_1 \leftarrow \mathbf{M}_1 + \underline{\mathbf{X}}(:, :, k)\mathbf{B}\,\mathrm{diag}(\mathbf{C}(k, :))$
  4: **end for**

- Algorithm 2: **Output:** $\mathbf{M}_2 \leftarrow \mathbf{X}_{(2)}(\mathbf{C} \odot \mathbf{A}) \in \mathbb{R}^{J \times F}$

  1: $\mathbf{M}_2 \leftarrow \mathbf{0}$
  2: **for** $k = 1, \ldots, K$ **do**
  3: $\quad \mathbf{M}_2 \leftarrow \mathbf{M}_2 + \underline{\mathbf{X}}(:, :, k)\mathbf{A}\mathrm{diag}(\mathbf{C}(k, :))$
  4: **end for**

- Algorithm 3: **Output:** $\mathbf{M}_3 \leftarrow \mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A}) \in \mathbb{R}^{K \times F}$

  1: **for** $k = 1, \ldots, K$ **do**
  2: $\quad \mathbf{M}_3(k, :) \leftarrow \mathbf{1}^T (\mathbf{A} * (\underline{\mathbf{X}}(:, :, k)\mathbf{B}))$
  3: **end for**

Features

- Essentially no additional intermediate memory needed - updates of $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ can be effectively performed in place
- Let I_k be the number of non-empty rows and J_k be the number of non-empty columns in $\underline{\mathbf{X}}(:,:,k)$ and define $\mathrm{NNZ}_1 := \sum_{k=1}^{K} I_k$, $\mathrm{NNZ}_2 := \sum_{k=1}^{K} J_k$
- Algorithm 1: $F\ \mathrm{NNZ}_1 + F\ \mathrm{NNZ}_2 + 2F\ \mathrm{NNZ}$ flops. Kang: $5F\mathrm{NNZ}$; Kolda $3F\mathrm{NNZ}$. Note $\mathrm{NNZ} > \mathrm{NNZ}_1, \mathrm{NNZ}_2$

Features

- Algorithms 1, 2, 3 share the same tensor data access pattern - enabling efficient orderly block caching / pre-fetching if the tensor is stored in slower / serially read memory, without need for three-fold replication ($\rightarrow$ asymmetry between Algorithms 1, 2 and Algorithm 3)
- The loops can be parallelized across K threads, where each thread only requires access to an $I \times J$ slice of the tensor. This favors parallel computation and distributed storage.

Imposing Constraints (contd.)

Parametric constraints:

- All the above are *non-parametric* constraints.
- There are also important *parametric constraints* that often arise, particularly in signal processing applications.
- Examples:
  1. Vandermonde or Toeplitz structure imposed on $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$
     - This may reduce the number of parameters needed and suppress noise
  2. Non-negativity can be parametrized as $\mathbf{A}(i,j) = \theta_{i,j}^2, \ \theta \in \mathbb{R}$
  3. Magnitude constraint $|\mathbf{A}(i,j)| = 1$ as $\mathbf{A}(i,j) = e^{\sqrt{-1}\theta_{i,j}}$
  4. Orthogonality may be parametrized via Jacobi rotations or Householder reflections.
  5. Smoothness, probability simplex, and linear constraints can be formulated as parametric constraints

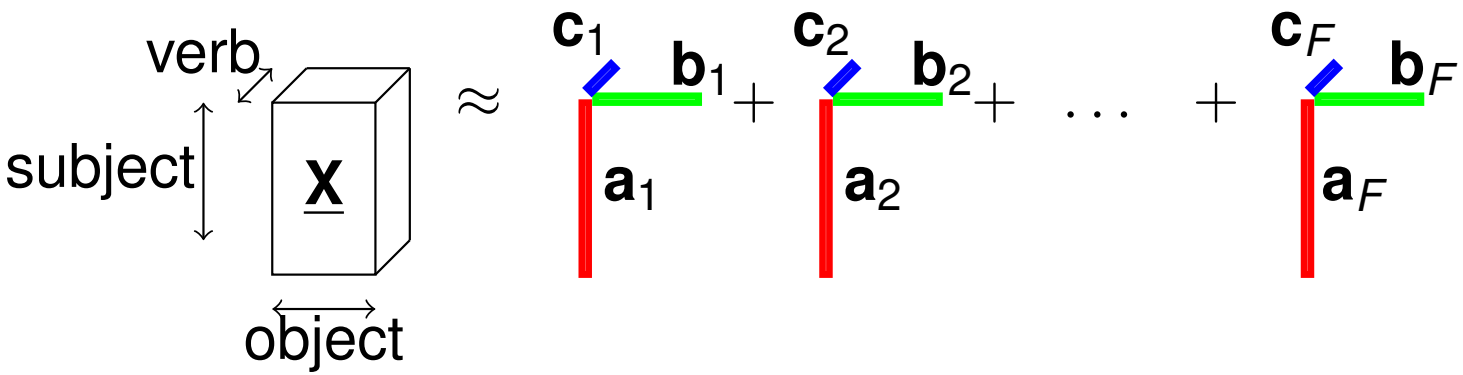
Reprise – what's coming up next

Applications in Machine Learning

- Knowledge base completion.
- Recommender systems - collaborative filtering.
- Factorization machines - multilinear classification.
- Gaussian mixture model parameter estimation.
- Topic mining.
- Multilinear subspace learning.

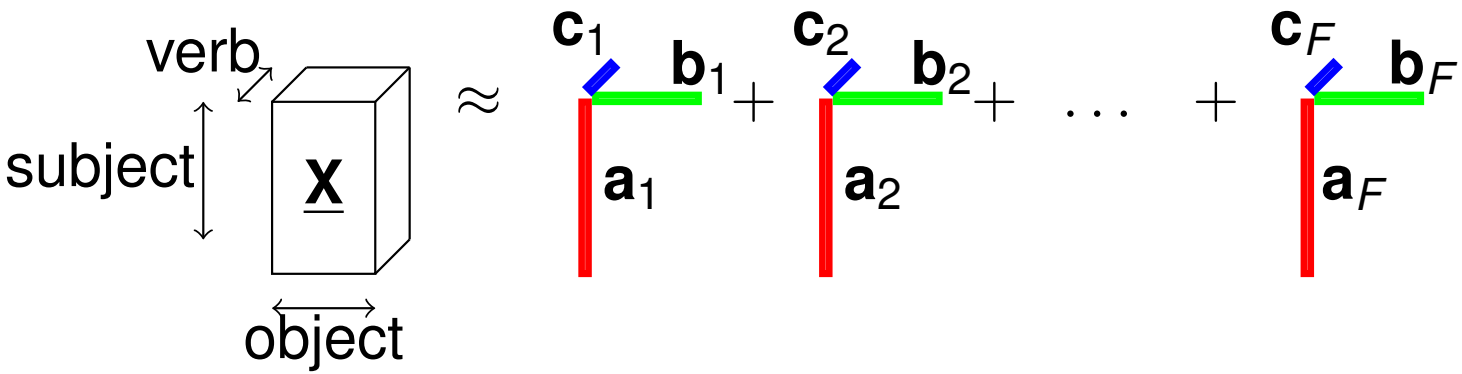
Knowledge Base Completion

- NELL – Tom Mitchell @ CMU
- Given known facts of (subject, verb, object)-type such as (Obama, likes, football), (Donald, likes, football), (Donald, is, American) infer additional knowledge, such as (Obama, is, American).



- You obviously need a model for this ... and the most basic algebraic model is low-rank ... but does low-rank make sense in this context?

Knowledge Base Completion

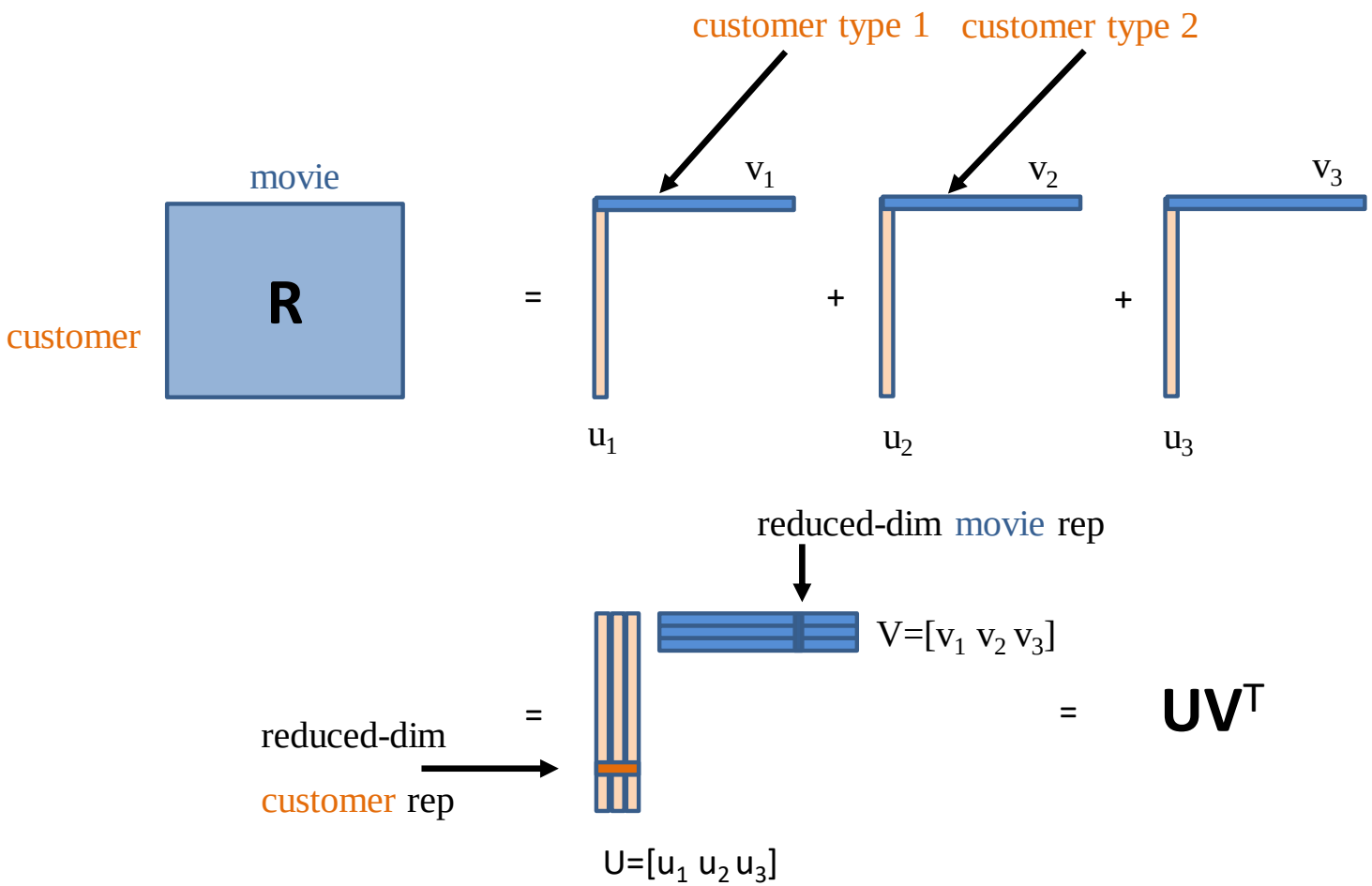


- Think ... about matrix case in which you unfold verb-object in one long mode: the long rows for Obama and Donald are similar, so you copy entries from one to the other to make them identical $\leftrightarrow$ matrix completion!
- If you don't unfold $\leftrightarrow$ tensor completion.

Recommender Systems: Collaborative Filtering

- Given users $\times$ movies ($I \times J$) ratings matrix $\mathbf{R}$.
- Low-rank approx. $\mathbf{R} \approx \mathbf{U}\mathbf{V}^T$; $\mathbf{U}$ ($I \times F$), $\mathbf{V}$ ($J \times F$), $F << \min(I, J)$.
- $\mathbf{U}(i, :)$: reduced-dim latent description of user i;
- $\mathbf{V}(j, :)$: reduced-dim latent description of movie j;
- $\mathbf{R} \approx \mathbf{U}\mathbf{V}^T$ implies that user i's rating of movie j is approximated as $\mathbf{R}(i, j) \approx \mathbf{U}(i, :)(\mathbf{V}(j, :))^T$, i.e., the inner product of the latent descriptions of the i-th user and the j-th movie. Intuitive!
- Premise: every user is a linear combination of few (F) user "types" (e.g., sports fan, college student, ... $\leftrightarrow$ rows of $\mathbf{V}^T$ / columns of $\mathbf{V}$).
- Every movie is a linear combination of few movie types (e.g., comedy, drama, documentary, ... $\leftrightarrow$ columns of $\mathbf{U}$). Intuitive!

Collaborative Filtering: Schematic



Fit model; predict using $R(i,j) = U(i,:)(V(j,:))^T = \sum_{f=1}^{F} u_f(i)v_f(j)$

Recommender Systems: Collaborative Filtering

- Only very small % of the entries of $\mathbf{R}$ is available – between 1 per thousand and 1 per 10^5 in practice. Few people provide feedback.
- Goal: predict a user's missing ratings using not only that user's past ratings but also the ratings of all other users – hence the term *collaborative filtering*.
- If we can find $\mathbf{U}$ and $\mathbf{V}$ from the available ratings, then we can impute the missing ones using inner products of columns of $\mathbf{U}$ and $\mathbf{V}$. This suggests using the following formulation.

$$\min_{\mathbf{U},\,\mathbf{V}} \left|\left| \mathbf{W} * \left(\mathbf{R} - \mathbf{U}\mathbf{V}^T \right) \right|\right|_F^2,$$

where $\mathbf{W}(i,j) = 1$ if $\mathbf{R}(i,j)$ is available, 0 otherwise.

Recommender Systems: Collaborative Filtering

- In practice, unclear what would be good F.
- Overestimate, use rank penalty to control over-fitting and improve generalization.
- Rank of $\mathbf{X}$ = # nonzero singular values of $\mathbf{X}$.
- Nuclear norm $||\mathbf{X}||_*$ (sum of singular values) is commonly used convex surrogate ($||\cdot||_1$ vs. $||\cdot||_0$ of vector of singular values).
- $||\mathbf{X}||_* = \min_{\mathbf{U},\mathbf{V} \mid \mathbf{X}=\mathbf{U}\mathbf{V}^T} \frac{1}{2}\left(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2\right)$, so
-
$$\min_{\mathbf{U},\,\mathbf{V}} \left|\left|\mathbf{W} * \left(\mathbf{R} - \mathbf{U}\mathbf{V}^T\right)\right|\right|_F^2 + \frac{\lambda}{2}\left(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2\right).$$

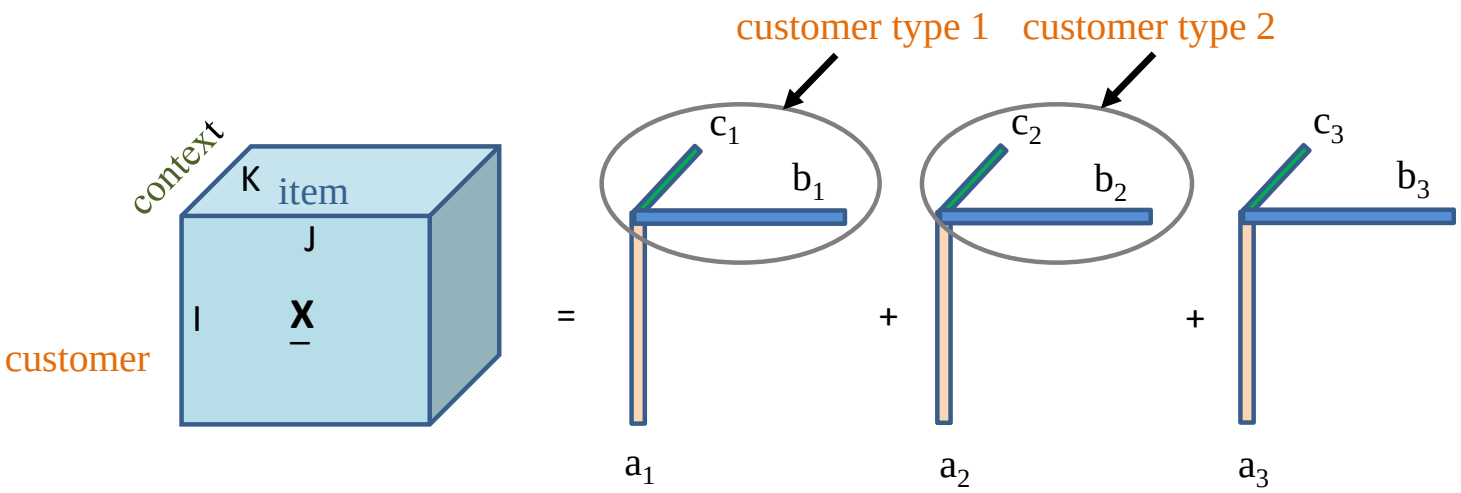
Recommender Systems: Collaborative Filtering

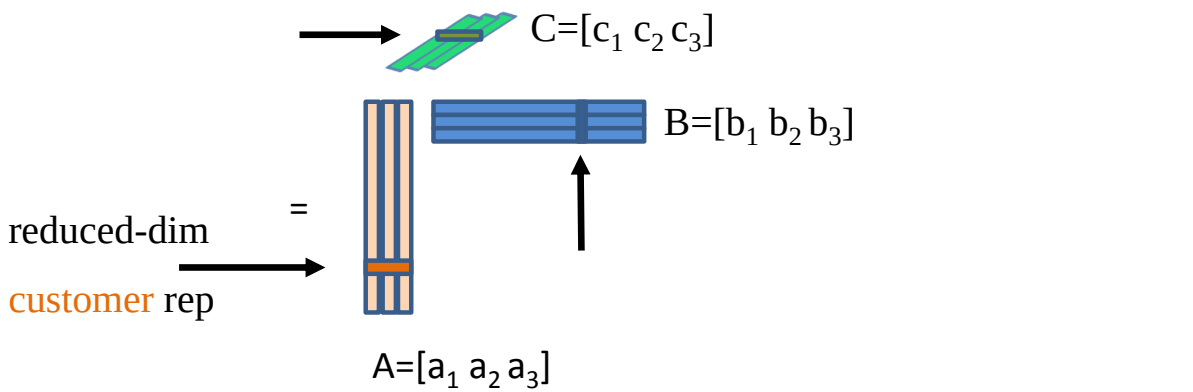
- Additional information on the context in which the ratings were given is often available: time, social context, gender, ...
- Every context type +1 mode $\rightarrow$ very sparse higher-order tensors.
- Bummer: approximating very sparse may require very high rank ...
- Analogy: matrix $\mathbf{I}$ is full rank.
- CPD of user $\times$ movie $\times$ time tensor $\mathbf{R}$ w/ elements $\mathbf{R}(i,j,k)$

$$\min_{\mathbf{A},\,\mathbf{B},\,\mathbf{C}} \sum_{k=1}^{K} \left|\left| \mathbf{W}(:,:,k) * \left(\mathbf{R}(:,:,k) - \mathbf{A}\mathbf{D}_k(\mathbf{C})\mathbf{B}^T \right) \right|\right|_F^2,$$

- Xiong *et al.*: rank + smoothness regularization, Bayesian MCMC.

Machine learning interpretation of CPD





$$X(i,j,k) = \sum_{f=1}^{F} a_f(i) b_f(j) c_f(k)$$

From Matrix to Tensor: Context

- Karatzoglou *et al.*: used age as the third mode ($<$ 18, 18-50, and $>$ 50), and non-$\perp$ Tucker instead of CPD.

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W},\mathbf{G}} ||\Gamma * (\mathbf{R} - (\mathbf{U},\mathbf{V},\mathbf{W},\mathbf{G}))||_F^2 +$$

$$\lambda \left(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2 + ||\mathbf{W}||_F^2 \right) + \mu ||\mathbf{G}||_F^2,$$

- Here $(\mathbf{U},\mathbf{V},\mathbf{W},\mathbf{G})$ stands for Tucker model generated by $\mathbf{U},\mathbf{V},\mathbf{W},\mathbf{G}$;
- $||\mathbf{X}||_F^2$ is the sum of squared elements of tensor $\mathbf{X}$.
- Note: $||\mathbf{U}||_F^2 = \text{Tr}(\mathbf{U}^T\mathbf{U})$, so $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ resists rank regularization.

Multilinear classification

- In support vector machine (SVM) classification, we use a linear mapping $\mathbf{w}^T\mathbf{x} = \sum_i \mathbf{w}(i)\mathbf{x}(i)$ to discriminate between vectors belonging to two different classes: sign$(\mathbf{w}^T\mathbf{x} - b)$.
- Augmenting each $\mathbf{x}$ with a unit as last element (i.e., replacing $\mathbf{x}$ by $[\mathbf{x}, 1]^T$) and likewise absorbing $-b$ in $\mathbf{w} \rightarrow$ sign$(\mathbf{w}^T\mathbf{x})$.
- Classifier design: choice of vector $\mathbf{w}$.
- Using a measure of classification error as cost function.
- Hinge loss plus $\lambda||\mathbf{w}||_2^2$ for the popular *soft margin* approach.
- What if classes are not linearly separable?
- Standard ML approach: use kernels and the kernel trick.

Multilinear classification

- What if classes are not linearly separable?
- Systems approach?
- Think about Taylor series ... higher-order polynomial approximation ... multivariate polynomial functions.
- First step: linear-quadratic function, also taking into account all pairwise products of input variables.
- Bilinear mapping $\mathbf{x}^T\mathbf{W}\mathbf{x} = \sum_{i,j} \mathbf{W}(i,j)\mathbf{x}(i)\mathbf{x}(j) = \text{vec}\left(\mathbf{x}^T\mathbf{W}\mathbf{x}\right) = \left(\mathbf{x}^T \otimes \mathbf{x}^T\right) \text{vec}(\mathbf{W}) = (\mathbf{x} \otimes \mathbf{x})^T \text{vec}(\mathbf{W})$.
- Augmenting $\mathbf{x}$ with a unit as last element (i.e., replacing $\mathbf{x}$ by $[\mathbf{x}, 1]^T$), higher-order mappings include lower-order ones.

Multilinear classification

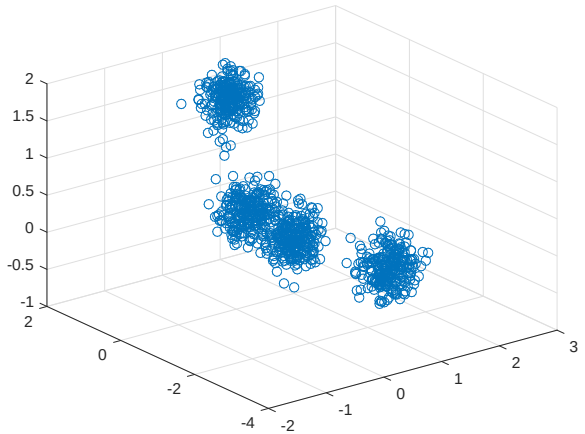
- More generally, multilinear mapping, e.g., $(\mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x})^T \text{vec}(\mathbf{W}) = \sum_{i,j,k} \mathbf{W}(i,j,k)\mathbf{x}(i)\mathbf{x}(j)\mathbf{x}(k)$.
- Classifier design problem boils down to designing a suitable matrix or tensor $\mathbf{W}$ of weights.
- In order to em keep the number of model parameters low relative to the number of training samples (to enable statistically meaningful learning and generalization), a low-rank tensor model such as CPD (Steffen Rendle, 2010), or low multilinear rank one such as Tucker can be employed.
- Model parameters learned using a measure of classification error as cost function.
- A simple optimization solution is to use SGD, drawing samples from the training set at random.

Gaussian mixture parameter estimation

- Consider F Gaussians $\mathcal{N}(\boldsymbol{\mu}_f, \sigma_f^2 \mathbf{I})$, where $\boldsymbol{\mu}_f \in \mathbb{R}^{I \times 1}$ is the mean vector and σ_f^2 is the variance of the elements of the f-th Gaussian.
- Let $\boldsymbol{\pi} = [\pi_1, \cdots, \pi_F]^T$ be a prior distribution.
- Experiment: first draw $f_m \sim \boldsymbol{\pi}$; then draw $\mathbf{x}_m \sim \mathcal{N}(\boldsymbol{\mu}_{f_m}, \sigma_{f_m}^2 \mathbf{I})$.
- The distribution of $\mathbf{x}_m$ is then a mixture of the F Gaussians, i.e., $\sum_{f=1}^{F} \pi_f \mathcal{N}(\boldsymbol{\mu}_f, \sigma_f^2 \mathbf{I})$.

Gaussian mixture parameter estimation

- Run M independent trials of the above experiment $\rightarrow \{\mathbf{x}_m\}_{m=1}^{M}$.



- Given $\{\mathbf{x}_m\}_{m=1}^{M}$, estimate mixture parameters $\{\boldsymbol{\mu}_f, \sigma_f^2, \pi_f\}_{f=1}^{F}$.
- Can also estimate F from $\{\mathbf{x}_m\}_{m=1}^{M}$, but assume F given to ease burden.
- Note conceptual similarity to k-means (here: F-means) clustering or vector quantization (VQ): main difference is that here we make an additional modeling assumption that the "point clouds" are isotropic Gaussian about their means.

Gaussian mixture parameter estimation

- Consider

$$E[\mathbf{x}_m] = \sum_{f_m=1}^{F} E[\mathbf{x}_m|f_m]\pi_{f_m} = \sum_{f=1}^{F} \boldsymbol{\mu}_f \pi_f = \mathbf{M}\boldsymbol{\pi},$$

where $\mathbf{M} := [\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_F]$ $(I \times F)$. Next, consider

$$E[\mathbf{x}_m\mathbf{x}_m^T] = \sum_{f_m=1}^{F} E[\mathbf{x}_m\mathbf{x}_m^T|f_m]\pi_{f_m} = \sum_{f=1}^{F} \left(\boldsymbol{\mu}_f\boldsymbol{\mu}_f^T + \sigma_f^2\mathbf{I}\right)\pi_f$$

$$= \mathbf{M}\text{Diag}(\boldsymbol{\pi})\mathbf{M}^T + \bar{\sigma}^2\mathbf{I}, \ \bar{\sigma}^2 := \sum_{f=1}^{F} \sigma_f^2\pi_f.$$

It is tempting to consider third-order moments, which are easier to write out in scalar form

$$E[\mathbf{x}_m(i)\mathbf{x}_m(j)\mathbf{x}_m(k)] = \sum_{f=1}^{F} E[\mathbf{x}_m(i)\mathbf{x}_m(j)\mathbf{x}_m(k)|f]\pi_f.$$

Gaussian mixture parameter estimation

- Conditioned on f, $\mathbf{x}_m(i) = \boldsymbol{\mu}_f(i) + \mathbf{z}_m(i)$, where $\mathbf{z}_m \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 \mathbf{I})$, and likewise for $\mathbf{x}_m(j)$ and $\mathbf{x}_m(k)$. Plugging these back into the above expression, and using that
  ▷ If two out of three indices i, j, k are equal, then $E[\mathbf{z}_m(i)\mathbf{z}_m(j)\mathbf{z}_m(k)|f] = 0$, due to zero mean and independence of the third; and
  ▷ If all three indices are equal, then $E[\mathbf{z}_m(i)\mathbf{z}_m(j)\mathbf{z}_m(k)|f] = 0$ because the third moment of a zero-mean Gaussian is zero, we obtain

$$E[\mathbf{x}_m(i)\mathbf{x}_m(j)\mathbf{x}_m(k)|f] = \boldsymbol{\mu}_f(i)\boldsymbol{\mu}_f(j)\boldsymbol{\mu}_f(k) +$$
$$\sigma_f^2 \left(\boldsymbol{\mu}_f(i)\delta(j-k) + \boldsymbol{\mu}_f(j)\delta(i-k) + \boldsymbol{\mu}_f(k)\delta(i-j)\right),$$

  where $\delta(\cdot)$ is the Kronecker delta.

Gaussian mixture parameter estimation

- Averaging over π_f,

$$\mathbf{R}(i,j,k) := E[\mathbf{x}_m(i)\mathbf{x}_m(j)\mathbf{x}_m(k)] = \sum_{f=1}^{F} \pi_f \boldsymbol{\mu}_f(i)\boldsymbol{\mu}_f(j)\boldsymbol{\mu}_f(k) +$$

$$\sum_{f=1}^{F} \pi_f \sigma_f^2 \left(\boldsymbol{\mu}_f(i)\delta(j-k) + \boldsymbol{\mu}_f(j)\delta(i-k) + \boldsymbol{\mu}_f(k)\delta(i-j)\right).$$

- Further assume, for simplicity, that $\sigma_f^2 = \sigma^2$, $\forall f$, and σ^2 is known. Then $\sum_{f=1}^{F} \pi_f \boldsymbol{\mu}_f(i) = E[\mathbf{x}_m(i)]$ can be easily estimated. So we may pre-compute the second term in the above equation, call it $\boldsymbol{\Gamma}(i,j,k)$, and form

$$\mathbf{R}(i,j,k) - \boldsymbol{\Gamma}(i,j,k) = \sum_{f=1}^{F} \pi_f \boldsymbol{\mu}_f(i)\boldsymbol{\mu}_f(j)\boldsymbol{\mu}_f(k),$$

which is evidently a symmetric CPD model of rank (at most) F.

Gaussian mixture parameter estimation

$$\mathbf{R}(i,j,k) - \mathbf{\Gamma}(i,j,k) = \sum_{f=1}^{F} \pi_f \boldsymbol{\mu}_f(i)\boldsymbol{\mu}_f(j)\boldsymbol{\mu}_f(k).$$

- Note that, due to symmetry and the fact that $\pi_f \geq 0$, there is no ambiguity regarding the sign of $\boldsymbol{\mu}_f$; but we can still set e.g., $\boldsymbol{\mu}_1^{'} = \rho^{\frac{1}{3}}\boldsymbol{\mu}_1$, $\pi_1^{'} = \frac{1}{\rho}\pi_1$, $\pi_2^{'} = \pi_1 + \pi_2 - \pi_1^{'} = \frac{\rho-1}{\rho}\pi_1 + \pi_2$, $\frac{1}{\gamma} = \frac{\pi_2^{'}}{\pi_2}$, and $\boldsymbol{\mu}_2^{'} = \gamma^{\frac{1}{3}}\boldsymbol{\mu}_2$, for some $\rho > 0$.
- However, we must further ensure that $\pi_2^{'} > 0$, and $\pi_1^{'} < \pi_1 + \pi_2$; both require $\rho > \frac{\pi_1}{\pi_1+\pi_2}$.
- We see that scaling ambiguity remains, and is important to resolve it here, otherwise we will obtain the wrong means and mixture probabilities.

Gaussian mixture parameter estimation

- Towards this end, consider lower-order statistics, namely $E[\mathbf{x}_m \mathbf{x}_m^T]$ and $E[\mathbf{x}_m]$. Note that,

$$(\mathbf{M} \odot \mathbf{M} \odot \mathbf{M})\boldsymbol{\pi} = ((\mathbf{M}\mathbf{D}^{1/3}) \odot (\mathbf{M}\mathbf{D}^{1/3}) \odot (\mathbf{M}\mathbf{D}^{1/3}))\mathbf{D}^{-1}\boldsymbol{\pi}$$

but

$$E[\mathbf{x}_m] = \mathbf{M}\boldsymbol{\pi} \neq (\mathbf{M}\mathbf{D}^{1/3})\mathbf{D}^{-1}\boldsymbol{\pi},$$

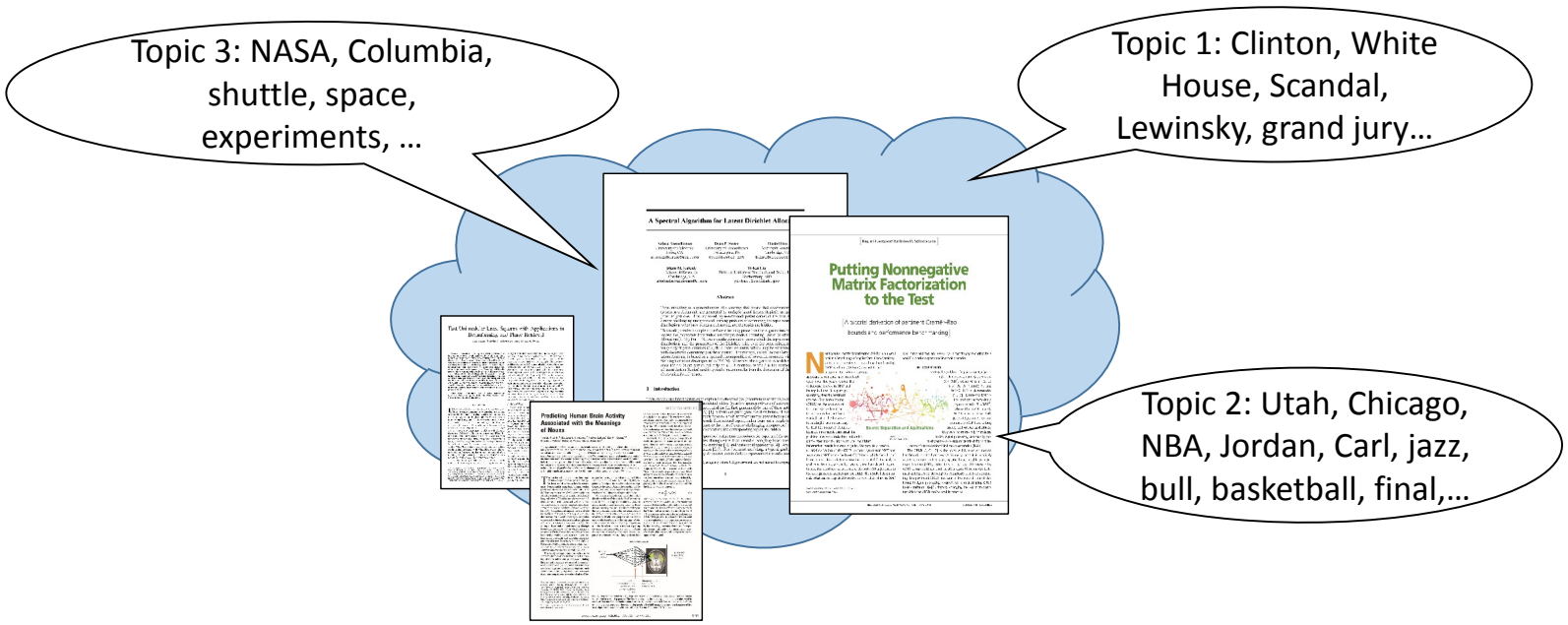
$$E[\mathbf{x}_m \mathbf{x}_m^T] - \bar{\sigma}^2 \mathbf{I} = \mathbf{M}\mathrm{Diag}(\boldsymbol{\pi})\mathbf{M}^T$$

$$\overset{\mathrm{vec}(\cdot)}{\rightarrow} (\mathbf{M} \odot \mathbf{M})\boldsymbol{\pi} \neq ((\mathbf{M}\mathbf{D}^{1/3}) \odot (\mathbf{M}\mathbf{D}^{1/3}))\mathbf{D}^{-1}\boldsymbol{\pi}.$$

This shows that no scaling ambiguity remains when we jointly fit third and second (or third and first) order statistics.

- For the general case, when the variances $\{\sigma_f^2\}_{f=1}^F$ are unknown and possibly different, see [Hsu, 2013]. A simpler work-around is to treat "diagonal slabs" (e.g., corresponding to $j = k$) as missing, fit the model, then use it to estimate $\{\sigma_f^2\}_{f=1}^F$ and repeat.

Topic Mining



Topic Mining

Table: Mined topics from 5 classes of (1,683) articles of the TDT2 corpus.

FastAnchor (classic)					AnchorFree (proposed)				
allegations	poll	columbia	gm	bulls	lewinsky	gm	shuttle	bulls	jonesboro
lewinsky	cnnusa	shuttle	motors	jazz	monica	motors	space	jazz	arkansas
clinton	gallup	space	plants	nba	starr	plants	columbia	nba	school
lady	allegations	crew	workers	utah	grand	flint	astronauts	chicago	shooting
white	clinton	astronauts	michigan	finals	white	workers	nasa	game	boys
hillary	presidents	nasa	flint	game	jury	michigan	crew	utah	teacher
monica	rating	experiments	strikes	chicago	house	auto	experiments	finals	students
starr	lewinsky	mission	auto	jordan	clinton	plant	rats	jordan	westside
house	president	stories	plant	series	counsel	strikes	mission	malone	middle
husband	approval	fix	strike	malone	intern	gms	nervous	michael	11year
dissipate	starr	repair	gms	michael	independent	strike	brain	series	fire
president	white	rats	idled	championship	president	union	aboard	championship	girls
intern	monica	unit	production	tonight	investigation	idled	system	karl	mitchell
affair	house	aboard	walkouts	lakers	affair	assembly	weightlessness	pippen	shootings
infidelity	hurting	brain	north	win	lewinskys	production	earth	basketball	suspects
grand	slipping	system	union	karl	relationship	north	mice	win	funerals
jury	americans	broken	assembly	lewinsky	sexual	shut	animals	night	children
sexual	public	nervous	talks	games	ken	talks	fish	sixth	killed
justice	sexual	cleansing	shut	basketball	former	autoworkers	neurological	games	13year
obstruction	affair	dioxide	striking	night	starrs	walkouts	seven	title	johnson

- K. Huang*, X. Fu* and N.D. Sidiropoulos, "**Anchor-Free Correlated Topic Modeling: Identifiability and Algorithm**", NIPS 2016. (*equal contribution)

Topic Mining

- Given a dictionary $\mathcal{D} = \{w_1, \cdots, w_I\}$ comprising I possible words, a *topic* is a probability mass function (pmf) over $\mathcal{D}$.
- Assume there are F topics overall, let $\mathbf{p}_f := \Pr(w_i|f)$ be the pmf associated with topic f, π_f be the probability that one may encounter a document associated with topic f, and $\boldsymbol{\pi} := [\pi_1, \cdots, \pi_f]^T$.
- We begin our discussion of topic modeling by assuming that each document is related to one and only one topic (or, document "type").

Topic Mining

- Consider the following experiment:
  1) Draw a document at random;
  2) Sample m words from it, independently, and at random (with replacement – the order in which words are drawn does not matter);
  3) Repeat (until you collect "enough samples" – to be qualified later).

Topic Mining

- Assume for the moment that F is known. Goal is to estimate $\{\mathbf{p}_f, \pi_f\}_{f=1}^F$.
- Clearly, $\Pr(w_i) = \sum_{f=1}^F \Pr(w_i|f)\pi_f$.
- Furthermore, the word co-occurrence probabilities $\Pr(w_i, w_j) :=$Pr(word i and word j are drawn from the same document) satisfy

$$\Pr(w_i, w_j) = \sum_{f=1}^F \Pr(w_i, w_j|f)\pi_f = \sum_{f=1}^F \mathbf{p}_f(i)\mathbf{p}_f(j)\pi_f,$$

since the words are independently drawn from the document.

Topic Mining

- Define the matrix of word co-occurrence probabilities $\mathbf{P}^{(2)}$ with elements $\mathbf{P}^{(2)}(i,j) := \Pr(w_i, w_j)$, and the matrix of conditional pmfs $\mathbf{C} := [\mathbf{p}_1, \cdots, \mathbf{p}_F]$. Then

$$\mathbf{P}^{(2)} = \mathbf{C}\mathrm{Diag}(\pi)\mathbf{C}^T.$$

- Next, consider "trigrams" – i.e., probabilities of triples of words being drawn from the same document

$$\Pr(w_i,w_j,w_k) = \sum_{f=1}^{F} \Pr(w_i,w_j,w_k|f)\pi_f = \sum_{f=1}^{F} \mathbf{p}_f(i)\mathbf{p}_f(j)\mathbf{p}_f(k)\pi_f.$$

- Define tensor $\mathbf{P}^{(3)}$ with elements $\mathbf{P}^{(3)}(i,j,k) := \Pr(w_i,w_j,w_k)$. Then $\mathbf{P}^{(3)}$ admits a symmetric non-negative CPD model of rank (at most) F:

$$\mathbf{P}^{(3)} = (\mathbf{C} \odot \mathbf{C} \odot \mathbf{C})\pi.$$

Topic Mining

- Similar to but in fact simpler from the case of Gaussian mixture parameter estimation, since here, due to *independent* sampling *with* replacement, the same expression holds even if two or three indices i, j, k are the same.
- We can estimate $\mathbf{C}$ and π from the tensor $\mathbf{P}^{(3)}$ and the matrix $\mathbf{P}^{(2)}$.
- In reality, we will use empirical word co-occurrence counts to estimate $\mathbf{P}^{(3)}$ and $\mathbf{P}^{(2)}$, and for this we need to sample enough triples ("enough samples").
- Once we have $\mathbf{C}$, we can classify any document by estimating (part of) its conditional word pmf and comparing it to the columns of $\mathbf{C}$.

Topic Mining

- Next, consider the more realistic situation where each document is a mixture of topics, modeled by a pmf $\mathbf{q}$ $(F \times 1)$ that is itself drawn from a distribution $\delta(\cdot)$ over the $(F-1)$-dimensional *probability simplex*.
- Our working experiment is now modified as follows:

1) For every document we sample, we draw $\mathbf{q} \sim \delta(\cdot)$;
2) For every word we sample from the given document, we first draw a topic t from $\mathbf{q}$ – i.e., topic f is selected with probability $\mathbf{q}(f)$;
3) Next, we draw a word $\sim \mathbf{p}_t$;
4) Goto 2, until you have sampled the desired number of words (e.g., three) from the given document;
5) Goto 1, until you have collected enough samples (e.g., enough triples of words).

Topic Mining

- Then,

$$\Pr(w_i, w_j | t_1, t_2, \mathbf{q}) = \mathbf{p}_{t_1}(i)\mathbf{p}_{t_2}(j) \Longrightarrow$$

$$\Pr(w_i, w_j | \mathbf{q}) = \sum_{t_1=1}^{F}\sum_{t_2=1}^{F} \mathbf{p}_{t_1}(i)\mathbf{p}_{t_2}(j)\mathbf{q}(t_1)\mathbf{q}(t_2) \Longrightarrow$$

$$\Pr(w_i, w_j) = \sum_{t_1=1}^{F}\sum_{t_2=1}^{F} \mathbf{p}_{t_1}(i)\mathbf{p}_{t_2}(j) E\left[\mathbf{q}(t_1)\mathbf{q}(t_2)\right],$$

  where we notice that what comes into play is the second-order statistics $E\left[\mathbf{q}(t_1)\mathbf{q}(t_2)\right]$ (the correlation) of $\delta(\cdot)$.

- The $I \times I$ matrix $\mathbf{Q}$ with elements $\mathbf{Q}(i,j) := \Pr(w_i, w_j)$ admits the decomposition $\mathbf{Q} = \mathbf{C}\mathbf{E}\mathbf{C}^T$, where $\mathbf{E}(t_1, t_2) := E\left[\mathbf{q}(t_1)\mathbf{q}(t_2)\right]$.

Topic Mining

- Likewise, it follows that, for the trigrams

$$\Pr(w_i, w_j, w_k) = \sum_{t_1=1}^{F} \sum_{t_2=1}^{F} \sum_{t_3=1}^{F} \mathbf{p}_{t_1}(i)\mathbf{p}_{t_2}(j)\mathbf{p}_{t_3}(k) E\left[\mathbf{q}(t_1)\mathbf{q}(t_2)\mathbf{q}(t_3)\right],$$

  which involves the third-order statistics tensor $\mathbf{G}$ of $\delta(\cdot)$ with elements $\mathbf{G}(t_1, t_2, t_3) := E\left[\mathbf{q}(t_1)\mathbf{q}(t_2)\mathbf{q}(t_3)\right]$.
- Defining the $I \times I \times I$ tensor $\mathbf{P}$ with elements $\mathbf{P}(i, j, k) := \Pr(w_i, w_j, w_k)$, it follows that $\mathbf{P}$ admits a symmetric Tucker decomposition:

$$\mathbf{P} = \text{Tucker}(\mathbf{C}, \mathbf{C}, \mathbf{C}, \mathbf{G}), \text{ with } \mathbf{C} = [\mathbf{p}_1, \cdots, \mathbf{p}_F].$$

Topic Mining

$$\mathbf{P} = \text{Tucker}(\mathbf{C}, \mathbf{C}, \mathbf{C}, \mathbf{G}), \text{ with } \mathbf{C} = [\mathbf{p}_1, \cdots, \mathbf{p}_F].$$

- Note that $\mathbf{C}$ is element-wise non-negative, but in principle $\mathbf{G}$ may have negative elements.
- As we know, Tucker models are not identifiable in general – there is linear transformation freedom.
- This can be alleviated when one can assume sparsity in $\mathbf{C}$ [Anandkumar, 2013], $\mathbf{G}$, or both (intuitively, this is because linear transformations generally do not preserve sparsity).

Topic Mining

- Recall: pairwise co-occurrence probability matrix $\mathbf{Q}$ with elements $\mathbf{Q}(i,j) := \Pr(w_i, w_j)$ admits the decomposition $\mathbf{Q} = \mathbf{C}\mathbf{E}\mathbf{C}^T$, where $\mathbf{E}(t_1, t_2) := E\left[\mathbf{q}(t_1)\mathbf{q}(t_2)\right]$ is the topic correlation matrix.
- Interestingly, $\mathbf{Q} = \mathbf{C}\mathbf{E}\mathbf{C}^T$ is already identifiable under mild conditions *if one uses the correct* (VolMin) criterion:
- K. Huang*, X. Fu* and N.D. Sidiropoulos, "**Anchor-Free Correlated Topic Modeling: Identifiability and Algorithm**", NIPS 2016. (*equal contribution)
- Implies slab-by-slab identifiability of $\mathbf{P} = \text{Tucker}(\mathbf{C}, \mathbf{C}, \mathbf{C}, \mathbf{G})$. More work in this direction currently underway.

Discriminative Subspace Learning

- Given $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_M]$ ($N \times M$) and associated class labels $\mathbf{z} = [z_1, \cdots, z_M]$ ($1 \times M$) for the columns of $\mathbf{X}$.
- Find a dimensionality-reducing linear transformation $\mathbf{U}$ of size $N \times F$, $F < N$ (usually $F \ll N$) such that

$$\min_{\mathbf{U}|\mathbf{U}^T\mathbf{U}=\mathbf{I}} \sum_{m=1}^{M} \left\{ (1-\lambda) \sum_{\ell=1|z_\ell=z_m}^{M} ||\mathbf{U}^T\mathbf{x}_m - \mathbf{U}^T\mathbf{x}_\ell||_2^2 - \right.$$

$$\left. \lambda \sum_{\ell=1|z_\ell \neq z_m}^{M} ||\mathbf{U}^T\mathbf{x}_m - \mathbf{U}^T\mathbf{x}_\ell||_2^2 \right\},$$

where the first (second) term measures the within-class (across-class) distance in reduced dimension space.

Discriminative Subspace Learning

- Interpretation: find a dimensionality-reducing transformation that will map points close to each other in terms of Euclidean distance if they have the same class label, far otherwise.
- Find a subspace to project onto where we can easily visualize (if $F = 2$ or 3) the point clouds of the different classes.
- Upon defining

$$w_{m,\ell} := (1-\lambda)^{1(z_\ell = z_m)}(-\lambda)^{1-1(z_\ell = z_m)},$$

where $1(z_\ell = z_m) = 1$ if $z_\ell = z_m$, 0 otherwise, we can compactly write the problem as follows

$$\min_{\mathbf{U}|\mathbf{U}^T\mathbf{U}=\mathbf{I}} \sum_{m=1}^{M}\sum_{\ell=1}^{M} ||\mathbf{U}^T\mathbf{x}_m - \mathbf{U}^T\mathbf{x}_\ell||_2^2 w_{m,\ell}.$$

Discriminative Subspace Learning

- Expanding the squared norm and using properties of $\mathrm{Tr}(\cdot)$, we can write the cost function as

$$\sum_{m=1}^{M}\sum_{\ell=1}^{M}||\mathbf{U}^T\mathbf{x}_m - \mathbf{U}^T\mathbf{x}_\ell||_2^2 w_{m,\ell} = \mathrm{Tr}(\mathbf{U}\mathbf{U}^T\mathbf{Y}),$$

where

$$\mathbf{Y} := \sum_{m=1}^{M}\sum_{\ell=1}^{M} w_{m,\ell}(\mathbf{x}_m - \mathbf{x}_\ell)(\mathbf{x}_m - \mathbf{x}_\ell)^T.$$

Notice that $w_{m,\ell} = w_{\ell,m}$ by definition, and $\mathbf{Y}$ is symmetric. Let $\mathbf{Y} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ be the eigendecomposition of $\mathbf{Y}$, and note that $\mathrm{Tr}(\mathbf{U}\mathbf{U}^T\mathbf{Y}) = \mathrm{Tr}(\mathbf{U}^T\mathbf{Y}\mathbf{U})$. Clearly, $\mathbf{U}_{\text{opt}} = F$ minor eigenvectors of $\mathbf{Y}$ (columns of $\mathbf{V}$ corresponding to the F smallest elements on the diagonal of $\mathbf{\Lambda}$).

Multilinear Subspace Learning

- Now, suppose that the columns in $\mathbf{X}$ are in fact vectorized tensors. As an example, suppose that there exist *common* bases $\mathbf{U}$ $(I \times r_1)$, $\mathbf{V}$ $(J \times r_2)$, $\mathbf{W}$ $(K \times r_3)$, such that

$$\mathbf{x}_m \approx (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})\mathbf{g}_m, \quad \forall m \in \{1, \cdots, M\},$$

  i.e., each $\mathbf{x}_m$ can be modeled using a $\perp$-Tucker model with common mode bases, but different cores for different m.
- Think of $(\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})^T$ $(r_1 r_2 r_3 \times IJK)$ as a (Kronecker) structured dimensionality reducing transformation;
- Think of the vectorized core array $\mathbf{g}_m$ as the low-dimensional $(r_1 r_2 r_3 \times 1)$ representation of $\mathbf{x}_m$.

Multilinear Subspace Learning

- Want to find $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ such that the $\mathbf{g}$'s corresponding to $\mathbf{x}$'s in the same (different) class are close (far) from each other.
- Following the same development as before, using

$$\widehat{\mathbf{g}}_m = (\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})^T \mathbf{x}_m$$

as the projection of $\mathbf{x}_m$ in reduced-dimension space, we arrive at

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \text{Tr}\left((\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})(\mathbf{U} \otimes \mathbf{V} \otimes \mathbf{W})^T \mathbf{Y}\right),$$

$$\text{subject to: } \mathbf{U}^T\mathbf{U} = \mathbf{I},\ \mathbf{V}^T\mathbf{V} = \mathbf{I},\ \mathbf{W}^T\mathbf{W} = \mathbf{I},$$

Multilinear Subspace Learning

- Equivalently,

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W}} \text{Tr}\left(((\mathbf{U}\mathbf{U}^T)\otimes(\mathbf{V}\mathbf{V}^T)\otimes(\mathbf{W}\mathbf{W}^T))\mathbf{Y}\right),$$

$$\text{subject to: } \mathbf{U}^T\mathbf{U}=\mathbf{I},\ \mathbf{V}^T\mathbf{V}=\mathbf{I},\ \mathbf{W}^T\mathbf{W}=\mathbf{I},$$

- It is now clear that, conditioned on, say, $\mathbf{U}$ and $\mathbf{V}$, the update with respect to $\mathbf{W}$ boils down to

$$\min_{\mathbf{W}|\mathbf{W}^T\mathbf{W}=\mathbf{I}} \text{Tr}\left(\mathbf{W}\mathbf{W}^T\mathbf{Z}\right),$$

for some matrix $\mathbf{Z}$ that depends on the values of $\mathbf{U}$ and $\mathbf{V}$.

References I


James Alexander and André Hirschowitz.
Polynomial interpolation in several variables.
Journal of Algebraic Geometry, 4(2):201–222, 1995.


B. W. Bader and T. G. Kolda.
Efficient MATLAB computations with sparse and factored tensors.
SIAM Journal on Scientific Computing, 30(1):205–231, December 2007.


R. Bro and N.D. Sidiropoulos.
Least squares regression under unimodality and non-negativity constraints.
Journal of Chemometrics, 12:223–247, 1998.


A. Beutel, P. Talukdar, A. Kumar, C. Faloutsos, E. Papalexakis, and E. Xing.
FlexiFaCT: Scalable Flexible Factorization of Coupled Tensors on Hadoop, chapter 13, pages 109–117.


P. Comon.
Tensors : A brief introduction.
Signal Processing Magazine, IEEE, 31(3):44–53, May 2014.


J. H. Choi and S. V. N. Vishwanathan.
DFacTo: Distributed factorization of tensors.
In *Advances in Neural Information Processing Systems*, pages 1296–1304, 2014.

References II


J. Håstad.
Tensor rank is np-complete.
J. Algorithms, 11(4):644–654, December 1990.


M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer.
Best low multilinear rank approximation of higher-order tensors, based on the riemannian trust-region scheme.
SIAM Journal on Matrix Analysis and Applications, 32(1):115–135, 2011.


T.G. Kolda and B.W. Bader.
Tensor decompositions and applications.
SIAM REVIEW, 51(3):455–500, 2009.


U Kang, E. Papalexakis, A. Harpale, and C. Faloutsos.
Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries.
In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 316–324. ACM, 2012.


P.M. Kroonenberg.
Applied multiway data analysis.
Wiley, 2008.

References III


J. Landsberg.
Tensors: geometry and applications, volume 128.
American Mathematical Soc., 2012.


J. Nocedal and S. Wright.
Numerical optimization.
Springer Science & Business Media, 2006.


C. Papalexakis, E.and Faloutsos and N. Sidiropoulos.
Parcube: Sparse parallelizable tensor decompositions.
In *Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part I*, ECML PKDD'12, pages 521–536, Berlin, Heidelberg, 2012. Springer-Verlag.


A. H. Phan, P. Tichavsky, and A. Cichocki.
Fast Alternating LS Algorithms for High Order CANDECOMP/PARAFAC Tensor Factorizations.
IEEE Transactions on Signal Processing, 61(19):4834–4846, October 2013.
doi: 10.1109/TSP.2013.2269903.


M. Rajih, P. Comon, and R. Harshman.
Enhanced line search: A novel method to accelerate parafac.
SIAM Journal on Matrix Analysis and Applications, 30(3):1128–1147, 2008.

References IV


N. Ravindran, N.D. Sidiropoulos, S. Smith, and G. Karypis.
Memory-efficient parallel computation of tensor and matrix products for big tensor decomposition.
In *Asilomar Conference on Signals, Systems, and Computers*, pages 581–585, 2014.


M. Sorensen, L. De Lathauwer, P. Comon, S. Icart, and L. Deneire.
Canonical Polyadic Decomposition with a Columnwise Orthonormal Factor Matrix.
SIAM Journal on Matrix Analysis and Applications, 33(4, Oct.-Dec.):1190–1213, 2012.


B. Savas and L.-H. Lim.
Quasi-newton methods on grassmannians and multilinear approximations of tensors.
SIAM Journal on Scientific Computing, 32(6):3352–3393, 2010.


S. Smith, N. Ravindran, N. D. Sidiropoulos, and G. Karypis.
SPLATT: Efficient and parallel sparse tensor-matrix multiplication.
In *IEEE International Parallel and Distributed Processing Symposium*, 2015.


L. Sorber, M. Van Barel, and L. De Lathauwer.
Optimization-based algorithms for tensor decompositions: canonical polyadic decomposition, decomposition in rank-(Lr,Lr,1) terms and a new generalization.
SIAM Journal on Optimization, 23(2):695–720, April 2013.

References V


G. Tomasi and R. Bro.
PARAFAC and missing values.
Chemometrics and Intelligent Laboratory Systems, 75(2):163–180, 2005.


G. Tomasi and R. Bro.
A comparison of algorithms for fitting the PARAFAC model.
Computational Statistics and Data Analysis, 50(7):1700–1734, 2006.


N. Vervliet and L. De Lathauwer.
A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors.
IEEE Journal of Selected Topics in Signal Processing, 10(2):284–295.


N. Vervliet, O. Debals, L. Sorber, and L De Lathauwer.
Breaking the Curse of Dimensionality Using Decompositions of Incomplete Tensors: Tensor-based scientific computing in big data analysis.
Signal Processing Magazine, IEEE, 31(5):71–79, September 2014.


N. Vannieuwenhoven, K. Meerbergen, and R Vandebril.
Computing the Gradient in Optimization Algorithms for the CP Decomposition in Constant Memory through Tensor Blocking.
SIAM Journal on Scientific Computing, 37(3):C415–C438, 2015.