Tensor Decomposition
Theory and Algorithms in the Era of Big Data

Nikos Sidiropoulos, UMN

EUSIPCO Inaugural Lecture, Sep. 2, 2014, Lisbon

Web, papers, software, credits

- Nikos Sidiropoulos, UMN `http://www.ece.umn.edu/~nikos/`
  - Signal and Tensor Analytics Research (STAR) group `https://sites.google.com/a/umn.edu/nikosgroup/home`
- Collaborators
  - Rasmus Bro, U. Copenhagen, `http://www.models.kvl.dk/users/rasmus/`
  - Vangelis Papalexakis, CMU `http://www.cs.cmu.edu/~epapalex/`
  - Christos Faloutsos, CMU `http://www.cs.cmu.edu/~christos/`
  - Athanasios Liavas, TUC `http://www.telecom.tuc.gr/~liavas/`
  - George Karypis, UMN `http://glaros.dtc.umn.edu/gkhome/index.php`
  - Anastasios Kyrillidis, EPFL `http://lions.epfl.ch/phdstudents/tasos`
- Sponsor
  - NSF-NIH/BIGDATA: Big Tensor Mining: Theory, Scalable Algorithms and Applications, Nikos Sidiropoulos, George Karypis (UMN), Christos Faloutsos, Tom Mitchell (CMU), NSF IIS-1247489/1247632.

What do these have in common?

- Machine learning - e.g., clustering and co-clustering, social network analysis
- Speech - separating unknown mixtures of speech signals in reverberant environments
- Audio - untangling audio sources in the spectrogram domain
- Communications, signal intelligence - unraveling CDMA mixtures, breaking codes
- Passive localization + radar (angles, range, Doppler, profiles)
- Chemometrics: Chemical signal separation, e.g., fluorescence, 'mathematical chromatography' (90's -)
- Psychometrics: Analysis of individual differences, preferences (70's -)

Matrices, rank decomposition

- A matrix (or *two-way* array) is a dataset $\mathbf{X}$ indexed by *two indices*, (i,j)-th entry $\mathbf{X}(i,j)$.
- Simple matrix $\mathbf{S}(i,j) = \mathbf{a}(i)\mathbf{b}(j)$, $\forall i,j$; separable, every row (column) proportional to every other row (column). Can write as $\mathbf{S} = \mathbf{a}\mathbf{b}^T$.
- rank($\mathbf{X}$) := smallest number of 'simple' (separable, rank-one) matrices needed to generate $\mathbf{X}$ - a measure of complexity.

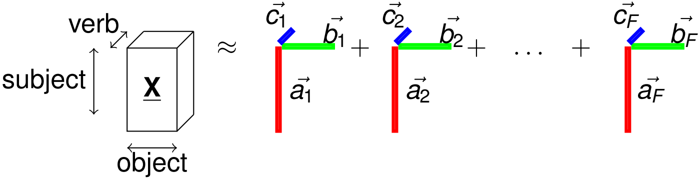
$$\mathbf{X}(i,j) = \sum_{f=1}^{F} \mathbf{a}_f(i)\mathbf{b}_f(j); \text{ or } \mathbf{X} = \sum_{f=1}^{F} \mathbf{a}_f\mathbf{b}_f^T = \mathbf{A}\mathbf{B}^T.$$

- Turns out rank($\mathbf{X}$) = maximum number of linearly independent rows (or, columns) in $\mathbf{X}$.
- Rank decomposition for matrices is not unique (except for matrices of rank = 1), as $\forall$ invertible $\mathbf{M}$:

$$\mathbf{X} = \mathbf{A}\mathbf{B}^T = (\mathbf{A}\mathbf{M})\left(\mathbf{M}^{-T}\mathbf{B}^T\right) = (\mathbf{A}\mathbf{M})\left(\mathbf{B}\mathbf{M}^{-1}\right)^T = \tilde{\mathbf{A}}\tilde{\mathbf{B}}^T.$$

Tensor? What is this?

- CS 'slang' for *three-way* array: dataset $\underline{\mathbf{X}}$ indexed by *three indices*, (i, j, k)-th entry $\underline{\mathbf{X}}(i, j, k)$.
- In plain words: a 'shoebox'!
- For two vectors $\mathbf{a}$ $(I \times 1)$ and $\mathbf{b}$ $(J \times 1)$, $\mathbf{a} \circ \mathbf{b}$ is an $I \times J$ rank-one matrix with (i, j)-th element $\mathbf{a}(i)\mathbf{b}(j)$; i.e., $\mathbf{a} \circ \mathbf{b} = \mathbf{a}\mathbf{b}^T$.
- For three vectors, $\mathbf{a}$ $(I \times 1)$, $\mathbf{b}$ $(J \times 1)$, $\mathbf{c}$ $(K \times 1)$, $\mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$ is an $I \times J \times K$ rank-one three-way array with (i, j, k)-th element $\mathbf{a}(i)\mathbf{b}(j)\mathbf{c}(k)$.
- The *rank of a three-way array* $\underline{\mathbf{X}}$ is the smallest number of outer products needed to synthesize $\underline{\mathbf{X}}$.
- Example: NELL / Tom Mitchell @ CMU



Rank decomposition for tensors

- Tensor:

$$\underline{\mathbf{X}} = \sum_{f=1}^{F} \mathbf{a}_f \circ \mathbf{b}_f \circ \mathbf{c}_f$$

- Scalar:

$$\underline{\mathbf{X}}(i,j,k) = \sum_{f=1}^{F} a_{i,f} b_{j,f} c_{k,f}, \quad \begin{array}{l} \forall i \in \{1, \cdots, I\} \\ \forall j \in \{1, \cdots, J\} \\ \forall k \in \{1, \cdots, K\} \end{array}$$

- Slabs:

$$\mathbf{X}_k = \mathbf{A}\mathbf{D}_k(\mathbf{C})\mathbf{B}^T, \ k = 1, \cdots, K$$

- Matrix:

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

- Tall vector:

$$\mathbf{x}^{(KJI)} := vec\left(\mathbf{X}^{(KJ \times I)}\right) = (\mathbf{A} \odot (\mathbf{B} \odot \mathbf{C}))\,\mathbf{1}_{F \times 1} = (\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\,\mathbf{1}_{F \times 1}$$

Tensors vs. Matrices

- Matrix rank always $\leq \min(I, J)$.
- $\text{rank}(\text{randn}(I, J)) = \min(I, J)$ w.p. 1.
- SVD is rank-revealing.
- SVD provides best rank-R approximation.

Whereas ...

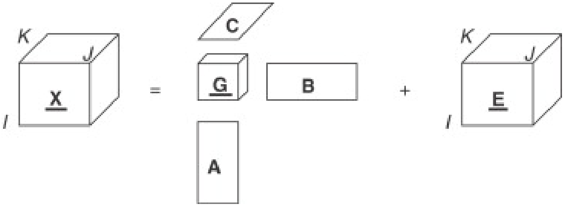
- Tensor rank can be $> \max(I, J, K)$; $\leq \min(IJ, JK, IK)$ always.
- $\text{rank}(\text{randn}(2, 2, 2)) \in \{2, 3\}$, both with positive probability.
- Finding tensor rank is NP-hard.
- Computing best rank-1 approximation to a tensor is NP-hard.
- Best rank-R approximation may not even exist.
- True for n-way arrays of any order $n \geq 3$ - matrices are the only exception!

Don't be turned off - there are many good things about tensors!

Tensor Singular Value Decomposition?

- For matrices, SVD is instrumental: rank-revealing, Eckart-Young
- So is there a tensor equivalent to the matrix SVD?
- Yes, ... and no! In fact there is no single tensor SVD.
- Two basic decompositions:
  - CANonical DECOMPosition (CANDECOMP), also known as PARAllel FACtor (PARAFAC) analysis, or CANDECOMP-PARAFAC (CP) for short: non-orthogonal, unique under certain conditions.
  - Tucker3, orthogonal without loss of generality, non-unique except for very special cases.
- Both are outer product decompositions, but with very different structural properties.
- Rule of thumb: use Tucker3 for subspace estimation and tensor approximation, e.g., compression applications; use PARAFAC for latent parameter estimation - recovering the 'hidden' rank-one factors.

Tucker3



- $I \times J \times K$ three-way array $\underline{\mathbf{X}}$
- $\mathbf{A}$: $I \times L$, $\mathbf{B}$: $J \times M$, $\mathbf{C}$: $K \times N$ *mode loading matrices*
- $\underline{\mathbf{G}}$: $L \times M \times N$ Tucker3 *core*

Tucker3, continued

- Consider an $I \times J \times K$ three-way array $\underline{\mathbf{X}}$ comprising K matrix slabs $\{\mathbf{X}_k\}_{k=1}^{K}$, arranged into matrix $\mathbf{X} := [\text{vec}(\mathbf{X}_1), \cdots, \text{vec}(\mathbf{X}_K)]$.
- The Tucker3 model can be written as

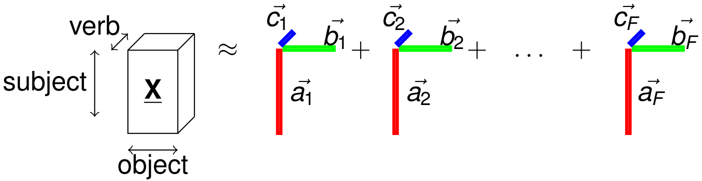
$$\mathbf{X} \approx (\mathbf{B} \otimes \mathbf{A})\mathbf{G}\mathbf{C}^T,$$

  where $\mathbf{G}$ is the Tucker3 core tensor $\underline{\mathbf{G}}$ recast in matrix form. The non-zero elements of the core tensor determine the interactions between columns of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$.
- The associated model-fitting problem is

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{G}} ||\mathbf{X} - (\mathbf{B} \otimes \mathbf{A})\mathbf{G}\mathbf{C}^T||_F^2,$$

  which is usually solved using an alternating least squares procedure.
- $\text{vec}(\mathbf{X}) \approx (\mathbf{C} \otimes \mathbf{B} \otimes \mathbf{A})\,\text{vec}(\mathbf{G})$.
- Highly non-unique - e.g., rotate $\mathbf{C}$, counter-rotate $\mathbf{G}$ using unitary matrix.
- Subspaces can be recovered; Tucker3 is good for *tensor approximation*, not latent parameter estimation.

PARAFAC



- Low-rank tensor decomposition / approximation

$$\underline{\mathbf{X}} \approx \sum_{f=1}^{F} \mathbf{a}_f \circ \mathbf{b}_f \circ \mathbf{c}_f,$$

- PARAFAC [Harshman '70-'72], CANDECOMP [Carroll & Chang, '70], now CP; also cf. [Hitchcock, '27]
- Combining slabs and using Khatri-Rao product,

$$\mathbf{X} \approx (\mathbf{B} \odot \mathbf{A})\mathbf{C}^T \Longleftrightarrow \text{vec}(\mathbf{X}) \approx (\mathbf{C} \odot \mathbf{B} \odot \mathbf{A})\,\mathbf{1}$$

Uniqueness

- Under certain conditions, PARAFAC is *essentially unique*, i.e., $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ can be identified from $\mathbf{X}$ up to permutation and scaling of columns - there's no rotational freedom; cf. [Kruskal '77, Sidiropoulos *et al* '00 - '07, de Lathauwer '04-, Stegeman '06-, Chiantini, Ottaviani '11-, ...]
- $I \times J \times K$ tensor $\underline{\mathbf{X}}$ of rank F, vectorized as $IJK \times 1$ vector $\mathbf{x} = (\mathbf{A} \odot \mathbf{B} \odot \mathbf{C})\,\mathbf{1}$, for some $\mathbf{A}$ $(I \times F)$, $\mathbf{B}$ $(J \times F)$, and $\mathbf{C}$ $(K \times F)$ - a PARAFAC model of size $I \times J \times K$ and order F parameterized by $(\mathbf{A}, \mathbf{B}, \mathbf{C})$.
- The *Kruskal-rank* of $\mathbf{A}$, denoted $k_{\mathbf{A}}$, is the maximum k such that *any* k columns of $\mathbf{A}$ are linearly independent ($k_{\mathbf{A}} \leq r_{\mathbf{A}} := \text{rank}(\mathbf{A})$). $\text{spark}(\mathbf{A}) = k_{\mathbf{A}} + 1$
- Given $\underline{\mathbf{X}}$ ($\Leftrightarrow \mathbf{x}$), if $k_{\mathbf{A}} + k_{\mathbf{B}} + k_{\mathbf{C}} \geq 2F + 2$, then $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ are unique up to a common column permutation and scaling/counter-scaling (e.g., multiply first column of $\mathbf{A}$ by 5, divide first column of $\mathbf{B}$ by 5, outer product stays the same) - cf. [Kruskal, 1977]
- N-way case: $\sum_{n=1}^{N} k_{\mathbf{A}^{(n)}} \geq 2F + (N-1)$ [Sidiropoulos & Bro, 2000]

Alternating Least Squares (ALS)

- Based on matrix view:

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

- Multilinear LS problem:

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

- NP-hard - even for a single component, i.e., vector $\mathbf{a}, \mathbf{b}, \mathbf{c}$. See [Hillar and Lim, "Most tensor problems are NP-hard," 2013]
- But ... given interim estimates of **B**, **C**, can easily solve for conditional LS update of **A**:

$$\mathbf{A}_{CLS} = \left((\mathbf{B} \odot \mathbf{C})^{\dagger} \mathbf{X}^{(KJ \times I)} \right)^T$$

- Similarly for the CLS updates of **B**, **C** (symmetry); alternate until cost function converges (monotonically).

Other algorithms?

- Many! - first-order (gradient-based), second-order (Hessian-based) Gauss-Newton, line search, Levenberg-Marquardt, weighted least squares, majorization
- Algebraic initialization (matters)
- See Tomasi and Bro, 2006, for a good overview
- Second-order advantage when close to optimum, but can (and do) diverge
- First-order often prone to local minima, slow to converge
- Stochastic gradient descent (CS community) - simple, parallel, but very slow
- Difficult to incorporate additional constraints like sparsity, non-negativity, unimodality, etc.

ALS

- No parameters to tune!
- Easy to program, uses standard linear LS
- Monotone convergence of cost function
- Does not require any conditions beyond model identifiability
- Easy to incorporate additional constraints, due to multilinearity, e.g., replace linear LS with linear NNLS for NN
- Even non-convex (e.g., FA) constraints can be handled with column-wise updates (optimal scaling lemma)
- Cons: sequential algorithm, convergence can be slow
- Still workhorse after all these years

Outliers, sparse residuals

- Instead of LS,

$$\min ||\mathbf{X}^{(KJ \times I)} - (\mathbf{B} \odot \mathbf{C})\mathbf{A}^T||_1$$

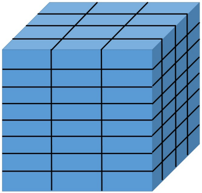
- Conditional update: LP
- Almost as good: coordinate-wise, using *weighted median filtering* (very cheap!) [Vorobyov, Rong, Sidiropoulos, Gershman, 2005]
- PARAFAC CRLB: [Liu & Sidiropoulos, 2001] (Gaussian); [Vorobyov, Rong, Sidiropoulos, Gershman, 2005] (Laplacian, etc differ only in pdf-dependent scale factor).
- Alternating optimization algorithms approach the CRLB when the problem is well-determined (meaning: not barely identifiable).

Big (tensor) data

- Tensors can easily become really big! - size exponential in the number of dimensions (‘ways’, or ‘modes’).
- Datasets with millions of items per mode - e.g., NELL, social networks, marketing, Google.
- Cannot load in main memory; may reside in cloud storage.
- Sometimes *very* sparse - can store and process as (i,j,k,value) list, nonzero column indices for each row, runlength coding, etc.
- (Sparse) Tensor Toolbox for Matlab [Kolda *et al*].
- Avoids explicitly computing dense intermediate results.

Tensor partitioning?

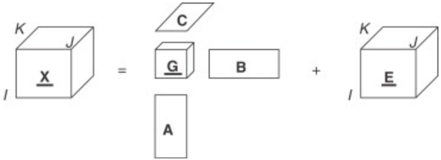
- Parallel algorithms for matrix algebra use data partitioning
- Can we reuse some of these ideas?



- Low-hanging fruit?
- First considered in [Phan, Cichocki, Neurocomputing, 2011] - identifiability issues, matching permutations and scalings?
- Later revisited in [Almeida, Kibangou, CAMSAP 2013, ICASSP 2014] - no loss of optimality (as if working w/ full data), but inter-process communication overhead, additional identifiability conditions.

Tensor compression

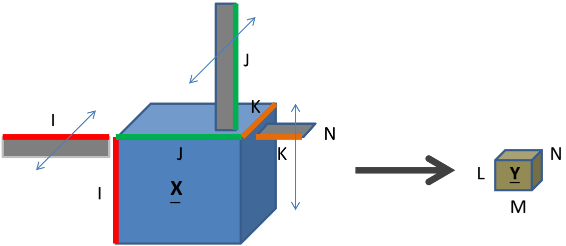
- Commonly used compression method for 'moderate'-size tensors: fit orthogonal Tucker3 model, regress data onto fitted mode-bases.



- Implemented in n-way toolbox (Rasmus Bro) `http://www.mathworks.com/matlabcentral/fileexchange/1088-the-n-way-toolbox`
- Lossless if exact mode bases used [CANDELINC]; but Tucker3 fitting is itself cumbersome for big tensors (big matrix SVDs), cannot compress below mode ranks without introducing errors

Tensor compression

- Consider compressing $\mathbf{x} = \text{vec}(\underline{\mathbf{X}})$ into $\mathbf{y} = \mathbf{S}\mathbf{x}$, where $\mathbf{S}$ is $d \times IJK$, $d \ll IJK$.
- In particular, consider a specially structured compression matrix $\mathbf{S} = \mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T$
- Corresponds to multiplying (every slab of) $\underline{\mathbf{X}}$ from the I-mode with $\mathbf{U}^T$, from the J-mode with $\mathbf{V}^T$, and from the K-mode with $\mathbf{W}^T$, where $\mathbf{U}$ is $I \times L$, $\mathbf{V}$ is $J \times M$, and $\mathbf{W}$ is $K \times N$, with $L \leq I$, $M \leq J$, $N \leq K$ and $LMN \ll IJK$



Key

- Due to a property of the Kronecker product

$$\left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)(\mathbf{A} \odot \mathbf{B} \odot \mathbf{C}) =$$

$$\left((\mathbf{U}^T\mathbf{A}) \odot (\mathbf{V}^T\mathbf{B}) \odot (\mathbf{W}^T\mathbf{C})\right),$$

from which it follows that

$$\mathbf{y} = \left((\mathbf{U}^T\mathbf{A}) \odot (\mathbf{V}^T\mathbf{B}) \odot (\mathbf{W}^T\mathbf{C})\right)\mathbf{1} = \left(\tilde{\mathbf{A}} \odot \tilde{\mathbf{B}} \odot \tilde{\mathbf{C}}\right)\mathbf{1}.$$

i.e., the compressed data follow a PARAFAC model of size $L \times M \times N$ and order F parameterized by $(\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{C}})$, with $\tilde{\mathbf{A}} := \mathbf{U}^T\mathbf{A}$, $\tilde{\mathbf{B}} := \mathbf{V}^T\mathbf{B}$, $\tilde{\mathbf{C}} := \mathbf{W}^T\mathbf{C}$.

Random multi-way compression can be better!

- Sidiropoulos & Kyrillidis, IEEE SPL Oct. 2012
- Assume that the columns of $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are sparse, and let n_a (n_b, n_c) be an upper bound on the number of nonzero elements per column of $\mathbf{A}$ (respectively $\mathbf{B}$, $\mathbf{C}$).
- Let the mode-compression matrices $\mathbf{U}$ ($I \times L, L \leq I$), $\mathbf{V}$ ($J \times M, M \leq J$), and $\mathbf{W}$ ($K \times N, N \leq K$) be randomly drawn from an absolutely continuous distribution with respect to the Lebesgue measure in $\mathbb{R}^{IL}$, $\mathbb{R}^{JM}$, and $\mathbb{R}^{KN}$, respectively.
- If

$$\min(L, k_{\mathbf{A}}) + \min(M, k_{\mathbf{B}}) + \min(N, k_{\mathbf{C}}) \geq 2F + 2, \quad \text{and}$$

$$L \geq 2n_a, \quad M \geq 2n_b, \quad N \geq 2n_c,$$

  then the original factor loadings $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are almost surely identifiable from the compressed data.
- Never have to see big data; significant computational complexity reduction as well.

Further compression - down to $O(\sqrt{F})$ in 2/3 modes

- Sidiropoulos & Kyrillidis, IEEE SPL Oct. 2012
- Assume that the columns of $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are sparse, and let n_a (n_b, n_c) be an upper bound on the number of nonzero elements per column of $\mathbf{A}$ (respectively $\mathbf{B}$, $\mathbf{C}$).
- Let the mode-compression matrices $\mathbf{U}$ ($I \times L, L \leq I$), $\mathbf{V}$ ($J \times M, M \leq J$), and $\mathbf{W}$ ($K \times N, N \leq K$) be randomly drawn from an absolutely continuous distribution with respect to the Lebesgue measure in $\mathbb{R}^{IL}$, $\mathbb{R}^{JM}$, and $\mathbb{R}^{KN}$, respectively.
- If

$$r_{\mathbf{A}} = r_{\mathbf{B}} = r_{\mathbf{C}} = F$$

$$L(L-1)M(M-1) \geq 2F(F-1), \; N \geq F, \quad \text{and}$$

$$L \geq 2n_a, \quad M \geq 2n_b, \quad N \geq 2n_c,$$

then the original factor loadings $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are almost surely identifiable from the compressed data up to a common column permutation and scaling.

Latest on PARAFAC uniqueness

- Luca Chiantini and Giorgio Ottaviani, On Generic Identifiability of 3-Tensors of Small Rank, SIAM. J. Matrix Anal. & Appl., 33(3), 1018–1037:
- Consider an $I \times J \times K$ tensor $\underline{\mathbf{X}}$ of rank F, and order the dimensions so that $I \leq J \leq K$
- Let i be maximal such that $2^i \leq I$, and likewise j maximal such that $2^j \leq J$
- If $F \leq 2^{i+j-2}$, then $\underline{\mathbf{X}}$ has a unique decomposition almost surely
- For I, J powers of 2, the condition simplifies to $F \leq \frac{IJ}{4}$
- More generally, condition implies:
  - if $F \leq \frac{(I+1)(J+1)}{16}$, then $\underline{\mathbf{X}}$ has a unique decomposition almost surely

Even further compression

- Assume that the columns of $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are sparse, and let n_a (n_b, n_c) be an upper bound on the number of nonzero elements per column of $\mathbf{A}$ (respectively $\mathbf{B}$, $\mathbf{C}$).
- Let the mode-compression matrices $\mathbf{U}$ ($I \times L, L \le I$), $\mathbf{V}$ ($J \times M, M \le J$), and $\mathbf{W}$ ($K \times N, N \le K$) be randomly drawn from an absolutely continuous distribution with respect to the Lebesgue measure in $\mathbb{R}^{IL}$, $\mathbb{R}^{JM}$, and $\mathbb{R}^{KN}$, respectively.
- Assume $L \le M \le N$, and L, M are powers of 2, for simplicity
- If

$$r_{\mathbf{A}} = r_{\mathbf{B}} = r_{\mathbf{C}} = F$$

$$LM \ge 4F, \; N \ge M \ge L, \quad \text{and}$$

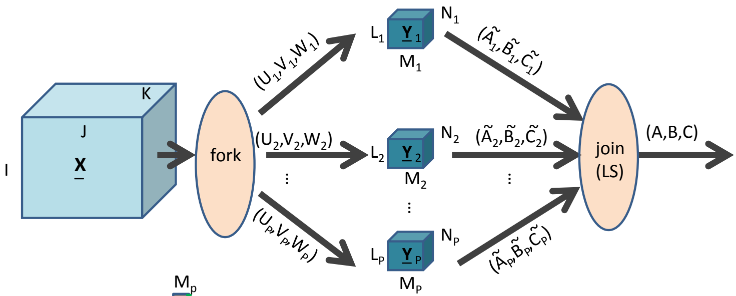
$$L \ge 2n_a, \quad M \ge 2n_b, \quad N \ge 2n_c,$$

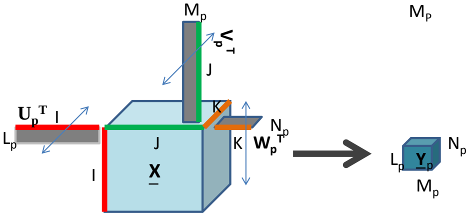
  then the original factor loadings $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are almost surely identifiable from the compressed data up to a common column permutation and scaling.
- Allows compression down to order of $\sqrt{F}$ in all three modes

What if $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are not sparse?

- If $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are sparse with respect to *known* bases, i.e., $\mathbf{A} = \mathbf{R}\check{\mathbf{A}}$, $\mathbf{B} = \mathbf{S}\check{\mathbf{B}}$, and $\mathbf{C} = \mathbf{T}\check{\mathbf{C}}$, with $\mathbf{R}, \mathbf{S}, \mathbf{T}$ the respective sparsifying bases, and $\check{\mathbf{A}}, \check{\mathbf{B}}, \check{\mathbf{C}}$ sparse
- Then the previous results carry over under appropriate conditions, e.g., when $\mathbf{R}, \mathbf{S}, \mathbf{T}$ are non-singular.
- OK, but what if such bases cannot be found?

PARACOMP: PArallel RAndomly COMPressed Cubes





PARACOMP

- Assume $\tilde{\mathbf{A}}_p, \tilde{\mathbf{B}}_p, \tilde{\mathbf{C}}_p$ identifiable from $\underline{\mathbf{Y}}_p$ (up to perm & scaling of cols)
- Upon factoring $\underline{\mathbf{Y}}_p$ into F rank-one components, we obtain

$$\tilde{\mathbf{A}}_p = \mathbf{U}_p^T \mathbf{A} \mathbf{\Pi}_p \mathbf{\Lambda}_p. \tag{1}$$

- Assume first 2 columns of each $\mathbf{U}_p$ are common, let $\bar{\mathbf{U}}$ denote this common part, and $\bar{\mathbf{A}}_p :=$ first two rows of $\tilde{\mathbf{A}}_p$. Then

$$\bar{\mathbf{A}}_p = \bar{\mathbf{U}}^T \mathbf{A} \mathbf{\Pi}_p \mathbf{\Lambda}_p.$$

- Dividing each column of $\bar{\mathbf{A}}_p$ by the element of maximum modulus in that column, denoting the resulting $2 \times F$ matrix $\hat{\mathbf{A}}_p$,

$$\hat{\mathbf{A}}_p = \bar{\mathbf{U}}^T \mathbf{A} \mathbf{\Lambda} \mathbf{\Pi}_p.$$

- $\mathbf{\Lambda}$ does not affect the ratio of elements in each 2×1 column. If ratios are distinct, then permutations can be matched by sorting the ratios of the two coordinates of each 2×1 column of $\hat{\mathbf{A}}_p$.

PARACOMP

- In practice using a few more 'anchor' rows will improve perm-matching.
- When S anchor rows are used, the opt permutation matching cast as

$$\min_{\mathbf{\Pi}} ||\hat{\mathbf{A}}_1 - \hat{\mathbf{A}}_p \mathbf{\Pi}||_F^2,$$

- Optimization over set of permutation matrices - hard?

$$||\hat{\mathbf{A}}_1 - \hat{\mathbf{A}}_p \mathbf{\Pi}||_F^2 = \mathrm{Tr}\left((\hat{\mathbf{A}}_1 - \hat{\mathbf{A}}_p \mathbf{\Pi})^T (\hat{\mathbf{A}}_1 - \hat{\mathbf{A}}_p \mathbf{\Pi}) \right) =$$

$$||\hat{\mathbf{A}}_1||_F^2 + ||\hat{\mathbf{A}}_p \mathbf{\Pi}||_F^2 - 2\mathrm{Tr}(\hat{\mathbf{A}}_1^T \hat{\mathbf{A}}_p \mathbf{\Pi}) =$$

$$||\hat{\mathbf{A}}_1||_F^2 + ||\hat{\mathbf{A}}_p||_F^2 - 2\mathrm{Tr}(\hat{\mathbf{A}}_1^T \hat{\mathbf{A}}_p \mathbf{\Pi}).$$

$$\iff \max_{\mathbf{\Pi}} \mathrm{Tr}(\hat{\mathbf{A}}_1^T \hat{\mathbf{A}}_p \mathbf{\Pi}),$$

- *Linear Assignment Problem* (LAP), efficient soln via *Hungarian Algorithm*.

PARACOMP

- After perm-matching, back to (1) and permute columns $\rightarrow$ $\breve{\mathbf{A}}_p$ satisfying

$$\breve{\mathbf{A}}_p = \mathbf{U}_p^T \mathbf{A} \mathbf{\Pi} \mathbf{\Lambda}_p.$$

- Remains to get rid of $\mathbf{\Lambda}_p$. For this, we can again resort to the first two common rows, and divide each column of $\breve{\mathbf{A}}_p$ with its top element $\rightarrow$

$$\breve{\mathbf{A}}_p = \mathbf{U}_p^T \mathbf{A} \mathbf{\Pi} \mathbf{\Lambda}.$$

For recovery of $\mathbf{A}$ up to perm-scaling of cols, we then require that

$$\begin{bmatrix} \breve{\mathbf{A}}_1 \\ \vdots \\ \breve{\mathbf{A}}_P \end{bmatrix} = \begin{bmatrix} \mathbf{U}_1^T \\ \vdots \\ \mathbf{U}_P^T \end{bmatrix} \mathbf{A} \mathbf{\Pi} \mathbf{\Lambda} \tag{2}$$

be full column rank.

PARACOMP

- If compression ratios in different modes are similar, makes sense to use longest mode for anchoring; if this is the last mode, then

$$P \geq \max\left(\frac{I}{L}, \frac{J}{M}, \frac{K-2}{N-2}\right)$$

- **Theorem:** Assume that $F \leq I \leq J \leq K$, and $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ are full column rank (F). Further assume that $L_p = L$, $M_p = M$, $N_p = N$, $\forall p \in \{1, \cdots, P\}$, $L \leq M \leq N$, $(L+1)(M+1) \geq 16F$, random $\{\mathbf{U}_p\}_{p=1}^P$, $\{\mathbf{V}_p\}_{p=1}^P$, each $\mathbf{W}_p$ contains two common anchor columns, otherwise random $\{\mathbf{W}_p\}_{p=1}^P$.
- Then $\left(\tilde{\mathbf{A}}_p, \tilde{\mathbf{B}}_p, \tilde{\mathbf{C}}_p\right)$ unique up to column permutation and scaling.
- If, in addition, $P \geq \max\left(\frac{I}{L}, \frac{J}{M}, \frac{K-2}{N-2}\right)$, then $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ are almost surely identifiable from $\left\{\left(\tilde{\mathbf{A}}_p, \tilde{\mathbf{B}}_p, \tilde{\mathbf{C}}_p\right)\right\}_{p=1}^P$ up to a common column permutation and scaling.

PARACOMP - Significance

- Indicative of a family of results that can be derived.
- Theorem shows that fully parallel computation of the big tensor decomposition is possible – first result that *guarantees* ID of the big tensor decomposition from the small tensor decompositions, without stringent additional constraints.
- **Corollary:** If $\frac{K-2}{N-2} = \max\left(\frac{I}{L}, \frac{J}{M}, \frac{K-2}{N-2}\right)$, then the memory / storage and computational complexity savings afforded by PARACOMP relative to brute-force computation are of order $\frac{IJ}{F}$.
- Note on complexity of solving master join equation: after removing redundant rows, system matrix in (2) will have approximately orthogonal columns for large $I \rightarrow$ left pseudo-inverse $\approx$ its transpose, complexity I^2F.

Color of compressed noise

- $\underline{\mathbf{Y}} = \underline{\mathbf{X}} + \underline{\mathbf{Z}}$, where $\underline{\mathbf{Z}}$: zero-mean additive white noise.
- $\mathbf{y} = \mathbf{x} + \mathbf{z}$, with $\mathbf{y} := \text{vec}\left(\underline{\mathbf{Y}}\right)$, $\mathbf{x} := \text{vec}\left(\underline{\mathbf{X}}\right)$, $\mathbf{z} := \text{vec}\left(\underline{\mathbf{Z}}\right)$.
- Multi-way compression $\rightarrow \underline{\mathbf{Y}}_c$

$$\mathbf{y}_c := \text{vec}\left(\underline{\mathbf{Y}}_c\right) = \left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)\mathbf{y} =$$

$$\left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)\mathbf{x} + \left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)\mathbf{z}.$$

- Let $\mathbf{z}_c := \left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)\mathbf{z}$. Clearly, $E[\mathbf{z}_c] = 0$; it can be shown that

$$E\left[\mathbf{z}_c\mathbf{z}_c^T\right] = \sigma^2\left(\left(\mathbf{U}^T\mathbf{U}\right) \otimes \left(\mathbf{V}^T\mathbf{V}\right) \otimes \left(\mathbf{W}^T\mathbf{W}\right)\right).$$

- $\Rightarrow$ If **U**, **V**, **W** are orthonormal, then noise in the compressed domain is white.

‘Universal’ LS

- $E[\mathbf{z}_c] = 0$, and

$$E\left[\mathbf{z}_c\mathbf{z}_c^T\right] = \sigma^2\left(\left(\mathbf{U}^T\mathbf{U}\right)\otimes\left(\mathbf{V}^T\mathbf{V}\right)\otimes\left(\mathbf{W}^T\mathbf{W}\right)\right).$$

- For large I and $\mathbf{U}$ drawn from a zero-mean unit-variance uncorrelated distribution, $\mathbf{U}^T\mathbf{U} \approx \mathbf{I}$ by the law of large numbers.
- Furthermore, even if $\mathbf{z}$ is not Gaussian, $\mathbf{z}_c$ will be approximately Gaussian for large IJK, by the Central Limit Theorem.
- Follows that least-squares fitting is approximately optimal in the compressed domain, even if it is not so in the uncompressed domain. Compression thus makes least-squares fitting ‘universal’!

Component energy $\approx$ preserved after compression

- Consider randomly compressing a rank-one tensor $\underline{\mathbf{X}} = \mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$, written in vectorized form as $\mathbf{x} = \mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c}$.
- The compressed tensor is $\underline{\tilde{\mathbf{X}}}$, in vectorized form

$$\tilde{\mathbf{x}} = \left(\mathbf{U}^T \otimes \mathbf{V}^T \otimes \mathbf{W}^T\right)(\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c}) = (\mathbf{U}^T\mathbf{a}) \otimes (\mathbf{V}^T\mathbf{b}) \otimes (\mathbf{W}^T\mathbf{c}).$$

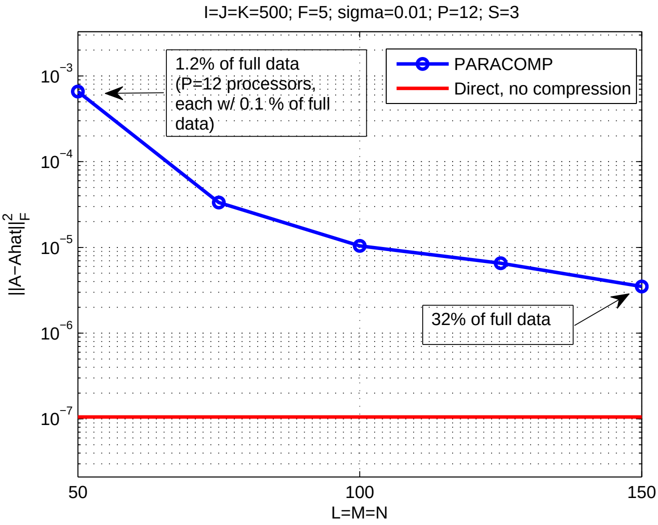
- Can be shown that, for moderate L, M, N and beyond, Frobenious norm of compressed rank-one tensor approximately proportional to Frobenious norm of the uncompressed rank-one tensor component of original tensor.
- In other words: compression approximately preserves component energy $\Rightarrow$ order.
- $\Rightarrow$ Low-rank least-squares approximation of the compressed tensor $\leftrightarrows$ low-rank least-squares approximation of the big tensor, approximately.
- $\Rightarrow$ Can match component permutations across replicas by sorting component energies.

PARACOMP: Numerical results

- Nominal setup:
  - $I = J = K = 500$; $F = 5$; **A**, **B**, **C** $\sim$ `randn(500,5);`
  - $L = M = N = 50$ (each replica = 0.1% of big tensor);
  - $P = 12$ replicas (overall cloud storage = 1.2% of big tensor).
  - $S = 3$ (vs. $S_{min} = 2$) anchor rows.
  - $\uparrow$ Satisfy identifiability without much 'slack'.
  - + WGN std $\sigma = 0.01$.
  - COMFAC `www.ece.umn.edu/~nikos` used for all factorizations, big and small

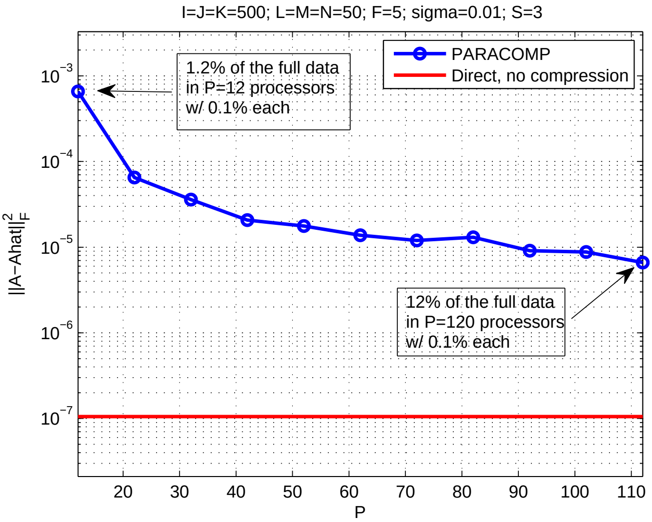
PARACOMP: MSE as a function of $L = M = N$

- Fix $P = 12$, vary $L = M = N$.



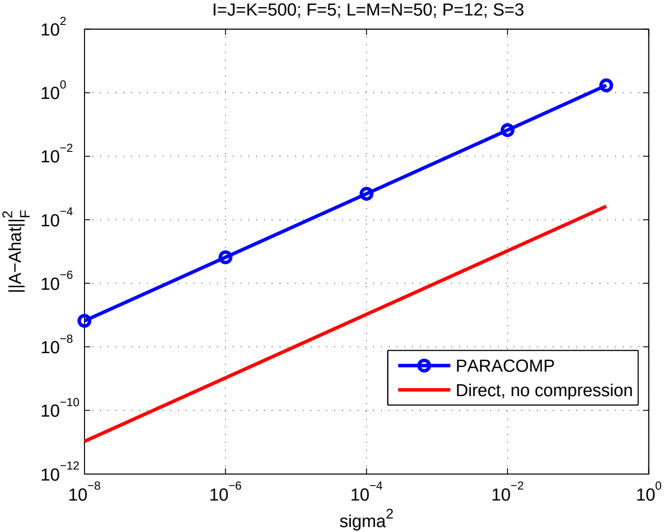
PARACOMP: MSE as a function of P

- Fix $L = M = N = 50$, vary P.



PARACOMP: MSE vs AWGN variance σ^2

- Fix $L = M = N = 50$, $P = 12$, vary σ^2.



Missing elements

- Recommender systems, NELL, many other datasets: over 90% of the values are missing!
- PARACOMP to the rescue: fortuitous fringe benefit of 'compression' (rather: taking linear combinations)!
- Let $\mathcal{T}$ denote the set of all elements, and Ψ the set of available elements.
- Consider one element of the compressed tensor, as it would have been computed had all elements been available; and as it can be computed from the available elements (notice normalization - important!):

$$Y_\nu(l,m,n) = \frac{1}{|\mathcal{T}|} \sum_{(i,j,k)\in\mathcal{T}} \mathbf{u}_l(i)\mathbf{v}_m(j)\mathbf{w}_n(k)\underline{\mathbf{X}}(i,j,k)$$

$$\widetilde{Y}_\nu(l,m,n) = \frac{1}{E[|\Psi|]} \sum_{(i,j,k)\in\Psi} \mathbf{u}_l(i)\mathbf{v}_m(j)\mathbf{w}_n(k)\underline{\mathbf{X}}(i,j,k)$$

Missing elements

- **Theorem:** [Marcos & Sidiropoulos, IEEE ISCCSP 2014] Assume a Bernoulli i.i.d. miss model, with parameter $\rho = Prob[(i,j,k) \in \Psi]$, and let $\underline{\mathbf{X}}(i,j,k) = \sum_{f=1}^{F} \mathbf{a}_f(i)\mathbf{b}_f(j)\mathbf{c}_f(k)$, where the elements of $\mathbf{a}_f$, $\mathbf{b}_f$ and $\mathbf{c}_f$ are all i.i.d. random variables drawn from $\mathbf{a}_f(i) \sim \mathcal{P}_a(\mu_a, \sigma_a)$, $\mathbf{b}_f(j) \sim \mathcal{P}_b(\mu_b, \sigma_b)$, and $\mathbf{c}_f(k) \sim \mathcal{P}_c(\mu_c, \sigma_c)$, with $p_a := \mu_a^2 + \sigma_a^2$, $p_b := \mu_b^2 + \sigma_b^2$, $p_c := \mu_c^2 + \sigma_c^2$, and $F' := (F-1)$. Then, for μ_a, μ_b, μ_c all $\neq 0$,

$$\frac{E[\|\mathcal{E}_\nu\|_F^2]}{E[\|Y_\nu\|_F^2]} \leq \frac{(1-\rho)}{\rho|\mathcal{T}|}(1+\frac{\sigma_U^2}{\mu_u^2})(1+\frac{\sigma_V^2}{\mu_v^2})(1+\frac{\sigma_W^2}{\mu_w^2})(\frac{F'}{F}+\frac{p_a p_b p_c}{F\mu_a^2\mu_b^2\mu_c^2})$$

- Additional results in paper.

Missing elements

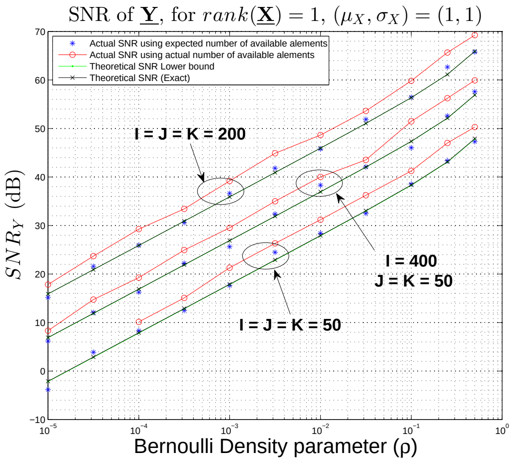


Figure : SNR of compressed tensor for different sizes of rank-one $\underline{\mathbf{X}}$

Missing elements

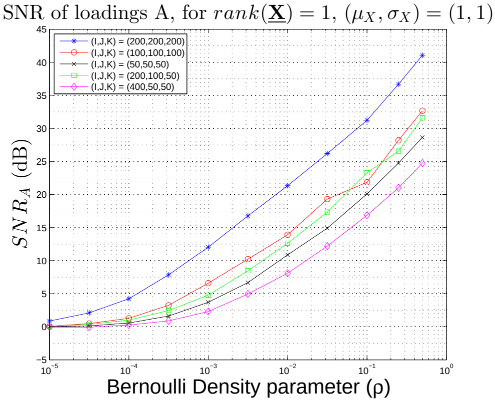


Figure : SNR of recovered loadings for different sizes of rank-one $\underline{\mathbf{X}}$

Missing elements

- Three-way (emission, excitation, sample) fluorescence spectroscopy

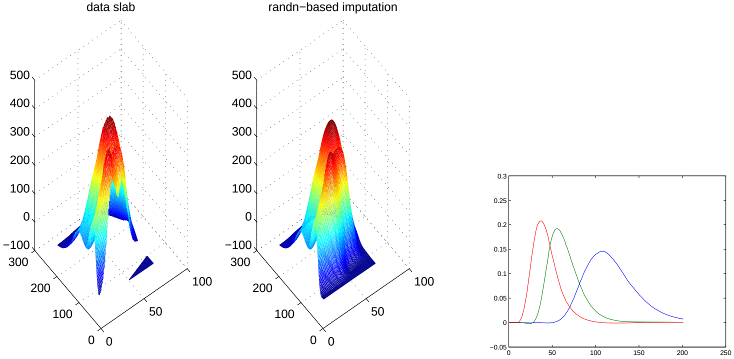


Figure : Measured and imputed data; recovered latent spectra

- Works even with systematically missing data!

Constrained Tensor Factorization & High-Performance Computing

- Constraints (e.g., non-negativity, sparsity) slow down things, cumbersome conditional updates, cannot take advantage of HPC infrastructure.
- New! A.P. Liavas and N.D. Sidiropoulos, “Parallel Algorithms for Constrained Tensor Factorization via the Alternating Direction Method of Multipliers,” *IEEE Trans. on Signal Processing*, submitted.
- Key advantages:
  1. Much smaller complexity/iteration: avoids solving constrained optimization problems, uses simple projections instead.
  2. Competitive with state-of-art for ‘small’ data problems (n-way toolbox / non-negative PARAFAC-ALS), especially for small ranks.
  3. Naturally amenable to parallel implementation on HPC (e.g., mesh) architectures for big tensor decomposition.
  4. Can more-or-less easily incorporate many other types of constraints.